Electronic Colloquium on Computational Complexity, Report No. 215 (2025)

Synergies Between Complexity Theory and Nondeterministic Kolmogorov Complexity

Halley Goldberg[*] Jinqiao Hu[†] Zhenjian Lu[‡] Jingyi Lyu[§] Igor C. Oliveira[¶]

November 24, 2025

Abstract

We investigate central questions in complexity theory through the lens of time-bounded Kolmogorov complexity, focusing on *nondeterministic* measures [AKRR03] and their extensions. In more detail, we consider succinct encodings of a string by programs that may be nondeterministic (nK), randomized (rK), or combine both resources – yielding richer notions such as rnK, pnK, and their generalizations to higher levels of the polynomial hierarchy. Among other contributions, we obtain the following results:

1. Unconditional Complexity Lower Bounds. Let $\mathcal{R}_{\mathsf{nKt}[s(n)]}$ denote the set of strings x such that $\mathsf{nKt}(x) \geq s(|x|)$. We show that $\mathcal{R}_{\mathsf{nKt}[\gamma\cdot n]} \notin \mathsf{NTIME}[2^{o(n)}] \cap \mathsf{coNTIME}[2^{o(n)}]$, for every constant $1/2 < \gamma < 1$. For the problem of estimating rnKt complexity, the randomized extension of nKt, we establish lower bounds against $\mathsf{AMTIME}[n^{\mathsf{polylog}(n)}]$ and $\mathsf{SIZE}[n^{\mathsf{polylog}(n)}]$. These results can be seen as a significant strengthening of the 15-year old lower bound $\mathcal{R}_{\mathsf{nKt}[\Omega(n)]} \notin \mathsf{NP} \cap \mathsf{coNP}$ obtained in [All10, AKRR11], and are among the strongest known *unconditional* lower bounds in meta-complexity.

2. Proof Complexity and Symmetry of Information. [HIL+23, HLO24] showed that cryptographic one-way functions exist if and only if the symmetry of information (SoI) principle *fails on average* for pKt complexity. In contrast, we establish that SoI *holds on average* in the presence of nondeterminism, i.e., $\mathsf{pnKt}(x, y) \approx \mathsf{pnKt}(y) + \mathsf{pnKt}(x \mid y)$ with high probability if (x, y) is generated by a polynomial-time sampler. On the other hand, we prove that if SoI *holds in the worst case* for pnKt complexity then $\mathsf{coNP} \not\subseteq \mathsf{AM}$. In particular, bridging the gap between average-case and worst-case SoI for pnKt would imply that certain coNP statements do not admit feasible proofs.

3. Average-Case vs Worst-Case Complexity. We consider the longstanding problem of relating the average-case and worst-case complexities of the polynomial hierarchy (PH), i.e., showing that $\mathsf{DistPH} \subseteq \mathsf{AvgBPP} \implies \mathsf{PH} \subseteq \mathsf{BPP}$. Building on [Hir20, HLO25], we prove that this implication holds if and only if $\mathsf{Gap\text{-}MINpKT}^{\mathsf{PH}} \in \mathsf{prBPP} \implies \mathsf{Mild\text{-}Gap\text{-}MINpKT}^{\mathsf{PH}} \in \mathsf{prBPP}$, i.e., the complexities of approximating pK^t with an oracle to PH with respect to different approximation regimes are related. Consequently, progress on the meta-complexity of nondeterministic variants of Kolmogorov complexity is *unavoidable* in order to understand the average-case complexity of the polynomial hierarchy.

These results reveal deep links between *complexity theory* and *nondeterministic Kolmogorov complexity*, pointing to several directions for further research.

[*] Simon Fraser University. E-mail: halley_goldberg@sfu.ca
[†] University of Warwick. E-mail: jinqiao.hu@warwick.ac.uk
[‡] University of Victoria. E-mail: zhenjianlu@uvic.ca
[§] Tsinghua University. E-mail: lv-jy22@mails.tsinghua.edu.cn
[¶] University of Warwick. E-mail: igor.oliveira@warwick.ac.uk

ISSN 1433-8092

Contents

1 Introduction

1.1 Overview

Over the last decade, several fundamental problems from algorithms and complexity theory have been investigated through the lens of (time-bounded) Kolmogorov complexity and meta-complexity. Significant and influential advances have occurred in areas such as learning theory [CIKK16], cryptography [LP20], and average-case complexity [Hir21]. In some cases, such as [Hir21] and subsequent work [GKLO22], although the result itself is not about Kolmogorov complexity, the only known proof technique uses Kolmogorov complexity in an essential way. We refer to the surveys [LO22, Hir22a, MP25] for more context and background.

These and other developments have led to a renewed interest in time-bounded Kolmogorov complexity, meta-complexity, and their applications in theoretical computer science (see, e.g., [Sim23]). An important perspective explored in many results, specially in the last few years, has been the use of extensions of the classical theory of time-bounded Kolmogorov complexity to more powerful computational settings, such as *probabilistic Kolmogorov complexity* [GKLO22, LO22]. Indeed, many if not most results in the area from the last few years employ probabilistic variants of Kolmogorov complexity (see, e.g., [Ila23, San23, HIL$^+$23, HN23, LP23, Hir23, GK23, HLO24, HLN24, LS24, HKLO24, GK24, LORS24, HN25, KK25, LP25]).

In this work, we advance this approach by further developing the theory of *nondeterministic Kolmogorov complexity*. Building on and extending previous work [AKRR03, BLvM05, All10, AKRR11, HLO25], we demonstrate its relevance to several central problems in complexity theory. Specifically, our contributions span three fundamental areas: unconditional complexity lower bounds, NP versus coNP and related separations, and the average-case complexity of the polynomial hierarchy. In each of these, we show that nondeterministic Kolmogorov complexity and its extensions offer a productive – and in some contexts, unavoidable – perspective on long-standing challenges in complexity theory.

Next, we explain our results in detail.

1.2 Results

1.2.1 Unconditional Complexity Lower Bounds for Estimating Complexity

We will consider certain natural extensions of Levin's Kt complexity [Lev84] to the setting of nondeterministic computations [AKRR11]. As usual, in order to define time-bounded notions of Kolmogorov complexity, we fix a time-efficient universal machine U, and consider programs Π encoded with respect to U (see Section 2).

For a string $x \in \{0,1\}^*$, the *nondeterministic time-bounded Kolmogorov complexity of* x is defined as

$$\mathsf{nKt}(x) \triangleq \min_{\Pi \in \{0,1\}^*,\, t \in \mathbb{N}} \left\{ |\Pi| + \lceil \log t \rceil \;\middle|\; \begin{array}{l} \forall w \in \{0,1\}^t, \Pi(w) \text{ outputs } x \text{ or } \perp \text{ in } t \text{ steps;} \\ \exists w \in \{0,1\}^t, \Pi(w) \text{ outputs } x \text{ in } t \text{ steps} \end{array} \right\}.$$

In other words, by viewing w as a nondeterministic string, in at least one computational path, the program Π outputs x, while in every computational path, Π outputs either x or the failure symbol $\perp$. For this reason, the program Π uniquely specifies x. In addition, x can be recovered from Π in time at most t using nondeterminism.

Lower bounds for nKt. Let MnKtP be the following problem: Given $(x, 1^s)$, where $x \in \{0,1\}^*$ and $s \in \mathbb{N}$, decide whether $\mathsf{nKt}(x) \leq s$ ("yes" case) or $\mathsf{nKt}(x) > s$ ("no" case). Moreover, for $s\colon \mathbb{N} \to \mathbb{N}$, we let $\mathcal{R}_{\mathsf{nKt}[s]} = \{x \in \{0,1\}^* \mid \mathsf{nKt}(x) \geq s(|x|)\}$. We say that a language L is *dense* if there is a positive constant b such that, for every large enough n, we have $|L \cap \{0,1\}^n| \geq 2^n/n^b$.

Theorem 1.1 (Complexity Lower Bounds for Estimating nKt)**.** *There is a constant $\varepsilon > 0$ such that*

$$\mathsf{MnKtP} \notin \mathsf{NTIME}[2^{\varepsilon n}] \cap \mathsf{coNTIME}[2^{\varepsilon n}].$$

Moreover, given a constant $1/2 < \gamma < 1$, there exists $\varepsilon = \varepsilon(\gamma) > 0$ such that, for every dense language $L \subseteq \mathcal{R}_{\mathsf{nKt}[\gamma \cdot n]}$, we have $L \notin \mathsf{NTIME}[2^{\varepsilon n}] \cap \mathsf{coNTIME}[2^{\varepsilon n}]$.

It was previously known that $\mathsf{MnKtP} \notin \mathsf{NP} \cap \mathsf{coNP}$ [All10, AKRR11]. In contrast, Theorem 1.1 provides an exponential lower bound on the complexity of MnKtP. More generally, it shows that distinguishing strings of bounded complexity ("structure") from random strings ("randomness") is hard.

We observe that, in order to obtain a lower bound of the form $2^{\Omega(n)}$ for $\mathcal{R}_{\mathsf{nKt}[\gamma \cdot n]}$, our techniques require assuming $\gamma > 1/2$; see Section 4.1.5 for details. Establishing lower bounds of magnitude $2^{\Omega(n)}$ for $\mathcal{R}_{\mathsf{nKt}[\gamma \cdot n]}$ when $0 < \gamma \leq 1/2$ remains an open problem.

Since $\mathsf{MnKtP} \in \mathsf{P}^{\mathsf{NEXP}} \subseteq \mathsf{NEXP}/\mathsf{poly}$ and every language in NEXP non-uniformly reduces to MnKtP [AKRR11], it follows that $\mathsf{MnKtP} \notin \mathsf{SIZE}[\mathsf{poly}]$ if and only if $\mathsf{NEXP} \not\subseteq \mathsf{SIZE}[\mathsf{poly}]$. In particular, a breakthrough complexity separation would follow if we could show that $\mathsf{MnKtP} \notin \mathsf{SIZE}[\mathsf{poly}]$. While we currently don't know how to prove the required lower bound for MnKtP, we are able to establish such a lower bound for the related problem of estimating rnKt, the *randomized extension* of nKt.

For a string $x \in \{0,1\}^*$, the *randomized nondeterministic time-bounded Kolmogorov complexity of x* is defined as

$$\mathsf{rnKt}(x) \triangleq \min_{\Pi \in \{0,1\}^*,\, t \in \mathbb{N}} \left\{ |\Pi| + \lceil \log t \rceil \;\middle|\; \mathop{\mathbf{Pr}}_{r \sim \{0,1\}^t} \begin{bmatrix} \forall w \in \{0,1\}^t, \Pi(w;r) \text{ outputs } x \text{ or } \perp \text{ in } t \text{ steps;} \\ \exists w \in \{0,1\}^t, \Pi(w;r) \text{ outputs } x \text{ in } t \text{ steps} \end{bmatrix} \geq \frac{2}{3} \right\}.$$

Equivalently, there exists a nondeterministic program $\Pi(r)$ such that, with probability at least $2/3$ over r, $\Pi(r)$ uniquely specifies x and enables its recovery in nondeterministic time at most t.

Lower bounds for rnKt. For two functions $s_1, s_2 \colon \mathbb{N} \to \mathbb{N}$, let $\mathsf{Gap\text{-}MrnKtP}[s_1, s_2]$ be the following promise problem: Given $x \in \{0,1\}^*$, decide whether $\mathsf{rnKt}(x) \leq s_1(|x|)$ ("yes" case) or $\mathsf{rnKt}(x) > s_2(|x|)$ ("no" case). For a promise problem $\Pi = (\mathsf{YES}, \mathsf{NO})$ and a complexity class $\mathcal{C}$, we say that $\Pi \notin \mathcal{C}$ if there is no language $L \in \mathcal{C}$ such that $\mathsf{YES} \subseteq L$ and $\mathsf{NO} \subseteq \{0,1\}^* \setminus L$. For a language $\mathcal{O}$ and a function $s \colon \mathbb{N} \to \mathbb{N}$, we let $\mathsf{SIZE}^{\mathcal{O}}[s]$ denote the set of languages computed by Boolean circuits with oracle access to $\mathcal{O}$ and of size at most $s(n)$. Similarly to $\mathcal{R}_{\mathsf{nKt}[s]}$, we can also consider the set $\mathcal{R}_{\mathsf{rnKt}[s]}$, which is defined in the natural way using rnKt instead of nKt.

Theorem 1.2 (Complexity Lower Bounds for Estimating rnKt)**.** *The following statements hold*:

1. *For every constant $1/2 < \gamma < 1$, there exists a constant $\varepsilon > 0$ such that*

$$\mathsf{Gap\text{-}MrnKtP}[\gamma \cdot n, n-1] \notin \mathsf{AMTIME}[2^{\varepsilon n}] \cap \mathsf{coAMTIME}[2^{\varepsilon n}].$$

   Moreover, given a constant $1/2 < \gamma < 1$, there exists $\varepsilon(\gamma) > 0$ such that, for every dense language $L \subseteq \mathcal{R}_{\mathsf{rnKt}[\gamma \cdot n]}$, we have $L \notin \mathsf{AMTIME}[2^{\varepsilon n}] \cap \mathsf{coAMTIME}[2^{\varepsilon n}]$.

2. *For every constant $\gamma > 0$,*

$$\mathsf{Gap\text{-}MrnKtP}\left[2^{\log^{\gamma} n}, n-1\right] \notin \mathsf{AMTIME}\left[n^{\mathsf{polylog}(n)}\right].$$

   Moreover, for every dense language $L \subseteq \mathcal{R}_{\mathsf{rnKt}[2^{\log^{\gamma} n}]}$, we have $L \notin \mathsf{coAMTIME}\left[n^{\mathsf{polylog}(n)}\right]$.

3. *For every constant $\gamma > 0$ and $\mathcal{O} \in \mathsf{AM} \cap \mathsf{coAM}$,*

$$\mathsf{Gap\text{-}MrnKtP}\left[2^{\log^\gamma n}, n-1\right] \notin \mathsf{SIZE}^{\mathcal{O}}\left[n^{\mathsf{polylog}(n)}\right].$$

Moreover, for every dense language $L \subseteq \mathcal{R}_{\mathsf{rnKt}[2^{\log^\gamma n}]}$, we have $L \notin \mathsf{SIZE}^{\mathcal{O}}\left[n^{\mathsf{polylog}(n)}\right]$.

To the best of our knowledge, no previous lower bounds against AM, $\mathsf{SIZE}[\mathsf{poly}]$, and exponential time nondeterministic classes were known for natural meta-computational problems. These results thus provide the strongest unconditional lower bounds in meta-complexity to date.

1.2.2 Proof Complexity and Symmetry of Information

Symmetry of Information (SoI) is a fundamental property of (time-unbounded) Kolmogorov complexity [ZL70] with applications in various areas (e.g., [SUV17, LV19]). It states that for any strings $x, y \in \{0,1\}^n$,

$$\mathsf{K}(x,y) \approx \mathsf{K}(y) + \mathsf{K}(x \mid y) \approx \mathsf{K}(x) + \mathsf{K}(y \mid x),$$

up to an additive $O(\log n)$ term. In other words, describing x and y jointly is no shorter, up to logarithmic terms, than describing one optimally and the other given the first. The trivial direction,

$$\mathsf{K}(x,y) \leq \mathsf{K}(x) + \mathsf{K}(y \mid x) + O(\log n),$$

follows easily by construction, while the non-trivial part asserts that

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

While SoI holds for time-unbounded Kolmogorov complexity and fails for Kt complexity [Ron04] (see Appendix A for related discussions), whether this property is valid for various other notions of time-bounded Kolmogorov complexity remains unknown. We refer to [KK25] and references therein for more background on SoI and its connections to complexity theory.

A sequence of papers [HIL+23, HLO24, HLN24] established that the failure of time-bounded versions of SoI (in a certain average-case sense) is equivalent to the existence of cryptographic one-way functions. In other words, the existence of provably secure cryptography is intimately connected to time-bounded SoI.

Here we establish a connection between time-bounded SoI and the NP vs. coNP problem (i.e., whether there is a proof system for which every tautology admits a short proof). In order to discuss our results and contrast them with previous work, we need to introduce *probabilistic* extensions of Kt and nKt, in the sense of [GKLO22] (see also [LO22]).

Probabilistic variants of Kt and nKt. Informally, pKt is simply Kt in the presence of public randomness, i.e., x has "small" pKt complexity if with probability at least $2/3$ over the randomness r, x has "small" Kt complexity given access to r. Formally, for $x \in \{0,1\}^*$, its *probabilistic time-bounded Kolmogorov complexity* [HLO24] is given by

$$\mathsf{pKt}(x) \triangleq \min\left\{ k \in \mathbb{N} \;\middle|\; \mathop{\mathbf{Pr}}_{r \sim \{0,1\}^{2^k}}\left[\mathsf{Kt}(x \mid r) \leq k\right] \geq \frac{2}{3} \right\},$$

where $\mathsf{Kt}(x \mid r)$ denotes the conditional Kt complexity of x given r, i.e., the minimum value $|\Pi| + \lceil \log t \rceil$ over all pairs (Π, t) such that Π is a program that outputs x when given r and runs in at most t steps.

Analogously, for a string $x \in \{0,1\}^*$, its *probabilistic nondeterministic time-bounded Kolmogorov complexity* [HLO24] is defined as

$$\mathsf{pnKt}(x) \triangleq \min\left\{ k \in \mathbb{N} \;\middle|\; \mathop{\mathbf{Pr}}_{r \sim \{0,1\}^{2^k}} \left[\mathsf{nKt}(x \mid r) \le k \right] \ge \frac{2}{3} \right\}.$$

In other words, bounded pnKt complexity means that in the presence of a typical random string r, x has bounded nKt complexity.

In more detail, [HLO24] showed that one-way functions exist if and only if SoI fails on average for pKt, i.e., there is a polynomial-time samplable distribution family $\{\mathcal{D}_n\}$, where each $\mathcal{D}_n$ is supported over $\{0,1\}^n \times \{0,1\}^n$, and a polynomial q such that for every constant $c > 0$ and for every large enough n,

$$\mathop{\mathbf{Pr}}_{(x,y) \sim \mathcal{D}_n} \left[\, \mathsf{pKt}(x,y) < \mathsf{pKt}(x) + \mathsf{pKt}(y \mid x) - c \cdot \log n \,\right] \ge \frac{1}{q(n)}.$$

This means that, under the widely-believed assumption that one-way functions exist, SoI *fails on average* for pKt (and for related notions such as Kt and rKt).

In contrast, adapting previous work [HLO25], we show (unconditionally) that SoI *holds on average* when both randomness and nondeterminism are present.

Theorem 1.3 (Average-Case Symmetry of Information for Nondeterministic Kolmogorov Complexity)**.** *For every polynomial-time samplable distribution family $\{\mathcal{D}_n\}_{n \in \mathbb{N}}$, where each $\mathcal{D}_n$ is supported over $\{0,1\}^n \times \{0,1\}^n$, there exists a constant $c > 0$ such that for all large enough $n, k \in \mathbb{N}$,*

$$\mathop{\mathbf{Pr}}_{(x,y) \sim \mathcal{D}_n} \left[\, \mathsf{pnKt}(x,y) \ge \mathsf{pnKt}(x) + \mathsf{pnKt}(y \mid x) - c \cdot \log n - \log k \,\right] \ge 1 - \frac{1}{k}.$$

Similarly,

$$\mathop{\mathbf{Pr}}_{(x,y) \sim \mathcal{D}_n} \left[\, \mathsf{rnKt}(x,y) \ge \mathsf{rnKt}(x) + \mathsf{rnKt}(y \mid x) - c \cdot \log^3 n - \log k \,\right] \ge 1 - \frac{1}{k}.$$

It is unclear how to extend the proof of Theorem 1.3 to the *worst case*, i.e., to prove that the relevant inequality holds for all strings $x, y \in \{0,1\}^n$. Interestingly, we are able to show that this would separate coNP from NP in a strong sense, as stated below. Let $\mathsf{TAUT} \subseteq \{0,1\}^*$ denote the set of tautologies, i.e., Boolean formulas F that always evaluate to 1.

Theorem 1.4. *Item 1 implies Item 2 in the following.*

1. (Worst-Case Symmetry of Information for Nondeterministic Kolmogorov Complexity). *There exists a constant $c > 0$ such that for all large enough $n \in \mathbb{N}$ and $x, y \in \{0,1\}^n$,*

   $$\mathsf{pnKt}(x,y) \ge \mathsf{pnKt}(x) + \mathsf{pnKt}(y \mid x) - e,$$

   where $e = \log^c n$ (resp. $e = c \cdot \log n$).

2. (Hardness of Proving Tautologies). $\mathsf{TAUT} \notin \mathsf{AMTIME}[t(n)]$*, where $t(n) = 2^{\log^d n}$, for any $d \ge 1$* $\big($resp. $t(n) = 2^{n^\varepsilon}$ *for some $\varepsilon \ge 0$*$\big)$.

The above implication remains valid if pnKt *is replaced with* rnKt *in the first item.*

While several results establish that lower bounds follow from the *failure* of SoI (see, e.g., [LW95, Hir22c, HIL+23, HLO24, HLN24]), an interesting aspect of Theorem 1.4 is that here a lower bound follows from the *validity* of SoI in a time-bounded setting. In a sense, Theorem 1.4 reduces the *lower bound* problem of separating coNP from AM to an *upper bound* task: showing that for every large enough n and $x, y \in \{0,1\}^n$, we have

$$\mathsf{pnKt}(y \mid x) \leq \mathsf{pnKt}(x, y) - \mathsf{pnKt}(x) + \mathsf{poly}(\log n). \tag{$*$}$$

It is worth mentioning that SoI has been successfully employed as an approach to obtain unconditional complexity lower bounds [Hir22c]. SoI also plays an important role in the proof of some results from Section 1.2.1, as explained in Section 1.3.

In Section 5.4, we show a general result implying that any proof of Equation ($*$) must employ non-relativizing techniques. On the positive side, we note that, in order to show the weaker separation $\mathsf{NP} \nsubseteq \mathsf{BPP}$, it is enough to establish Equation ($*$) under the assumption that $\mathsf{NP} \subseteq \mathsf{BPP}$ (since $\mathsf{TAUT} \notin \mathsf{AM}$ implies in particular that $\mathsf{NP} \nsubseteq \mathsf{BPP}$). Consequently, as an approach to lower bounds, symmetry of information might be useful even if Equation ($*$) turns out to be false. Interestingly, it is known that $\mathsf{NP} \subseteq \mathsf{BPP}$ implies that symmetry of information holds for other time-bounded Kolmogorov complexity measures [Hir22c]. By a similar argument, to show $\mathsf{NEXP} \nsubseteq \mathsf{BPP}$, it suffices to prove that $\mathsf{NEXP} \subseteq \mathsf{BPP} \Longrightarrow$ Equation ($*$). This might be a fruitful research direction.

1.2.3 Average-Case vs. Worst-Case Complexity in the Polynomial Hierarchy

Understanding the relation between the worst-case and average-case complexities of NP is a central open problem in computational complexity theory. In particular, resolving this question is a crucial milestone for the goal of constructing cryptographic schemes grounded in the worst-case intractability of NP (see, e.g., [BT06]).

Formally, we would like to show that if NP is easy on average (i.e., $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$) then NP is easy in the worst case (i.e., $\mathsf{NP} \subseteq \mathsf{BPP}$). In order to establish this implication, it is necessary to first establish a similar result for problems in the polynomial hierarchy (PH), i.e., to prove that if $\mathsf{DistPH} \subseteq \mathsf{AvgBPP}$ then $\mathsf{PH} \subseteq \mathsf{BPP}$.[1] In the terminology of Impagliazzo's five possible complexity worlds [Imp95], this corresponds to excluding PH-Heuristica, that is, excluding the possibility that we live in a computational world where every problem in PH is easy on average but some problems in PH are hard in the worst case.

We show that excluding PH-Heuristica is *equivalent* to a problem about the meta-complexity of nondeterministic Kolmogorov complexity. In order to explain this result, we need to introduce some definitions.

$\mathsf{Gap\text{-}MINpKT}^{\mathsf{PH}}$ and $\mathsf{Mild\text{-}Gap\text{-}MINpKT}^{\mathsf{PH}}$. First, we observe that, up to a constant factor, $\mathsf{nKt}(x)$ is simply $\mathsf{Kt}^{||\mathsf{SAT}}(x)$, i.e., the Kt complexity of x when we consider programs that can make *non-adaptive* queries to a SAT oracle (see Section 3.1 for the details). More generally, we will consider time-bounded programs that can make *adaptive* queries to problems in the polynomial hierarchy.

For a string $x \in \{0,1\}^n$, a time bound t, a parameter $\lambda \in [0,1]$, and an oracle $\mathcal{O} \subseteq \{0,1\}^*$, the *probabilistic t-time bounded Kolmogorov complexity* of x is defines as

$$\mathsf{pK}_{\lambda}^{t,\mathcal{O}}(x) \triangleq \min\left\{ s \in \mathbb{N} \;\middle|\; \mathop{\mathbf{Pr}}_{r \sim \{0,1\}^t}\left[\mathsf{K}^{t,\mathcal{O}}(x \mid r) \leq s\right] \geq \lambda \right\},$$

where $\mathsf{K}^{t,\mathcal{O}}(x \mid r)$ denotes the conditional t-time bounded Kolmogorov complexity of x with oracle access to $\mathcal{O}$. We often omit the parameter λ when $\lambda = 2/3$.

[1] In order to see this, it is enough to observe that $\mathsf{NP} \subseteq \mathsf{BPP}$ if and only if $\mathsf{PH} \subseteq \mathsf{BPP}$.

For a function $\rho\colon \mathbb{N} \to \mathbb{N}$ and an oracle $\mathcal{O}$, let $\mathsf{Gap}_\rho\text{-}\mathsf{MINpKT}^{\mathcal{O}}$ be the following promise problem: Given $(x, 1^s, 1^t)$, where $x \in \{0,1\}^*$ and $s, t \in \mathbb{N}$, decide whether $\mathsf{pK}^{t,\mathcal{O}}(x) \leq s$ ("yes" case) or $\mathsf{pK}^{\rho(t),\mathcal{O}}(x) > s + \log \rho(t)$ ("no" case).

Let $\mathcal{O}$ be an oracle. We say that $\mathsf{Gap}\text{-}\mathsf{MINpKT}^{\mathcal{O}} \in \mathsf{prBPP}$ if there exist a polynomial ρ and a probabilistic polynomial-time algorithm that solves $\mathsf{Gap}_\rho\text{-}\mathsf{MINpKT}^{\mathcal{O}}$. Also, we say that $\mathsf{Gap}\text{-}\mathsf{MINpKT}^{\mathsf{PH}} \in \mathsf{prBPP}$ if for every $\mathcal{O} \in \mathsf{PH}$, $\mathsf{Gap}\text{-}\mathsf{MINpKT}^{\mathcal{O}} \in \mathsf{prBPP}$.

We also consider the problem $\mathsf{Mild}\text{-}\mathsf{Gap}\text{-}\mathsf{MINpKT}^{\mathcal{O}}$, which is similar to $\mathsf{Gap}\text{-}\mathsf{MINpKT}^{\mathcal{O}}$ except for that in the no case, we have $\mathsf{pK}^{t+\rho(|x|),\mathcal{O}}(x) > s + \log \rho(t)$ instead of $\mathsf{pK}^{\rho(t),\mathcal{O}}(x) > s + \log \rho(t)$. Thus the key difference between these promise problems is the gap in the time bounds, which are of the form t versus $\mathsf{poly}(t)$ and t versus $t + \mathsf{poly}(n)$, respectively.[2]

We are now ready to state our result.

Theorem 1.5. *The following statements are equivalent*:

1. (Exclusion of PH-Heuristica). $\mathsf{DistPH} \subseteq \mathsf{AvgBPP} \implies \mathsf{PH} \subseteq \mathsf{BPP}$.
2. (Meta-Complexity of $\mathsf{pK}^{\mathsf{PH}}$). $\mathsf{Gap}\text{-}\mathsf{MINpKT}^{\mathsf{PH}} \in \mathsf{prBPP} \implies \mathsf{Mild}\text{-}\mathsf{Gap}\text{-}\mathsf{MINpKT}^{\mathsf{PH}} \in \mathsf{prBPP}$.

We can also obtain a deterministic analogue of the above result by considering the problem of estimating $\mathsf{K}^{t,\mathsf{PH}}$ instead of $\mathsf{pK}^{t,\mathsf{PH}}$ (see Theorem 6.21).

Consequently, the average-case vs. worst-case complexity of PH is intrinsically connected to the relative complexity of estimating the PH-oracle time-bounded Kolmogorov complexity in two different approximation regimes.

As alluded to in Section 1.1, [Hir21, GKLO22] established using techniques from time-bounded Kolmogorov complexity and meta-complexity that if $\mathsf{DistPH} \subseteq \mathsf{AvgBPP}$ then $\mathsf{PH} \subseteq \mathsf{BPTIME}[2^{O(n/\log n)}]$ (a weaker form of Item 1 in Theorem 1.5). Moreover, this is the only known proof of this result. Theorem 1.5 sheds light into this by showing that understanding the complexity of computing time-bounded Kolmogorov complexity is unavoidable. For this reason, we believe that this result is of particular conceptual interest.

It follows from the proof of Theorem 1.5 and previous work [Imp11, HN21] that any efficient reduction from $\mathsf{Mild}\text{-}\mathsf{Gap}\text{-}\mathsf{MINpKT}^{\mathsf{PH}}$ to $\mathsf{Gap}\text{-}\mathsf{MINpKT}^{\mathsf{PH}}$ must use non-relativizing techniques. In other words, there is an oracle world where $\mathsf{Gap}\text{-}\mathsf{MINpKT}^{\mathsf{PH}} \in \mathsf{prBPP}$ while $\mathsf{Mild}\text{-}\mathsf{Gap}\text{-}\mathsf{MINpKT}^{\mathsf{PH}} \notin \mathsf{prBPP}$ (see Section 6.5 for more details). We note that some reductions in meta-complexity employ non-relativizing techniques, such as [Hir22b]. In particular, we see this relativization result as a guiding principle rather than an insurmountable barrier. Indeed, Theorem 1.5 and prior work [Hir21, GKLO22, Hir22b] suggest that ruling out PH-Heuristica using techniques from probabilistic and nondeterministic Kolmogorov complexity is a promising research direction.

1.3 Techniques

We now discuss the key ideas and techniques behind the proof of each result stated in Section 1.2.

Lower bounds for computing nKt (Theorem 1.1). In previous work, Allender et al. obtained the weaker lower bound $\mathsf{MnKtP} \notin \mathsf{NP} \cap \mathsf{coNP}$ [AKRR11]. Firstly, the authors proved that

$$\mathsf{NEXP} \subseteq \mathsf{P}^{\mathsf{MnKtP}}/\mathsf{poly},$$

[2] Formally, we need to allow a small gap in the probability λ (in $\mathsf{pK}^{t,\mathcal{O}}_{\lambda}(x)$) when specifying the "yes" and "no" cases of $\mathsf{Mild}\text{-}\mathsf{Gap}\text{-}\mathsf{MINpKT}^{\mathcal{O}}$. However, for simplicity we avoid this discussion in the exposition given here. We refer to Section 2.2 for the precise definition of each computational problem.

using reconstruction of pseudo-random generators. Assuming $\mathsf{MnKtP} \in \mathsf{NP} \cap \mathsf{coNP}$ and applying a Karp-Lipton style argument, it follows that

$$\mathsf{NEXP} \subseteq \mathsf{PSPACE}.$$

Secondly, they proved

$$\mathsf{PSPACE} \subseteq \mathsf{ZPP}^{\mathsf{MnKtP}},$$

again using reconstruction of PRGs. By assumption, this implies

$$\mathsf{PSPACE} \subseteq \mathsf{ZPP}^{\mathsf{NP} \cap \mathsf{coNP}} \subseteq \mathsf{NP} \cap \mathsf{coNP}.$$

Combining the aforementioned inclusions, $\mathsf{NEXP} \subseteq \mathsf{NP} \cap \mathsf{coNP}$, contradicting the non-deterministic time-hierarchy theorem.

We notice that the proof of [AKRR11] can be extended to a larger time bound, but due to the two separate steps, the approach faces a *half-exponential barrier*. In particular, within the framework of [AKRR11], the strongest lower bound one can hope to prove is $\mathsf{MnKtP} \notin \mathsf{NTIME}[T(n)] \cap \mathsf{coNTIME}[T(n)]$ for a time bound $T(n)$ satisfying $T(T(n)) = o(2^n)$. Without a significant new idea, the approach cannot prove lower bounds with $T(n) = 2^{n^{\varepsilon}}$ for a constant $\varepsilon > 0$, let alone $T(n) = 2^{\varepsilon n}$.

In Theorem 1.1, we overcome the half-exponential barrier and establish a strong lower bound for any dense language that only accepts strings of large nKt complexity. The proof works by a combination of the following elements:

- The SoI lower bound technique from [Hir22c];
- Derandomization using nondeterminism [AKRR11];
- Probability amplification for rK with controlled randomness complexity;
- Iterated construction of a string of large complexity.

Next, we provide more background and explain how these techniques are combined in the proof of Theorem 1.1. We begin by discussing the ingredients in the proof that $\mathsf{MnKtP} \notin \mathsf{NTIME}[2^{\varepsilon n}] \cap \mathsf{coNTIME}[2^{\varepsilon n}]$ (ie. the first part of Theorem 1.1).

Hirahara developed an approach to lower bounds that exploits a connection between Symmetry of Information and the easiness of meta-computational problems [Hir22c]. The approach is inspired by a technique from the Ph.D. thesis of Ronneburger, which excludes SoI for Kt unconditionally [Ron04]. Since we will build on this approach, we give a brief overview below.

Specifically, [Hir22c] shows that $\mathsf{MrKtP} \notin \mathsf{BPTIME}[2^{\varepsilon n}]$ for some constant $\varepsilon > 0$. The proof consists of two steps. First, the author shows that $\mathsf{MrKtP} \in \mathsf{BPTIME}[2^{\varepsilon n}]$ implies SoI for rKt. That is, for any two strings $x, y \in \{0,1\}^n$,

$$\mathsf{rKt}(x,y) \geq \mathsf{rKt}(x \mid y) + \mathsf{rKt}(y) - O(\varepsilon n).$$

The proof of this implication employs reconstruction of pseudo-random generators. Define $R_{\mathsf{rKt}} := \{x \mid \mathsf{rKt}(x) \geq |x|\}$, and use $\mathcal{U}_l$ to denote uniform distribution over $\{0,1\}^l$. Then, roughly speaking, the key intuition is that R_{rKt} cannot distinguish between the following three distributions:

$$\begin{aligned}
\mathcal{D}_1 &:= G_1(x;\mathcal{U}_{d_1}) \circ G_2(y;\mathcal{U}_{d_2}),\\
\mathcal{D}_2 &:= \mathcal{U}_{m_1} \circ G_2(y;\mathcal{U}_{d_2}),\\
\mathcal{D}_3 &:= \mathcal{U}_{m_1} \circ \mathcal{U}_{m_2},
\end{aligned}$$

where $G_i : \{0,1\}^n \times \{0,1\}^{d_i} \to \{0,1\}^{m_i}$ for $i \in \{1,2\}$ are some PRGs based on a construction of Raz et al. [RRV02] (see [Hir20, Section 4.2]), $d_1, d_2 = O(\log^3 n)$, and we leave $m_1, m_2 \leq 2n$ to be specified later (See Lemma 2.27).

If R_{rKt} distinguishes $\mathcal{D}_1$ from $\mathcal{D}_2$ in presence of y, a randomized reconstruction procedure with oracle access to R_{rKt} can produce x given y with non-trivial probability in polynomial time, yielding an upper bound on $\mathsf{rK}^{\mathsf{poly}(n),R_{\mathsf{rKt}}}(x \mid y)$ (See Corollary 4.6). Furthermore, assuming $\mathsf{MrKtP} \in \mathsf{BPTIME}[2^{\varepsilon n}]$ enables us to eliminate the oracle in this $\mathsf{rK}^{\mathsf{poly}(n)}$ upper bound. Specifically, one obtains

$$\mathsf{rKt}(x \mid y) \leq m_1 + O(\varepsilon n).$$

Given the arbitrariness of m_1, we may now pick $m_1 = \mathsf{rKt}(x \mid y) - O(\varepsilon n)$ to contradict the previous inequality, thereby establishing indistinguishability (for R_{rKt}) between $\mathcal{D}_1$ and $\mathcal{D}_2$. Similarly, picking $m_2 = \mathsf{rKt}(y) - O(\varepsilon n)$ establishes indistinguishability between $\mathcal{D}_2$ and $\mathcal{D}_3$. By a hybrid argument, R_{rKt} cannot distinguish $\mathcal{D}_1$ from $\mathcal{D}_3$. Therefore, with non-zero probability over $s \sim \mathcal{D}_1$,

$$\mathsf{rKt}(s) \gtrsim m_1 + m_2 - O(1) = \mathsf{rKt}(x \mid y) + \mathsf{rKt}(y) - O(\varepsilon n).$$

On the other hand, since $d_1, d_2 \leq O(\log^3 n)$, and G_1, G_2 are efficiently computable given x, y, and seeds of length d_1 and d_2, it holds that $\mathsf{rKt}(s) \approx \mathsf{rKt}(x, y)$ for any such s. This gives us SoI for rKt.

The second step of the proof in [Hir22c] follows the technique of [Ron04] (along with the assumption that $\mathsf{MrKtP} \in \mathsf{BPTIME}[2^{\varepsilon n}]$) for showing that SoI does not hold. By a standard counting argument, there exists a string $y \in \{0,1\}^n$ such that $\mathsf{rKt}(y) \geq n$. We may find the lexicographically first such string y^* by brute-force, assuming easiness of MrKtP. There likewise exists a string $x \in \{0,1\}^n$ such that $\mathsf{rKt}(x \mid y^*) \geq n$, and by SoI, for such x, we have $\mathsf{rKt}(x, y^*) \geq 2n - O(\varepsilon n)$. Again, we may find the lexicographically first such string x^* by brute-force. Note that the two brute-force searches to find y^* and x^* take time $2^{n+O(\varepsilon n)}$ overall, with only n as input. This implies that $\mathsf{rKt}(x^*, y^*) \leq n + O(\varepsilon n)$, a contradiction for small enough $\varepsilon > 0$. It follows that MrKtP is hard.

A limitation of the argument from [Hir22c] above, which blocks its direct translation to MnKtP, is the randomness required in the reconstruction procedure of the pseudo-random generator. Randomized complexity measures such as rKt naturally incorporate the presence of randomness *by definition*, but randomness is not allowed in the definition of nKt. Thus, it is not straightforward to give an upper bound on nKt as in Corollary 4.6. (For the same reason, the argument does not generalize to prove MKtP lower bounds, and in fact it remains open whether $\mathsf{MKtP} \notin \mathsf{P}$ unconditionally.)

We get around this limitation in the case of nKt by making use of non-determinism to derandomize reconstruction. As a first attempt, we notice that if $\mathsf{MnKtP} \in \mathsf{NTIME}[2^{\varepsilon n}] \cap \mathsf{coNTIME}[2^{\varepsilon n}]$ for a small constant $\varepsilon > 0$, then for any $m \in \mathbb{N}$, any string s such that $\mathsf{nKt}(s) \geq m$ will have MnKtP-oracle circuit complexity at least $m/\mathsf{polylog}(m)$. In particular, arguing in the contrapositive, assume that s is the truth-table of an MnKtP-oracle circuit C of size less than $m/(\log m)^5$. Then, we may encode the circuit C with fewer than $m/\log m$ bits into an nKt-description for s, and moreover, we may collapse the non-determinism required to simulate the assumed $\mathsf{NTIME}[2^{\varepsilon n}] \cap \mathsf{coNTIME}[2^{\varepsilon n}]$ oracle gates into the non-determinism in the definition of nKt. Overall, this implies

$$\begin{aligned} \mathsf{nKt}(s) &< \frac{m}{\log m} + \log(m \cdot 2^{\varepsilon m}) \\ &< m. \end{aligned}$$

To obtain such s efficiently, again applying the assumption that MnKtP is easy, we can non-deterministically guess s and refute when $\mathsf{nKt}(s)$ is not large enough. Then, we can plug s into the (relativized) pseudo-random generator from [IW97] to derandomize the reconstruction procedure of [RRV02, Hir20], resulting in an $\mathsf{nKt}^{\mathsf{MnKtP}}$ upper bound rather than an $\mathsf{rKt}^{\mathsf{MnKtP}}$ upper bound. In this case, we can again use the assumed easiness of MnKtP to collapse the oracle into the non-determinism in the definition of nKt, resulting in an nKt upper bound without any oracle.

The technique in the previous paragraph can be used to obtain the lower bound $\mathsf{MnKtP} \notin \mathsf{NTIME}[T(n)] \cap \mathsf{coNTIME}[T(n)]$ for $T(n) = 2^{n^{\varepsilon}}$, but it does not suffice for $T(n) = 2^{\varepsilon n}$. This is because, in order to derandomize the reconstruction procedure of [RRV02, Hir20] via [IW97], we would need a string of MnKtP-oracle complexity some large polynomial in n, so we would need to query the MnKtP-oracle at length n^c for some constant $c \in \mathbb{N}$. Hence, the derandomized reconstruction would take time at least $T(n^c)$, and when $T(n) > 2^{n^{1/c}}$, it holds that $T(n^c) > 2^n$, yielding a trivial upper bound on the relevant nKt complexity. To obtain a lower bound with $T(n) = 2^{\varepsilon n}$, a key observation is that the reconstruction procedure in question only requires $O(n)$ bits of randomness. Then, instead of [IW97], we apply the PRG of [RRV02, Hir20] *for a second time*, instantiated with a different set of parameters, to derandomize the reconstruction procedure. The upshot is that a truth-table with nKt complexity $O(n)$ suffices to generate the $O(n)$ pseudo-random bits to fool reconstruction. In order for the argument to go through in this setting, we use some similar ideas as described in the previous paragraph: arguing, for example, that assuming easiness of MnKtP, a string of high nKt complexity must also have high $\mathsf{rKt}^{\mathsf{MnKtP}}$ complexity. We note that the analysis here is quite delicate. For example, one must keep careful track of the randomness length in the definition of rK as well as the input lengths of oracle queries. See Section Section 4.1.3 for more details.

However, one problem remains: the reconstruction of the pseudo-random generator in [Hir22c] only recovers x with $1/\mathsf{poly}(n)$ probability, so we need some additional advice to identify x after derandomization. But the advice depends on our non-deterministic choice of hard truth table s, so we cannot store it in the description. Nevertheless, using ideas from pair-wise independent sampling (see e.g. [Gol97]), we can amplify the success probability of reconstruction from $1/\mathsf{poly}(n)$ to $2/3$, but still only using $O(n)$ bits of randomness (see Section 4.1.1). This eliminates the additional advice after derandomization, because we can just output the string that appears the most often.

We have given a brief overview of the proof for the main part of Theorem 1.1. Next we discuss the proof for the moreover part. We observe that most of the arguments (reconstruction, derandomization, etc.) in the first step of the proof (i.e., getting weak symmetry of information from easiness of MnKtP) still work, because we are only using the fact that $\mathcal{R}_{\mathsf{nKt}[n]}$ is dense. Therefore the proof extends to any dense language $L \subseteq \mathcal{R}_{\mathsf{nKt}[n]}$.

For the second step of the proof, weak symmetry of information only guarantees existence of x such that $\mathsf{nKt}(x, y^*) \geq 2n - O(\varepsilon n)$, but does not guarantee $(x, y^*) \in L$, so it is possible we cannot find such x^* in the second brute-force. Fortunately, the first step of the proof actually gives us something stronger than symmetry of information: For any $x, y \in \{0,1\}^*$, when output length of G_1, G_2 are configured as before, $\exists z_1, z_2$ such that $G_1(x, z_1) \circ G_2(y, z_2) \in L$ (see Lemma 4.9 for more detail). Then in the second step, given $y \in L$, we enumerate over not only x, but also z_1, z_2, to find $G_1(x, z_1) \circ G_2(y, z_2) \in L$, which are guaranteed to exist and verifiable in presence of L-oracle.

This works for any dense language $L \in \mathcal{R}_{\mathsf{nKt}[(1-o(1))\cdot n]}$. However, if $L \in \mathcal{R}_{\mathsf{nKt}[\gamma \cdot n]}$ for some $\gamma \in (0,1)$, then we only get

$$\mathsf{nKt}(x, y) \geq \gamma \cdot (\mathsf{nKt}(x \mid y) + \mathsf{nKt}(y)) - O(\varepsilon n)$$

for the first step. Then in the second step, we can find $y_1^* \in L$ so that $\mathsf{nKt}(y_1^*) \geq \gamma \cdot n$, and then we can find $x_1^* \in \{0,1\}^n$ and $y_2^* = G_1(x_1^*, z_{1,1}) \circ G_2(y_1^*, z_{1,2}) \in L$ such that

$$\mathsf{nKt}(y_2^*) \geq \gamma \cdot (\mathsf{nKt}(x_1^* \mid y_1^*) + \mathsf{nKt}(y_1^*)) - O(\varepsilon n) \geq (\gamma + \gamma^2) \cdot n - O(\varepsilon n).$$

We can see the problem here: when $\gamma \leq (\sqrt{5} - 1)/2 \approx 0.618$, we are not guaranteed that $\mathsf{nKt}(y_2^*) \geq (1 + \Omega(1)) \cdot n$, so we cannot derive a contradiction. However, we apply our extended weak SoI (Lemma 4.9) again on y_2^*, which enables us to find $x_2^* \in \{0,1\}^n$ and $y_3^* = G_1(x_2^*, z_{2,1}) \circ G_2(y_2^*, z_{2,2})$ such that

$$\mathsf{nKt}(y_3^*) \geq \gamma \cdot (\mathsf{nKt}(x_2^* \mid y_2^*) + \mathsf{nKt}(y_2^*)) - O(\varepsilon n) \geq (\gamma + \gamma^2 + \gamma^3) \cdot n - O(\varepsilon n).$$

Iterating the construction k times, we will obtain a string y^*_k, such that

$$\mathsf{nKt}(y^*_k) \geq (\gamma + \gamma^2 + \cdots + \gamma^k) \cdot n - O(k\varepsilon n) = \frac{\gamma - \gamma^{k+1}}{1-\gamma} \cdot n - O(k\varepsilon n).$$

Hence when $\gamma > 1/2$, we can find a large enough constant k such that $\mathsf{nKt}(s^*_k) \geq (1 + \Omega(1)) \cdot n - O(k\varepsilon n)$. For each iteration, we do exhaustive search only to find $x^*_k \in \{0,1\}^n$ and two seeds of length $\mathsf{polylog}(n)$, so the time it takes to construct s^*_k is only $k \cdot 2^n \cdot 2^{O(\varepsilon n)}$, resulting in a contradiction for a small enough constant $\varepsilon > 0$.

The detailed proof can be found in Section 4.1.4. On the other hand, Section 4.1.5 discusses the limits of our techniques for $\gamma \leq 1/2$.

Lower bounds for computing rnKt (Theorem 1.2). We now move on to techniques behind Theorem 1.2. The proof of Item 1 is a natural extension to the rKt result in [Hir22c], in the sense that the correspondence between AM computation and rnK complexity (see Lemma 3.11) is a natural analogy to that between BPTIME computation and rK complexity. Conceptually, it is even simpler than the proof for Theorem 1.1 since derandomization is not required here. Still, the combination of randomness and nondeterminism makes some parts of the argument slightly more involved, and we include a complete proof in Section 4.2.1 to keep the presentation self-contained.

Proof of Item 2 and Item 3 exploit new results from [CLL24], where *explicit construction algorithm* for dense coAM properties is proposed in the form of Arthur-Merlin protocol with non-binary output. Briefly speaking, a *non-binary public coin protocol* (P, V) has procedure identical to the normal version of Arthur-Merlin protocol, except that V outputs a string instead of a bit in the end. At the start of the protocol, prover P (which is computationally unbounded) and verifier V both receive an *advice* string α. Then in the i-th round, V sends uniform randomness r_i to P, and P returns answer w_i based on α, $\{r_j\}_{j\leq i}$ and $\{w_j\}_{j<i}$. After constant rounds of interaction, V outputs a string x, based on α, $\{r_j\}$ and $\{w_j\}$. [CLL24] then defined *single-valued Arthur-Merlin protocol* as non-binary public coin protocol with *conformity* and *resiliency* conditions respectively, analogous to *completeness* and *soundness* condition in normal Arthur-Merlin protocol. See Section 4.2.2 for detail.

The main technical result of [CLL24] (see Section 4.2.2 for details) proves that for every dense enough $L \in \mathsf{coAM}$, there exists a sequence $\{x_n\}$ and non-binary public-coin protocol (P, V) satisfying

- (P, V) is single-valued Arthur-Merlin protocol computing $\{x_n\}$;
- The advice complexity $l(n)$ is small, and V can be computed efficiently;
- For infinitely many n, $x_n \in L \cap \{0,1\}^n$.

We note that sequence $\{x_n\}$ computed by such efficient non-binary Arthur-Merlin protocol admits small rnK complexity, say, $\mathsf{rnKt}(x_n) \leq \theta(n) < n - 1$. This is a non-binary analogy of instantiating non-deterministic computation into non-deterministic Kolmogorov complexity (as in Lemma 3.5 and Lemma 3.11). Then we consider any dense language $L \subseteq \mathcal{R}_{\mathsf{rnKt}[\theta(n)+1]}$ and assume $L \in \mathsf{coAM}$. By the existence of such Arthur-Merlin protocol, we know that for infinitely many n, $L \cap \{0,1\}^n$ contains string with rnKt complexity smaller than $\theta(n) + 1$, leading to immediate contradiction. In our formal proof, we extend this idea to quasi-polynomial time regime by simple padding argument.

One important corollary of this explicit construction protocol in [CLL24] is $(\mathsf{AMEXP} \cap \mathsf{coAMEXP})/2^{n^{\varepsilon}} \not\subseteq \mathsf{SIZE}^{\mathcal{O}}[2^n/n]$ for any oracle $\mathcal{O} \in \mathsf{AM} \cap \mathsf{coAM}$, as stated in [CLL24, Theorem 1.2]. This follows from the close connection between explicit construction and circuit lower bounds: Roughly speaking, if we define Π_{hard} as the set of strings in $\{0,1\}^{2^n}$ that are not truth table of circuits of size at most $2^n/n$, then Π_{hard} is dense, and $\Pi_{\mathsf{hard}} \in \mathsf{coAM}$. Then using the explicit construction protocol from above, we can efficiently construct a sequence of strings $\{h_{2^n}\}$ such that $h_{2^n} \in \Pi_{\mathsf{hard}} \cap \{0,1\}^{2^n}$ for infinitely many n. If we define a language L_{hard} whose truth table on input length n is h_{2^n}, then $L_{\mathsf{hard}} \in (\mathsf{AMEXP} \cap \mathsf{coAMEXP})/2^{n^{\varepsilon}} \setminus \mathsf{SIZE}^{L}[2^n/n]$.

Our idea for proving Theorem 1.2, Item 3 is to instantiate classical pseudo-random generator construction from [IW97] by truth table of L_{hard}, obtaining a family of multi-sets $\{T_s\}_{s\in\mathbb{N}}$ which are pseudo-random against $(\mathsf{AM} \cap \mathsf{coAM})$-oracle circuits of size s on infinitely many input lengths s.

Note that, because L_{hard} is computable in $(\mathsf{AMEXP}{\cap}\mathsf{coAMEXP})/2^{n^{\varepsilon}}$ and [IW97] generator is polynomial-time computable, every element inside T_s has rnKt complexity bounded by $2^{O(\log^{\varepsilon} s)}$. Therefore, any dense language L consisting only of strings with large rnKt complexity distinguishes T_s from uniform random strings, for every sufficiently large s. The pseudorandom condition then gives $L \notin \mathsf{SIZE}^{\mathsf{AM}\cap\mathsf{coAM}}[\mathsf{poly}(n)]$. Similarly to Item 2, the lower bound is later amplified to quasi-polynomial by padding argument.

Proof complexity and symmetry of information (Theorems 1.3 and 1.4). We now discuss the techniques used to prove Theorem 1.3 and Theorem 1.4. We focus here on the versions of these statements concerning pnKt rather than rnKt, since both versions admit similar proofs.

Theorem 1.3 states that, unconditionally, SoI holds for pnKt with high probability over random strings sampled efficiently. The theorem is proved by an argument closely resembling one from [HLO25]. Briefly, the starting point is a worst-case "semi-SoI" statement for pnK^t due to [HLO25].[3] That is, for some polynomial p, for all sufficiently large $n, t \in \mathbb{N}$ and strings $x, y \in \{0,1\}^n$,

$$\mathsf{pK}^t(x,y) \gtrsim \mathsf{pnK}^{p(t)}(x) + \mathsf{pnK}^{p(t)}(y \mid x). \tag{1}$$

Following [HLO25], we apply the fact that, for distributions $\mathcal{D} \in \mathsf{PSAMP}$, with high probability over $(x,y) \sim \mathcal{D}$,

$$\mathsf{K}(x,y) \gtrsim \mathsf{pK}^{\mathsf{poly}(n)}(x,y),$$

and $\mathsf{pnKt}(x,y) \gtrsim \mathsf{K}(x,y)$ unconditionally. Given these observations, Theorem 1.3 follows from a simple adaptation of the right-hand side of Equation (1) to the setting of pnKt: noting, for example, that $\mathsf{pnKt}(x) \lesssim \mathsf{pnK}^{\mathsf{poly}(n)}(x)$.

We now move on to describe the proof of Theorem 1.4, which builds on techniques from [Ron04, CLL24, HLO24]. We would like to argue in the contrapositive: namely, assuming NP is contained in $\mathsf{coAMTIME}[n^{\mathsf{polylog}(n)}]$ and demonstrating *asymmetry* of information for pnKt. Here, our starting point is the proof technique excluding SoI for Kt in [Ron04].

Unfortunately, the same argument does not yield asymmetry in the case of nKt, let alone pnKt. The problem in the case of nKt is that, during the exhaustive search over programs of length n, one must determine whether a given program has a unique nondeterministic output. This requires searching over witnesses of length up to 2^n, meaning that exhaustive search would take time doubly-exponential in n. The problem in cases of randomized notions of Kt complexity is that known exponential-time procedures for determining complexity are probabilistic, so they cannot be relied upon to produce canonical high-complexity strings. In our case, of course, we must overcome both of these issues.

To that end, a first observation is that there exists a randomized SAT-oracle algorithm (which we denote B^{SAT}) for approximating pnKt complexity. More specifically, given a pair of strings $u \in \{0,1\}^n$ and $v \in \{0,1\}^{\leq 2^n}$, B^{SAT} runs in time $2^{O(n)}$, accepts with high probability if $\mathsf{pnKt}(u \mid v) \geq 2n/3$, and rejects with high probability if $\mathsf{pnKt}(u \mid v) < n/2$.[4] At a high level, our approach entails derandomizing B^{SAT}

[3] In fact, by an alternate proof, we obtain a stronger form of the statement with $\mathsf{pK}^{p(t)}(x)$ replacing $\mathsf{pnK}^{p(t)}(x)$. See Section 5.1.1 for details.

[4] An algorithm with these parameters can be obtained from Lemma 5.4 combined with success amplification for pnKt following [GKLO22].

appropriately, yielding an adequately efficient procedure to produce a canonical string x with $\mathsf{pnKt}(x) \geq n/2$ (and, likewise, a canonical string y with $\mathsf{pnKt}(y \mid x) \geq n/2$).

We now describe how to derandomize B^{SAT}. Here we apply the framework due to Chen et al. [CLL24]. In more detail, we adapt results of [CLL24] to obtain a family of multisets $\{T_s\}_{s \in \mathbb{N}}$ pseudorandom infinitely often against ($\mathsf{AM} \cap \mathsf{coAM}$)-oracle circuits of size s. Moreover, for any desired constant $\delta > 0$, each T_s has a representation of rnKt complexity at most $2^{(\log s)^{\delta}}$, and each T_s contains at most $\mathsf{poly}(s)$ elements. Making use of our assumption that $\mathsf{NP} \subseteq \mathsf{coAMTIME}[n^{\mathsf{polylog}(n)}]$, on input (u, v), B^{SAT} can be simulated by an ($\mathsf{AM} \cap \mathsf{coAM}$)-oracle circuit $C_{(u,v)}$ of size at most $2^{\mathsf{poly}(n)}$ that treats its input as internal randomness for B^{SAT}. Thus, for infinitely many choices of $s = 2^{\mathsf{poly}(n)}$, the pseudorandom set T_s fools such circuits C.

Combining the above ideas, for infinitely many input lengths $n \in \mathbb{N}$ and any desired constant $\delta' > 0$, we may obtain a pair of strings $x, y \in \{0,1\}^n$ with $\mathsf{pnKt}(x) \geq n/2$ and $\mathsf{pnKt}(y \mid x) \geq n/2$, but $\mathsf{pnKt}(x, y) \leq \mathsf{rnKt}(x, y) \leq 2^{n^{\delta'}}$. For example, let x be the lexicographically first string of length n such that $C_{(x,\epsilon)}$ accepts with probability greater than $1/2$ over T_s, where ϵ denotes the empty string. On one hand, the lower bound $\mathsf{pnKt}(x) \geq n/2$ follows from the definition of $C_{(x,\epsilon)}$ and the pseudorandom property of T_s. (One may define y such that $\mathsf{pnKt}(y \mid x) \geq n/2$ similarly.) On the other hand, given T_s, an exhaustive procedure to *construct* x and y runs in time $2^{\mathsf{poly}(n)}$ provided an ($\mathsf{AM} \cap \mathsf{coAM}$)-oracle. This, together with the upper bound on $\mathsf{rnKt}(T_s)$, implies $\mathsf{rnKt}^{\mathsf{AM} \cap \mathsf{coAM}}(x, y) \leq 2^{n^{\delta'}/2}$. The desired bound $\mathsf{rnKt}(x, y) \leq 2^{n^{\delta'}}$ follows by "collapsing" the ($\mathsf{AM} \cap \mathsf{coAM}$)-oracle into the nondeterminism in the definition of rnKt.

Of course, the pair x, y obtained above does not yet witness asymmetry, since $n/2 + n/2 \ll 2^{n^{\delta'}}$. To obtain a suitable pair of strings, we apply the above approach iteratively, as in [HLO24]; in particular, we obtain a sequence of $\ell := 2^{n^{\delta'}}$ strings $x_1, ..., x_\ell \in \{0,1\}^n$ such that

$$\sum_{i \in [\ell]} \mathsf{pnKt}(x_i \mid x_1, ..., x_{i-1}) \geq \ell \cdot n/2$$

but

$$\mathsf{pnKt}(x_1, ..., x_\ell) \leq \ell.$$

The above can then be used to show a violation of the standard formulation of SoI for pnKt; see Section 5.2 for further details.

Average-case complexity and meta-complexity (Theorem 1.5). Our contribution in Theorem 1.5 is mainly conceptual. The proof builds on techniques from prior work [Hir20, HLO25]. Recall that our goal is to establish the equivalence between the following two implications:

- $\mathsf{DistPH} \subseteq \mathsf{AvgBPP} \implies \mathsf{NP} \subseteq \mathsf{BPP}$ (note that $\mathsf{NP} \subseteq \mathsf{BPP}$ if and only if $\mathsf{PH} \subseteq \mathsf{BPP}$),
- $\mathsf{Gap\text{-}MINpKT}^{\mathsf{PH}} \in \mathsf{prBPP} \implies \mathsf{Mild\text{-}Gap\text{-}MINpKT}^{\mathsf{PH}} \in \mathsf{prBPP}$.

To establish this equivalence, it suffices to prove the following two results:

1. $\mathsf{DistPH} \subseteq \mathsf{AvgBPP} \iff \mathsf{Gap\text{-}MINpKT}^{\mathsf{PH}} \in \mathsf{prBPP}$,
2. $\mathsf{NP} \subseteq \mathsf{BPP} \iff \mathsf{Mild\text{-}Gap\text{-}MINpKT}^{\mathsf{PH}} \in \mathsf{prBPP}$.

The first item can be seen as a randomized extension of the main result from [Hir20], which shows that $\mathsf{DistPH} \subseteq \mathsf{AvgP}$ if and only if $\mathsf{Gap\text{-}MINKT}^{\mathsf{PH}} \in \mathsf{P}$. It is worth noting that a significant portion of that work is devoted to showing that if $\mathsf{Gap\text{-}MINKT}^{\mathsf{PH}} \in \mathsf{P}$, then one can construct an explicit pseudorandom generator. This, in turn, implies that $\mathsf{BPP} = \mathsf{P}$, which is essential for establishing that $\mathsf{DistPH} \subseteq \mathsf{AvgP}$. By instead considering the randomized setting and the notion of probabilistic Kolmogorov complexity $\mathsf{pK}^{t,\mathsf{PH}}$ in place of $\mathsf{K}^{t,\mathsf{PH}}$, we bypass the need for such a pseudorandom generator construction, thereby also significantly simplifying the proof.

The second item follows the approach developed in [HLO25], which shows that if $\mathsf{MINnKT} \in \mathsf{BPP}$, then $\mathsf{NP} \subseteq \mathsf{BPP}$. In fact, the same conclusion is proved for the problem $\mathsf{MINpnKT}$. A key technical component of their proof is a language compression result for the notion of probabilistic nondeterministic Kolmogorov complexity, pnK^t (see [HLO25, Theorem 4]).

Our first observation is that $\mathsf{pK}^{t,\mathsf{SAT}}$ is a stronger notion than pnK^t, in the sense that $\mathsf{pK}^{\mathsf{poly}(t),\mathsf{SAT}}(x) \lesssim \mathsf{pnK}^t(x)$ for any x and t (see Lemma 3.15). Therefore, $\mathsf{pK}^{t,\mathsf{SAT}}$ also satisfies the desired language compression property. This turns out to be sufficient for the proof to go through in the case where $\mathsf{MINpKT}^{\mathsf{SAT}}$ is assumed to be easy.

Another observation is that, while the original proof crucially explore the fact that in the problem MINnKT the time bounds in the yes and no cases are the same—unlike in $\mathsf{Gap}\text{-}\mathsf{MINnKT}$, where the time bound is t in the yes case and $\mathsf{poly}(t)$ in the no case—the argument can still be adapted to accommodate a mild gap, where the no case has a time bound of the form $t+p(n)$, with $p(n)$ being a polynomial independent of t. This, in turn, allows us to assume only that $\mathsf{Mild}\text{-}\mathsf{Gap}\text{-}\mathsf{MINpKT}^{\mathsf{SAT}}$ is easy.

Acknowledgements. This work received support from the UKRI Frontier Research Guarantee Grant EP/Y007999/1 and the Centre for Discrete Mathematics and its Applications (DIMAP) at the University of Warwick.

2 Preliminaries

We use ϵ to denote the empty string. The length of a string x is denoted $|x|$. Given strings $x \in \{0,1\}^n$ and $y \in \{0,1\}^m$, we let $x \circ y \in \{0,1\}^{n+m}$ denote their concatenation.

We write $x \sim \mathcal{D}$ to denote that x is sampled according to the distribution $\mathcal{D}$. For an element a in the support of $\mathcal{D}$, we let $\mathcal{D}(a)$ denote its probability. We use $\mathcal{U}_\ell$ to denote the uniform distribution over $\{0,1\}^\ell$.

2.1 Computational Models

We consider multi-tape Turing machines over the alphabet $\Sigma = \{0,1,\perp\}$. In addition to the usual input tape and work tapes, in some contexts we allow a machine to access three additional tapes: one holding an auxiliary string, one holding a non-deterministic guess, and one holding the randomness. We assume that the machine has *sequential access* (i.e., no random access) to the randomness tape and the non-deterministic tape.[5]

For a machine M, time bound t, and string x, we let $\mathsf{K}^t_M(x)$ be the minimum length of a string $p \in \{0,1\}^*$ such that M outputs x on input string p when running for at most t steps. We say that a (universal) machine U is *time-efficient* if for every machine U' there is a constant C such that, for every string x and time bound t, we have $\mathsf{K}_U^{C\cdot t\log t}(x) \leq \mathsf{K}^t_{U'}(x) + C$. It is known that time-efficient machines exist (see, e.g., [LV19]). We fix such a machine U when specifying Kolmogorov complexity.

We also assume an encoding function $\langle\cdot\rangle$ for machines (which we omit for notational simplicity) such that U, when given (the encoding of) a machine M, an input α, an auxiliary string x, a non-deterministic guess w, and a random string r, runs in time $\widetilde{O}(|M|,t)$ and outputs the string produced by $M(\alpha;x;w;r)$, where $|M|$ denotes the encoding length of M and t is the number of steps M runs on $(\alpha;x;w;r)$.

[5] To avoid confusion, sequential access means that the tape head can remain at a given tape cell or move left or right at each step of the computation. In particular, we do not require read-only or one-way access to the tape.

A *program* Π is defined as a pair $\Pi := (M, \alpha)$, where M is a valid encoding of a machine and $\alpha \in \{0,1\}^*$ is an "advice" string. The output of Π on $(x; w; r)$ (which corresponds to the auxiliary input, the non-deterministic guess, and the randomness), denoted $\Pi(x; w; r)$, is given by $M(\alpha; x; w; r)$.

The *running time* of the program Π on input (x, w, r) is defined to be the number of steps M runs on input $(\alpha; x; w; r)$.

We also define the *size* of the program Π. To encode a machine-advice pair (M, α), we use the following format:

$$s_1\, 01\, \langle M \rangle\, s_2\, 01\, \alpha,$$

where $\langle M \rangle$ is an encoding of the machine M,[6] and s_1 (resp. s_2) is the binary representation of the integer specifying the length of $\langle M \rangle$ (resp. α), with each bit duplicated. The size of Π, denoted $|\Pi|$, is the total bit length of this description.

We say that a program Π *non-deterministically* outputs y on input $(x; r)$ in time t if:

- $\forall w \in \{0,1\}^t$, $\Pi(x; w; r)$ outputs either y or $\perp$ in time t, and
- $\exists w \in \{0,1\}^t$, $\Pi(x; w; r)$ outputs y in time t.

A Computational Model with Efficient Composition. The above computational model suffices for most of our results, except for Theorem 1.5. As in the main result of [HLO25], Theorem 1.5 requires a computational model that possesses an additional natural efficiency property called "efficient composition". Specifically, this property states that there exists a polynomial p such that the following holds. For any $x, y, z, r \in \{0,1\}^*$, if:

- there is a program of size s_1 that outputs y on input $(x; r)$ in time t_1, and
- there is a program of size s_2 that outputs z on input y in time t_2,

then there is a program of size $s_1 + s_2 + \log p(|x| + |y| + |z|)$ that outputs z on input $(x; r)$ in time $t_1 + p(t_2) + p(|x| + |y| + |z|)$. (As discussed in detail in [HLO25], this property is typically implicitly assumed in the literature on time-bounded Kolmogorov complexity.)

To guarantee the efficient composition property, we consider *a sequence of machine-advice pairs*. More formally, a program Π is now defined as

$$\Pi := ((M_1, \alpha_1), \ldots, (M_\ell, \alpha_\ell)),$$

where each M_i is a valid encoding of a machine and $\alpha_i \in \{0,1\}^*$. The output of Π on $(x; r)$ (which corresponds to the auxiliary input and the randomness), denoted $\Pi(x; r)$, is obtained as follows:[7]

- Let y_1 be the output of $M_1(\alpha_1; x; r)$. Let t_1 be the number of steps $M_1(\alpha_1; x; r)$ takes to produce y_1.
- Let y_2 be the output of $M_2(\alpha_2; x \circ y_1; r_{[t_1+1:|r|]})$, and let t_2 be the corresponding running time.

$$\vdots$$

- Let y_ℓ be the output of $M_\ell(\alpha_\ell; x \circ y_{\ell-1}; r_{[t_{\ell-1}+1:|r|]})$, and let t_ℓ be the corresponding running time.
- Output y_ℓ.

[6] As noted above, we assume a fixed standard encoding for Turing machines.

[7] For a string z of length ℓ and a subset $S \subseteq [\ell]$, where $[\ell] = \{1, 2, \ldots, \ell\}$, we let z_S denote the substring of z obtained by concatenating the bits of z at indices in S. For $a < b$, we let $[a : b]$ denote the set $\{a, a+1, \ldots, b\}$.

The running time of the program Π on input $(x; r)$ is given by $\sum_{i=1}^{\ell} t_i$.

We now describe how the program Π is encoded. A machine-advice pair is encoded as described before. The full description of Π is obtained by concatenating the encodings of all pairs (M_i, α_i) for $i \in [\ell]$. The size of Π, denoted $|\Pi|$, is the total bit length of this description.

It can be shown that this definition of a program satisfies the efficient composition property (see [HLO25, Proposition 57]). The definition can be extended to programs that have oracle access to a language $\mathcal{O}$ in the natural way.

2.2 Kolmogorov Complexity

2.2.1 Definitions

In this subsection, we provide formal definitions for various notions of time-bounded Kolmogorov complexity.

Time-Bounded Kolmogorov Complexity. We begin with deterministic versions of time-bounded Kolmogorov complexity.

Definition 2.1 (K^t)**.** For $x, y, r \in \{0,1\}^*$, $t \in \mathbb{N}$, and a oracle $\mathcal{O}$, the *t-time-bounded Kolmogorov complexity of x conditioning on $(y; r)$ and given oracle $\mathcal{O}$* is defined as

$$\mathsf{K}^{t,\mathcal{O}}(x \mid y; r) \triangleq \min_{\Pi \in \{0,1\}^*} \left\{ |\Pi| \;\middle|\; \Pi^{\mathcal{O}}(y; \epsilon; r) \text{ outputs } x \text{ within } t \text{ steps} \right\}.$$

We define $\mathsf{K}^{t,\mathcal{O}}(x \mid y)$ as $\mathsf{K}^{t,\mathcal{O}}(x \mid y; \epsilon)$, and $\mathsf{K}^{t,\mathcal{O}}(x)$ as $\mathsf{K}^{t,\mathcal{O}}(x \mid \epsilon)$.

Definition 2.2 (Kt)**.** For $x, y, r \in \{0,1\}^*$ and a oracle $\mathcal{O}$, the *time-bounded Levin–Kolmogorov complexity of x conditioning on $(y; r)$ and given oracle $\mathcal{O}$* is defined as

$$\mathsf{Kt}^{\mathcal{O}}(x \mid y; r) \triangleq \min_{\substack{\Pi \in \{0,1\}^* \\ t \in \mathbb{N}}} \left\{ |\Pi| + \lceil \log t \rceil \;\middle|\; \Pi^{\mathcal{O}}(y; \epsilon; r) \text{ outputs } x \text{ within } t \text{ steps} \right\}.$$

We define $\mathsf{Kt}^{\mathcal{O}}(x \mid y)$ as $\mathsf{Kt}^{\mathcal{O}}(x \mid y; \epsilon)$, and $\mathsf{Kt}^{\mathcal{O}}(x)$ as $\mathsf{K}^{\mathcal{O}}(x \mid \epsilon)$.

We can consider the case where the oracle calls are *non-adaptive*. In this case, we will write $\mathsf{K}^{t,\|\mathcal{O}}(x)$ or $\mathsf{Kt}^{\|\mathcal{O}}(x)$.

Next, we will define other variants of the time-bounded Kolmogorov complexity measure. For simplicity, we will define only the basic versions without oracles. These can easily be generalized to settings where an oracle is present.

Time-Bounded Kolmogorov Complexity with Randomness. We now define probabilistic variants of time-bounded Kolmogorov complexity. These definitions include a parameter λ in the subscript to denote the "success probability". We will often omit the subscript when $\lambda = 2/3$.

Definition 2.3 (rK^t)**.** For $x, y \in \{0,1\}^*$, $\lambda \in [0,1]$, $t \in \mathbb{N}$, the *randomized t-time-bounded Kolmogorov complexity of x conditioning on y* is defined as

$$\mathsf{rK}_{\lambda}^{t,\mathcal{O}}(x \mid y) = \min_{\Pi \in \{0,1\}^*} \left\{ |\Pi| \;\middle|\; \Pr_{r \sim \{0,1\}^t}[\Pi(y; \epsilon; r) \text{ outputs } x \text{ within } t \text{ steps}] \geq \lambda \right\}.$$

Definition 2.4 (rKt)**.** For $x, y \in \{0,1\}^*$, $\lambda \in [0,1]$, the *randomized time-bounded Levin–Kolmogorov complexity of x conditioning on y* is defined as

$$\mathsf{rKt}^{\mathcal{O}}_{\lambda}(x \mid y) = \min_{\substack{\Pi\in\{0,1\}^* \\ t\in\mathbb{N}}} \left\{ |\Pi| + \lceil \log t \rceil \;\middle|\; \mathop{\mathbf{Pr}}_{r\sim\{0,1\}^t}[\Pi(y;\epsilon;r) \text{ outputs } x \text{ within } t \text{ steps}] \geq \lambda \right\}.$$

Definition 2.5 (pK^t)**.** For $x, y \in \{0,1\}^*$, $\lambda \in [0,1]$, $t \in \mathbb{N}$, the *probabilistic t-time-bounded Kolmogorov complexity of x conditioning on y* is defined as

$$\mathsf{pK}^t_{\lambda}(x \mid y) \triangleq \min \left\{ s \in \mathbb{N} \;\middle|\; \mathop{\mathbf{Pr}}_{r\sim\{0,1\}^t}[\exists \Pi \in \{0,1\}^s \text{ s.t. } \Pi(y;\epsilon;r) \text{ outputs } x \text{ within } t \text{ steps}] \geq \lambda \right\}.$$

Equivalently,

$$\mathsf{pK}^t_{\lambda}(x \mid y) \triangleq \min \left\{ s \in \mathbb{N} \;\middle|\; \mathop{\mathbf{Pr}}_{r\sim\{0,1\}^t}\left[\mathsf{K}^t(x \mid y; r) \leq s\right] \geq \lambda \right\}.$$

Definition 2.6 (pKt)**.** For $x, y \in \{0,1\}^*$, $\lambda \in [0,1]$, $t \in \mathbb{N}$, the *probabilistic time-bounded Levin–Kolmogorov complexity of x conditioning on y* is defined as

$$\mathsf{pKt}_{\lambda}(x \mid y) \triangleq \min \left\{ s \in \mathbb{N} \;\middle|\; \mathop{\mathbf{Pr}}_{r\sim\{0,1\}^{2^s}}[\mathsf{Kt}(x \mid y; r) \leq s] \geq \lambda \right\}.$$

Time-Bounded Kolmogorov Complexity with Nondeterminism. Next, we define nondeterministic variants of time-bounded Kolmogorov complexity.

Definition 2.7 (nK^t)**.** For $x, y, r \in \{0,1\}^*$ and $t \in \mathbb{N}$, the *nondeterministic t-time-bounded Kolmogorov complexity of x conditioning on $(y; r)$* is defined as

$$\mathsf{nK}^t(x \mid y; r) \triangleq \min_{\Pi\in\{0,1\}^*} \left\{ |\Pi| \;\middle|\; \begin{array}{l} \forall w \in \{0,1\}^t, \Pi(y;w;r) \text{ outputs } x \text{ or } \bot \text{ in } t \text{ steps;} \\ \exists w \in \{0,1\}^t, \Pi(y;w;r) \text{ outputs } x \text{ in } t \text{ steps} \end{array} \right\}.$$

We define $\mathsf{nK}^t(x \mid y)$ as $\mathsf{nK}^t(x \mid y; \epsilon)$, and $\mathsf{nK}^t(x)$ as $\mathsf{nK}^t(x \mid \epsilon)$.

Definition 2.8 (nKt)**.** For $x, y, r \in \{0,1\}^*$, the *nondeterministic time-bounded Levin–Kolmogorov complexity of x conditioning on $(y; r)$* is defined as

$$\mathsf{nKt}(x \mid y; r) \triangleq \min_{\substack{\Pi\in\{0,1\}^* \\ t\in\mathbb{N}}} \left\{ |\Pi| + \lceil \log t \rceil \;\middle|\; \begin{array}{l} \forall w \in \{0,1\}^t, \Pi(y;w;r) \text{ outputs } x \text{ or } \bot \text{ in } t \text{ steps;} \\ \exists w \in \{0,1\}^t, \Pi(y;w;r) \text{ outputs } x \text{ in } t \text{ steps} \end{array} \right\}.$$

We define $\mathsf{nKt}(x \mid y)$ as $\mathsf{nKt}(x \mid y; \epsilon)$, and $\mathsf{nKt}(x)$ as $\mathsf{nKt}(x \mid \epsilon)$.

Time-Bounded Kolmogorov Complexity with Randomness and Nondeterminism. Finally, we define time-bounded Kolmogorov complexity notions that incorporate both randomness and nondeterminism.

Definition 2.9 (rnK^t)**.** For $x, y \in \{0,1\}^*$, $\lambda \in [0,1]$ and $t \in \mathbb{N}$, the *randomized nondeterministic t-time-bounded time-bounded Kolmogorov complexity of x conditioning on y* is defined as

$$\mathsf{rnK}^t_{\lambda}(x \mid y) \triangleq \min_{\Pi\in\{0,1\}^*} \left\{ |\Pi| \;\middle|\; \mathop{\mathbf{Pr}}_{r\sim\{0,1\}^t} \begin{bmatrix} \forall w \in \{0,1\}^t, \Pi(y;w;r) \text{ outputs } x \text{ or } \bot \text{ in } t \text{ steps;} \\ \exists w \in \{0,1\}^t, \Pi(y;w;r) \text{ outputs } x \text{ in } t \text{ steps} \end{bmatrix} \geq \lambda \right\}.$$

Definition 2.10 (rnKt)**.** For $x, y \in \{0,1\}^*$, $\lambda \in [0,1]$, the *randomized nondeterministic time-bounded Levin–Kolmogorov complexity of x conditioning on y* is defined as

$$\mathsf{rnKt}_\lambda(x \mid y) \triangleq \min_{\substack{\Pi \in \{0,1\}^* \\ t \in \mathbb{N}}} \left\{ |\Pi| + \lceil \log t \rceil \;\middle|\; \Pr_{r \sim \{0,1\}^t} \begin{bmatrix} \forall w \in \{0,1\}^t, \Pi(y; w; r) \text{ outputs } x \text{ or } \perp \text{ in } t \text{ steps;} \\ \exists w \in \{0,1\}^t, \Pi(y; w; r) \text{ outputs } x \text{ in } t \text{ steps} \end{bmatrix} \geq \lambda \right\}.$$

Definition 2.11 (pnK^t)**.** For $x, y \in \{0,1\}^*$, $\lambda \in [0,1]$ and $t \in \mathbb{N}$, the *probabilistic nondeterministic t-time-bounded time-bounded Kolmogorov complexity of x conditioning on y* is defined as

$$\mathsf{pnK}^t_\lambda(x \mid y) \triangleq \min \left\{ s \in \mathbb{N} \;\middle|\; \Pr_{r \sim \{0,1\}^t} \begin{bmatrix} \exists \Pi \in \{0,1\}^{\leq s} \text{ such that the following conditions hold:} \\ \forall w \in \{0,1\}^t, \Pi(y; w; r) \text{ outputs } x \text{ or } \perp \text{ in } t \text{ steps;} \\ \exists w \in \{0,1\}^t, \Pi(y; w; r) \text{ outputs } x \text{ in } t \text{ steps} \end{bmatrix} \geq \lambda \right\}.$$

Equivalently,

$$\mathsf{pnK}^t_\lambda(x \mid y) \triangleq \min \left\{ s \in \mathbb{N} \;\middle|\; \Pr_{r \sim \{0,1\}^t} \left[\mathsf{nK}^t(x \mid y; r) \leq s \right] \geq \lambda \right\}.$$

Definition 2.12 (pnKt)**.** For $x, y \in \{0,1\}^*$, $\lambda \in [0,1]$, $t \in \mathbb{N}$, the *probabilistic nondeterministic time-bounded Levin–Kolmogorov complexity of x conditioning on y* is defined as

$$\mathsf{pnKt}_\lambda(x \mid y) \triangleq \min \left\{ s \in \mathbb{N} \;\middle|\; \Pr_{r \sim \{0,1\}^{2^s}} [\mathsf{nKt}(x \mid y; r) \leq s] \geq \lambda \right\}.$$

Computational Problems. In the remainder of this subsection, we formally define promise languages regarding decision of time-bounded complexity.

Definition 2.13. Let α, β be any $\mathbb{N} \to \mathbb{N}$ function such that $\alpha(n) < \beta(n)$ holds for any $n \in \mathbb{N}$. We define promise language $\mathsf{MKtP}[\alpha(n), \beta(n)] \triangleq (\mathsf{YES}, \mathsf{NO})$ as

$$\begin{cases} \mathsf{YES} = \{x \mid \mathsf{Kt}(x) \leq \alpha(n)\} \\ \mathsf{NO} = \{x \mid \mathsf{Kt}(x) \geq \beta(n)\} \end{cases}.$$

We also define an alternative of MKtP with variable threshold, namely, $\mathsf{Gap}_\rho\text{-}\mathsf{MKtP} \triangleq (\mathsf{YES}, \mathsf{NO})$, such that

$$\begin{cases} \mathsf{YES} = \{(x, 1^s) \mid \mathsf{Kt}(x) \leq s\} \\ \mathsf{NO} = \{(x, 1^s) \mid \mathsf{Kt}(x) \geq s + \rho(|x|)\} \end{cases},$$

where ρ is any $\mathbb{N} \to \mathbb{N}$ function. For a complexity class $\mathcal{C}$, we say that $\mathsf{Gap}_\rho\text{-}\mathsf{MKtP} \in \mathcal{C}$ if there exists a constant $c > 0$ such that $\mathsf{Gap}_\rho\text{-}\mathsf{MKtP} \in \mathcal{C}$ with $\rho(n) = c \log n$.

Definition 2.14. Let $\theta : \mathbb{N} \to \mathbb{N}$. Define $\mathcal{R}_{\mathsf{Kt}[\theta(n)]}$ as

$$\mathcal{R}_{\mathsf{Kt}[\theta(n)]} \triangleq \{x \mid \mathsf{rnKt}(x) \geq \theta(|x|)\}$$

We can extend the above definitions to any measure in $\{\mathsf{Kt}, \mathsf{rKt}, \mathsf{pKt}, \mathsf{nKt}, \mathsf{rnKt}, \mathsf{pnKt}\}$, resulting in notions such as MrnKtP, $\mathcal{R}_{\mathsf{rnKt}[\theta(n)]}$, and so on.

Definition 2.15 (MINKT)**.** For an oracle $\mathcal{O}$, we define the promise language $\mathsf{MINKT}^{\mathcal{O}} \triangleq (\mathsf{YES}, \mathsf{NO})$ as

$$\begin{cases} \mathsf{YES} = \left\{(x, 1^s, 1^t) \mid \mathsf{K}^{t,\mathcal{O}}(x) \leq s\right\} \\ \mathsf{NO} = \left\{(x, 1^s, 1^t) \mid \mathsf{K}^{t,\mathcal{O}}(x) > s\right\} \end{cases}.$$

Definition 2.16 (Gap-MINKT)**.** For an oracle $\mathcal{O}$ and $\rho\colon \mathbb{N} \to \mathbb{N}$, we define the promise language

$$\mathsf{Gap}_{\rho}\text{-}\mathsf{MINKT}^{\mathcal{O}} \triangleq (\mathsf{YES}, \mathsf{NO})$$

as

$$\begin{cases} \mathsf{YES} = \left\{(x, 1^s, 1^t) \mid \mathsf{K}^{t,\mathcal{O}}(x) \leq s\right\} \\ \;\mathsf{NO} = \left\{(x, 1^s, 1^t) \mid \mathsf{K}^{\rho(t),\mathcal{O}}(x) \geq s + \log \rho(t)\right\} \end{cases}.$$

For a complexity class $\mathcal{C}$, we say that $\mathsf{Gap}_{\rho}\text{-}\mathsf{MINKT} \in \mathcal{C}$ if there exists a constant $c > 0$ such that $\mathsf{Gap}_{\rho}\text{-}\mathsf{MINKT} \in \mathcal{C}$ with $\rho(n) = n^c$.

Definition 2.17 (Mild-Gap-MINKT)**.** For an oracle $\mathcal{O}$ and a function $\rho\colon \mathbb{N} \to \mathbb{N}$, we define the promise language $\mathsf{Mild}\text{-}\mathsf{Gap}_{\rho}\text{-}\mathsf{MINKT}^{\mathcal{O}} \triangleq (\mathsf{YES}, \mathsf{NO})$ as

$$\begin{cases} \mathsf{YES} = \left\{(x, 1^s, 1^t) \mid \mathsf{K}^{t,\mathcal{O}}(x) \leq s\right\} \\ \;\mathsf{NO} = \left\{(x, 1^s, 1^t) \mid \mathsf{K}^{t+\rho(|x|),\mathcal{O}}(x) \geq s + \log \rho(t)\right\} \end{cases}.$$

For a complexity class $\mathcal{C}$, we say that $\mathsf{Gap}_{\rho}\text{-}\mathsf{MINKT} \in \mathcal{C}$ if there exists a constant $c > 0$ such that $\mathsf{Gap}_{\rho}\text{-}\mathsf{MINKT} \in \mathcal{C}$ with $\rho(n) = n^c$.

Definition 2.18 (MINpKT)**.** For an oracle $\mathcal{O}$, we define the promise language $\mathsf{MINpKT}^{\mathcal{O}} \triangleq (\mathsf{YES}, \mathsf{NO})$ as

$$\begin{cases} \mathsf{YES} = \left\{(x, 1^s, 1^t, 1^a, 1^b) \mid \mathsf{pK}^{t,\mathcal{O}}_{b/a}(x) \leq s\right\} \\ \;\mathsf{NO} = \left\{(x, 1^s, 1^t, 1^a, 1^b) \mid \mathsf{pK}^{t,\mathcal{O}}_{(b-1)/a}(x) > s\right\} \end{cases}.$$

Definition 2.19 (Gap-MINpKT)**.** For an oracle $\mathcal{O}$ and $\rho\colon \mathbb{N} \to \mathbb{N}$, we define the promise language

$$\mathsf{Gap}_{\rho}\text{-}\mathsf{MINpKT}^{\mathcal{O}} \triangleq (\mathsf{YES}, \mathsf{NO})$$

as

$$\begin{cases} \mathsf{YES} = \left\{(x, 1^s, 1^t, 1^a, 1^b) \mid \mathsf{pK}^{t,\mathcal{O}}_{b/a}(x) \leq s\right\} \\ \;\mathsf{NO} = \left\{(x, 1^s, 1^t, 1^a, 1^b) \mid \mathsf{pK}^{\rho(t+a+b),\mathcal{O}}_{b/a}(x) \geq s + \log \rho(t+a+b)\right\} \end{cases}.$$

For a complexity class $\mathcal{C}$, we say that $\mathsf{Gap}_{\rho}\text{-}\mathsf{MINpKT} \in \mathcal{C}$ if there exists a constant $c > 0$ such that $\mathsf{Gap}_{\rho}\text{-}\mathsf{MINpKT} \in \mathcal{C}$ with $\rho(n) = n^c$.

Definition 2.20 (Mild-Gap-MINpKT)**.** For an oracle $\mathcal{O}$ and a function $\rho\colon \mathbb{N} \to \mathbb{N}$, we define the promise language $\mathsf{Mild}\text{-}\mathsf{Gap}_{\rho}\text{-}\mathsf{MINpKT}^{\mathcal{O}} \triangleq (\mathsf{YES}, \mathsf{NO})$ as

$$\begin{cases} \mathsf{YES} = \left\{(x, 1^s, 1^t, 1^a, 1^b) \mid \mathsf{pK}^{t,\mathcal{O}}_{b/a}(x) \leq s\right\} \\ \;\mathsf{NO} = \left\{(x, 1^s, 1^t, 1^a, 1^b) \mid \mathsf{pK}^{t+\rho(|x|+a+b),\mathcal{O}}_{(b-1)/a}(x) \geq s + \log \rho(t+a+b)\right\} \end{cases}.$$

For a complexity class $\mathcal{C}$, we say that $\mathsf{Gap}_{\rho}\text{-}\mathsf{MINpKT} \in \mathcal{C}$ if there exists a constant $c > 0$ such that $\mathsf{Gap}_{\rho}\text{-}\mathsf{MINpKT} \in \mathcal{C}$ with $\rho(n) = n^c$.

2.2.2 Useful Tools

Here, we state some useful tools about Kolmogorov complexity.

Lemma 2.21 (Standard Counting Argument)**.** *For any $\lambda \in (1/2, 1)$ and $\kappa \in \{\mathsf{K}, \mathsf{K}^t, \mathsf{rK}^t_\lambda, \mathsf{nK}^t, \mathsf{rnK}^t_\lambda\}$, and for all $n, \gamma \in \mathbb{N}$, we have*

$$\Pr_{x \sim \{0,1\}^n}[\kappa(x) < n - \gamma] < \frac{1}{2^\gamma}.$$

Also, for any $\lambda \in (0, 1)$ and $\kappa \in \{\mathsf{pK}^t_\lambda, \mathsf{pnK}^t_\lambda\}$, and for all $n, \gamma \in \mathbb{N}$, we have

$$\Pr_{x \sim \{0,1\}^n}[\kappa(x) < n - \gamma] < \frac{1}{\lambda \cdot 2^\gamma}.$$

Moreover, the inequalities above continue to hold in the presence of an oracle $\mathcal{O}$ and when conditioning on a fixed string y.

Proof. For simplicity, we consider the oracle-free case and $y = \epsilon$. (The general case is analogous.) The case of K directly follows from counting. Indeed, the number of Turing machines with length at most $n - \gamma$ is less than $2^{n-\gamma}$, therefore

$$\Pr_{x \sim \{0,1\}^n}[\mathsf{K}(x) < n - \gamma] < \frac{2^{n-\gamma}}{2^n} = \frac{1}{2^\gamma}.$$

The remaining cases (except $\kappa = \mathsf{pK}^t_\lambda$ and $\kappa = \mathsf{pnK}^t_\lambda$) follows similarly, since each Turing machine can correspond to at most one string under the setting that $\lambda > 1/2$.

For $\kappa = \mathsf{pK}^t_\lambda, \mathsf{pnK}^t_\lambda$, consider a bipartite graph where the left vertices represent randomness-program pairs $\{0,1\}^t \times \{0,1\}^{<n-\gamma}$, and the right vertices represent strings in $\{0,1\}^n$. There is an edge between a left vertex (r, M) and a right vertex y if $M(x)$ outputs x. For a right vertex x to have pK^t_λ-complexity at most $n - \gamma$, it must have degree at least $2^t \cdot \lambda$.

However, there are less than $2^t \cdot 2^{n-\gamma}$ total edges in the graph. Therefore, the number of right vertices with degree at least $2^t \cdot \lambda$ is less than $2^{n-\gamma}/\lambda$. □

Lemma 2.22 (See [HIL+23, Lemma 9])**.** *There exists a universal constant $b > 0$ such that for any distribution family $\{\mathcal{D}_n\}_{n\in\mathbb{N}}$, where each $\mathcal{D}_n$ is over $\{0,1\}^n$, and $\gamma \in \mathbb{N}$,*

$$\Pr_{x \sim \mathcal{D}_n}\left[\mathsf{K}(x) < \log \frac{1}{\mathcal{D}_n(x)} - \gamma\right] < \frac{n^b}{2^\gamma}.$$

Theorem 2.23 (Efficient Coding Theorem [LOZ22])**.** *For every polynomial-time samplable distribution family $\{\mathcal{D}_n\}_{n\in\mathbb{N}}$, where each $\mathcal{D}_n$ is supported over $\{0,1\}^n$, there exists a polynomial p such that for every $x \in \mathsf{Support}(\mathcal{D}_n)$,*

$$\mathsf{pK}^{p(n)}(x) \le \log \frac{1}{\mathcal{D}_n(x)} + \log p(n).$$

Lemma 2.24 (Following [GKLO22, Lemma 18])**.** *There exists a universal constant $c > 0$ such that for any $x \in \{0,1\}^*$, time bound $t \in \mathbb{N}$ and computable language L, $\mathsf{K}(x \mid t) \le \mathsf{pK}^{t,L}(x) + c \cdot \log|x|$.*

2.3 Average-Case Complexity

We review some basic definitions and facts in average-case complexity. For more information about average-case complexity, we refer to [BT06].

Recall that a pair $(L, \mathcal{D})$ is a *distributional problem* if $L \subseteq \{0,1\}^*$ and $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$ is a distribution family, where each $\mathcal{D}_n$ is a distribution over $\{0,1\}^*$.

We let DistNP denote the set of distributional problems $(L, \mathcal{D})$ such that $L \in \mathsf{NP}$ and $\mathcal{D}$ is polynomial-time samplable. Here, a distribution family $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$ is *polynomial-time samplable* if there exists a randomized polynomial-time algorithm A such that for every $n \geq 1$, the output of $A(1^n)$ is distributed according to $\mathcal{D}_n$. We let PSAMP be the collection of distribution families that can be sampled in polynomial time. For a complexity class $\mathcal{C}$ (e.g., $\mathcal{C}$ = NP), we let Dist$\mathcal{C}$ denote the set of distributional problems $(L, \mathcal{D})$ with $L \in \mathcal{C}$ and $\mathcal{D} \in \mathsf{PSAMP}$.

A distributional problem $(L, \mathcal{D})$ is said to admit a (errorless) *heuristic scheme* if there exists a deterministic polynomial-time algorithm A such that the following holds for every $n, k \in \mathbb{N}$:

1. For every $x \in \mathsf{Support}(\mathcal{D}_n)$, $A(x, 1^n, 1^k) \in \{L(x), \perp\}$,
2. and
$$\Pr_{x\sim\mathcal{D}_n}\left[A(x, 1^n, 1^k) = \perp]\right] \geq 1 - \frac{1}{k}.$$

We let AvgP denote the set of distributional problems that admit a heuristic scheme.

Similarly, a distributional problem $(L, \mathcal{D})$ is said to admit a *randomized* (errorless) *heuristic scheme* if there exists a *probabilistic* polynomial-time algorithm A such that the following holds for every $n, k \in \mathbb{N}$:

1. For every $x \in \mathsf{Support}(\mathcal{D}_n)$,
$$\Pr_{A}\left[A(x, 1^n, 1^k) \in \{L(x), \perp\}\right] \geq \frac{4}{5},$$
2. and
$$\Pr_{x\sim\mathcal{D}_n}\left[\Pr_{A}[A(x, 1^n, 1^k) = \perp] < \frac{1}{5}\right] \geq 1 - \frac{1}{k}.$$

We let AvgBPP denote the set of distributional problems that admit a randomized heuristic scheme.

We will need the following result.

Theorem 2.25 ([IL90]). *Let A be any oracle. If $(\mathsf{NP}^A, \mathcal{U}) \subseteq \mathsf{AvgBPP}$, then for every distribution $\mathcal{D} \in \mathsf{PSAMP}$ and $L \in \mathsf{NP}^A$, we have $(L, \mathcal{D}) \in \mathsf{AvgBPP}$.*

2.4 Pseudorandomness

Theorem 2.26 ([IW97]). *There is a polynomial-time computable function $F : \{0, 1\}^* \times \{0, 1\}^* \to \{0, 1\}^*$ such that the following holds. For every $\varepsilon > 0$, there exists $c, d \in \mathbb{N}$ such that*
$$F : \{0, 1\}^{n^c} \times \{0, 1\}^{d \log n} \to \{0, 1\}^n,$$
and if $f : \{0, 1\}^{c \log n} \to \{0, 1\}$ is a function with L-oracle Boolean circuit complexity at least $n^{\varepsilon c}$, then the function $G_f(\text{-}) := F(\mathsf{tt}(f), \text{-})$ is a pseudorandom generator mapping $\{0, 1\}^{d \log n}$ to $\{0, 1\}^n$, such that for any L-oracle Boolean circuit $D : \{0, 1\}^n \to \{0, 1\}$ of size at most n,
$$\left|\Pr_{x\in\{0,1\}^n}[D(x) = 1] - \Pr_{s\in\{0,1\}^{d \log n}}[D(G_f(s)) = 1]\right| \leq \frac{1}{n}.$$

We will also need the following black-box pseudorandom generator. We closely follow the construction in [Hir20, Section 4.2], which uses the weak design from [RRV02] and the list-decodable error-correcting code from [Sud97]. But our construction is different in two ways. First, in [Hir20], ECC list-decodable for distance $1/2 - 1/2m^2$ is used. Here we have an additional parameter ε (which is the advantage of the distinguisher), and we use the same family of ECC, but the list-decodable distance is set as $1/2 - \varepsilon/2m$. Second, in [Hir20], the reconstruction algorithm outputs a list of strings. Here after we obtain the list of strings, we just take a uniformly random sample from the list.

Lemma 2.27. *For all sufficiently large $n, m \in \mathbb{N}$ such that $m \leq 2n$ and any $\varepsilon > 0$, there exists a triple $(G, A, R^{(\text{-})})$ such that*

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

and for every $x \in \{0,1\}^n$ and any $D : \{0,1\}^m \to \{0,1\}$ such that

$$\Pr_{z \sim \mathcal{U}_d}[D(G(x;z)) = 1] - \Pr_{w \sim \mathcal{U}_m}[D(w) = 1] \geq \varepsilon,$$

it holds that

$$\Pr_{\substack{w \sim \mathcal{U}_r \\ z \sim \mathcal{U}_d}}\left[R^D(A(x,z), z, w) = x\right] \geq \frac{1}{\mathsf{poly}(m/\varepsilon)}.$$

We define the parameter $\varepsilon > 0$ as security parameter. Here, we have $d = O(\log^3(n/\varepsilon))$, $r = O(m)$. Moreover, G and A can be computed in time $\mathsf{poly}(n/\varepsilon)$, and R^D can be computed in time $\mathsf{poly}(n/\varepsilon)$ with oracle access to D.

Proof. Similar to [Hir20, Section 4.2]. □

Direct Product Generator . For $x, y \in \{0,1\}^n$, we denote their inner product by $x \cdot y = \bigoplus_{i=1}^n x_i y_i$.

Definition 2.28 (Direct Product Generator (DPG) [Hir21])**.** For $n, k \in \mathbb{N}$, the k-wise direct product generator is the function

$$\mathsf{DP}_k \colon \{0,1\}^n \times \{0,1\}^{nk} \to \{0,1\}^{nk+k},$$

defined by

$$\mathsf{DP}_k(x, z_1 \circ \cdots \circ z_k) = (z_1 \circ \cdots \circ z_k \circ x \cdot z_1 \circ \cdots \circ x \cdot z_k).$$

Lemma 2.29 (pK^t Reconstruction Lemma [GKLO22])**.** *There is a polynomial p_{DP} such that the following holds. For every $n, m, k \in \mathbb{N}^+$, $x \in \{0,1\}^n$, let D be a function that $(1/m)$-distinguishes $\mathsf{DP}_k(x; \mathcal{U}_{nk})$ from $\mathcal{U}_{nk+k}$. Then*

$$\mathsf{pK}^{p_{\mathsf{DP}}(n+m+k), \| D}(x) \leq k + \log p_{\mathsf{DP}}(n+m+k).$$

3 Useful Results About Nondeterministic Kolmogorov Complexity

In this section, we present several useful facts about various nondeterministic and relativized notions of Kolmogorov complexity. We note that all the results presented here also hold for the corresponding notions in the presence of any oracle. For simplicity, in below, we consider only the cases without oracles. It is straightforward to adapt all the proofs to handle the more general setting.

3.1 nK

Lemma 3.1. *There exists a constant $c > 0$ such that for any strings $x, y \in \{0,1\}^*$ and $t \in \mathbb{N}$,*

$$\mathsf{K}^{t', \| \mathsf{SAT}}(x \mid y) \leq \mathsf{nK}^t(x \mid y) + c \cdot \log t,$$

where $t' = (t + |y|)^c$.

Proof. Let $x, y \in \{0,1\}^*$ and $t \in \mathbb{N}$. We assume without loss of generality that $t \geq |x|$. Define $k = \mathsf{nK}^t(x \mid y)$, and let $\Pi \in \{0,1\}^k$ be a program that nondeterministically outputs x on input y in time t.

Given Π, $|x|$, t, and y, we reconstruct x as follows: For each $i \in [|x|]$, we use the SAT oracle to answer the following query:

> Does there exist a $w \in \{0,1\}^t$ such that $\Pi^{\mathcal{O}}(y; w)$ outputs a string $z \in \{0,1\}^{|x|}$ in time t and the i-th bit of z is 1?

We then set the i-th bit of x to be 1 if and only if the answer to this query is yes.

It is easy to verify that the above procedure runs in time $\mathsf{poly}(|x| + |y| + t)$. Also, by the fact that Π nondeterministically outputs x on input y in time t, it can correctly recover x. Thus, for $t' = \mathsf{poly}(|y| + t)$, we obtain

$$\begin{aligned}\mathsf{K}^{t',\|\mathsf{SAT}}(x \mid y) &\leq |\Pi| + O(\log |t|) \\ &\leq k + O(\log t),\end{aligned}$$

as desired. □

Corollary 3.2. *There exists a constant $c > 0$ such that for any strings $x, y \in \{0,1\}^*$,*

$$\mathsf{Kt}^{\|\mathsf{SAT}}(x \mid y) \leq c \cdot \mathsf{nKt}(x \mid y) + O(\log |y|).$$

Proof. Let $x, y \in \{0,1\}^*$ and $k = \mathsf{nKt}(x \mid y)$. Then there exists a nondeterministic program Π that outputs x given y in time t, where $|\Pi| + \lceil \log t \rceil \leq k$. This implies $\mathsf{nK}^t(x \mid y) \leq |\Pi|$.

By Lemma 3.1, we have

$$\mathsf{K}^{t',\|\mathsf{SAT}}(x \mid y) \leq |\Pi| + O(\log t),$$

where $t' = \mathsf{poly}(|y| + t)$. It follows that there exists a nondeterministic program of size at most $|\Pi| + O(\log t)$ that outputs x given y in time t', which implies

$$\mathsf{nKt}(x \mid y) \leq |\Pi| + O(\log t) + \log t' \leq \mathsf{nKt}(x \mid y) + O(\log |y|),$$

as desired. □

Lemma 3.3. *For every language $A \in \mathsf{NP}$, there exists a constant $c \in \mathbb{N}$ such that for any strings $x, y \in \{0,1\}^*$ and $t \in \mathbb{N}$,*

$$\mathsf{nK}^{t^c}(x \mid y) \leq \mathsf{K}^{t,\|A}(x \mid y) + c \cdot \log t.$$

Proof. Let $x, y \in \{0,1\}^*$ and $t \in \mathbb{N}$, and define $k = \mathsf{K}^{t,\|A}(x \mid y)$. Then there exists a program $\Pi \in \{0,1\}^k$ that on input y, makes only non-adaptive queries to A and outputs x within time t.

As in the proof of [HLO25, Lemma 31], we will eliminate the oracle access to A by guessing the answers to the (non-adaptive) queries to A and verifying them with the help of an additional small amount of advice. We provide the details for completeness.

Suppose A is the following oracle:

$$A(q) = 1 \iff \exists w \in \{0,1\}^{\mathsf{poly}(|q|)} \text{ such that } V(q, w) = 1,$$

where V is some polynomial-time verifier. Let $q_1, q_2, \ldots, q_\ell$, where $\ell \leq t$, be the (non-adaptive) queries made by d to the oracle A. Also, let p be the number of positive answers to these queries, i.e., $p := \sum_{i=1}^{\ell} A(q_i)$. We then non-deterministically guess ℓ witnesses $(w_1, w_2, \ldots, w_\ell) =: w$ and count the number of indices i for which $V(q_i, w_i) = 1$. Denote this number by u_w. If u_w is not equal to p, we output $\perp$. Otherwise, for each w_i that satisfies the verifier V, we set the answer to the query q_i to be 1, and we let the rest of the answers be 0. We then run d while simulating the oracle A using the answers obtained as described above.

Note that, given the number p, which can be specified using $\lceil \log \ell \rceil \leq \lceil \log t \rceil$ bits, the above procedure can be implemented to run in time $\mathsf{poly}(t)$. Also, it is easy to observe that there exists some guess of $w = (w_1, w_2, \ldots, w_\ell)$ that satisfies $u_w = p$. Moreover, for any guess of w that satisfies $u_w = p$, the answers obtained are the correct answers to the queries $(q_1, q_2, \ldots, q_\ell)$ for A. It follows that there exists a program of size at most $|\Pi| + O(\log t)$ that, given y, runs in time $\mathsf{poly}(t)$ and outputs x non-deterministically. Thus, for some constant $c \in \mathbb{N}$,

$$\begin{aligned} \mathsf{nK}^{t^c}(x \mid y) &\leq |\Pi| + c \cdot \log t \\ &\leq k + c \cdot \log t, \end{aligned}$$

as desired. □

As a corollary of Lemma 3.3, we get the following.

Corollary 3.4. *For every language $A \in \mathsf{NP}$, there exists a constant $c \in \mathbb{N}$ such that for any strings $x, y \in \{0,1\}^*$,*

$$\mathsf{nKt}(x \mid y) \leq c \cdot \mathsf{Kt}^{\| A}(x \mid y).$$

Lemma 3.5. *There is a constant $c > 0$ such that the following holds. For any time-constructible, non-decreasing $T : \mathbb{N} \to \mathbb{N}$, $T(n) \geq n$, and every $L \in \mathsf{NTIME}[T(n)] \cap \mathsf{coNTIME}[T(n)]$, $x, y \in \{0,1\}^*$ and $t \in \mathbb{N}$,*

$$\mathsf{nK}^{T(t)^c}(x \mid y) \leq \mathsf{nK}^{t,L}(x \mid y).$$

Specifically, restricting query length to at most $\ell \in \mathbb{N}$, for every $x, y \in \{0,1\}^*$ and $t \in \mathbb{N}$,

$$\mathsf{nK}^{(t+T(\ell))^c}(x \mid y) \leq \mathsf{nK}^{t,L_{\leq \ell}}(x \mid y).$$

Proof. Let $L \in \mathsf{NTIME}[T(n)] \cap \mathsf{coNTIME}[T(n)]$, $x, y \in \{0,1\}^*$, and $t \in \mathbb{N}$. Define $k = \mathsf{nK}^{t,L}(x \mid y)$, and let $\Pi \in \{0,1\}^k$ be a program that non-deterministically outputs x in time t given y and *oracle access to L*.

We now simulate the program Π *without* oracle access to L. To do this, whenever Π makes a query $q \in \{0,1\}^\ell$ to L, we non-deterministically guess an answer ("yes" or "no") along with a witness $w_q \in \{0,1\}^{T(\ell)}$ certifying that answer. Since $L \in \mathsf{NTIME}[T(n)] \cap \mathsf{coNTIME}[T(n)]$, each such guess can be verified in time $\mathsf{poly}(T(\ell))$. Whenever the verification fails, halt with $\perp$ being the output. This yields a nondeterministic program of size at most $|\Pi| + O(\log t)$ that outputs x in time $\mathsf{poly}(t + T(\ell))$ given y (without oracle access to L). Therefore,

$$\mathsf{nK}^{(t+T(\ell))^c}(x \mid y) \leq \mathsf{nK}^{t,L}(x \mid y) + c \cdot \log \log t$$

for some constant $c > 0$, as desired. □

Again, as a corollary of Lemma 3.5, we get the following.

Corollary 3.6. *For every $L \in \mathsf{NP} \cap \mathsf{coNP}$, there exists a constant $c > 0$ such that for every $x, y \in \{0,1\}^*$,*

$$\mathsf{nKt}(x \mid y) \leq c \cdot \mathsf{nKt}^L(x \mid y).$$

3.2 rnK

We prove that the success probability of rnK^t_δ can be amplified, a task that is non-trivial if we start with a probability of success δ that is not noticeably larger than $1/2$. For randomized notions of Kolmogorov complexity such as rK^t, success amplification in this regime can be achieved using standard hashing-based techniques. However, in rnK^t we have both randomness and nondeterminism, and must ensure that a good choice of the randomness leads to a nondeterministic encoding that never outputs the wrong string. This creates an additional difficulty and requires a modified construction.

To achieve success amplification for rnK^t, we use an idea from [BLvM05]. Roughly speaking, on top of the hashing-based framework for success amplification, we also consider the average number of positive NP queries. Then, using a frequency analysis and concentration bounds, one can prove that when we take enough samples, with high probability the number of wrong NP-oracle answers is small. The technique essentially allows one to reduce the argument to that of randomized time-bounded Kolmogorov complexity, i.e., when nondeterminism is not present.

Lemma 3.7 (Success Amplification for rnK^t)**.** *For any oracle $\mathcal{O}$, any time bound $t \in \mathbb{N}$, any $\varepsilon, \delta \in (0,1)$ and any strings $x, y \in \{0,1\}^*$ such that $|x| + |y| \le t$, we have*

$$\mathsf{rnK}^{p(t\log(1/\varepsilon)/\delta),\mathcal{O}}_{1-\varepsilon}(x \mid y) \le \mathsf{rnK}^{t,\mathcal{O}}_{\delta}(x \mid y) + \log p(t\log(1/\varepsilon)/\delta).$$

Proof. Let M be the program for $\mathsf{rnK}^{t,\mathcal{O}}_{\delta}(x \mid y)$. By definition, we have

$$\Pr_{r\sim\mathcal{U}_t}\begin{bmatrix}\exists w \in \{0,1\}^t, M^{\mathcal{O}}(y,r,w) \text{ outputs } x \text{ in } t \text{ steps},\\ \forall w \in \{0,1\}^t, M^{\mathcal{O}}(y,r,w) \text{ outputs } x \text{ or } \perp \text{ in } t \text{ steps.}\end{bmatrix} \ge \delta.$$

For any string $r \in \{0,1\}^t$, define a string $s_r \in \{0,1\}^n$, such that the i'th bit of s_r is 1 iff there exists some $w \in \{0,1\}^t$ such that the i'th bit of $M^{\mathcal{O}}(y,r,w)$ is 1. In other words, s_r is the bit-wise OR of outputs on all computation paths when the random string is r. We also define $d_r \in [0,1]$ as

$$d_r := \frac{w(s_r)}{n},$$

where $w(\cdot)$ is the Hamming weight of a string. We define $\overline{d}$ as the average d_r over $r \sim \mathcal{U}_t$, i.e.

$$\overline{d} = \mathop{\mathbf{E}}_{r\sim\mathcal{U}_t}[d_r].$$

For any $\lambda \in (0,1)$, define A_λ as

$$A_\lambda := \left\{ s \in \{0,1\}^n \;\middle|\; \Pr_{r\sim\mathcal{U}_t}[s_r = s] \ge \lambda \right\}.$$

By definition, $x \in A_\delta$. We take k independent samples $r_1, \ldots, r_k \sim \mathcal{U}_t$. The following claim is crucial to our proof:

Claim 3.8. *For $k = \left\lceil \frac{1024n^2\log(6/\varepsilon)}{\delta^2} \right\rceil$, with probability at least $1-\varepsilon$ over $r_1, \ldots, r_k \sim \mathcal{U}_t$, the following three conditions hold*:

(i) $|\{i \in [k] \mid s_{r_i} = x\}| \ge \frac{\delta k}{2}$.

(ii) For any string $x' \notin A_{\delta/8}$, $|\{i \in [k] \mid s_{r_i} = x'\}| \le \frac{\delta k}{4}$.

(iii) $|\frac{d_{r_1}+\cdots+d_{r_k}}{k} - \overline{d}| \le \frac{\delta}{32n}$.

Proof of Claim 3.8. Since $\mathbf{Pr}_{r\sim\mathcal{U}_t}[s_r = x] \geq \delta$, by Chernoff bound, the probability of Item (i) being violated is at most

$$\mathop{\mathbf{Pr}}_{r_1,\dots,r_k\sim\mathcal{U}_t}\left[|\{i\in[k] \mid s_{r_i} = x\}| \leq \frac{\delta}{2}\cdot k\right] \leq \exp\left(-2k\cdot\frac{\delta^2}{4}\right) \leq \frac{\varepsilon}{3}.$$

Similarly, the probability of Item (iii) being violated is at most

$$\mathop{\mathbf{Pr}}_{r_1,\dots,r_k\sim\mathcal{U}_t}\left[\left|\frac{d_{r_1}+\cdots+d_{r_k}}{k} - \overline{d}\right| \geq \frac{\delta}{32n}\right] \leq 2\exp\left(-2k\cdot\frac{\delta^2}{1024n^2}\right) \leq \frac{\varepsilon}{3}.$$

For Item (ii), let x' be any string in $\{0,1\}^n \setminus A_{\delta/16}$. Then we have $\mathbf{Pr}_{r\sim\mathcal{U}_t}[s_r = x'] \leq \delta/16$. Then by Chernoff bound, we get

$$\mathop{\mathbf{Pr}}_{r_1,\dots,r_k\sim\mathcal{U}_t}\left[|\{i\in[k] \mid s_{r_i} = x'\}| \geq \frac{\delta}{4}\cdot k\right] \leq \exp\left(-2k\cdot\frac{\delta^2}{64}\right) \leq \frac{\varepsilon}{3}\cdot 2^{-n}.$$

Applying union bound over all $x' \in \{0,1\}^n \setminus A_{\delta/16}$, the probability of Item (ii) being violated is at most $\varepsilon/3$. Applying union bound over Items (i) to (iii), these three conditions hold together with probability at least $1-\varepsilon$. ◇

Notice that $|A_{\delta/8}| \leq 8/\delta$. Applying Lemma 4.3 to $A_{\delta/8}$, there exists a string $u \in \{0,1\}^{O(\log(n/\delta))}$, such that the algorithm $B(1^n, u)$ runs in $\mathsf{poly}(n)$ time and outputs a circuit that computes a function $H : \{0,1\}^n \to \{0,1\}^{O(\log(1/\delta))}$, satisfying $H(x) \neq H(x')$ for every distinct pair $x, x' \in A_{\delta/8}$. Then our algorithm for recovering x works as follows:

(a) Set $k = \left\lceil \frac{1024n^2\log(6/\varepsilon)}{\delta^2} \right\rceil$, and take k independent samples $r_1,\dots,r_k \sim \mathcal{U}_t$.

(b) For each $i \in [k]$, non-deterministically guess n strings $w_{i,1},\dots w_{i,n} \in \{0,1\}^t$. We then compute $\hat{s}_{r_i} \in \{0,1\}^n$, where the j'th bit of $\hat{s}_{r_i}$ is 1 iff the j'th bit of $M^{\mathcal{O}}(y, r_i, w_{i,j})$ is 1. We also compute $\hat{d}_{r_i} := w(\hat{s}_{r_i})/n$, where $w(\cdot)$ is the Hamming weight of a string.

(c) Set the threshold $a = \lceil k(\overline{d} - \frac{\delta}{32n})\rceil$. If $\hat{d}_{r_1} + \cdots + \hat{d}_{r_k} < a$, then outputs $\perp$ and halts.

(d) Let $\hat{A}$ be the set of strings appearing at least $\lfloor \frac{3\delta}{8}\cdot k\rfloor$ times in $\hat{s}_{r_1},\dots,\hat{s}_{r_k}$. If there exists a unique string $\hat{x} \in \hat{A}$ such that $H(\hat{x})$ matches the stored hash value $H(x)$, outputs $\hat{x}$; otherwise outputs $\perp$.

Notice that we only need to store $(M, n, k, a, u, H(x))$, which takes $|M| + O(\log\frac{n\log(1/\varepsilon)}{\delta})$ bits. The running time of the above algorithm is also bounded by $\mathsf{poly}(\frac{nt\log(1/\varepsilon)}{\delta})$. It remains to estimate the success probability.

Claim 3.9. *If the random strings $r_1,\dots,r_k$ sampled by our algorithm satisfy Items (i) to (iii) of Claim 3.8, then*

1. *For all nondeterministic guesses $\{w_{i,j}\}_{i\in[k],j\in[n]}$, our algorithm either outputs x or $\perp$; and*
2. *There exists some nondeterministic guess $\{w_{i,j}\}_{i\in[k],j\in[n]}$ such that our algorithm outputs x.*

Proof of Claim 3.9. We first prove Item 1. If the algorithm reaches step (d) of the algorithm, then we have

$$\hat{d}_{r_1} + \cdots + \hat{d}_{r_k} \geq k\left(\overline{d} - \frac{\delta}{32n}\right).$$

But since $r_1, \dots, r_k$ also satisfies Item (iii) of Claim 3.8, we have

$$d_{r_1} + \cdots + d_{r_k} \le k \left(\overline{d} + \frac{\delta}{32n} \right).$$

Subtracting the two inequalities, we get

$$(d_{r_1} - \hat{d}_{r_1}) + \cdots + (d_{r_k} - \hat{d}_{r_k}) \le \frac{\delta k}{16n}.$$

By averaging argument, we get

$$\left| \left\{ i \in [k] \;\middle|\; d_{r_i} - \hat{d}_{r_i} \ge \frac{1}{n} \right\} \right| \le \frac{\delta k}{16}.$$

Since d_{r_i} counts the number of 1 in the bitwise OR of all computation paths, $d_{r_i} \ge \hat{d}_{r_i}$; and if $d_{r_i} = \hat{d}_{r_i}$, then $s_{r_i} = \hat{s}_{r_i}$ must hold. Also because $d_{r_i}, \hat{d}_{r_i}$ are both multiples of $1/n$, $d_{r_i} \ne \hat{d}_{r_i}$ implies $d_{r_i} - \hat{d}_{r_i} \ge 1/n$. Therefore $s_{r_i} \ne \hat{s}_{r_i}$ implies $d_{r_i} - \hat{d}_{r_i} \ge 1/n$, and we get

$$|\{i \in [k] \mid s_{r_i} \ne \hat{s}_{r_i}\}| \le \frac{\delta k}{16}. \tag{2}$$

By Item (i) of Claim 3.8, we get

$$|\{i \in [k] \mid s_{r_i} = x\}| \ge \frac{\delta k}{2}. \tag{3}$$

Combining Equations (2) and (3), we get

$$|\{i \in [k] \mid \hat{s}_{r_i} = x\}| \ge \frac{\delta k}{2} - \frac{\delta k}{16} = \frac{7\delta k}{16} > \frac{3\delta k}{8}.$$

Therefore, by step (d) of the algorithm, $x \in \hat{A}$ holds. Also, by Item (ii) of Claim 3.8, for any $x' \notin A_{\delta/8}$, we have

$$|\{i \in [k] \mid s_{r_i} = x'\}| \le \frac{\delta k}{4}. \tag{4}$$

Combining Equations (2) and (4), we get

$$|\{i \in [k] \mid \hat{s}_{r_i} = x'\}| \le \frac{\delta k}{4} + \frac{\delta k}{16} = \frac{5\delta k}{16} < \frac{3\delta k}{8}.$$

Therefore, by step (d) of the algorithm, $x' \notin \hat{A}$ holds. Thus we get $\hat{A} \subseteq A_{\delta/8}$. By Lemma 4.3, we can always find the correct x in step (d).

We have shown that whenever the algorithm reaches step (d), it always outputs x. Therefore to prove Item 2 of the claim, it suffices to show that there exists some $\{w_{i,j}\}_{i \in [k], j \in [n]}$ such that the algorithm reaches step (c). In fact, since s_{r_i} is the bitwise OR of all computation paths, there exists some $\{w_{i,j}\}_{i \in [k], j \in [n]}$, such that for all $i \in [k]$ we have $s_{r_i} = \hat{s}_{r_i}$, which implies $d_{r_i} = \hat{d}_{r_i}$. In this case, Item (iii) guarantees that our algorithm reaches step (d). ◇

Therefore the success probability is at least $1 - \varepsilon$ by Claim 3.8, which finishes our proof. □

Lemma 3.10. *For every $L \in \mathsf{NP}$, there is a a constant $c > 0$ such that for every $t \in \mathbb{N}$, $\lambda \in (0,1)$, and $x, y \in \{0,1\}^*$,*

$$\mathsf{rnK}^{t^c}_{\lambda/t}(x \mid y) \le \mathsf{rK}^{t,\parallel L}_{\lambda}(x \mid y) + c \cdot \log t.$$

Proof. Let $x, y \in \{0,1\}^*$ and $t \in \mathbb{N}$, and define $k = \mathsf{rK}^{t,\|A}(x \mid y)$. Then there exists a program $\Pi \in \{0,1\}^k$ such that for a λ fraction of $r \in \{0,1\}^t$, Π, when run on $(y;r)$ with non-adaptive oracle access to L, can non-deterministically output x in time t.

Fix any good r such that the above holds. Then, as in the proof of Lemma 3.3, one can eliminate the oracle access to L by hard-coding the number p of *positive* answers to the oracle queries made by Π, and non-deterministically guessing witnesses (of length at most $\mathsf{poly}(t)$) for each query. Simulate Π, attempting to answer the queries using these witnesses: if a witness verifies, answer "yes"; otherwise, answer "no." At the end, verify that exactly p witnesses were accepted.

Note that in the above, the additional advice p depends on the randomness r, and therefore we cannot have a single advice string that works for all good choices of r. However, such advice can be encoded using at most $\log t$ bits. Then for any good r, if we uniformly pick an advice string from $\{0,1\}^{\log t}$, the program can successfully output x in a nondeterministic manner as described above. It follows that there exists a program, which includes the description of Π and has size at most $|\Pi| + O(\log t)$, such that with probability at least

$$\lambda \cdot \frac{1}{2^{\log t}} = \lambda/t,$$

over $r \sim \{0,1\}^t$ and $p \sim \{0,1\}^{\log t}$, it outputs x non-deterministically in time $\mathsf{poly}(t)$. This concludes the proof. □

Lemma 3.11. *There is a constant $c > 0$ such that the following holds. For any time-constructible, non-decreasing $T : \mathbb{N} \to \mathbb{N}$, $T(n) \geq n$, and every $L \in \mathsf{NTIME}[T(n)] \cap \mathsf{coNTIME}[T(n)]$, $x, y \in \{0,1\}^*$ and $t \in \mathbb{N}$,*

$$\mathsf{rnK}^{T(t)^c}(x \mid y) \leq \mathsf{rnK}^{t,L}(x \mid y).$$

Specifically, restricting query length to at most $\ell \in \mathbb{N}$, there is a constant $c > 0$ such that for every $x, y \in \{0,1\}^*$ and $t \in \mathbb{N}$,

$$\mathsf{rnK}^{(t+T(\ell))^c}(x \mid y) \leq \mathsf{rnK}^{t,L_{\leq \ell}}(x \mid y).$$

Proof Sketch. The proof is similar to that of Lemma 3.5. It relies on the idea that, given a nondeterministic program with oracle access to a language $L \in \mathsf{AMTIME}[T(n)] \cap \mathsf{coAMTIME}[T(n)]$, one can eliminate the oracle by guessing an answer to a query along with a witness certifying that answer, and then verifying the answer efficiently in time time $\mathsf{poly}(T(n))$.

The main difference here is that L lies in $L \in \mathsf{AMTIME}[T(n)] \cap \mathsf{coAMTIME}[T(n)]$, so the verifiers are randomized. However, using standard error reduction techniques, we can ensure that all the verifications (of which there are at most t) have perfect completeness and soundness error at most $O(t^{-2})$, within time overhead of $O(\log t)$. Then, by union bound, the overall success probability of the simulated program is at least $2/3 - O(t^{-1})$, which can be amplified back to $2/3$ using Lemma 3.7 at the expense of constant overhead. □

Corollary 3.12. *For every $L \in \mathsf{AM} \cap \mathsf{coAM}$, there is a constant $c > 0$ such that for every $x, y \in \{0,1\}^*$,*

$$\mathsf{rnKt}(x \mid y) \leq c \cdot \mathsf{rnKt}^L(x \mid y).$$

3.3 pnK

Lemma 3.13. *There is a constant $c > 0$ such that the following holds. For any time-constructible, non-decreasing $T : \mathbb{N} \to \mathbb{N}$, $T(n) \geq n$, and every $L \in \mathsf{NTIME}[T(n)] \cap \mathsf{coNTIME}[T(n)]$, $x, y \in \{0,1\}^*$ and $t \in \mathbb{N}$,*

$$\mathsf{pnK}^{T(t)^c}(x \mid y) \leq \mathsf{pnK}^{t,L}(x \mid y).$$

Specifically, restricting query length to at most $\ell \in \mathbb{N}$, there is a constant $c > 0$ such that for every $x, y \in \{0,1\}^*$ and $t \in \mathbb{N}$,

$$\mathsf{pnK}^{(t+T(\ell))^c}(x \mid y) \leq \mathsf{pnK}^{t,L_{\leq \ell}}(x \mid y).$$

Proof Sketch. The poof can be easily adapted from that of Lemmas 3.5 and 3.11. We omit the details here. □

Corollary 3.14. *For every $L \in \mathsf{AM} \cap \mathsf{coAM}$, there is a constant $c > 0$ such that for every $x, y \in \{0,1\}^*$,*

$$\mathsf{pnKt}(x \mid y) \leq c \cdot \mathsf{pnKt}^L(x \mid y).$$

Lemma 3.15. *For every oracle $\mathcal{O}$ and every language L that is hard for $\mathsf{NP}^{\mathcal{O}}$ under deterministic polynomial-time reductions, there exists a constant $c > 0$ such that, for all $x, y \in \{0,1\}^*$ and $t \in \mathbb{N}$,*

$$\mathsf{pK}^{t^c,L}(x \mid y) \leq \mathsf{pnK}^{t,\mathcal{O}}(x \mid y) + c \cdot \log t.$$

Proof. The proof follows the same idea as Lemma 3.1. Let $k = \mathsf{pnK}^{t,\mathcal{O}}(x \mid y)$, and suppose that with probability at least $2/3$ over $r \sim \{0,1\}^t$, there exist $\Pi \in \{0,1\}^k$ such that, on input $(y; r)$ and with oracle access to $\mathcal{O}$, Π non-deterministically outputs x within time t.

Let $L_0 \in \mathsf{NP}^{\mathcal{O}}$ be the following language:

$$L_0 = \left\{ (\Pi, y, r, i, 1^t, 1^n) \;\middle|\; \begin{array}{l} \exists w \in \{0,1\}^t \text{ such that } \Pi^{\mathcal{O}}(y; w; r) \text{ outputs a string} \\ x \in \{0,1\}^n \text{ within time } t \text{ and the } i\text{-th bit of } x \text{ is } 1. \end{array} \right\}.$$

Then, with probability at least $2/3$ over $r \sim \{0,1\}^t$, given (Π, y, r, t, n) as above hard-coded, one can recover the string x within time $\mathsf{poly}(t)$ given oracle access to L_0. In particular, one can simply query $(\Pi, y, r, i, 1^t, 1^n)$ to L_0 for every $i \in [n]$ and return the concatenation of the answers from the oracle. By definition of Π and t, this process is guaranteed to output x.

Lastly, for any $\mathsf{NP}^{\mathcal{O}}$-complete language L, we may replace the L_0-oracle with an L-oracle by simulating the reduction from L_0 to L. This increases the time to produce x by at most a factor of $\mathsf{poly}(t)$.

Overall, we have

$$\begin{aligned} \mathsf{pK}^{\mathsf{poly}(t),L}(x \mid y) &\leq |(\Pi, t, n)| + O(\log t) \\ &\leq |\Pi| + O(\log t) \\ &= \mathsf{pnK}^{t,\mathcal{O}}(x \mid y)) + O(\log t), \end{aligned}$$

as desired. □

Corollary 3.16. *For every oracle $\mathcal{O}$ and every language L that is hard for $\mathsf{NP}^{\mathcal{O}}$ under deterministic c polynomial-time reductions, there exists a constant $c > 0$ such that, for all $x, y \in \{0,1\}^*$,*

$$\mathsf{pKt}^L(x \mid y) \leq c \cdot \mathsf{pnKt}^{\mathcal{O}}(x \mid y).$$

4 Unconditional Lower Bounds for Estimating Complexity

4.1 Hardness of MnKtP

In this section we prove the moreover part of Theorem 1.1, from which the main part follows directly. We first define our technical tool: rK, but with restricted length of random bits.

Definition 4.1. For any oracle $\mathcal{O}$, any time bound $t \in \mathbb{N}$, any length $r \in \mathbb{N}$, any real number $\delta \in [0,1]$, and any strings $x, y \in \{0,1\}^*$, we define $\mathsf{rK}_\delta^{(t;r),\mathcal{O}}(x \mid y)$ as

$$\mathsf{rK}_\delta^{(t;r),\mathcal{O}}(x \mid y) = \min_M \left\{ |M| \middle| \Pr_{w \sim \{0,1\}^r} \left[M^{\mathcal{O}}(y,w) = x \text{ in } t \text{ steps} \right] \geq \delta \right\}.$$

We use $\mathsf{rK}_\delta^{(t;r),\mathcal{O}_{\leq l}}(x \mid y)$ when M only make queries to $\mathcal{O}$ on strings of length at most l. We may omit the conditional string y when it is empty, simplify $(t;r)$ to t when $r \geq t$, and omit δ when $\delta = 2/3$.

4.1.1 Randomness Efficient Success Amplification for rK

In this section, we show how to amplify the success probability of rK with only 2 times randomness used.

Lemma 4.2. *For any oracle $\mathcal{O}$, any time bound $t \in \mathbb{N}$, any length $n, m, r, l \in \mathbb{N}$, any real number $\delta \in [0,1]$, and any strings $x \in \{0,1\}^n, y \in \{0,1\}^m$ satisfying $n + m \leq t$, we have*

$$\mathsf{rK}_{2/3}^{(\mathsf{poly}(t/\delta);2r),\mathcal{O}_{\leq l}}(x \mid y) \leq \mathsf{rK}_\delta^{(t;r),\mathcal{O}_{\leq l}}(x \mid y) + O(\log n/\delta).$$

Our proof is similar to [LO21], but to save randomness, we use the pairwise independent sampling trick. We then use the following string isolation lemma ([LO21, Lemma 15]), and save the correct hash value of x.

Lemma 4.3 (String Isolation Lemma [LO21]). *There is a deterministic algorithm A for which the following holds. For any set $W \subseteq \{0,1\}^n$ of size l, there exists a string $u \in \{0,1\}^{O(\log(n \cdot l))}$ such that $A(1^n, u)$ runs in $\mathsf{poly}(n)$ time and outputs a Boolean circuit that computes a function $H : \{0,1\}^n \to \{0,1\}^{O(\log l)}$ with the following property:*

$$H(w) \neq H(w') \textit{ for every distinct pair } w, w' \in W.$$

Moreover, the same guarantee is achieved by a random string u of the same length with probability at least 0.99.

Proof of Lemma 4.2. Let $s = \mathsf{rK}_\delta^{(t;r),\mathcal{O}_{\leq l}}(x \mid y)$, and let M be the corresponding program. Let A be the algorithm of Lemma 4.3. For any set $T \subseteq \{0,1\}^n$, we define p_T as

$$p_T = \Pr_{w \sim \mathcal{U}_r} \left[M^{\mathcal{O}_{\leq l}}(y,w) \text{ outputs some } z \in T \text{ in } t \text{ steps} \right].$$

And for any $\gamma \in \{0,1\}$, we define S_γ as

$$S_\gamma = \left\{ z \in \{0,1\}^n \mid p_{\{z\}} \geq \gamma \right\}.$$

One can easily see that $|S_p| \leq 1/p$. Now applying Lemma 4.3 to $S_{\delta/8}$, there exists a string $u \in \{0,1\}^{O(\log n/\delta)}$ such that $A(1^n, u)$ runs in time $\mathsf{poly}(n)$ and outputs a Boolean circuit computing a function $H : \{0,1\}^n \to \{0,1\}^{O(\log 1/\delta)}$, such that

$$H(z) \neq H(z') \text{ for every distinct pair } z, z' \in S_{\delta/8}.$$

Our description is $(M, A, u, H(x))$. Our algorithm B that recovers x from the description runs as follows:

1. Set $k = 256/\delta^2$. Sample $a, b \sim \mathcal{U}_r$, and compute $w_i = a \cdot i + b$ for $i \in [k]$, where the addition and multiplication are done in $\mathsf{GF}(2^r)$.

2. For each $i \in [k]$, we run $M^{\mathcal{O}_{\leq l}}(y, w_i)$ for t steps, and let x_i be its output. (If it does not halt within t steps or the output is not of length n, we set $x_i = \perp$.)
3. Let S' be the set of strings appearing at least $\frac{\delta}{2} \cdot k$ times (ignoring $\perp$) in $\{x_i\}_{i\in[k]}$.
4. Run $A(1^n, u)$ to obtain the circuit C. Compute $C(x')$ for every string $x' \in S'$. Output the first string x' such that $C(x')$ is equal to the stored $H(x)$. (If no such string exists, output $\perp$.)

Clearly the above algorithm runs in time $\mathsf{poly}(t/\delta)$, and only uses $2r$ random bits when sampling a, b. It remains to argue about the probability of outputting x. We will use the following claim.

Claim 4.4. *With probability at least $2/3$ over $a, b \sim \mathcal{U}_r$, we have*

1. *$x \in S'$, and*
2. *$S' \subseteq S_{\delta/8}$.*

Proof of Claim 4.4. For every set of strings $T \subseteq \{0,1\}^n$ and every $i \in [k]$, we define a random variable $V_{i,T} \in \{0,1\}$ such that $V_{i,T} = 1$ iff $x_i \in T$. By properties of finite fields, $\{w_i\}_{i\in[k]}$ (and hence $\{x_i\}_{i\in[k]}$) are identically distributed and pairwise independent. Therefore, for any fixed T, the random variables $\{V_{i,T}\}_{i\in[k]}$ are also identically distributed and pairwise independent. From our assumptions on x, we have $p_{\{x\}} \geq \delta$, so the expected value of $V_{i,\{x\}}$ is at least δ for every $i \in [k]$. By Chebyshev's inequality, we get

$$\mathbf{Pr}\left[\sum_{i=1}^{k} V_{i,\{x\}} \leq \frac{\delta}{2} \cdot k\right] \leq \frac{\mathrm{Var}[\sum_{i=1}^{k} V_{i,\{x\}}]}{(kp_{\{x\}} - \delta k/2)^2} \leq \frac{kp_{\{x\}}}{k^2 p_{\{x\}}^2/4} \leq \frac{4}{k\delta} \leq \frac{\delta}{64}. \tag{5}$$

That is, the probability of $x \notin S'$ is at most $\delta/64$. Next, we need to argue that with constant probability, S' is contained in $S_{\delta/8}$, i.e. S' does not contain any string of $\overline{S_{\delta/8}}$. First, we divide $\overline{S_{\delta/8}}$ into d disjoint buckets $B_1, \ldots, B_d$, such that $p_{B_i} \in [\delta/8, \delta/4]$ holds for every $i \in [d-1]$. This can be done by putting the strings of $\overline{S_{\delta/8}}$ into buckets one by one, and whenever the total probability of the current bucket exceeds $\delta/8$, we add a new empty bucket. Since $p_{\{x'\}} \leq \delta/8$ for any $x' \in \overline{S_{\delta/8}}$, the total probability of each bucket is in $[\delta/8, \delta/4]$ except for the last one, which might be smaller than $\delta/8$. Now for each bucket B_j, by Chebyshev's inequality, we have

$$\mathbf{Pr}\left[\sum_{i=1}^{k} V_{i,B_j} \geq \frac{\delta}{2} \cdot k\right] \leq \frac{\mathrm{Var}[\sum_{i=1}^{k} V_{i,B_j}]}{(kp_{B_j} - \delta k/2)^2} \leq \frac{kp_{B_j}}{k^2\delta^2/16} \leq \frac{4}{k\delta} \leq \frac{\delta}{64}. \tag{6}$$

Since for each $i \in [d-1]$, $p_{B_i} \geq \delta/8$ holds, we have $d \leq 8/\delta + 1 \leq 16/\delta$. Applying union bound to Equation (6), we get

$$\mathbf{Pr}\left[\exists j \in [d] \text{ s.t. } \sum_{i=1}^{k} V_{i,B_j} \geq \frac{\delta}{2} \cdot k\right] \leq \frac{\delta}{64} \cdot d \leq \frac{1}{4}.$$

Since each string x' in $\overline{S_{\delta/8}}$ belongs to some bucket B_j, if x' appears more than $\delta k/2$ times, then B_j appears more than $\delta k/2$ times. Hence we can estimate the probability of $S' \not\subseteq S_{\delta/8}$:

$$\begin{aligned}\mathbf{Pr}\left[S' \not\subseteq S_{\delta/8}\right] &= \mathbf{Pr}\left[\exists x' \in \overline{S_{\delta/8}} \text{ s.t. } \sum_{i=1}^{k} V_{i,\{x'\}} \geq \frac{\delta}{2} \cdot k\right] \\ &\leq \mathbf{Pr}\left[\exists j \in [d] \text{ s.t. } \sum_{i=1}^{k} V_{i,B_j} \geq \frac{\delta}{2} \cdot k\right] \\ &\leq \frac{1}{4}. \end{aligned} \tag{7}$$

Applying union bound to Equations (5) and (7), with probability at least $3/4 - \delta/64 \geq 2/3$ over $a, b \sim \mathcal{U}_r$, both Item 1 and Item 2 are satisfied. $\diamond$

Now with probability at least $2/3$ over $a, b \sim \mathcal{U}_r$, we have both $x \in S'$ and $S' \subseteq S_{\delta/8}$. Conditioning on this, by Lemma 4.3, $H(x) \neq H(x')$ for any $x' \in S' \setminus \{x\}$. Therefore, with probability $2/3$, the algorithm B can always find the correct x. $\square$

4.1.2 Reconstruction from Oracle Access to Dense Language

Lemma 4.5. *There exists some constant $C \in \mathbb{N}$ such that the following holds. For any $b \in \mathbb{N}$ and any language L satisfying $|L \cap \{0,1\}^n| \geq 2^n/n^b$ and any strings $x, y \in \{0,1\}^*$, let $n_1 = |x|$ and $n_2 = |y|$, and $n = n_1 + n_2$. Then for any $m_1, m_2 \in \mathbb{N}$ satisfying*

$$m_1 \leq \mathsf{rK}^{(n^{C \cdot b}; C \cdot n), L_{\leq 3n}}(x \mid y) - C \cdot \log^3(n^b),$$
$$m_2 \leq \mathsf{rK}^{(n^{C \cdot b}; C \cdot n), L_{\leq 3n}}(y) - C \cdot \log^3(n^b),$$

we define G_1, G_2 to be the PRG of Lemma 2.27, with parameter $\varepsilon = 1/2n^b$:

$$G_1 : \{0,1\}^{n_1} \times \{0,1\}^{d_1} \to \{0,1\}^{m_1},$$
$$G_2 : \{0,1\}^{n_2} \times \{0,1\}^{d_2} \to \{0,1\}^{m_2}.$$

Then there exists $z_1 \in \{0,1\}^{d_1}, z_2 \in \{0,1\}^{d_2}$ such that $G_1(x, z_1) \circ G_2(y, z_2) \in L$.

To prove this lemma, we need the following corollary regarding rK upper bound, which comes from the reconstruction procedure of Lemma 2.27.

Corollary 4.6. *For sufficiently large $n, m \in \mathbb{N}$ such that $m \leq 2n$, and any $\varepsilon \in (0,1)$, let G be the generator of Lemma 2.27. Consider any string $x \in \{0,1\}^n$, oracle $\mathcal{O}$, parameters $l, n', t \in \mathbb{N}$, string $y \in \{0,1\}^{n'}$, and algorithm $B^{(\text{-})} : \{0,1\}^m \times \{0,1\}^{n'} \times \{0,1\}^{r'} \to \{0,1\}$ running in time t. If it holds that*

$$\Pr_{\substack{z \sim \mathcal{U}_d \\ w' \sim \mathcal{U}_{r'}}} \left[B^{\mathcal{O}_{\leq l}}(G(x; z), y, w') = 1\right] - \Pr_{\substack{w \sim \mathcal{U}_m \\ w' \sim \mathcal{U}_{r'}}} \left[B^{\mathcal{O}_{\leq l}}(w, y, w') = 1\right] \geq 2\varepsilon,$$

then we have

$$\mathsf{rK}^{(\mathsf{poly}(n \cdot t/\varepsilon); O(r'+m)), \mathcal{O}_{\leq l}}(x \mid y) \leq m + |B| + O(\log^3(n/\varepsilon)),$$

where $|B|$ is the description length of B.

Proof. By Lemma 4.2, it suffices to show that $\mathsf{rK}^{(\mathsf{poly}(t/\varepsilon); O(r'+m)), \mathcal{O}_{\leq l}}_{1/\mathsf{poly}(n/\varepsilon)}(x \mid y) \leq m + |B| + O(\log^3(n/\varepsilon))$. By Markov's inequality, we have

$$\Pr_{w' \sim \mathcal{U}_{r'}} \left[\Pr_{z \sim \mathcal{U}_d} \left[B^{\mathcal{O}_{\leq l}}(G(x; z), y, w') = 1\right] - \Pr_{w \sim \mathcal{U}_m} \left[B^{\mathcal{O}_{\leq l}}(w, y, w') = 1\right] \geq \varepsilon \right] \geq \varepsilon.$$

That is, with probability at least ε over $w' \sim \mathcal{U}_{r'}$, $D_{w'}(\text{-}) := B^{\mathcal{O}_{\leq l}}(\text{-}, y, w')$ is a ε-distinguisher for G. Therefore by Lemma 2.27, we have

$$\Pr_{\substack{w' \sim \mathcal{U}_{r'} \\ w \sim \mathcal{U}_r \\ z \sim \mathcal{U}_d}} \left[R^{D_{w'}}(A(x, z), z, w) = x\right] \geq \frac{1}{\mathsf{poly}(n/\varepsilon)}.$$

By averaging, there exists some $z \in \{0,1\}^d$ such that

$$\mathop{\mathbf{Pr}}_{\substack{w' \sim \mathcal{U}_{r'} \\ w \sim \mathcal{U}_r}} \left[R^{D_{w'}}(A(x,z), z, w) = x \right] \geq \frac{1}{\mathsf{poly}(n/\varepsilon)}.$$

Therefore, we can store z, $A(x,z)$, as well as the description of R and B. Then given y and oracle access to $\mathcal{O}_{\leq l}$, by sampling $w' \sim \mathcal{U}_{r'}$ and $w \sim \mathcal{U}_r$, we can simulate $R^{D_{w'}}(A(x,z), z, w)$ in time $\mathsf{poly}(t/\varepsilon)$ and recover x with probability $1/\mathsf{poly}(n \cdot t/\varepsilon)$. Since $r = O(m)$, this gives

$$\mathsf{rK}_{1/\mathsf{poly}(n/\varepsilon)}^{(\mathsf{poly}(t/\varepsilon); O(r'+m)), \mathcal{O}_{\leq l}}(x \mid y) \leq m + |B| + O(\log^3(n/\varepsilon)).$$

Applying Lemma 4.2 completes the proof. □

Proof of Lemma 4.5. For the sake of contradiction, assume the contrary. Then we have

$$\mathop{\mathbf{Pr}}_{\substack{z_1 \sim \mathcal{U}_{d_1} \\ z_2 \sim \mathcal{U}_{d_2}}} [G_1(x; z_1) \circ G_2(y; z_2) \in L] = 0.$$

Notice that when C is a large enough constant, $m_1 \leq n_1$ and $m_2 \leq n_2$ holds. Hence by our assumption on L, we have

$$\mathop{\mathbf{Pr}}_{\substack{w_1 \sim \mathcal{U}_{m_1} \\ w_2 \sim \mathcal{U}_{m_2}}} [w_1 \circ w_2 \in L] \geq \frac{1}{(m_1 + m_2)^b} \geq \frac{1}{(n_1 + n_2)^b} = \frac{1}{n^b}.$$

Then by a hybrid argument, we either have

$$\mathop{\mathbf{Pr}}_{\substack{w_1 \sim \mathcal{U}_{m_1} \\ z_2 \sim \mathcal{U}_{d_2}}} [w_1 \circ G_2(y; z_2) \in L] - \mathop{\mathbf{Pr}}_{\substack{z_1 \sim \mathcal{U}_{d_1} \\ z_y \sim \mathcal{U}_{d_2}}} [G_1(x; z_1) \circ G_2(y; z_2) \in L] \geq \frac{1}{2n^b},$$

or

$$\mathop{\mathbf{Pr}}_{\substack{w_1 \sim \mathcal{U}_{m_1} \\ w_2 \sim \mathcal{U}_{m_2}}} [w_1 \circ w_2 \in L] - \mathop{\mathbf{Pr}}_{\substack{w_1 \sim \mathcal{U}_{m_1} \\ z_2 \sim \mathcal{U}_{d_2}}} [w_1 \circ G_2(y; z_2) \in L] \geq \frac{1}{2n^b}.$$

For the former case, given y and oracle access to L, there exists a uniform algorithm that $1/2n^b$-distinguishes $G_1(x; \mathcal{U}_{d_1})$ from $\mathcal{U}_{m_1}$ in linear time using linear randomness: given a string $s \in \{0,1\}^{m_1}$, it samples $z_2 \sim \mathcal{U}_{d_2}$, and accepts iff $s \circ G_2(y; z_2) \in L$, which can be checked with oracle access to L on length $m_1 + m_2 \leq 3n$. Hence by Corollary 4.6, we have

$$\mathsf{rK}^{(\mathsf{poly}(n^b); O(n)), L_{\leq 3n}}(x \mid y) \leq m_1 + O(\log^3(n^b)).$$

But by definition, $m_1 = \mathsf{rK}^{(n^{C \cdot b}; C \cdot n), L_{\leq 3n}}(x \mid y) - C \cdot \log^3(n^b)$, hence when C is a large enough constant, we have a contradiction.

Similarly, for the latter case, given oracle access to L, there exists a uniform algorithm that $1/2n^b$-distinguishes $G_2(y; \mathcal{U}_{d_2})$ from $\mathcal{U}_{m_2}$ in linear time using linear randomness: given a string $s \in \{0,1\}^{m_2}$, it samples $w_1 \sim \mathcal{U}_{m_1}$, and accepts iff $w_1 \circ s \in L$, which can be checked with oracle access to L on length $m_1 + m_2 \leq 3n$. Hence by Corollary 4.6, we have

$$\mathsf{rK}^{(\mathsf{poly}(n^b); O(n)), L_{\leq 3n}}(y) \leq m_2 + O(\log^3(n^b)).$$

But by definition, $m_2 = \mathsf{rK}^{(n^{C \cdot b}; C \cdot n), L_{\leq 3n}}(y) - C \cdot \log^3(n^b)$, hence when C is a large enough constant, we have a contradiction. □

4.1.3 Derandomizing rK^L

Lemma 4.7. *There exists some constant C such that the following holds. If there exists $\varepsilon, \gamma \in (0,1)$ such that a language L satisfies the following three properties:*

1. *$L \in \mathsf{NTIME}[2^{\varepsilon n}] \cap \mathsf{coNTIME}[2^{\varepsilon n}]$,*
2. *$\forall x \in L$, $\mathsf{nKt}(x) \geq \gamma \cdot |x|$,*
3. *$\forall n \in \mathbb{N}$, $L \cap \{0,1\}^n \neq \varnothing$,*

then for all large enough $n, m \in \mathbb{N}$, any $l, t, r \in \mathbb{N}$ and any strings $x \in \{0,1\}^n, y \in \{0,1\}^m$, we have

$$\mathsf{nKt}(x \mid y) \leq \mathsf{rK}^{(t;r),L_{\leq l}}(x \mid y) + C \cdot (\varepsilon \cdot (n+m+l+r+1/\gamma)/\gamma + \log^3(nmlr/\gamma) + \log t).$$

Proof of Lemma 4.7. Let $n' = C_1(n+m+l+r+\lceil \log t\rceil + \lceil 1/\gamma \rceil)$, where C_1 is some constant whose value will be determined later, and let $N' = \lceil n'/\gamma \rceil$. Let $G : \{0,1\}^{N'} \times \{0,1\}^d \to \{0,1\}^r$ be the pseudorandom generator of Lemma 2.27, with $\varepsilon = 1/6$. Let $M^{(\text{-})}$ be the program of $\mathsf{rK}^{(t;r),L_{\leq l}}(x \mid y)$, such that

$$\Pr_{w \sim \mathcal{U}_r}\left[M^{L_{\leq l}}(y,w) = x \text{ in } t \text{ steps}\right] \geq \frac{2}{3}.$$

Then we claim that given a string with high nKt complexity, we can derandomize rK^L:

Claim 4.8. *If C_1 is some large enough constant, then for any $s \in \{0,1\}^{N'}$ satisfying $\mathsf{nKt}(s) \geq n'$, it holds that*

$$\Pr_{z \sim \mathcal{U}_d}\left[M^{L_{\leq l}}(y, G(s,z)) = x \textit{ in } t \textit{ steps}\right] > \frac{1}{2}.$$

Proof of Claim 4.8. Suppose the contrary. Then it is not hard to see that the following algorithm $B^{(-)}$ with $L_{\leq l}$ oracle $1/6$-distinguishes $G(s, \mathcal{U}_d)$ from $\mathcal{U}_r$, and runs in time $\mathsf{poly}(t)$:

- B is given as input $u \in \{0,1\}^r$ along with x and y as above.
- B runs $M^{L_{\leq l}}(y,u)$ for t steps, and accepts iff it halts and outputs x.

Since B only need to store the description of $M^{(\text{-})}$, by Corollary 4.6, we have

$$\begin{aligned}\mathsf{rK}^{(\mathsf{poly}(t\cdot N');O(r)),L_{\leq l}}(s \mid x,y) &\leq r + |M^{(\text{-})}| + O(\log^3 N') + O(\log t)\\ &= r + \mathsf{rK}^{(t;r),L_{\leq l}}(x \mid y) + O(\log^3 N') + O(\log t)\\ &\leq O(r + n + \log^3 N' + \log t).\end{aligned}$$

Because we can store $x \in \{0,1\}^n, y \in \{0,1\}^m$, and a "good" choice of $O(r)$-length randomness for rK, we get that

$$\mathsf{K}^{\mathsf{poly}(t\cdot N'),L_{\leq l}}(s) \leq O(r+n+m+\log^3 N' + \log t). \tag{8}$$

Since $L \in \mathsf{NTIME}[2^{\varepsilon n}] \cap \mathsf{coNTIME}[2^{\varepsilon n}]$, we get

$$\begin{aligned}\mathsf{nKt}(s) &\leq \mathsf{nK}^{\mathsf{poly}(t\cdot N'\cdot 2^{\varepsilon l})}(s) + O(\log t + \log N' + \varepsilon l)\\ &\leq \mathsf{nK}^{\mathsf{poly}(t\cdot N'),L_{\leq l}}(s) + O(\log t + \log N' + \varepsilon l) && \text{(by Lemma 3.5)}\\ &\leq \mathsf{K}^{\mathsf{poly}(t\cdot N'),L_{\leq l}}(s) + O(\log t + \log N' + \varepsilon l)\\ &\leq O(n+m+l+r+\log t + \log^3 N'). && \text{(by Equation (8))}\end{aligned}$$

Notice that

$$\begin{aligned}\log^3 N' &\le O(\log^3 C_1 + \log^3(n+m+l+r+\log t + 1/\gamma) + \log^3(1/\gamma)) \\ &\le O(C_1 + n + m + l + r + \log t + 1/\gamma).\end{aligned}$$

Hence we have $\mathsf{nKt}(s) \le O(C_1 + n + m + l + r + \log t + 1/\gamma)$. But by our assumption, $\mathsf{nKt}(s) \ge n' = C_1 \cdot (n + m + l + r + \lceil \log t \rceil + \lceil 1/\gamma \rceil)$. Hence when C_1 is some large enough constant, we get a contradiction. $\diamond$

Now let C_1 be a large enough constant such that Claim 4.8 holds. We define a non-deterministic algorithm $A^{(\cdot)}$ such that, when given the description of $M^{(\cdot)}$ and y, as well as oracle access to $L_{\le N'}$, recovers x. $A^{L_{\le N'}}$ works as follows:

1. $A^{L_{\le N'}}$ non-deterministically guesses a string $s \in \{0,1\}^{N'}$.

2. $A^{L_{\le N'}}$ checks if $s \in L$, using its oracle access to $L_{\le N'}$. If $s \notin L$, then it outputs $\perp$ and halts.

3. $A^{L_{\le N'}}$ enumerates over every $z \in \{0,1\}^d$, and simulates $M^{L_{\le l}}(y, G(s,z))$ for t steps. (Because $l \le N'$, $A^{L_{\le N'}}$ can answer the oracle queries by $M^{L_{\le l}}$.) Let $\{x_z\}_{z\in\{0,1\}^d}$ be the output of the simulations. If there exists some string $\hat{x}$ appearing more than 2^{d-1} times in $\{x_z\}_{z\in\{0,1\}^d}$, then $A^{L_{\le N'}}$ outputs $\hat{x}$; otherwise it outputs $\perp$.

By Claim 4.8, whenever $\mathsf{nKt}(s) \ge n'$, the majority of $\{x_z\}_{z\in\{0,1\}^d}$ is x. Since for any $s \in L \cap \{0,1\}^{N'}$, $\mathsf{nKt}(s) \ge \gamma \cdot N' \ge n'$, for any non-deterministic guess of s, A always outputs either x or $\perp$. Because $L \cap \{0,1\}^{N'} \ne \varnothing$ for any N', there always exists some non-deterministic guess of s such that A outputs x. Hence by the definition of nK, we have

$$\mathsf{nK}^{\mathsf{poly}(2^d \cdot t \cdot N'), L_{\le N'}}(x \mid y) \le \mathsf{rK}^{L_{\le l},(t;r)}(x \mid y) + O(\log(t \cdot N')). \tag{9}$$

Therefore we have

$$\begin{aligned}\mathsf{nKt}(x \mid y) &\le \mathsf{nK}^{\mathsf{poly}(2^{\varepsilon N'} \cdot 2^d \cdot t)}(x \mid y) + O(\varepsilon N' + d + \log t) \\ &\le \mathsf{nK}^{\mathsf{poly}(2^d \cdot t \cdot N'), L_{\le N'}}(x \mid y) + O(\varepsilon N' + d + \log t) && \text{(by Lemma 3.5)} \\ &\le \mathsf{rK}^{(t;r), L_{\le l}}(x \mid y) + O(\varepsilon N' + d + \log t) && \text{(by Equation (9))} \\ &\le \mathsf{rK}^{(t;r), L_{\le l}}(x \mid y) + O(\varepsilon N' + \log^3 N' + \log t) && \text{(Since } d = O(\log^3 N'))\end{aligned}$$

Replacing N' by $\lceil C_1(n + m + l + r + \lceil \log t \rceil + \lceil 1/\gamma \rceil)/\gamma \rceil$ finishes the proof. □

4.1.4 Iterative Construction of High-Complexity Strings

Combining Lemma 4.5 and Lemma 4.7, we have the following lemma, which we will use to prove Theorem 1.1.

Lemma 4.9. *There exists some constant $C \in \mathbb{N}$ such that the following holds. If there exists $\varepsilon, \gamma \in (0,1)$ and $b \in \mathbb{N}$ such that a language L satisfies the following three properties*:

1. *$L \in \mathsf{NTIME}[2^{\varepsilon n}] \cap \mathsf{coNTIME}[2^{\varepsilon n}]$,*

2. *$\forall x \in L$, $\mathsf{nKt}(x) \ge \gamma \cdot |x|$,*

3. *$\forall n \in \mathbb{N}$, $|L \cap \{0,1\}^n| \ge 2^n / n^b$,*

and for any strings $x, y \in \{0,1\}^*$, let $n_1 = |x|$ and $n_2 = |y|$, and $n = n_1 + n_2$. Then for any $m_1, m_2 \in \mathbb{N}$ satisfying

$$m_1 \le \mathsf{nKt}(x \mid y) - C \cdot (\varepsilon \cdot (n + 1/\gamma)/\gamma + \log^3(n^b/\gamma))$$
$$m_2 \le \mathsf{nKt}(y) - C \cdot (\varepsilon \cdot (n + 1/\gamma)/\gamma + \log^3(n^b/\gamma))$$

We define G_1, G_2 to be the PRG of Lemma 2.27, with parameter $\varepsilon = 1/2n^b$:

$$G_1 : \{0,1\}^{n_1} \times \{0,1\}^{d_1} \to \{0,1\}^{m_1},$$
$$G_2 : \{0,1\}^{n_2} \times \{0,1\}^{d_2} \to \{0,1\}^{m_2}.$$

Then there exists $z_1 \in \{0,1\}^{d_1}, z_2 \in \{0,1\}^{d_2}$ such that $G_1(x, z_1) \circ G_2(y, z_2) \in L$.

Proof. Follows from Lemma 4.5 and Lemma 4.7. □

Proof of Theorem 1.1. We prove the "moreover" part of this theorem, of which the main part is an immediate corollary. For the sake of contradiction, assume that there exists some language L satisfying the following three properties:

1. $L \in \mathsf{NTIME}[2^{\varepsilon n}] \cap \mathsf{coNTIME}[2^{\varepsilon n}]$ for any $\varepsilon > 0$,
2. $\exists \gamma \in (1/2, 1)$ such that $\forall x \in L$, $\mathsf{nKt}(x) \ge \gamma \cdot |x|$,
3. $\exists b \in \mathbb{N}$ such that $\forall n \in \mathbb{N}$, $|L \cap \{0,1\}^n| \ge 2^n/n^b$.

Then we use the following algorithm to explicitly construct a long string $s \in L$ in short deterministic time with oracle access to L.

Algorithm 1 Iterative construction of long string in L

```
procedure $A^L(\gamma, 1^n)$
    $s_0 := \bot, N_1 := n, \theta_1 := \gamma n$.
    for $s' \in \{0,1\}^n$ by lexicographical order do
        if $s' \in L$ then
            $s_1 \leftarrow s'$.
            break
    $\gamma' := (\gamma + 1/2)/2$.
    for $i = 2, 3, \cdots$ do
        $m_{i,1} := \lceil \gamma' n/\gamma \rceil, m_{i,2} := \lceil \gamma' \theta_{i-1}/\gamma \rceil$.
        $N_i := m_{i,1} + m_{i,2}, \varepsilon_i := (n + N_{i-1})^{-b}/2$.
        $\theta_i := \gamma N_i$.
        Instantiate $G_1 : \{0,1\}^n \times \{0,1\}^{d_1} \to \{0,1\}^{m_{i,1}}$ in Lemma 2.27 with security parameter $\varepsilon_i$.
        Instantiate $G_2 : \{0,1\}^{N_{i-1}} \times \{0,1\}^{d_2} \to \{0,1\}^{m_{i,2}}$ in Lemma 2.27 with security parameter $\varepsilon_i$.
        $s_i := \bot$.
        for $x \in \{0,1\}^n, z_1 \in \{0,1\}^{d_1}, z_2 \in \{0,1\}^{d_2}$ by lexicographical order do
            if $G_1(x; z_1) \circ G_2(s_{i-1}; z_2) \in L$ then
                $s_i := G_1(x; z_1) \circ G_2(s_{i-1}; z_2)$.
                break
        if $\theta_i > n/(2 - 2\gamma')$ then
            Output $s_i$.
            return
    Output $\bot$.
```

Then we have the following claim for the above procedure A^L:

Claim 4.10. *For a fixed γ, if ε is small enough, then for all large enough n, there exists some integer $i_0 \leq \log_{\gamma'}(1 - \frac{1}{2\gamma'}) + 2$ such that*

1. *For any $i \leq i_0$, $s_i \in L$, and*
2. *$\theta_{i_0} > n/(2 - 2\gamma')$.*

Proof of Claim 4.10. We need to prove two things: first, for every i during the iteration, we can find some s_i s.t. $s_i \neq \perp$, i.e. $s_i \in L$; second, our algorithm terminates at some i_0, i.e. $\theta_{i_0} > n/(2 - 2\gamma')$.

To prove the first argument, we use induction on i. For $i = 1$, because $L \cap \{0,1\}^n \neq \varnothing$, we can always find $s' \in L$, so $s_1 \neq \perp$. Then suppose $s_{i-1} \neq \perp$ for some $i \geq 2$. Since $\gamma > 1/2$, by Lemma 4.9, there exists some universal constant C such that whenever $m_{i,1}$ and $m_{i,2}$ satisfy

$$m_{i,1} \leq \mathsf{nKt}(x \mid s_{i-1}) - C \cdot (\varepsilon(n + N_{i-1}) + \log^3((n + N_{i-1})^b))$$
$$m_{i,2} \leq \mathsf{nKt}(s_{i-1}) - C \cdot (\varepsilon(n + N_{i-1}) + \log^3((n + N_{i-1})^b),$$

there exists some $z_1 \in \{0,1\}^{d_1}, z_2 \in \{0,1\}^{d_2}$ such that

$$G_1(x, z_1) \circ G_2(s_{i-1}, z_2) \in L.$$

By inductive hypothesis, $s_{i-1} \in L$, hence $\mathsf{nKt}(s_{i-1}) \geq \gamma|s_{i-1}| = \theta_{i-1}$. By a counting argument (Lemma 2.21), there also exists some $x \in \{0,1\}^n$ such that $\mathsf{nKt}(x \mid s_{i-1}) \geq n$. Therefore, it suffices to show that

$$m_{i,1} = \lceil \gamma' n/\gamma \rceil \leq n - C \cdot (\varepsilon(n + N_{i-1}) + \log^3((n + N_{i-1})^b))$$
$$m_{i,2} = \lceil \gamma' \theta_{i-1}/\gamma \rceil \leq \theta_{i-1} - C \cdot (\varepsilon(n + N_{i-1}) + \log^3((n + N_{i-1})^b).$$

Since we did not terminate at $i-1$, we are guaranteed that $\theta_{i-1} \leq n/(2-2\gamma')$, therefore $N_{i-1} \leq \frac{n}{\gamma \cdot (2-2\gamma')} \leq 4n$ (By definition $\gamma \geq 1/2$ and $\gamma' \leq 3/4$). Since $\gamma'/\gamma < 1$, if ε is small enough (i.e. $\varepsilon < \frac{\gamma - \gamma'}{6C\gamma}$), then for all large enough n, the above two inequalities holds, and we can find $z_1 \in \{0,1\}^{d_1}, z_2 \in \{0,1\}^{d_2}$ such that $G_1(x, z_1) \circ G_2(x, z_2) \in L$, in other words, s_i exists.

To prove the second argument, notice that $|s_1| = N_1 = n$, and $N_i \geq \gamma' \cdot (N_{i-1} + n/\gamma)$. Expanding the recursion, we get

$$N_i \geq \left((\gamma')^1 + \cdots + (\gamma')^{i-1}\right) \cdot \frac{n}{\gamma} + (\gamma')^{i-1} \cdot n = \left(\frac{\gamma'}{\gamma} \cdot \frac{1 - (\gamma')^{i-1}}{1 - \gamma'} + (\gamma')^{i-1}\right) \cdot n.$$

Hence we have

$$\theta_i \geq \frac{\gamma' \cdot (1 - (\gamma')^{i-1})}{1 - \gamma'} \cdot n$$

When $i - 1 \geq \log_{\gamma'}(1 - \frac{1}{2\gamma'})$, we have $\theta_i \geq \frac{n}{2-2\gamma'}$. Hence the algorithm terminates when $i \leq \log_{\gamma'}(1 - \frac{1}{2\gamma'}) + 2$, i.e. $i_0 \leq \log_{\gamma'}(1 - \frac{1}{2\gamma'}) + 2$. ◇

The running time of algorithm $A^L(\gamma, 1^n)$ is bounded by $2^n \cdot \mathsf{poly}(n) \cdot 2^{O(\log^3 n)} \cdot i_0$, and we only need to store n and γ. Without loss of generality, we may assume that $\gamma = 1/2 + 1/2^k$ for some $k \in \mathbb{N}$, and we can store k, which takes $\lceil \log\log(\frac{1}{\gamma - 1/2}) \rceil$ bits. Also since $N_{i_0 - 1} \leq 4n$, we have $N_{i_0} \leq 5n$, so A only need to query L on inputs of length at most $5n$. Therefore we have

$$\mathsf{K}^{2^n \cdot \mathsf{poly}(n) \cdot 2^{O(\log^3 n)}, \cdot i_0, L_{\leq 5n}}(s_{i_0}) \leq O\left(\log n + \log\log\left(\frac{1}{\gamma - 1/2}\right)\right). \tag{10}$$

Then we have

$$\begin{aligned}
\mathsf{nKt}(s_{i_0}) &\le \mathsf{nK}^{2^n \cdot \mathsf{poly}(n) \cdot 2^{O(\log^3 n)} \cdot i_0 \cdot 2^{O(\varepsilon n)}}(s_{i_0}) + n + O(\varepsilon n + \log^3 n + \log i_0) \\
&\le \mathsf{K}^{2^n \cdot \mathsf{poly}(n) \cdot 2^{O(\log^3 n)} \cdot i_0, L_{\le 5n}}(s_{i_0}) + n + O(\varepsilon n + \log^3 n + \log i_0) && \text{(by Lemma 3.5)} \\
&\le n + O\left(\varepsilon n + \log^3 n + \log i_0 + \log\log\left(\frac{1}{\gamma - 1/2}\right)\right) && \text{(by Equation (10))} \\
&\le n + O\left(\varepsilon n + \log^3 n + \log\log\left(\frac{1}{\gamma - 1/2}\right)\right). && (11)
\end{aligned}$$

Where the last inequality is true because $i_0 \le O(\log \frac{1}{\gamma - 1/2})$, which follows from $i_0 \le \log_{\gamma'}(1 - \frac{1}{2\gamma'}) + 2$ (Claim 4.10), $\gamma' = (\gamma + 1/2)/2$ and $1/2 < \gamma < 1$. However, on the other hand, we know that $s_{i_0} \in L$, hence we get

$$\mathsf{nKt}(s_{i_0}) \ge \gamma \cdot |s_{i_0}| = \theta_{i_0} \ge n/(2 - 2\gamma'). \tag{12}$$

Since $1/(2 - 2\gamma') > 1$ and $i_0 \le \log_{\gamma'}(1 - \frac{1}{2\gamma'}) + 2$, when ε is small enough (i.e. $\varepsilon < \frac{2\gamma' - 1}{2C_1(2 - 2\gamma')}$, where C_1 is the constant hidden by the big O of Equation (11)), for all large enough n, Equations (11) and (12) contradicts each other. □

4.1.5 Limitations of the Iterative Construction for $\gamma \le 1/2$

We believe the statement is also true for $0 < \gamma \le 1/2$. However, our technique only works for $\gamma > 1/2$. In order to see this, recall that we iteratively construct a string s in time $2^{(1+o(1)) \cdot n}$ such that $\mathsf{nKt}(s) \ge (1 + \Omega(1)) \cdot n$, resulting in a contradiction. But by the iterative construction, we are only guaranteed that $\mathsf{nKt}(s) \ge (\gamma + \gamma^2 + \gamma^3 + \cdots) \cdot n - o(n)$, which is upper bounded by $\frac{\gamma}{1-\gamma} \cdot n - o(n)$. Hence when $\gamma \le 1/2$, $\frac{\gamma}{1-\gamma} \le 1$, and we don't have $\mathsf{nKt}(s) \ge (1 + \Omega(1)) \cdot n$, so the proof does not work here.

A more intuitive explanation to this barrier arises from the SoI intuition behind this technical iterative construction (see Section 1.3). In the setting of a dense language $L \subseteq \mathcal{R}_{\mathsf{nKt}[\gamma \cdot n]}$, we are able to prove that

$$\mathsf{nKt}(x, y) \ge \gamma(\mathsf{nKt}(x \mid y) + \mathsf{nKt}(y)) - C\varepsilon n$$

by reconstruction (Lemma 4.5) and derandomization (Lemma 4.7). This degrades to a trivial inequality when $\gamma \le 1/2$, since

$$\mathsf{nKt}(x, y) \ge \max\{\mathsf{nKt}(x), \mathsf{nKt}(y)\} \ge \max\{\mathsf{nKt}(x \mid y), \mathsf{nKt}(y)\} \ge \frac{1}{2}(\mathsf{nKt}(x \mid y) + \mathsf{nKt}(y))$$

holds unconditionally. Thus the derived SoI inequality does not seem helpful. On the other hand, the statement is non-trivial when $\gamma > 1/2$, and it is strong enough for us to derive a contradiction from it.

4.2 Hardness of Gap-MrnKtP

In this section, we prove the "moreover" parts in Theorem 1.2. These parts are strengthened versions of the plain complexity lower bounds for Gap-MrnKtP in Theorem 1.2. To see this, we observe the following simple lemma.

Lemma 4.11. *Let $\theta : \mathbb{N} \to \mathbb{N}$ be such that $\theta(n) < n - 1$ and $\mathcal{C}$ be a complexity class. Suppose for every dense language $L \subseteq \mathcal{R}_{\mathsf{rnKt}[\theta(n)]}$, it is the case that $L \notin \mathcal{C}$. Then* Gap-MrKtP$[\theta(n), n-1] \notin$ co-$\mathcal{C}$.

Proof. We prove the contrapositive. Suppose $\mathsf{Gap\text{-}MrKtP}[\theta(n), n-1] \in \mathsf{co}\text{-}\mathcal{C}$. Then there exists a language $L_0 \in \mathsf{co}\text{-}\mathcal{C}$ that is consistent with $\mathsf{Gap\text{-}MrKtP}[\theta(n), n-1]$, in the sense that for every $x \in \{0,1\}^n$: if $\mathsf{rnKt}(x) \leq \theta(n)$, then $x \in L_0$, and if $\mathsf{rnKt}(x) \geq n-1$, then $x \notin L_0$.

Let L be the complement of L_0. Note that $L \in \mathcal{C}$. Moreover, by construction, $L \subseteq \mathcal{R}_{\mathsf{rnKt}[\theta(n)]}$. Finally, note that L is dense, since L_n contains the set $\{x \mid \mathsf{rnKt}(x) \geq n-1\}$, whose size is at least $2^n/2$ by Lemma 2.21. □

4.2.1 Exponential $\mathsf{AMTIME} \cap \mathsf{coAMTIME}$ Lower Bound via Iterative Construction

In this subsection, we give proof to Theorem 1.2, Item 1. The plan is similar to the proof of Theorem 1.1, where we break reconstructive black-box PRG (See Lemma 2.27) under easiness assumption of L, then iteratively construct a long string in L to derive contradiction. It shares same technical barrier $\gamma > 1/2$ with Theorem 1.1, as discussed in Section 4.1.5.

Lemma 4.12. *Fix any $c > 0, 1/2 < \gamma < 1$, any $L \subseteq \mathcal{R}_{\mathsf{rnKt}[\gamma \cdot n]}$ satisfying $|L_n| \geq 2^n/n^c$, there exist constant $C_1, C_2 > 0$ establishing the following.*

For all sufficiently large $n_1, n_2 \in \mathbb{N}$ and any string $x \in \{0,1\}^{n_1}, y \in \{0,1\}^{n_2}$, we fix any $m_1, m_2 \in \mathbb{Z}$ such that

$$m_1 \leq \mathsf{rK}^{(n_1+n_2)^{C_1}, L_{\leq 2n_1+2n_2}}(x \mid y) - C_2 \log^3(n_1+n_2),$$
$$m_2 \leq \mathsf{rK}^{(n_1+n_2)^{C_1}, L_{\leq 2n_1+2n_2}}(y) - C_2 \log^3(n_1+n_2).$$

Let $m := m_1 + m_2$. If $m_1, m_2 > 0$, for any pseudorandom generators G_1, G_2 instantiated from Lemma 2.27 with parameters $\varepsilon := m^{-c-1}/2$ and

$$G_1 : \{0,1\}^{n_1} \times \{0,1\}^{d_1} \to \{0,1\}^{m_1},$$
$$G_2 : \{0,1\}^{n_2} \times \{0,1\}^{d_2} \to \{0,1\}^{m_2},$$

there exist $z_1 \in \{0,1\}^{d_1}, z_2 \in \{0,1\}^{d_2}$ such that

$$G_1(x; z_1) \circ G_2(y; z_2) \in L.$$

Proof. For the sake of contradiction, assume the contrary, i.e.,

$$\Pr_{\substack{z_1 \sim \mathcal{U}_{d_1} \\ z_2 \sim \mathcal{U}_{d_2}}} [G_1(x; z_1) \circ G_2(y; z_2) \in L] = 0.$$

By density assumption on L, we have

$$\Pr_{\substack{w_1 \sim \mathcal{U}_{m_1} \\ w_2 \sim \mathcal{U}_{m_2}}} [w_1 \circ w_2 \in L] \geq \frac{1}{m^c}.$$

We consider two cases: either

$$\Pr_{\substack{w_1 \sim \mathcal{U}_{m_1} \\ z_2 \sim \mathcal{U}_{d_2}}} [w_1 \circ G_2(y; z_2) \in L] \leq \frac{1}{2m^c},$$

or

$$\Pr_{\substack{w_1 \sim \mathcal{U}_{m_1} \\ z_2 \sim \mathcal{U}_{d_2}}} [w_1 \circ G_2(y; z_2) \in L] \geq \frac{1}{2m^c}.$$

For the former case, we design D^L as follow: Given a string $s \in \{0,1\}^{m_2}$, it samples $w_1 \sim \mathcal{U}_{m_1}$ and returns whether $w_1 \circ s \in L$. Note that $m_1 \le 2n_1, m_2 \le 2n_2$, we have query length here bounded by $2n_1 + 2n_2$, as well as $m \le \mathsf{poly}(n_1+n_2), \varepsilon^{-1} \le \mathsf{poly}(n_1+n_2)$.

We can easily see that D^L ε-distinguishes $G_2(y;\mathcal{U}_{d_2})$ from $\mathcal{U}_{m_2}$ in $O(m)$ time. Applying Corollary 4.6 on G_2, we obtain

$$\mathsf{rK}^{\mathsf{poly}(m/\varepsilon),L_{\le 2n_1+2n_2}}(y) \le m_2 + O(\log^3(n_2/\varepsilon)) = m_2 + O(\log^3(n_1+n_2)).$$

But by definition, $m_2 = \mathsf{rK}^{(n_1+n_2)^{C_1},L_{\le 2n_1+2n_2}}(x \mid y) - C_2 \log^3(n_1+n_2)$, hence when C_1, C_2 are large enough constants, we have a contradiction.

Similarly, for the later case, we design D^L which ε-distinguishes $G_1(x;\mathcal{U}_{d_1})$ from $\mathcal{U}_{m_1}$ in polynomial time: On $s \in \{0,1\}^{m_1}$, it samples $z_2 \sim \mathcal{U}_{d_2}$, returns whether $s \circ G_2(y;z_2) \in L$. Hence by Corollary 4.6, we have

$$\mathsf{rK}^{\mathsf{poly}(m/\varepsilon),L_{\le 2n_1+2n_2}}(x \mid y) \le m_1 + O(\log^3(n_1+n_2)).$$

But by definition, $m_2 = \mathsf{rK}^{(n_1+n_2)^{C_1},L_{\le 2n_1+2n_2}}(x \mid y) - C_2 \log^3(n_1+n_2)$, we then obtain a contradiction when C_1, C_2 are large enough. □

We aim to derive contradiction from Lemma 4.12, as in the proof of Theorem 1.1, which finishes the proof of Theorem 1.2, Item 1.

Proof of Theorem 1.2, Item 1. Fix $c > 0, 1/2 < \gamma < 1$ and $L \subseteq \mathcal{R}_{\mathsf{rnKt}[\gamma\cdot n]}$ such that $|L_n| \ge 2^n/n^c$. For sake of contradiction we assume that

$$L \in \mathsf{AMTIME}[2^{\varepsilon n}] \cap \mathsf{coAMTIME}[2^{\varepsilon n}]$$

for any $\varepsilon > 0$. Similar to Lemma 4.9, we use the following algorithm to explicitly construct a long string $s \in L$ in short deterministic time with oracle access to L.

Algorithm 2 Iterative construction of long string in L

```
procedure $A^L(\gamma, 1^n)$
    $s_0 := \bot, N_1 := n, \theta_1 := \gamma n$.
    for $s' \in \{0,1\}^n$ by lexicographical order do
        if $s' \in L$ then
            $s_1 \leftarrow s'$.
            break
    $\gamma' := (\gamma + 1/2)/2$.
    for $i = 2, 3, \cdots$ do
        $m_{i,1} := \lceil \gamma' n/\gamma \rceil, m_{i,2} := \lceil \gamma' \theta_{i-1}/\gamma \rceil$.
        $N_i := m_{i,1} + m_{i,2}, \varepsilon_i := N_i^{-c-1}/2$.
        $\theta_i := \gamma N_i$.
        Instantiate $G_1 : \{0,1\}^n \times \{0,1\}^{d_1} \to \{0,1\}^{m_{i,1}}$ in Lemma 2.27 with security parameter $\varepsilon_i$.
        Instantiate $G_2 : \{0,1\}^{N_{i-1}} \times \{0,1\}^{d_2} \to \{0,1\}^{m_{i,2}}$ in Lemma 2.27 with security parameter $\varepsilon_i$.
        $s_i := \bot$.
        for $x \in \{0,1\}^n, z_1 \in \{0,1\}^{d_1}, z_2 \in \{0,1\}^{d_2}$ by lexicographical order do
            if $G_1(x;z_1) \circ G_2(s_{i-1};z_2) \in L$ then
                $s_i := G_1(x;z_1) \circ G_2(s_{i-1};z_2)$.
                break
        if $\theta_i > n/(2 - 2\gamma')$ then
            Output $s_i$.
            **return**
    Output $\bot$.
```

We hereby analyse behaviour of A^L.

Claim 4.13. *For any $1/2 < \gamma < 1$, there exists $i^* \in \mathbb{N}$ such that for all sufficiently large $n \in \mathbb{N}$, $A^L(\gamma, 1^n)$ outputs in $O_\gamma(2^{n+O(\log^3 n)})$ time the string $s_{i^*} \neq \bot$ with $\mathsf{rnKt}(s) > n/(2-2\gamma')$. Here, the constant factor hidden in big-O notation depends on γ, but can be fixed when γ is fixed.*

To prove Claim 4.13, we prove the following crucial statement.

Claim 4.14. *Fix any $1/2 < \gamma < 1$, for all sufficiently large $n \in \mathbb{N}$, the following holds for all $i \in \mathbb{N}$: On input $(\gamma, 1^n)$, either $s_i \neq \bot$ is produced successfully, or the algorithm terminates with output s_j for some $j < i$.*

Proof of Claim 4.14. We prove by induction. For $i = 1$, since L is dense, it is guaranteed that $L_n \neq \emptyset$, and s_1 is the lexicographically smallest string in L_n.

Now assume for $i \geq 2$, we have $s_i \in L$ and $\theta_i \leq n/(2-2\gamma')$. We aim to prove that algorithm successfully finds $s_{i+1} \in L$. Since we are considering sufficiently large n, we omit some detailed discussion on inequalities when they hold asymptotically with respect to n and N_i.

By induction assumption, $\mathsf{rnKt}(s_i) \geq \gamma N_i = \theta_i$. Let $C_1, C_2 > 0$ be the constants from Lemma 4.12. Applying Lemma 3.11, there exists $c' > 0$ such that

$$\begin{aligned}\mathsf{rnKt}(s_i) &\leq \mathsf{rnK}^{O((n+N_i)^{C_1+1})\cdot 2^{c'\varepsilon(4n+4N_i)}}(s_i) + c'\varepsilon(4n+4N_i) + O(\log(n+N_i)) \\ &\leq \mathsf{rK}^{(n+N_i)^{C_1}, L_{\leq 2n+2N_i}}(s_i) + c'\varepsilon(4n+4N_i) + O(\log(n+N_i))\end{aligned}$$

holds for any $\varepsilon > 0$. Therefore,

$$\begin{aligned}\mathsf{rK}^{(n+N_i)^{C_1}, L_{\leq 4n+4N_i}}(s_i) &\geq \mathsf{rnKt}(s_i) - c'\varepsilon(4n+4N_i) - O(\log(n+N_i)) \\ &\geq \theta_i - c'\varepsilon(4n+4N_i) - O(\log(n+N_i))\end{aligned}$$

holds for any $\varepsilon > 0$. Note that, since $\gamma > 1/2$, $\gamma'/\gamma = 1/2 + 1/4\gamma < 1$, picking small enough $\varepsilon > 0$ gives

$$m_{i+1,2} = \frac{\gamma'}{\gamma}\theta_i \leq \mathsf{rK}^{(n+N_i)^{C_1}, L_{\leq 2n+2N_i}}(s_i) - C_2 \log^3(n+N_i). \tag{13}$$

By standard counting argument of rK, there exists $x \in \{0,1\}^n$ such that

$$\mathsf{rK}^{(n+N_i)^{C_1}, L_{\leq 2n+2N_i}}(x \mid s_i) \geq n,$$

so there exists $x \in \{0,1\}^n$ satisfying

$$m_{i+1,1} = \frac{\gamma'}{\gamma} n \leq n - C_2 \log^3(n+N_i) \leq \mathsf{rK}^{(n+N_i)^{C_1}, L_{\leq 2n+2N_i}}(x \mid s_i) - C_2 \log^3(n+N_i). \tag{14}$$

Combining Equation (14) and Equation (13), we apply Lemma 4.12, which indicates for any sufficiently large n, $A^L(\gamma, 1^n)$ successfully produces $s_{i+1} \in L$. $\diamond$

Proof of Claim 4.13. Fix any $1/2 < \gamma < 1$. For any $i \geq 2$, observe that if $A^L(\gamma, 1^n)$ produces $s_i \neq \bot$, we will have

$$\theta_i = \gamma' n + \gamma' \theta_{i-1},$$

and $\theta_1 = \gamma n$ holds. Simple telescoping gives

$$\theta_i = \left(\sum_{j=1}^{i-1} \gamma'^j + \gamma'^{i-1}\gamma \right) n.$$

Applying $\gamma' < \gamma < 1$, we have

$$\theta_i > \left(\sum_{j=1}^{i} \gamma'^j \right) n = \frac{\gamma'(1-\gamma'^i)n}{1-\gamma'}, \quad \theta_i < \left(\sum_{j=1}^{i} \gamma^j \right) n < \frac{\gamma n}{1-\gamma}, \tag{15}$$

holding for every i. Note that $\gamma' = (1+2\gamma)/4 > 1/2$, we always have $\gamma'/(1-\gamma') > 1/(2-2\gamma')$, indicating that there exists I_γ such that whenever $i \geq I_\gamma$, $\theta_i > n/(2-2\gamma')$. Combining this with Claim 4.14 we can derive that $A^L(\gamma, 1^n)$ terminates with output $s_{i^*} \in L$ for all sufficiently large n where $i^* \leq I_\gamma$. Given $L \subseteq \mathcal{R}_{\mathsf{rnKt}[\gamma \cdot n]}$, we have

$$\mathsf{rnKt}(s_{i^*}) \geq \gamma |s_{i^*}| = \theta_{i^*} > n/(2-2\gamma').$$

The running time is bounded by

$$\begin{aligned} O(2^n) + \sum_{i=2}^{i^*} O(2^{n+O(\log^3 n)+O(\log^3 N_{i-1})}) &= O(2^n) + \sum_{i=2}^{i^*} O(2^{n+O(\log^3 n)+O(\log^3(\theta_{i-1}/\gamma))}) \\ &\leq O(2^n) + I_\gamma \cdot O(2^{n+O(\log^3 n)+O(\log^3(n/(1-\gamma)))}) \end{aligned}$$

Note that I_γ is independent from n, the running time is therefore bounded by $O_\gamma(2^{n+O(\log^3 n)})$. ◇

Fix any $1/2 < \gamma < 1$. For all sufficiently large n, by Claim 4.13, we compute $s_{i^*} := A^L(\gamma, 1^n)$. we have $\mathsf{rnKt}(s_{i^*}) > n/(2-2\gamma')$; On the other hand, the running time bound given in Claim 4.13 indicates

$$\mathsf{rnK}^{t(n), L_{\leq \max\{N_1, N_2, \cdots, N_{i^*}\}}}(s_{i^*}) = O(1),$$

where $t(n) = O(2^{n+O(\log^3 n)})$. By Equation (15), we have

$$\max\{N_1, N_2, \cdots, N_{i^*}\} = \frac{1}{\gamma} \max\{\theta_1, \theta_2, \cdots, \theta_{i^*}\} < \frac{n}{1-\gamma}.$$

Therefore,

$$\mathsf{rnK}^{t(n), L_{\leq n/(1-\gamma)}}(s_{i^*}) = O(1).$$

Applying Lemma 3.11, we have

$$\begin{aligned} \mathsf{rnKt}(s_{i^*}) &\leq \mathsf{rnK}^{O(t(n)\log t(n)) \cdot \mathsf{poly}(2^{2\varepsilon n/(1-\gamma)})} + \frac{\varepsilon n}{1-\gamma} + \log t(n) + O(\log\log t(n)) \\ &\leq \mathsf{rnK}^{t(n), L_{\leq n/(1-\gamma)}}(s_{i^*}) + \frac{\varepsilon n}{1-\gamma} + \log t(n) + O(\log\log t(n)) + O(\log n). \end{aligned}$$

Substituting $t(n) = O(2^{n+O(\log^3 n)})$ gives $\mathsf{rnKt}(s_{i^*}) \leq n + O(\log^3 n)$, which contradicts $\mathsf{rnKt}(s_{i^*}) > n/(2-2\gamma')$ since $\gamma' > 1/2$. □

4.2.2 AMTIME Lower Bound via Explicit Constructions

We formally state the technical result from [CLL24], which is mentioned in Section 1.3. We begin with a formal definition of *single-valued Arthur-Merlin protocol*, which is a natural interpretation from [CLL24].

Definition 4.15. A *non-binary public coin protocol* (P, V) is defined as follow: On any input x, V, P interact by constantly many rounds; In the k-th round, V (the verifier) sends plain uniform randomness $r_k \in \{0,1\}^{n_k}$ to P; P (the prover) then returns w_k according to $x, r_1, w_1, \cdots, r_{k-1}, w_{k-1}, r_k$. After the last round, V outputs some string y as answer, which deterministically depends on the whole interaction transcript (all w_i, r_i's) and input x.

Non-binary public coin protocol (P, V) is a *single-valued Arthur-Merlin* protocol computing $\{x_n\}_{n\in\mathbb{N}}$, if there exists $l : \mathbb{N} \to \mathbb{N}$ such that the following holds for every $n \in \mathbb{N}$:

- **Conformity:** There exists $\alpha \in \{0,1\}^{l(n)}$ such that $(P, V)(1^n, \alpha_n) = x_n$ holds with probability 1 over randomness of V;
- **Resiliency:** For every $\zeta_n \in \{0,1\}^{l(n)}$, there exists $y_n \in \{0,1\}^n$ such that, for every (possibly with unbounded computational power) prover P', $(P', V)(1^n, \zeta_n) \in \{y_n, \perp\}$ with probability at least $2/3$ over randomness of V.

We refer to $l(n)$ as the *advice complexity* of (P, V).

Then we present below the main result from [CLL24] regarding explicit algorithms that hits any dense language in coAM.

Lemma 4.16 (See [CLL24], Theorem 1.3)**.** *For every dense language $L \in \mathsf{coAM}$ and constant $k \geq 1$, there exists $\{x_n\}_{n\in\mathbb{N}}$ and single-valued Arthur-Merlin protocol (P, V) computing $\{x_n\}_{n\in\mathbb{N}}$ such that*

- ***Advice efficiency:** The advice complexity of (P, V) is $l(n) = 2^{\log^{1/k} n}$;*
- ***Time efficiency:** On input $(1^n, \zeta_n)$ where $|\zeta_n| = l(n)$, V runs in time $2^{\log^{O(k)} n}$;*
- ***Hitting:** For infinitely many $n \in \mathbb{N}$, $x_n \in L$.*

Note that the original result in [CLL24] works for only languages with half density, i.e., $|L \cap \{0,1\}^n| \geq 1/2$. However, it is easy to amplify the density by parallel repetition, sketched as follow: For language L and some $p : \mathbb{N} \to \mathbb{N}$, define L' as

$$L'_n = \{x \mid x_{[1\,:\,l]} \in L, \text{ or } \cdots, \text{ or } x_{[(p(n)-1)\cdot l+1\,:\,p(n)\cdot l]} \in L, \text{ where } l = \lfloor |x|/p(n) \rfloor\}$$

we can show $L \in \mathsf{coAM}$ implies $L' \in \mathsf{coAM}$. Furthermore, if L only has inverse-polynomial density (i.e., $|L \cap \{0,1\}^n| \geq 2^n/n^c$), when $p(n)$ is some appropriately picked polynomial function, we can have $|L' \cap \{0,1\}^n| \geq 2^{n-1}$, so the original explicit construction result can be applied to obtain single-valued Arthur-Merlin protocol (P', V') hitting L'. Then we apply $\log p$ bits of additional advice to locate the L-hitting subsection of an L'-hitting output from (P', V').

Then we show that this non-binary Arthur-Merlin algorithm hits coAM properties with strings of small rnKt complexity.

Lemma 4.17. *Let $L \in \mathsf{coAM}$ be a dense language, then for any $k > 1$, there exists infinitely many $n \in \mathbb{N}$ and $x_n \in \{0,1\}^n$, such that $x_n \in L \cap \{0,1\}^n$ and $\mathsf{rnKt}(x_n) < O(2^{\log^{1/k} n})$.*

Proof. Fix any $k > 1$, define $T(n) = 2^{\log^{O(k)} n}$, $l(n) = 2^{\log^{1/k} n}$. There exists s-round single-valued Arthur-Merlin protocol (P, V) with running time $T(n)$ and advice complexity $l(n)$, as described in Lemma 4.16. We may then assume that for each randomness r_i and prover response w_i there is $|r_i| = |w_i| = T(n)$, and the verifier only probe a prefix of each.

Define deterministic Turing machine $V'_n(w, r) : \{0,1\}^{s \cdot T(n)} \times \{0,1\}^{s \cdot T(n)} \to \{0,1\}^n$ as follow: With n and α_n embedded inside the machine description, V'_n prints $(1^n, \alpha_n)$ on its tape first, parses $r = r_1 \circ \cdots \circ r_s$ and $w = w_1 \circ \cdots \circ w_s$ where $r_i, w_i \in \{0,1\}^{T(n)}$, then simulates $(P, V)(1^n, \alpha_n)$. When V tosses its random coin r_i, V'_n simulates by r_i; When P needs to answer w_i, V'_n answers by w_i.

We then prove that V'_n satisfies definition of $\mathsf{rnKt}(x_n)$. For any prover P and $r_1, \cdots, r_s \in \{0,1\}^{T(n)}$, denote the output of (P, V) by $(P, V)(1^n, \alpha_n; r)$ when V has overall random coin $r = r_1 \circ \cdots \circ r_s$. Regarding conformity of (P, V), $(P, V)(1^n, \alpha_n; r) = x_n$ holds for every $r \in \{0,1\}^{s \cdot T(n)}$. Let $w^{(r)} = w_1 \circ \cdots \circ w_s$, where w_i is the prover response in the i-th round during the execution of $(P, V)(1^n, \alpha_n; r)$. By definition of V'_n, $V'_n(w^{(r)}, r) = (P, V)(1^n, \alpha_n; r)$ holds for every $r \in \{0,1\}^T$, i.e.,

$$\Pr_{r \sim \mathcal{U}_{s \cdot T(n)}}[\exists w \in \{0,1\}^{s \cdot T(n)}, V'_n(w, r) \text{ outputs } x \text{ in } s \cdot T(n) \text{ steps}] = 1.$$

By resiliency of V, there exists $y_n \in \{0,1\}^n$ such that for any oracle P', $\mathbf{Pr}[(P', V)(1^n, \alpha_n) \in \{y_n, \perp\}] \geq 2/3$. Fixing P' to be P, we obtain $x_n \in \{y_n, \perp\}$, which indicates $y_n = x_n$. To prove by contradiction, assume

$$\Pr_{r \sim \mathcal{U}_{s \cdot T(n)}}[\exists w^{(r)} \in \{0,1\}^{s \cdot T(n)}, V'_n(w_r, r) \notin \{x_n, \perp\}] \geq \frac{1}{3}.$$

We can then define prover P' that always answers $w^{(r)}_{[(i-1) \cdot T(n)+1 : \, i \cdot T(n)]}$ in the i-th round if $w^{(r)}$ exists and always answers $0^{T(n)}$ otherwise. With probability at least $1/3$ over r, V receives responses $w^{(r)}_1, \cdots w^{(r)}_s$ during execution of $(P', V)(1^n, \alpha_n; r)$. By definition of V', V does not output x_n or $\perp$ given r and those response. This contradicts the resiliency property and refutes the previous assumption. Therefore,

$$\Pr_{r \in \mathcal{U}_{s \cdot T(n)}}[\forall w \in \{0,1\}^{s \cdot T(n)}, V'_n(w, r) \text{ outputs } x \text{ or } \perp \text{ in } s \cdot T(n) \text{ steps}] \geq \frac{2}{3}.$$

Since s is constant, we can verify that V'_n witnesses $\mathsf{rnKt}(x_n) < O(l) + O(\log T(n)) = O(2^{\log^{1/k} n}) + O(\log^{O(k)} n) = O(2^{\log^{1/k} n})$. Recall the hitting property of (P, V), there exists infinitely many sufficiently large $n \in \mathbb{N}$ and $x_n \in L_n$ such that, $\mathsf{rnKt}(x_n) < O(2^{\log^{1/k} n})$. □

We are then ready to prove the central result of this subsection.

Proof of Theorem 1.2, Item 2. Towards contradiction we assume the contrary, i.e., there exists $\gamma \in (0,1)$, $d > 1$, $L \subseteq \mathcal{R}_{\mathsf{rnKt}[2^{\log^\gamma n}]}$ such that some algorithm $A \in \mathsf{coAM}[2^{\log^d n}]$ decides L, and L is dense.

Applying standard padding argument, we define $l(n) = 2^{\log^d n}$ and

$$L' = \{x \circ y \mid x \in L, \lfloor l^{-1}(|x| + |y|) \rfloor = |x|\}.$$

To decide L' on $x \in \{0,1\}^n$, we might first compute $m = \lfloor l^{-1}(n) \rfloor$, which can be done in $\mathsf{poly}(n)$ time assuming that l is efficiently computable. Then L' can be decided in coAM, just by truncating the input into first m bits and run the coAM protocol in time $2^{\log^d m} = l(m) \leq n$.

By assumption, L is dense, which immediately gives L' is dense. Applying Lemma 4.17, we can conclude for any constant $k > 1$ that there exist infinitely many $n \in \mathbb{N}$ and $z_n \in L'_n$ such that $\mathsf{rnKt}(z_n) < O(2^{\log^{1/k} n})$. For such z_n, we decompose it into $z_n = x_n \circ y_n$ such that $|x_n| = \lfloor l^{-1}(n) \rfloor$, then

- By definition of L', given that $x_n \circ y_n \in L'_n$, we have $x_n \in L$. Recall that $L \subseteq \mathcal{R}_{\mathsf{rnKt}[2^{\log^\gamma n}]}$, we have $\mathsf{rnKt}(x_n) \geq 2^{\log^{\gamma/d} n}$;
- By definition of rnKt, trivially, we have $\mathsf{rnKt}(x_n) \leq \mathsf{rnKt}(x_n \circ y_n) + O(1) < O(2^{\log^{1/k} n})$.

This gives

$$2^{\log^{\gamma/d} n} < O(2^{\log^{1/k} n}).$$

Note the arbitrariness of $k > 1$, we take $k = 2d/\gamma$, then

$$O(2^{\log^{1/k} n}) = O(2^{\log^{\gamma/2d} n}) < 2^{\log^{\gamma/d} n}.$$

which contradicts the previous inequality. □

4.2.3 Non-Uniform Lower Bound via Near-Maximum Circuit Lower Bounds

From now on we manage to prove Theorem 1.2, Item 3, which begins with some technical tools.

Lemma 4.18. *For any language $\mathcal{O} \in \mathsf{AM} \cap \mathsf{coAM}$ and any constant $\varepsilon > 0$, there exists a family of multisets $T = \{T_s \subseteq \{0,1\}^s\}_{s\in\mathbb{N}}$ such that each T_s satisfies $|T_s| \leq \mathsf{poly}(s)$ and the following properties hold*:

Efficiency: *For all sufficiently large s, we have $\mathsf{rnKt}(T_s) \leq 2^{\log^\varepsilon s}$. Here, the multiset T_s with k elements is represented as a binary string of the form $e_1\, 01\, e_2\, 01\, \ldots\, 01\, e_k$, where each e_i is a $2s$-bit string encoding an element of T_s, with each bit duplicated.*

Pseudorandomness: *For infinitely many $s \in \mathbb{N}$, the following holds: for every $\mathcal{O}$-oracle circuit $C\colon \{0,1\}^s \to \{0,1\}$ of size at most s,*

$$\left| \mathop{\mathbf{Pr}}_{x\sim\{0,1\}^s}[C(x) = 1] - \mathop{\mathbf{Pr}}_{x\sim T_s}[C(x) = 1] \right| \leq \frac{1}{s}.$$

Next, we show Lemma 4.18. We need the following tools.

Lemma 4.19 ([CLL24, Theorem 1.2])**.** *For any language $\mathcal{O} \in \mathsf{AM} \cap \mathsf{coAM}$ and any $0 < \varepsilon < 1$, $(\mathsf{AMEXP} \cap \mathsf{coAMEXP})/2^{n^\varepsilon} \not\subseteq \mathsf{SIZE}^L[2^n/n]$.*

The following lemma gives upper bound on rnKt complexity of truth table decidable in AMEXP ∩ coAMEXP with short advice.

Lemma 4.20. *For sufficiently large $n \in \mathbb{N}$ and $0 < \varepsilon < 1$, if $f : \{0,1\}^* \to \{0,1\}$ is computable in $(\mathsf{AMEXP} \cap \mathsf{coAMEXP})/2^{n^\varepsilon}$, then $\mathsf{rnKt}(\mathsf{tt}(f_n)) = O(2^{n^\varepsilon})$.*

Proof. We first state a extended version of Lemma 3.11, which handles oracle computed by non-uniform Arthur-Merlin protocol.

Claim 4.21. *For any $T, \alpha : \mathbb{N} \to \mathbb{N}$, $T(n) \geq n$, and every $\mathcal{O} \in (\mathsf{AMTIME}[T(n)] \cap \mathsf{coAMTIME}[T(n)])/\alpha(n)$, restricting query length to exactly $l \in \mathbb{N}$, there is a constant $c > 0$ such that for every $x, y \in \{0,1\}^*$ and $t \in \mathbb{N}$,*

$$\mathsf{rnK}^{O(t \log t)\cdot T(l)^c}(x \mid y) \leq \mathsf{rnK}^{t,\mathcal{O}_{=l}}(x \mid y) + c \cdot \log l + \alpha(l).$$

Proof Sketch of Claim 4.21. The proof is basically identical to the proof of Lemma 3.11, but the non-uniform advice required to correctly simulate Arthur-Merlin protocols is fixed in the Turing machine description, resulting in $\alpha(l)$ extra bits. ◇

The proof then is a straightforward application of Claim 4.21. Assume there exists $k > 0$ and $0 < \varepsilon < 1$ such that $f \in (\mathsf{AMTIME}[2^{n^k}] \cap \mathsf{coAMTIME}[2^{n^k}])/2^{n^\varepsilon}$. Hence there exists a uniform algorithm $M^{(\text{-})}$ such that, with access to oracle f on input length n, it outputs $\mathsf{tt}(f_n)$ in time $O(2^n)$, so

$$\mathsf{rnK}^{2^n, f_{=n}}(\mathsf{tt}(f_n)) = O(1).$$

By Claim 4.21, we have

$$\mathsf{rnKt}(\mathsf{tt}(f_n)) \leq \mathsf{rnK}^{O(2^n \cdot n)\cdot 2^{cn^k}}(\mathsf{tt}(f_n)) + O(n^k) + 2^{n^\varepsilon} = O(2^{n^\varepsilon}),$$

as desired. □

As an application of Lemmas 4.19 and 4.20, we get the following.

Lemma 4.22. *For any language $\mathcal{O} \in \mathsf{AM} \cap \mathsf{coAM}$ and any constant $\gamma > 0$, there exists a sequence $\{x_N\}_{N \in \mathbb{N}}$ such that the following hold:*

- *For every large enough N, $\mathsf{rnKt}(x_N) \leq 2^{\log^\gamma N}$.*
- *For infinitely many N that are powers of 2, the string x_N, interpreted as the truth table of a function $f\colon \{0,1\}^{\log N} \to \{0,1\}$, has $\mathcal{O}$-oracle-circuit complexity at least $N/\log N$.*

Proof. Lemma 4.19 states that there exists a language in $(\mathsf{AMEXP} \cap \mathsf{coAMEXP})/2^{n^\gamma}$ whose truth tables have nearly maximal $\mathcal{O}$-oracle circuit complexity. Lemma 4.20 then states that each such truth table has rnKt-complexity at most $O\big(2^{\log^\gamma N}\big)$, where N denotes the length of the truth table. □

We now present the proof of Lemma 4.18.

Proof of Lemma 4.18. The proof combines Lemma 4.22 and Theorem 2.26.

Let $\mathcal{O} \in \mathsf{AM} \cap \mathsf{coAM}$. We start by applying Theorem 2.26 with $\varepsilon := 1/2$, and let c, d be the corresponding constants from the lemma.

Consider Lemma 4.22 with $\gamma := \varepsilon/2$ and let $\{x_N\}$ be the corresponding sequence of strings.

Let $s \in \mathbb{N}$. We call s *good* if there exists a number N such that N is the smallest integer satisfying $s^c \leq N < (s+1)^c$, and the string x_N has $\mathcal{O}$-oracle-circuit complexity at least $N/\log N$. Observe that there are infinitely many good s, and the corresponding N satisfies $N \leq (s+1)^c$.

For each good s, we will apply the function F from Theorem 2.26 to x_N. Note that the output length of $F(x_N, -)$ is at least $N^{1/c} \geq s$. Finally, define

$$T_s := \left\{ F(x_N, z)_{[1:s]} \mid z \in \{0,1\}^{d \log N} \right\}.$$

We first argue that T_s is satisfies the pseudorandom condition for good s. This follows from the fact that if s is good, then the corresponding x_N has high $\mathcal{O}$-oracle-circuit complexity (as a truth table) and the security guarantee provided by the pseudorandom generator F in Theorem 2.26.

Next, we address the efficiency condition. Note that for a (sufficiently large) good s, given the integer s and the string x_N, where N is the integer witnessing the goodness of s, we can construct T_s in time $\mathsf{poly}(N) \leq \mathsf{poly}(s)$. By the efficiency condition stated in Lemma 4.22, we have

$$\mathsf{rnKt}(x_N) \leq 2^{\log^\gamma N}.$$

It follows that

$$\begin{aligned} \mathsf{rnKt}(T_s) &\leq 2^{\log^\gamma N} + \log \mathsf{poly}(s) \\ &\leq 2^{O(\log s)^\gamma} + O(\log s) \\ &\leq 2^{\log^\varepsilon s}, \end{aligned}$$

as desired. □

Putting it all together, we are now ready to show Theorem 1.2, Item 3.

Proof of Theorem 1.2, Item 3. Towards contradiction, we assume the contrary. That is, there exists $\gamma > 0, \mathcal{O} \in \mathsf{AM} \cap \mathsf{coAM}, c > 1, d \geq 1, L \subseteq \mathcal{R}_{\mathsf{rnKt}[2^{\log^\gamma n}]}$ satisfying

- L can be decided by $\mathcal{O}$-oracle circuit family $\{C_n^{\mathcal{O}}\}_{n\in\mathbb{N}}$ of size $2^{\log^d n}$;
- There exists $N \in \mathbb{N}$ such that $|L \cap \{0,1\}^n| \geq 2^n/n^c$ for all $n > N$.

We instantiate Lemma 4.18 with $\varepsilon := \gamma/2$, obtain the corresponding family of multi-sets $T = \{T_s\}_{s\in\mathbb{N}}$. Consider the following algorithm $D : \{0,1\}^* \to \{0,1\}$ trying to break the pseudo-randomness:

Algorithm 3 A non-uniform algorithm for breaking pseudo-randomness

```
procedure $D(x)$
    $s := |x|$.
    $m := 2^{(2/(c-1)\cdot\log s)^{1/(d+1)}}$.
    if $m \leq N$ then
        Output 0.
    $k := m^{(c-1)/2}$.
    if $km > s$ then
        Output 0.
    for $i \in [k]$ do
        Compute $a_i := C_m^{\mathcal{O}}(x_{[(i-1)m+1\,:\,im]})$.
        if $a_i = 1$ then
            Output 1.
    Output 0.
```

Assume $s_0 \in \mathbb{N}$ is large enough such that for every $s \geq s_0, x \in T_s$, $\mathsf{rnKt}(x) \leq \mathsf{rnKt}(T_s) \leq 2^{\log^{\gamma/2} s}$, whose existence is guaranteed by efficiency of T. Given that $L \subseteq \mathcal{R}_{\mathsf{rnKt}[2^{\log^\gamma n}]} \subseteq \mathcal{R}_{\mathsf{rnKt}[2^{\log^{\gamma/2} n}]}$, $L \cap T_s = \emptyset$. Since circuit family $\{C_n^{\mathcal{O}}\}_{n\in\mathbb{N}}$ computes L, $D(x) = 0$ holds for any $x \in T_s$. This indicates that, for any $s \geq s_0$,

$$\mathop{\mathbf{Pr}}_{x\sim T_s}[D(x) = 1] = 0.$$

Pick $s_1 = 2^{c/2\cdot\log^{d+1} N}$, we can verify that for any $s \geq s_1$,

$$m = 2^{(2/(c-1)\cdot\log s)^{1/(d+1)}} \geq 2^{(c/(c-1))^{1/(d+1)}\cdot\log N} > N.$$

In this case, $|L \cap \{0,1\}^m| \geq 2^m/m^c$ by our assumption on S. Furthermore, we have $km = m^{(c+1)/2} = 2^{\log^{1/(d+1)} s\cdot F(c)}$ where $F(c) = (2/(c-1))^{1/(d+1)} \cdot (c+1)/2$ is universally constant, so we pick $s_2 := \max\{16, 2^{F(c)^4}\}$, which gives

$$km = 2^{\log^{1/(d+1)} s\cdot F(c)} \leq 2^{\log^{1/(d+1)} s\cdot\log^{1/4} s} \leq 2^{\log^{3/4} s} < \frac{s}{2}.$$

for all $s \geq s_2$ (the second last inequality follows from $d \geq 1$, while the last follows from $s \geq 16$). Now for all $s \geq \max\{s_1, s_2\}$ and $x \in \{0,1\}^s$,

$$D(x) = \bigvee_{i=1}^{k} C_m^{\mathcal{O}}(x_{[(i-1)m+1\,:\,im]}).$$

Therefore,

$$\begin{aligned}\Pr_{x\sim\{0,1\}^s}[D(x)=0] &= \Pr_{x\sim\{0,1\}^s}[x_{[1\,:\,m]}\notin L,\cdots,x_{[(k-1)m+1\,:\,km]}\notin L]\\ &= \left(\Pr_{y\sim\{0,1\}^m}[y\notin L]\right)^k \le \left(1-\frac{1}{m^c}\right)^k\\ &< \exp\left(-\frac{k}{m^c}\right).\end{aligned}$$

Given that $k = m^{(c-1)/2}$, we have

$$k = \frac{m^{c-1}}{k} > m^c \ln\left(1+\frac{1}{km}\right),$$

immediately giving

$$\exp\left(-\frac{k}{m^c}\right) < \exp\left(-m^c\ln\left(1+\frac{1}{km}\right)\cdot\frac{1}{m^c}\right) = 1-\frac{1}{km+1} \le 1-\frac{2}{s+2} < 1-\frac{1}{s}.$$

In conclusion, for all $s \ge \max\{s_0, s_1, s_2\}$, we have

$$\Pr_{x\sim\{0,1\}^s}[D(x)=1] - \Pr_{x\sim T_s}[D(x)=1] > \frac{1}{s} - 0 = \frac{1}{s}.$$

which contradicts the pseudo-randomness of $T = \{T_s\}_{s\in\mathbb{N}}$ in Lemma 4.18. □

5 Nondeterminism, Symmetry of Information, and Proof Complexity

In this section, we prove our results stated in Section 1.2.2.

Proof of Theorem 1.3. Theorem 5.2, stated and proved in Section 5.1, shows that average-case symmetry of information holds for pnKt. Similarly, Theorem 5.10, proved in Section 5.3, establishes an analogous result for rnKt. □

Proof of Theorem 1.4. Theorem 5.3, stated and proved in Section 5.2, shows that if worst-case symmetry of information holds for pnKt, then $\mathsf{coNP} \not\subseteq \mathsf{AM}$. Theorem 5.11 establishes an analogous result for rnKt and is proved in Section 5.3. □

5.1 Average-Case Symmetry of Information for pnKt

In this subsection, we show that Symmetry of Information holds for pnKt *on average* over efficiently samplable distributions.

Toward proving the above, we observe that Symmetry of Information holds for $\mathsf{pK}^{\mathsf{poly}}$ if one replaces the term $\mathsf{pK}^{\mathsf{poly}}(y \mid x)$ with its nondeterministic counterpart $\mathsf{pnK}^{\mathsf{poly}}(y \mid x)$.

5.1.1 A Semi-Symmetry of Information Theorem for pnK^t

We obtain a version of semi-SoI for pnK^t via the worst-case conditional coding theorem of [HLO25]. The proof is similar to an argument from [HIL+23] used to conditionally show average-case SoI for pK^t. The statement slightly improves on the worst-case semi-SoI theorem of [HLO25], which would include a term for $\mathsf{pnK}^{\mathsf{poly}(t)}(y)$ rather than $\mathsf{pK}^{\mathsf{poly}(t)}(y)$.

Theorem 5.1. *There exists a polynomial p such that, for all sufficiently large $n, t \in \mathbb{N}$ with $t \geq 4n$, the following holds. For all $x, y \in \{0,1\}^n$,*

$$\mathsf{pK}^t(x,y) \geq \mathsf{pK}^{p(t)}(x) + \mathsf{pnK}^{p(t)}(y \mid x) - \log p(t).$$

Proof. Let $\mathcal{D} \in \mathsf{PSAMP}$ be a distribution family as in Lemma 36 of [LOZ22] that dominates the universal distribution for pK^t; that is, for some constant $c \in \mathbb{N}$, for any $n, t \in \mathbb{N}$ and $z \in \{0,1\}^{2n}$,

$$\mathcal{D}_{n,t}(z) \geq \frac{1}{2^{\mathsf{pK}^t(z)} \cdot n^c}. \tag{16}$$

For $n, t \in \mathbb{N}$, let $\mathcal{D}^{(2)}_{n,t}$ denote the marginal distribution over the second half of the output of $\mathcal{D}_{n,t}$, and note that $\mathcal{D}^{(2)} = \{\mathcal{D}^{(2)}_{n,t}\}_{n,t\in\mathbb{N}} \in \mathsf{PSAMP}$.

Now fix $n, t \in \mathbb{N}$ such that $t \geq 4n$ along with strings $x, y \in \{0,1\}^n$. By the worst-case conditional coding theorem for pnK^t with respect to $\mathcal{D}$ ([HLO25, Theorem 6]), for some polynomial q,

$$\begin{aligned}
\mathsf{pnK}^{q(t)}(y \mid x) &\leq \log \frac{1}{\mathcal{D}_{n,t}(y \mid x)} + \log q(t) \\
&= \log \frac{\mathcal{D}^{(2)}_{n,t}(x)}{\mathcal{D}_{n,t}(x,y)} + \log q(t) \\
&= \log \frac{1}{\mathcal{D}_{n,t}(x,y)} - \log \frac{1}{\mathcal{D}^{(2)}_{n,t}(x)} + \log q(t) \\
&\leq \mathsf{pK}^t(x,y) - \log \frac{1}{\mathcal{D}^{(2)}_{n,t}(x)} + \log q(t) + c \log n. && \text{(by Equation (16))}
\end{aligned}$$

Moreover, by the worst-case coding theorem for pK^t complexity [LOZ22], for some polynomial q',

$$\mathsf{pK}^{q'(t)}(x) \leq \log \frac{1}{\mathcal{D}^{(2)}_{n,t}(x)} + \log q'(t).$$

Combining this with the above,

$$\mathsf{pnK}^{q(t)}(y \mid x) + \mathsf{pK}^{q'(t)}(x) \leq \mathsf{pK}^t(x,y) + \log q(t) + c \log n + \log q'(t).$$

Let the polynomial p be such that $p(m) \geq 4 \cdot \max\{q(m), m^c, q'(m)\}$ for all $m \in \mathbb{N}$. It follows that

$$\mathsf{pnK}^{p(t)}(y \mid x) + \mathsf{pK}^{p(t)}(x) \leq \mathsf{pK}^t(x,y) + \log p(t).$$

This completes the proof. □

5.1.2 From Semi-SoI for pnK^t to Average-Case SoI for pnKt

Making use of the statement from the previous subsection, we prove that average-case Symmetry of Information holds for pnKt. The proof is a simple adaptation of an argument from [HLO25]; we include the details for completeness.

Below, we state Theorem 1.3 in a seemingly stronger form, with a term for $\mathsf{pKt}(x)$ on the right-hand side rather than $\mathsf{pnKt}(x)$. However, we remark that this statement is actually essentially equivalent to Theorem 1.3, since $\mathsf{pKt}(x) \leq \mathsf{pnKt}(x) + O(\log n)$ with high probability over strings x sampled efficiently.

Theorem 5.2. *For every polynomial-time samplable distribution family $\{\mathcal{D}_n\}_{n\in\mathbb{N}}$ supported over $\{0,1\}^n \times \{0,1\}^n$, there exists a constant $c > 0$ such that for all large enough $n, k \in \mathbb{N}$,*

$$\Pr_{(x,y)\sim\mathcal{D}_n}[\ \mathsf{pnKt}(x,y) \geq \mathsf{pKt}(x) + \mathsf{pnKt}(y \mid x) - c\cdot\log n - \log k\ ] \geq 1 - \frac{1}{k}.$$

Proof. The proof is following [HLO25, Theorem 8]. Let $\{\mathcal{D}_n\}_{n\in\mathbb{N}}$ be as in the statement of this theorem, and let the polynomial p be as in Theorem 5.1. Fix any $n \in \mathbb{N}$, and for a sufficiently large polynomial q, let $t = q(t_D(n))$, where $t_D(n)$ upper-bounds the time required to sample $\mathcal{D}_n$. Observe that, for any $x, y \in \{0,1\}^n$,

$$\mathsf{pnKt}(y \mid x) \leq \mathsf{pnK}^{p(t)}(y \mid x) + O(\log p(t))$$

and

$$\mathsf{pKt}(x) \leq \mathsf{pK}^{p(t)}(x) + O(\log p(t)).$$

By the coding theorem for pK^t (Theorem 2.23) and Lemma 2.22, the following holds: for any $k \in \mathbb{N}$, with probability at least $1 - k^{-1}$ over $(x,y) \sim D_n$,

$$\begin{aligned}\mathsf{pK}^t(x,y) &\leq \log\frac{1}{D_n(x,y)} + O(\log n)\\ &\leq \mathsf{K}(x,y) + O(\log n) + \log k\\ &\leq \mathsf{pnKt}(x,y) + O(\log n) + \log k.\end{aligned}$$

Combining the above with Theorem 5.1 and letting $c \in \mathbb{N}$ be a sufficiently large constant, we have that with probability at least $1 - k^{-1}$ over $(x,y) \sim D_n$,

$$\mathsf{pnKt}(x,y) \geq \mathsf{pKt}(x) + \mathsf{pnKt}(y \mid x) - c\log n - \log k,$$

as desired. □

5.2 Worst-Case Symmetry of Information for pnKt and coNP vs. AM

In this section, we prove Theorem 1.4 for the case of pnKt. Here, we will only present a proof that, if worst-case symmetry of information for pnKt holds with an error term $e = \mathsf{polylog}(n)$, then $\mathsf{coNP} \not\subseteq \mathsf{AMTIME}[n^{\mathsf{polylog}(n)}]$. The case with a smaller error term $e = O(\log n)$ — leading to $\mathsf{coNP} \not\subseteq \mathsf{AMTIME}[2^{n^{o(1)}}]$, as stated in Theorem 1.4 — can be handled similarly by appropriately adjusting the parameters in the proof.

Theorem 5.3. *Item 1 implies Item 2 in the following.*

1. **(Worst-Case Symmetry of Information for pnKt).** *There exists a constant $c > 0$ such that for all large enough $n \in \mathbb{N}$ and $x, y \in \{0,1\}^n$,*

$$\mathsf{pnKt}(x,y) \geq \mathsf{pnKt}(x) + \mathsf{pnKt}(y \mid x) - \log^c n.$$

2. $\mathsf{UNSAT} \notin \mathsf{AMTIME}\left[2^{\log^d n}\right]$ *for any $d \geq 1$.*

We will need the following lemmas.

Lemma 5.4. *There exists a probabilistic oracle algorithm $B^{(-)}$ such that, given inputs $x \in \{0,1\}^m$ and $y \in \{0,1\}^{\leq 2^m\cdot m}$, the algorithm $B^{\mathsf{SAT}}(x,y)$ runs in time $2^{O(m)}$ with access to a SAT-oracle, and*

- *if* $\mathsf{pnKt}_{2/3}(x \mid y) \le m/2$*, then* $B^{\mathsf{SAT}}(x,y)$ *rejects with probability at least* $2/3$*, and*
- *if* $\mathsf{pnKt}_{1/3}(x \mid y) > m/2$*, then* $B^{\mathsf{SAT}}(x,y)$ *accepts with probability at least* $2/3$*.*

Proof. Consider the following algorithm: on input (x, y), where $x \in \{0,1\}^m$ and $y \in \{0,1\}^{\le 2^m \cdot m}$,

1. Sample $r \sim \{0,1\}^{2^{m/2}}$.
2. For all programs $\Pi \in \{0,1\}^*$ and running times $t \in \mathbb{N}$ such that $|\Pi| + \lceil \log t \rceil \le m/2$, use the SAT oracle to answer the following two queries:
   (a) Does there exist a $w \in \{0,1\}^t$ such that $\Pi(y; w; r)$ outputs x in t steps?
   (b) Does there exist a $w \in \{0,1\}^t$ such that $\Pi(y; w; r)$ either fails to terminate within t steps or outputs some $x' \notin \{x, \perp\}$ within t steps?
3. Reject if and only if there exists a pair (Π, t) such that the answer to the first query is yes and the answer to the second query is no.

It is clear that the running time of the above algorithm is $2^{O(m)}$, and correctness follows directly from the definition of pnKt. Indeed, if $\mathsf{pnKt}_{2/3}(x \mid y) \le m/2$, then the algorithm rejects with probability at least $2/3$ (over the randomness r), and if $\mathsf{pnKt}_{1/3}(x \mid y) > m/2$, the algorithm rejects with probability at most $1/3$. □

Lemma 5.5. *Suppose* $\mathsf{SAT} \in \mathsf{coAMTIME}[n^{\mathsf{polylog}(n)}]$*. Then for every constant* $d > 0$*, there are infinitely many integers* m *such that for every* $\ell \le 2^m$*, there exist strings* $x_1, \ldots, x_\ell \in \{0,1\}^m$ *satisfying*

$$\sum_{i \in [\ell]} \mathsf{pnKt}(x_i \mid x_1, \ldots, x_{i-1}) \ge \ell \cdot \frac{m}{2}$$

and

$$\mathsf{rnKt}(x_1, \ldots, x_\ell) \le 2^{m^{1/d}}.$$

Proof. Let B be the (probabilistic) algorithm from Lemma 5.4. We first observe the following.

Claim 5.6. *There exists a constant* $c > 0$ *and a language* $L \in \mathsf{AM} \cap \mathsf{coAM}$ *such that the following holds. Given any* $x \in \{0,1\}^m$ *and* $y \in \{0,1\}^{\le 2^m \cdot m}$*, one can construct, in deterministic time* $\mathsf{poly}(2^{m^c})$*, an* L*-oracle circuit* $C_{(x,y)}$ *of size* 2^{m^c} *that treats its input as internal randomness used by the algorithm* B *and computes* $B^{\mathsf{SAT}}(x,y)$*; that is,* $C_{(x,y)}(r) = B^{\mathsf{SAT}}(x, y; r)$ *for every* r*.*

Proof of Claim 5.6. By Lemma 5.4, for every $x \in \{0,1\}^m$ and $y \in \{0,1\}^{\le 2^m \cdot m}$, we can construct a SAT-oracle circuit C_0 of size $s = 2^{O(m)}$ that treats its input as internal randomness and computes $B^{\mathsf{SAT}}(x,y)$. Note that C_0 makes queries to SAT of length at most s.

By assumption, $\mathsf{SAT} \in \mathsf{coAMTIME}[t(n)]$, where $t(n) = n^{\mathsf{polylog}(n)}$. Applying a standard padding argument, we can define a language $L \in \mathsf{AM} \cap \mathsf{coAM}$ that is an appropriately padded version of SAT, such that C_0 can be simulated by an L-oracle circuit of size at most $\mathsf{poly}(t(s)) = 2^{\mathsf{poly}(m)}$. ◇

Let $c > 0$ be the constant and $L \in \mathsf{AM} \cap \mathsf{coAM}$ be the language from Claim 5.6. Consider Lemma 4.18 with $\varepsilon := 1/(4cd)$. We say that an integer m is *good* if there exists an s such that s is the smallest integer that is at least $\ge 2^{m^c}$ and the multiset T_s from Lemma 4.18 is pseudorandom against L-oracle circuits of size s. Note that there are infinitely many good values of m.

Fix any (sufficiently large) good m. Let s be the integer that witnesses the goodness of m, and let T_s be the corresponding pseudorandom set.

Consider the following deterministic procedure A, with oracle access to L, which takes as input m, T_s, and a string $y \in \{0,1\}^{\le 2^m \cdot m}$, and proceeds as follows:

For each $x \in \{0,1\}^m$, compute

$$\mu_x := \mathop{\mathbf{Pr}}_{z \sim T_s}\left[C_{(x,y)}(z) = 1\right],$$

where $C_{(x,y)}$ is the L-oracle circuit obtained using Claim 5.6. Output the first x such that $\mu_x > \frac{1}{3} + \frac{1}{10}$.

Claim 5.7. *The procedure A, on input (m, T_s, y) with $y \in \{0,1\}^{\leq 2^m \cdot m}$, runs in deterministic time at most $2^{\mathsf{poly}(m)}$ when given oracle access to L and outputs an m-bit string x satisfying $\mathsf{pnKt}(x \mid y) \geq m/2$.*

Proof of Claim 5.7. We first analyze the running time. Since $|T_s| \leq \mathsf{poly}(s)$ and $s \leq 2^{(m+1)^c}$, it follows that the algorithm runs in time at most $2^{\mathsf{poly}(m)}$.

We now show correctness. Because T_s $(1/10)$-fools $C_{(x,y)}$ for every $x \in \{0,1\}^m$ and $y \in \{0,1\}^{\leq 2^m \cdot m}$, the value μ_x approximates $\mathbf{Pr}_B[B^{\mathsf{SAT}}(x,y) = 1]$ within additive error $1/10$. Thus, any output x cannot satisfy $\mathsf{pnKt}_{2/3}(x \mid y) \leq m/2$, as that would imply $\mathbf{Pr}_B[B^{\mathsf{SAT}}(x,y) = 1] < 1/3$ and hence $\mu_x < 1/3 + 1/10$, contradicting the condition for output. Furthermore, since we enumerate all $x \in \{0,1\}^m$, at least one of them satisfies $\mathsf{pnKt}_{1/3}(x) > m/2$. It follows that $B^{\mathsf{SAT}}(-,y)$ accepts at least one such x. Therefore, we are guaranteed to find one with $\mu_x \geq 2/3 - 1/10 \geq 1/3 + 1/10$. ◇

Define $x_1, x_2, \ldots, x_\ell \in \{0,1\}^m$ as follows:

- x_1 is the output of $A(m, T_s, \epsilon)$.
- For $i = 2, 3, \ldots, \ell$, let x_i be the output of $A(m, T_s, x_1, \ldots, x_{i-1})$.

By correctness of A, for each $i \in [\ell]$, we have

$$\mathsf{pnKt}(x_i \mid x_1, \ldots, x_{i-1}) \geq \frac{m}{2}$$

It follows that

$$\sum_{i \in [\ell]} \mathsf{pnKt}(x_i \mid x_1, \ldots, x_{i-1}) \geq \ell \cdot m/2,$$

as desired.

To upper bound the rnKt-complexity, observe that we can first compute T_s and then run procedure A repeatedly to obtain the sequence $x_1, \ldots, x_\ell$.

From the efficiency guarantee in Lemma 4.18, we have:

$$\mathsf{rnKt}(T_s) \leq 2^{\log^{\varepsilon} s} \leq 2^{m^{1/3d}},$$

Given m and T_s, the sequence $x_1, \ldots, x_\ell$ can be computed via ℓ invocations of A, each running in time $2^{\mathsf{poly}(m)}$ *given oracle access to L*. This yields:

$$\begin{aligned}\mathsf{rnKt}^L(x_1, \ldots, x_\ell) &\leq O\Big(2^{m^{1/3d}}\Big) + \log\Big(\ell \cdot 2^{\mathsf{poly}(m)}\Big) \\ &\leq 2^{m^{1/(2d)}}. \qquad (17)\end{aligned}$$

In the above we use the fact that m is sufficiently large.

Finally, since $L \in \mathsf{AM} \cap \mathsf{coAM}$, we get

$$\begin{aligned}\mathsf{rnKt}(x_1, \ldots, x_\ell) &\leq O\big(\mathsf{rnKt}^L(x_1, \ldots, x_\ell)\big) && \text{(by Corollary 3.12)} \\ &\leq O\Big(2^{m^{1/(2d)}}\Big) && \text{(by Equation (17))} \\ &\leq 2^{m^{1/d}}.\end{aligned}$$

This completes the proof. □

We now proceed to prove Theorem 5.3.

Proof of Theorem 5.3. We show the contrapositive. Assume that $\mathsf{UNSAT} \in \mathsf{AMTIME}[n^{\mathsf{polylog}(n)}]$. Note that this implies $\mathsf{SAT} \in \mathsf{coAMTIME}[n^{\mathsf{polylog}(n)}]$. Below, we show the failure of worst-case symmetry of information for pnKt. That is, for every constant $c > 0$, there exist infinitely many $n \in \mathbb{N}$ and $x, y \in \{0,1\}^n$ such that

$$\mathsf{pnKt}(x, y) < \mathsf{pnKt}(x) + \mathsf{pnKt}(y \mid x) - \log^c n.$$

Let $c > 1$ be any constant. We first establish the following claim.

Claim 5.8. *The following holds for infinitely many values of m. There exist $v \in \{0,1\}^m$ and $u \in \{0,1\}^n$, where $m \le n \le 2^{m^{1/(2c)}}$, such that*

$$\mathsf{pnKt}(u, v) < \mathsf{pnKt}(u) + \mathsf{pnKt}(v \mid u) - \frac{m}{4},$$

Proof of Claim 5.8. Suppose, for the sake of contradiction, that for all but finitely many m, the following holds: for all $v \in \{0,1\}^m$ and $u \in \{0,1\}^n$ satisfying $m \le n \le 2^{m^{1/(2c)}}$, we have

$$\mathsf{pnKt}(u, v) \ge \mathsf{pnKt}(u) + \mathsf{pnKt}(v \mid u) - \frac{m}{4}.$$

Consider Lemma 5.5 with $d := 3c$. Let m be a sufficiently large "good" integer such that the statement of Lemma 5.5 holds for $\ell := 2^{m^{1/d}}$, and let $x_1, \ldots, x_\ell$ be the associated sequence of strings.

Observe that $\ell \cdot m \le 2^{m^{1/(2c)}}$. Therefore, we can apply the assumption to $v = x_\ell$ and $u = (x_1, \ldots, x_{\ell-1})$, which yields:

$$\mathsf{pnKt}(x_1, \ldots, x_\ell) \ge \mathsf{pnKt}(x_1, \ldots, x_{\ell-1}) + \mathsf{pnKt}(x_\ell \mid x_1, \ldots, x_{\ell-1}) - \frac{m}{4}$$

Repeating this argument inductively gives:

$$\begin{aligned}
\mathsf{pnKt}(x_1, \ldots, x_\ell) &\ge \sum_{i \in [\ell]} \mathsf{pnKt}(x_i \mid x_1, \ldots, x_{i-1}) - \ell \cdot \frac{m}{4} \\
&\ge \ell \cdot \frac{m}{2} - \ell \cdot \frac{m}{4} && \text{(by Lemma 5.5)} \\
&= 2^{m^{1/d}} \cdot \frac{m}{4}.
\end{aligned}$$

However, this contradicts the fact that

$$\mathsf{pnKt}(x_1, \ldots, x_\ell) \le 2^{m^{1/d}}.$$

This completes the proof of the claim. ◇

By Claim 5.8, for infinitely many values of m, there exist $v \in \{0,1\}^m$ and $u \in \{0,1\}^n$ such that

$$m \le n \le 2^{m^{1/(2c)}}, \tag{18}$$

and

$$\mathsf{pnKt}(u, v) < \mathsf{pnKt}(u) + \mathsf{pnKt}(v \mid u) - \frac{m}{4}. \tag{19}$$

Define

$$x := u \quad \text{and} \quad y := v0^{n-m}. \tag{20}$$

Then we have:

$$\begin{aligned}
\mathsf{pnKt}(x,y) &\le \mathsf{pnKt}(u,v) + O(\log n) && \text{(by Equation (20))}\\
&\le \mathsf{pnKt}(u) + \mathsf{pnKt}(v \mid u) - \frac{m}{4} + O(\log n) && \text{(by Equation (19))}\\
&\le \mathsf{pnKt}(x) + \mathsf{pnKt}(y \mid x) + O(\log n) - \frac{m}{4} + O(\log n) && \text{(by Equation (20))}\\
&\le \mathsf{pnKt}(x) + \mathsf{pnKt}(y \mid x) + O(\log n) - \frac{\log^{2c} n}{4} && \text{(by Equation (18))}\\
&< \mathsf{pnKt}(x) + \mathsf{pnKt}(y \mid x) - \log^{c} n,
\end{aligned}$$

as desired. □

5.3 The Case of rnKt

We first show a version of semi-symmetry of information for rnK^t, which is analogous to Theorem 5.1.

Theorem 5.9. *There exists a polynomial p such that, for all sufficiently large $n, t \in \mathbb{N}$ with $t \ge 4n$, the following holds. For all $x, y \in \{0,1\}^n$,*

$$\mathsf{pK}^t(x,y) \ge \mathsf{rnK}^{p(t)}(x) + \mathsf{rnK}^{p(t)}(y \mid x) - \log p(t) - \log^3 p(n).$$

Proof Sketch. The proof follows a similar approach to that used in [GK22, Hir22c] to establish time-bounded versions of SoI under the assumption that MINKT is easy (see also [HKLO24, Appendix A]). This approach employs the pseudorandom generator construction stated in Lemma 2.27 and its reconstruction property, while using MINKT (which is in NP) as a distinguisher. In fact, by adapting this approach and using the fact that the reconstruction procedure makes only non-adaptive queries, one can show that for any $x, y \in \{0,1\}^n$ and sufficiently large t that is polynomial in n,

$$\mathsf{pK}^t(x,y) \ge \mathsf{rK}^{\mathsf{poly}(t),\|\mathsf{MINKT}}_{1/\mathsf{poly}(n)}(x) + \mathsf{rK}^{\mathsf{poly}(t),\|\mathsf{MINKT}}_{1/\mathsf{poly}(n)}(y \mid x) - O(\log^3 n) - O(\log t). \tag{21}$$

We now upper-bound the right-hand side of the above. We have

$$\begin{aligned}
\mathsf{rK}^{\mathsf{poly}(t),\|\mathsf{MINKT}}_{1/\mathsf{poly}(n)}(x) &\ge \mathsf{rnK}^{\mathsf{poly}(t)}_{1/\mathsf{poly}(n\cdot t)}(x) - O(\log t) && \text{(by Lemma 3.10)}\\
&\ge \mathsf{rnK}^{\mathsf{poly}(t)}(x) - O(\log t) && \text{(by Lemma 3.7)}
\end{aligned}$$

By a similar argument, we have

$$\mathsf{rK}^{\mathsf{poly}(t),\|\mathsf{MINKT}}_{1/\mathsf{poly}(n)}(y \mid x) \ge \mathsf{rnK}^{\mathsf{poly}(t)}(y \mid x) - O(\log t).$$

Substituting the above two inequalities into Equation (21) yields the desired semi-symmetry of information for rnK^t. □

The following corresponds to Theorem 1.3 for the case of rnKt

Theorem 5.10. *For every polynomial-time samplable distribution family $\{\mathcal{D}_n\}_{n\in\mathbb{N}}$ supported over $\{0,1\}^n \times \{0,1\}^n$, there exists a constant $c > 0$ such that for all large enough $n, k \in \mathbb{N}$,*

$$\Pr_{(x,y)\sim\mathcal{D}_n}\left[\ \mathsf{rnKt}(x,y) \ge \mathsf{rnKt}(x) + \mathsf{rnKt}(y \mid x) - c\cdot\log^3 n - \log k\ \right] \ge 1 - \frac{1}{k}.$$

Proof Sketch. The proof can be easily adapted from that of Theorem 5.2, with Theorem 5.1 replaced by Theorem 5.9. We omit the details. □

The following corresponds to Theorem 1.4 for the case of rnKt.

Theorem 5.11. *Item 1 implies Item 2 in the following.*

1. **(Worst-Case Symmetry of Information for rnKt).** *There exists a constant $c > 0$ such that for all large enough $n \in \mathbb{N}$ and $x, y \in \{0,1\}^n$,*

$$\mathsf{rnKt}(x, y) \geq \mathsf{rnKt}(x) + \mathsf{rnKt}(y \mid x) - \log^c n.$$

2. $\mathsf{UNSAT} \notin \mathsf{AMTIME}\left[2^{\log^d n}\right]$ *for any $d \geq 1$.*

Proof Sketch. The proof is a easy adaption of that of Theorem 5.3, which employs Lemma 5.5. It suffices to note that in Lemma 5.5, we can replace pnKt with rnKt, since for every $x, y \in \{0,1\}^n$, it holds that $\mathsf{pnKt}(x \mid y) \geq \mathsf{rnKt}(x \mid y)$. □

5.4 Relativization Barriers for Symmetry of Information

We will prove the following theorem.

Theorem 5.12. *There exist an oracle $\mathcal{O}$ and a constant $\varepsilon > 0$ as follows. For all sufficiently large $n \in \mathbb{N}$, there exist strings $x, y \in \{0,1\}^n$ such that, for every complexity measure $\mu \in \{\mathsf{rKt}, \mathsf{pKt}, \mathsf{nKt}, \mathsf{rnKt}, \mathsf{pnKt}\}$,*

$$\mu^{\mathcal{O}}(x, y) < \mu^{\mathcal{O}}(x) + \mu^{\mathcal{O}}(y \mid x) - \varepsilon \cdot n.$$

The proof is basically a relativizing version of the argument used in Theorem 5.3, adapted so that it works for all measures $\mu \in \{\mathsf{rKt}, \mathsf{pKt}, \mathsf{nKt}, \mathsf{rnKt}, \mathsf{pnKt}\}$ and gives a linear gap $\varepsilon \cdot n$ instead of just polylogarithmic.

We will make use of the following lemmas.

Lemma 5.13 ([BGS75]). *There exists an oracle $\mathcal{O}$ such that $\mathsf{PH}^{\mathcal{O}} \subseteq \mathsf{P}^{\mathcal{O}}$.*

Lemma 5.14. *There is a deterministic polynomial-time oracle machine $M_0^{(-)}$ as follows. Let $\mathcal{O}$ be any oracle. There exists an oracle $\mathcal{X} \in \Sigma_3^{\mathsf{P},\mathcal{O}}$ such that for any $n \in \mathbb{N}$, $M_0^{\mathcal{X}}(1^n)$ outputs a string $x \in \{0,1\}^n$ with $\mathsf{NP}^{\mathcal{O}}$-oracle circuit complexity at least $n^{1/2}$.*

Proof. For a given oracle $\mathcal{O}$, let $\mathcal{X} \in \Sigma_3^{\mathsf{P},\mathcal{O}}$ be as follows.

$$\mathcal{X} = \{(1^n, y) \mid y \in \{0,1\}^{<n} \;;\; \exists x \in \{0,1\}^n \text{ and } w \in \{0,1\}^{n-|y|-1} \text{ such that } x = y0w \text{ and}$$
$$\forall \text{ oracle circuit } C^{(-)} \text{ of size at most } n^{1/2}, x \text{ is not the truth-table of } C^{\mathsf{NP}^{\mathcal{O}}}\},$$

The machine $M_0^{(-)}$ is defined as follows.

> On input 1^n, start with y being the empty string. Repeat the following n times. Query $(1^n, y)$ to the oracle $\mathcal{X}$; if $\mathcal{X}$ accepts, let $y := y0$, and if $\mathcal{X}$ rejects, let $y := y1$. After this is done, return the string $x := y \in \{0,1\}^n$.

By a standard counting argument, there exists a string $x \in \{0,1\}^n$ with $\mathsf{NP}^{\mathcal{O}}$-oracle circuit complexity greater than $n^{1/2}$. (In fact, most strings have this property.) Therefore, it is clear that $M_0^{\mathcal{X}}$ will find such a string in polynomial time, as desired. □

Lemma 5.15. *There exist an oracle $\mathcal{O}$ and a deterministic polynomial-time oracle machine $M_1^{(-)}$ as follows. For any $n \in \mathbb{N}$, $M_1^{\mathcal{O}}(1^n)$ outputs a string $x \in \{0,1\}^n$ with $\mathsf{NP}^{\mathcal{O}}$-oracle circuit complexity at least $n^{1/2}$. Moreover, $\mathsf{PH}^{\mathcal{O}} \subseteq \mathsf{P}^{\mathcal{O}}$.*

Proof. Let $\mathcal{O}$ be the oracle of Lemma 5.13. Let $M_0^{(-)}$ be the machine and $\mathcal{X} \in \Sigma_3^{\mathsf{P},\mathcal{O}}$ the oracle of Lemma 5.14 applied to A. Since $\mathsf{PH}^{\mathcal{O}} \subseteq \mathsf{P}^{\mathcal{O}}$, there exists a deterministic polynomial-time oracle machine $N^{(-)}$ such that $N^{\mathcal{O}}$ decides $\mathcal{X}$. Let $M_1^{\mathcal{O}}$ be the machine that simulates $M_0^{(-)}$ and answers its oracle queries with the machine $N^{\mathcal{O}}$. By Lemma 5.14, for any $n \in \mathbb{N}$, $M_1^{\mathcal{O}}(1^n)$ produces a string $x \in \{0,1\}^n$ with $\mathsf{NP}^{\mathcal{O}}$-oracle circuit complexity at least $n^{1/2}$. The lemma follows. □

We will require the following lemma, which is similar to Lemma 5.4, but pertains to $\mathsf{pKt}^{\mathcal{O}}$ complexity for arbitrary oracles $\mathcal{O}$ rather than pnKt complexity.

Lemma 5.16. *There exists a probabilistic oracle algorithm $B^{(-)}$ such that, for any oracle $\mathcal{O}$, given inputs $x \in \{0,1\}^m$ and $y \in \{0,1\}^{\leq m^2}$, $B^{\mathcal{O}}(x,y)$ runs in time $2^{O(m)}$ with oracle access to $\mathcal{O}$, and*

- *if $\mathsf{pKt}_{2/3}^{\mathcal{O}}(x \mid y) \leq m/2$, then $B^{\mathcal{O}}(x,y)$ rejects with probability at least $2/3$, and*
- *if $\mathsf{pKt}_{1/3}^{\mathcal{O}}(x \mid y) > m/2$, then $B^{\mathcal{O}}(x,y)$ accepts with probability at least $2/3$.*

Proof. The algorithm $B^{\mathcal{O}}$ is as follows. Given (x,y),

1. Sample $r \sim \{0,1\}^{2^{m/2}}$.
2. For all programs Π and running times $t \in \mathbb{N}$ with $|\Pi| + \lceil \log t \rceil \leq m/2$, check if $\Pi^{\mathcal{O}}(y;r)$ outputs x within t steps. Reject if so.
3. Accept if Step 2 did not reject.

It is easy to see that $B^{\mathcal{O}}$ runs in time $2^{O(m)}$ given oracle access to $\mathcal{O}$. Correctness follows easily from the definition of $\mathsf{pKt}^{\mathcal{O}}$. □

Lemma 5.17. *There exist an oracle $\mathcal{O}$ and a constant $a \in \mathbb{N}$ such that, for every $\ell, n \in \mathbb{N}$ with $\ell \leq n$, there exist strings $x_1, ..., x_\ell \in \{0,1\}^n$ satisfying*

$$\sum_{i \in [\ell]} \mathsf{pnKt}^{\mathcal{O}}(x_i \mid x_1, ..., x_{i-1}) > \ell \cdot \frac{n}{a}$$

and

$$\mathsf{Kt}^{\mathcal{O}}(x_1, ..., x_\ell) < a \cdot n.$$

Proof. We follow the proof technique of Lemma 5.5. Let $B^{(-)}$ be the randomized oracle algorithm of Lemma 5.16. Recall that, on inputs $x \in \{0,1\}^n$ and $y \in \{0,1\}^{\leq n^2}$, $B^{(-)}$ runs in time at most $2^{O(n)}$. For a constant $b \in \mathbb{N}$, $x \in \{0,1\}^n$, and $y \in \{0,1\}^{\leq n^2}$, let $C_{(x,y)}^{(-)} : \{0,1\}^{2^{bn}} \to \{0,1\}$ be the oracle circuit of size at most 2^{bn} that treats its input as internal randomness and simulates $B^{(-)}(x,y)$.

Let $\mathcal{O}$ be the oracle and M_1 the machine of Lemma 5.15. Let $c, d \in \mathbb{N}$ be as in Theorem 2.26 applied with $\varepsilon = 1/2$, and let $z \in \{0,1\}^{2^{bc \cdot n}}$ be the output of $M_1^{\mathcal{O}}(1^{2^{bc \cdot n}})$. That is, z has $\mathsf{NP}^{\mathcal{O}}$-oracle circuit complexity greater than $2^{bc \cdot n/2}$. Let $G_z : \{0,1\}^{bd \cdot n} \to \{0,1\}^{2^{bn}}$ be the PRG of Theorem 2.26. In particular, G_z is computable in time at most $\mathsf{poly}(2^{bc \cdot n}, bd \cdot n) = 2^{O(n)}$ and $(1/2^{bn})$-fools $C_{(x,y)}^{\mathsf{NP}^{\mathcal{O}}}$.

Consider the following procedure A.

On input $n \in \mathbb{N}$, $y \in \{0,1\}^{\leq n^2}$, and $z \in \{0,1\}^{2^{bc \cdot n}}$, for each $x \in \{0,1\}^n$, compute

$$\rho_x = \mathop{\mathbf{Pr}}_{\sigma \sim \{0,1\}^{bd \cdot n}}[C_{(x,y)}^{\mathsf{NP}^{\mathcal{O}}}(G_z(\sigma)) = 1].$$

Output the first x such that $\rho_x > \frac{1}{3} + \frac{1}{10}$.

It is easy to see that, with z as above, $A(n, y, z)$ runs in time at most $2^{O(n)}$, using an $\mathsf{NP}^{\mathcal{O}}$ oracle to simulate $C^{\mathsf{NP}^{\mathcal{O}}}$. Moreover, by the guarantee of Lemma 5.16, if $\mathsf{pKt}^{\mathsf{NP}^{\mathcal{O}}}(x \mid y) \leq n/2$, then $B^{\mathsf{NP}^{\mathcal{O}}}$ accepts with probability at most $1/3$. Therefore, A must output a string $x \in \{0,1\}^n$ such that $\mathsf{pKt}^{\mathsf{NP}^{\mathcal{O}}}(x \mid y) > n/2$.

Now, define

- $x_1 = A(n, \epsilon, z)$, where ε is the empty string;
- $x_i = A(n, (x_1, ..., x_{i-1}), z)$ for $i \in \{2, ..., \ell\}$.

We claim that the strings $x_1, ..., x_\ell \in \{0,1\}^n$ have the properties stated in the lemma. Firstly, by the correctness of A and Corollary 3.16, for some constant $k \in \mathbb{N}$, for each $i \in [\ell]$, we have

$$\mathsf{pnKt}^{\mathcal{O}}(x_i \mid x_1, ..., x_{i-1}) \geq \frac{1}{k} \cdot \mathsf{pKt}^{\mathsf{NP}^{\mathcal{O}}}(x_i \mid x_1, ..., x_{i-1}) \geq \frac{n}{2k}.$$

Thus,

$$\sum_{i \in [\ell]} \mathsf{pnKt}^{\mathcal{O}}(x_i \mid x_1, ..., x_{i-1}) \geq \ell \cdot \frac{n}{2k}.$$

Secondly, we observe that $x_1, ..., x_\ell$ can be produced uniformly by running $M_1^{\mathcal{O}}$ to produce a hard truth-table z followed by $\ell \leq n$ invocations of A. This takes deterministic time at most $2^{O(n)}$ given an $\mathsf{NP}^{\mathcal{O}}$ oracle. Thus,

$$\mathsf{Kt}^{\mathsf{NP}^{\mathcal{O}}}(x_1, ..., x_\ell) \leq O(n).$$

Since $\mathsf{PH}^{\mathcal{O}} \subseteq \mathsf{P}^{\mathcal{O}}$, for any language $L \in \mathsf{NP}^{\mathcal{O}}$, there exists a deterministic polynomial-time oracle machine $N^{\mathcal{O}}$ that decides L. Therefore, replacing oracle calls to $L \in \mathsf{NP}^{\mathcal{O}}$ by simulating $N^{\mathcal{O}}$, we obtain

$$\begin{aligned}\mathsf{Kt}^{\mathcal{O}}(x_1, ..., x_\ell) &< 2a' \cdot (\mathsf{Kt}^{\mathsf{NP}^{\mathcal{O}}}(x_1, ..., x_\ell)) \\ &\leq O(n),\end{aligned}$$

where $a' \in \mathbb{N}$ is such that $t^{a'}$ upper-bounds the running time of $N^{(-)}$ on inputs of length t.

This completes the proof. □

The following is immediate from the definitions of the complexity measures in question.

Corollary 5.18. *There exist an oracle $\mathcal{O}$ and a constant $d \in \mathbb{N}$ as follows. For every $\ell, n \in \mathbb{N}$ with $\ell \leq n$, there exist strings $x_1, ..., x_\ell \in \{0,1\}^n$ such that, for every complexity measure $\mu \in \{\mathsf{rKt}, \mathsf{pKt}, \mathsf{nKt}, \mathsf{rnKt}, \mathsf{pnKt}\}$,*

$$\sum_{i \in [\ell]} \mu^{\mathcal{O}}(x_i \mid x_1, ..., x_{i-1}) > \ell \cdot \frac{n}{d}$$

and

$$\mu^{\mathcal{O}}(x_1, ..., x_\ell) < d \cdot n.$$

Given the above, we are ready to prove Theorem 5.12 as stated at the beginning of this section. The argument is quite similar to that of Theorem 5.3. We include a proof for completeness.

Proof of Theorem 5.12. We start with the following claim.

Claim 5.19. *Let $\mathcal{O}$ be the oracle of Corollary 5.18. For some constant $c \in \mathbb{N}$, for all sufficiently large $m \in \mathbb{N}$, there exist $j \in \mathbb{N}$ with $m \leq j \leq c^2 \cdot m$ and strings $u \in \{0,1\}^j$ and $v \in \{0,1\}^m$ such that*

$$\mu^{\mathcal{O}}(u, v) < \mu^{\mathcal{O}}(u) + \mu^{\mathcal{O}}(v \mid u) - \frac{m}{c}.$$

Proof of Claim 5.19. Toward a contradiction, suppose that for any constant $c \in \mathbb{N}$, there exist infinitely many $m \in \mathbb{N}$ such that for all $j \in \mathbb{N}$ with $m \leq j \leq c^2 \cdot m$ and all $u \in \{0,1\}^j$ and $v \in \{0,1\}^m$, it holds that

$$\mu^{\mathcal{O}}(u, v) \geq \mu^{\mathcal{O}}(u) + \mu^{\mathcal{O}}(v \mid u) - \frac{m}{c}. \tag{22}$$

Let d be the constant from Corollary 5.18, let $c = 2d$, and let m be as in Equation (22). Let $\ell := c^2$, and let $x_1, ..., x_\ell \in \{0,1\}^m$ be the strings guaranteed by Corollary 5.18. That is,

$$\begin{aligned}\sum_{i \in [\ell]} \mu^{\mathcal{O}}(x_i \mid x_1, ..., x_{i-1}) > \ell \cdot \frac{m}{d} \\ = \frac{(2d)^2 \cdot m}{d} = 4d \cdot m\end{aligned}$$

and

$$\mu^{\mathcal{O}}(x_1, ..., x_\ell) < d \cdot m. \tag{23}$$

Consider $u = (x_1, ..., x_{\ell-1}) \in \{0,1\}^{(\ell-1)\cdot m}$ and $v = x_\ell \in \{0,1\}^m$. By Equation (22), we have

$$\mu^{\mathcal{O}}(x_1, ..., x_\ell) \geq \mu^{\mathcal{O}}(x_1, ..., x_{\ell-1}) + \mu^{\mathcal{O}}(x_\ell \mid x_1, ..., x_{\ell-1}) - \frac{m}{c}.$$

Repeating inductively, we obtain

$$\begin{aligned}\mu^{\mathcal{O}}(x_1, ..., x_\ell) &\geq \sum_{i \in [\ell]} \mu^{\mathcal{O}}(x_i \mid x_1, ..., x_{i-1}) - \ell \cdot \frac{m}{c} \\ &\geq 4d \cdot m - 2d \cdot m \\ &= 2d \cdot m,\end{aligned}$$

contradicting Equation (23). □

Now, let the constant c be as in Claim 5.19. For any sufficiently large $n \in \mathbb{N}$, let $m = \lfloor n/c^4 \rfloor$. By Claim 5.19, there exists $j \in \mathbb{N}$ with $m \leq j \leq c^2 \cdot m < n$ and strings $u \in \{0,1\}^j$ and $v \in \{0,1\}^m$ violating SoI. Then, define n-length strings

$$x = u1^{n-j} \text{ and } y = v1^{n-m}.$$

Observe that

$$\mu^{\mathcal{O}}(x, y) \leq \mu^{\mathcal{O}}(u, v) + O(\log n)$$

and similarly

$$\mu^{\mathcal{O}}(u) \leq \mu^{\mathcal{O}}(x) + O(\log n)$$

and

$$\mu^{\mathcal{O}}(v \mid u) \leq \mu^{\mathcal{O}}(y \mid x) + O(\log n).$$

It follows that

$$\begin{aligned}\mu^{\mathcal{O}}(x,y) &\le \mu^{\mathcal{O}}(u,v) + O(\log n)\\ &< \mu^{\mathcal{O}}(u) + \mu^{\mathcal{O}}(v \mid u) - \frac{m}{c} + O(\log n)\\ &\le \mu^{\mathcal{O}}(x) + \mu^{\mathcal{O}}(y \mid x) - \frac{m}{c} + O(\log n)\\ &\le \mu^{\mathcal{O}}(x) + \mu^{\mathcal{O}}(y \mid x) - \varepsilon \cdot n. && \text{(choosing } \varepsilon := 1/c^6\text{)}\end{aligned}$$

This completes the proof of the theorem. □

6 Average-Case Versus Worst-Case Complexity

In this section, we prove Theorem 1.5, which is restated below.

Theorem 1.5. *The following statements are equivalent*:

1. (Exclusion of PH-Heuristica). $\mathsf{DistPH} \subseteq \mathsf{AvgBPP} \implies \mathsf{PH} \subseteq \mathsf{BPP}$.
2. (Meta-Complexity of $\mathsf{pK}^{\mathsf{PH}}$). $\mathsf{Gap\text{-}MINpKT}^{\mathsf{PH}} \in \mathsf{prBPP} \implies \mathsf{Mild\text{-}Gap\text{-}MINpKT}^{\mathsf{PH}} \in \mathsf{prBPP}$.

Theorem 1.5 follows directly from the following two theorems using that $\mathsf{NP} \subseteq \mathsf{BPP}$ implies $\mathsf{PH} \subseteq \mathsf{BPP}$.

Theorem 6.1. *The following holds.*

$$\mathsf{NP} \subseteq \mathsf{BPP} \iff \mathsf{Mild\text{-}Gap\text{-}MINpKT}^{\mathsf{PH}} \in \mathsf{prBPP}.$$

Theorem 6.2. *The following holds.*

$$\mathsf{DistPH} \subseteq \mathsf{AvgBPP} \iff \mathsf{Gap\text{-}MINpKT}^{\mathsf{PH}} \in \mathsf{prBPP}.$$

In the remainder of this section, we show Theorem 6.1 and Theorem 6.2. We prove Theorem 6.1 in Section 6.1. The forward direction of Theorem 6.1 is established by Lemma 6.11, which is proved in Section 6.2. The backward direction follows from Lemma 6.13, with the proof provided in Section 6.3.

6.1 Characterizing Easiness of NP via Easiness of $\mathsf{Mild\text{-}Gap\text{-}MINpKT}^{\mathsf{PH}}$

In this section, we prove Theorem 6.1. The theorem follows directly from Lemma 6.3 and Lemma 6.4, which are proved in the subsections below.

6.1.1 The Easy Direction

Lemma 6.3. *If* $\mathsf{NP} \subseteq \mathsf{BPP}$, *then* $\mathsf{Mild\text{-}Gap\text{-}MINpKT}^{\mathsf{PH}} \in \mathsf{prBPP}$.

Proof. It is easy to see that, for any oracle $\mathcal{O}$, we have $\mathsf{Mild\text{-}Gap\text{-}MINpKT}^{\mathcal{O}} \in \mathsf{prAM}^{\mathcal{O}}$.[8] Also, if $\mathsf{NP} \subseteq \mathsf{BPP}$, then $\mathsf{PH} \subseteq \mathsf{BPP}$. It follows that $\mathsf{prAM}^{\mathsf{PH}} \subseteq \mathsf{prAM}^{\mathsf{BPP}} = \mathsf{prAM} \subseteq \mathsf{prBPP}^{\mathsf{NP}} \subseteq \mathsf{prBPP}^{\mathsf{BPP}} = \mathsf{prBPP}$. □

[8] This is true even for $\mathsf{MINpKT}^{\mathcal{O}}$.

6.1.2 The Hard Direction

Lemma 6.4. *If* $\mathsf{Mild\text{-}Gap\text{-}MINpKT}^{\mathsf{SAT}} \in \mathsf{prBPP}$*, then* $\mathsf{NP} \subseteq \mathsf{BPP}$.

The proof of Lemma 6.4 follows the approach in [HLO25], which shows that if $\mathsf{MINnKT} \in \mathsf{BPP}$, then $\mathsf{NP} \subseteq \mathsf{BPP}$. Here, we sketch the proof and highlight the key steps and modifications.

We begin by describing the approach at a high level. Let $L \in \mathsf{NP}$, and let $x \in \{0,1\}^n$ be an input instance. Our goal is to find a witness for x. To this end, suppose $x \in L$, and let A_x denote the set of L-witnesses for x. We aim to establish the following: for every $y \in A_x$,

$$\mathsf{pK}^{\mathsf{poly}(n)}(y \mid x) \leq \log |A_x| + O(\log n). \tag{24}$$

The above is a consequence of a result known as *language compression* (for pK^t), which states that every witness $y \in A_x$ can be compressed to approximately $\log |A_x|$ bits.

Note that if $\mathsf{pK}^{\mathsf{poly}(n)}(y \mid x) \leq k$, then by the definition of pK^t, if we randomly sample a string $r \sim \{0,1\}^{\mathsf{poly}(n)}$ and a program $\Pi \sim \{0,1\}^{\leq k}$, and output $\Pi(y; r)$, we obtain y with probability at least $1/(2^k \cdot O(n))$. Then, by Equation (24), for every $y \in A_x$, the above procedure samples y with probability at least $1/(|A_x| \cdot O(n))$. Since this holds for each of the $|A_x|$ elements in A_x, it follows that we obtain some string in A_x with probability at least $1/O(n)$. By repeating this procedure, we can obtain a witness for x with high probability.

It is not known whether pK^t admits a language compression property as stated in Equation (24). However, it was shown in [HLO25] that its nondeterministic variant, pnK^t, indeed admits such a property. Since $\mathsf{pK}^{t,\mathsf{SAT}}$ is a more powerful notion than pnK^t (Lemma 3.15), it follows that $\mathsf{pK}^{t,\mathsf{SAT}}$ also admits this language compression property. Thus, it suffices to show that if $\mathsf{Mild\text{-}Gap\text{-}MINpKT}^{\mathsf{SAT}} \in \mathsf{prBPP}$, then for every $x, y \in \{0,1\}^n$ and every t that is a sufficiently large polynomial in n,

$$\mathsf{pK}^{\mathsf{poly}(t)}(y \mid x) \leq \mathsf{pK}^{\mathsf{poly}(n),\mathsf{SAT}}(y \mid x) + O(\log t). \tag{25}$$

We first note that, following previous work, one can establish a weaker form of Equation (25), which states that

$$\mathsf{pK}^{\mathsf{poly}(t)}(y \mid x) \leq \mathsf{pK}^{\mathsf{poly}(n),\mathsf{SAT}}(y \mid x) + \mathsf{pK}^t(x) - \mathsf{pK}^{\mathsf{poly}(t)}(x) + O(\log t). \tag{26}$$

Note that this weaker bound is not sufficient for our purposes, since the term $\mathsf{pK}^t(x) - \mathsf{pK}^{\mathsf{poly}(t)}(x)$, known as the (time-bounded) computational depth, can be large for some x. Nevertheless, we sketch how Equation (26) can be shown under the assumption that $\mathsf{Mild\text{-}Gap\text{-}MINpKT}^{\mathsf{SAT}}$ is easy.

The idea is to show both an upper bound and a lower bound for the quantity

$$\mathsf{pK}^{t+\mathsf{poly}(n),\mathsf{SAT}}(x, y).$$

Roughly speaking, we aim to show that:

- **(Upper bound)** $\mathsf{pK}^{t+\mathsf{poly}(n),\mathsf{SAT}}(x, y) \lesssim \mathsf{pK}^{\mathsf{poly}(n),\mathsf{SAT}}(y \mid x) + \mathsf{pK}^{t,\mathsf{SAT}}(x)$.
- **(Lower bound)** $\mathsf{pK}^{t+\mathsf{poly}(n),\mathsf{SAT}}(x, y) \gtrsim \mathsf{pK}^{\mathsf{poly}(t),\mathsf{SAT}}(y \mid x) + \mathsf{pK}^{\mathsf{poly}(t)}(x)$.

It is easy to combine the two inequalities above to obtain Equation (26).

The upper bound is straightforward. The lower bound can be established under the assumption that $\mathsf{Mild\text{-}Gap\text{-}MINpKT}^{\mathsf{SAT}}$ is easy. For simplicity, let us assume that we have a deterministic polynomial-time algorithm A that given $(z, 1^s, 1^t)$ decides whether $\mathsf{pK}^{t,\mathsf{SAT}} \leq s$.

Let $\mathsf{DP}_k \colon \{0,1\}^n \times \{0,1\}^{nk} \to \{0,1\}^{nk+k}$ be the direct product generator defined in Definition 2.28, where

$$k := \mathsf{pK}^{t^c}(y \mid x) - c \cdot \log t,$$

for some large constant $c > 1$ to be specified later. We then consider the following two distributions:

$$\mathcal{D}_1 := (x, \mathsf{DP}_k(y, \mathcal{U}_{nk})),$$
$$\mathcal{D}_2 := (x, \mathcal{U}_{nk+k}).$$

The goal is to show that $\mathcal{D}_1$ and $\mathcal{D}_2$ are indistinguishable under the algorithm $A(-, 1^s, 1^t)$ for some appropriate s. The reason for doing this is that, assuming Mild-Gap-MINpKT$^{\mathsf{SAT}}$ is easy, one can show a weak symmetry of information for $\mathsf{pK}^{t,\mathsf{SAT}}$ [Hir21] (see also [Ila23]), which states that, with high probability over $w \sim \{0,1\}^{nk+k}$, we have

$$\mathsf{pK}^{t,\mathsf{SAT}}(x, w) \geq \mathsf{pK}^{\mathsf{poly}(t),\mathsf{SAT}}(x) + nk + k - O(\log t).$$

Therefore, with high probability (over $\mathcal{D}_2$), $A(\mathcal{D}_2, 1^s, 1^t) = 0$ for $s := \mathsf{pK}^{\mathsf{poly}(t),\mathsf{SAT}}(x) + nk + k - O(\log t)$.

If $\mathcal{D}_1$ is indeed indistinguishable from $\mathcal{D}_2$, then, also, with high probability (over $\mathcal{D}_1$), we have $A(\mathcal{D}_1, 1^s, 1^t) = 0$, which implies that there exists some $z \in \{0,1\}^{nk}$ such that

$$\mathsf{pK}^{t,\mathsf{SAT}}(x, \mathsf{DP}_k(y, z)) \geq s.$$

Also, since $\mathsf{DP}_k(y, z)$ is efficiently computable given y and z, we have

$$\mathsf{pK}^{t,\mathsf{SAT}}(x, \mathsf{DP}_k(y, z)) \leq \mathsf{pK}^{t+\mathsf{poly}(n),\mathsf{SAT}}(x, y) + nk + O(\log t).$$

Combining the above two inequalities, we get

$$\begin{aligned} \mathsf{pK}^{t+\mathsf{poly}(n),\mathsf{SAT}}(x, y) &\geq s - nk + k - O(\log t) \\ &= \mathsf{pK}^{\mathsf{poly}(t),\mathsf{SAT}}(x) + k - O(\log t) \\ &= \mathsf{pK}^{\mathsf{poly}(t),\mathsf{SAT}}(x) + \mathsf{pK}^{\mathsf{poly}(t)}(y \mid x) - O(\log t), \end{aligned}$$

which gives the desired lower bound.

Now, let us explain why $\mathcal{D}_1$ and $\mathcal{D}_2$ are indistinguishable. Indeed, if $A(-, 1^s, 1^t)$ could distinguish $\mathcal{D}_1$ from $\mathcal{D}_2$, then by the fact that A is polynomial-time computable and by the reconstruction property of DP_k (Lemma 2.29), we would obtain

$$\mathsf{pK}^{\mathsf{poly}(t)}(y \mid x) \leq k + O(\log t),$$

which contradicts our choice of k be letting c be a sufficiently large constant.

Next, let us return to Equation (26). It turns out that if we could show a more fine-grained form:

$$\mathsf{pK}^{\mathsf{poly}(t)}(y \mid x) \leq \mathsf{pK}^{\mathsf{poly}(n),\mathsf{SAT}}(y \mid x) + \mathsf{pK}^{t,\mathsf{SAT}}(x) - \mathsf{pK}^{t+p(n),\mathsf{SAT}}(x) + O(\log n), \tag{27}$$

where $p(n)$ is a polynomial independent of t, then we could aim to conclude the proof by showing that

$$\mathsf{pK}^{t,\mathsf{SAT}}(x) - \mathsf{pK}^{t+p(n),\mathsf{SAT}}(x) = O(\log n).$$

While this may not hold for a particular value of t, one can argue, via a telescoping sum trick, that such a bound holds for some t' that is not too large. More concretely, for any fixed time bound t, let $t_i = t + i \cdot p(n)$ for $i \in \{0, 1, \ldots, n\}$. Note that $\mathsf{pK}^{t_0,\mathsf{SAT}}(x) - \mathsf{pK}^{t_n,\mathsf{SAT}}(x) \leq 2n$, since any n-bit string x can be described using at most $2n$ bits. Therefore,

$$\sum_{i=0}^{n-1} \left(\mathsf{pK}^{t_i,\mathsf{SAT}}(x) - \mathsf{pK}^{t_{i+1},\mathsf{SAT}}(x)\right) = \mathsf{pK}^{t_0,\mathsf{SAT}}(x) - \mathsf{pK}^{t_n,\mathsf{SAT}}(x) \leq 2n,$$

which implies that there exists an index i such that $\mathsf{pK}^{t_i,\mathsf{SAT}}(x) - \mathsf{pK}^{t_{i+1},\mathsf{SAT}}(x) \leq 2$. In particular, for some time bound $t' \leq t + n \cdot p(n)$, we have $\mathsf{pK}^{t',\mathsf{SAT}}(x) - \mathsf{pK}^{t'+p(n),\mathsf{SAT}}(x) = O(\log n)$, which would be sufficient for our purpose.

Unfortunately, it is not clear how to prove Equation (27) exactly. As illustrated earlier, a key difficulty lies in the use of weak symmetry of information, which introduces an overhead of the form $\mathsf{poly}(t)$ rather than an *additive* overhead of the form $t + p(n)$.

Here, we follow the approach developed in [HLO25] which incorporates the ideas described above and suffices to prove Lemma 6.4. A key ingredient is the following new notion of time-bounded Kolmogorov complexity.

Definition 6.5 (ℓ-$\mathsf{pK}^t_{\lambda,\gamma}$)**.** For any $t, \ell \in \mathbb{N}$, $\lambda, \gamma \in (0,1)$, oracle $\mathcal{O}$, and string $x \in \{0,1\}^*$, we define

$$\ell\text{-}\mathsf{pK}^{t,\mathcal{O}}_{\lambda,\gamma}(x) = \min\left\{ s \in \mathbb{N} \;\middle|\; \Pr_{\hat{r} \sim \mathcal{U}_\ell}\left[\mathsf{pK}^{t,\mathcal{O}}_{\lambda}(x \circ \hat{r}) \leq s + \ell\right] \geq \gamma \right\}.$$

It can be shown that ℓ-$\mathsf{pK}^t_{\lambda,\gamma}$ is a well-defined measure that satisfies several basic properties one would expect from a Kolmogorov complexity measure; see [HLO25, Section 4.2.1]. Moreover, if the mild-gap version of approximating pK^t is easy, then so is approximating ℓ-$\mathsf{pK}^t_{\lambda,\gamma}$.

Definition 6.6 (Mild-Gap-MINℓ-pKT)**.** For an oracle $\mathcal{O}$ and $\rho\colon \mathbb{N} \to \mathbb{N}$, we define the promise language $\mathsf{Mild\text{-}Gap}_\rho\text{-}\mathsf{MIN}\ell\text{-}\mathsf{pKT}^{\mathcal{O}} = (\mathsf{YES}, \mathsf{NO})$ as

$$\begin{cases} \mathsf{YES} = \left\{(x, 1^s, 1^t, 1^\ell, 1^a, 1^b, 1^c, 1^d) \mid \mathsf{pK}^{t,\mathcal{O}}_{b/a,d/c}(x) \leq s\right\}, \\ \mathsf{NO} = \left\{(x, 1^s, 1^t, 1^a, 1^b) \mid \mathsf{pK}^{t+\rho(|x|+a+b+c+d),\mathcal{O}}_{(b-1)/a,(d-1)/c}(x) \geq s + \log\rho(t+a+b+c+d)\right\}. \end{cases}$$

We say that $\mathsf{Gap}_\rho\text{-}\mathsf{MIN}\ell\text{-}\mathsf{pKT}^{\mathcal{O}} \in \mathsf{prBPP}$ if there exists a constant $g > 0$ such that $\mathsf{Gap}_\rho\text{-}\mathsf{MIN}\ell\text{-}\mathsf{pKT}^{\mathcal{O}} \in \mathsf{prBPP}$ with $\rho(n) = n^g$.

Lemma 6.7. *For any oracle $\mathcal{O}$, if* $\mathsf{Mild\text{-}Gap\text{-}MINpKT}^{\mathcal{O}} \in \mathsf{prBPP}$*, then* $\mathsf{Mild\text{-}Gap\text{-}MIN}\ell\text{-}\mathsf{pKT}^{\mathcal{O}} \in \mathsf{prBPP}$.

Proof. The proof is adapted from that of [HLO25, Lemma 69], which shows a similar result for the case of pnK^t and ℓ-pnK^t without the mild gap. For simplicity, we assume $\mathcal{O} = \emptyset$; it is straightforward to verify that the proof also holds for any oracle $\mathcal{O}$.

Let A be a probabilistic polynomial-time algorithm that decides $\mathsf{Mild\text{-}Gap}_\rho\text{-}\mathsf{MINpKT}$ with error at most $1/3$, for some polynomial ρ. Let A^m denote the algorithm that runs A m times and returns the majority vote. By the Chernoff bound, the error of A^m is bounded by $2^{-m/18}$.

Suppose we are given an instance

$$z := (x \in \{0,1\}^n, 1^s, 1^t, 1^\ell, 1^a, 1^b, 1^c, 1^d)$$

of $\mathsf{Mild\text{-}Gap}_\rho\text{-}\mathsf{MIN}\ell\text{-}\mathsf{pKT}$.

On the one hand, if z is a yes-instance, then ℓ-$\mathsf{pK}^t_{b/a,d/c}(x) \leq s$, and by the definition of ℓ-pK, we have

$$\Pr_{\hat{r} \sim \mathcal{U}_\ell}\left[\mathsf{pK}^t_{b/a}(x \circ \hat{r}) \leq s + \ell\right] \geq \frac{d}{c}.$$

Then by the correctness of A, we get

$$\Pr_{\hat{r} \sim \mathcal{U}_\ell}\left[\Pr_{A}\left[A^{\lceil 54 \log c \rceil}(x \circ \hat{r}, 1^{s+\ell}, 1^t, 1^a, 1^b) = 1\right] \geq 1 - \frac{1}{c^3}\right] \geq \frac{d}{c},$$

which, by a union bound, yields

$$\mathop{\mathbf{Pr}}_{\hat{r}\sim\mathcal{U}_\ell, A}\Big[A^{\lceil 54\log c\rceil}(x\circ\hat{r}, 1^{s+\ell}, 1^t, 1^a, 1^b) = 1\Big] \geq \frac{d}{c} - \frac{1}{c^3}. \tag{28}$$

On the other hand, if z is a no-instance, then

$$\ell\text{-}\mathsf{pK}^{t+\rho(n+\ell+a+b+c+d)}_{(b-1)/a,(d-1)/c}(x) > s + \log\rho\big(t+\ell+a+b+c+d\big),$$

and hence

$$\mathop{\mathbf{Pr}}_{\hat{r}\sim\mathcal{U}_\ell}\Big[\mathsf{pK}^{t+\rho(n+\ell+a+b+c+d)}_{(b-1)/a}(x\circ\hat{r}) \leq s+\ell+\log\rho\big(t+\ell+a+b+c+d\big)\Big] < \frac{d}{c} - \frac{1}{c}.$$

Note that the above implies

$$\mathop{\mathbf{Pr}}_{\hat{r}\sim\mathcal{U}_\ell}\Big[\mathsf{pK}^{t+\rho(n+a+b)}_{(b-1)/a}(x\circ\hat{r}) \leq s+\ell+\log\rho(t+a+b)\Big] < \frac{d}{c} - \frac{1}{c}.$$

Again, by the correctness of A, the above implies

$$\mathop{\mathbf{Pr}}_{\hat{r}\sim\mathcal{U}_\ell}\Big[\mathop{\mathbf{Pr}}_{A}\Big[A^{\lceil 54\log c\rceil}(x\circ\hat{r}, 1^{s+\ell}, 1^t, 1^a, 1^b) = 1\Big] > \frac{1}{c^3}\Big] < \frac{d}{c} - \frac{1}{c},$$

which, by an averaging argument, gives

$$\mathop{\mathbf{Pr}}_{\hat{r}\sim\mathcal{U}_\ell, A}\Big[A^{\lceil 54\log c\rceil}(x\circ\hat{r}, 1^{s+\ell}, 1^t, 1^a, 1^b) = 1\Big] < \frac{d}{c} - \frac{1}{c} + \frac{1}{c^3}. \tag{29}$$

Comparing Equation (28) and Equation (29), we obtain a polynomial-time algorithm that distinguishes yes- and no-instances with a gap of at least $1/c - 2/c^3 \geq 1/(2c)$. Such a gap can be amplified by repeating the algorithm $\mathsf{poly}(c)$ times. □

Next, assume that $\mathsf{Mild\text{-}Gap\text{-}MIN}\ell\text{-}\mathsf{pKT}^{\mathsf{SAT}} \in \mathsf{prBPP}$. We will show a result that informally states the following: for every $x, y \in \{0,1\}^n$ and for appropriate settings of parameters ℓ and t, we have

$$\mathsf{pK}^{\mathsf{poly}(t)}(y \mid x) \lesssim \mathsf{pK}^{\mathsf{poly}(n),\mathsf{SAT}}(y \mid x) + \ell\text{-}\mathsf{pK}^{t,\mathsf{SAT}}(x) - (\ell+\mathsf{poly}(n))\text{-}\mathsf{pK}^{t+\mathsf{poly}(n),\mathsf{SAT}}(x).$$

Observe the similarity between the above and Equation (27). The precise technical lemma is as follows:

Lemma 6.8. *Suppose* $\mathsf{Mild\text{-}Gap\text{-}MIN}\ell\text{-}\mathsf{pKT}^{\mathsf{SAT}} \in \mathsf{prBPP}$. *Then there exists a polynomial* p *and an integer* $N_0 \in \mathbb{N}^+$ *such that the following holds. For any* $n, m, \tau, \ell, t, b, d \in \mathbb{N}^+$ *and any strings* $x \in \{0,1\}^n$ *and* $y \in \{0,1\}^m$ *satisfying the following constraints*:

- $2/3 < b/n^3, d/n^3 < 1$,
- $N_0 \leq n, m \leq \tau$,
- $\tau \leq \ell \leq \tau^4$,
- $t \geq 2p(\tau)$,

we have

$$\mathsf{pK}^{p(t)}(y \mid x) \leq \mathsf{pK}^{\tau,\mathsf{SAT}}_{1-1/n^3}(y \mid x) + \ell\text{-}\mathsf{pK}^{t-p(\tau),\mathsf{SAT}}_{b/n^3,(d+1)/n^3}(x) - (\ell+\tau^3)\text{-}\mathsf{pK}^{t+p(\tau),\mathsf{SAT}}_{(b-1)/n^3,(d-2)/n^3}(x) + \log p(t).$$

Proof Sketch. The proof closely follows that of [HLO25, Lemma 74], which establishes a similar result for the measure ℓ-pnK. The main difference is that the former assumes the easiness of the non-gap version, whereas here we assume the easiness of the mild-gap version. We provide a proof sketch below.

Let $x \in \{0,1\}^n$ and $y \in \{0,1\}^m$. Similar to the earlier discussion, the idea is to show both upper and lower bounds for the quantity

$$\ell\text{-}\mathsf{pK}^{t,\mathsf{SAT}}_{b/n^3,\,d/n^3}(x, y).$$

Upper Bound.

Claim 6.9. *There exists a polynomial p_1 such that*

$$\ell\text{-}\mathsf{pK}^{t,\mathsf{SAT}}_{b/n^3,\,d/n^3}(x,y) \le \mathsf{pK}^{\tau,\mathsf{SAT}}_{1-1/n^3}(y \mid x) + \ell\text{-}\mathsf{pK}^{t-p_1(\tau)}_{b/n^3,\,(d+1)/n^3}(x) + \log p_1(\tau).$$

Proof Sketch of Claim 6.9. Intuitively, the claim states that if there is a program that outputs x in time $t - \mathsf{poly}(\tau)$, and a program that outputs y given x in time τ, then one can obtain a program that outputs (x,y) in time t. This intuition was shown to hold for ℓ-pK in [HLO25]. See the proof of [HLO25, Lemma 72] for details. $\diamond$

Lower Bound.

Claim 6.10. *There exists a polynomial p_2 such that*

$$\ell\text{-}\mathsf{pK}^{t,\mathsf{SAT}}_{b/n^3,\,d/n^3}(x,y) \ge (\ell+\tau^3)\text{-}\mathsf{pK}^{t+p_2(\tau),\mathsf{SAT}}_{(b-1)/n^3,\,(d-2)/n^3}(x) - \mathsf{pK}^{p_2(t)}(y \mid x) - \log p_2(t).$$

Proof of Claim 6.10. Let $\mathsf{DP}_k \colon \{0,1\}^m \times \{0,1\}^{mk} \to \{0,1\}^{mk+k}$ be the direct product generator (Definition 2.28), where

$$k := \mathsf{pK}^{q(t)}(y \mid x) - \log q(t),$$

and q is a sufficiently large polynomial specified later.

As in the previous case, we consider two distributions:

$$\mathcal{D}_1 := \big(x, \mathsf{DP}_k(y, \mathcal{U}_{mk})\big),$$
$$\mathcal{D}_2 := \big(x, \mathcal{U}_{mk+k}\big).$$

We aim to show that $\mathcal{D}_1$ and $\mathcal{D}_2$ are indistinguishable, and that with high probability over an element from $\mathcal{D}_2$, its complexity is large. To show the latter, we again use weak symmetry of information. However, the key difference here is that we will be able to use a *highly efficient* weak symmetry of information for the notion of ℓ-pK. More specifically, we have the following: For any $n, m', k, t', \ell \in \mathbb{N}^+$, any $\lambda, \gamma, \alpha \in (0,1)$ satisfying $\alpha < \gamma$, and any string $x \in \{0,1\}^n$, we have

$$\Pr_{z \sim \mathcal{U}_{m'}}\left[\ell\text{-}\mathsf{pK}^{t',\mathsf{SAT}}_{\lambda,\gamma}(x \circ z) \ge (\ell+m')\text{-}\mathsf{pK}^{t',\mathsf{SAT}}_{\lambda,\gamma-\alpha}(x) + m'\right] \ge \alpha. \tag{30}$$

The proof of this result appears in [HLO25, Lemma 71].

Now, for simplicity, let us assume that we have a deterministic polynomial-time algorithm A that solves $\mathsf{Mild\text{-}Gap}_\rho\text{-}\mathsf{MIN}\ell\text{-}\mathsf{pKT}^{\mathsf{SAT}}$ for some polynomial ρ. Let

$$t' := t + q(\tau) + \rho(\tau^5).$$

and

$$s := \Big((\ell+mk+k)\text{-}\mathsf{pK}^{t',\mathsf{SAT}}_{(b-1)/n^3,(d-2)/n^3}(x) + mk + k\Big) - \log \rho(t^4).$$

We get

$$\begin{aligned}
&\Pr_{w \sim \mathcal{U}_{mk+k}}\left[\ell\text{-}\mathsf{pK}^{t+q(\tau)+\rho(n+\ell+b+d+2n^3),\mathsf{SAT}}_{(b-1)/n^3,(d-1)/n^3}(x \circ w) \ge s + \log \rho(t+q(\tau)+\ell+b+d+2n^3)\right]\\
&\ge \Pr\Big[\ell\text{-}\mathsf{pK}^{t+q(\tau)+\rho(\tau^5),\mathsf{SAT}}_{(b-1)/n^3,(d-1)/n^3}(x \circ w) \ge s + \log \rho(t^4)\Big]\\
&= \Pr\Big[\ell\text{-}\mathsf{pK}^{t',\mathsf{SAT}}_{(b-1)/n^3,(d-1)/n^3}(x \circ w) \ge (\ell+mk+k)\text{-}\mathsf{pK}^{t',\mathsf{SAT}}_{(b-1)/n^3,(d-2)/n^3}(x) + mk + k\Big]\\
&\ge \frac{1}{n^3}. \qquad \text{(by Equation (30) with } \alpha = 1/n^3 \text{ and } \lambda = (b-1)/n^3\text{)}
\end{aligned}$$

In fact, for the first inequality, we need the notion ℓ-pK^t to be monotone in t, meaning that ℓ-pK^t decreases as t increases (see [HLO25, Lemma 66]).

Note that the above implies that, with probability at least $1/n^3$ over $\mathcal{D}_2$,

$$A(\mathcal{D}_2, 1^s, 1^{t+q(\tau)}, 1^{\ell}, 1^{n^3}, 1^b, 1^{n^3}, 1^d) = 0.$$

For the sake of contradiction, suppose the procedure $A(-, 1^s, 1^{t+q(\tau)}, 1^{\ell}, 1^{n^3}, 1^b, 1^{n^3}, 1^d)$ can $(1/n^4)$-distinguish $\mathcal{D}_1$ and $\mathcal{D}_2$. Then, by the fact that A is a polynomial-time algorithm and by the reconstruction property of DP_k (Lemma 2.29), we would obtain

$$\mathsf{pK}^{\mathsf{poly}(t)}(y \mid x) \leq k + O(\log t),$$

which contradicts our choice of k, provided that q is chosen to be a sufficiently large polynomial.

In other words, we have that

$$A(\mathcal{D}_1, 1^s, 1^{t+q(\tau)}, 1^{\ell}, 1^{n^3}, 1^b, 1^{n^3}, 1^d) = 0$$

with probability at least $1/n^3 - 1/n^4 > 0$. It follows that there exists $z \in \{0,1\}^{mk}$ such that

$$\ell\text{-}\mathsf{pK}^{t+q(\tau),\mathsf{SAT}}_{b/n^3,d/n^3}(x, \mathsf{DP}_k(y,z)) \geq s. \tag{31}$$

Moreover, it can also be shown (see [HLO25, Lemma 70]) that, for q a sufficiently large polynomial,

$$\ell\text{-}\mathsf{pK}^{t+q(\tau),\mathsf{SAT}}_{b/n^3,d/n^3}(x, \mathsf{DP}_k(y,z)) \leq \ell\text{-}\mathsf{pK}^{t,\mathsf{SAT}}_{b/n^3,d/n^3}(x,y) + mk + \log q(\tau). \tag{32}$$

Intuitively, this is because $\mathsf{DP}_k(y,z)$ is efficiently computable given y and the seed z.

It follows from Equation (31) and Equation (32) that

$$\begin{aligned}
\ell\text{-}\mathsf{pK}^{t,\mathsf{SAT}}_{b/n^3,\,d/n^3}(x,y) &\geq s - mk - \log q(\tau) \\
&= (\ell + mk + k)\text{-}\mathsf{pK}^{t',\mathsf{SAT}}_{(b-1)/n^3,(d-2)/n^3}(x) + k - \log \rho(t^4) - \log q(\tau) \\
&= (\ell + mk + k)\text{-}\mathsf{pK}^{t+q(\tau)+\rho(\tau^5),\mathsf{SAT}}_{(b-1)/n^3,\,(d-2)/n^3}(x) + \mathsf{pK}^{q(t)}(y \mid x) \\
&\qquad - \log q(t) - \log \rho(t^4) - \log q(\tau) \\
&\geq (\ell + \tau^3)\text{-}\mathsf{pK}^{t+q(\tau^5)+\rho(\tau^5),\mathsf{SAT}}_{(b-1)/n^3,\,(d-2)/n^3}(x) + \mathsf{pK}^{q(t)}(y \mid x) \\
&\qquad - \log q(t) - \log \rho(t^4) - \log q(\tau^5),
\end{aligned}$$

where the last inequality uses the monotonicity of ℓ-pK^t in ℓ (see [HLO25, Lemma 67]), which roughly states that for any sufficiently large polynomial q, we have that for every $\ell < \ell'$, $x \in \{0,1\}^*$ and $t' \in \mathbb{N}$,

$$\ell'\text{-}\mathsf{pK}^{t'+q(|x|+|y|+\ell'),\mathsf{SAT}}(x) \leq \ell\text{-}\mathsf{pK}^{t',\mathsf{SAT}}(x) + \log q(|x| + |y| + \ell').$$

Finally, we can conclude that there exists a polynomial p_2 such that

$$\ell\text{-}\mathsf{pK}^{t,\mathsf{SAT}}_{b/n^3,\,d/n^3}(x,y) \geq (\ell + \tau^3)\text{-}\mathsf{pK}^{t+p_2(\tau),\mathsf{SAT}}_{(b-1)/n^3,\,(d-2)/n^3}(x) + \mathsf{pK}^{p_2(t)}(y \mid x) - \log p_2(t).$$

This completes the proof of the lower bound. ◇

Putting It All Together. By combining the upper and lower bounds (Claim 6.9 and Claim 6.10) and rearranging, we obtain

$$\begin{aligned}\mathsf{pK}^{p_2(t)}(y \mid x) \leq \mathsf{pK}^{\tau,\mathsf{SAT}}_{1-1/n^3}(y \mid x) + \ell\text{-}\mathsf{pK}^{t-p_1(\tau)}_{b/n^3,\,(d+1)/n^3}(x) - (\ell+\tau^3)\text{-}\mathsf{pK}^{t+p_2(\tau),\mathsf{SAT}}_{(b-1)/n^3,\,(d-2)/n^3}(x) \\ + \log p_1(\tau) + \log p_2(t),\end{aligned}$$

which, by the monotonicity of ℓ-pK^t with respect to t, implies the existence of a polynomial p such that

$$\mathsf{pK}^{p(t)}(y \mid x) \leq \mathsf{pK}^{\tau,\mathsf{SAT}}_{1-1/n^3}(y \mid x) + \ell\text{-}\mathsf{pK}^{t-p(\tau)}_{b/n^3,\,(d+1)/n^3}(x) - (\ell+\tau^3)\text{-}\mathsf{pK}^{t+p(\tau),\mathsf{SAT}}_{(b-1)/n^3,\,(d-2)/n^3}(x) + \log p(t),$$

as desired. □

Proof Sketch of Lemma 6.4. The proof is essentially the same as that of [HLO25, Lemma 75]. We sketch the high-level ideas here and refer the reader to [HLO25] for full details.

Suppose $\mathsf{Mild\text{-}Gap\text{-}MINpKT}^{\mathsf{SAT}} \in \mathsf{prBPP}$, which, by Lemma 6.7, implies $\mathsf{Mild\text{-}Gap\text{-}MIN}\ell\text{-}\mathsf{pKT}^{\mathsf{SAT}} \in \mathsf{prBPP}$. Given Lemma 6.8, we can apply a slightly more sophisticated telescoping sum argument than described earlier and show that for every $x \in \{0,1\}^n$ and any τ that is a large enough polynomial in n, there exist t, ℓ, b, d that are at most $\mathsf{poly}(n)$ such that

$$\ell\text{-}\mathsf{pK}^{t-p(\tau),\mathsf{SAT}}_{b/n^3,(d+1)/n^3}(x) - (\ell+\tau^3)\text{-}\mathsf{pK}^{t+p(\tau),\mathsf{SAT}}_{(b-1)/n^3,(d-2)/n^3}(x) + \log p(t)$$

is at most $O(1)$. Again, this can be achieved because the increase in the time bounds between the two quantities is only an *additive* $\mathsf{poly}(n)$ term. As a result, for every x and y, we obtain

$$\mathsf{pK}^{\mathsf{poly}(n)}(y \mid x) \leq \mathsf{pK}^{\tau,\mathsf{SAT}}_{1-1/n^3}(y \mid x) + O(\log n). \tag{33}$$

Also, as mentioned earlier, $\mathsf{pK}^{t,\mathsf{SAT}}$ admits the language compression property (see [HLO25, Theorem 4]), that is, for every set A_x whose membership can be decided in polynomial time given x, there exists polynomial τ such that

$$\mathsf{pK}^{\tau,\mathsf{SAT}}_{1-1/n^3}(y \mid x) \leq \log |A_x| + O(\log n). \tag{34}$$

Now let $L \in \mathsf{NP}$ and consider any $x \in \{0,1\}^n$ that is a yes-instance of L. Let A_x be the set of L-witnesses for x. Then Equations (33) and (34) imply that for every $y \in A_x$,

$$\mathsf{pK}^{p(n)}(y \mid x) \leq \log |A_x| + O(\log n).$$

As described earlier, this enables an efficient sampling procedure which, given x, outputs some element in A_x with non-trivial probability $1/O(n)$. Repeating this procedure sufficiently many times then yields a witness for x with high probability. □

6.2 Worst-Case Easiness of $\mathsf{Gap\text{-}MINpKT}^{\mathsf{PH}}$ from Average-Case Easiness of PH

Lemma 6.11. *If* $\mathsf{DistPH} \subseteq \mathsf{AvgBPP}$*, then* $\mathsf{Gap\text{-}MINpKT}^{\mathsf{PH}} \in \mathsf{prBPP}$.

To show Lemma 6.11, we need the following.

Lemma 6.12. *For any oracle* $\mathcal{O}$*, if* $(\mathsf{MINKT}^{\mathcal{O}}, \mathsf{PSAMP}) \subseteq \mathsf{AvgBPP}$*, then* $\mathsf{Gap\text{-}MINpKT}^{\mathcal{O}} \in \mathsf{prBPP}$.

Proof Sketch. The proof is similar to that of [HLO25, Proposition 58], using techniques from worst-case-to-average-case reductions developed by [Hir18]. □

Proof of Lemma 6.11. Suppose $\mathsf{DistPH} \subseteq \mathsf{AvgBPP}$. Then for every oracle $\mathcal{O} \in \mathsf{PH}$, $(\mathsf{MINKT}^{\mathcal{O}}, \mathsf{PSAMP}) \in \mathsf{AvgBPP}$, since $\mathsf{MINKT}^{\mathcal{O}} \in \mathsf{PH}$. This, by Lemma 6.12, implies that $\mathsf{Gap\text{-}MINpKT}^{\mathcal{O}} \in \mathsf{prBPP}$. It follows that $\mathsf{Gap\text{-}MINpKT}^{\mathsf{PH}} \in \mathsf{prBPP}$. □

6.3 Average-Case Easiness of PH from Worst-Case Easiness of $\mathsf{Gap\text{-}MINpKT}^{\mathsf{PH}}$

In this subsection, we prove the following which shows the forward direction of Theorem 1.5.

Lemma 6.13. *If* $\mathsf{Gap\text{-}MINpKT}^{\mathsf{PH}} \in \mathsf{prBPP}$*, then* $\mathsf{DistPH} \subseteq \mathsf{AvgBPP}$.

We begin with some technical tools.

6.3.1 Technical Tools

The following is the main technical lemma for this subsection.

Lemma 6.14. *Let $k_0 \in \mathbb{N}$ and $\mathcal{O}$ be a $\Sigma_{k_0}^{\mathsf{P}}$-complete language. If there exist polynomials ρ and τ such that* $\mathsf{Gap}_\rho\text{-}\mathsf{MINpKT}^{\mathcal{O}} \in \mathsf{prBPTIME}[\tau(n)]$*, then for every $k \leq k_0$, $L \in \Sigma_k^{\mathsf{P}}$, and polynomial q, there exists a randomized algorithm A such that for every $\ell \in \mathbb{N}$ and $x \in \{0,1\}^n$, with probability at least $1 - 2^{-\ell}$ (over the internal randomness of A) both of the following hold.*

1. *$A(x, 1^\ell)$ runs in time*
$$2^{O\left(\mathsf{pK}^{q(n),\mathcal{O}}(x) - \mathsf{K}(x)\right)} \cdot \mathsf{poly}(n, \ell),$$
where the constants hidden in $O(\cdot)$ and $\mathsf{poly}(\cdot)$ depend only on $\mathcal{O}$, ρ, τ, k, L, and q.

2. *$A(x, 1^\ell) = 1$ if and only if $x \in L$.*

The rest of this subsection is devoted to proving Lemma 6.14. For this, we need the following additional tools.

Lemma 6.15 (Symmetry of Information for pK^t; see, e.g., [HKLO24, Lemma 36])**.** *Let $\mathcal{O}$ be any oracle. If* $\mathsf{Gap\text{-}MINpKT}^{\mathcal{O}} \in \mathsf{prBPP}$*, then there exist polynomials p_{Sol} and p_0 such that for all sufficiently large $x, y \in \{0,1\}^*$ and every $t \geq p_0(|x| + |y|)$,*

$$\mathsf{pK}^{p_{\mathsf{Sol}}(t)}(y \mid x) \leq \mathsf{pK}^t(x, y) - \mathsf{pK}^{p_{\mathsf{Sol}}(t)}(x) + \log p_{\mathsf{Sol}}(t).$$

Proof Sketch. To get the desired symmetry of information property, it suffices have an efficient algorithm that accepts the set of strings with low $\mathsf{K}^{\mathsf{poly}}$-complexity while rejecting a large fraction of random strings. Note that such an algorithm follows from $\mathsf{Gap\text{-}MINpKT}^{\mathcal{O}} \in \mathsf{prBPP}$ for any $\mathcal{O}$. □

Lemma 6.16. *Let $\mathcal{O}$ be any oracle. If* $\mathsf{Gap\text{-}MINpKT}^{\mathcal{O}} \in \mathsf{prBPP}$*, then there exists a polynomial p_1 such that for all $z \in \{0,1\}^*$ and $t \in \mathbb{N}$,*

$$\mathsf{pK}^{p_1(t)}(z) \leq \mathsf{pK}^{t,\mathcal{O}}(z) + \log p_1(t).$$

Proof Sketch. The proof can be easily adapted from that of [HLO25, Lemma 56]. □

We are now ready to prove Lemma 6.14.

Proof of Lemma 6.14. Let $k_0 \in \mathbb{N}$ and let $\mathcal{O}$ be a $\Sigma_{k_0}^{\mathsf{P}}$-complete language. Suppose $\mathsf{Gap}_\rho\text{-}\mathsf{MINpKT}^{\mathcal{O}} \in \mathsf{prBPTIME}[\tau(n)]$ for some polynomials ρ and τ. The proof is by induction on k.

The base case for $k = 0$ is trivial.

For the induction step, suppose the claimed conclusion holds for $k - 1$, where $1 \leq k < k_0$. Let $L \in \Sigma_k^{\mathsf{P}}$ and let V be a corresponding verifier. That is, for $x \in \{0,1\}^n$, $x \in L$ if and only if

$$\exists y_1 \in \{0,1\}^{v(n)}, \forall y_2 \in \{0,1\}^{v(n)}, \ldots, Q^k y_k \in \{0,1\}^{v(n)}, V(x, y_1, y_2, \ldots, y_k) = 1,$$

where v is a polynomial $Q^k \in \{\exists, \forall\}$. Let q be a polynomial.

To decide L, we aim to find a witness for a given instance, if one exists. Consider a yes-instance $x \in \{0,1\}^n$ of L. Let y_1^* be the lexicographically first witness for x, i.e., y_1^* is the lexicographically smallest string in $\{0,1\}^{v(n)}$ that satisfies

$$\forall y_2 \in \{0,1\}^{v(n)}, \ldots, Q^k y_k \in \{0,1\}^{v(n)}, V(x, y_1, y_2, \ldots, y_k) = 1,$$

where $Q^k \in \{\exists, \forall\}$. Let t be a sufficiently large polynomial specified later. We first observe the following.

Claim 6.17. *There exists a polynomial p such that*

$$\mathsf{pK}^{p(t(n))}(y_1^* \mid x) \leq \mathsf{pK}^{t(n),\mathcal{O}}(x) - \mathsf{pK}^{p(t(n))}(x) + \log p(t(n)).$$

Proof of Claim 6.17. Note that by Lemma 6.15 and the assumption that $\mathsf{Gap\text{-}MINpKT}^{\mathcal{O}}$ is easy, we get symmetry of information for pK^t. Now let p_1 be the polynomial from Lemma 6.16 and let $t := t(n)$. We have

$$\begin{aligned}
\mathsf{pK}^{p_{\mathsf{Sol}}(p_1(2t))}(y_1^* \mid x) &\leq \mathsf{pK}^{p_1(2t)}(x, y_1^*) - \mathsf{pK}^{p_{\mathsf{Sol}}(p_1(2t))}(x) + \log p_{\mathsf{Sol}}(p_1(2t)) \\
&\leq \mathsf{pK}^{2t,\mathcal{O}}(x, y_1^*) - \mathsf{pK}^{p_{\mathsf{Sol}}(p_1(2t))}(x) + \log p_{\mathsf{Sol}}(p_1(2t)) \qquad \text{(by Lemma 6.16)} \\
&\leq \mathsf{pK}^{\mathsf{poly}(n),\mathcal{O}}(y_1^* \mid x) + \mathsf{pK}^{t,\mathcal{O}}(x) - \mathsf{pK}^{p_{\mathsf{Sol}}(p_1(2t))}(x) + \log p_{\mathsf{Sol}}(p_1(2t)) \\
&\leq \mathsf{pK}^{\mathsf{poly}(n),\mathcal{O}}(y_1^* \mid x) + \mathsf{pK}^{t}(x) - \mathsf{pK}^{p_{\mathsf{Sol}}(p_1(2t))}(x) + \log p_{\mathsf{Sol}}(p_1(2t)).
\end{aligned}$$

Let p_2 be a sufficiently large polynomial, the above yields

$$\mathsf{pK}^{p_2(t)}(y_1^* \mid x) \leq \mathsf{pK}^{\mathsf{poly}(n),\mathcal{O}}(y_1^* \mid x) + \mathsf{pK}^{t}(x) - \mathsf{pK}^{p_2(t)}(x) + \log p_2(t),$$

Now note that $\mathsf{pK}^{\mathsf{poly}(n),\mathcal{O}}(y_1^* \mid x) \leq O(1)$ since given x and oracle access to $\mathcal{O}$, one can recover the lexicographically first witness for x in time $\mathsf{poly}(n)$. Therefore, we get

$$\mathsf{pK}^{p_2(t)}(y_1^* \mid x) \leq O(1) + \mathsf{pK}^{t,\mathcal{O}}(x) - \mathsf{pK}^{p_2(t)}(x) + \log p_2(t).$$

The claims follows by choosing p to be a sufficiently large polynomial. ◇

We will generate a set of candidate strings that is guaranteed to include a witness for x (specifically, y_1^*) if $x \in L$. Given this, it suffices to solve the task of checking whether a given string is a witness for x. This is achieved using the language $L' \in \Sigma_{k-1}^{\mathrm{P}}$ defined as follows.

$$(x, y_1) \in L' \text{ if and only if } \exists y_2, \ldots, Q^k y_k, V(x, y_1, y_2, \ldots, y_k) = 0, \text{ where } Q^k \in \{\exists, \forall\}.$$

Let $q' > q$ be a sufficiently large polynomial specified later. We have the following.

Claim 6.18. *There exists a randomized algorithm A' such that every $\ell' \in \mathbb{N}$, $x \in \{0,1\}^n$ and $y_1 \in \{0,1\}^{v(n)}$, with probability at least $1 - 2^{-\ell'}$ (over the internal randomness of A') both of the following hold.*

1. $A'\Big((x, y_1), 1^{\ell'}\Big)$ *runs in time*

$$2^{O\left(\mathsf{pK}^{q'(n+v(n)),\mathcal{O}}(x,y) - \mathsf{K}(x,y)\right)} \cdot \mathsf{poly}(\ell', n).$$

2. $A'\Big((x, y_1), 1^{\ell'}\Big) = 1$ *if and only if* $(x, y_1) \in L'$.

Proof of Claim 6.18. This follows directly from our induction hypothesis. ◇

In order to generate a set of "good" candidate witnesses, we will also need the following subroutines.

Claim 6.19. *There exists a randomized polynomial-time algorithm* $\mathsf{Approx\text{-}depth}$ *such that, for every* $x \in \{0,1\}^n$ *and* $\ell' \in \mathbb{N}$*,* $\mathsf{Approx\text{-}depth}(x, \ell')$ *outputs, with probability at least* $1 - 2^{-\ell'}$*, an integer* s *satisfying*

$$\mathsf{pK}^{t(n),\mathcal{O}}(x) - \mathsf{pK}^{p(t(n)),\mathcal{O}}(x) \leq s \leq \mathsf{pK}^{\rho^{-1}(t(n)),\mathcal{O}}(x) - \mathsf{pK}^{\rho(p(t(n))),\mathcal{O}}(x) + \log t(n) + \log \rho(p(t(n))),$$

where p *is the polynomial defined in Claim 6.17.*

Proof of Claim 6.19. Let Approx be a randomized polynomial-time algorithm that satisfies the following: Given $z \in \{0,1\}^m$, $1^{t'}$ and 1^κ, $\mathsf{Approx}(z, 1^{t'}, 1^\kappa)$ outputs, with probability at least $1 - 2^{-\kappa}$, an integer s_0 such that

$$\mathsf{pK}^{\rho(t')\mathcal{O}}(z) - \log \rho(t') \leq s_0 \leq \mathsf{pK}^{t'.\mathcal{O}}(z).$$

It is easy to see that such an algorithm Approx can be obtained from the assumption that $\mathsf{Gap}_\rho\text{-}\mathsf{MINpKT}^{\mathcal{O}} \in \mathsf{prBPP}$ and using success amplification techniques. (See, e.g., [GKLO22, Lemma 28].)

Let $t := t(n)$. By running $\mathsf{Approx}(x, 1^{\rho^{-1}(t)}, 1^{\ell'+1})$, with probability at least $1 - 2^{-\ell'}/2$, we get an integer s_1 such that

$$\mathsf{pK}^{t,\mathcal{O}}(x) - \log t \;\leq\; s_1 \;\leq\; \mathsf{pK}^{\rho^{-1}(t),\mathcal{O}}(x).$$

Similarly, by running $\mathsf{Approx}(x, 1^{p(t)}, 1^{\ell+1})$, with probability at least $1 - 2^{-\ell'}/2$, we get some integer s_2 such that

$$\mathsf{pK}^{\rho(p(t)),\mathcal{O}}(x) - \log \rho(p(t)) \;\leq\; s_2 \;\leq\; \mathsf{pK}^{p(t),\mathcal{O}}(x).$$

Then by a union bound, with probability at least $1 - 2^{-\ell'}$, we have

$$s_1 - s_2 \leq \mathsf{pK}^{\rho^{-1}(t),\mathcal{O}}(x) - \left(\mathsf{pK}^{\rho(p(t)),\mathcal{O}}(x) - \log \rho(p(t))\right) \tag{35}$$

and

$$s_1 - s_2 \geq \left(\mathsf{pK}^{t,\mathcal{O}}(x) - \log(t)\right) - \mathsf{pK}^{p(t),\mathcal{O}}(x). \tag{36}$$

We can then output

$$s := s_1 - s_2 + \log(t).$$

It is easy to verify that s satisfies the stated condition in the claim. ◇

Claim 6.20. *There exists a randomized polynomial-time algorithm* Valid *such that for every* $(x, y_1, 1^{\ell'}) \in \{0,1\}^n \times \{0,1\}^{v(n)} \times \mathbb{N}$*, with probability at least* $1 - 2^{-\ell'}$ *(over the internal randomness of* Valid*) both of the following hold:*

1. *If* $y_1 = y_1^*$*, then* $\mathsf{Valid}(x, y_1, 1^{\ell'}) = 1$.
2. *If* $\mathsf{Valid}(x, y_1, 1^{\ell'}) = 1$*, then it holds that*

$$\mathsf{pK}^{q'(n+v(n)),\mathcal{O}}(x, y_1) \leq \mathsf{pK}^{q(n),\mathcal{O}}(x) + \log q'(n + v(n)) + \log \rho(q(n)).$$

Proof of Claim 6.20. Again, let Approx be a randomized polynomial-time algorithm that satisfies the following: Given $z \in \{0,1\}^m$, $1^{t'}$ and 1^κ, $\mathsf{Approx}(z, 1^{t'}, 1^\kappa)$ outputs an integer s such that

$$\mathsf{pK}^{\rho(t'),\mathcal{O}}(z) - \log \rho(t') \leq s \leq \mathsf{pK}^{t'.\mathcal{O}}(z),$$

with probability at least $1 - 2^{-\kappa}$, where ρ is the polynomial such that $\mathsf{Gap}_\rho\text{-}\mathsf{MINpKT}^{\mathcal{O}} \in \mathsf{prBPP}$.

Let Valid be the following algorithm.

Given $(x, y_1, 1^{\ell'})$, let

$$\theta := \mathsf{Approx}\Big(x, 1^{q(n)}, 1^{\ell'+1)}\Big),$$

and

$$\mu := \mathsf{Approx}\Big((x, y_1), 1^{\rho^{-1}(q'(n+v(n)))}, 1^{\ell'+1)}\Big).$$

We accept if and only if $\mu \leq \theta + \log \rho(q(n))$.

We argue that the above algorithm satisfies the two conditions stated in the claim. Indeed, by the correctness of Approx and a union bound, we have that, with probability at least $1 - 2^{-\ell'}$, both the following two hold.

$$\mathsf{pK}^{\rho(q(n)),\mathcal{O}}(x) - \log \rho(q(n)) \leq \theta \leq \mathsf{pK}^{q(n),\mathcal{O}}(x),$$

and

$$\mathsf{pK}^{q'(n+v(n)),\mathcal{O}}(x, y_1) - \log q'(n + v(n)) \leq \mu \leq \mathsf{pK}^{\rho^{-1}(q'(n+v(n))).\mathcal{O}}(x, y_1).$$

Now to see that $\mathsf{Valid}(x, y_1^*, 1^{\ell}) = 1$, observe that in this case

$$\begin{aligned}\mu &\leq \mathsf{pK}^{\rho^{-1}(q'(n+v(n))),\mathcal{O}}(x, y_1^*)\\ &\leq \mathsf{pK}^{\rho(q(n)),\mathcal{O}}(x)\\ &\leq \theta + \log \rho(q(n)),\end{aligned}$$

where the second inequality uses the facts that q' is a sufficiently large polynomial and that given oracle access to $\mathcal{O}$, one can recover y_1^* in polynomial time given x.

For the second condition, if $\mathsf{Valid}(x, y_1, 1^{\ell}) = 1$, then it implies that

$$\begin{aligned}\mathsf{pK}^{q'(n+v(n)),\mathcal{O}}(x, y_1) - \log q'(n + v(n)) &\leq \mu\\ &\leq \theta + \log \rho(q(n))\\ &\leq \mathsf{pK}^{q(n),\mathcal{O}}(x) + \log \rho(q(n)).\end{aligned}$$

This completes the proof. ◇

We are now ready to describe our algorithm for deciding L.

Algorithm 4 An algorithm for deciding L

```
procedure $A(x, 1^{\ell})$
    $n := |x|$.
    $s := \mathsf{Approx\text{-}depth}(x, 1^{\ell+2})$.
    $s' := s + \log p(t(n))$.

    for $i \in [\ell]$ do
        $r :=$ a uniformly random string in $\{0,1\}^{p(t(n))}$.
        for $\Pi \in \{0,1\}^{\leq s'}$ do
            $y_1 :=$ the output of $\Pi(x; r)$ after running $p(t(n))$ steps.
            if $|y_1| = v(n)$ and $\mathsf{Valid}\Big(x, y_1, 1^{\ell+\log(\ell)+s'+4}\Big) = 1$ then
                if $A'\Big(x, y_1, 1^{\ell+\log(\ell)+s'+4}\Big) = 0$ then
                    Output True.
    Output False.
```

Correctness. We argue that the algorithm A correctly decides L. Fix $x \in \{0,1\}^n$ and $\ell \in \mathbb{N}$.

First, we assume that the (randomized) algorithm $\mathsf{Approx\text{-}depth}$ outputs a correct answer for x—that is, it satisfies the condition stated in Claim 6.19. Note that this occurs with probability at least $1-2^{-\ell}/4$, due to the choice of the second parameter in the invocation of $\mathsf{Approx\text{-}depth}$. We also assume that both algorithms, Valid and A', succeed (meaning that the conditions in Claim 6.20 and Claim 6.18, respectively, are satisfied) in all of their at most $\ell \cdot 2^{s'+1}$ executions. By a union bound and the parameter choices in their invocations, this occurs with probability at least $1 - 2^{-\ell}/4$.

Now consider a single execution of the for loop at Line 6 (i.e., Lines 7–12). Suppose $x \notin L$. Note that, in this case, by the definition of the language L' and the fact that A' correctly decides L' (Claim 6.18), the condition at Line 11 will not be satisfied for any y_1. Therefore, Line 12 will never be reached in this case.

On the other hand, if $x \in L$, then by Claim 6.17, with probability at least $2/3$ over $r \sim \{0,1\}^{p(t(n))}$, the set of candidate witnesses

$$\mathcal{S} := \Big\{ y_1 \mid y_1 \text{ is the output of } \Pi(x;r) \text{ after } p(t) \text{ steps for some } \Pi \in \{0,1\}^{\leq s} \Big\}, \tag{37}$$

generated by Line 9, contains y_1^*, which is the lexicographically first witness for x. To see this, observe that if the algorithm $\mathsf{Approx\text{-}depth}$ outputs a correct answer s for x, then we have

$$\begin{aligned} \mathsf{pK}^{p(t(n))}(y_1^* \mid x) &\leq \mathsf{pK}^{t,\mathcal{O}}(x) - \mathsf{pK}^{p(t(n))}(x) + \log p(t(n)) && \text{(by Claim 6.17)} \\ &\leq s + \log p(t(n)) && \text{(by Claim 6.19)} \\ &\leq s'. \end{aligned}$$

Consider the case when $y_1 = y_1^*$. It follows from Claim 6.20 (Item 1) that the condition at Line 10 is satisfied. Also, since y_1^* is a witness for x, by the definition of L', we have $(x, y_1^*) \notin L'$, and hence, by the fact that A' decides L' (Claim 6.18), the condition at Line 11 will be satisfied. Therefore, True will be output in this case.

Since the above process is executed ℓ times in the for loop, the probability that it never outputs True when $x \in L$ is at most $1/3^\ell$.

To conclude, with probability at least

$$1 - \frac{1}{4 \cdot 2^\ell} - \frac{1}{4 \cdot 2^\ell} - \frac{1}{4 \cdot 3^\ell} > 1 - \frac{1}{2^\ell},$$

the algorithm will correctly decide whether $x \in L$.

Running Time. We now analyze the running time of the algorithm. Again, we assume that the (randomized) algorithms $\mathsf{Approx\text{-}depth}$, Valid, and A' succeed in all their executions, which occurs with probability at least $1 - 2^{-\ell}$.

Consider a single execution of the for loop at Line 6. Define the following subset of $\mathcal{S}$ (which is defined in Equation (37)):

$$\mathcal{S}' := \{ y_1 \in \mathcal{S} \mid y_1 \text{ satisfies the condition at Line } 12 \}.$$

Now, it is easy to see that a single execution (Lines 7–12) takes time at most

$$2^{s+1} \cdot \mathsf{poly}(t(n)) + \sum_{y_1 \in \mathcal{S}'} \mathsf{Runtime}\Big(A'(x, y_1, 1^\ell) \Big). \tag{38}$$

Note that for every $y_1 \in \mathcal{S}'$, it follows from Claim 6.20 (Item 2) that

$$\mathsf{pK}^{q'(n+v(n)),\mathcal{O}}(x, y_1) \leq \mathsf{pK}^{q(n),\mathcal{O}}(x) + \log q'(n + v(n)) + \log \rho(q(n)). \tag{39}$$

Therefore, for every such y_1, we have

$$\begin{aligned}
&\mathsf{Runtime}\Big(A'(x, y_1, 1^\ell)\Big)\\
&\le \exp\Big(\mathsf{pK}^{q'(n+v(n)),\mathcal{O}}(x, y_1) - \mathsf{K}(x, y_1)\Big) \cdot \mathsf{poly}(n, \ell) && \text{(by Claim 6.18)}\\
&\le \exp\Big(\mathsf{pK}^{q(n),\mathcal{O}}(x) + \log q'(n + v(n)) + \log \rho(q(n)) - \mathsf{K}(x, y_1)\Big) \cdot \mathsf{poly}(n, \ell) && \text{(by Equation (39))}\\
&\le \exp\Big(\mathsf{pK}^{q(n),\mathcal{O}}(x) - \mathsf{K}(x)\Big) \cdot \mathsf{poly}(n, \ell)
\end{aligned}$$

It follows that the quantity in Equation (38) is at most

$$\begin{aligned}
&2^{s'+1} \cdot \mathsf{poly}(t(n)) + 2^{s'+1} \cdot \exp\Big(\mathsf{pK}^{q(n),\mathcal{O}}(x) - \mathsf{K}(x)\Big) \cdot \mathsf{poly}(n, \ell) && \text{(by } |\mathcal{S}'| \le 2^{s'+1}\text{)}\\
&\le \exp(s') \cdot \exp\Big(\mathsf{pK}^{q(n),\mathcal{O}}(x) - \mathsf{K}(x)\Big) \cdot \mathsf{poly}(n, \ell)\\
&\le \exp\Big(\mathsf{pK}^{\rho^{-1}(t(n)),\mathcal{O}}(x) - \mathsf{pK}^{\rho(p(t(n))),\mathcal{O}}(x) + \log t(n) + \log \rho(p(t(n))) + \log p(t(n))\Big)\\
& && \text{(by Claim 6.19)}\\
&\quad \cdot \exp\Big(\mathsf{pK}^{q(n),\mathcal{O}}(x) - \mathsf{K}(x)\Big) \cdot \mathsf{poly}(n, \ell) \cdot \mathsf{poly}(n, \ell)\\
&\le \exp\Big(\mathsf{pK}^{\rho^{-1}(t(n)),\mathcal{O}}(x) - \mathsf{K}(x)\Big) \cdot \exp\Big(\mathsf{pK}^{q(n),\mathcal{O}}(x) - \mathsf{K}(x)\Big) \cdot \mathsf{poly}(n, \ell) && \text{(by Lemma 2.24)}\\
&\le \exp\Big(\mathsf{pK}^{q(n),\mathcal{O}}(x) - \mathsf{K}(x)\Big) \cdot \exp\Big(\mathsf{pK}^{q(n),\mathcal{O}}(x) - \mathsf{K}(x)\Big) \cdot \mathsf{poly}(n, \ell)\\
&\le \exp\Big(\mathsf{pK}^{q(n),\mathcal{O}}(x) - \mathsf{K}(x)\Big) \cdot \mathsf{poly}(n, \ell),
\end{aligned}$$

where the second last inequality holds by letting $t(\cdot) \ge \rho(q(\cdot))$. This completes the proof. □

6.3.2 Proof of Lemma 6.13

We are now ready to show Lemma 6.13.

Proof of Lemma 6.13. Assume $\mathsf{Gap\text{-}MINpKT}^{\mathsf{PH}} \in \mathsf{prBPP}$. Let $k \in \mathbb{N}$ and $L \in \Sigma_k^{\mathsf{P}}$.

Given Theorem 2.25, it suffices to show how to decide L on average over the uniform distribution. The idea is to obtain a heuristic scheme by using the worst-case algorithm from Lemma 6.14 for deciding L, along with the fact that $\mathsf{pK}^{t,\mathsf{PH}}(x) - \mathsf{K}(x)$ is small for a uniformly random $x \in \{0,1\}^n$. Details follow.

Let $\mathcal{O}$ be a Σ_k^{P}-complete language, and let A be the algorithm from Lemma 6.14 that satisfies the following. Given $\ell \in \mathbb{N}$ and $x \in \{0,1\}^n$, with probability at least $1 - 2^{-\ell}$, $A(x, 1^\ell)$ runs in time

$$2^{O\left(\mathsf{pK}^{q(n),\mathcal{O}}(x) - \mathsf{K}(x)\right)} \cdot \mathsf{poly}(n, \ell)$$

and decides whether $x \in L$, where q is a polynomial satisfying that for all $x \in \{0,1\}^*$, $\mathsf{pK}^{q(|x|),\mathcal{O}}(x) \le |x| + O(\log |x|)$.

We now describe a heuristic scheme for solving L over the uniform distribution. Let B be the following algorithm:

> On input $(x, 1^n, 1^\kappa)$, where $|x| = n$, set cutoff $:= p(n, \kappa)$, where p is a polynomial specified later. Run $A(x, 1^{n+2})$ for cutoff steps. If it halts, output the returned answer; otherwise output $\perp$.

We argue the correctness of B. By the fact that for every $x \in \{0,1\}^n$, $A(x, 1^3)$ only errs with probability at most $2^{-3} = 1/8$ (over the internal randomness of A), we get that

$$\Pr_B\left[B(x, 1^n, 1^k) \in \{L(x), \bot\}\right] \geq \frac{4}{5}.$$

It remains to show that

$$\Pr_{x \sim \{0,1\}^n}\left[\Pr_B[B(x, 1^n, 1^\kappa) = \bot] < \frac{1}{5}\right] \geq 1 - \frac{1}{\kappa}.$$

Note that by construction, it is sufficient to show that with probability at least $1 - 1/\kappa$ over $x \sim \{0,1\}^n$, we have

$$\Pr_A\left[\mathsf{Runtime}\big(A(x, 1^3)\big) \leq \mathsf{cutoff}\right] > \frac{4}{5}.$$

First of all, for every $x \in \{0,1\}^n$, we have that with probability at least $7/8$ (which is greater than $4/5$) over its internal randomness, $A(x, 1^3)$ runs in time

$$2^{O\left(\mathsf{pK}^{q(n),\mathcal{O}}(x) - \mathsf{K}(x)\right)} \cdot \mathsf{poly}(n). \tag{40}$$

Firstly, for every $x \in \{0,1\}^n$, we have

$$\mathsf{pK}^{q(n),\mathcal{O}}(x) \leq n + O(\log n).$$

Secondly, by Lemma 2.22, with probability at least $1 - 1/\kappa$ over $x \sim \{0,1\}^n$,

$$\mathsf{K}(x) \leq n - O(\log n) - \log \kappa.$$

It follows that with probability at least $1 - 1/\kappa$ over $x \sim \{0,1\}^n$, the runtime given by Equation (40) is at most

$$\exp(n + O(\log n)) - n + O(\log n) + \log \kappa)) \cdot \mathsf{poly}(n) \leq \mathsf{poly}(n, \kappa).$$

By letting p be a sufficiently large polynomial, the above is at most cutoff. This completes the proof. □

6.4 The Deterministic Case

Theorem 6.21. *The following statements are equivalent*:

1. (Exclusion of PH-Heuristica). $\mathsf{DistPH} \subseteq \mathsf{AvgP} \implies \mathsf{PH} = \mathsf{P}$.
2. (Meta-Complexity of K^{PH}). $\mathsf{Gap\text{-}MINKT}^{\mathsf{PH}} \in \mathsf{P} \implies \mathsf{Mild\text{-}Gap\text{-}MINKT}^{\mathsf{PH}} \in \mathsf{P}$.

As in the proof of Theorem 1.5, Theorem 6.21 follows from the following results.

Theorem 6.22 ([Hir20])**.** *The following holds:*

$$\mathsf{DistPH} \subseteq \mathsf{AvgP} \iff \mathsf{Gap\text{-}MINKT}^{\mathsf{PH}} \in \mathsf{P}.$$

Theorem 6.23. *The following holds*:

$$\mathsf{NP} = \mathsf{P} \iff \mathsf{Mild\text{-}Gap\text{-}MINKT}^{\mathsf{PH}} \in \mathsf{P}.$$

To prove Theorem 6.22, we first establish the following.

Lemma 6.24. *If* $\mathsf{Mild\text{-}Gap\text{-}MINKT}^{\mathsf{SAT}} \in \mathsf{BPP}$, *then* $\mathsf{NP} \subseteq \mathsf{BPP}$.

Proof Sketch. The proof is essentially the same as that of Lemma 6.4. It suffices to consider the notion ℓ-$\mathsf{K}^{t,\mathsf{SAT}}$ in place of ℓ-$\mathsf{pK}^{t,\mathsf{SAT}}$. See also the proof of [HLO25, Theorem 34] for details. □

Proof of Theorem 6.23. The forward direction is immediate since $\mathsf{Mild\text{-}Gap\text{-}MINKT}^{\mathsf{PH}}$ lies in PH. Note that $\mathsf{PH} = \mathsf{P}$ if $\mathsf{NP} = \mathsf{P}$.

For the backward direction, first note that Lemma 6.24 implies the following.

$$\mathsf{Mild\text{-}Gap\text{-}MINKT}^{\mathsf{PH}} \in \mathsf{P} \implies \mathsf{NP} \subseteq \mathsf{BPP}.$$

Moreover, as mentioned in Section 1.3, [Hir20] shows that if $\mathsf{Gap\text{-}MINKT}^{\mathsf{PH}} \in \mathsf{P}$, then $\mathsf{BPP} = \mathsf{P}$ (see [Hir20, Theorem 1.17]). It follows that if $\mathsf{Mild\text{-}Gap\text{-}MINKT}^{\mathsf{PH}} \in \mathsf{P}$, then we have both $\mathsf{NP} \subseteq \mathsf{BPP}$ and $\mathsf{BPP} = \mathsf{P}$, which together imply $\mathsf{NP} = \mathsf{P}$, as desired. □

6.5 A Relativization Barrier

In this section, we exhibit an oracle world where $\mathsf{Gap\text{-}MINpKT}^{\mathsf{PH}}$ is easy while $\mathsf{Mild\text{-}Gap\text{-}MINpKT}^{\mathsf{PH}}$ is hard. To make the statement that we aim to prove precise, we first explain the relativization of these computational problems. As usual, this is obtained by providing oracle access to all machines involved in the computation.

Relativization. For a string $x \in \{0,1\}^n$, a time bound t, a parameter $\lambda \in [0,1]$, and languages $L, \mathcal{O} \subseteq \{0,1\}^*$, the $(L,\mathcal{O})$-*oracle probabilistic t-time bounded Kolmogorov complexity* of x is defines as

$$\mathsf{pK}_{\lambda}^{t,L,\mathcal{O}}(x) \triangleq \min\left\{ s \in \mathbb{N} \;\middle|\; \mathop{\mathbf{Pr}}_{r \sim \{0,1\}^t}\left[\mathsf{K}^{t,L,\mathcal{O}}(x \mid r) \leq s\right] \geq \lambda \right\},$$

where $\mathsf{K}^{t,L,\mathcal{O}}(x \mid r)$ denotes the conditional t-time bounded Kolmogorov complexity of x with oracle access to both L and $\mathcal{O}$.

For a function $\rho\colon \mathbb{N} \to \mathbb{N}$ and oracles L and $\mathcal{O}$, let $\mathsf{Gap}_\rho\text{-}\mathsf{MINpKT}^{L,\mathcal{O}}$ be the following promise problem: Given $(x, 1^s, 1^t)$, where $x \in \{0,1\}^*$ and $s,t \in \mathbb{N}$, decide whether $\mathsf{pK}_{2/3-1/100}^{t,L,\mathcal{O}}(x) \leq s$ ("yes" case) or $\mathsf{pK}_{2/3}^{\rho(t),L,\mathcal{O}}(x) > s + \log \rho(t)$ ("no" case).

Let $\mathcal{O}$ be an oracle. We say that $\mathsf{Gap\text{-}MINpKT}^{\mathsf{PH}^{\mathcal{O}},\mathcal{O}} \in \mathsf{prBPP}^{\mathcal{O}}$ if for every language $L \in \mathsf{PH}^{\mathcal{O}}$, there is a polynomial ρ and an oracle probabilistic polynomial-time algorithm that solves $\mathsf{Gap}_\rho\text{-}\mathsf{MINpKT}^{L,\mathcal{O}}$ when given oracle access to $\mathcal{O}$.

Analogously, we consider the collection of problems $\mathsf{Mild\text{-}Gap\text{-}MINpKT}^{\mathsf{PH}^{\mathcal{O}},\mathcal{O}}$, for each $L \in \mathsf{PH}^{\mathcal{O}}$. Each problem $\mathsf{Mild\text{-}Gap}_\rho\text{-}\mathsf{MINpKT}^{L,\mathcal{O}}$ is similar to $\mathsf{Gap}_\rho\text{-}\mathsf{MINpKT}^{L,\mathcal{O}}$ except for that in the no case, we have $\mathsf{pK}_{2/3}^{t+\rho(|x|),L,\mathcal{O}}(x) > s + \log \rho(t)$ instead of $\mathsf{pK}_{2/3}^{\rho(t),L,\mathcal{O}}(x) > s + \log \rho(t)$.

The statement $\mathsf{Mild\text{-}Gap\text{-}MINpKT}^{\mathsf{PH}^{\mathcal{O}},\mathcal{O}} \notin \mathsf{prBPP}^{\mathcal{O}}$ means that there is a language $L \in \mathsf{PH}^{\mathcal{O}}$ such that for no constant $c \geq 1$ and polynomial $\rho(n) = c \cdot n^c$ we have $\mathsf{Mild\text{-}Gap}_\rho\text{-}\mathsf{MINpKT}^{L,\mathcal{O}} \in \mathsf{prBPP}^{\mathcal{O}}$.

Theorem 6.25 (Oracle world where $\mathsf{Gap\text{-}MINpKT}^{\mathsf{PH}}$ is easy while $\mathsf{Mild\text{-}Gap\text{-}MINpKT}^{\mathsf{PH}}$ is hard)**.** *There is an oracle $\mathcal{O}$ such that* $\mathsf{Gap\text{-}MINpKT}^{\mathsf{PH}^{\mathcal{O}},\mathcal{O}} \in \mathsf{prBPP}^{\mathcal{O}}$ *but* $\mathsf{Mild\text{-}Gap\text{-}MINpKT}^{\mathsf{PH}^{\mathcal{O}},\mathcal{O}} \notin \mathsf{prBPP}^{\mathcal{O}}$.

Recall that, for an oracle $\mathcal{O}$, $\mathsf{DistPH}^{\mathcal{O}}$ denotes the set of all distributional problems $(L,\mathcal{D})$ where $L \in \mathsf{PH}^{\mathcal{O}}$ and $\mathcal{D} \in \mathsf{PSAMP}^{\mathcal{O}}$. The proof of Theorem 6.25 relies on the following oracle separation, which is an immediate consequence of [Imp11, HN21].

Theorem 6.26 ([Imp11, HN21])**.** *There is an oracle $\mathcal{O} \subseteq \{0,1\}^*$ such that* $\mathsf{DistPH}^{\mathcal{O}} \subseteq \mathsf{AvgBPP}^{\mathcal{O}}$ *and* $\mathsf{PH}^{\mathcal{O}} \not\subseteq \mathsf{BPP}^{\mathcal{O}}$.

We are now ready to prove Theorem 6.25.

Proof Sketch of Theorem 6.25. Since the argument is straightforward given Theorem 6.26 and the proof of Theorem 1.5, we only provide a sketch here. We consider the oracle $\mathcal{O} \subseteq \{0,1\}^*$ provided by Theorem 6.26, and verify that it satisfies the required properties in Theorem 6.25.

1) Proof that $\mathsf{Gap\text{-}MINpKT}^{\mathsf{PH}^{\mathcal{O}},\mathcal{O}} \in \mathsf{prBPP}^{\mathcal{O}}$**.** By assumption, $\mathsf{DistPH}^{\mathcal{O}} \subseteq \mathsf{AvgBPP}^{\mathcal{O}}$. Let $L \in \mathsf{PH}^{\mathcal{O}}$. We need to prove that there is a polynomial ρ and an oracle probabilistic polynomial-time algorithm that solves $\mathsf{Gap}_{\rho}\text{-}\mathsf{MINpKT}^{L,\mathcal{O}}$ when given oracle access to $\mathcal{O}$. We rely on the following lemma.[9]

Lemma 6.27 (Relativized form of Lemma 6.12)**.** *For oracles $L_1, L_2 \subseteq \{0,1\}^*$, if* $(\mathsf{MINKT}^{L_1,L_2},\mathcal{U}) \in \mathsf{AvgBPP}^{L_2}$*, then* $\mathsf{Gap}_{\rho}\text{-}\mathsf{MINpKT}^{L_1,L_2} \in \mathsf{prBPP}^{L_2}$ *for some polynomial ρ.*

Assuming Lemma 6.27, we proceed as follows. We apply the lemma with $L_1 = L$ and $L_2 = \mathcal{O}$, which simplifies the assumption to $(\mathsf{MINKT}^{L,\mathcal{O}}, \mathsf{PSAMP}^{\mathcal{O}}) \subseteq \mathsf{AvgBPP}^{\mathcal{O}}$. Since it is not hard to see that $\mathsf{MINKT}^{L,\mathcal{O}} \in \mathsf{PH}^{\mathcal{O}}$, the assumption of Lemma 6.27 holds (using that $\mathsf{DistPH}^{\mathcal{O}} \subseteq \mathsf{AvgBPP}^{\mathcal{O}}$). Therefore, it follows that $\mathsf{Gap}_{\rho}\text{-}\mathsf{MINpKT}^{L,\mathcal{O}} \in \mathsf{prBPP}^{\mathcal{O}}$ for some polynomial ρ, as desired.

It remains to prove Lemma 6.27. The proof is a simple adaptation of that of [HLO25, Proposition 58] to the relativized setting. Since it requires no new idea, we omit the details.

2) Proof that $\mathsf{Mild\text{-}Gap\text{-}MINpKT}^{\mathsf{PH}^{\mathcal{O}},\mathcal{O}} \notin \mathsf{prBPP}^{\mathcal{O}}$**.** Recall that, by construction, $\mathsf{PH}^{\mathcal{O}} \not\subseteq \mathsf{BPP}^{\mathcal{O}}$. Next, we use this separation to argue that $\mathsf{Mild\text{-}Gap\text{-}MINpKT}^{\mathsf{PH}^{\mathcal{O}},\mathcal{O}} \notin \mathsf{prBPP}^{\mathcal{O}}$. In more detail, we establish the contrapositive, i.e., if $\mathsf{Mild\text{-}Gap\text{-}MINpKT}^{\mathsf{PH}^{\mathcal{O}},\mathcal{O}} \in \mathsf{prBPP}^{\mathcal{O}}$ then $\mathsf{PH}^{\mathcal{O}} \subseteq \mathsf{BPP}^{\mathcal{O}}$. Since $\mathsf{NP}^{\mathcal{O}} \subseteq \mathsf{BPP}^{\mathcal{O}}$ yields $\mathsf{PH}^{\mathcal{O}} \subseteq \mathsf{BPP}^{\mathcal{O}}$, it suffices to argue the assumption implies that $\mathsf{NP}^{\mathcal{O}} \subseteq \mathsf{BPP}^{\mathcal{O}}$.

To achieve this, it is sufficient to establish the following lemma.

Lemma 6.28 (Relativized form of Lemma 6.4)**.** *For any $\mathcal{O} \subseteq \{0,1\}^*$, if* $\mathsf{Mild\text{-}Gap\text{-}MINpKT}^{\mathsf{PH}^{\mathcal{O}},\mathcal{O}} \in \mathsf{prBPP}^{\mathcal{O}}$*, then* $\mathsf{NP}^{\mathcal{O}} \subseteq \mathsf{BPP}^{\mathcal{O}}$.

It is straightforward to check that the proof of Lemma 6.4 extends to Lemma 6.28. The key point is that all techniques employed in the proof relativize. We omit the details. □

References

[AKRR03] Eric Allender, Michal Koucký, Detlef Ronneburger, and Sambuddha Roy. Derandomization and distinguishing complexity. In *Conference on Computational Complexity* CCC, pages 209–220, 2003.

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

[All10] Eric Allender. Avoiding simplicity is complex. In *Programs, Proofs, Processes – Conference on Computability in Europe* (CiE), pages 1–10, 2010.

[9] Analogously to the definitions introduced above, the measure $\mathsf{K}^{t,L_1,L_2}(x)$ and the corresponding meta-computational problem MINKT^{L_1,L_2} are defined in the natural way, i.e., by considering programs that have oracle access to both L_1 and L_2.

[BGS75] Theodore P. Baker, John Gill, and Robert Solovay. Relativizations of the P =? NP question. *SIAM J. Comput.*, 4(4):431–442, 1975.

[BLvM05] Harry Buhrman, Troy Lee, and Dieter van Melkebeek. Language compression and pseudorandom generators. *Comput. Complex.*, 14(3):228–255, 2005.

[BT06] Andrej Bogdanov and Luca Trevisan. Average-case complexity. *Found. Trends Theor. Comput. Sci.*, 2(1), 2006.

[CIKK16] Marco L. Carmosino, Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova. Learning algorithms from natural proofs. In *Conference on Computational Complexity* (CCC), pages 10:1–10:24, 2016.

[CLL24] Lijie Chen, Jiatu Li, and Jingxun Liang. Maximum circuit lower bounds for exponential-time Arthur Merlin. *Electron. Colloquium Comput. Complex.*, TR24-182, 2024.

[GK22] Halley Goldberg and Valentine Kabanets. A simpler proof of the worst-case to average-case reduction for polynomial hierarchy via symmetry of information. *Electron. Colloquium Comput. Complex.*, 7:1–14, 2022.

[GK23] Halley Goldberg and Valentine Kabanets. Improved learning from Kolmogorov complexity. In *Computational Complexity Conference* (CCC), pages 12:1–12:29, 2023.

[GK24] Halley Goldberg and Valentine Kabanets. Consequences of randomized reductions from SAT to time-bounded Kolmogorov complexity. In *Approximation, Randomization, and Combinatorial Optimization: Algorithms and Techniques* (APPROX/RANDOM), pages 51:1–51:19, 2024.

[GKLO22] Halley Goldberg, Valentine Kabanets, Zhenjian Lu, and Igor C. Oliveira. Probabilistic Kolmogorov complexity with applications to average-case complexity. In *Computational Complexity Conference* (CCC), pages 16:1–16:60, 2022.

[Gol97] Oded Goldreich. A sample of samplers - A computational perspective on sampling (survey). *Electron. Colloquium Comput. Complex.*, TR97-020, 1997.

[HIL$^+$23] Shuichi Hirahara, Rahul Ilango, Zhenjian Lu, Mikito Nanashima, and Igor C. Oliveira. A duality between one-way functions and average-case symmetry of information. In *Symposium on Theory of Computing* (STOC), pages 1039–1050, 2023.

[Hir18] Shuichi Hirahara. Non-black-box worst-case to average-case reductions within NP. In *Symposium on Foundations of Computer Science* (FOCS), pages 247–258, 2018.

[Hir20] Shuichi Hirahara. Characterizing average-case complexity of PH by worst-case meta-complexity. In *Symposium on Foundations of Computer Science* (FOCS), pages 50–60, 2020.

[Hir21] Shuichi Hirahara. Average-case hardness of NP from exponential worst-case hardness assumptions. In *Symposium on Theory of Computing* (STOC), pages 292–302, 2021.

[Hir22a] Shuichi Hirahara. Meta-computational average-case complexity: A new paradigm toward excluding Heuristica. *Bull. EATCS*, 136, 2022.

[Hir22b] Shuichi Hirahara. NP-hardness of learning programs and partial MCSP. In *Symposium on Foundations of Computer Science* (FOCS), pages 968–979, 2022.

[Hir22c] Shuichi Hirahara. Symmetry of information from meta-complexity. In *Computational Complexity Conference* (CCC), pages 26:1–26:41, 2022.

[Hir23] Shuichi Hirahara. Capturing one-way functions via NP-hardness of meta-complexity. In *Symposium on Theory of Computing* (STOC), pages 1027–1038, 2023.

[HKLO24] Shuichi Hirahara, Valentine Kabanets, Zhenjian Lu, and Igor C. Oliveira. Exact search-to-decision reductions for time-bounded Kolmogorov complexity. In *Computational Complexity Conference* (CCC), pages 29:1–29:56, 2024.

[HLN24] Shuichi Hirahara, Zhenjian Lu, and Mikito Nanashima. Optimal coding for randomized kolmogorov complexity and its applications. In *Symposium on Foundations of Computer Science* (FOCS), pages 369–378, 2024.

[HLO24] Shuichi Hirahara, Zhenjian Lu, and Igor C. Oliveira. One-way functions and pKt complexity. In *Theory of Cryptography* (TCC), pages 253–286, 2024.

[HLO25] Jinqiao Hu, Zhenjian Lu, and Igor C. Oliveira. Meta-complexity of nondeterministic Kolmogorov complexity. *Preprint*, 2025.

[HN21] Shuichi Hirahara and Mikito Nanashima. On worst-case learning in relativized Heuristica. In *Symposium on Foundations of Computer Science* (FOCS), pages 751–758, 2021.

[HN23] Shuichi Hirahara and Mikito Nanashima. Learning in Pessiland via inductive inference. In *Symposium on Foundations of Computer Science,* (FOCS), pages 447–457, 2023.

[HN25] Shuichi Hirahara and Mikito Nanashima. Complexity-theoretic inductive inference. *Electron. Colloquium Comput. Complex.*, TR25-92, 2025.

[IL90] Russell Impagliazzo and Leonid A. Levin. No better ways to generate hard NP instances than picking uniformly at random. In *Symposium on Theory of Computing* (STOC), pages 812–821, 1990.

[Ila23] Rahul Ilango. SAT reduces to the minimum circuit size problem with a random oracle. In *Symposium on Foundations of Computer Science* (FOCS), pages 733–742, 2023.

[Imp95] Russell Impagliazzo. A personal view of average-case complexity. In *Proceedings of the Tenth Annual Structure in Complexity Theory Conference*, pages 134–147, 1995.

[Imp11] Russell Impagliazzo. Relativized separations of worst-case and average-case complexities for NP. In *Conference on Computational Complexity* (CCC), pages 104–114, 2011.

[IW97] Russell Impagliazzo and Avi Wigderson. P = BPP if E requires exponential circuits: Derandomizing the XOR lemma. In *Symposium on Theory of Computing* (STOC), pages 220–229. ACM, 1997.

[KK25] Valentine Kabanets and Antonina Kolokolova. Chain rules for time-bounded Kolmogorov complexity. *Electron. Colloquium Comput. Complex.*, TR25-89, 2025.

[Lev84] Leonid A. Levin. Randomness conservation inequalities; information and independence in mathematical theories. *Information and Control*, 61(1):15–37, 1984.

[LO21] Zhenjian Lu and Igor C. Oliveira. An efficient coding theorem via probabilistic representations and its applications. In *International Colloquium on Automata, Languages, and Programming* (ICALP), pages 94:1–94:20, 2021.

[LO22] Zhenjian Lu and Igor C. Oliveira. Theory and applications of probabilistic Kolmogorov complexity. *Bull. EATCS*, 137, 2022.

[LORS24] Zhenjian Lu, Igor C. Oliveira, Hanlin Ren, and Rahul Santhanam. On the complexity of avoiding heavy elements. In *Symposium on Foundations of Computer Science* (FOCS), pages 2403–2412, 2024.

[LOZ22] Zhenjian Lu, Igor C. Oliveira, and Marius Zimand. Optimal coding theorems in time-bounded Kolmogorov complexity. In *International Colloquium on Automata, Languages, and Programming* (ICALP), pages 92:1–92:14, 2022.

[LP20] Yanyi Liu and Rafael Pass. On one-way functions and Kolmogorov complexity. In *Symposium on Foundations of Computer Science* (FOCS), pages 1243–1254, 2020.

[LP23] Yanyi Liu and Rafael Pass. One-way functions and the hardness of (probabilistic) time-bounded Kolmogorov complexity w.r.t. samplable distributions. In *Annual Cryptology Conference* (CRYPTO), pages 645–673, 2023.

[LP25] Yanyi Liu and Rafael Pass. Hardness along the boundary towards one-way functions from the worst-case hardness of time-bounded Kolmogorov complexity. In *Annual Cryptology Conference* (CRYPTO), 2025.

[LS24] Zhenjian Lu and Rahul Santhanam. Impagliazzo's worlds through the lens of conditional Kolmogorov complexity. In *International Colloquium on Automata, Languages, and Programming* (ICALP), pages 110:1–110:17, 2024.

[LV19] Ming Li and Paul M. B. Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications, 4th Edition*. Texts in Computer Science. Springer, 2019.

[LW95] Luc Longpré and Osamu Watanabe. On symmetry of information and polynomial time invertibility. *Inf. Comput.*, 121(1):14–22, 1995.

[MP25] Noam Mazor and Rafael Pass. Guest column: On cryptography and meta-complexity. *SIGACT News*, 56(2):63–101, June 2025.

[Ron04] Detlef Ronneburger. *Kolmogorov Complexity and Derandomization*. PhD thesis, Rutgers University, 2004.

[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.

[San23] Rahul Santhanam. An algorithmic approach to uniform lower bounds. In *Computational Complexity Conference* (CCC), pages 35:1–35:26, 2023.

[Sim23] Meta-Complexity. Research Program at the Simons Institute for the Theory of Computing, UC Berkeley, Spring 2023. https://simons.berkeley.edu/programs/Meta-Complexity2023.

[Sud97] Madhu Sudan. Decoding of reed solomon codes beyond the error-correction bound. *J. Complex.*, 13(1):180–193, 1997.

[SUV17] Alexander Shen, Vladimir A. Uspensky, and Nikolay Vereshchagin. *Kolmogorov complexity and algorithmic randomness*. American Mathematical Society, 2017.

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

A On Symmetry of Information for nKt

[Ron04] showed that Symmetry of Information (SoI) does not hold for Kt. Since the argument relativizes, it follows that SoI does not hold for $\mathsf{Kt}^{\mathsf{SAT}}$ either (where the queries are allowed to be adaptive). Recall that nKt is related, within a constant, to Kt complexity with *non-adaptive* queries to a SAT oracle (see Corollary 3.2 and Corollary 3.4). Nevertheless, the proof technique of [Ron04] does not seem to extend to show failure of SoI for nKt.

To clarify why this technique falls short and to highlight the role of adaptivity, we present a proof that SoI fails in a hybrid setting: the left-hand side uses Kt with access to a SAT oracle with up to two levels of adaptivity, while the right-hand side terms employ nKt. Whether symmetry of information fails for nKt remains an open problem.

Theorem A.1. *There is a constant $\varepsilon > 0$ such that for every large enough $n \in \mathbb{N}$, there exist strings $x, y, \in \{0,1\}^n$ such that*

$$\mathsf{Kt}^{\|_2\mathsf{SAT}}(x,y) < \mathsf{nKt}(x) + \mathsf{nKt}(y \mid x) - \varepsilon n.$$

Here, for an oracle $\mathcal{O}$ and a constant $d \in \mathbb{N}$, $\mathsf{Kt}^{\|_d\mathcal{O}}$ refers to the complexity measure Kt wherein the universal machine may make oracle calls to $\mathcal{O}$ with up to d levels of adaptivity.

We will need the following lemma.

Lemma A.2. *For every $n \in \mathbb{N}$, constant $d \in \mathbb{N}$, and language $A \in \mathsf{NP}$, there exist strings $x_1, ..., x_d \in \{0,1\}^n$ satisfying*

$$\sum_{i \in [d]} \mathsf{Kt}^{\|A}(x_i \mid x_1, ..., x_{i-1}) > d \cdot \frac{n}{2}$$

and

$$\mathsf{Kt}^{\|_d A}(x_1, ..., x_d) \leq \frac{n}{2} + O(\log n).$$

Proof. We employ an idea from [Ron04]. Fix $n, d \in \mathbb{N}$, and for $i \in [d]$, let x_i be the lexicographically first string of length n such that

$$\mathsf{Kt}^{\|A}(x_i \mid x_1, ..., x_{i-i}) > \frac{n}{2}.$$

Note that

$$\sum_{i \in [d]} \mathsf{Kt}^{\|A}(x_i \mid x_1, ..., x_{i-1}) > d \cdot \frac{n}{2}.$$

Now, consider the following brute-force algorithm to find x_i given $(x_1, ..., x_{i-1})$.

> Let S be an empty set at first. For every $j \in [n/2]$, for every $w \in \{0,1\}^j$, letting $t = 2^{n/2-j}$, let z_w be the output of a universal non-adaptive A-oracle TM run on input $(w, x_1, ..., x_{i-1})$ for t steps. (If the input does not encode a program making all of its oracle queries in parallel, do not output anything.) Add the output z_w to S.

Once the above is complete, sort S lexicographically, and let x_i be the lexicographically smallest string not in S.

We note that when simulating the procedure described above, we can collect the set of queries to the SAT oracle requested by each machine and submit all queries simultaneously to the oracle. For this reason, non-adaptive access to the SAT oracle is sufficient.

Observe that the algorithm described above runs in time at most

$$\frac{n}{2} \cdot 2^j \cdot 2^{n/2-j} + O(|S| \cdot \log |S|) < n^2 \cdot 2^{n/2}$$

and requires program size at most $O(\log n)$ provided non-adaptive access to an A oracle.

Moreover, repeating the above d times to produce all strings $x_1, ..., x_d$ takes time at most $d \cdot n^2 \cdot 2^{n/2}$, making queries to a A oracle with at most d levels of adaptivity. We obtain

$$\begin{aligned} \mathsf{Kt}^{\|_d A}(x_1, ..., x_d) &\leq O(\log n) + \log(d \cdot n^2 \cdot 2^{n/2}) \\ &\leq \frac{n}{2} + O(\log n). \end{aligned}$$

This completes the proof of the lemma.

□

We are now ready to prove Theorem A.1.

Proof of Theorem A.1. Applying Lemma A.2 with $d = 2$ and $A = \mathsf{SAT}$, for every $n \in \mathbb{N}$, we obtain strings $x, y \in \{0,1\}^n$ such that

$$\mathsf{Kt}^{\|_2 \mathsf{SAT}}(x, y) \leq \mathsf{Kt}^{\|\mathsf{SAT}}(x) + \mathsf{Kt}^{\|\mathsf{SAT}}(y \mid x) - \frac{n}{2} + O(\log n).$$

Then, by Lemma 3.2,

$$\mathsf{Kt}^{\|_2 \mathsf{SAT}}(x, y) < \mathsf{nKt}(x) + \mathsf{nKt}(y \mid x) - \frac{n}{4},$$

as desired. □