Electronic Colloquium on Computational Complexity, Report No. 290 (2026)

Asymmetry and Complexity of Nondeterministic Computations

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

October 13, 2026

Abstract

Symmetry of Information (SoI) [ZL70] is a fundamental theorem of classical Kolmogorov complexity whose status for time-bounded computation is closely related to central open problems in complexity theory and cryptography. In a recent development, [HMO26] showed that SoI fails for rKt complexity, namely in the setting of randomized computation. By contrast, many basic properties of Kolmogorov complexity are known to remain valid in the presence of nondeterminism [HLO26], suggesting that SoI may provide a fruitful structural principle for exploring the relative power of randomness and nondeterminism in computation.

Our results reveal that the validity of SoI for nondeterministic computation, namely for nKt complexity, is a subtle question connected to several frontiers. We show that symmetry of information holds on average for randomized extensions of nKt, while its worst case validity would imply breakthrough complexity separations. Although the standard cryptographic argument for asymmetry does not extend to the nondeterministic setting, we provide evidence that information remains asymmetric in nondeterministic computations. Among other results, we show that SoI would imply a collapse of exponential time classes to low levels of the polynomial hierarchy.

Then, building on the work of [Hir22b, GK22, KK26, HMO26], we develop a general framework connecting symmetry of information, meta-complexity, and explicit constructions in the nondeterministic setting. More precisely, we prove that suitable formulations of these three notions are quantitatively equivalent. We apply different aspects of this correspondence to make progress on complexity lower bounds and on the failure of SoI for nondeterministic computation. These consequences are unconditional and provide some of the strongest known lower bounds in meta-complexity.

*University of Warwick. E-mail: halley.goldberg@warwick.ac.uk
†University of Warwick. E-mail: jinqiao.hu@warwick.ac.uk
‡University of Victoria. E-mail: zhenjianlu@uvic.ca
§University of Michigan. E-mail: jylyu@umich.edu
¶University of Warwick. E-mail: igor.oliveira@warwick.ac.uk

ISSN 1433-8092

Contents

1 Introduction **4**
- 1.1 Overview . . . 4
  - 1.1.1 Context and Motivation . . . 4
  - 1.1.2 Summary of Contributions . . . 5
- 1.2 Results . . . 7
  - 1.2.1 The Intricate Setting of Nondeterministic Computation . . . 8
  - 1.2.2 Nondeterministic Computation is Likely Asymmetric . . . 9
  - 1.2.3 A Correspondence Between Asymmetry, Hardness, and Explicit Constructions . . . 10
  - 1.2.4 Unconditional Consequences: Complexity Lower Bounds and Asymmetry of Information . . . 11

2 Techniques **14**

3 Preliminaries **21**
- 3.1 Notation . . . 21
- 3.2 Computational Model . . . 21
- 3.3 Time-Bounded Kolmogorov Complexity . . . 22
- 3.4 Meta-Computational Problems . . . 24
- 3.5 Basic Tools in Meta-Complexity . . . 24
- 3.6 Pseudorandomness . . . 25
- 3.7 Technical Results . . . 28

4 Useful Results About Nondeterministic Kolmogorov Complexity **28**
- 4.1 nK . . . 29
- 4.2 rnK . . . 31
- 4.3 pnK . . . 35

5 Average-Case SoI for Nondeterministic Computation: Proof of Theorem 1 **36**
- 5.1 Average-Case SoI for pnKt . . . 36
  - 5.1.1 A Semi-Symmetry of Information Theorem for pnK^t . . . 36
  - 5.1.2 From Semi-SoI for pnK^t to Average-Case SoI for pnKt . . . 37
- 5.2 Average-Case SoI for rnKt . . . 38

6 Complexity Separations from SoI: Proof of Theorem 2 **39**
- 6.1 Consequence of Worst-Case SoI for pnKt . . . 39
- 6.2 Consequence of Worst-Case SoI for rnKt . . . 43

7 Relativization Barrier for SoI: Proof of Theorem 3 **43**

8 Complexity Collapses from SoI: Proof of Theorem 4 **47**
- 8.1 Uniform Collapses . . . 47
- 8.2 Nonuniform Inclusions . . . 50
  - 8.2.1 The Case of nKt . . . 51
  - 8.2.2 The Case of rnKt . . . 53

9 Refined Meta-Complexity Landscape Under SoI: Proof of Theorem 5 **57**
- 9.1 Nonuniform Upper Bound . . . 57
- 9.2 Uniform Lower Bound . . . 57

10 Asymmetry, Hardness, and Explicit Constructions: Proof of Theorem 6 **59**
10.1 SoI to Meta-Complexity . . . 59
10.1.1 The Case of rnKt . . . 59
10.1.2 The Case of nKt . . . 63
10.2 Meta-Complexity to SoI . . . 63
10.3 Natural Properties and Explicit Constructions . . . 65

11 A Quick Lower Bound Result via the Correspondence Theorem: Proof of Theorem 7 **67**

12 Unconditional Hardness of MnKtP: Proof of Theorem 8 **67**
12.1 Randomness Efficient Success Amplification for rK . . . 67
12.2 Reconstruction from Oracle Access to Dense Language . . . 70
12.3 Derandomizing rK^L . . . 72
12.4 Iterative Construction of High-Complexity Strings . . . 73
12.5 Limitations of the Iterative Construction for $\gamma \leq 1/2$ . . . 77

13 Unconditional Hardness of Gap-MrnKtP: Proof of Theorem 9 **77**
13.1 Exponential AMTIME ∩ coAMTIME Lower Bound via Iterative Construction . . . 78
13.2 AMTIME Lower Bound via Explicit Constructions . . . 82
13.3 Non-Uniform Lower Bound via Near-Maximum Circuit Lower Bounds . . . 85

14 Unconditional Failure of Conditional SoI: Proof of Theorem 10 **89**

A The Role of Adaptivity I: Ronneburger's Technique **96**

B The Role of Adaptivity II: Nonadaptive Reconstruction, Explicit Constructions, and Failure of SoI **97**

1 Introduction

1.1 Overview

1.1.1 Context and Motivation

An overarching goal in theoretical computer science is to understand the relative power of deterministic, randomized, and nondeterministic computation. In addition to comparing the problems that these resources can solve, one can seek structural principles that hold in one computational setting but fail in another. Such principles offer a different lens on the separation and collapse questions at the heart of complexity theory.

The structural principle at the center of this work is *Symmetry of Information* (SoI), a fundamental theorem of algorithmic information theory [ZL70]. This result states that for all n-bit strings x and y,

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

where $\mathsf{K}(z)$ denotes the Kolmogorov complexity of z. Equivalently, an optimal description of (x,y) can be decomposed into an optimal description of one string and an optimal description of the other conditioned on the first. In this sense, SoI is a powerful structural principle: describing two objects separately, while allowing one to serve as side information for the other, is essentially as efficient as describing them jointly. Another fundamental consequence of these equations is that the information that x contains about y, namely $\mathsf{K}(y) - \mathsf{K}(y \mid x)$, is roughly the same as the information that y contains about x, namely $\mathsf{K}(x) - \mathsf{K}(x \mid y)$.[1] In this sense, information is symmetric when considering *time-unbounded* computations.

The classical proof of SoI relies on exhaustive search and does not extend to time-bounded computations. Indeed, the status of SoI for time-bounded Kolmogorov complexity is intertwined with fundamental questions about the power of efficient computation. Its validity for polynomial-time measures is connected to the P versus NP problem [LW95, Hir22b], the existence of one-way functions [HIL+23, HLN24], and the feasibility of learning [HN23, HN25]. Collectively, these connections provide strong evidence that SoI fails for polynomial-time measures such as K^t, rK^t, and pK^t, although proving this unconditionally would resolve longstanding open problems.

The situation is better understood for Levin-style measures, which charge only logarithmically for decoding time and therefore permit exponentially long computations.[2] Ronneburger [Ron04] proved unconditionally that SoI fails for Levin's Kt complexity, and more recently [HMO26] established the same conclusion for its randomized analogue rKt. Thus, in both the deterministic and randomized settings, efficient descriptions violate a central structural principle of classical Kolmogorov complexity. We refer the reader to [HMO26] for applications of this phenomenon to complexity lower bounds and cryptography.

In this work, we investigate what happens when descriptions are allowed to use *nondeterminism*. Our main focus in on the nondeterministic measure nKt [AKRR03] and its probabilistic extensions rnKt and pnKt.[3] Informally, nKt is the nondeterministic analogue of Levin's Kt. A short nondeterministic program for x must uniquely specify x: every computation path outputs either x or a failure symbol, and at least one path outputs x. Nondeterminism may therefore yield more succinct descriptions, even though deterministically recovering the encoded string can require searching over the program's choices.

[1] This is analogous to the corresponding phenomenon in Shannon's information theory, where the mutual information between two random variables X and Y satisfies $I(X;Y) = H(X) - H(X \mid Y) = H(Y) - H(Y \mid X)$.

[2] Levin's Kt complexity of a string x is defined as $\mathsf{Kt}(x) = \min_{\Pi,t} |\Pi| + \lceil \log t \rceil$: Π outputs x within t steps,with respect to a fixed universal machine. Thus, Kt accounts for both description length and decoding time.

[3] Under plausible derandomization assumptions, it is not hard to show that $\mathsf{pnKt} \approx \mathsf{rnKt} \approx \mathsf{nKt}$, just as $\mathsf{pKt} \approx \mathsf{rKt} \approx \mathsf{Kt}$. We refer to the Appendix of [GKLO22] for similar results.

We say that SoI holds for nKt with error $e(n)$ if, for all $x, y \in \{0,1\}^n$,

$$\mathsf{nKt}(x, y) \geq \mathsf{nKt}(x) + \mathsf{nKt}(y \mid x) - e(n).$$

If this inequality holds with $e(n) = O(\log n)$, we simply say that SoI holds for nKt.[4] We use the analogous terminology for rnKt and pnKt.

There is a basic reason why this question is particularly subtle for nondeterministic computation. For deterministic and randomized time-bounded measures, the standard evidence against SoI comes from the asymmetry created by cryptographic hardness. If f is a one-way function, then $y = f(x)$ is easy to compute from x, whereas recovering x from $f(x)$ is hard on average. A joint description can exploit the easy forward computation, while a decomposition in the opposite order encounters the hard inverse problem. This intuition does not transfer directly to nondeterministic descriptions: even if $\mathsf{nKt}(x \mid f(x)) = O(\log n)$, such a short description gives no evident way to find a successful computation path and hence no efficient procedure for inverting f.[5]

At the same time, nondeterministic Kolmogorov complexity retains several features of the classical, time-unbounded theory that are conjectured to fail for deterministic and randomized time-bounded measures, including forms of language compression and conditional coding [HLO26]. This suggests that nondeterministic Kolmogorov complexity may be structurally closer to its time-unbounded counterpart. If SoI were also preserved, it would provide an intrinsic distinction between randomized and nondeterministic computation and could point toward a new route for separating these resources. If it fails, then its failure must be explained by a source of asymmetry beyond the standard cryptographic argument.

These considerations lead to the following basic question:

Does Symmetry of Information hold for nondeterministic computation?

Despite substantial progress on nondeterministic Kolmogorov complexity and its applications [AKRR03, BLvM05, All10, AKRR11, HLO26], this question has remained elusive. More broadly, studying SoI in the nondeterministic setting offers a possible route to new complexity lower bounds against nondeterministic computation, continuing a line of work that uses information symmetry as a lens on complexity lower bounds [Hir22b, HMO26].

1.1.2 Summary of Contributions

We develop a general theory for symmetry of information for nondeterministic computation, connecting this problem to questions in complexity lower bounds and explicit constructions. We then explore different aspects of this theory to obtain new unconditional results concerning meta-complexity and asymmetry of information. Our main contributions can be summarized as follows:[6]

1. First, we highlight a significant distinction between nondeterministic and randomized computations: while the existence of cryptographic one-way functions implies that SoI fails *on average* for rKt, we observe that SoI holds *on average* for rnKt. In particular, it follows that the standard cryptographic argument does not yield failure of SoI for nondeterministic computation.

[4] The reverse inequality holds for all the complexity measures considered here under natural assumptions on the computational model.

[5] We mention that it is also unclear whether the super-bit conjecture of [Rud97] implies the failure of SoI for nKt.

[6] A preliminary version of a subset of these results appeared as part of the unpublished ECCC preprint [GHL+25].

2. Formalizing the intuition that SoI might be useful for separating complexity classes, we prove that if SoI holds for rnKt in the *worst case* then $\mathsf{coNP} \not\subseteq \mathsf{AMTIME}[n^{\mathsf{poly}(\log n)}]$ (and in particular $\mathsf{NP} \not\subseteq \mathsf{BPP}$). We complement this result by establishing that any proof of worst-case SoI for nondeterministic computation must use a non-relativizing technique.

3. Bypassing the cryptographic approach, we provide evidence that nondeterministic computation is *asymmetric*. We show that the validity of SoI for either nKt or rnKt would imply the unlikely inclusion of nondeterministic exponential-time classes in $\mathsf{BPP}^{\mathsf{NP}}$ and $\mathsf{NP/poly} \cap \mathsf{coNP/poly}$. In particular, if PH does not collapse then SoI fails for both nKt and rnKt.

4. Building on recent work in the simpler setting of randomized computation, we develop a framework connecting *symmetry of information*, *meta-complexity*, and *explicit constructions* in the nondeterministic setting. We show that suitable formulations of these notions are quantitatively equivalent. Thus progress on explicit constructions is formally necessary to show the failure of SoI.

5. We apply different facets of this correspondence to derive new *unconditional* results. We consider the task of estimating nKt and rnKt, and derive lower bounds against quasi-polynomial size Boolean circuits and exponential-time Arthur-Merlin classes. Finally, we combine one of these complexity lower bounds with additional ideas to establish that SoI fails for nondeterministic computation in the presence of side information.

Our techniques build on the early literature on nondeterministic Kolmogorov complexity [AKRR03, BLvM05, All10, AKRR11] and on recent advances in the study of meta-complexity [Hir22b, GK22, HLO26], explicit constructions [CLL25], and time-bounded Kolmogorov complexity [HLO24, KK26, HMO26], among other results from the theory of computational pseudorandomness.

Broader Perspective. Over the last decade, several fundamental problems from algorithms and complexity theory have been investigated through the lens of Kolmogorov complexity and meta-complexity. Influential advances have occurred in areas such as learning theory [CIKK16], cryptography [LP20], and average-case complexity [Hir21]. In some cases, such as [Hir21], 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, LP25, KK26]).

In this work, we advance this program by further developing the theory of *nondeterministic Kolmogorov complexity*. We connect this theory to some questions at the frontier of algorithms and complexity theory and use these connections to establish new lower bounds in meta-complexity.

We discuss our results in detail in Section 1.2 below. In Section 2, we present an overview of some of the underlying proofs and techniques.

1.2 Results

First, we need to define the main time-bounded Kolmogorov complexity measures appearing in our results. As alluded to above, we consider a natural extension of Levin's Kt complexity [Lev84] to the setting of nondeterministic computation [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 3).

Notions from Time-Bounded Kolmogorov Complexity. 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.

We will also consider probabilistic extensions of nKt complexity in the sense of [Oli19] and [GKLO22]. For a string $x \in \{0,1\}^*$, the *randomized nondeterministic time-bounded Kolmogorov complexity of* x is defined as

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

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.

Recall that 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* (see also [HLO26]) is defined as

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

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

1.2.1 The Intricate Setting of Nondeterministic Computation

As explained in Section 1.1.1, it is unclear how to rule out SoI for nondeterminsitic measures assuming the existence of one-way functions. Note that, in the randomized setting, this cryptographic assumption provides an even stronger guarantee: [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,

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

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, we show (unconditionally) that SoI *holds on average* for the probabilistic extensions of nKt.

Theorem 1 (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 $n, k \in \mathbb{N}$,*

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

Similarly,

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

It is unclear how to extend the proof of Theorem 1 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 2 (Complexity Lower Bounds from Worst-Case SoI)**.** *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 $x, y \in \{0,1\}^n$,*

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

   where $e = \log^c n$.

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

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

Note that the consequence $\mathsf{TAUT} \notin \mathsf{AM}$ implies that $\mathsf{NP} \not\subseteq \mathsf{BPP}$. In particular, Theorem 2 formalizes the intuition discussed in Section 1.1.1 that the validity of SoI in a nondeterminsitic setting could point toward a new route for separating the computational power of randomness and nondeterminism.

While several results establish that lower bounds follow from the *failure* of SoI (see, e.g., [LW95, Hir22b, HIL+23, HLO24, HLN24]), an interesting aspect of Theorem 2 is that here a lower bound follows from the *validity* of SoI in a time-bounded setting. In a sense, Theorem 2 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 [Hir22b].

Next, we show a general result implying that any proof of Equation ($*$) must employ non-relativizing techniques.

Theorem 3 (Strong Failure of SoI in Some Oracle World)**.** *There exist an oracle $\mathcal{O}$ and a constant $\varepsilon > 0$ such that, for every complexity measure $\mu \in \{\mathsf{rKt}, \mathsf{pKt}, \mathsf{nKt}, \mathsf{rnKt}, \mathsf{pnKt}\}$, the following holds. For all sufficiently large $n \in \mathbb{N}$, there exist strings $x, y \in \{0,1\}^n$ such that*

$$\mu^{\mathcal{O}}(x, y) < \mu^{\mathcal{O}}(x) + \mu^{\mathcal{O}}(y \mid x) - \varepsilon \cdot n.$$

The relativized failure of SoI was known for deterministic and non-deterministic polynomial-time (distinguishing) complexity (see [BF95, LR05, Lee06]). In contrast, Theorem 3 considers Levin-style measures and their randomized extensions, providing an oracle world where SoI fails in a strong sense for all these measures.

We observe that, in order to show the weaker separation $\mathsf{NP} \not\subseteq \mathsf{BPP}$, it is enough to establish Equation ($*$) under the assumption that $\mathsf{NP} \subseteq \mathsf{BPP}$ (since as observed above $\mathsf{TAUT} \notin \mathsf{AM}$ implies in particular that $\mathsf{NP} \not\subseteq \mathsf{BPP}$). By a similar argument, to show $\mathsf{NEXP} \not\subseteq \mathsf{BPP}$, it suffices to prove that $\mathsf{NEXP} \subseteq \mathsf{BPP} \implies$ Equation ($*$). Consequently, as an approach to lower bounds, symmetry of information might be useful even if Equation ($*$) turns out to be false.[7]

1.2.2 Nondeterministic Computation is Likely Asymmetric

As discussed in Section 1.1.1 and explained in the preceding section, any failure of SoI for nKt and its extensions must arise from a source of asymmetry not captured by the standard cryptographic argument. We pursue a different approach and obtain strong evidence that SoI indeed fails in the nondeterministic setting.

Recall that FewEXP is the set of languages in NEXP where each positive instance admits at most $2^{\mathsf{poly}(n)}$ witnesses. Note that $\mathsf{EXP} \subseteq \mathsf{FewEXP} \subseteq \mathsf{NEXP}$. We use $||\mathsf{NP}$ to denote parallel access to an NP-oracle.

Theorem 4 (Strong Complexity Collapses from SoI)**.** *The following results hold:*

(*i*) [Uniform Collapses]
- *If SoI holds for* nKt *or* rnKt *with error $e(n) = O(\log n)$, then* $\mathsf{FewEXP} \subseteq \mathsf{BPP}^{\mathsf{NP}}$.
- *If SoI holds for* nKt *or* rnKt *with error $e(n) = \mathsf{poly}(\log n)$, then* $\mathsf{FewEXP} \subseteq \mathsf{BPTIME}^{\mathsf{NP}}[n^{\mathsf{poly}(\log n)}]$.

(*ii*) [Nonuniform Collapses]
- *If SoI holds for* rnKt *with error $e(n) = O(\log n)$, then* $\mathsf{BPEXP}^{||\mathsf{NP}} \subseteq \mathsf{NP/poly} \cap \mathsf{coNP/poly}$.
- *If SoI holds for* nKt *with error $e(n) = O(\log n)$, then* $\mathsf{EXP}^{||\mathsf{NP}} \subseteq \mathsf{NP/poly} \cap \mathsf{coNP/poly}$.

In particular, it follows from Theorem 4 Part (*ii*) and a standard result [Yap83] that if SoI holds for either rnKt or nKt, the polynomial-time hierarchy collapses to the third level. Thus, under a widely believed computational assumption, SoI does not hold for nondeterministic computation.[8]

[7] Interestingly, it is known that $\mathsf{NP} \subseteq \mathsf{BPP}$ implies that symmetry of information holds for other time-bounded Kolmogorov complexity measures [Hir22b].

[8] In a concurrent and independent work [KB26], Viacheslav Khrushchev also established that the validity of SoI for nKt implies the collapse of the polynomial hierarchy. We thank Bruno Bauwens for bringing this to our attention.

Before stating our next result, we need the following definitions. For a complexity measure κ (such as Kt, nKt, or rnKt), we consider the standard formulation of the corresponding meta-computational problem, where given a string $x \in \{0,1\}^n$ and a threshold parameter s, the goal is to decide if $\kappa(x) \leq s$. We will also consider the gap version of this problem, where for fixed functions $s_1(n)$ and $s_2(n)$, we are given the promise that either $\kappa(x) \leq s_1(n)$, or $\kappa(x) \geq s_2(n)$, and the goal is to distinguish the two cases. As usual, these computational problems will be denoted by $\mathsf{M}\kappa\mathsf{P}$ and $\mathsf{Gap\text{-}M}\kappa\mathsf{P}[s_1, s_2]$, respectively. When $s_1(n) = n^{\varepsilon}$ for a fixed but arbitrarily small $\varepsilon > 0$ and $s_2(n) = n - 1$, we simply write $\mathsf{Gap\text{-}M}\kappa\mathsf{P}$. For a promise problem $\Pi = (\mathsf{YES}, \mathsf{NO})$ and a complexity class $\mathcal{C}$, we say that $\Pi \in \mathcal{C}$ if there is a language $L \in \mathcal{C}$ such that $\mathsf{YES} \subseteq L$ and $\mathsf{NO} \subseteq \{0,1\}^* \setminus L$.

Recall that showing that $\mathsf{MKtP} \notin \mathsf{P}$ is a longstanding open problem in meta-complexity that dates back at least to [ABK+06]. In the next result, we show that the validity of SoI in the nondeterministic setting implies both *upper bounds* and *lower bounds* for meta-computational problems.

Theorem 5 (Refined Meta-Complexity Landscape Under SoI)**.** *The following results hold:*

(*i*) [Nonuniform Upper Bound] *If SoI with error $e(n) = O(\log n)$ holds for rnKt, then*

$$\mathsf{Gap\text{-}MrnKtP}[n - O(\log n), n - 1] \in \mathsf{NP/poly} \cap \mathsf{coNP/poly}.$$

(*ii*) [Uniform Lower Bound] *If SoI with error $e(n) = \mathsf{poly}(\log n)$ holds for either nKt or rnKt, then*

$$\mathsf{MKtP} \notin \mathsf{NTIME}[n^{\mathsf{poly}(\log n)}] \cap \mathsf{coNTIME}[n^{\mathsf{poly}(\log n)}].$$

It seems unlikely that an approximation to $\mathsf{Kt}(x)$ can be certified in $\mathsf{NP/poly} \cap \mathsf{coNP/poly}$, let alone for the more expressive measure rnKt, as in the conclusion of Theorem 5 Item (*i*). Therefore, we interpret Theorem 5 as further evidence that information is likely asymmetric in nondeterministic computation.

1.2.3 A Correspondence Between Asymmetry, Hardness, and Explicit Constructions

Next, we establish a connection among symmetry of information for rnKt, the meta complexity of estimating rnKt, and $\mathsf{pr\text{-}BPTIME}[\cdot]^{\|\mathsf{NP}}$-explicit constructions. This correspondence extends recent work [Hir22b, GK22, KK26, HMO26] relating SoI to meta-complexity in the simpler setting of randomized computation. We prove a more general result for nondeterministic computation, where the analysis is more subtle, as explained below and in Section 2.

Theorem 6 (Correspondence Between Asymmetry of Information, Hardness, and Explicit Constructions)**.** *The following statements are equivalent:*

(i) (SoI holds for rnKt) *Symmetry of information holds for rnKt with error $e(n) = \mathsf{poly}(\log n)$.*

(ii) (Upper bound for $\mathsf{Gap\text{-}MrnKtP}$) $\mathsf{Gap\text{-}MrnKtP}[n - \mathsf{poly}(\log n), n - 1] \in \mathsf{pr\text{-}BPTIME}[n^{\mathsf{poly}(\log n)}]^{\|\mathsf{NP}}$.

(iii) (Natural property against rnKt) *There is $P = (\mathcal{YES}, \mathcal{NO}) \in \mathsf{pr\text{-}BPTIME}[n^{\mathsf{poly}(\log n)}]^{\|\mathsf{NP}}$ such that*

- $|\mathcal{YES} \cap \{0,1\}^n| \geq 2^{n-1}$*, and*
- *For every $x \in \{0,1\}^n$ where $\mathsf{rnKt}(x) < n - \mathsf{poly}(\log n)$, $x \in \mathcal{NO}$.*

(iv) (Hardness of explicit constructions for dense $\mathsf{pr\text{-}BPTIME}[\cdot]^{\|\mathsf{NP}}$ properties) *The following is false:*

For any promise problem $P = (\mathcal{YES}, \mathcal{NO}) \in \mathsf{pr\text{-}BPTIME}[n^{\mathsf{poly}(\log n)}]^{\|\mathsf{NP}}$ *such that* $|\mathcal{YES} \cap \{0,1\}^n| \geq 2^{n-1}$*, there exists* $t(\cdot), \ell(\cdot) : \mathbb{N} \to \mathbb{N}$ *where* $\ell(n) + \log t(n) < n - \mathsf{poly}(\log(n))$ *for each* n*, and an algorithm* $A(1^n, \cdot)$ *such that, for infinitely many* n*, there exists advice* $\alpha_n \in \{0,1\}^{\ell(n)}$ *and a string* $x_n \in \{0,1\}^n \setminus \mathcal{NO}$*, such that*

$$\Pr_{r \sim \{0,1\}^{t(n)}} \begin{bmatrix} \forall w \in \{0,1\}^{t(n)}, A(1^n, \alpha_n, w, r) \text{ outputs } x \text{ or } \perp \text{ in } t(n) \text{ steps}; \\ \exists w \in \{0,1\}^{t(n)}, A(1^n, \alpha_n, w, r) \text{ outputs } x \text{ in } t(n) \text{ steps} \end{bmatrix} \geq \frac{2}{3}.$$

General explicit construction results are known for dense properties in pr-BPP [OS17] and in coAM [CLL25]. In contrast, Theorem 6 shows that SoI fails for rnKt if a suitable positive result can be established for dense properties in $\mathsf{pr\text{-}BPP}^{\|\mathsf{NP}}$. Moreover, the design of such an algorithm is formally necessary to establish the asymmetry of nondeterministic computation.

Interestingly, the *nonadaptive access* to the NP oracle and the precise formulation of the nondeterministic complexity of the explicit construction algorithm A are essential for the equivalence result to hold. In fact, the construction from [OS17] relativizes to an NP-oracle, but the access to the NP-oracle by the corresponding algorithm A is *adaptive*, which is insufficient to rule out SoI. This is another subtle aspect of the nondeterministic setting, and we discuss it in more detail in Section B.

More on Adaptive vs Nonadaptive Queries. For every string x, $\mathsf{nKt}(x) = \Theta(\mathsf{Kt}^{\|\mathsf{SAT}}(x))$ (see Corollary 42 and Corollary 44). In other words, in settings where a constant multiplicative overhead is inessential, we can think of nKt as Kt in the presence of *nonadaptive* queries to a SAT oracle. (Note that this is not necessarily the case for SoI, where a large overhead when relating different measures can impact SoI.) Interestingly, since Ronneburger's proof [Ron04] of the failure of SoI for Kt relativizes (see Section A for a related discussion), we know that SoI fails for both Kt and $\mathsf{Kt}^{\mathsf{SAT}}$. However, similarly to the situation for nKt and related measures, we currently do not know if SoI fails for $\mathsf{Kt}^{\|\mathsf{SAT}}$. Thus the validity of SoI in the nondeterministic setting is closely related to the role of adaptive queries in computation. For another example of the relevance of (non)adaptive queries for SoI, see Section B.

In the next section, we apply Theorem 6 and its underlying correspondences to derive new unconditional lower bounds in meta-complexity and to establish the failure of SoI in the presence of side information.

1.2.4 Unconditional Consequences: Complexity Lower Bounds and Asymmetry of Information

As highlighted in Section 1.2.2, showing that Kt cannot be computed in polynomial time, i.e., that $\mathsf{MKtP} \notin \mathsf{P}$, is a longstanding open problem in meta-complexity [ABK+06]. By considering whether SoI holds for rnKt with error $\mathsf{poly}(\log n)$, Theorem 5 (Item 2) and Theorem 6 imply the following *unconditional* lower bound result.

Theorem 7 (A Quick Lower Bound Consequence)**.** *The following result holds:*

$$\mathsf{MKtP} \notin \mathsf{NTIME}[n^{\mathsf{poly}(\log n)}] \cap \mathsf{coNTIME}[n^{\mathsf{poly}(\log n)}] \text{ or } \mathsf{Gap\text{-}MrnKtP}[s_1, s_2] \notin \mathsf{pr\text{-}BPTIME}[n^{\mathsf{poly}(\log n)}]^{\|\mathsf{NP}},$$

where $s_1(n) = n - \mathsf{poly}(\log n)$ *and* $s_2(n) = n - 1$.

This result provides a quick example of how SoI can offer a fruitful viewpoint for establishing complexity lower bounds. In our next results, we apply the correspondences behind Theorem 6 in a more sophisticated, non-black-box way to establish new results on the asymmetry and complexity of nondeterministic computations.

Lower Bounds for nKt via Symmetry of Information. 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 8 (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].[9] In contrast, Theorem 8 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 12.5 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 $\mathsf{MnKtP} \in \mathsf{P}^{\mathsf{NEXP}} \subseteq \mathsf{NEXP/poly}$ 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.

Lower Bounds for rnKt via Explicit Constructions. Fix two functions $s_1, s_2\colon \mathbb{N} \to \mathbb{N}$. We 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). Recall that 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 9 (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 $L \notin \mathsf{coAMTIME}\left[n^{\mathsf{polylog}(n)}\right]$.

[9] We remark that the proof of [AKRR11] can be extended to a larger time bound, but the approach fundamentally faces a *half-exponential barrier*. In particular, within the framework of [All10, 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 $T(T(n)) = o(2^n)$.

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* $L \notin \mathsf{SIZE}^{\mathcal{O}}\left[n^{\mathsf{polylog}(n)}\right]$.

While we were able to establish that $\mathsf{MrnKtP} \notin \mathsf{AM}$, it remains an interesting open problem to show that $\mathsf{MnKtP} \notin \mathsf{NP}$.

To the best of our knowledge, no previous lower bounds against AM, SIZE[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.

An Unconditional Failure of SoI via Computational Hardness. While we are currently unable to determine, unconditionally, if symmetry of information fails in a nondeterministic setting, we obtain an unconditional result in the presence of a conditional string, which one can think of as an advice string or side information. In more detail, we say that conditional SoI holds for nKt with error $e(n)$ if for any strings $x, y, w \in \{0,1\}^*$, we have that

$$\mathsf{nKt}(x, y \mid w) \geq \mathsf{nKt}(x \mid y, w) + \mathsf{nKt}(y \mid w) - e(|x| + |y|).$$

We note that we do not assume random-access to the conditional string in the setting of conditional symmetry of information. Then, unconditionally, it is possible to show that conditional SoI for nKt does not hold with error $e(n) = \mathsf{poly}(\log n)$.

Theorem 10 (Failure of Conditional SoI for nKt)**.** *For any* $e(n) \leq \mathsf{polylog}(n)$, *conditional SoI for* nKt *does not hold with error* $e(n)$.

Our setting is related to the failure of SoI (without a conditional string) in some oracle world (Theorem 3). There are however a couple of differences. While the oracle can be interpreted as a conditional string, in the oracle setting one assumes query access to the oracle, while here we do not need to assume query access to the conditional string. In addition, the oracle construction requires techniques from computational pseudorandomness, while the proof given here relies on a reduction to a computational lower bound, in the spirit of Theorem 6.

Use of AI Tools. All ideas, arguments, and proofs presented in this paper were developed by the authors. The proofs were checked for correctness using GPT 5.6 and GPT 6.0.

Acknowledgements. We thank Jiatu Li, Hanlin Ren, Ninad Rajgopal, Rahul Santhanam, Rohan Acharya, and Yahel Manor for related discussions. This work was supported in part by the UKRI Frontier Research Guarantee Grant EP/Y007999/1 and the Centre for Discrete Mathematics and its Applications (DIMAP) at the University of Warwick. The authors would also like to thank the Isaac Newton Institute for Mathematical Sciences, Cambridge, for support and hospitality during the programme Logical Foundations of Complexity Theory (LFC), where part of the work on this paper was undertaken. The LFC programme was supported by EPSRC grant EP/Z000580/1.

2 Techniques

We now describe the key ideas and techniques employed in the proof of our main results.

Theorem 1 states that, unconditionally, SoI holds for pnKt with high probability over random strings sampled efficiently. The theorem may proved by an argument closely resembling one from [HLO26], so we will not give a further sketch here. We now move on to describe the proof of Theorem 2.

Complexity separations from symmetry of information (Theorem 2). The argument builds on techniques from [Ron04, CLL25, HLO24]. We would like to argue in the contrapositive: namely, assuming $\mathsf{NP} \subseteq \mathsf{coAMTIME}[n^{\mathsf{polylog}(n)}]$ and demonstrating *asymmetry* of information for pnKt.

Our starting point is the approach of [Ron04] for excluding SoI for Kt. 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 variants of Kt complexity is that known exponential-time procedures for determining complexity are probabilistic, and 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 (denoted A^{SAT} here) for approximating pnKt complexity. More specifically, given a pair of strings $u \in \{0,1\}^n$ and $v \in \{0,1\}^{\leq 2^n}$, A^{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 $\mathsf{pnKt}(u \mid v) < n/2$.[10] At a high level, our approach entails derandomizing A^{SAT} appropriately, yielding an efficient enough 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 A^{SAT}. Here, we apply a recent *explicit construction* result due to Chen et al. [CLL25] (discussed further below), which we adapt 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. 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), A^{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 A^{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 $\mathsf{pnKt}(x, y) \leq \mathsf{rnKt}(x, y) \leq 2^{n^{\delta'}}$. In particular, 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.

[10] An algorithm with these parameters can be obtained from Lemma 60 combined with success amplification for pnKt following [GKLO22].

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 6.1 for further details.

The proof of Theorem 3 is basically a relativizing version of the argument used in Theorem 2, 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.[11]

Complexity collapses from symmetry of information (Theorem 4). Theorem 4 shows that surprising uniform and non-uniform complexity collapses would follow from SoI for nKt or rnKt. We start by discussing the proof of the item stating that $\mathsf{FewEXP} \subseteq \mathsf{BPP}^{\mathsf{NP}}$ if SoI holds for nKt with error $e(n) = O(\log n)$. The proof employs the PRG construction of [IW01, TV07] as well as a version of the "easy witness method" due to [Kab00].

Later in this section, we outline the proof of a correspondence theorem (Theorem 6), which shows, in part, that SoI for rnKt implies the easiness of Gap-MrnKtP for randomized algorithms with non-adaptive access to an NP oracle. We may rely on very similar techniques here to show that SoI for nKt with error $O(\log n)$ implies $\mathsf{Gap\text{-}MnKtP} \in \mathsf{pr\text{-}P}^{\|\mathsf{NP}}$. See below for more detail on this piece of the argument. The rest of the proof of our stated uniform collapse consists in contradicting this easiness of Gap-MnKtP, assuming $\mathsf{FewEXP} \not\subseteq \mathsf{BPP}^{\mathsf{NP}}$.

We follow a win-win approach to obtain a PRG computable single-valued non-deterministically with small advice and secure against NP-oracle circuits. On one hand, if $\mathsf{EXP} \not\subseteq \mathsf{BPP}^{\mathsf{NP}}$, then the desired PRG follows from [IW01, TV07]; this step employs an intermediate win-win argument (see Corollary 30). On the other hand, if $\mathsf{EXP} \subseteq \mathsf{BPP}^{\mathsf{NP}}$, then $\mathsf{FewEXP} \neq \mathsf{EXP}$ by the assumption that $\mathsf{FewEXP} \not\subseteq \mathsf{BPP}^{\mathsf{NP}}$. In this case, for any fixed $\mathsf{FewEXP} \setminus \mathsf{EXP}$ language L, there are infinitely many strings $x \in L$ that do not have any witnesses of NP-oracle circuit complexity $\mathsf{poly}(|x|)$. The idea is then to instantiate the PRG of [BFNW93] with the lexicographically first witness for $x \in L$. The advice required is $(x, |W_x|) \in \{0,1\}^{\mathsf{poly}(|x|)}$, where W_x is the witness set of x. Using this advice, one may non-deterministically guess all of the $2^{\mathsf{poly}(|x|)}$-many witnesses for x and output the lexicographically first as a canonical high-complexity truth-table, taking time at most $2^{\mathsf{poly}(|x|)}$.

The resulting PRG is secure against polynomial-size NP-oracle circuits. Moreover, the outputs of this PRG can be shown to have bounded nKt complexity, so any algorithm for an appropriate gap version of MnKtP could be used to distinguish PRG outputs from random. Therefore, $\mathsf{Gap\text{-}MnKtP} \notin \mathsf{pr\text{-}P}^{\mathsf{NP}}$, yielding the desired contradiction.

We omit proof sketches for the remaining uniform collapses, since all the proofs are quite similar to the one described above. Next, we briefly describe the proof of the result stating that, if SoI holds for nKt

[11] To the best of our knowledge, our approach to show failure of SoI in some relativized world is completely different than the one in [LR05, Lee06].

with error $e(n) = O(\log n)$, then $\mathsf{EXP}^{\|\mathsf{NP}} \subseteq \mathsf{NP}/\mathsf{poly} \cap \mathsf{coNP}/\mathsf{poly}$. Again, we will make use of the fact that SoI for nKt implies $\mathsf{Gap\text{-}MnKtP} \in \mathsf{pr\text{-}P}^{\|\mathsf{NP}}$. Using techniques similar to those of [ABK+06], one may show that any fixed language $L \in \mathsf{E}^{\|\mathsf{NP}}$ has a non-uniform non-adaptive poly-time Turing reduction to $\mathsf{Gap\text{-}MnKtP}$. Namely, using L to instantiate the PRG of [BFNW93], outputs of the PRG have bounded nKt complexity (by hard-wiring the number of YES queries of the NP oracle), while uniformly random strings almost always have near-maximum nKt complexity. The reconstruction property of the PRG then yields a reduction from L to $\mathsf{Gap\text{-}MnKtP}$, and therefore we have $\mathsf{E}^{\|\mathsf{NP}} \subseteq \mathsf{P}^{\|\mathsf{Gap\text{-}MnKtP}}/\mathsf{poly} \subseteq \mathsf{P}^{\|\mathsf{NP}}/\mathsf{poly}$. The stated inclusion $\mathsf{EXP}^{\|\mathsf{NP}} \subseteq \mathsf{NP}/\mathsf{poly} \cap \mathsf{coNP}/\mathsf{poly}$ follows from a downward collapse result of [SU05], given that $\mathsf{E}^{\|\mathsf{NP}}$ allows *low-degree extension*, together with a padding argument.

The result stating that SoI for rnKt implies $\mathsf{BPEXP}^{\|\mathsf{NP}} \subseteq \mathsf{NP}/\mathsf{poly} \cap \mathsf{coNP}/\mathsf{poly}$ follows a similar approach together with the unconditional inclusion $\mathsf{BPE}^{\|\mathsf{NP}} \subseteq \mathsf{AME}/O(n)$.

Theorem 5 gives a non-uniform upper-bound and a uniform lower-bound for meta-complexity that hold under SoI for nKt or rnKt. The proofs rely on the complexity collapse results described above, the correspondence theorem described next, a result from [ABK+06], and a straightforward diagonalization argument. We refer to Section 9 for more detail.

Correspondence Theorem (Theorem 6). We now describe the techniques used to prove Theorem 6, which exhibits a broad equivalence between asymmetry of information for rnKt, hardness of approximating rnKt, natural properties against rnKt, and explicit constructions in the context of $\mathsf{pr\text{-}BPTIME}[\cdot]^{\|\mathsf{NP}}$. The equivalence between Item (iii) and Item (iv) is straightforward, and likewise the implication Item (ii) $\implies$ Item (iii), so we will not discuss the proofs further. In order to prove the Item (iii) $\implies$ Item (ii), obtaining a worst-case algorithm for $\mathsf{Gap\text{-}MrnKtP}$ from a natural property against rnKt, we rely on the worst-case to average-case reduction framework of [Hir18]. A couple of differences in our setting compared to [Hir18] are as follows. We make use of a PRG based on a construction of Raz et al. [RRV02] as in [Hir20] to obtain $\mathsf{rK}^{\mathsf{poly}(n,t_D),\|D}$ upper bounds in the case that D is a t_D-time distinguisher. Moreover, the distinguisher algorithm runs in time $\mathsf{quasi\text{-}poly}(n)$ and makes non-adaptive NP-oracle queries, resulting in an upper bound on $\mathsf{rK}^{\mathsf{quasi\text{-}poly}(n),\|\mathsf{NP}}$ and thus $\mathsf{rnK}^{\mathsf{quasi\text{-}poly}(n)}$ (see Lemma 50).

The proof of Item (ii) $\implies$ Item (i), stating that easiness of $\mathsf{Gap\text{-}MrnKtP}$ for $\mathsf{pr\text{-}BPTIME}[n^{\mathsf{poly}(\log n)}]^{\|\mathsf{NP}}$ implies SoI for rnKt with error $e(n) = \mathsf{poly}\log n$, follows along the framework of [Hir22b, GK22] for proving SoI via hybrid argument, assuming easiness of meta-complexity. This framework is applied in a setting similar to that described above: the PRG of [RRV02, Hir20], distinguishers in $\mathsf{BPTIME}[n^{\mathsf{poly}(\log n)}]^{\|\mathsf{NP}}$, and bounds on $\mathsf{rK}^{\mathsf{quasi\text{-}poly}(n),\|\mathsf{NP}}$ collapsing to bounds on $\mathsf{rnK}^{\mathsf{quasi\text{-}poly}(n)}$.

To prove that Symmetry of Information for rnKt implies a worst-case $\mathsf{pr\text{-}BPTIME}[n^{\mathsf{poly}(\log n)}]^{\|\mathsf{NP}}$ algorithm for $\mathsf{Gap\text{-}MrnKtP}[n - \mathsf{poly}(\log n), n-1]$ (Item (i) $\implies$ Item (ii)), our starting point is a recent paper from [HMO26], which proved a similar statement for the case of rKt. Though we make use of some key ideas from that work, the setting of rnKt introduces additional challenges.

One concept from [HMO26] that we are able to borrow quite directly yields that, assuming SoI for rnKt with error $e(n)$, for some $t \leq 2^{O(e(n))}$, for any $x \in \{0,1\}^n$, $\mathsf{rnK}^t(x) \leq \mathsf{rnKt}(x) + O(e(n))$.[12] Moreover, arguing again along the same lines as [HMO26], we can show that if m is a witness for $\mathsf{rnK}^t(x) \leq \mathsf{rnKt}(x) + O(e(n))$ (namely, $|m| \leq \mathsf{rnKt}(x) + O(e(n))$ and m is an rnK-style description of x with running time t), then $\mathsf{rnKt}(m \mid x) \leq O(e(n))$. At this point, with this statement for the setting of rKt, the task of [HMO26] is nearly complete: omitting some technical details concerning randomness in the complexity

[12] Actually, [HMO26] obtained the statement with $O(e(n) \cdot \log n)$ as the leftmost term; we use an alternative proof that slightly improves this parameter. See Lemma 87.

measure, one may essentially brute-force over programs of length $O(e(n))$ to obtain a list containing m, run each program on the list to see if it outputs x, and output the length of the shortest one that does.

In our setting, of course, the same idea does not work directly. Even with a candidate program m in hand, we cannot simply run it to see if it outputs x. The same is true in the first part of the process, where we would brute-force over all $O(e(n))$-length strings and collect a list of their outputs. We are also not guaranteed that any such string represents a valid single-valued nondeterministic procedure. Luckily, these issues may be handled quite easily by the availability of an NP oracle. Roughly, with a string v in hand, we can ask queries of the following form, where r is chosen randomly and U denotes a universal TM:

- Does there exist a witness w such that $U(v, w, r) = z$ with $z \neq \perp$ in time t?
- Do there exist two witnesses w and w' such that $U(v, w, r) = z$ and $U(v, w', r) = z'$ in time t, with $z \neq z'$?
- Does there exist a witness w such that $U(v, w, r) = z \neq \perp$ in time t, and $z_i = 1$?

Clearly, if the first query is true and the second is false, then non-adaptive queries of the third type can be used to determine the unique nondeterministic output of v on randomness r.

However, a more serious obstacle remains: if we use non-adaptive NP queries to produce a list of strings containing m, and then we ask another round of queries to determine the outputs of the strings on this list, then we have introduced a layer of adaptivity. At a high level, we overcome this with a careful restructuring of the procedure so that the second layer of queries does not depend on the strings collected in the first. For every string $v \in O(e(n))$, we really only need to know the length of the smallest m such that (v, x) outputs m and m outputs x. The key observation is that this kind of question can be collapsed into a single NP query, which does not require m as input. Several auxiliary queries will be still be required: checking that (v, x) and m are single-valued nondeterministic, for example, and determining the value of their outputs on several independent choices of randomness to pinpoint the majority output in the sense of rnK. Fortunately, this can all be handled non-adaptively. See Lemma 88 for more detail.

As mentioned above, the proof of Theorem 7 follows from Theorem 5 (Item 2) and Theorem 6 via a win-win argument. We refer to Section 11.

Lower bounds for computing nKt via symmetry of information (Theorem 8). We now discuss the ingredients in the proof that $\mathsf{MnKtP} \notin \mathsf{NTIME}[2^{\varepsilon n}] \cap \mathsf{coNTIME}[2^{\varepsilon n}]$: i.e., the first part of Theorem 8. We will build on an approach to lower bounds due to [Hir22b], which was originally used to prove that $\mathsf{MrKtP} \notin \mathsf{BPTIME}[2^{\varepsilon n}]$ for some constant $\varepsilon > 0$. The proof consists of two steps. In the first step, the author shows that $\mathsf{MrKtP} \in \mathsf{BPTIME}[2^{\varepsilon n}]$ implies SoI for rKt, employing the "meta-complexity $\implies$ SoI" hybrid argument approach discussed above. The second step of the proof 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. Combining these steps, [Hir22b] reaches a contradiction and thus may conclude that MrKtP is hard.

A limitation of this argument, which blocks its direct translation to MnKtP, is the randomness required in the reconstruction procedure of the PRG of [RRV02, Hir20], which is employed in the first step. 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 via reconstruction. (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 nondeterminism 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 $m/(\log m)^2$. 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. Next, we may 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. 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 12.3 for more details.

One problem still remains: the reconstruction procedure of the PRG 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 12.1). This eliminates the need for additional advice, as we can just output the string that appears the most often.

We have given a brief overview of the proof of the main part of Theorem 8. Next, we discuss the "moreover" part. We observe that the first step of the proof essentially extends to any dense language $L \subseteq \mathcal{R}_{\mathsf{nKt}[n]}$. On the other hand, for the second step of the proof, letting $y \in \{0,1\}^n$ be such that $\mathsf{nKt}(y) \geq n$ and $x \in \{0,1\}^n$ such that $\mathsf{nKt}(x \mid y) \geq n$, SoI guarantees that $\mathsf{nKt}(x,y) \gtrsim 2n$, but it does not guarantee that $(x,y) \in L$, so it is possible that we cannot find x by brute-force with access to L. Fortunately, the first step of the proof actually gives us something stronger than SoI: for any $x, y \in \{0,1\}^*$, letting G_1 and G_2 be PRGs with appropriate output lengths approximately $\mathsf{nKt}(x \mid y) \geq n$ and $\mathsf{nKt}(y)$, there exist z, z' such that $G_1(x,z) \circ G_2(y,z') \in L$. (See Lemma 107 for more detail.) Then, given $y \in L$, we enumerate not only over x, but also over z, z' to find $G_1(x,z) \circ G_2(y,z') \in L$.

This works for any dense language $L \in \mathcal{R}_{\mathsf{nKt}[(1-o(1))\cdot n]}$, but not for $L \in \mathcal{R}_{\mathsf{nKt}[\gamma\cdot n]}$ for $\gamma \in (0,1)$ in general. In particular, assuming easiness of L, we can only find $y \in L$ such that $\mathsf{nKt}(y) \geq \gamma \cdot n$, and then we can find $x \in \{0,1\}^n$ and $y' = G_1(x,z) \circ G_2(y_1, z') \in L$ such that

$$\mathsf{nKt}(y') \gtrsim \gamma \cdot (n + \mathsf{nKt}(y)) \geq (\gamma + \gamma^2) \cdot n.$$

We can see the problem here: when $\gamma \leq (\sqrt{5}-1)/2 \approx 0.618$, we are not guaranteed that $\mathsf{nKt}(y') \geq (1+\Omega(1)) \cdot n$, so we cannot derive a contradiction. To get around this obstacle, the key idea is to *iterate the construction k times* for a large enough constant k, applying our extended SoI statement again to y', and so on. The detailed proof can be found in Section 12.4. On the other hand, Section 12.5 discusses the limits of our techniques for $\gamma \leq 1/2$.

Lower bounds for computing rnKt via explicit constructions (Theorem 9). We now move on to discuss the techniques behind Theorem 9. The proof of Item 1 naturally extends the lower bound proof for rKt in [Hir22b]. Conceptually, this setting is simpler than the proof of Theorem 8 since derandomization is not required. Still, the combination of randomness and nondeterminism makes some parts of the argument slightly more involved, and we include a complete proof in Section 13.1 to keep the presentation self-contained.

The proofs of Item 2 and Item 3 exploit new results from [CLL25]: namely, an *explicit construction algorithm* for dense coAM properties in the form of Arthur-Merlin protocols with non-binary output. At the start of the protocol, prover P (which is computationally unbounded) and verifier V both receive an advice string α. After $k = O(1)$ rounds of interaction, V outputs a string x based on α, the public randomness $\{r_j\}_{j\in[k]}$, and the prover's responses $\{w_j\}_{j\in[k]}$. Such a protocol is said to be *single-valued* if it satisfies certain *conformity* and *resiliency* properties; in particular, these properties imply that $(P,V)(\alpha)$ always outputs a canonical string x, and that, for any prover P', $(P',V)(\alpha)$ belongs to $\{x,\perp\}$ with high probability. See Section 13.2 for detail.

The main technical result of [CLL25] proves that for every dense enough $L \in \mathsf{coAM}$, there exists a sequence of strings $\{x_n\}$ and a non-binary public-coin protocol (P,V) satisfying

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

A sketch of the proof of Item 2 is as follows. We first observe that, after applying round reduction to the protocol above, the sequence $\{x_n\}$ admits small rnK complexity: say, $\mathsf{rnKt}(x_n) \leq \theta(n)$ for some $\theta(n) < n-1$. 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 an Arthur-Merlin protocol as above, we know that for infinitely many n, $L \cap \{0,1\}^n$ contains a string with rnKt complexity smaller than $\theta(n)+1$, contradicting the definition of L. In our formal proof of Item 2, we extend this idea to quasi-polynomial time regime by a simple padding argument.

One important corollary of the explicit construction protocol in [CLL25] states that $(\mathsf{AMEXP} \cap \mathsf{coAMEXP})/2^{n^\varepsilon} \not\subseteq \mathsf{SIZE}^{\mathcal{O}}[2^n/n]$ for any oracle $\mathcal{O} \in \mathsf{AM} \cap \mathsf{coAM}$ (see [CLL25, Theorem 1.2]). This follows from the close connection between explicit constructions and circuit lower bounds: roughly speaking, if Π_{hard} is defined as the set of strings in $\{0,1\}^{2^n}$ that are not truth-tables of $\mathcal{O}$-oracle circuits of size at most $2^n/n$, then Π_{hard} is dense and $\Pi_{\mathsf{hard}} \in \mathsf{coAM}$. By the explicit construction protocol from above, one obtains a sequence of truth-tables defining a language L_{hard} such that $L_{\mathsf{hard}} \in (\mathsf{AMEXP} \cap \mathsf{coAMEXP})/2^{n^\varepsilon}$ but $L_{\mathsf{hard}} \notin \mathsf{SIZE}^{\mathcal{O}}[2^n/n]$.

Our idea for proving Theorem 9, Item 3 is to instantiate the PRG of [IW97] with the truth-table of L_{hard}, obtaining a family of multi-sets $\{T_s\}_{s\in\mathbb{N}}$ which are pseudorandom 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 the [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 uniformly 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.

Unconditional failure of conditional SoI (Theorem 10) Lastly, we describe the proof of Theorem 10, which states that conditional SoI does not hold: for any error term $e(n)=\mathsf{poly}\log n$, there exist strings x,y, and w such that

$$\mathsf{nKt}(x,y\mid w)<\mathsf{nKt}(x\mid y,w)+\mathsf{nKt}(y\mid w)-e(|x|+|y|).$$

The proof of this statement builds on the unconditional lower bound for computing $\mathsf{Gap\text{-}MnKtP}$ as in Theorem 8. That is, assuming conditional SoI, we exhibit an algorithm for $\mathsf{Gap\text{-}MnKtP}$ contradicting this lower bound.

As a first attempt, assume conditional SoI, and consider the following deterministic algorithm for $\mathsf{Gap\text{-}MnKtP}[0.6n,0.9n]$.

> On input $x\in\{0,1\}^n$, partition x into $k=n/(100e(n))$ blocks $(x_1,...,x_k)$, with each x_i being of length roughly $100e(n)$. For each $i\in[k]$, compute $v_i:=\mathsf{nKt}(x_i\mid x_1,...,x_{i-1})$ by brute force, and accept iff $\sum_{i\in[k]}v_i\le 0.75n$.

On one hand, it holds trivially that

$$\mathsf{nKt}(x)\le\sum_{i\in[k]}\mathsf{nKt}(x_i\mid x_1,...,x_{i-1})+O(\log n)=\sum v_i.$$

On the other hand, by our assumption of conditional SoI applied k times,[13] we have that

$$\mathsf{nKt}(x)\ge\sum\mathsf{nKt}(x_i\mid x_1,...,x_{i-1})-k\cdot e(n)=\sum v_i-\frac{n}{100}.$$

Since $\sum v_i$ closely approximates $\mathsf{nKt}(x)$, the algorithm above clearly solves $\mathsf{Gap\text{-}MnKtP}[0.6n,0.9n]$.

The problem with this approach is that our algorithm is highly inefficient. Specifically, to determine $\mathsf{nKt}(x_i\mid x_1,...,x_{i-1})$ by brute force, one must consider non-deterministic programs running in time up to $2^{100e(n)}=2^{\mathsf{poly}\log n}$ and therefore must exhaustively search over witnesses of that length, taking time $2^{2^{\mathsf{poly}\log n}}$. The lower bound of Theorem 8 does not rule out an algorithm with this time complexity.

To improve the running time of the algorithm, we argue again as in [HMO26] and the proof of Theorem 6; namely, we argue that rnKt approximately upper-bounds rnK^t. Crucially, by assuming conditional SoI, we get a stronger version of the statement that allows us to condition on an auxiliary string w. In particular, for any strings x,w,

$$\mathsf{nK}^{O(2^{e(|x|)})}(x\mid w)\le\mathsf{nKt}(x\mid w)+O(e(|x|)).$$

[13] In fact, standard (non-conditional) SoI would suffice for this step.

Returning to our algorithm for Gap-MnKtP$[0.6n, 0.9n]$, instead of computing $\mathsf{nKt}(x_i \mid x_1, ..., x_{i-1})$ by brute force, we may instead compute $\mathsf{nK}^t(x_i \mid x_1, ..., x_{i-1})$ by brute force, where $t = O(2^{e(|x_i|)}) = O(2^{e(100e(n))})$. This only incurs an additional additive error term of $n/(\mathsf{poly}\log n)$. Moreover, the algorithm now takes time only $2^{100e(n)} \cdot 2^{O(2^{e(100e(n))})} \leq 2^{2^{\mathsf{poly}(\log\log n)}} = 2^{o(n)}$, contradicting the lower bound of Theorem 8.

3 Preliminaries

3.1 Notation

We denote the empty string by ϵ, and the length of a string x by $|x|$. For strings $x \in \{0,1\}^n$ and $y \in \{0,1\}^m$, their concatenation is written as $x \circ y \in \{0,1\}^{n+m}$.

If $x \sim \mathcal{D}$, then x is sampled from the distribution $\mathcal{D}$. For any a in the support of $\mathcal{D}$, $\mathcal{D}(a)$ denotes its probability. We write $\mathcal{U}_\ell$ for the uniform distribution over $\{0,1\}^\ell$.

3.2 Computational Model

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.[14]

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 $\mathsf{K}^{C\cdot t\log t}_U(x) \leq \mathsf{K}^t_{U'}(x) + C$. 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 $(\alpha; x; w; r)$.

A *program* Π is defined as a pair $\Pi := (M, \alpha)$, where M is a Turing 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 $\Pi = (M, \alpha)$ on input (x, w, r) is defined to be the number of steps U runs on input $(\langle M\rangle; \alpha; x; w; r)$, where $\langle M\rangle$ is an encoding of the machine M.[15]

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

$$s_\alpha\, 01\, \langle M\rangle\, \alpha,$$

where s_α is the binary representation of the integer specifying the length of α, with each bit duplicated. The size of Π, denoted $|\Pi|$, is the total length of this description.

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

[14] 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.

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

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

3.3 Time-Bounded Kolmogorov Complexity

We fix a time-efficient universal machine U, and define time-bounded Kolmogorov complexity with respect to U (see [LV19] for the required background in Kolmogorov complexity). In other words, whenever we say that a program Π outputs x in t steps, formally, this means that $U(\Pi) = x$ when U runs for at most t steps on input Π. Our results are robust with respect to encoding choices.

Next, we provide formal definitions for various notions of time-bounded Kolmogorov complexity. We begin with deterministic versions of time-bounded Kolmogorov complexity.

Definition 11 (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 12 (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 13 (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

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

Definition 14 (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

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

Definition 15 (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

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

Equivalently,

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

Definition 16 (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

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

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

Definition 17 (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

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

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

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

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 19 (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

$$\mathsf{rnK}^t_\lambda(x \mid y) \triangleq \min_{\Pi \in \{0,1\}^*} \left\{ |\Pi| \;\middle|\; \Pr_{r\sim\{0,1\}^t}\left[\begin{array}{l} \forall w \in \{0,1\}^t, \Pi(y; w; r) \text{ outputs } x \text{ or } \perp \text{ in } t \text{ steps;} \\ \exists w \in \{0,1\}^t, \Pi(y; w; r) \text{ outputs } x \text{ in } t \text{ steps} \end{array} \right] \geq \lambda \right\}.$$

Definition 20 (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

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

Definition 21 (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

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

Equivalently,

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

Definition 22 (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

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

3.4 Meta-Computational Problems

We let MKtP denote the set of input pairs $(x, 1^s)$ such that $x \in \{0,1\}^*$ and $\mathsf{Kt}(x) \le s$.
We also consider the promise problem $\mathsf{GapMnKtP}[\alpha, \beta]$ on inputs $x \in \{0,1\}^*$ defined by

$$\text{YES} \triangleq \{x : \mathsf{nKt}(x) \le \alpha(|x|)\}, \qquad \text{NO} \triangleq \{x : \mathsf{nKt}(x) \ge \beta(|x|)\}.$$

Similarly, we can consider meta-computational problems associated with other Kolmogorov complexity measures, e.g., $\mathsf{GapMKtP}[\alpha, \beta]$ captures the problem of estimating Kt complexity.

3.5 Basic Tools in Meta-Complexity

Here, we state some useful tools about Kolmogorov complexity.

Lemma 23 (Standard Counting Argument)**.** *For any $\lambda \in (1/2, 1)$, $t \in \mathbb{N}$, 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*

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

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

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

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 programs with length at most $n - \gamma$ is less than $2^{n-\gamma}$, therefore

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

The remaining cases (except $\kappa = \mathsf{rK}^t_\lambda$ and $\kappa = \mathsf{rnK}^t_\lambda$) follows similarly, since each randomized programs 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 24 (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 25 (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 26 (Following [GKLO22, Lemma 18])**.** *For every computable language L, there exists a constant $c > 0$ such that for any $x \in \{0,1\}^*$ and time bound $t \in \mathbb{N}$, $\mathsf{K}(x \mid t) \le \mathsf{pK}^{t,L}(x) + c \cdot \log|x|$.*

3.6 Pseudorandomness

For a finite set A, we write $x \sim A$ to denote that x is uniformly distributed over A.

Let $\mathcal{D}_n$ be a distribution supported over $\{0,1\}^n$. Let $\varepsilon \in [0,1]$. Finally, let $f\colon \{0,1\}^n \to \{0,1\}$. We say that $\mathcal{D}_n$ *ε-fools* f if

$$\left|\Pr_{x\sim\mathcal{D}_n}[f(x)=1] - \Pr_{x\sim\{0,1\}^n}[f(x)=1]\right| \le \varepsilon.$$

Theorem 27 ([BFNW93, KvM02]; see also Theorem 17 of [ABK+06])**.** *For every $\varepsilon > 0$ and $b \in \mathbb{N}$, there exist $\delta < \varepsilon$, a constant $a \in \mathbb{N}$, and a function*

$$F\colon \{0,1\}^{2^{n^\delta}} \times \{0,1\}^{n^\varepsilon} \to \{0,1\}^{n^b},$$

computable in time $\mathsf{poly}(2^{n^\delta})$ and satisfying the following for all sufficiently large $n \in \mathbb{N}$.

For any function $f : \{0,1\}^{n^\delta} \to \{0,1\}$, let $G_f(\text{-})$ denote $F(\mathsf{tt}(f), \text{-})$. Then for any such f and $D : \{0,1\}^{n^b} \to \{0,1\}$ such that

$$\left|\Pr_{z\sim\mathcal{U}_{n^\varepsilon}}[D(G_f(z))=1] - \Pr_{w\sim\mathcal{U}_{n^b}}[D(w)=1]\right| \ge \frac{1}{n^b},$$

there exists a non-adaptive oracle circuit $R^{(\text{-})}$ of size n^a such that $R^D(x) = f(x)$ for all $x \in \{0,1\}^{n^\delta}$.

As a consequence, there exists a constant $c \in \mathbb{N}$ (chosen so that $n^{c\delta}$ exceeds $n^a \cdot n^b$) such that for any oracle O, if $f : \{0,1\}^{n^\delta} \to \{0,1\}$ is a function with O-oracle Boolean circuit complexity at least $n^{c\delta}$, then $G_f : \{0,1\}^{n^\varepsilon} \to \{0,1\}^{n^b}$ is a pseudorandom generator that (n^{-b})-fools every O-oracle Boolean circuit $D : \{0,1\}^{n^b} \to \{0,1\}$ of size at most n^b.

Theorem 28 ([IW97]). *For every $\varepsilon > 0$, there exist a polynomial-time computable function F and $c, d \in \mathbb{N}$ such that such that the following holds for any oracle O.*

$$F : \{0,1\}^{n^c} \times \{0,1\}^{d \log n} \to \{0,1\}^n,$$

and if $f : \{0,1\}^{c \log n} \to \{0,1\}$ is a function with O-oracle Boolean circuit complexity at least $n^{\varepsilon c}$, then the function $G_f(\text{-}) := F(\mathtt{tt}(f), \text{-})$ is a pseudorandom generator mapping $\{0,1\}^{d \log n}$ to $\{0,1\}^n$ that $(1/n)$-fooles every O-oracle Boolean circuit $D : \{0,1\}^n \to \{0,1\}$ of size at most n.

Theorem 29 ([IW01, TV07]). *Let O be any oracle. Assume $\mathsf{PSPACE} \not\subseteq \mathsf{BPP}^O$. Then for every $\varepsilon > 0$ and $b \in \mathbb{N}$, there is a sequence $\{G_n\}_{n \in \mathbb{N}}$, where $G_n \colon \{0,1\}^{n^\varepsilon} \to \{0,1\}^{n^b}$ is computable in time $2^{O(n^\varepsilon)}$, such that the following holds. For every distribution family $\{\mathcal{C}_n\}_{n \in \mathbb{N}}$ of Boolean O-oracle circuits samplable in time n^b, there are infinitely many $n \in \mathbb{N}$ such that with probability at least $1 - n^{-b}$ over C sampled from $\mathcal{C}_n$, G_n (n^{-b})-fools C.*

Corollary 30. *Let $O \in \mathsf{PSPACE}$. Assume $\mathsf{EXP} \not\subseteq \mathsf{BPP}^O$. Then for every $\varepsilon > 0$ and $b \in \mathbb{N}$, there is a sequence $\{G_n\}_{n \in \mathbb{N}}$, where $G_n \colon \{0,1\}^{n^\varepsilon} \to \{0,1\}^{n^b}$ is computable in time $2^{O(n^\varepsilon)}$, such that the following holds. For every distribution family $\{\mathcal{C}_n\}_{n \in \mathbb{N}}$ of Boolean O-oracle circuits samplable in time n^b, there are infinitely many $n \in \mathbb{N}$ such that with probability at least $1 - n^{-b}$ over C sampled from $\mathcal{C}_n$, G_n (n^{-b})-fools C.*

Proof Sketch. Assume $\mathsf{EXP} \neq \mathsf{BPP}^O$. We consider two cases.

If $\mathsf{PSPACE} \not\subseteq \mathsf{BPP}^O$, then the desired PRG follows directly from Theorem 29.

Otherwise, suppose $\mathsf{PSPACE} \subseteq \mathsf{BPP}^O$. Combined with the assumption that $\mathsf{EXP} \not\subseteq \mathsf{BPP}^O$, this implies that $\mathsf{EXP} \neq \mathsf{PSPACE}$. We claim that $\mathsf{EXP} \not\subseteq \mathsf{SIZE}^O[\mathsf{poly}]$. Indeed, if $\mathsf{EXP} \subseteq \mathsf{SIZE}^O[\mathsf{poly}]$, then

$$\begin{aligned} \mathsf{EXP} &\subseteq \mathsf{MA}^O && \text{(by [BFNW93])} \\ &\subseteq \mathsf{PSPACE}, \end{aligned}$$

where the last inclusion follows from the fact that $O \in \mathsf{PSPACE}$, so both the MA computation and the oracle queries can be simulated within PSPACE. This yields $\mathsf{EXP} \subseteq \mathsf{PSPACE}$, a contradiction.

Therefore, $\mathsf{EXP} \not\subseteq \mathsf{SIZE}^O[\mathsf{poly}]$. Applying Theorem 27, we obtain an infinitely-often secure PRG with sub-polynomial seed length based on the hard truth table of a language in EXP. □

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

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

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

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

Here, we have $d = O(\log^3(n/\varepsilon))$ *and* $r = O(m)$*. 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* non-adaptive *oracle access to* D*.*

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

Corollary 32. *For any* $n, m \in \mathbb{N}$ *and* $\varepsilon > 0$*, let* $G : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *be as in Lemma 31. Consider any* $x \in \{0,1\}^n$*,* $\ell \leq \mathsf{poly}(n/\varepsilon)$*, and* $v \in \{0,1\}^{\leq \ell}$*, and let* $D : \{0,1\}^m \times \{0,1\}^\ell \times \{0,1\}^{\leq \ell} \to \{0,1\}$ *be any function such that*

$$\Pr_{\substack{z\sim\mathcal{U}_d \\ u\sim\mathcal{U}_\ell}}[D(G(x;z),u,v)=1] - \Pr_{\substack{w\sim\mathcal{U}_m \\ u\sim\mathcal{U}_\ell}}[D(w,u,v)=1] \geq 2\varepsilon.$$

Then, for some $t = \mathsf{poly}(n/\varepsilon)$*, it holds that*

$$\mathsf{rK}^{t,\|D}(x \mid v) \leq m + O(\log^3(n/\varepsilon)).$$

Proof Sketch. By Markov's inequality, we have that

$$\Pr_{u\sim\mathcal{U}_\ell}\left[\Pr_{z\sim\mathcal{U}_d}[D(G(x;z),u,v)=1] - \Pr_{w\sim\mathcal{U}_m}[D(w,u,v)=1] \geq \varepsilon\right] \geq \varepsilon.$$

In other words, with probability at least ε over $u \sim \mathcal{U}_\ell$, $D(\cdot,u,v)$ is an ε-distinguisher for $G(x;\cdot)$. By Lemma 31, for such u, the reconstruction succeeds with probability $1/\mathsf{poly}(m/\varepsilon)$. Hence we have

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

By an averaging argument, there exists some z such that

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

Therefore we can design a program that saves z, $A(x,z)$, and the description of R. When given access to v, it samples w, u uniformly, then runs for $\mathsf{poly}(n/\varepsilon)$ steps, queries D non-adaptively, and finally prints x with probability at least $1/\mathsf{poly}(n/\varepsilon)$. By definition of rK, this gives

$$\mathsf{rK}^{\mathsf{poly}(n/\varepsilon),\|D}_{1/\mathsf{poly}(m/\varepsilon)}(x \mid v) \leq |z| + |A(x,z)| + |R| \leq m + O(\log^3(n/\varepsilon)).$$

By success amplification of rK^t [LO21], we have

$$\mathsf{rK}^{\mathsf{poly}(n/\varepsilon),\|D}(x \mid v) \leq \mathsf{rK}^{\mathsf{poly}(n/\varepsilon),\|D}_{1/\mathsf{poly}(m/\varepsilon)}(x \mid v) + O(\log(m/\varepsilon)) \leq m + O(\log^3(n/\varepsilon)).$$

□

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 33 (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 34 (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*

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

3.7 Technical Results

Lemma 35 ([ABK+06])**.** *If* $\mathsf{MKtP} \in \mathsf{P}$ *then* $\mathsf{PSPACE} \subseteq \mathsf{ZPP}$.

Lemma 36 ([AKRR11])**.** *For every $0 < \varepsilon < 1$, we have* $\mathsf{GapMnKtP}[n^\varepsilon, n-1] \notin \mathsf{NP} \cap \mathsf{coNP}$.[16]

We will use the downward collapse theorem of Shaltiel–Umans.

Definition 37 (Low-degree extension [SU06, Def. 3.1])**.** We say a complexity class $\mathcal{C}$ allows low-degree extension if $\mathsf{E}^{\mathcal{C}_{\leq O(n)}} \subseteq \mathcal{C}$, where $\mathcal{C}_{\leq O(n)}$ means that the machine only makes linear-length queries to $\mathcal{C}$.

Lemma 38 (Shaltiel–Umans downward collapse [SU06, Thm. 3.2])**.** *For any complexity class $\mathcal{C}$ that allows low-degree extension, we have*

$$\mathcal{C} \subseteq \mathsf{P}^{\|\mathsf{NP}}/\mathsf{poly} \implies \mathcal{C} \subseteq \mathsf{NP}/\mathsf{poly} \cap \mathsf{coNP}/\mathsf{poly}.$$

It is not hard to verify that both $\mathsf{E}^{\|\mathsf{NP}}$ and $\mathsf{BPE}^{\|\mathsf{NP}}$ allows low-degree extension. Hence we get the following corollaries:

Corollary 39. $\mathsf{E}^{\|\mathsf{NP}} \subseteq \mathsf{P}^{\|\mathsf{NP}}/\mathsf{poly} \implies \mathsf{E}^{\|\mathsf{NP}} \subseteq \mathsf{NP}/\mathsf{poly} \cap \mathsf{coNP}/\mathsf{poly}$

Corollary 40. $\mathsf{BPE}^{\|\mathsf{NP}} \subseteq \mathsf{P}^{\|\mathsf{NP}}/\mathsf{poly} \implies \mathsf{BPE}^{\|\mathsf{NP}} \subseteq \mathsf{NP}/\mathsf{poly} \cap \mathsf{coNP}/\mathsf{poly}$

4 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.

[16] In other words, there is no language $L \in \mathsf{NP} \cap \mathsf{coNP}$ that agrees with $\mathsf{GapMnKtP}[n^\varepsilon, n-1]$.

4.1 nK

Lemma 41. *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 42. *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 41, 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

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

as desired. □

Lemma 43. *There exists a constant $c' \in \mathbb{N}$ as follows. 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 [HLO26, 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 43, we get the following.

Corollary 44. *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 45. *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 sufficiently large $t \in \mathbb{N}$,*

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

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

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

Proof. Let $L \in \mathsf{NTIME}[T(n)] \cap \mathsf{coNTIME}[T(n)]$, $x, y \in \{0,1\}^*$, and $t \in \mathbb{N}$. Without loss of generality, assume t is a power of 2; if not, replace t with $2^{\lceil \log t \rceil}$ in the argument below at the cost of at most a factor of 2 in the resulting running time. Define $k = \mathsf{nK}^{t, L^{\leq \ell}}(x \mid y)$, and let $\Pi \in \{0,1\}^k$ be a program that non-deterministically outputs x in time t given y and *oracle access to* $L^{\leq \ell}$.

We now simulate the program Π *without* oracle access to L. To do this, whenever Π makes a query $q \in \{0,1\}^{\leq \ell}$ to L, we non-deterministically guess an answer ("yes" or "no") along with a witness $w_q \in \{0,1\}^{\leq T(\ell)}$ certifying that answer. Since $L \in \mathsf{NTIME}[T(n)] \cap \mathsf{coNTIME}[T(n)]$, each such guess can be verified in time $\widetilde{O}(T(\ell))$. Whenever the verification fails, halt with $\perp$ being the output. Observe that the resulting program is single-valued, by definition of Π and since all oracle queries are verified.

This yields a nondeterministic program of size at most $|\Pi| + O(\log\log t)$ that outputs x in time $t \cdot T(\ell) \cdot \mathsf{poly}\log(T(\ell) \cdot t)$ given y (without oracle access to L). Therefore,

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

for some constant $c > 0$.[17]

The first bound is the case $\ell = t$. This concludes the proof.

□

Again, as a corollary of Lemma 45, we get the following.

Corollary 46. *For every $L \in \mathsf{NP} \cap \mathsf{coNP}$, there exists a constant $c > 0$ such that for every $x, y \in \{0,1\}^*$,*

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

4.2 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 47 (Success Amplification for rnK^t)**.** *For any oracle $\mathcal{O}$, any $n \in \mathbb{N}$, any time bound $t \in \mathbb{N}$, any $\varepsilon, \delta \in (0,1)$ and any strings $x, y \in \{0,1\}^*$ such that $|x| = n$ and $|x| \leq t$, we have*

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

Proof. Let M be the program for $\mathsf{rnK}^{t,\mathcal{O}}_\delta(x \mid y)$. For simplicity, assume t is a power of 2; if not, the argument can be adapted by replacing t with $2^{\lceil \log t \rceil} \leq 2t$. By definition, we have

$$\Pr_{r \sim \mathcal{U}_t}\left[\begin{array}{l}\exists w \in \{0,1\}^t, M^{\mathcal{O}}(y, r, w) \text{ outputs } x \text{ in } t \text{ steps,} \\ \forall w \in \{0,1\}^t, M^{\mathcal{O}}(y, r, w) \text{ outputs } x \text{ or } \perp \text{ in } t \text{ steps.}\end{array}\right] \geq \delta.$$

[17] Note that the L-verifier descriptions and their constant-factor simulation overhead depend on L but not on t, and are therefore absorbed into the stated bounds once t is sufficiently large; the exponent c comes only from the universal machine's simulation overhead and is independent of L.

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

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

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

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

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

Proof of Claim 48. 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

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

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

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

For Item (ii), let x' be any string in $\{0,1\}^n \setminus A_{\delta/8}$. Then we have $\mathbf{Pr}_{r \sim \mathcal{U}_t}[s_r = x'] \leq \delta/8$. Then by Chernoff bound, we get

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

Applying union bound over all $x' \in \{0,1\}^n \setminus A_{\delta/8}$, 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 101 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 $r_1, \ldots, r_k \sim \mathcal{U}_t$.

(b) For each $i \in [k]$, non-deterministically guess n strings $w_{i,1}, \ldots w_{i,n} \in \{0,1\}^t$. By simulating M for at most t steps nk times (and treating non-halting runs as outputting $\perp$), 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 $a = (1/n) \cdot \lceil nk(\overline{d} - \frac{\delta}{32n}) \rceil$. 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 $\hat{s}_{r_1}, \ldots, \hat{s}_{r_k}$. 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 $(M, n, \log t, k, \delta, a, u, H(x))$, which takes $|M| + O(\log \frac{n \log(t/\varepsilon)}{\delta})$ bits. (Here, a is stored as the integer $\lceil nk(\overline{d} - \frac{\delta}{32n}) \rceil$.) The running time of the above algorithm is also bounded by $t \cdot \mathsf{poly}(\frac{n \log(t/\varepsilon)}{\delta})$. It remains to estimate the success probability.

Claim 49. *If the random strings $r_1, \ldots, r_k$ sampled by our algorithm satisfy Items (i) to (iii) of Claim 48, 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 49. 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 $r_1, \ldots, r_k$ also satisfies Item (iii) of Claim 48, we have

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

Subtracting the two inequalities, we get

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

By averaging argument, we get

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

Since d_{r_i} counts the number of 1 in the bitwise OR of all computation paths, $d_{r_i} \geq \hat{d}_{r_i}$; 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 $1/n$, $d_{r_i} \neq \hat{d}_{r_i}$ implies $d_{r_i} - \hat{d}_{r_i} \geq 1/n$. Therefore $s_{r_i} \neq \hat{s}_{r_i}$ implies $d_{r_i} - \hat{d}_{r_i} \geq 1/n$, and we get

$$|\{i \in [k] \mid s_{r_i} \neq \hat{s}_{r_i}\}| \leq \frac{\delta k}{16}. \tag{1}$$

By Item (i) of Claim 48, we get

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

Combining Equations (1) and (2), we get

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

Therefore, by step (d) of the algorithm, $x \in \hat{A}$ holds. Also, by Item (ii) of Claim 48, for any $x' \notin A_{\delta/8}$, we have

$$|\{i \in [k] \mid s_{r_i} = x'\}| \leq \frac{\delta k}{4}. \tag{3}$$

Combining Equations (1) and (3), we get

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

Therefore, by step (d) of the algorithm, $x' \notin \hat{A}$ holds. Thus we get $\hat{A} \subseteq A_{\delta/8}$. By Lemma 101, 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 (d). 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) gives $\hat{d}_{r_1} + \cdots + \hat{d}_{r_k} = d_{r_1} + \cdots + d_{r_k} \geq k\left(\overline{d} - \frac{\delta}{32n}\right)$. Since $n \cdot (\hat{d}_{r_1} + \cdots + \hat{d}_{r_k})$ is an integer, it follows that $\hat{d}_{r_1} + \cdots + \hat{d}_{r_k} \geq a$, so our algorithm reaches step (d). ◇

Therefore the success probability is at least $1 - \varepsilon$ by Claim 48, which finishes our proof. □

Lemma 50. *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 $x, y \in \{0,1\}^*$,*

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

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 43, 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 51. *There exists a constant c' such that the following holds. For any time-constructible, non-decreasing $T : \mathbb{N} \to \mathbb{N}$, $T(n) \geq n$, for every $L \in \mathsf{AMTIME}[T(n)] \cap \mathsf{coAMTIME}[T(n)]$, all $x, y \in \{0,1\}^*$, and sufficiently large $t \in \mathbb{N}$ with $t > |x|$,*

$$\mathsf{rnK}^{T(t)^{c'}}(x \mid y) \leq \mathsf{rnK}^{t,L}(x \mid y) + O(\log t).$$

Moreover, restricting query length to at most ℓ for an integer $\ell \geq 1$,

$$\mathsf{rnK}^{t\cdot(T(\ell)\cdot|x|\cdot\log t)^{c'}}(x \mid y) \leq \mathsf{rnK}^{t,L^{\leq \ell}}(x \mid y) + O(\log(\ell \cdot |x| \cdot \log t)).$$

Proof Sketch. The proof is similar to that of Lemma 45. 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 $\mathsf{poly}(T(\ell))$.

The main difference here is that L lies 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 the verifications have perfect completeness and soundness error at most $O(2^{-2\ell})$, within time overhead of $O(\ell)$. We use the same choice of randomness on every query on every nondeterministic branch. Then, by union bound over possible inputs to L of length at most ℓ, the overall success probability of the simulated program is at least $2/3 - O(2^{-\ell})$, which can be amplified back to $2/3$ using Lemma 47 at the expense of $t \cdot \mathsf{poly}(T(\ell) \cdot |x| \cdot \log t)$ time overhead and $O(\log(\ell \cdot |x| \cdot \log t))$ description overhead. □

Corollary 52. *For every $L \in \mathsf{AM} \cap \mathsf{coAM}$, there is a constant $c > 0$ such that for every $x, y \in \{0,1\}^*$,*

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

4.3 pnK

Lemma 53. *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 41. 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

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

as desired. □

Corollary 54. *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 $x, y \in \{0,1\}^*$,*

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

5 Average-Case SoI for Nondeterministic Computation: Proof of Theorem 1

5.1 Average-Case SoI for pnKt

In this subsection, we show that Symmetry of Information holds for pnKt *on average* over efficiently samplable distributions. Toward proving this result, 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 [HLO26]. 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 [HLO26], which would include a term for $\mathsf{pnK}^{\mathsf{poly}(t)}(y)$ rather than $\mathsf{pK}^{\mathsf{poly}(t)}(y)$.

Theorem 55. *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 $x, y \in \{0,1\}^n$,*

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

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 $z \in \{0,1\}^{2n}$,

$$\mathcal{D}_{n,t}(z) \ge \frac{1}{2^{\mathsf{pK}^t(z)} \cdot n^c}. \tag{4}$$

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 $t \ge 4n$ along with strings $x, y \in \{0,1\}^n$. By the worst-case conditional

coding theorem for pnK^t with respect to $\mathcal{D}$ ([HLO26, Theorem 6]), for some polynomial q,

$$\begin{aligned}
\mathsf{pnK}^{q(t)}(y \mid x) &\leq \log \frac{1}{\mathcal{D}_{n,t}(y \mid x)} + \log q(t) \\
&= \log \frac{\mathcal{D}^{(2)}_{n,t}(x)}{\mathcal{D}_{n,t}(x,y)} + \log q(t) \\
&= \log \frac{1}{\mathcal{D}_{n,t}(x,y)} - \log \frac{1}{\mathcal{D}^{(2)}_{n,t}(x)} + \log q(t) \\
&\leq \mathsf{pK}^t(x,y) - \log \frac{1}{\mathcal{D}^{(2)}_{n,t}(x)} + \log q(t) + c \log n. && \text{(by Equation (4))}
\end{aligned}$$

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 [HLO26]; we include the details for completeness.

Below, we state Theorem 1 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, since $\mathsf{pKt}(x) \leq \mathsf{pnKt}(x) + O(\log n)$ with high probability over strings x sampled efficiently.

Theorem 56. *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 $n, k \in \mathbb{N}$,*

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

Proof. The proof is following [HLO26, 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 55. 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 25) and Lemma 24, the following holds: for any $k \in \mathbb{N}$, with probability at least $1 - k^{-1}$ over $(x, y) \sim D_n$,

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

Combining the above with Theorem 55 and letting $c \in \mathbb{N}$ be a sufficiently large constant, we have that with probability at least $1 - k^{-1}$ over $(x, y) \sim D_n$,

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

as desired. □

5.2 Average-Case SoI for rnKt

We first show a version of semi-symmetry of information for rnK^t, which is analogous to Theorem 55.

Theorem 57. *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, Hir22b] 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 31 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,

$$\mathsf{pK}^t(x,y) \ge \mathsf{rK}^{\mathsf{poly}(t),\|\mathsf{MINKT}}_{1/\mathsf{poly}(n)}(x) + \mathsf{rK}^{\mathsf{poly}(t),\|\mathsf{MINKT}}_{1/\mathsf{poly}(n)}(y \mid x) - O(\log^3 n) - O(\log t). \tag{5}$$

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

$$\begin{aligned}\mathsf{rK}^{\mathsf{poly}(t),\|\mathsf{MINKT}}_{1/\mathsf{poly}(n)}(x) &\ge \mathsf{rnK}^{\mathsf{poly}(t)}_{1/\mathsf{poly}(n\cdot t)}(x) - O(\log t) && \text{(by Lemma 50)} \\ &\ge \mathsf{rnK}^{\mathsf{poly}(t)}(x) - O(\log t) && \text{(by Lemma 47)}\end{aligned}$$

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 (5) yields the desired semi-symmetry of information for rnK^t. □

The following corresponds to Theorem 1 for the case of rnKt.

Theorem 58. *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 56, with Theorem 55 replaced by Theorem 57. We omit the details. □

6 Complexity Separations from SoI: Proof of Theorem 2

6.1 Consequence of Worst-Case SoI for pnKt

Theorem 59, stated and proved in Section 6.1, shows that if worst-case symmetry of information holds for pnKt, then coNP $\not\subseteq$ AM. Theorem 65 establishes an analogous result for rnKt and is proved in Section 6.2.

Theorem 59. *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 60. *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 $\mathsf{pnKt}_{2/3}(x \mid y) \leq m/2$, 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 $y \in \{0,1\}^{\leq 2^m \cdot m}$,

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 $|\Pi| + \lceil \log t \rceil \leq m/2$, 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 $\mathsf{pnKt}_{2/3}(x \mid y) \leq m/2$, 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 61. *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 $\ell \leq 2^m$, there exist strings $x_1, \ldots, x_\ell \in \{0,1\}^m$ satisfying*
$$\sum_{i \in [\ell]} \mathsf{pnKt}(x_i \mid x_1, \ldots, x_{i-1}) \geq \ell \cdot \frac{m}{2}$$
and
$$\mathsf{rnKt}(x_1, \ldots, x_\ell) \leq 2^{m^{1/d}}.$$

Proof. Let B be the (probabilistic) algorithm from Lemma 60. We first observe the following.

Claim 62. *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 $y \in \{0,1\}^{\leq 2^{m \cdot m}}$, 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 62. By Lemma 60, for every $x \in \{0,1\}^m$ and $y \in \{0,1\}^{\leq 2^{m \cdot m}}$, 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 $\mathsf{poly}(t(s)) \leq 2^{m^c}$ for a constant c chosen large enough.

◇

Let $c > 0$ be the constant, and let $L \in \mathsf{AM} \cap \mathsf{coAM}$ be the language from Claim 62. Consider Lemma 117 with $\varepsilon := 1/(4cd)$, and let $\mathcal{S} \subseteq \mathbb{N}$ be the infinite set of lengths s at which the multiset T_s is pseudorandom against L-oracle circuits of size s. For $s \in \mathcal{S}$, let

$$m(s) \;:=\; \max\{\, m \in \mathbb{N} \;:\; 2^{m^c} \leq s \,\},$$

so that $2^{m(s)^c} \leq s < 2^{(m(s)+1)^c}$. We say that an integer m is *good* if $m = m(s)$ for some $s \in \mathcal{S}$. 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 $y \in \{0,1\}^{\leq 2^{m \cdot m}}$, and proceeds as follows:

> For each $x \in \{0,1\}^m$, compute
>
> $$\mu_x := \Pr_{z \sim T_s}\left[C_{(x,y)}(z) = 1\right],$$
>
> where $C_{(x,y)}$ is the L-oracle circuit obtained using Claim 62. Output the first x such that $\mu_x > \frac{1}{3} + \frac{1}{10}$.

Claim 63. *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 63. 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 $\mathsf{pnKt}_{1/3}(x \mid y) > m/2$. 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 117, 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:

$$\begin{aligned} \mathsf{rnKt}^L(x_1, \ldots, x_\ell) &\leq O\left(2^{m^{1/3d}}\right) + \log\left(\ell \cdot 2^{\mathsf{poly}(m)}\right) \\ &\leq 2^{m^{1/(2d)}}. \end{aligned} \tag{6}$$

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

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

$$\begin{aligned} \mathsf{rnKt}(x_1, \ldots, x_\ell) &\leq O\big(\mathsf{rnKt}^L(x_1, \ldots, x_\ell)\big) && \text{(by Corollary 52)} \\ &\leq O\left(2^{m^{1/(2d)}}\right) && \text{(by Equation (6))} \\ &\leq 2^{m^{1/d}}. \end{aligned}$$

This completes the proof. □

We now proceed to prove Theorem 59.

Proof of Theorem 59. 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 64. *The following holds for infinitely many values of m. There exist $v \in \{0,1\}^m$ and $u \in \{0,1\}^n$, where $m \leq n \leq 2^{m^{1/(2c)}}$, such that*

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

Proof of Claim 64. 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 61 with $d := 3c$. Let m be a sufficiently large "good" integer such that the statement of Lemma 61 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:

$$\begin{aligned}
\mathsf{pnKt}(x_1, \ldots, x_\ell) &\ge \sum_{i \in [\ell]} \mathsf{pnKt}(x_i \mid x_1, \ldots, x_{i-1}) - \ell \cdot \frac{m}{4} \\
&\ge \ell \cdot \frac{m}{2} - \ell \cdot \frac{m}{4} && \text{(by Lemma 61)} \\
&= 2^{m^{1/d}} \cdot \frac{m}{4}.
\end{aligned}$$

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

$$m \le n \le 2^{m^{1/(2c)}}, \tag{7}$$

and

$$\mathsf{pnKt}(u,v) < \mathsf{pnKt}(u) + \mathsf{pnKt}(v \mid u) - \frac{m}{4}. \tag{8}$$

Define

$$x := u \quad \text{and} \quad y := v0^{n-m}. \tag{9}$$

Then we have:

$$\begin{aligned}
\mathsf{pnKt}(x,y) &\le \mathsf{pnKt}(u,v) + O(\log n) && \text{(by Equation (9))} \\
&\le \mathsf{pnKt}(u) + \mathsf{pnKt}(v \mid u) - \frac{m}{4} + O(\log n) && \text{(by Equation (8))} \\
&\le \mathsf{pnKt}(x) + \mathsf{pnKt}(y \mid x) + O(\log n) - \frac{m}{4} + O(\log n) && \text{(by Equation (9))} \\
&\le \mathsf{pnKt}(x) + \mathsf{pnKt}(y \mid x) + O(\log n) - \frac{\log^{2c} n}{4} && \text{(by Equation (7))} \\
&< \mathsf{pnKt}(x) + \mathsf{pnKt}(y \mid x) - \log^c n,
\end{aligned}$$

as desired. □

6.2 Consequence of Worst-Case SoI for rnKt

The following corresponds to Theorem 2 for the case of rnKt.

Theorem 65. *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 59, which employs Lemma 61. It suffices to note that in Lemma 61, we can replace pnKt with rnKt, since for every $x, y \in \{0,1\}^n$, it holds that $\mathsf{pnKt}(x \mid y) \leq \mathsf{rnKt}(x \mid y)$. □

7 Relativization Barrier for SoI: Proof of Theorem 3

We will prove Theorem 3, restated here for convenience.

Theorem 66. *There exist an oracle $\mathcal{O}$ and a constant $\varepsilon > 0$ such that, for every complexity measure $\mu \in \{\mathsf{rKt}, \mathsf{pKt}, \mathsf{nKt}, \mathsf{rnKt}, \mathsf{pnKt}\}$, the following holds. For all sufficiently large $n \in \mathbb{N}$, there exist strings $x, y \in \{0,1\}^n$ such that*

$$\mu^{\mathcal{O}}(x, y) < \mu^{\mathcal{O}}(x) + \mu^{\mathcal{O}}(y \mid x) - \varepsilon \cdot n.$$

The proof is basically a relativizing version of the argument used in Theorem 59, 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 67 ([BGS75])**.** *There exists an oracle $\mathcal{O}$ such that $\mathsf{PH}^{\mathcal{O}} \subseteq \mathsf{P}^{\mathcal{O}}$.*

Lemma 68. *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 69. *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 67. Let $M_0^{(-)}$ be the machine and $\mathcal{X} \in \Sigma_3^{\mathsf{P},\mathcal{O}}$ the oracle of Lemma 68 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 68, 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 60, but pertains to $\mathsf{pKt}^{\mathcal{O}}$ complexity for arbitrary oracles $\mathcal{O}$ rather than pnKt complexity.

Lemma 70. *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 $\mathsf{pKt}^{\mathcal{O}}_{2/3}(x \mid y) \leq m/2$, then $B^{\mathcal{O}}(x,y)$ rejects with probability at least $2/3$, and*
- *if $\mathsf{pKt}^{\mathcal{O}}_{1/3}(x \mid y) > m/2$, 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 71. *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 61. Let $B^{(-)}$ be the randomized oracle algorithm of Lemma 70. 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 69. Let $c, d \in \mathbb{N}$ be as in Theorem 28 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 28. 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 $C^{\mathsf{NP}^{\mathcal{O}}}_{(x,y)}$.

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

$$\rho_x = \mathop{\mathbf{Pr}}_{\sigma \sim \{0,1\}^{bd \cdot n}}[C^{\mathsf{NP}^{\mathcal{O}}}_{(x,y)}(G_z(\sigma)) = 1].$$

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 70, 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 54, 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 72. *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 3 as stated at the beginning of this section. The argument is quite similar to that of Theorem 59. We include a proof for completeness.

Proof of Theorem 3. Fix $\mu \in \{\mathsf{rKt}, \mathsf{pKt}, \mathsf{nKt}, \mathsf{rnKt}, \mathsf{pnKt}\}$. We start with the following claim.

Claim 73. *Let $\mathcal{O}$ be the oracle of Corollary 72. 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 73. 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

$$\mu^{\mathcal{O}}(u,v) \geq \mu^{\mathcal{O}}(u) + \mu^{\mathcal{O}}(v \mid u) - \frac{m}{c}. \tag{10}$$

Let d be the constant from Corollary 72, let $c = 2d$, and let m be as in Equation (10). Let $\ell := c^2$, and let $x_1, ..., x_\ell \in \{0,1\}^m$ be the strings guaranteed by Corollary 72. 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

$$\mu^{\mathcal{O}}(x_1, ..., x_\ell) < d \cdot m. \tag{11}$$

Consider $u = (x_1, ..., x_{\ell-1}) \in \{0,1\}^{(\ell-1)\cdot m}$ and $v = x_\ell \in \{0,1\}^m$. By Equation (10), 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 (11). □

Now, let the constant c be as in Claim 73. For any sufficiently large $n \in \mathbb{N}$, let $m = \lfloor n/c^4 \rfloor$. By Claim 73, 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

$$\begin{aligned}
\mu^{\mathcal{O}}(x,y) &\leq \mu^{\mathcal{O}}(u,v) + O(\log n) \\
&< \mu^{\mathcal{O}}(u) + \mu^{\mathcal{O}}(v \mid u) - \frac{m}{c} + O(\log n) \\
&\leq \mu^{\mathcal{O}}(x) + \mu^{\mathcal{O}}(y \mid x) - \frac{m}{c} + O(\log n) \\
&\leq \mu^{\mathcal{O}}(x) + \mu^{\mathcal{O}}(y \mid x) - \varepsilon \cdot n. && \text{(choosing } \varepsilon := 1/c^6\text{)}
\end{aligned}$$

This completes the proof of the theorem. □

8 Complexity Collapses from SoI: Proof of Theorem 4

8.1 Uniform Collapses

In this subsection, we prove the first item of Theorem 4. We present the proof for the case where the SoI error is $O(\log n)$; it can be easily adapted to the case of $\mathsf{poly}(\log n)$ error.

Theorem 74 (SoI for $\mathsf{rnKt} \Rightarrow \mathsf{FewEXP} \subseteq \mathsf{BPP}^{\mathsf{NP}}$)**.** *If either* rnKt *or* nKt *satisfies* symmetry of information *with logarithmic error, then* $\mathsf{FewEXP} \subseteq \mathsf{BPP}^{\mathsf{NP}}$.

To prove Theorem 74, the key ingredient is the following lemma, which states that if $\mathsf{FewEXP} \not\subseteq \mathsf{BPP}^{O}$ for some $O \in \mathsf{PSPACE}$, then there exists a pseudorandom generator with certain efficiency guarantees that fools randomized algorithms with oracle access to O.

Lemma 75 (Hardness-to-PRG)**.** *Let* $O \in \mathsf{PSPACE}$*. If* $\mathsf{FewEXP} \not\subseteq \mathsf{BPP}^{O}$*, then for every constant* $\eta > 0$ *and every constant* $b \in \mathbb{N}$*, there exists a sequence of functions* $G = \{G_n\}_{n \in \mathbb{N}}$ *with*

$$G_n : \{0,1\}^{n^{\eta/2}} \to \{0,1\}^n$$

such that:

1. *Each* G_n *is computable in single-valued nondeterministic time* $2^{O(n^{\eta/2})}$ *with* $O(n^{\eta/2})$ *bits of advice.*
2. *For every distribution family* $\{C_n\}$ *of* O*-oracle Boolean circuits samplable in time* n^b*, there are infinitely many* n *such that, with probability at least* $1 - n^{-b}$ *over* $C \sim C_n$*,* G_n n^{-b}*-fools* C*.*

Proof. Fix $O \in \mathsf{PSPACE}$, and assume $\mathsf{FewEXP} \not\subseteq \mathsf{BPP}^{O}$. We consider two cases.

Case 1: $\mathsf{EXP} \not\subseteq \mathsf{BPP}^{O}$**.** The desired PRG follows directly from Corollary 30.

Case 2: $\mathsf{EXP} \subseteq \mathsf{BPP}^O$. Under the assumption that $\mathsf{FewEXP} \not\subseteq \mathsf{BPP}^O$, it follows that $\mathsf{FewEXP} \neq \mathsf{EXP}$. We show the following claim.

Claim 76. *If $\mathsf{FewEXP} \neq \mathsf{EXP}$, then there exist a verifier V and constants $\alpha, k > 0$ such that for infinitely many ℓ, there exist $x_\ell \in \{0,1\}^\ell$ and a set $W_\ell \subseteq \{0,1\}^{2^{\ell^\alpha}}$ satisfying:*

- *$V(x_\ell, w) = 1$ for every $w \in W_\ell$, where V runs in time $2^{O(\ell^\alpha)}$;*
- *$1 \leq |W_\ell| \leq 2^{\ell^k}$;*
- *for every $w \in W_\ell$, when w is interpreted as the truth table of a Boolean function, it cannot be computed by any O-oracle circuit of size $\mathsf{poly}(\ell)$.*

Proof of Claim 76. Let $L \in \mathsf{FewEXP} \setminus \mathsf{EXP}$, and let V be a verifier for L. For each $x \in \{0,1\}^\ell$, let W_x denote the set of witnesses for x with respect to V (possibly empty). Without loss of generality, assume each witness has length exactly 2^{ℓ^α} for some constant $\alpha > 0$. Also, by the definition of FewEXP, there exists a constant $k > 0$ such that $|W_x| \leq 2^{\ell^k}$.

Suppose, for the sake of contradiction, that for all but finitely many $x \in L$, some witness $w \in W_x$, when viewed as the truth table of a Boolean function on ℓ^α input bits, has polynomial-size O-oracle circuit complexity.

We claim that this assumption implies $L \in \mathsf{EXP}$. On input $x \in \{0,1\}^\ell$, enumerate all O-oracle circuits of size at most $\mathrm{poly}(\ell^\alpha) = \mathsf{poly}(\ell)$. For each such circuit, evaluate it on all inputs to obtain a truth table $y \in \{0,1\}^{2^{\ell^\alpha}}$, and check whether $V(x, y) = 1$. Accept if and only if such a y exists.

Since $O \in \mathsf{PSPACE}$, each oracle query can be simulated in time $2^{\mathrm{poly}(\ell)}$, and the total number of circuits of size $\mathrm{poly}(\ell)$ is at most $2^{\mathrm{poly}(\ell)}$. Therefore, evaluating each circuit on all inputs and verifying $V(x, y) = 1$ can be done in time $2^{\mathrm{poly}(\ell)}$, giving an exponential-time algorithm for L.

This contradicts the assumption that $L \notin \mathsf{EXP}$. Hence, for infinitely many yes-instances x, every witness in W_x (which is nonempty) must have super-polynomial O-oracle circuit complexity. ◇

Next, we show that Claim 76 allows us to efficiently obtain, in nondeterministic time with small advice, hard truth tables that have super-polynomial-size O-oracle circuit complexity.

Claim 77. *If $\mathsf{FewEXP} \neq \mathsf{EXP}$, there exist a constant $\beta > 0$ and a sequence $\{y_m \in \{0,1\}^{2^m}\}_m$ such that the following holds for infinitely many m:*

- *y_m, viewed as a truth table, is not computable by any O-oracle circuit of size $\mathsf{poly}(m)$.*
- *y_m can be computed in single-valued nondeterministic time 2^{m^β} with m^β bits of advice.*

Proof of Claim 77. Let V, α, and k be as in Claim 76. By that claim, for infinitely many lengths ℓ, there exist an input $x_\ell \in \{0,1\}^\ell$ and a (nonempty) set of witnesses $W_\ell \subseteq \{0,1\}^{2^{\ell^\alpha}}$ that satisfies the conditions specified in Claim 76. Fix such a good ℓ. We describe how to efficiently obtain one of these "hard" witnesses in single-valued nondeterministic time with small advice.

Take the advice string to be $(x_\ell, |W_\ell|)$. We can now perform the following nondeterministic procedure to obtain the lexicographically first witness in W_ℓ:

1. Nondeterministically guess $|W_\ell|$ distinct strings $y_1, \ldots, y_{|W_\ell|} \in \{0,1\}^{2^{\ell^\alpha}}$.
2. Verify for each y_i that $V(x_\ell, y_i) = 1$.

3. Output the lexicographically first y_i that passes verification.

This procedure is single-valued because, on any nondeterministic branch, the output is uniquely determined as the lex-first valid witness. The running time is at most $O(|W_\ell| \cdot 2^{\ell^\alpha}) = 2^{\mathrm{poly}(\ell)}$.

Let $m = \ell^\alpha$. Then:

- The output of the above procedure is a witness in W_ℓ and has length 2^m.
- The advice length is $O(\ell + \ell^k)$, which is at most m^β for some constant $\beta > 0$.
- The nondeterministic running time is $2^{\mathrm{poly}(\ell)} = 2^{m^\beta}$ for some $\beta > 0$.

Finally, by Claim 76, this lex-first witness is hard against any $\mathsf{poly}(m)$ O-oracle circuits. ◇

Now, we can plug the hard truth tables obtained from Claim 77 into the hardness-to-randomness framework to obtain a PRG.

Fix constants $0 < \eta < 1$ and $b \in \mathbb{N}$. Let $\varepsilon := \eta/2$, and let $\delta \le \varepsilon$ and constant $c \in \mathbb{N}$ be as guaranteed by Theorem 27.

Let $\{y_m \in \{0,1\}^{2^m}\}_m$ be the sequence from Claim 77. For infinitely many m, the function y_m, viewed as a Boolean function on m variables, does not admit O-oracle circuits of size $\mathsf{poly}(m)$, and is computable in single-valued nondeterministic time 2^{m^β} with m^β bits of advice, where $\beta > 0$ is a constant.

Set

$$n := m^{\beta/\delta}.$$

Note that we have $m \le n^\delta$. By padding, we can extend y_m to a function

$$z_{n^\delta} : \{0,1\}^{n^\delta} \to \{0,1\}$$

by ignoring the first $n^\delta - m$ input bits. Note that z_{n^δ} inherits hardness: if it had an O-oracle circuit of size $n^{c\delta}$, then by fixing the extra inputs we would obtain an O-oracle circuit of size $\mathsf{poly}(m)$ computing y_m, contradicting the hardness of y_m.

Now define the generator

$$G_n(s) := F(z_{n^\delta}, s),$$

where $s \in \{0,1\}^{n^\varepsilon}$. By the guarantee of Theorem 27 and the hardness of z_{n^δ}, it follows that for infinitely many n, the generator G_n n^{-b}-fools all O-oracle circuits of size at most n^b.

It remains to verify efficiency. By Claim 77, the truth table y_m (and hence z_{n^δ}) can be computed in single-valued nondeterministic time $2^{m^\beta} = 2^{O(n^\varepsilon)}$ and with $m^\beta = O(n^\varepsilon)$ bits of advice. Since F is computable in polynomial time, it follows that each G_n can be computed in single-valued nondeterministic time $2^{O(n^\varepsilon)}$ with $O(n^\varepsilon)$ bits of advice. □

Next, we use our PRG in Lemma 75 to show that if $\mathsf{FewEXP} \not\subseteq \mathsf{BPP}^O$, where $O \in \mathsf{PSPACE}$, then the following meta-complexity problem must be hard.

Definition 78. Let $s_1, s_2 \colon \mathbb{N} \to \mathbb{N}$. Define the promise problem $\mathsf{Gap}\text{-}(\mathsf{nKt}\text{ v.s. }\mathsf{K})[s_1, s_2]$ as the promise problem $(\mathcal{YES}, \mathcal{NO})$, where

$$\mathcal{YES} \triangleq \{x \mid \mathsf{nKt}(x) \le s_1(|x|)\},$$
$$\mathcal{NO} \triangleq \{x \mid \mathsf{K}(x) > s_2(|x|)\}.$$

Lemma 79. *Let $O \in \mathsf{PSPACE}$. If $\mathsf{FewEXP} \not\subseteq \mathsf{BPP}^O$, then for every $\zeta > 0$, $\mathsf{Gap}\text{-}(\mathsf{nKt}\text{ v.s. }\mathsf{K})[n^\zeta, n-1] \notin \mathsf{BPP}^O$.*

Proof. Fix $O \in \mathsf{PSPACE}$, and assume $\mathsf{FewEXP} \not\subseteq \mathsf{BPP}^O$.

Toward a contradiction, suppose $\mathsf{Gap}\text{-}(\mathsf{nKt}\text{ v.s. }\mathsf{K})[n^\zeta, n-1] \in \mathsf{BPP}^O$ for some $\zeta > 0$. Let D be a probabilistic polynomial-time O-oracle algorithm such that, except with exponentially small probability over its internal randomness, D rejects every string with nKt-complexity $\leq n^\zeta$, and accepts every string with nKt-complexity $> n$. Such a distinguisher can be obtained via standard error reduction followed by a union bound. Equivalently, we may view D as a distribution over O-oracle circuits obtained by fixing its internal randomness.

Let G be the PRG from Lemma 75. Note that since G is computable in single-valued nondeterministic time $2^{O(n^{\eta/2})}$ with $O(n^{\eta/2})$ bits of advice, we have for all sufficiently large n and every $r \in \{0,1\}^{n^{\zeta/2}}$,

$$\mathsf{nKt}(G_n(r)) \leq O(n^{\zeta/2}) \leq n^\zeta.$$

By the correctness of the algorithm D, except with exponentially small probability over the choice of D, it holds that for all but finitely many n,

$$D(G(r)) = 0$$

for every $r \in \{0,1\}^{n^{\eta/2}}$.

On the other hand, with probability at least $1/2$ over $x \sim \{0,1\}^n$, we have $\mathsf{K}(x) > n-1$. It follows that

$$\Pr_{x \sim \{0,1\}^n}[D(x) = 1] \geq \tfrac{1}{2}.$$

Hence, D distinguishes the output distribution of the PRG from the uniform distribution, contradicting Lemma 75. □

We can now complete the proof of Theorem 74

Proof of Theorem 74. We prove the contrapositive. Suppose that $\mathsf{FewEXP} \not\subseteq \mathsf{BPP}^{\mathsf{NP}}$. Then, by Lemma 79, for every $\zeta > 0$,

$$\mathsf{Gap}\text{-}(\mathsf{nKt}\text{ v.s. }\mathsf{K})[n^\zeta, n-1] \notin \mathsf{BPP}^{\mathsf{NP}}.$$

Observe that if either $\mathsf{Gap}\text{-}\mathsf{MrnKtP}[n - O(\log n), n-1]$ or $\mathsf{Gap}\text{-}\mathsf{MnKtP}[n - O(\log n), n-1]$ were in $\mathsf{pr}\text{-}\mathsf{BPP}^{||\mathsf{NP}}$, then $\mathsf{Gap}\text{-}(\mathsf{nKt}\text{ v.s. }\mathsf{K})[n^\zeta, n-1]$ would also be in $\mathsf{BPP}^{\mathsf{NP}}$ for some $\zeta > 0$. Indeed, every YES (resp. NO) instance of the latter problem is also a YES (NO) instance of the former.[18] This contradicts the previous paragraph. Hence, both $\mathsf{Gap}\text{-}\mathsf{MrnKtP}[n - O(\log n), n-1]$ and $\mathsf{Gap}\text{-}\mathsf{MnKtP}[n - O(\log n), n-1]$ are not in $\mathsf{pr}\text{-}\mathsf{BPP}^{||\mathsf{NP}}$. It follows from Lemma 88 and Lemma 94 that SoI does not hold for either rnKt or nKt. □

8.2 Nonuniform Inclusions

In this subsection, we prove the second item of Theorem 4. Again, we present the proof for the case where the SoI error is $O(\log n)$, and it can be easily adapted to the case of $\mathsf{poly}(\log n)$ error.

[18] Recall in particular that, for all strings x, $\mathsf{K}(x) \leq \mathsf{rnKt}(x) \leq \mathsf{nKt}(x)$.

8.2.1 The Case of nKt

Theorem 80 (SoI for nKt $\Rightarrow$ $\mathsf{EXP}^{\|\mathsf{NP}} \subseteq \mathsf{NP/poly} \cap \mathsf{coNP/poly}$)**.** *Assume that* nKt *satisfies symmetry of information with logarithmic error, i.e., there exists a constant $c > 0$ such that for all strings x, y,*

$$\mathsf{nKt}(x, y) \;\geq\; \mathsf{nKt}(x) \;+\; \mathsf{nKt}(y \mid x) \;-\; c \log(|x| + |y|). \tag{12}$$

Then

$$\mathsf{EXP}^{\|\mathsf{NP}} \;\subseteq\; \mathsf{NP/poly} \;\cap\; \mathsf{coNP/poly}.$$

Proof. Assume (12). We combine the consequences already established in Section 8.1 with two ingredients from the meta-complexity/pseudorandomness literature. This is implemented in four steps, as explained next.

Step 1: A large-gap MnKtP **promise problem is in** pr-$\mathsf{P}^{\|\mathsf{NP}}$**.** Since we assume SoI holds for nKt with logarithmic error, by Lemma 94, we get that $\mathsf{Gap\text{-}MnKtP}[n - O(\log n), n - 1] \in \mathsf{pr\text{-}P}^{\|\mathsf{NP}}$. This implies that for some constant $\delta \in (0, 1)$,

$$\mathsf{Gap\text{-}MnKtP}[n^{\delta}, n - 1] \;\in\; \mathsf{pr\text{-}P}^{\|\mathsf{NP}}. \tag{13}$$

Step 2: $\mathsf{E}^{\|\mathsf{NP}}$ **reduces to a large-gap** MnKtP**.** We now invoke a PRG/reconstruction completeness phenomenon for MnKtP.

Lemma 81 (Nonuniform truth-table reduction for nKt)**.** *For any constant $\delta \in (0, 1)$ and language $L \in \mathsf{E}^{\|\mathsf{NP}}$, L has a nonuniform polynomial-time truth-table reduction to* $\mathsf{Gap\text{-}MnKtP}[n^{\delta}, n - 1]$*. Equivalently,*

$$\mathsf{E}^{\|\mathsf{NP}} \;\subseteq\; \mathsf{P}^{\mathsf{Gap\text{-}MnKtP}[n^{\delta}, n-1]}/\mathsf{poly},$$

where all oracle queries to $\mathsf{MnKtP}[n^{\delta}, n - 1]$ *are nonadaptive.*

Proof Sketch. Fix a language $L \in \mathsf{E}^{\|\mathsf{NP}}$. We sketch a standard Nisan–Wigderson (NW) generator and reconstruction argument showing that, for a constant $d \geq 1$ as specified below, the language L has a nonuniform polynomial-time truth-table reduction to the promise problem $\mathsf{MnKtP}[m^{\delta}, m - 1]$ on inputs of length $m = n^d$.

Define an NW/BFNW-style generator from the truth table. Fix a constant $\varepsilon \in (0, 1)$ and let $b \in \mathbb{N}$ be large enough that $b\delta > \varepsilon$, where δ is as in the statement of this lemma. Let $\delta' < \varepsilon$, a, and F be as guaranteed by Theorem 27 applied to ε and b. Set $N \triangleq \lceil n^{1/\delta'} \rceil$ so that $N^{\delta'} \geq n$, and let $f \colon \{0,1\}^{N^{\delta'}} \to \{0,1\}$ be the function L_n, padded to ignore all but its last n input bits. Set $m \triangleq N^b$ (so that $m = n^d$ with $d \triangleq b/\delta'$) and define the generator

$$G_f(z) \;\triangleq\; F(\mathsf{tt}(f), z) \in \{0,1\}^m, \qquad z \in \{0,1\}^{N^{\varepsilon}}.$$

Generator outputs are nKt*-easy.* Fix any seed z. We describe a single-valued nondeterministic program that outputs $G_f(z)$. The program hardwires z together with the number p of positive answers among all of the NP queries made by the $\mathsf{E}^{\|\mathsf{NP}}$ machine for L across all 2^n inputs of length n. It first simulates that machine on every input of length n, collecting the (non-adaptive) list of queries; this takes time $2^{O(n)}$, and there are at most $2^{O(n)}$ queries. It then nondeterministically guesses witnesses for exactly p of these queries, verifies them, and outputs $\perp$ unless the count matches. Observe that this yields a single-valued nondeterministic

construction of $\mathsf{tt}(f)$. The program then computes $F(\mathsf{tt}(f), z)$, taking time $\mathsf{poly}(2^{N^{\delta'}}) = \mathsf{poly}(2^n)$ by Theorem 27.

The total running time is $t \le 2^{O(n)}$, and the description length is $|z| + |p| + O(1) \le N^{\varepsilon} + O(n)$. Hence,

$$\mathsf{nKt}(G_f(z)) \;\le\; N^{\varepsilon} + O(n) + O(\log t) \;=\; N^{\varepsilon} + O(N^{\delta'}) \;=\; O(N^{\varepsilon}).$$

By the choice of b, we get $\mathsf{nKt}(G_f(z)) \le m^{\delta}$ for all sufficiently large n, so $G_f(z)$ is a yes-instance of $\mathsf{MnKtP}[m^{\delta}, m-1]$ for every z.

Uniform strings are nKt*-hard with constant probability.* By a counting argument, a uniformly random $y \sim \mathcal{U}_m$ satisfies $\mathsf{nKt}(y) \ge m-1$ with probability at least $1/2$. Therefore any language D consistent with the promise problem $\mathsf{MnKtP}[m^{\delta}, m-1]$ accepts every $G_f(z)$ and rejects all but a $1/2$ fraction of $\mathcal{U}_m$, and in particular

$$\left| \Pr_{z \sim \mathcal{U}_{N^{\varepsilon}}} [D(G_f(z)) = 1] - \Pr_{w \sim \mathcal{U}_m} [D(w) = 1] \right| \;\ge\; \frac{1}{2} \ge \frac{1}{N^b}.$$

Reconstruction yields a nonuniform reduction. By Theorem 27, there is an oracle circuit $R^{(\cdot)}$ of size $N^a = \mathsf{poly}(n)$, making nonadaptive queries to D, such that $R^D(x) = f(x)$ for every $x \in \{0,1\}^{N^{\delta'}}$. Hardwiring R as nonuniform advice yields a nonuniform polynomial-time truth-table reduction from L to $\mathsf{MnKtP}[m^{\delta}, m-1]$, with queries of length $m = \mathsf{poly}(n)$ depending only on the input and the advice. Note that this holds for any language D consistent with the promise.

□

Combining Lemma 81 with (13) and substituting the $\mathsf{pr}\text{-}\mathsf{P}^{\|\mathsf{NP}}$-procedure for the parallel queries to the oracle yields

$$\mathsf{E}^{\|\mathsf{NP}} \;\subseteq\; \mathsf{P}^{\|\mathsf{NP}}/\mathsf{poly}. \tag{14}$$

Step 3: Downward collapse to $\mathsf{NP}/\mathsf{poly} \cap \mathsf{coNP}/\mathsf{poly}$.

Applying Corollary 39 to (14) gives

$$\mathsf{E}^{\|\mathsf{NP}} \subseteq \mathsf{NP}/\mathsf{poly} \cap \mathsf{coNP}/\mathsf{poly}. \tag{15}$$

Step 4: Standard padding from $\mathsf{E}^{\|\mathsf{NP}}$ to $\mathsf{EXP}^{\|\mathsf{NP}}$. Let $L \in \mathsf{EXP}^{\|\mathsf{NP}}$. Then there exists $k \ge 1$ such that $L \in \mathsf{DTIME}(2^{n^k})^{\|\mathsf{NP}}$. Define the padded language L' by

$$L'(x0^{|x|^k - |x|}) \;\triangleq\; L(x).$$

Then $L' \in \mathsf{E}^{\|\mathsf{NP}}$, and hence by (15) we have $L' \in \mathsf{NP}/\mathsf{poly} \cap \mathsf{coNP}/\mathsf{poly}$. Using the advice for length $|x|^k$ and running the corresponding verifier on the padded input $y = x0^{|x|^k - |x|}$ yields $\mathsf{NP}/\mathsf{poly}$ and $\mathsf{coNP}/\mathsf{poly}$ verifiers for L itself (with advice size still polynomial in $|x|$). Therefore,

$$\mathsf{EXP}^{\|\mathsf{NP}} \subseteq \mathsf{NP}/\mathsf{poly} \cap \mathsf{coNP}/\mathsf{poly},$$

as claimed. □

8.2.2 The Case of rnKt

Theorem 82 (SoI for rnKt $\Rightarrow \mathsf{BPEXP}^{\|\mathsf{NP}} \subseteq \mathsf{NP/poly} \cap \mathsf{coNP/poly}$)**.** *Assume that* rnKt *satisfies symmetry of information with logarithmic error, i.e., there exists a constant $c > 0$ such that for all strings x, y,*

$$\mathsf{rnKt}(x, y) \;\geq\; \mathsf{rnKt}(x) \;+\; \mathsf{rnKt}(y \mid x) \;-\; c \log(|x| + |y|). \tag{16}$$

Then

$$\mathsf{BPEXP}^{\|\mathsf{NP}} \;\subseteq\; \mathsf{NP/poly} \;\cap\; \mathsf{coNP/poly}.$$

The proof of Theorem 82 is similar to that of Theorem 80.

We will make use of the following inclusion, which follows from standard techniques. We include a proof for completeness.

Lemma 83. $\mathsf{BPE}^{\|\mathsf{NP}} \subseteq \mathsf{AME}/O(n)$.

Proof. Consider $L \in \mathsf{BPE}^{\|\mathsf{NP}}$, witnessed by a randomized machine $M^{\|\mathsf{SAT}}$ running in time $t = 2^{O(n)}$ on inputs of length n. Without loss of generality, assume $M^{\|\mathsf{SAT}}$ has failure probability $o(1)$. On input $x \in \{0,1\}^n$ with randomness $r \sim \mathcal{U}_t$, M generates queries $q_1(x, r), \ldots, q_k(x, r)$ with $k \leq t$, receives oracle answers, and accepts or rejects after some t-time computation. For each (x, r), the correct oracle answers are fixed values $b_1(x, r), \ldots, b_k(x, r) \in \{0, 1\}$.

For each $r \in \{0,1\}^t$, define

$$S_r := \{(x, j) \in \{0,1\}^n \times [k] \mid b_j(x, r) = 1\}.$$

Let $k' := k \cdot 2^n$ and

$$\mu_n \;:=\; \mathop{\mathbb{E}}_{r \sim \mathcal{U}_t}\big[|S_r|\big] \in [0, k'],$$

the expected number of "yes" queries on the whole truth-table over $r \in \{0,1\}^t$. Setting $\ell := (k')^2 \cdot n$, we take the advice string to be the integer

$$\alpha_n \;:=\; \lfloor \ell \cdot \mu_n \rfloor \;\in\; [0,\; \ell \cdot k'],$$

so that $\ell\mu_n - 1 < \alpha_n \leq \ell\mu_n$. Since $\ell \cdot k' \leq 2^{O(n)}$, the advice α_n can be represented with $O(n)$ bits.

We now describe an Arthur-Merlin protocol with advice α_n to decide L on inputs of length n.

> On input x^*, Arthur samples $\ell = 2^{O(n)}$ independent strings $r_1, \ldots, r_\ell \sim \mathcal{U}_t$ and sends them to Merlin.
>
> For each $i \in [\ell]$, Merlin sends:
>
> - a set $\hat{S}_i \subseteq \{0,1\}^n \times [k]$, claimed to be the set of "yes" queries of $M^{\|\mathsf{SAT}}$ on randomness r_i (i.e., S_{r_i} as defined above);
> - for each $(x, j) \in \hat{S}_i$, a witness $w_{x,i,j} \in \{0,1\}^{\leq t}$ for the SAT instance $q_j(x, r_i)$.
>
> Arthur verifies Merlin's message in the following steps:
>
> - for each $i \in [\ell]$, and $(x, j) \in \hat{S}_i$, verify that $w_{x,i,j}$ is a valid witness for $q_j(x, r_i)$, and reject if any is not;

- compute the total number of "yes" queries over all r_i,

$$X := \sum_{i=1}^{\ell} |\hat{S}_i|,$$

and reject unless

$$\alpha_n - X \leq \ell/6;$$

- accept iff

$$\frac{1}{\ell} \cdot \left|\left\{ i \in [\ell] \;\middle|\; M^{\hat{S}_{x^*,i}}(x^*, r_i) \text{ accepts } \right\}\right| \;\geq\; \frac{1}{2},$$

where $\hat{S}_{x^*,i} := \{(x, j) \in \hat{S}_i \mid x = x^*\}$ and $M^{\hat{S}_{x^*,i}}$ denotes the computation of $M^{(-)}$ with oracle queries answered according to $\hat{S}_{x^*,i}$.

Note that the size of Merlin's message is at most $2^{O(n)} \cdot \ell \cdot k \cdot t = 2^{O(n)}$. Also note that Arthur runs in time at most $2^{O(n)}$.

Now we argue for the correctness of the protocol.

Completeness. Suppose $x^* \in L$ and Merlin is honest, i.e., $\hat{S}_i = S_{r_i}$ and all witnesses sent by Merlin are correct. Clearly, the first step of Arthur's verification passes. Further, since $\alpha_n \leq \ell\mu_n$, we have $\alpha_n - X \leq \ell\mu_n - X$, and hence by Hoeffding's Inequality,

$$\begin{aligned}
\Pr_{r_1,\ldots,r_\ell \sim \mathcal{U}_t}[\, \alpha_n - X > \ell/6 \,] &\leq \Pr_{r_1,\ldots,r_\ell \sim \mathcal{U}_t}[\, \ell\mu_n - X > \ell/6 \,] \\
&\leq \exp\left(-\frac{2 \cdot (\ell/6)^2}{\ell \cdot (k')^2}\right) \\
&\leq \exp(-n/18).
\end{aligned}$$

So, with probability $1 - o(1)$, the second step of Arthur's verification passes.

Since $x^* \in L$, the honest computation accepts x^* on all but a $o(1)$-fraction of random r. Then, by a Chernoff bound, letting A_i be an indicator random variable for the event that $M^{\|\mathsf{SAT}}(x^*, r_i)$ accepts, we have

$$\Pr\left[\frac{1}{\ell}\sum_{i=1}^{\ell} A_i \leq \frac{9}{10}\right] \;\leq\; \exp(-\Omega(n)). \tag{17}$$

In the case that $\frac{1}{\ell}\sum_{i=1}^{\ell} A_i > \frac{9}{10}$, with all queries answered honestly, the third verification step passes. Overall, the protocol accepts with probability $1 - o(1)$.

Soundness. Suppose $x^* \notin L$. We show that for any Merlin strategy, Arthur rejects with probability $1 - o(1)$.

Since Merlin must provide a valid witness for every $(x, j) \in \hat{S}_i$, the first verification step enforces $\hat{S}_i \subseteq S_{r_i}$ for all i, so Merlin can only answer such that

$$X \leq \sum_{i=1}^{\ell} |S_{r_i}| =: Y.$$

By Hoeffding's inequality, with probability $1 - \exp(-\Omega(n))$ we have $Y \leq \ell\mu_n + \ell/6 < \alpha_n + 1 + \ell/6$. In this event, if Merlin omits more than $\ell/3 + 1$ yes-queries (i.e. $Y - X > \ell/3 + 1$), then $\alpha_n - X > \ell/6$, and Arthur rejects in the second step.

Therefore, supposing Merlin passes the second step, he must omit at most $\ell/3 + 1$ total "yes" queries, meaning in particular that $\hat{S}_{x^*,i} \neq S_{x^*,r_i}$ for at most $\ell/3+1$ indices i. On the remaining indices, $M^{\hat{S}_{x^*,i}}(x^*, r_i)$ agrees with the honest computation of $M^{\|\mathsf{SAT}}(x^*, r_i)$. With $x^* \notin L$, arguing as in Equation (17), we have

$$\Pr\left[\frac{1}{\ell}\sum_{i=1}^{\ell} A_i \;\geq\; \frac{1}{10}\right] \;\leq\; \exp(-\Omega(n)).$$

Hence, with probability $1 - \exp(-\Omega(n))$, the number of accepting indices in the third step is at most $\ell/3 + 1 + \ell/10 < \ell/2$, in which case Arthur rejects.

This concludes the proof.

□

We will need the following randomized analogue of Lemma 81.

Lemma 84 (Nonuniform truth-table reduction for rnKt)**.** *For every constant $\delta \in (0,1)$ and language $L \in \mathsf{BPE}^{\|\mathsf{NP}}$, L has a nonuniform polynomial-time truth-table reduction to* $\mathsf{Gap\text{-}MrnKtP}[n^\delta, n-1]$*. Equivalently,*

$$\mathsf{BPE}^{\|\mathsf{NP}} \subseteq \mathsf{P}^{\mathsf{Gap\text{-}MrnKtP}[n^\delta, n-1]}/\mathsf{poly},$$

where all oracle queries to $\mathsf{Gap\text{-}MrnKtP}[n^\delta, n-1]$ *are nonadaptive. If in addition we have*

$$\mathsf{Gap\text{-}MrnKtP}[n^\delta, n-1] \in \mathsf{pr\text{-}BPTIME}[T]^{\|\mathsf{NP}},$$

then

$$\mathsf{BPE}^{\|\mathsf{NP}} \subseteq \mathsf{DTIME}[\mathsf{poly}(T(\mathsf{poly}))]^{\|\mathsf{NP}}/\mathsf{poly}(T(\mathsf{poly})).$$

Proof of Lemma 84. We first argue that $\mathsf{AME}/O(n) \subseteq \mathsf{P}^{\mathsf{Gap\text{-}MrnKtP}[n^\delta, n-1]}/\mathsf{poly}$, following along the same lines as the proof of Lemma 81. Fix a language $L \in \mathsf{AME}/O(n)$.

Define a generator from the truth table. Fix a constant $\varepsilon \in (0,1)$ and let $b \in \mathbb{N}$ be large enough that $b\delta > \varepsilon$, where δ is the constant of the statement. Let $\delta' < \varepsilon$, a, and F be as guaranteed by Theorem 27 applied to ε and b. Set $N \triangleq \lceil n^{1/\delta'} \rceil$, so that $N^{\delta'} \geq n$, and let $f\colon \{0,1\}^{N^{\delta'}} \to \{0,1\}$ be the function L_n, padded to ignore all but its last n input bits. Set $m \triangleq N^b$, so that $m = n^d$ with $d \triangleq b/\delta'$, and define the generator

$$G_f(z) \;\triangleq\; F(\mathsf{tt}(f), z) \in \{0,1\}^m, \qquad z \in \{0,1\}^{N^\varepsilon}.$$

Generator outputs are rnKt*-easy.* Fix any seed z. Without loss of generality, we may assume that there is a non-uniform verifier V_L for L with failure probability at most 2^{-2n}. By a union bound, this implies that with probability greater than $2/3$ over $r \sim \mathcal{U}_{2^{O(n)}}$, for all $x \in \{0,1\}^n$,

$$x \in L \iff \exists w \in \{0,1\}^{2^{O(n)}},\; V_L(x, r, w) = 1.$$

We describe a randomized nondeterministic program that outputs $G_f(z)$. The program hardwires z (of length N^ε), the non-uniformity of V_L, and the number $k \in [2^n]$ of inputs $x \in \{0,1\}^n$ with $L(x) = 1$ (both of length $O(n)$). Given randomness r as above, it nondeterministically guesses a set $Y \subseteq \{0,1\}^n$

of size k together with witnesses $w_x \in \{0,1\}^{2^{O(n)}}$ for each $x \in Y$, and verifies that $V_L(x, r, w_x) = 1$ for every $x \in Y$; if any verification fails, it outputs $\perp$. Observe that, since $|Y| = k = |L_n|$, every nondeterministic path not resulting in $\perp$ correctly classifies all yes- and no-instances of L_n; this yields a single-valued nondeterministic construction of $\mathsf{tt}(f)$. The program then computes $F(\mathsf{tt}(f), z)$, taking time $\mathsf{poly}(2^{N^{\delta'}}) = \mathsf{poly}(2^n)$ by Theorem 27.

The total running time is $t \leq 2^{O(n)}$, and the description length is $|z| + O(n) + O(1) \leq N^{\varepsilon} + O(n)$. Hence,

$$\mathsf{rnKt}(G_f(z)) \;\leq\; N^{\varepsilon} + O(n) + O(\log t) \;=\; N^{\varepsilon} + O(N^{\delta'}) \;=\; O(N^{\varepsilon}).$$

By the choice of b, we get $\mathsf{rnKt}(G_f(z)) \leq m^{\delta}$ for all sufficiently large n, so $G_f(z)$ is a yes-instance of $\mathsf{Gap\text{-}MrnKtP}[m^{\delta}, m-1]$ for every z.

Uniform strings are rnKt*-hard with constant probability.* By a counting argument as in Lemma 23, a uniformly random $y \sim \mathcal{U}_m$ satisfies $\mathsf{rnKt}(y) \geq m - 1$ with probability at least $1/2$. Therefore, any language D consistent with the promise problem $\mathsf{Gap\text{-}MrnKtP}[m^{\delta}, m-1]$ breaks G_f with advantage at least $1/2 \geq N^{-b}$, meeting the hypothesis of Theorem 27.

Reconstruction yields a non-uniform reduction. By Theorem 27, there is an oracle circuit $R^{(\cdot)}$ of size $N^a = \mathsf{poly}(n)$, making nonadaptive queries to D, such that $R^D(x) = f(x)$ for every $x \in \{0,1\}^{N^{\delta'}}$. Hardwiring R as nonuniform advice and restricting f to L_n yields a nonuniform polynomial-time truth-table reduction from L to $\mathsf{Gap\text{-}MrnKtP}[m^{\delta}, m-1]$, with queries of length $m = \mathsf{poly}(n)$ depending only on the input and the advice. In particular, this holds for any language D consistent with $\mathsf{Gap\text{-}MrnKtP}[m^{\delta}, m-1]$; applying Lemma 83, we obtain

$$\mathsf{BPE}^{\|\mathsf{NP}} \subseteq \mathsf{AME}/O(n) \subseteq \mathsf{P}^{\|D}/\mathsf{poly}$$

for any such D.

For the "in addition" part of the lemma, suppose $\mathsf{Gap\text{-}MrnKtP}[m^{\delta}, m-1] \in \mathsf{pr\text{-}BPTIME}[T]^{\|\mathsf{NP}}$, witnessed by a randomised oracle algorithm $A^{\|\mathsf{NP}}$ running in time T. Without loss of generality, we may assume that $A^{\|\mathsf{NP}}$ has failure probability at most 2^{-2m} on inputs of length m. This implies that with probability $1 - 2^{-\Omega(n)}$ over the internal randomness r of $A^{\|\mathsf{NP}}$, $A^{\|\mathsf{NP}}(-; r)$ is consistent with $\mathsf{Gap\text{-}MrnKtP}[m^{\delta}, m-1]$ on every input of length $m = \mathsf{poly}(n)$. Fix some good choice of randomness, $r^* \in \{0,1\}^{\mathsf{poly}(T(m))}$, and apply the reduction above to $D(-) \triangleq A^{\|\mathsf{NP}}(-; r^*)$.

We may then simulate any $\mathsf{TIME}[\mathsf{poly}(n)]^{\|\mathsf{Gap\text{-}MrnKtP}[m^{\delta}, m-1]}/\mathsf{poly}(n)$ procedure by answering all queries via $A^{\|\mathsf{NP}}(-; r^*)$. Since all oracle queries are non-adaptive, we obtain

$$\mathsf{BPE}^{\|\mathsf{NP}} \subseteq \mathsf{DTIME}[\mathsf{poly}(T(\mathsf{poly}))]^{\|\mathsf{NP}}/\mathsf{poly}(T(\mathsf{poly}))$$

as desired.

□

We are now ready to describe the proof of Theorem 82.

Proof Sketch of Theorem 82. The proof closely follows that of Theorem 80.

Assuming SoI holds for rnKt with logarithmic error, by Lemma 88 we obtain that $\mathsf{Gap\text{-}MrnKtP}[n - O(\log n), n-1] \in \mathsf{pr\text{-}BPP}^{\|\mathsf{NP}}$, which further implies that for some constant $\delta > 0$,

$$\mathsf{Gap\text{-}MrnKtP}[n^{\delta}, n-1] \;\in\; \mathsf{pr\text{-}BPP}^{\|\mathsf{NP}}.$$

Next, by Lemma 84, every language in $\mathsf{BPE}^{\|\mathsf{NP}}$ admits a nonuniform polynomial-time truth-table reduction to $\mathsf{Gap\text{-}MrnKtP}[n^\delta, n-1]$. Therefore, we obtain

$$\mathsf{BPE}^{\|\mathsf{NP}} \subseteq \mathsf{P}^{\|\mathsf{NP}}/\mathsf{poly}.$$

Applying Corollary 40, we conclude that

$$\mathsf{BPE}^{\|\mathsf{NP}} \subseteq \mathsf{NP}/\mathsf{poly} \cap \mathsf{coNP}/\mathsf{poly}.$$

Finally, by a standard padding argument, the above yields

$$\mathsf{BPEXP}^{\|\mathsf{NP}} \subseteq \mathsf{NP}/\mathsf{poly} \cap \mathsf{coNP}/\mathsf{poly},$$

as desired. □

9 Refined Meta-Complexity Landscape Under SoI: Proof of Theorem 5

9.1 Nonuniform Upper Bound

The proof of Theorem 5 Item (*i*) follows from a combination of results, as explained next.

Proof of Theorem 5 Item (*i*). Suppose that SoI with error $e(n) = O(\log n)$ holds for rnKt. Then, by Theorem 6, it follows that

$$\mathsf{Gap\text{-}MrnKtP}[n - O(\log n), n-1] \in \mathsf{pr\text{-}BPP}^{\|\mathsf{NP}}.$$

Let $M(x)$ be a $\mathsf{pr\text{-}BPP}^{\|\mathsf{NP}}$-machine that agrees with $\mathsf{Gap\text{-}MrnKtP}[n - O(\log n), n-1]$. Since M runs in polynomial time, every oracle query made by M is of length $\mathsf{poly}(n)$, and the random string employed by M is also of length $\mathsf{poly}(n)$. Consequently, we can deterministically compute the acceptance probability of M on any given input in exponential time. In particular, this implies the existence of a language $L \in \mathsf{EXP}$ that agrees with the promise problem $\mathsf{Gap\text{-}MrnKtP}[n - O(\log n), n-1]$.

Under the initial SoI assumption, we can apply Theorem 82 to L, which gives the upper bound $L \in \mathsf{NP}/\mathsf{poly} \cap \mathsf{coNP}/\mathsf{poly}$. In other words,

$$\mathsf{Gap\text{-}MrnKtP}[n - O(\log n), n-1] \in \mathsf{NP}/\mathsf{poly} \cap \mathsf{coNP}/\mathsf{poly}.$$

This completes the proof. □

9.2 Uniform Lower Bound

We will use the following result about MKtP by [ABK+06]:

Lemma 85 ([ABK+06])**.** $\mathsf{EXP} \subseteq \mathsf{NP}^{\mathsf{MKtP}}$.

Additionally, we will use the following simple separation.

Lemma 86. $\mathsf{EE} \not\subseteq \mathsf{DTIME}[n^{\mathsf{poly}(\log n)}]^{\|\mathsf{NP}}/n^{\mathsf{poly}(\log n)}$.

Proof. For $n \in \mathbb{N}$, let $B := 2^{\lceil n^{2/3} \rceil}$. For every machine M of description length at most $\log n$ and every advice string $\alpha \in \{0,1\}^{\leq B}$, let $\tau_{M,\alpha} \in \{0,1\}^{2^n}$ be the table whose entry at $x \in \{0,1\}^n$ is the output of $M^{\mathsf{SAT}}(x, \alpha)$ after B steps (taken to be 0 if M has not halted with output 1). Let Λ_n be the list of all such tables. Note that

$$|\Lambda_n| \;\leq\; 2n \cdot 2^{B+1} \;<\; 2^{2^n}.$$

Let τ_n be the lexicographically first string of $\{0,1\}^{2^n}$ not in Λ_n, and set $L_{\mathrm{diag}}(x) := \tau_{|x|}(x)$.

To compute τ_n, first tabulate the answers to all SAT queries of length at most B by exhaustive search, in time $2^{O(B)}$. Using this table, simulate each of the at most $2n \cdot 2^{B+1}$ pairs (M, α) on all 2^n inputs for B steps, and then check the first $|\Lambda_n| + 1$ strings of $\{0,1\}^{2^n}$ against Λ_n. The total time is $2^{O(B)}$, so $L_{\mathrm{diag}} \in \mathsf{DTIME}[2^{O(2^n)}] \subseteq \mathsf{EE}$.

Now let $L \in \mathsf{DTIME}[n^{(\log n)^c}]^{\|O}/n^{(\log n)^c}$ for some constant c and some $O \in \mathsf{NP}$, witnessed by a machine M and advice $(\alpha_n)_{n\in\mathbb{N}}$. Let M' simulate M, replacing each query to O by its image under a fixed polynomial-time reduction to SAT. On inputs of length n, M' runs in time $\mathsf{poly}(n^{(\log n)^c}) \leq B$. Moreover, $|M'| \leq \log n$ and $|\alpha_n| \leq B$. For such n, τ_{M',α_n} is the truth table of L at length n, so it lies in Λ_n and differs from τ_n. Hence L and L_{diag} differ at every sufficiently large input length. □

We are now ready to prove Theorem 5 Item (*ii*).

Proof of Theorem 5 Item (ii). We consider first the case of rnKt. Assume that SoI holds for rnKt with error $e(n) = \mathsf{poly}(\log n)$. Then by Theorem 6, we have

$$\mathsf{Gap\text{-}MrnKtP}[n - \mathsf{poly}(\log n), n-1] \in \mathsf{pr\text{-}BPTIME}[n^{\mathsf{poly}(\log n)}]^{\|\mathsf{NP}}.$$

Then by Lemma 84 and a standard padding argument,

$$\mathsf{BPEXP}^{\|\mathsf{NP}} \subseteq \mathsf{DTIME}[n^{\mathsf{poly}(\log n)}]^{\|\mathsf{NP}}/n^{\mathsf{poly}(\log n)}. \tag{18}$$

Now assume that $\mathsf{MKtP} \in \mathsf{NTIME}[n^{\mathsf{poly}(\log n)}] \cap \mathsf{coNTIME}[n^{\mathsf{poly}(\log n)}]$. Then by Lemma 85, we get

$$\mathsf{EXP} \subseteq \mathsf{NP}^{\mathsf{MKtP}} \subseteq \mathsf{NP}^{\mathsf{NTIME}[n^{\mathsf{poly}(\log n)}] \cap \mathsf{coNTIME}[n^{\mathsf{poly}(\log n)}]} \subseteq \mathsf{NTIME}[n^{\mathsf{poly}(\log n)}],$$

where the final inclusion follows from guessing the witnesses for both the NP and coNP machine for the oracle queries. Now by a padding argument, we get

$$\mathsf{EE} = \mathsf{DTIME}[2^{2^{O(n)}}] \subseteq \mathsf{NTIME}[2^{\mathsf{poly}(n)}] = \mathsf{NEXP}.$$

By Equation (18), this gives

$$\mathsf{EE} \subseteq \mathsf{NEXP} \subseteq \mathsf{BPEXP}^{\|\mathsf{NP}} \subseteq \mathsf{DTIME}[n^{\mathsf{poly}(\log n)}]^{\|\mathsf{NP}}/n^{\mathsf{poly}(\log n)}.$$

However, this is in contradiction to Lemma 86.

Similarly, assume that SoI holds for nKt with error $e(n) = \mathsf{poly}(\log n)$. Then analogously to the previous proof, but applying Lemma 81 instead of Lemma 84, it follows that

$$\mathsf{EXP}^{\|\mathsf{NP}} \subseteq \mathsf{DTIME}[n^{\mathsf{poly}(\log n)}]^{\|\mathsf{NP}}/\mathsf{poly}(n).$$

Since $\mathsf{NEXP} \subseteq \mathsf{EXP}^{\|\mathsf{NP}}$, it is enough to note that this is sufficient to obtain a contradiction, by continuing as in the proof for rnKt described above. □

10 Asymmetry, Hardness, and Explicit Constructions: Proof of Theorem 6

10.1 SoI to Meta-Complexity

In this subsection, we show that if SoI holds for rnKt (resp. nKt), then approximating rnKt-complexity (resp. nKt-complexity) becomes easy with parallel access to an NP oracle.

10.1.1 The Case of rnKt

We first establish the following using ideas from [HMO26].

Lemma 87 (Efficient Simulation Under SoI)**.** *Assume SoI holds for* rnKt *with error* $e(n) \geq \log n$*. Then, there exists a time bound* $t \leq 2^{O(e(n))}$ *such that for every string* $x \in \{0,1\}^n$*, it holds that* $\mathsf{rnK}^t(x) \leq \mathsf{rnKt}(x) + O(e(n))$.

Proof. We adapt the proof of a similar result from [HMO26] stated in the context of rKt complexity.[19]

Let $n = |x|$. Let the n bits of x be $x_1x_2\cdots x_n$. For any integers $1 \leq i \leq j \leq n$, we define $x_{[i,j]} = x_i \cdots x_j$. For each $i \leq j$, we define $s_{[i,j]}$ and $t_{[i,j]}$ to be the program size and running time of an optimal rnKt program for $x_{[i,j]}$ conditioning on $x_{[1,i-1]}$. If there are multiple programs achieving the optimal bound $\mathsf{rnKt}(x_{[i,j]})$, we take one with the smallest running time.

We let $1 \leq i \leq n$ be the smallest integer such that $t_{[1,i]} \geq 2^{C\cdot\log n+1} \cdot 2^{e(n)}$, where $C \geq 1$ is a suitable universal constant defined below. If no such i exists, then the optimal rnKt program for x already runs in time $2^{O(e(n))}$, and the proof is finished.

Note that we can assume without loss of generality that $1 < i < n$. Indeed, if C is large enough, using that $e(n) \geq \log n$ it is easy to see that $i > 1$. The case $i = n$ can be handled by an easy modification of the argument above.

Now, by symmetry of information, we have

$$\begin{aligned}\mathsf{rnKt}(x) &\geq \mathsf{rnKt}(x_{[1,i]}) + \mathsf{rnKt}(x_{[i+1,n]} \mid x_{[1,i]}) - e(n)\\ &= s_{[1,i]} + \log(t_{[1,i]}) + s_{[i+1,n]} + \log(t_{i+1,n}) - e(n). \end{aligned} \tag{19}$$

But on the other hand, since we can print x by simply running the programs in sequence, we get[20]

$$\begin{aligned}\mathsf{rnKt}_{(2/3)^2}(x) &\leq s_{[1,i]} + s_{[i+1,n]} + O(1) + \log(O(t_{[1,i]} + t_{[i+1,n]} + n))\\ &\leq s_{[1,i]} + s_{[i+1,n]} + \log(t_{[1,i]} + t_{[i+1,n]} + n) + C, \end{aligned} \tag{20}$$

for some universal constant $C \geq 1$ that can depend on the computational model. Furthermore, by applying success amplification (Lemma 47) to Equation (20), we get

$$\mathsf{rnKt}(x) \leq s_{[1,i]} + s_{[i+1,n]} + \log(t_{[1,i]} + t_{[i+1,n]} + n) + C \cdot \log n. \tag{21}$$

Combining Equations (19) and (21), we get

$$\log(t_{[1,i]} + t_{[i+1,n]} + n) \geq \log(t_{[1,i]}) + \log(t_{[i+1,n]}) - e(n) - C \cdot \log n.$$

[19] The proof given here admits slightly better parameters, and relies on an alternative argument suggested to us by Hanlin Ren.

[20] Here we use the aforementioned properties of the encoding of programs in the definition of Kolmogorov complexity.

By taking the exponential of both sides, and using that $e(n) \geq \log n$ and $t_{[1,i]} \geq 2^{C \log n + 1} \cdot 2^{e(n)}$, we get

$$t_{[i+1,n]} \leq \frac{1 + \frac{n}{t_{[1,i]}}}{\frac{1}{2^{e(n)+C\log n}} - \frac{1}{t_{[1,i]}}} \leq \frac{2}{\frac{1}{2^{e(n)+C\log n}} - \frac{1}{t_{[1,i]}}} \leq 2^{C\log n + 2} \cdot 2^{e(n)}.$$

Hence by combining the programs for $x_{[1,i-1]}$ and $x_{[i+1,n]}$, and by storing the bit x_i and the position i, we can print x in (nondeterministic) time $2^{O(e(n))}$. Therefore, it follows that

$$\begin{aligned}
\mathsf{rnK}^{2^{O(e(n))}}(x) &\leq s_{[1,i-1]} + s_{[i+1,n]} + O(\log n) \\
&\leq \mathsf{rnKt}(x_{[1,i-1]}) + \mathsf{rnKt}(x_{[i+1,n]} \mid x_{[1,i]}) + O(\log n) \\
&\leq \mathsf{rnKt}(x_{[1,i]}) + \mathsf{rnKt}(x_{[i+1,n]} \mid x_{[1,i]}) + O(\log n) \\
&\leq \mathsf{rnKt}(x) + O(e(n)), && \text{(By SoI for } \mathsf{rnKt})
\end{aligned}$$

as desired. □

As a consequence of the above, we obtain the following lemma, which completes the proof of Item (i) $\implies$ Item (ii) in Theorem 6.

Lemma 88. *Assume SoI holds for* rnKt *with error* $e(n) \geq \log n$*. Then,* $\mathsf{Gap\text{-}MrnKtP}[s_1, s_2]$ *is contained in* $\mathsf{pr\text{-}BPTIME}[2^{O(e(n))}]^{\|\mathsf{NP}}$*, where* $s_1(n) = n - O(e(n))$ *and* $s_2(n) = n - 1$*.*

Proof. For $t \in \mathbb{N}$, we say that a string y is an rnK^t*-description* of a string x if

$$\Pr_{r \sim \{0,1\}^t}\left[\begin{array}{l}\forall w \in \{0,1\}^t, \Pi(y; w; r) \text{ outputs } x \text{ or } \perp \text{ in } t \text{ steps;} \\ \exists w \in \{0,1\}^t, \Pi(y; w; r) \text{ outputs } x \text{ in } t \text{ steps}\end{array}\right] \geq 2/3.$$

(Note that we do not require y to be of minimum length among rnK^t-descriptions here.)

By Lemma 87, for some constant $b \in \mathbb{N}$ and $t' = 2^{b \cdot e(n)}$, for every string $x \in \{0,1\}^n$, it holds that $\mathsf{rnK}^{t'}(x) \leq \mathsf{rnKt}(x) + b \cdot e(n)$. In particular, for every $x \in \{0,1\}^n$, there exists a string m of length at most $\mathsf{rnKt}(x) + b \cdot e(n)$ such that m is an $\mathsf{rnK}^{t'}$-description for x and $\mathsf{rnK}^{t'}(x) = |m|$.

By our assumption of SoI for rnKt,

$$\begin{aligned}
\mathsf{rnKt}(m \mid x) &\leq \mathsf{rnKt}(m, x) - \mathsf{rnKt}(x) + e(n) \\
&\leq |m| + \log t' + O(\log n) - \mathsf{rnKt}(x) + e(n) \\
&\leq \mathsf{rnK}^{t'}(x) - \mathsf{rnKt}(x) + O(e(n)) \\
&\leq c \cdot e(n), && (22)
\end{aligned}$$

for some constant $c \in \mathbb{N}$. This implies that, for $t = 2^{c \cdot e(n)}$,

$$\mathsf{rnK}^t(m \mid x) \leq c \cdot e(n).$$

As a first step, we will define a randomised algorithm B equipped with non-adaptive access to a SAT oracle. Ultimately, this algorithm will be used as a sub-procedure in our algorithm to estimate rnKt. Roughly, $B^{\|\mathsf{SAT}}$ will decide, for a given pair (v, x), whether v is an rnK^t description of a string $m \in \{0,1\}^s$ that is an $\mathsf{rnK}^{t'}$ description of x; if so, it will output s.

For $n, t, t', s, i \in \mathbb{N}$ and strings y, r, r', define the following SAT queries.

- $Q_1(y, r, t, s) \in \mathsf{SAT} \iff \exists w \in \{0,1\}^{\leq t}$ s.t. $U(y; w; r)$ outputs some string $z \in \{0,1\}^s \backslash \{\perp\}$ in time at most t;
- $Q_2(y, r, t) \in \mathsf{SAT} \iff \exists w, w' \in \{0,1\}^{\leq t}$ such that either $U(y; w; r)$ fails to halt within t steps, or $U(y; w; r) = z$ and $U(y; w'; r) = z'$ in time at most t, where $z \neq z'$ and $z, z' \neq \perp$.
- $Q_3(y, r, t, i) \in \mathsf{SAT} \iff \exists w \in \{0,1\}^{\leq t}$ s.t. $U(y; w; r)$ outputs some string $z \neq \perp$ in time at most t, and $z_i = 1$.
- $Q_4(y, r, r', t, t', n) \in \mathsf{SAT} \iff \exists w \in \{0,1\}^{\leq t}$ s.t. $U(y; w; r)$ outputs some string $z \neq \perp$ in time at most t, and $Q_1(z, r', t', n) \in \mathsf{SAT}$.
- $Q_5(y, r, r', t, t') \in \mathsf{SAT} \iff \exists w \in \{0,1\}^{\leq t}$ s.t. $U(y; w; r)$ outputs some string $z \neq \perp$ in time at most t, and $Q_2(z, r', t') \in \mathsf{SAT}$.
- $Q_6(y, r, r', t, t', i) \in \mathsf{SAT} \iff \exists w \in \{0,1\}^{\leq t}$ s.t. $U(y; w; r)$ outputs some string $z \neq \perp$ in time at most t, and $Q_3(z, r', t', i) \in \mathsf{SAT}$.

The algorithm $B^{\|\mathsf{SAT}}$ is as follows.

1. On input (v, x), sample randomness $r_1, ..., r_{n^2} \sim \mathcal{U}_t$ and $r'_1, ..., r'_{n^2} \sim \mathcal{U}_{t'}$, where $n = |x|$, and t and t' are as defined above.
2. For each $j \in [n^2]$, $s \in [2n]$, and $i \in [s]$, query $Q_1((v, x), r_j, t, s)$, $Q_2((v, x), r_j, t)$, and $Q_3((v, x), r_j, t, i)$ to the SAT oracle, and let $z^{(j)}$ be the concatenation of the values of $\mathsf{SAT}(Q_3((v, x), r_j, t, i))$ for $i = 1, ..., s$.
3. For each $i, j, k \in [n^2]$, query $Q_4((v, x), r_j, r'_k, t, t', n)$, $Q_5((v, x), r_j, r'_k, t, t')$, and $Q_6((v, x), r_j, r'_k, t, t', i)$ to the SAT oracle, and let $z^{(j,k)}$ be the concatenation of the values of $\mathsf{SAT}(Q_6((v, x), r_j, r'_k, t, t', i))$ for $i = 1, ..., n$.
4. Let $S \subseteq [n^2]$ be the set of rounds j in which $Q_1((v, x), r_j, t, s)$ was accepted for some choice of s and $Q_2((v, x), r_j, t)$ was rejected. Let m' be the majority value of $(z^{(j)})_{j \in S}$, and let $s' = |m'|$. Let $S' = \{j \in S \mid z^{(j)} = m'\}$. If $|S'|/n^2 \leq (1/2) + (1/10)$, halt and output $\perp$.
5. Fix the lexicographically first $j \in S'$. Let $T \subseteq [n^2]$ be the set of rounds k in which $Q_4((v, x), r_j, r'_k, t, t', n)$ was accepted and $Q_5((v, x), r_j, r'_k, t, t')$ was rejected. Let x' be the majority value of $z^{(j,k)}$ over all $k \in T$. Let $T' = \{k \in T \mid z^{(j,k)} = x'\}$. If $|T'|/n^2 > (1/2) + (1/10)$ and $x' = x$, halt and output s'. Otherwise, output $\perp$.

Observe that $B^{\|\mathsf{SAT}}$ queries its oracle non-adaptively and that it runs in time $2^{O(e(n))}$.

Claim 89. *If (v, x) is an rnK^t-description for some string $m \in \{0,1\}^s$, where m is an $\mathsf{rnK}^{t'}$-description for x, then $B^{\|\mathsf{SAT}}(v, x)$ outputs s with probability $1 - 2^{-\Omega(n^2)}$.*

Proof. Since (v, x) is an rnK^t-description for m, with probability at least $2/3$ over a random r, it holds that $Q_1((v, x), r, t, s) \in \mathsf{SAT}$, $Q_2((v, x), r, t) \notin \mathsf{SAT}$, and the concatenation of $\mathsf{SAT}(Q_3((v, x), r_j, t, i))$ for $i = 1, ..., s$ equals m. By a concentration bound, with probability $1 - 2^{-\Omega(n^2)}$, the fraction of r_j on which the above items hold is greater than $(1/2) + (1/10)$, in which case m is the majority value of $(z^{(j)})_{j \in S}$, and $|S'|/n^2 \geq (1/2) + (1/10)$.

When the above holds, for any $j \in S'$ (including the lexicographically first), m is the unique string not equal to $\perp$ such that there exists $w \in \{0,1\}^{\leq t}$ with $U((x,v),w,r_j) = m$ in time at most t. Thus, the first condition of Queries 4-6 with inputs (v,x), r_j, and t is always true, with $z = m$.

Since m is an $\mathsf{rnK}^{t'}$-description for x, with probability at least $2/3$ over a random choice of r', it holds that $Q_4((v,x),r_j,r',t,t',n) \in \mathsf{SAT}$, $Q_5((v,x),r_j,r'_k,t,t') \notin \mathsf{SAT}$, and moreover the concatenation of $\mathsf{SAT}(Q_6((v,x),r_j,r'_k,t,t',i))$ for $i = 1,...,n$ equals x. By a concentration bound, with probability $1 - 2^{-\Omega(n^2)}$, the fraction of r'_k on which the above items hold is greater than $(1/2) + (1/10)$, in which case x is the majority value of $(z^{(j,k)})_{k \in T}$, and $|T'|/n^2 \geq (1/2) + (1/10)$. In this case, $B^{\|\mathsf{SAT}}$ halts and outputs s. □

Claim 90. *If $\mathsf{rnK}^{t'}_{1/2}(x) > s$, then for any $v \in \{0,1\}^*$, with probability at least $1 - 2^{-\Omega(n^2)}$, $B^{\|\mathsf{SAT}}(v,x)$ does not output s.*

Proof. Suppose $\mathsf{rnK}^{t'}_{1/2}(x) > s$. Suppose that $B^{\|\mathsf{SAT}}(v,x)$ does not output $\perp$ in Step (4); otherwise, we are done. Let $m' \in \{0,1\}^s$ be the majority value of $(z^{(j)})_{j \in S}$, and fix j as in Step (5). In particular, m' is the unique string not equal to $\perp$ such that there exists $w \in \{0,1\}^{\leq t}$ with $U((x,v),w,r_j) = m'$ in time at most t. Note that $|m'|$ must be equal to s; otherwise, $B^{\|\mathsf{SAT}}$ would not output s, and we are done. Further, note that the first condition of Queries 4-6 with inputs (v,x), r_j, and t is always true, with $z = m'$.

By definition of $\mathsf{rnK}^{t'}_{1/2}$, with probability at least $1/2$ over a random choice of r', either $Q_4((v,x),r_j,r'_k,t,t',n) \notin \mathsf{SAT}$, $Q_5((v,x),r_j,r'_k,t,t') \in \mathsf{SAT}$, or the concatenation of $\mathsf{SAT}(Q_6((v,x),r_j,r'_k,t,t',i))$ for $i = 1,...,n$ is not equal to x. By a concentration bound, with probability at least $1 - 2^{-\Omega(n^2)}$ over the choice of $r'_1,...,r'_{n^2}$, the fraction of r'_k with this property is at least $(1/2) - (1/100)$.

Recall that, if x is not the majority value x' of $(z^{j,k})_{k \in T}$, then $B^{\|\mathsf{SAT}}$ outputs $\perp$. On the other hand, if x is x', then by the above, the fraction of $k \in [n^2]$ such that that $Q_5((v,x),r_j,r'_k,t,t') \notin \mathsf{SAT}$ *and* the concatenation of $\mathsf{SAT}(Q_6((v,x),r_j,r'_k,t,t',i))$ equals x' is at most $(1/2) + (1/100)$. It follows that $|T'|/n^2 < (1/2) + (1/10)$, and $B^{\|\mathsf{SAT}}$ outputs $\perp$. □

Define an algorithm $A^{\|\mathsf{SAT}}$ as follows.

> On input $x \in \{0,1\}^n$, enumerate all $v \in \{0,1\}^{c \cdot e(n)}$, and let $s_v = B^{\|\mathsf{SAT}}(v,x)$. Return the minimum s_v. If $s_v = \perp$ for all v, return $2n$.

Observe that, by definition of $B^{\|\mathsf{SAT}}$, $A^{\|\mathsf{SAT}}$ queries its oracle non-adaptively and runs in time $2^{O(e(n))}$.

Claim 91. *If $\mathsf{rnK}^{t'}(x) \leq s$, then $A^{\|\mathsf{SAT}}(x) \leq s$ with probability $1 - 2^{-\Omega(n)}$.*

Proof. If $\mathsf{rnK}^{t'}(x) \leq s$, then by Equation (22), $\mathsf{rnK}^t(m \mid x) \leq c \cdot e(n)$, where $m \in \{0,1\}^{\leq s}$ is an $\mathsf{rnK}^{t'}$-description for x such that $\mathsf{rnK}^{t'}(x) = |m|$. In particular, for some $v \in \{0,1\}^{c \cdot e(n)}$, (v,x) is an rnK^t-description for m. By Claim 89, with probability at least $1 - 2^{-\Omega(n^2)} \geq 1 - 2^{\Omega(n)}$, $B^{\|\mathsf{SAT}}(v,x)$ outputs $s_v = |m|$. In this case, $A^{\|\mathsf{SAT}}(x) = \min_v\{s_v\} \leq s$. □

Claim 92. *If $\mathsf{rnK}^{t'}_{1/2}(x) > s$, then $A^{\|\mathsf{SAT}}(x) > s$ with probability $1 - 2^{-\Omega(n)}$.*

Proof. By Claim 90, for any $v \in \{0,1\}^{c \cdot e(n)}$, for any $s' \leq s$, the probability that $B^{\|\mathsf{SAT}}(v,x)$ outputs s' is at most $2^{-\Omega(n^2)}$. The claim follows by a union bound. □

Finally, we explain how to use $A^{\|\mathsf{SAT}}$ to decide $\mathsf{Gap\text{-}MrnKtP}[s_1, s_2]$. Specifically, we will set $s_1(n) = n - 2d \cdot e(n)$ and $s_2(n) = n - 1$, for a constant $d > b$ to be specified below. Let $\theta = n - d \cdot e(n)$. To decide $\mathsf{Gap\text{-}MrnKtP}[s_1, s_2]$ on input x, simulate $A^{\|\mathsf{SAT}}(x)$, and accept iff the output is at most θ.

We now argue for the correctness of this approach. First suppose $\mathsf{rnKt}(x) \leq s_1(n) = n - 2d \cdot e(n)$. Then, by Lemma 87,

$$\mathsf{rnK}^{t'}(x) \leq n - 2d \cdot e(n) + b \cdot e(n) < \theta.$$

By Claim 91, $A^{\|\mathsf{SAT}}(x) \leq \theta$ with high probability, in which case x is accepted.

Now suppose $\mathsf{rnKt}(x) \geq s_2(n) = n - 1$. Applying success amplification for rnKt (Lemma 47), there exists a polynomial p such that

$$\mathsf{rnK}^{p(t')}(x) \leq \mathsf{rnK}^{t'}_{1/2}(x) + \log p(t'). \tag{23}$$

Also observe that $\mathsf{rnKt}(x) \leq \mathsf{rnK}^{p(t')}(x) + \log p(t')$. Thus, with high probability,

$$\begin{aligned}
A^{\|\mathsf{SAT}}(x) &\geq \mathsf{rnK}^{t'}_{1/2}(x) && \text{(Claim 92)}\\
&\geq \mathsf{rnK}^{p(t')}(x) - \log p(t') && \text{(Equation (23))}\\
&\geq \mathsf{rnKt}(x) - 2\log p(t')\\
&\geq n - 2\log p(t') - 1\\
&\geq n - 3k \cdot e(n)\\
&> n - d \cdot e(n) = \theta,
\end{aligned}$$

where we let $k \in \mathbb{N}$ be such that $p(m)^b \leq m^k$ for all $m \in \mathbb{N}$, and we set $d := 4k$. In this case, x is rejected.

This completes the proof of the lemma. □

10.1.2 The Case of nKt

Here, we extend the above results to nKt, which will be needed in the proof of Theorem 4.

Lemma 93. *Assume SoI holds for* nKt *with error* $e(n) \geq \log n$*. Then, there exists a time bound* $t \leq 2^{O(e(n))}$ *such that for every string* $x \in \{0,1\}^n$*, it holds that* $\mathsf{nK}^t(x) \leq \mathsf{nKt}(x) + O(e(n))$.

Proof Sketch. The proof follows essentially the same argument as in Lemma 87; see also the proof of Lemma 122. □

Lemma 94. *Assume SoI holds for* nKt *with error* $e(n) \geq \log n$*. Then,* $\mathsf{Gap\text{-}MnKtP}[s_1, s_2]$ *is contained in* $\mathsf{pr\text{-}DTIME}[2^{O(e(n))}]^{\|\mathsf{NP}}$*, where* $s_1(n) = n - O(e(n))$ *and* $s_2(n) = n - 1$.

Proof Sketch. The proof is a straightforward adaptation of that of Lemma 88, using Lemma 93 in place of Lemma 87. We omit the details. □

10.2 Meta-Complexity to SoI

The following lemma completes the proof of Item (ii) $\implies$ Item (i) in Theorem 6.

Lemma 95. *If* $\mathsf{Gap\text{-}MrnKtP}[s_1, s_2] \in \mathsf{pr\text{-}BPTIME}[n^{\mathsf{poly}(\log n)}]^{\|\mathsf{NP}}$*, where* $s_1(n) = n - \mathsf{poly}(\log n)$ *and* $s_2(n) = n - 1$*, then SoI holds for* rnKt *with error* $e(n) = \mathsf{poly}(\log n)$.

Proof. Let $D^{\|\mathsf{NP}}$ be a randomized quasi-polynomial time algorithm solving $\mathsf{Gap\text{-}MrnKtP}[s_1, s_2]$, making non-adaptive queries to some oracle in NP. Without loss of generality, we may assume that $D^{\|\mathsf{NP}}$ has failure probability $o(1)$.

The proof of SoI will proceed by a hybrid argument. Fix $n \in \mathbb{N}$. For $m \leq 2n$, let $G_m : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ be the generator of Lemma 31, where $d = O(\log^3 n)$. Consider any strings $x, y \in \{0,1\}^n$, and let $k, k' \in \mathbb{N}$ be parameters to be specified later satisfying $k, k' = \omega(1)$ and $k, k' \leq 2n$. By Lemma 23, for uniformly random strings $w \sim \mathcal{U}_k$ and $w \ \sim \mathcal{U}_{k'}$, with probability at least $1/2$,

$$\mathsf{rnKt}(w, w') \geq k + k' - 1.$$

Therefore,

$$\Pr_{w \sim \mathcal{U}_k;\, w' \sim \mathcal{U}_{k'}} \left[D^{\|\mathsf{NP}}(w, w') = 0 \right] \geq 1/2 - o(1),$$

Toward a contradiction, suppose that

$$\Pr_{z \sim \mathcal{U}_d;\, w' \sim \mathcal{U}_{k'};\, D} \left[D^{\|\mathsf{NP}}(G_k(x; z), w') = 1 \right] > \frac{2}{3}.$$

Then, $D^{\|\mathsf{NP}}(-, \mathcal{U}_{k'})$ is a distinguisher for $G_k(x; -)$, in which case, by Corollary 32,

$$\mathsf{rK}^{\mathsf{poly}(n),\, \|D^{\|\mathsf{NP}}}(x) \leq k + O(\log^3 n).$$

We can include a description of D in the bound on rnK, allowing non-adaptive queries to an NP oracle as needed. Since $D^{\|\mathsf{NP}}$ runs in time $n^{\mathsf{poly}(\log n)}$ on inputs of length $\mathsf{poly}(n)$, we have

$$\mathsf{rK}^{n^{\mathsf{poly}(\log n)},\, \|\mathsf{NP}}(x) \leq k + O(\log^3 n).$$

By Lemma 50, using the definition of rnK to remove the non-adaptive NP-oracle queries,

$$\mathsf{rnK}^{n^{\mathsf{poly}(\log n)}}_{n - \mathsf{poly}(\log n)}(x) \leq k + \mathsf{poly}(\log n).$$

Applying success amplification (Lemma 47),

$$\mathsf{rnK}^{n^{(\log n)^c}}(x) < k + \log^c n,$$

for some constant $c \in \mathbb{N}$.

Then, setting $k := \mathsf{rnK}^{n^{(\log n)^c}}(x) - \log^c n$ yields a contradiction.[21] It follows that

$$\Pr_{z \sim \mathcal{U}_d;\, w' \sim \mathcal{U}_{k'};\, D} \left[D^{\|\mathsf{NP}}(G_k(x; z), w') = 1 \right] \leq \frac{2}{3}.$$

By a very similar argument, assuming that the constant $c \in \mathbb{N}$ is sufficiently large, setting $k' := \mathsf{rnK}^{n^{(\log n)^c}}(y \mid x) - \log^c n$ implies that

$$\Pr_{z, z' \sim \mathcal{U}_d;\, D} \left[D^{\|\mathsf{NP}}(G_k(x; z), G_{k'}(y; z')) = 1 \right] \leq \frac{3}{4}.$$

[21] We may assume without loss of generality that $k = \mathsf{rnK}^{n^{(\log n)^c}}(x) - \log^c n \geq \log n$; otherwise, $\mathsf{rnKt}(x) \leq \mathsf{rnK}^{n^{\mathsf{poly}(\log n)}}(x) + \mathsf{poly}(\log n) \leq \mathsf{poly}(\log n)$, and the lemma follows from the fact that $\mathsf{rnKt}(x, y) \geq \mathsf{rnKt}(y \mid x)$.

It follows that

$$\Pr_{z,z' \sim \mathcal{U}_d} \left[\mathsf{rnKt}(G_k(x;z), G_{k'}(y;z')) \leq k + k' - \mathsf{poly}(\log n) \right] \leq \frac{3}{4} + o(1).$$

So, there exist $z, z' \in \{0,1\}^d$ such that

$$\begin{aligned} \mathsf{rnKt}(G_k(x;z), G_{k'}(y;z') &> k + k' - \mathsf{poly}(\log n) \\ &\geq \mathsf{rnK}^{n^{(\log n)^c}}(x) + \mathsf{rnK}^{n^{(\log n)^c}}(y \mid x) - \mathsf{poly}(\log n) \\ &\geq \mathsf{rnKt}(x) + \mathsf{rnKt}(y \mid x) - \mathsf{poly}(\log n), \end{aligned}$$

where the last inequality follows from the fact that $\mathsf{rnKt}(u \mid v) \leq \mathsf{rnK}^t(u \mid v) + O(\log t)$ for any time-bound $t \in \mathbb{N}$ and $u, v \in \{0,1\}^*$.

Lastly, observe that

$$\mathsf{rnKt}(G_k(x;z), G_{k'}(y;z')) \leq \mathsf{rnKt}(x,y) + O(d) + O(\log n).$$

Combined with the above, this yields

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

as desired. □

10.3 Natural Properties and Explicit Constructions

In this section we show that Items (ii) to (iv) of Theorem 6 are equivalent. In fact, the equivalence between Items (iii) and (iv) is straightforward: one can verify that they are in fact exactly the same statement. Item (ii) $\implies$ Item (iii) is also trivial, by defining $\mathcal{YES} = \{x \mid \mathsf{rnKt}(x) \geq |x| - 1\}$, and $\mathcal{NO} = \{x \mid \mathsf{rnKt}(x) \leq |x| - \mathsf{poly}(\log |x|)\}$. So the only non-trivial implication we need to prove is Item (iii) $\implies$ Item (ii), which we restate as a lemma below:

Lemma 96 (Worst-case to Average-case Reduction for Gap-MrnKtP)**.** *Assume that there exist a constant C and a promise problem $P = (\mathcal{YES}, \mathcal{NO}) \in \mathsf{pr\text{-}BPTIME}[n^{\log^C n}]^{\|\mathsf{NP}}$ such that*

- $|\mathcal{YES} \cap \{0,1\}^n| \geq 2^{n-1}$*, and*
- *For every $x \in \{0,1\}^n$ where $\mathsf{rnKt}(x) < n - \log^C n$, $x \in \mathcal{NO}$.*

Then there exists a constant C', such that

$$\mathsf{Gap\text{-}MrnKtP}[n - \log^{C'} n, n-1] \in \mathsf{pr\text{-}BPTIME}[n^{\log^{C'} n}]^{\|\mathsf{NP}}.$$

Using ideas similar to the worst-case to average-case reduction in [Hir18], we are able to prove this lemma.

Proof. Let $C' = C + 10$. Let A be the algorithm in $\mathsf{pr\text{-}BPTIME}[n^{\log^C n}]^{\|\mathsf{NP}}$ which accepts any string in $\mathcal{YES}$ with probability at least $9/10$, and rejects any string in $\mathcal{NO}$ with probability at least $9/10$. Let G be the pseudo-random generator of Lemma 31, with $\varepsilon = 1/n$ and $m = n - \frac{\log^{C'} n}{2}$. This would give $d = O(\log^3 n)$. Our algorithm B for deciding $\mathsf{Gap\text{-}MrnKtP}[n - \log^{C'} n, n-1]$ works as follows:

On input x, B uniformly samples a string $z \sim U_d$, and accepts iff $A(G(x,z))$ rejects.

It is easy to see that B makes non-adaptive queries to NP and runs in time $\mathsf{poly}(n) \cdot n^{\log^C n}$, which is smaller than $n^{\log^{C'}(n)}$ for all large enough n. It remains to argue for the correctness of B.

Claim 97. *For any x such that $\mathsf{rnKt}(x) \leq n - \log^{C'} n$, B accepts x with probability at least $9/10$.*

Proof. For any $z \in \{0,1\}^d$, by storing z in the description of the program and noticing that G is poly-time computable, for large enough n we have that

$$\mathsf{rnKt}(G(x,z)) \leq \mathsf{rnKt}(x) + |z| + O(\log n) \leq n - \log^{C'}(n) + O(\log^3 n) \leq n - \log^C(n).$$

Hence $G(x,z)$ is in $\mathcal{NO}$, and A rejects with probability at least $9/10$. In other words, $B(x)$ accepts with probability at least $9/10$. □

Claim 98. *For any x such that $\mathsf{rnKt}(x) \geq n - 1$, B accepts x with probability at most $7/10$.*

Proof. For the sake of contradiction, assume there exists x such that $\mathsf{rnKt}(x) \geq n-1$, but B accepts x with probability larger than $7/10$. By definition of B, we get

$$\mathop{\mathbf{Pr}}_{z \sim U_d, A}\left[A^{||\mathsf{NP}}(G(x,z)) = 1\right] \leq \frac{3}{10}.$$

Notice that for s uniformly sampled from $\{0,1\}^m$, with probability at least $1/2$ s is in $\mathcal{YES}$, so we get

$$\mathop{\mathbf{Pr}}_{s \sim U_m, A}\left[A^{||\mathsf{NP}}(s) = 1\right] \geq \frac{9}{20}.$$

Hence we have that

$$\mathop{\mathbf{Pr}}_{s \sim U_m, A}\left[A^{||\mathsf{NP}}(s) = 1\right] - \mathop{\mathbf{Pr}}_{z \sim U_d, A}\left[A^{||\mathsf{NP}}(G(x,z)) = 1\right] \geq \frac{3}{20} \geq \frac{2}{n}.$$

In other words, $A^{||\mathsf{NP}}(\cdot)$ is a $2/n$-distinguisher for $G(x,\cdot)$. Then by Corollary 32, we would get

$$\mathsf{rK}^{\mathsf{poly}(n), ||A^{||\mathsf{NP}}}(x) \leq m + O(\log^3 n).$$

Since A runs in time $n^{\log^C n}$, we can store A in the description, and let the program simulate A when constructing x. Then the program runs in time $\mathsf{poly}(n) \cdot n^{\log^C n} \leq n^{\log^{C+1} n}$ for all large enough n. This would give

$$\mathsf{rK}^{n^{\log^{C+1} n}, ||\mathsf{NP}}(x) \leq m + O(\log^3 n).$$

By Lemmas 47 and 50, for all large enough n we have

$$\mathsf{rnK}^{n^{\log^{C+2} n}}(x) \leq \mathsf{rK}^{n^{\log^{C+1} n}, ||\mathsf{NP}}(x) + O(\log^{C+2} n) \leq m + O(\log^{C+2} n).$$

Hence for all large enough n we get

$$\mathsf{rnKt}(x) \leq \mathsf{rnK}^{n^{\log^{C+2} n}}(x) + \log^{C+3} n \leq m + O(\log^{C+3} n) < n - 1.$$

But this contradicts the assumption that $\mathsf{rnKt}(x) \geq n - 1$. □

Combining Claim 97 and Claim 98, there is a constant gap between the accepting probability of two cases for B. By a standard success amplification argument, we obtain an algorithm B' that decides $\mathsf{Gap\text{-}MrnKtP}[n - \log^{C'} n, n-1]$ with error at most $1/3$, while running in time $n^{\log^{C'} n}$ and making non-adaptive queries to NP. □

11 A Quick Lower Bound Result via the Correspondence Theorem: Proof of Theorem 7

In this section, for completeness, we provide a proof of Theorem 7, which follows from Theorem 5 and Theorem 6.

Proof of Theorem 7. First consider the case that, for some constant $c \in \mathbb{N}$, SoI holds for rnKt with error $e(n) = (\log n)^c$. Then, by Item (ii) of Theorem 5,

$$\mathsf{MKtP} \notin \mathsf{NTIME}[n^{\mathsf{poly}(\log n)}] \cap \mathsf{coNTIME}[n^{\mathsf{poly}(\log n)}].$$

Otherwise, for every constant $c \in \mathbb{N}$, SoI does not hold for rnKt with error $e(n) = (\log n)^c$. In this case, by the implication from Item (ii) to Item (i) in Theorem 6 (as shown in Lemma 95),

$$\mathsf{Gap\text{-}MrnKtP}[n - \mathsf{poly}(\log n), n-1] \notin \mathsf{pr\text{-}BPTIME}[n^{\mathsf{poly}(\log n)}]^{\|\mathsf{NP}}.$$

Since one of the conditions above must hold, at least one of the consequences must hold. This completes the proof. □

12 Unconditional Hardness of MnKtP: Proof of Theorem 8

In this section we prove the moreover part of Theorem 8, from which the main part follows directly. We first define our technical tool: rK, but with restricted length of random bits.

Definition 99. 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 $\delta = 2/3$.

12.1 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 100. *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 $n < t$, 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 101 (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:*

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

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

Proof of Lemma 100. 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 101. 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

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

One can easily see that $|S_p| \leq 1/p$. Now applying Lemma 101 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 $(M, A, u, H(x))$. For now, assume $2^r \geq 256/\delta^2$. 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 102. *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 102. 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

$$\mathbf{Pr}\left[\sum_{i=1}^{k} V_{i,\{x\}} \leq \frac{\delta}{2}\cdot k\right] \leq \frac{\mathrm{Var}[\sum_{i=1}^{k} V_{i,\{x\}}]}{(kp_{\{x\}} - \delta k/2)^2} \leq \frac{kp_{\{x\}}}{k^2 p_{\{x\}}^2/4} \leq \frac{4}{k\delta} \leq \frac{\delta}{64}. \tag{24}$$

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

$$\mathbf{Pr}\left[\sum_{i=1}^{k} V_{i,B_j} \geq \frac{\delta}{2}\cdot k\right] \leq \frac{\mathrm{Var}[\sum_{i=1}^{k} V_{i,B_j}]}{(kp_{B_j} - \delta k/2)^2} \leq \frac{kp_{B_j}}{k^2\delta^2/16} \leq \frac{4}{k\delta} \leq \frac{\delta}{64}. \tag{25}$$

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 (25), 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 $S' \not\subseteq S_{\delta/8}$:

$$\begin{aligned}
\mathbf{Pr}\left[S' \not\subseteq S_{\delta/8}\right] &= \mathbf{Pr}\left[\exists x' \in \overline{S_{\delta/8}} \text{ s.t. } \sum_{i=1}^{k} V_{i,\{x'\}} \geq \frac{\delta}{2}\cdot k\right] \\
&\leq \mathbf{Pr}\left[\exists j \in [d] \text{ s.t. } \sum_{i=1}^{k} V_{i,B_j} \geq \frac{\delta}{2}\cdot k\right] \\
&\leq \frac{1}{4}.
\end{aligned} \tag{26}$$

Applying union bound to Equations (24) and (26), 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. ◇

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 101, $H(x) \neq H(x')$ for any $x' \in S' \setminus \{x\}$. Therefore, with probability $2/3$, the algorithm B can always find the correct x.

It remains to handle the case $2^r < 256/\delta^2$. Here, the algorithm is deterministic: it enumerates all $2^r < k = O(1/\delta^2)$ strings $w \in \{0,1\}^r$, runs $M^{\mathcal{O}_{\leq l}}(y, w)$ for t steps on each, and lets S' be the set of outputs occurring for at least a δ fraction of the strings w. By the definition of δ, we have $x \in S'$. By construction, $S' \subseteq S_\delta \subseteq S_{\delta/8}$, so the stored hash value $H(x)$ identifies x uniquely within S' by Lemma 101. The algorithm therefore outputs x with probability 1, runs in time $2^r \cdot t \leq \mathsf{poly}(t/\delta)$, and has the same description length as above.

□

12.2 Reconstruction from Oracle Access to Dense Language

Lemma 103. *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 31, 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 31.

Corollary 104. *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 31. 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 $B^{(\cdot)} : \{0,1\}^m \times \{0,1\}^{n'} \times \{0,1\}^{r'} \to \{0,1\}$ 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 100, 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 31, 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

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

Therefore, we can store z, $A(x,z)$, as well as the description of R and B. Then given y and oracle access to $\mathcal{O}_{\le l}$, 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

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

Applying Lemma 100 completes the proof. □

We may now prove Lemma 103.

Proof of Lemma 103. For the sake of contradiction, assume the contrary. Then we have

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

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

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

Then by a hybrid argument, we either have

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

or

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

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 $m_1 + m_2 \le 3n$. Hence by Corollary 104, we have

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

But by definition, $m_1 = \mathsf{rK}^{(n^{C\cdot b};C\cdot n),L_{\le 3n}}(x \mid y) - C \cdot \log^3(n^b)$, 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 $m_1 + m_2 \le 3n$. Hence by Corollary 104, we have

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

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

12.3 Derandomizing rK^L

Lemma 105. *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 105. 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 31, with $\varepsilon = 1/6$. Let $M^{(\cdot)}$ 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 106. *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*

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

Proof of Claim 106. Suppose the contrary. Then it is not hard to see that the following algorithm $B^{(\cdot)}$ 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 $M^{(\cdot)}$, by Corollary 104, we have

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

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

$$\mathsf{K}^{\mathsf{poly}(t\cdot N'),L_{\leq l}}(s) \leq O(r + n + m + \log^3 N' + \log t). \tag{27}$$

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

$$\begin{aligned}
\mathsf{nKt}(s) &\leq \mathsf{nK}^{\mathsf{poly}(t\cdot N'\cdot 2^{\varepsilon l})}(s) + O(\log t + \log N' + \varepsilon l) \\
&\leq \mathsf{nK}^{\mathsf{poly}(t\cdot N'),L_{\leq l}}(s) + O(\log t + \log N' + \varepsilon l) && \text{(by Lemma 45)} \\
&\leq \mathsf{K}^{\mathsf{poly}(t\cdot N'),L_{\leq l}}(s) + O(\log t + \log N' + \varepsilon l) \\
&\leq O(n + m + l + r + \log t + \log^3 N'). && \text{(by Equation (27))}
\end{aligned}$$

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. ◇

Now let C_1 be a large enough constant such that Claim 106 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 106, 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 $L \cap \{0,1\}^{N'} \neq \varnothing$ 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

$$\mathsf{nK}^{\mathsf{poly}(2^d\cdot t\cdot N'),L_{\le N'}}(x \mid y) \le \mathsf{rK}^{L_{\le l},(t;r)}(x \mid y) + O(\log(t\cdot N')). \tag{28}$$

Therefore we have

$$\begin{aligned}\mathsf{nKt}(x \mid y) &\le \mathsf{nK}^{\mathsf{poly}(2^{\varepsilon N'}\cdot 2^d\cdot t)}(x \mid y) + O(\varepsilon N' + d + \log t)\\ &\le \mathsf{nK}^{\mathsf{poly}(2^d\cdot t\cdot N'),L_{\le N'}}(x \mid y) + O(\varepsilon N' + d + \log t) && \text{(by Lemma 45)}\\ &\le \mathsf{rK}^{(t;r),L_{\le l}}(x \mid y) + O(\varepsilon N' + d + \log t) && \text{(by Equation (28))}\\ &\le \mathsf{rK}^{(t;r),L_{\le l}}(x \mid y) + O(\varepsilon N' + \log^3 N' + \log t) && \text{(Since } d = O(\log^3 N'))\end{aligned}$$

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

12.4 Iterative Construction of High-Complexity Strings

Combining Lemma 103 and Lemma 105, we have the following lemma, which we will use to prove Theorem 8.

Lemma 107. *There exists some constant $C \in \mathbb{N}$ such that the following holds. Suppose 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. $\forall n \in \mathbb{N}$, $|L \cap \{0,1\}^n| \geq 2^n/n^b$.

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

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

We define G_1, G_2 to be the PRG of Lemma 31, with distinguishing parameter $1/2n^b$:

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

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 103 and Lemma 105. □

Proof of Theorem 8. We prove the "moreover" part of this theorem, of which the main part is an immediate corollary. Fix some $\gamma \in (1/2, 1)$ and $b \in \mathbb{N}$, and let ε be a parameter to be fixed below. For the sake of contradiction, assume that there exists some language L satisfying the following three properties:

1. $L \in \mathsf{NTIME}[2^{\varepsilon n}] \cap \mathsf{coNTIME}[2^{\varepsilon n}]$,
2. $\forall x \in L$, $\mathsf{nKt}(x) \geq \gamma \cdot |x|$,
3. for all sufficiently large $n \in \mathbb{N}$, $|L \cap \{0,1\}^n| \geq 2^n/n^b$.

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 31 with security parameter ε_i.
    Instantiate $G_2 : \{0,1\}^{N_{i-1}} \times \{0,1\}^{d_2} \to \{0,1\}^{m_{i,2}}$ in Lemma 31 with security parameter ε_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 108. *For a fixed γ, if ε is small enough, then for all large enough n, there exists some integer $i_0 \le \log_{\gamma'}(1 - \frac{1}{2\gamma'}) + 2$ such that*

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

Proof of Claim 108. We need to prove two things: first, for every i during the iteration, we can find some s_i s.t. $s_i \neq \bot$, 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 $s_1 \neq \bot$. Then suppose $s_{i-1} \neq \bot$ for some $i \ge 2$. Since $\gamma > 1/2$, by Lemma 107, there exists some universal constant C such that whenever $m_{i,1}$ and $m_{i,2}$ satisfy

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

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 23), 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

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

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. $\varepsilon < \frac{\gamma - \gamma'}{20C}$), 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 $G_1(x, z_1) \circ G_2(s_{i-1}, z_2) \in L$, 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 $t_0 := 2^n \cdot \mathsf{poly}(n) \cdot 2^{O(\log^3 n)} \cdot i_0$, 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

$$\mathsf{K}^{t_0, L_{\leq 5n}}(s_{i_0}) \leq O\left(\log n + \log\log\left(\frac{1}{\gamma - 1/2}\right)\right). \tag{29}$$

Let c be the constant of Lemma 45. Applying that lemma with $T(m) = 2^{\varepsilon m}$ and query length $\ell := 5n$, and using that $\mathsf{K}^{t_0, L_{\leq 5n}}(s_{i_0}) \geq \mathsf{nK}^{t_0, L_{\leq 5n}}(s_{i_0})$, we obtain

$$\mathsf{nK}^{t_1}(s_{i_0}) \leq \mathsf{K}^{t_0, L_{\leq 5n}}(s_{i_0}) + O(\log\log t_0), \qquad t_1 := t_0 \cdot 2^{5\varepsilon n} \cdot \log^c\left(2^{5\varepsilon n} \cdot t_0\right).$$

Since $\log t_0 = n + O(\log^3 n + \log i_0)$, we have $\log t_1 \leq n + 5\varepsilon n + O(\log^3 n + \log i_0)$, and therefore

$$\begin{aligned} \mathsf{nKt}(s_{i_0}) &\leq \mathsf{nK}^{t_1}(s_{i_0}) + \log t_1 \\ &\leq \mathsf{K}^{t_0, L_{\leq 5n}}(s_{i_0}) + n + 5\varepsilon n + O(\log^3 n + \log i_0) \\ &\leq n + 5\varepsilon n + O\left(\log^3 n + \log i_0 + \log\log\left(\frac{1}{\gamma - 1/2}\right)\right) && \text{(by Equation (29))} \\ &\leq n + 5\varepsilon n + O\left(\log^3 n + \log\log\left(\frac{1}{\gamma - 1/2}\right)\right), && (30) \end{aligned}$$

where the last inequality uses $i_0 \leq O(\log \frac{1}{\gamma - 1/2})$, which follows from $i_0 \leq \log_{\gamma'}(1 - \frac{1}{2\gamma'}) + 2$ (Claim 108), $\gamma' = (\gamma + 1/2)/2$ and $1/2 < \gamma < 1$.

On the other hand, since $s_{i_0} \in L$,

$$\mathsf{nKt}(s_{i_0}) \;\geq\; \gamma \cdot |s_{i_0}| = \theta_{i_0} \;\geq\; \frac{n}{2-2\gamma'}. \tag{31}$$

Writing $\Delta_\gamma := \frac{1}{2-2\gamma'} - 1 > 0$, which is positive because $\gamma' > 1/2$, Equation (31) implies that $\mathsf{nKt}(s_{i_0}) \geq n + \Delta_\gamma \cdot n$; thus, the two bounds are contradictory for all sufficiently large n provided $5\varepsilon < \Delta_\gamma/2$. Recalling the constraint from Claim 108, it therefore suffices to choose

$$\varepsilon \;<\; \min\left\{\frac{\gamma-\gamma'}{20C},\; \frac{\Delta_\gamma}{10}\right\},$$

where C is the constant of Lemma 107. This choice depends only on γ and on the constants of Lemma 107 and Lemma 45, which completes the proof.

□

12.5 Limitations of the Iterative Construction for $\gamma \leq 1/2$

We believe the statement is also true for $0 < \gamma \leq 1/2$. 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 $\mathsf{nKt}(s) \geq (1+\Omega(1)) \cdot n$, resulting in a contradiction. But by the iterative construction, we are only guaranteed that $\mathsf{nKt}(s) \geq (\gamma + \gamma^2 + \gamma^3 + \cdots) \cdot n - o(n)$, which is upper bounded by $\frac{\gamma}{1-\gamma} \cdot n - o(n)$. Hence when $\gamma \leq 1/2$, $\frac{\gamma}{1-\gamma} \leq 1$, and we don't have $\mathsf{nKt}(s) \geq (1+\Omega(1)) \cdot n$, 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 2). In the setting of a dense language $L \subseteq \mathcal{R}_{\mathsf{nKt}[\gamma\cdot n]}$, we are able to prove that

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

by reconstruction (Lemma 103) and derandomization (Lemma 105). This degrades to a trivial inequality when $\gamma \leq 1/2$, since

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

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.

13 Unconditional Hardness of Gap-MrnKtP: Proof of Theorem 9

In this section, we prove the "moreover" parts in Theorem 9. These parts are strengthened versions of the plain complexity lower bounds for Gap-MrnKtP in Theorem 9. To see this, we observe the following simple lemma.

Lemma 109. *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 $\mathsf{Gap\text{-}MrnKtP}[\theta(n), n-1] \notin \mathsf{co}\text{-}\mathcal{C}$.*

Proof. We prove the contrapositive. Suppose $\mathsf{Gap\text{-}MrnKtP}[\theta(n), n-1] \in \mathsf{co}\text{-}\mathcal{C}$. Then there exists a language $L_0 \in \mathsf{co}\text{-}\mathcal{C}$ that is consistent with $\mathsf{Gap\text{-}MrnKtP}[\theta(n), n-1]$, 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 23. □

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

In this subsection, we give proof to Theorem 9, Item 1. The plan is similar to the proof of Theorem 8, where we break reconstructive black-box PRG (See Lemma 31) 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 8, as discussed in Section 12.5.

Lemma 110. *For any $c > 0, 1/2 < \gamma < 1$, there exist constants $C_1, C_2 > 0$ such that for any $L \subseteq \mathcal{R}_{\mathsf{rnKt}[\gamma \cdot n]}$ satisfying $|L_n| \geq 2^n/n^c$, the following holds.*

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 31 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 follows: 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 $m_1 \leq 2n_1, m_2 \leq 2n_2$, we have query length here bounded by $2n_1 + 2n_2$, as well as $m \leq \mathsf{poly}(n_1+n_2), \varepsilon^{-1} \leq \mathsf{poly}(n_1+n_2)$.

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

Similarly, for the latter 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 104, 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, $m_1 \le \mathsf{rK}^{(n_1+n_2)^{C_1},L_{\le 2n_1+2n_2}}(x \mid y) - C_2\log^3(n_1+n_2)$, we then obtain a contradiction when C_1, C_2 are large enough. □

We aim to derive contradiction from Lemma 110, as in the proof of Theorem 8, which finishes the proof of Theorem 9, Item 1.

Proof of Theorem 9, 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

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

where $\varepsilon > 0$ is chosen below independently of L.

Similar to Lemma 107, 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 31 with security parameter ε_i.
    Instantiate $G_2 : \{0,1\}^{N_{i-1}} \times \{0,1\}^{d_2} \to \{0,1\}^{m_{i,2}}$ in Lemma 31 with security parameter ε_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.

21: **return**
22: Output $\perp$.

We hereby analyse behaviour of A^L.

Claim 111. *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 $s_{i^*} \neq \perp$ with $\mathsf{rnKt}(s_{i^*}) > n/(2-2\gamma')$. Here, the constant factor hidden in big-O notation depends on γ, but can be fixed when γ is fixed.*

To prove Claim 111, we prove the following crucial statement.

Claim 112. *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 $s_i \neq \perp$ is produced successfully, or the algorithm terminates with output s_j for some $j < i$.*

Proof of Claim 112. 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 $s_{i+1} \in L$.

By induction assumption, $\mathsf{rnKt}(s_i) \geq \gamma N_i = \theta_i$. Let $C_1, C_2 > 0$ be the constants from Lemma 110. Let c' be the constant of Lemma 51, and write $t_i := (n+N_i)^{C_1}$ and $\ell_i := 2n + 2N_i$. Applying the second part of Lemma 51 with $T(m) = 2^{\varepsilon m}$, and noting that $|s_i| = N_i \leq \ell_i$, we obtain

$$\mathsf{rnK}^{t_i'}(s_i) \leq \mathsf{rnK}^{t_i, L_{\leq \ell_i}}(s_i) + O(\log(n+N_i)), \qquad t_i' := t_i \cdot \left(2^{\varepsilon \ell_i} \cdot |s_i| \cdot \log t_i\right)^{c'},$$

so that $\log t_i' \leq 2c'\varepsilon(n+N_i) + O(\log(n+N_i))$. Since every deterministic oracle program is in particular a randomized nondeterministic one, we have $\mathsf{rnK}^{t_i, L_{\leq \ell_i}}(s_i) \leq \mathsf{rK}^{t_i, L_{\leq \ell_i}}(s_i)$, and therefore

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

Rearranging and using $\mathsf{rnKt}(s_i) \geq \theta_i$, we obtain

$$\mathsf{rK}^{t_i, L_{\leq \ell_i}}(s_i) \;\geq\; \theta_i - 2c'\varepsilon(n+N_i) - O(\log(n+N_i)).$$

Note that, since $\gamma > 1/2$, we have $\gamma'/\gamma = 1/2 + 1/(4\gamma) < 1$. Recall also that $N_i \leq 4n$, so $n + N_i \leq 5n \leq 5\theta_i/\gamma$. Hence, for ε chosen small enough as a function of γ and c' (and independently of L), we get

$$m_{i+1,2} = \frac{\gamma'}{\gamma}\theta_i \leq \mathsf{rK}^{t_i, L_{\leq \ell_i}}(s_i) - C_2 \log^3(n+N_i). \tag{32}$$

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

$$m_{i+1,1} = \frac{\gamma'}{\gamma}n \leq n - C_2 \log^3(n+N_i) \leq \mathsf{rK}^{(n+N_i)^{C_1}, L_{\leq 2n+2N_i}}(x \mid s_i) - C_2 \log^3(n+N_i). \tag{33}$$

Combining Equation (33) and Equation (32), we apply Lemma 110, which indicates for any sufficiently large n, $A^L(\gamma, 1^n)$ successfully produces $s_{i+1} \in L$. $\diamond$

Proof of Claim 111. Fix any $1/2 < \gamma < 1$. For any $i \geq 2$, observe that if $A^L(\gamma, 1^n)$ produces $s_i \neq \perp$, 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

$$\theta_i > \left(\sum_{j=1}^{i} \gamma'^j \right) n = \frac{\gamma'(1-\gamma'^i)n}{1-\gamma'}, \quad \theta_i < \left(\sum_{j=1}^{i} \gamma^j \right) n < \frac{\gamma n}{1-\gamma}, \tag{34}$$

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 112 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 111, we compute $s_{i^*} := A^L(\gamma, 1^n)$, and we have $\mathsf{rnKt}(s_{i^*}) > n/(2-2\gamma')$. On the other hand, the running time bound given in Claim 111 indicates

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

where $t(n) = O(2^{n+O(\log^3 n)})$. By Equation (34), 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,

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

Let $\ell := n/(1-\gamma)$. Applying the second part of Lemma 51 with $T(m) = 2^{\varepsilon m}$, and letting $t' := t(n) \cdot \left(2^{\varepsilon \ell} \cdot |s_{i^*}| \cdot \log t(n)\right)^{c'}$ be the resulting time-bound from that lemma, we obtain

$$\mathsf{rnK}^{t'}(s_{i^*}) \leq \mathsf{rnK}^{t(n), L_{\leq \ell}}(s_{i^*}) + O(\log n) = O(\log n).$$

Substituting $t(n) = O(2^{n+O(\log^3 n)})$ and $|s_{i^*}| \le \ell = O(n)$ gives $\log t' \le n + \frac{c'\varepsilon n}{1-\gamma} + O(\log^3 n)$, and therefore

$$\mathsf{rnKt}(s_{i^*}) \le \mathsf{rnK}^{t'}(s_{i^*}) + \log t' \le n + \frac{c'\varepsilon n}{1-\gamma} + O(\log^3 n).$$

On the other hand, $\mathsf{rnKt}(s_{i^*}) > n/(2-2\gamma')$, and $1/(2-2\gamma') > 1$ since $\gamma' > 1/2$. Writing $\Delta_\gamma := \frac{1}{2-2\gamma'} - 1 > 0$ and choosing ε such that $\varepsilon < \Delta_\gamma(1-\gamma)/(2c')$ and Claim 112 holds, the two bounds contradict each other for all sufficiently large n.

□

13.2 AMTIME Lower Bound via Explicit Constructions

We formally state the technical result from [CLL25], which is mentioned in Section 2. We begin with a formal definition of *single-valued Arthur-Merlin protocol*, which is a natural interpretation from [CLL25].

Definition 113. A *non-binary public coin protocol* (P, V) is defined as follows: On any input x, V, P interact by constantly many rounds; In the i-th round, V (the verifier) sends plain uniform randomness $r_i \in \{0,1\}^{n_i}$ to P; P (the prover) then returns w_i according to $x, r_1, w_1, \cdots, r_{i-1}, w_{i-1}, r_i$. After the last round, V outputs some string y as answer, which deterministically depends on the whole interaction transcript (all w_j, r_j'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 $\alpha_n \in \{0,1\}^{l(n)}$ and an honest prover P 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).

Lemma 114 (Round reduction for single-valued Arthur-Merlin protocol)**.** *For any k-round single-valued Arthur-Merlin protocol (P, V) computing $\{x_n\}_{n\in\mathbb{N}}$ with advice complexity $l(n)$ where the verifier V runs in time $T(n)$, there exists a 2-round single-valued Arthur-Merlin protocol (P', V') computing $\{x_n\}_{n\in\mathbb{N}}$ with advice complexity $l(n)$ where the verifier V runs in time $T(n)^{O_k(1)}$.*

Proof. We translate the single-valued Arthur-Merlin protocol into a decision problem, using the round reduction results for decision problems with Arthur-Merlin protocol, and finally translate it back into the single-valued Arthur-Merlin protocol.

We define a promise problem $\mathcal{P} = (\mathcal{YES}, \mathcal{NO})$, where

$$(1^n, \alpha, i, b) \in \mathcal{YES} \iff \exists \text{ prover } P, \Pr_V[(P, V)(1^n, \alpha) \neq \perp \wedge i\text{'th bit of } (P, V)(1^n, \alpha) \text{ is } b] \ge \frac{3}{5},$$

$$(1^n, \alpha, i, b) \in \mathcal{NO} \iff \forall \text{ prover } P', \Pr_V\left[(P', V)(1^n, \alpha) \neq \perp \wedge i\text{'th bit of } (P', V)(1^n, \alpha) \text{ is } b\right] \le \frac{2}{5}.$$

By definition, $\mathcal{P}$ has a $T(n)$-time binary Arthur-Merlin protocol with k rounds. By the standard round reduction and error reduction for binary Arthur-Merlin protocol[BM88, GS86], $\mathcal{P}$ has a $T(n)^{O_k(1)}$-time 2-round binary Arthur-Merlin protocol with perfect completeness and small soundness error, in other words,

$\mathcal{P} \in \mathsf{prAMTIME}[T(n)^{O_k(1)}]$. Let $\hat{V}$ be the verifier in the protocol, then we have,

$$(1^n, \alpha, i, b) \in \mathcal{YES} \implies \Pr_r\left[\exists w, \hat{V}(1^n, \alpha, i, b, r, w) = 1\right] = 1,$$
$$(1^n, \alpha, i, b) \in \mathcal{NO} \implies \Pr_r\left[\exists w, \hat{V}(1^n, \alpha, i, b, r, w) = 1\right] \leq \frac{1}{3n}.$$

Next, we define a 2-round single-valued Arthur-Merlin protocol that computes $\{x_n\}_{n\in\mathbb{N}}$. Given input $(1^n, \alpha)$, the verifier V' first samples $2n$ random strings $r_1^0, r_2^0, \ldots, r_n^0, r_1^1, r_2^1, \ldots, r_n^1$ and send them to the prover. Then the prover sends back $2n$ strings $w_1^0, w_2^0, \ldots, w_n^0, w_1^1, w_2^1, \ldots, w_n^1$ and a proposed answer $\hat{x} \in \{0,1\}^n$. The verifier then runs $\hat{V}(1^n, \alpha, i, b, r_i^b, w_i^b) =: v_i^b$ for each $i \in [n]$ and $b \in \{0,1\}$. If for some i, $v_i^0 = v_i^1 = 0$, then V' prints $\perp$. Otherwise, V' outputs a string $x \in \{0,1\}^n$ defined as follows. For each $i \in [n]$, if $v_i^0 = v_i^1 = 1$, then set the i'th bit of x as the i'th bit of $\hat{x}$. Otherwise set the i'th bit of x as the b such that $v_i^b = 1$.

It is not hard to see that V' runs in time $T(n)^{O_k(1)}$, and has advice complexity of $|\alpha| = l(n)$. It remains to argue about the correctness, i.e. the conformity and resiliency of V'.

- **Conformity:** By the conformity of V, there exists α_n and honest prover P such that $(P, V)(1^n, \alpha_n) = x_n$ with probability 1 over randomness of V. Let $x_{n,i}$ denote the i'th bit of x_n. Then by definition of $\mathcal{P}$, we know that $(1^n, \alpha_n, i, x_{n,i}) \in \mathcal{YES}$ for each $i \in [n]$. The honest prover set $\hat{x} = x_n$, select $w_i^{x_{n,i}}$ such that $v_i^{x_{n,i}} = \hat{V}(1^n, \alpha_n, i, x_{n,i}, r_i^{x_{n,i}}, w_i^{x_{n,i}}) = 1$, and set $w_i^{1-x_{n,i}}$ as the all 0 string. If $v_i^{1-x_{n,i}} = \hat{V}(1^n, \alpha_n, i, 1-x_{n,i}, r_i^{1-x_{n,i}}, w_i^{1-x_{n,i}}) = 1$, then V' would set the i'th bit of x as $\hat{x}$, which is equal to x_n because the prover is honest. Otherwise, $v_i^{1-x_{n,i}} = 0$, and V' would set the i'th bit of x as $x_{n,i}$, which is also correct. To summarize, an honest prover would make V' print x_n with probability 1.

- **Resiliency:** By the resiliency of V, for any $\zeta_n \in \{0,1\}^{l(n)}$, there exists $y_n \in \{0,1\}^n$ such that, for any prover P', $(P', V)(1^n, \zeta_n) \in \{y_n, \perp\}$ with probability at least $2/3$ over randomness of V. Let $y_{n,i}$ denote the i'th bit of y_n. Then by the resiliency of V, $(1^n, \zeta_n, i, 1-y_{n,i}) \in \mathcal{NO}$ for every $i \in [n]$. Thus for each $i \in [n]$, the probability of existing $w_i^{1-y_{n,i}}$ such that $\hat{V}(1^n, \alpha, i, 1-y_{n,i}, r_i^{1-y_{n,i}}, w_i^{1-y_{n,i}}) = 1$ is at most $\frac{1}{3n}$. By a union bound, with probability at least $2/3$, for every $i \in [n]$, for every $w_i^{1-y_{n,i}}$, we have $v_i^{1-y_{n,i}} = \hat{V}(1^n, \alpha, i, 1-y_{n,i}, r_i^{1-y_{n,i}}, w_i^{1-y_{n,i}}) = 0$. Conditioning on this, if for some i, $v_i^{y_{n,i}} = 0$, then V' would output $\perp$. Otherwise, for every i, $v_i^{y_{n,i}} = 1$, then V' would output y_n. Hence with probability at least $2/3$ over the randomness of V', regardless of prover's strategy, V' either prints y_n or $\perp$.

□

Then we present below the main result from [CLL25] regarding explicit algorithms that hits any dense language in coAM.

Lemma 115 (See [CLL25], 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 [CLL25] works for only languages with half density, i.e., $|L \cap \{0,1\}^n| \geq 2^n/2$. However, it is easy to amplify the density by parallel repetition, sketched as follows: 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 116. *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 constant k. By Lemma 115, there exists a constant-round single-valued Arthur-Merlin protocol computing $\{x_n\}_{n\in\mathbb{N}}$, such that for infinitely many $n \in \mathbb{N}$, $x_n \in L$. Then by Lemma 114, there exists a 2-round single-valued Arthur-Merlin protocol (P, V) computing $\{x_n\}_{n\in\mathbb{N}}$ in time $2^{\log^{O(k)} n}$ with advice complexity $2^{\log^{1/k} n}$. Let $\alpha_n \in \{0,1\}^{2^{\log^{1/k} n}}$ be the advice for x_n. Then by the conformity of V, we know that

$$\Pr_r[\exists w, V(1^n, \alpha_n, r, w) = x_n] = 1.$$

By the resiliency of V, we have

$$\Pr_r[\forall w, V(1^n, \alpha_n, r, w) \in \{x_n, \perp\}] \geq \frac{2}{3}.$$

Thus for at least $2/3$ of r, there exists w such that $V(1^n, \alpha_n, r, w) = x_n$, and for every w, $V(1^n, \alpha_n, r, w) \in \{x_n, \perp\}$. This exactly matches the definition of rnK, thus we get

$$\mathsf{rnK}^{2^{\log^{O(k)} n}}(x_n) \leq |V| + |\alpha_n| + O(\log n) \leq O(2^{\log^{1/k} n}).$$

By definition of rnKt, we get

$$\mathsf{rnKt}(x_n) \leq \mathsf{rnK}^{2^{\log^{O(k)} n}}(x_n) + \log^{O(k)} n \leq O(2^{\log^{1/k} n}).$$

□

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

Proof of Theorem 9, 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 $2^{\log^d m} = l(m) \le n$.

By assumption, L is dense, which immediately gives L' is dense. Applying Lemma 116, 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 $\mathsf{rnKt}(x_n) \ge 2^{\log^{\gamma/d} n}$;
- By definition of rnKt, trivially, we have $\mathsf{rnKt}(x_n) \le \mathsf{rnKt}(x_n \circ y_n) + O(1) < O(2^{\log^{1/k} n})$.

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. □

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

We prove Theorem 9, Item 3, which begins with some technical tools.

Lemma 117. *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 $|T_s| \le \mathsf{poly}(s)$ and the following properties hold:*

Efficiency: *For all sufficiently large s, we have $\mathsf{rnKt}(T_s) \le 2^{\log^\varepsilon s}$. 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,*

$$\left| \Pr_{x \sim \{0,1\}^s}[C(x) = 1] - \Pr_{x \sim T_s}[C(x) = 1] \right| \le \frac{1}{s}.$$

Next, we show Lemma 117. We need the following tools.

Lemma 118 ([CLL25, Theorem 1.2])**.** *For any language $\mathcal{O} \in \mathsf{AM} \cap \mathsf{coAM}$ and any $0 < \varepsilon < 1$, $(\mathsf{AMEXP} \cap \mathsf{coAMEXP})/2^{n^\varepsilon} \not\subseteq \mathsf{SIZE}^{\mathcal{O}}[2^n/n]$.*

The following lemma gives upper bound on rnKt complexity of truth table decidable in $\mathsf{AMEXP} \cap \mathsf{coAMEXP}$ with short advice.

Lemma 119. *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 51, which handles oracle computed by non-uniform Arthur-Merlin protocol.

Claim 120. *For any time-constructible, non-decreasing $T : \mathbb{N} \to \mathbb{N}$, $T(n) \geq n$, and any $\alpha : \mathbb{N} \to \mathbb{N}$, there exists a constant $c \in \mathbb{N}$ such that for every $\mathcal{O} \in (\mathsf{AMTIME}[T(n)] \cap \mathsf{coAMTIME}[T(n)])/\alpha(n)$, all $x, y \in \{0,1\}^*$, all integers $\ell \geq 1$, and all sufficiently large $t \in \mathbb{N}$ with $t > |x|$, restricting query length ℓ,*

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

Proof Sketch of Claim 120. The proof is basically identical to that of Lemma 51, except that the non-uniform advice needed to simulate the Arthur–Merlin protocols on queries of length at most ℓ is stored in the description, contributing $\alpha(\ell)$ extra bits. ◇

The proof then is a straightforward application of Claim 120. 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 $M^{(\cdot)}$ 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 120, we have

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

as desired. □

As an application of Lemmas 118 and 119, we get the following.

Lemma 121. *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 118 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 119 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 117.

Proof of Lemma 117. The proof combines Lemma 121 and Theorem 28.

Let $\mathcal{O} \in \mathsf{AM} \cap \mathsf{coAM}$. We start by applying Theorem 28 with $\varepsilon := 1/2$, and let c, d be the corresponding constants from the lemma.

Consider Lemma 121 with $\gamma := \varepsilon/2$ and let $\{x_N\}$ be the corresponding sequence of strings.

Let $\mathcal{N} \subseteq \mathbb{N}$ be the infinite set of powers of 2 at which the string x_N from Lemma 121 has $\mathcal{O}$-oracle-circuit complexity at least $N/\log N$. For each $N \in \mathcal{N}$, let $s(N) := \lfloor N^{1/c} \rfloor$, so that $s(N)^c \leq N < (s(N)+1)^c$. We call an integer s *good* if $s = s(N)$ for some $N \in \mathcal{N}$; note that there are infinitely many good s.

For a good s, with $N \in \mathcal{N}$ the corresponding length, the output length of $F(x_N, -)$ is at least $N^{1/c} \geq s$, and we define

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

For every other s, define $T_s = \{0^s\}$; the efficiency bound below applies to every s, while pseudorandomness is claimed only at good s.

We first argue that T_s 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 28.

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 121, 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. Also note that efficiency holds trivially when s is not good.

□

Putting it all together, we are now ready to show Theorem 9, Item 3.

Proof of Theorem 9, 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 117 with $\varepsilon := \gamma/(2(d+1))$, 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.

Let $s_0 \in \mathbb{N}$ be large enough that $\mathsf{rnKt}(T_s) \le 2^{(\log s)^\varepsilon}$ for every $s \ge s_0$. Fix any $s \ge s_0$, any $x \in T_s$, and any block $u := x_{[(i-1)m+1\,:\,im]}$ with $i \in [k]$, where m and k are as computed by D on inputs of length s. Since u can be recovered from T_s together with the indices of x and i, we have

$$\mathsf{rnKt}(u) \;\le\; \mathsf{rnKt}(T_s) + O(\log s) \;\le\; 2^{(\log s)^\varepsilon} + O(\log s).$$

By the definition of m we have $(\log s)^{1/(d+1)} = \frac{c-1}{2}\log m$, and therefore $(\log s)^\varepsilon = O\big(\log^{\gamma/2} m\big)$. Hence, $\mathsf{rnKt}(u) < 2^{\log^\gamma m}$ for all sufficiently large m, so $u \notin L$ and $C^{\mathcal{O}}_m(u) = 0$. As this holds for every block $i \in [k]$, we get $D(x) = 0$ for every $x \in T_s$, and consequently, for all $s \ge s_0$,

$$\Pr_{x\sim T_s}[D(x) = 1] = 0.$$

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

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

In this case, $|L \cap \{0,1\}^m| \ge 2^m/m^c$ by our assumption on L. 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

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

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

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

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 $s \geq \max\{s_0, s_1, s_2\}$, we have

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

Finally, D is an $\mathcal{O}$-oracle circuit of size at most s: it consists of k copies of $C_m^{\mathcal{O}}$ joined by an OR gate, where $\log k = \frac{c-1}{2}\log m$ and $\log^d m$ are both $o(\log s)$. This contradicts the pseudo-randomness of $T = \{T_s\}_{s\in\mathbb{N}}$ in Lemma 117.

□

14 Unconditional Failure of Conditional SoI: Proof of Theorem 10

We say that *conditional SoI holds for* nKt *with error* $e(n)$, if for any strings $x, y, z \in \{0,1\}^*$, we have

$$\mathsf{nKt}(x, y \mid w) \geq \mathsf{nKt}(x \mid y, w) + \mathsf{nKt}(y \mid w) - e(|x| + |y|).$$

Note that in this definition the error bound does not depend on $|w|$. We do not assume random access to w.

We need to impose the following mild conditions on the computational model in the definition of Kolmogorov complexity:[22]

- If a program $M_1(w)$ runs in time t_1, and a program $M_2(w)$ runs in time t_2, then we can run both programs on input w in time $O(t_1 + t_2)$.
- The resulting program has length at most $|M_1| + |M_2| + O(1)$.

Under these mild encoding assumptions, we establish unconditionally that SoI fails in the presence of a conditional string (Theorem 10).

To prove Theorem 10, we will need the following lemma.

Lemma 122 (Efficient Simulation Under Conditional SoI)**.** *If conditional SoI holds for* nKt *with error* $e(n) \geq \log n$*, then for any* $x, w \in \{0,1\}^*$*, we have*

$$\mathsf{nK}^{O(2^{e(|x|)})}(x \mid w) \leq \mathsf{nKt}(x \mid w) + O(e(|x|)).$$

Proof. The proof is essentially the same as that of Lemma 93, but with the addition of an auxiliary conditional string. We present the details here for completeness.

Let $n = |x|$. Let the n bits of x be $x_1x_2\cdots x_n$. For any integers $1 \leq i \leq j \leq n$, we define $x_{[i,j]} = x_i \cdots x_j$. For each $i \leq j$, we define $s_{[i,j]}$ and $t_{[i,j]}$ to be the program size and running time of an optimal nKt program for $x_{[i,j]}$ conditioning on w and $x_{[1,i-1]}$. If there are multiple programs achieving the optimal bound $\mathsf{nKt}(x_{[i,j]} \mid w)$, we take one with the smallest running time.

We let $1 \leq i \leq n$ be the smallest integer such that $t_{[1,i]} \geq 2^{C+1} \cdot 2^{e(n)}$, where $C \geq 1$ is a suitable universal constant defined below. If no such i exists, then the optimal nKt program for x conditioning on w already runs in time $O(2^{e(n)})$, and the proof is finished.

Note that we can assume without loss of generality that $1 < i < n$. Indeed, if C is large enough, using that $e(n) \geq \log n$ it is easy to see that $i > 1$. The case $i = n$ can be handled by an easy modification of the argument above.

[22] These are standard assumptions in Kolmogorov complexity and meta-complexity, and often appear only implicitly in proofs.

Now, by conditional symmetry of information, we have

$$\begin{aligned}\mathsf{nKt}(x \mid w) &\geq \mathsf{nKt}(x_{[1,i]} \mid w) + \mathsf{nKt}(x_{[i+1,n]} \mid x_{[1,i]}, w) - e(n) \\ &= s_{[1,i]} + \log(t_{[1,i]}) + s_{[i+1,n]} + \log(t_{i+1,n}) - e(n). \end{aligned} \tag{35}$$

But on the other hand, since we can print x by simply running the programs in sequence, we get[23]

$$\begin{aligned}\mathsf{nKt}(x \mid w) &\leq s_{[1,i]} + s_{[i+1,n]} + O(1) + \log(O(t_{[1,i]} + t_{[i+1,n]} + n)) \\ &\leq s_{[1,i]} + s_{[i+1,n]} + \log(t_{[1,i]} + t_{[i+1,n]} + n) + C, \end{aligned} \tag{36}$$

for some universal constant $C \geq 1$ that only depends on the computational model. Combining Equations (35) and (36), we get

$$\log(t_{[1,i]} + t_{[i+1,n]} + n) \geq \log(t_{[1,i]}) + \log(t_{[i+1,n]}) - e(n) - C.$$

By taking the exponential of both sides, and using that $e(n) \geq \log n$ and $t_{[1,i]} \geq 2^{C+1} \cdot 2^{e(n)}$, we get

$$t_{[i+1,n]} \leq \frac{1 + \frac{n}{t_{[1,i]}}}{\frac{1}{2^{e(n)+C}} - \frac{1}{t_{[1,i]}}} \leq \frac{2}{\frac{1}{2^{e(n)+C}} - \frac{1}{t_{[1,i]}}} \leq 2^{C+2} \cdot 2^{e(n)}.$$

Hence by combining the programs for $x_{[1,i-1]}$ and $x_{[i+1,n]}$, and by storing the bit x_i and the position i, we can print x in (nondeterministic) time $O(2^{e(n)})$. Therefore, it follows that

$$\begin{aligned}\mathsf{nK}^{O(2^{e(n)})}(x \mid w) &\leq s_{[1,i-1]} + s_{[i+1,n]} + O(\log n) \\ &\leq \mathsf{nKt}(x_{[1,i-1]} \mid w) + \mathsf{nKt}(x_{[i+1,n]} \mid x_{[1,i]}, w) + O(\log n) \\ &\leq \mathsf{nKt}(x_{[1,i]} \mid w) + \mathsf{nKt}(x_{[i+1,n]} \mid x_{[1,i]}, w) + O(\log n) \\ &\leq \mathsf{nKt}(x \mid w) + O(e(n)), \qquad \text{(By conditional SoI for nKt)}\end{aligned}$$

as desired. □

We will use the upper bound on the running time provided by Lemma 122 to significantly speed-up the computation of nKt, leading to a contradiction to an existing complexity lower bound. We need the following result.

Theorem 123 (Hardness of Approximating nKt; Corollary of Theorem 8)**.** *There exists $\varepsilon > 0$ such that the following result holds:*

$$\mathsf{Gap\text{-}MnKtP}[0.6n, 0.9n] \notin \mathsf{DTIME}[2^{\varepsilon \cdot n}].$$

Proof. It is enough to note that if $\mathsf{Gap\text{-}MnKtP}[0.6n, 0.9n] \in \mathsf{DTIME}[2^{\varepsilon \cdot n}]$, then there would be a dense language L containing only strings of nKt complexity at least $0.6n$ such that $L \in \mathsf{NTIME}[2^{\varepsilon \cdot n}] \cap \mathsf{coNTIME}[2^{\varepsilon \cdot n}]$. This directly contradicts the "moreover" part of Theorem 8. □

We are now ready to prove the main result of this section.

[23] Here we use the aforementioned properties of the encoding of programs in the definition of Kolmogorov complexity.

Proof of Theorem 10. Suppose that conditional SoI holds for nKt with error $e(n) \leq \mathsf{polylog}(n)$. We can assume without loss of generality that $e(n) \geq \log n$. We show how to estimate $\mathsf{nKt}(x)$ with additive error $n/10$ in deterministic time $2^{2^{\mathsf{poly}(\log\log(n))}}$. This contradicts the complexity lower bound of Theorem 123.

We set $k = \lceil \frac{n}{100e(n)} \rceil$, and split x into k pieces $x = x_1 \circ x_2 \circ \cdots \circ x_k$, where each x_i is of length at most $100e(n)$. By Lemma 122, we have

$$\mathsf{nK}^{O(2^{e(100e(n))})}(x_i \mid x_1 \circ \cdots \circ x_{i-1}) \leq \mathsf{nKt}(x_i \mid x_1 \circ \cdots \circ x_{i-1}) + O(e(100e(n))).$$

Therefore, for each i, instead of computing $\mathsf{nKt}(x_i \mid x_1 \circ \cdots \circ x_{i-1})$ using an exhaustive search over small nondeterministic programs and assuming access to an NP oracle, we can compute $\mathsf{nK}^{O(2^{e(100e(n))})}(x_i \mid x_1 \circ \cdots \circ x_{i-1})$ deterministically by brute-forcing over all small nondeterministic programs and over all nondeterministic choices, which takes time $2^{100e(n)} \cdot 2^{O(2^{e(100e(n))})}$. The total running time is at most

$$n \cdot 2^{100e(n)} \cdot 2^{O(2^{e(100e(n))})} \leq 2^{2^{\mathsf{poly}(\log\log(n))}}.$$

By conditional SoI with error $e(n)$ (applied k times) and Lemma 122, the total estimation error is at most

$$k \cdot (2e(n) + O(e(100e(n)))) \leq \frac{n}{10}.$$

This contradicts Theorem 123, as alluded to above. □

References

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

[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.

[BF95] Harry Buhrman and Lance Fortnow. Distinguishing complexity and symmetry of information. Technical Report TR-95-11, Department of Computer Science, The University of Chicago, 1995.

[BFNW93] László Babai, Lance Fortnow, Noam Nisan, and Avi Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3:307–318, 1993.

[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.

[BM88] László Babai and Shlomo Moran. Arthur-merlin games: A randomized proof system, and a hierarchy of complexity classes. *J. Comput. Syst. Sci.*, 36(2):254–276, 1988.

[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.

[CLL25] Lijie Chen, Jiatu Li, and Jingxun Liang. Maximum circuit lower bounds for exponential-time Arthur Merlin. In *Symposium on Theory of Computing* (STOC), pages 1348–1358, 2025.

[Coo88] Stephen A. Cook. Short propositional formulas represent nondeterministic computations. *Inf. Process. Lett.*, 26(5):269–270, 1988.

[For00] Lance Fortnow. Time-space tradeoffs for satisfiability. *J. Comput. Syst. Sci.*, 60(2):337–353, 2000.

[GHL+25] Halley Goldberg, Jinqiao Hu, Zhenjian Lu, Jingyi Lyu, and Igor C. Oliveira. Synergies between complexity theory and nondeterministic Kolmogorov complexity. *Electron. Colloquium Comput. Complex.*, TR25-215, 2025.

[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.

[GS86] Shafi Goldwasser and Michael Sipser. Private coins versus public coins in interactive proof systems. In *Symposium on Theory of Computing* (STOC), pages 59–68, 1986.

[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. 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.

[HLO26] Jinqiao Hu, Zhenjian Lu, and Igor C. Oliveira. Hardness of computing nondeterministic Kolmogorov complexity. In *Computational Complexity Conference* (CCC), 2026.

[HLR23] Shuichi Hirahara, Zhenjian Lu, and Hanlin Ren. Bounded relativization. In *Computational Complexity Conference* CCC, pages 6:1–6:45, 2023.

[HMO26] Jinqiao Hu, Yahel Manor, and Igor C. Oliveira. Failure of symmetry of information for randomized computations. In *Symposium on Theory of Computing* (STOC), 2026.

[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.

[HW20] Shuichi Hirahara and Osamu Watanabe. On nonadaptive security reductions of hitting set generators. In *International Workshop on Randomization and Approximation Techniques* (RANDOM), pages 15:1–15:14, 2020.

[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.

[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.

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

[Kab00] Valentine Kabanets. Easiness assumptions and hardness tests: Trading time for zero error. In *Conference on Computational Complexity* (CCC), pages 150–157, 2000.

[KB26] Viacheslav Khrushchev and Bruno Bauwens. Private Communication, June 2026.

[KK26] Valentine Kabanets and Antonina Kolokolova. Chain rules for time-bounded Kolmogorov complexity. In *Symposium on Theory of Computing* (STOC), 2026.

[KvM02] Adam R. Klivans and Dieter van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM J. Comput.*, 31(5):1501–1526, 2002.

[Lee06] Troy Lee. *Kolmogorov complexity and formula lower bounds*. PhD thesis, University of Amsterdam, 2006.

[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.

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

[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.

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

[Oli19] Igor C. Oliveira. Randomness and intractability in Kolmogorov complexity. In *International Colloquium on Automata, Languages, and Programming* (ICALP), pages 32:1–32:14, 2019.

[OS17] Igor C. Oliveira and Rahul Santhanam. Pseudodeterministic constructions in subexponential time. In *Symposium on Theory of Computing* (STOC), pages 665–677, 2017.

[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.

[Rud97] Steven Rudich. Super-bits, demi-bits, and NP/qpoly-natural proofs. In *International Workshop on Randomization and Approximation Techniques* (RANDOM), pages 85–93, 1997.

[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.

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

[SU06] Ronen Shaltiel and Christopher Umans. Pseudorandomness for approximate counting and sampling. *Comput. Complex.*, 15(4):298–341, 2006.

[Sud97] Madhu Sudan. Decoding of reed solomon codes beyond the error-correction bound. *J. Complex.*, 13(1):180–193, 1997.

[TV07] Luca Trevisan and Salil P. Vadhan. Pseudorandomness and average-case complexity via uniform reductions. *Computational Complexity*, 16(4):331–364, 2007.

[Yap83] Chee-Keng Yap. Some consequences of non-uniform conditions on uniform classes. *Theor. Comput. Sci.*, 26:287–300, 1983.

[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 The Role of Adaptivity I: Ronneburger's Technique

[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 42 and Corollary 44). 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 sketch 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 124. *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 125. *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 124.

Proof of Theorem 124. Applying Lemma 125 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 an optimized variant of Corollary 42 with $c = 1$,[24]

$$\mathsf{Kt}^{\|_2 \mathsf{SAT}}(x, y) < \mathsf{nKt}(x) + \mathsf{nKt}(y \mid x) - \frac{n}{4},$$

as desired. □

B The Role of Adaptivity II: Nonadaptive Reconstruction, Explicit Constructions, and Failure of SoI

In order to highlight the relevance of (non)adaptivity and the challenges that it poses, we show in this section that if the pseudorandom generator construction of Trevisan and Vadhan [TV07] can be modified to admit a *non-adaptive* reconstruction routine, then SoI fails for rnKt.

To this end, we first observe that the algorithm of Oliveira and Santhanam [OS17], which builds on [TV07] to obtain pseudodeterministic explicit constructions for dense BPP-properties[25], (a) is PSPACE-relativizing (in the sense of [HLR23]), and (b) can be extended to work for dense pr-BPP-properties with the aid of small advice.

[24] This can be obtained by reducing a nondeterministic time-t computation to a SAT instance of size $t \cdot \mathsf{poly}(\log t)$, where the reduction runs in time $t \cdot \mathsf{poly}(\log t)$ (see [Coo88, For00]).

[25] [OS17] state their result for dense P-properties; however, the argument extends straightforwardly to dense BPP-properties.

Theorem 126. *For any $O \in \mathsf{PSPACE}$, any dense $\mathsf{pr\text{-}BPP}^O$ property P, and any $\gamma > 0$, there exists an infinitely-often pseudodeterministic explicit construction for P that runs in time 2^{n^γ}, given oracle access to O and n^γ bits of advice.*[26]

Proof Sketch. The proof follows closely the argument of [OS17]. We begin by constructing a pseudodeterministic PRG.

Claim 127. *Let $O \in \mathsf{PSPACE}$. For every $\varepsilon > 0$ and $b \in \mathbb{N}$, there exists a sequence $\{G_n\}_{n\in\mathbb{N}}$, where*

$$G_n : \{0,1\}^{n^\varepsilon} \to \{0,1\}^{n^b},$$

computable in pseudodeterministic time $2^{O(n^\varepsilon)}$ with oracle access to O, such that the following holds. For every n^b-time samplable distribution family $\{\mathcal{C}_n\}_{n\in\mathbb{N}}$ of Boolean O-oracle circuits, there exist infinitely many n such that, with probability at least $1 - n^{-b}$ over $C \sim \mathcal{C}_n$, the generator G_n (n^{-b})-fools C.

Proof of Claim 127. We consider two cases.

Case 1: $\mathsf{PSPACE} \subseteq \mathsf{BPP}^O$. In this case, by a standard padding argument,

$$\mathsf{DSPACE}\left[2^{O(n)}\right] \subseteq \mathsf{BPE}^O.$$

Also, by a diagonalization argument, there exists a language

$$L \in \mathsf{DSPACE}\left[2^{O(n)}\right]$$

such that for all but finitely many input lengths n, the language L does not admit O-oracle circuits of size $2^{0.9n}$. Indeed, using $2^{O(n)}$ space, one can compute the lexicographically first truth table of length 2^n that is not computed by any O-oracle circuit of size $2^{0.9n}$. This is possible since $O \in \mathsf{PSPACE}$, allowing such circuits to be simulated within $2^{O(n)}$ space.

By the containment above, we obtain $L \in \mathsf{BPE}^O$. Such a hard language yields a pseudorandom generator with logarithmic seed length, computable in pseudodeterministic polynomial time with oracle access to O, and secure against non-uniform polynomial-size O-oracle circuits [IW97, KvM02, OS17].

Case 2: $\mathsf{PSPACE} \not\subseteq \mathsf{BPP}^O$. In this case, the desired PRG follows directly from the constructions of [IW01, TV07] (see Theorem 29). ◇

Now suppose, toward a contradiction, that the statement of the theorem is false. Then there exist $O \in \mathsf{PSPACE}$, a dense $\mathsf{pr\text{-}BPP}^O$ property $P = (\mathcal{YES}, \mathcal{NO})$, and some $\gamma > 0$ such that for every algorithm A, for all sufficiently large n, and for every advice string $a_n \in \{0,1\}^{n^\gamma}$, if $A(a_n)$ runs in pseudodeterministic time 2^{n^ε}, then its canonical output lies in $\mathcal{NO}$.

Let D be a probabilistic polynomial-time O-oracle algorithm such that, except with exponentially small probability over its internal randomness, D accepts every string in $\mathcal{YES}$ and rejects every string in $\mathcal{NO}$. Such a distinguisher can be obtained via standard error reduction and a union bound. We may equivalently view D as a distribution over O-oracle circuits obtained by fixing its internal randomness. Let $A(a_n)$ be the procedure $G(a_n)$ computing the PRG from Claim 127 with $\varepsilon = \gamma/2$ and seed string fixed to a_n.

[26] In other words, if $P = (\mathcal{YES}_n, \mathcal{NO}_n)$, then there are infinitely many values of n such that we can pseudodeterministically construct in subexponential time with a small amount of advice a string $x_n \notin \mathcal{NO}_n$.

By the assumptions above, with all but exponentially small probability over the choice of D, it holds that for all but finitely many n,

$$\Pr_A\big[D(A(a_n)) = 0\big] \geq \tfrac{2}{3}$$

for every $a_n \in \{0,1\}^{n^\varepsilon}$.

On the other hand, since P is dense, we have

$$\Pr_{x\sim\{0,1\}^n}[D(x) = 1] \geq \tfrac{1}{2}.$$

Thus, D distinguishes the output of the PRG from the uniform distribution, contradicting Claim 127. □

As a corollary of Theorem 126, we obtain the following result.

Corollary 128. *For any $O \in \mathsf{NP}$, any dense $\mathsf{pr\text{-}BPP}^O$ property P, and any $\gamma > 0$, there exists an infinitely-often pseudodeterministic explicit construction for P that runs in time 2^{n^γ}, given oracle access to O and n^γ bits of advice.*

It is not difficult to extend the proof of this result to the setting of quasi-polynomial time, and it is natural to ask whether one can further modify the above explicit construction so that the oracle access in both the property and the construction algorithm is *non-adaptive*. By Theorem 6, this would allow us to refute SoI for rnKt, as explained below.

Proposition 129. *Suppose that for any $O \in \mathsf{NP}$, any dense $\mathsf{pr\text{-}BPTIME}[n^{\mathsf{poly}(\log n)}]^{||O}$ property P, and any $\gamma > 0$, there exists an infinitely-often pseudodeterministic explicit construction for P that runs in time 2^{n^γ}, given nonadaptive oracle access to O and n^γ bits of advice. Then symmetry of information does not hold for rnKt with error $e(n) = \mathsf{poly}(\log n)$.*

Proof Sketch. By Theorem 6, it is enough to show that the assumption of Proposition 129 implies the following statement:

> For any promise problem $P = (\mathcal{YES}, \mathcal{NO}) \in \mathsf{pr\text{-}BPTIME}[n^{\mathsf{poly}(\log n)}]^{||\mathsf{NP}}$ such that $|\mathcal{YES} \cap \{0,1\}^n| \geq 2^{n-1}$, there exists $t(\cdot), \ell(\cdot) : \mathbb{N} \to \mathbb{N}$ where $\ell(n) + \log t(n) < n - \mathsf{poly}(\log(n))$ for each n, and an algorithm $A(1^n, \cdot)$ such that, for infinitely many n, there exists advice $\alpha_n \in \{0,1\}^{\ell(n)}$ and a string $x_n \in \{0,1\}^n \setminus \mathcal{NO}$, such that
>
> $$\mathbf{Pr}_{r\sim\{0,1\}^{t(n)}}\begin{bmatrix}\forall w \in \{0,1\}^{t(n)}, A(1^n, \alpha_n, w, r) \text{ outputs } x \text{ or } \perp \text{ in } t(n) \text{ steps;}\\ \exists w \in \{0,1\}^{t(n)}, A(1^n, \alpha_n, w, r) \text{ outputs } x \text{ in } t(n) \text{ steps}\end{bmatrix} \geq \frac{2}{3}.$$

To see this, let $P = (\mathcal{YES}, \mathcal{NO})$ be any such property, and consider the pseudodeterministic algorithm $B^{||\mathsf{NP}}(1^n, \beta_n, r)$ for P given by the hypothesis of Proposition 129, where β_n denotes the correct advice string on a good input length n, and r denotes its randomness. To establish the desired statement, it is sufficient to obtain from $B^{||\mathsf{NP}}(1^n, \beta_n, r)$ an algorithm $A(1^n, \alpha_n, w, r)$ as above, where A also gets a nondeterministic string w and a bounded-length string β_n extending α_n, but A does not have oracle access to $||\mathsf{NP}$.

From the perspective of time-bounded Kolmogorov complexity, this is essentially equivalent to a proof that $\mathsf{rK}^{||\mathsf{NP}}$ can be simulated by rnK (with some overhead), which is the randomized analogue of the simulation of $\mathsf{Kt}^{||\mathsf{NP}}$ by nKt obtained by storing as advice the number of positive answers given by the oracle. It is possible to implement the desired simulation in the randomized setting using the techniques described in Section 8.2.2. □

A natural attempt is to modify the proof of Claim 127 by considering two different cases: $\mathsf{PSPACE} \subseteq \mathsf{BPP}^{\|O}$ and $\mathsf{PSPACE} \not\subseteq \mathsf{BPP}^{\|O}$. For the first case, by inspecting the proof and the PRG constructions of [IW97, KvM02, OS17], one can still obtain a PRG that is secure against polynomial-size O-oracle circuits and computable in pseudodeterministic polynomial time using only *non-adaptive* oracle access to O. Such a PRG would yield a pseudodeterministic explicit construction for dense $\mathsf{pr\text{-}BPP}^{\|O}$ properties, with running time 2^{n^γ} given *non-adaptive* oracle access to O and n^γ bits of advice.

Therefore, it remains to handle the second case. More specifically:

Problem 130. Show a non-adaptive version of Theorem 29. That is, prove the following. Let $O \in \mathsf{NP}$. Assume $\mathsf{PSPACE} \not\subseteq \mathsf{BPP}^{\|O}$. Then for every $\varepsilon > 0$ and $b \in \mathbb{N}$, there exists a sequence $\{G_n\}_{n\in\mathbb{N}}$, where

$$G_n : \{0,1\}^{n^\varepsilon} \to \{0,1\}^{n^b}$$

is computable in time $2^{O(n^\varepsilon)}$, such that the following holds. For every n^b-time samplable distribution family $\{\mathcal{C}_n\}_{n\in\mathbb{N}}$ of Boolean O-oracle circuits with non-adaptive oracle queries, there exist infinitely many n such that, with probability at least $1 - n^{-b}$ over $C \sim \mathcal{C}_n$, the generator G_n (n^{-b})-fools C.

Recall that the PRG in Theorem 29 is obtained via the hardness–vs.–randomness paradigm. At a high level, one takes the truth table of a (presumably) hard PSPACE-complete language L with suitable structural properties [TV07], and plugs it into a standard PRG construction (e.g., [NW94]) that admits a reconstruction procedure. Roughly speaking, the reconstruction property asserts that if there exists an efficient distinguisher (e.g., a $\mathsf{pr\text{-}BPP}^{\|O}$ algorithm) that breaks the security of the PRG, then one can use this distinguisher to obtain a circuit for computing L, thereby contradicting the assumed hardness of L.

However, to make this reconstruction *uniform*, the reconstruction algorithm must also be able to answer oracle queries to the language L itself. To address this issue, one exploits the *downward self-reducibility* of the special language L. Informally, this property allows one to compute L on inputs of a given length by making oracle queries to L on strictly smaller input lengths. Using this, one can iteratively construct a sequence of bounded-size circuits, starting from small input lengths and building up to the desired length.

Now suppose that L is hard against $\mathsf{pr\text{-}BPP}^{\|O}$. In order to contradict this assumed hardness, one must ensure that the above reconstruction procedure uses only non-adaptive queries to O. However, the oracle queries arising from this iterative construction are inherently sequential: at the i-th stage, we build a bounded-size circuit C_i that computes L on inputs of length i, where the construction of C_i, while making non-adaptive queries to O, requires the algorithm to query the previously constructed circuit C_{i-1}. Consequently, the overall pattern of oracle access is necessarily adaptive. This presents a fundamental obstacle to obtaining a non-adaptive version of the construction.

Finally, we mention that existing results strongly suggest that if a non-adaptive reconstruction procedure exists, it must employ non-black-box techniques (unless $\mathsf{PSPACE} \subseteq \mathsf{AM} \cap \mathsf{coAM}$). We refer to [HW20] and reference therein for more details and related discussions.

ECCC ISSN 1433-8092
https://eccc.weizmann.ac.il