

The Gap Between Average-Case and Worst-Case Complexity

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

October 13, 2026

Abstract

Understanding the average-case complexity of NP is a central goal in complexity theory and cryptography. A necessary milestone is to understand the corresponding average-case versus worst-case question for the polynomial hierarchy (PH). In a celebrated result, Hirahara [Hir21] proved that if every distributional problem in PH is easy on average, then every language in PH admits worst-case algorithms running in time $2^{O(n/\log n)}$. Remarkably, although the statement of Hirahara’s theorem makes no reference to meta-complexity, all known proofs appear to use time-bounded Kolmogorov complexity and its associated meta-computational problems in an essential way.

We show that the reliance on techniques and perspectives from meta-complexity is not simply a coincidence. Building on previous work of Hirahara [Hir20] and on a recent result of Hu, Lu, and Oliveira [HLO26], we prove that relating the average-case and worst-case complexities of PH is in fact *equivalent* to a purely meta-computational implication: efficient algorithms for a *standard gap version* of MINpKT with PH oracle access imply efficient algorithms for a *mild gap version*. Crucially, in these promise problems the standard gap separates a YES instance with time bound t from a NO instance with time bound $\text{poly}(t)$, whereas the mild gap keeps the YES threshold t and moves the NO threshold to $t + \text{poly}(n)$.

Together with prior work, this equivalence points to the study of *gap reductions* as a potentially promising route toward understanding the relation between average-case and worst-case complexity. We make progress by proving that the gap in running time can be eliminated in the average-case setting. Finally, we prove that in the worst-case setting any efficient reduction between the two computational problems must use nonrelativizing techniques.

^{*}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

Contents

1	Introduction	3
1.1	Overview	3
1.2	Results	4
1.3	Techniques	5
1.4	Related Work	10
2	Preliminaries	12
2.1	Computational Models	12
2.2	Time-Bounded Kolmogorov Complexity	13
2.3	Meta-Complexity	14
2.4	Useful Tools	15
2.5	Average-Case Complexity	17
3	Proof of the Equivalence Theorem	19
3.1	Characterizing Easiness of NP via Easiness of Mild-Gap-MINpKT ^{PH}	19
3.1.1	The Easy Direction	20
3.1.2	The Hard Direction	20
3.2	Worst-Case Easiness of Gap-MINpKT ^{PH} from Average-Case Easiness of PH	27
3.3	Average-Case Easiness of PH from Worst-Case Easiness of Gap-MINpKT ^{PH}	27
3.3.1	Technical Tools	27
3.3.2	Proof of Lemma 3.13	33
4	The Deterministic Case	34
5	Equivalence of Gap and No-Gap Hardness in the Average-Case Setting	35
5.1	The Zero-Error Case	35
5.2	The Two-Sided-Error Case	36
6	A Relativization Barrier	38

1 Introduction

1.1 Overview

Understanding the average-case complexity of NP is a fundamental research direction in complexity theory. A central problem in this area asks whether worst-case hardness of NP can be transformed into average-case hardness of NP against efficient algorithms. This question is particularly relevant to the complexity-theoretic foundations of cryptography, where one would ideally like to base cryptographic primitives on worst-case intractability assumptions about NP problems. We refer to the survey of Bogdanov and Trevisan [BT06] for background on average-case complexity and its connections to cryptography.

Formally, before resolving the average-case complexity of NP, one must first understand the analogous question for the Polynomial Hierarchy (PH).¹ In the language of Impagliazzo’s five worlds [Imp95], the desired statement is the exclusion of PH Heuristica:

$$\text{DistPH} \subseteq \text{AvgBPP} \implies \text{PH} \subseteq \text{BPP}. \quad (1)$$

Equivalently, if every problem in PH is easy on every polynomial-time samplable distribution, then every problem in PH is easy in the worst case.

A few years ago, a breakthrough result of Hirahara [Hir21] (see also [GKLO22]) showed that a weaker worst-case conclusion follows from the average-case easiness of PH:

$$\text{DistPH} \subseteq \text{AvgBPP} \implies \text{PH} \subseteq \text{BPTIME}[2^{O(n/\log n)}]. \quad (2)$$

Hirahara’s Theorem (Equation (2)) offered the first significant progress on this problem after a long gap. Note that the theorem itself is a statement about complexity classes and their relation. Nevertheless, Hirahara’s argument seems to employ Meta-Complexity in an essential way, through Kolmogorov complexity and related compression problems.

Recall that meta-complexity studies the complexity of computational tasks whose inputs are themselves computational objects, such as strings, or truth tables. A central example is time-bounded Kolmogorov complexity, which measures the length of the shortest program that outputs a given string within a prescribed time bound. The associated meta-computational problem asks for the time-bounded Kolmogorov complexity of a given input string. For more information about meta-complexity, we refer to [Sim23]. For a modern treatment of time-bounded Kolmogorov complexity and its extensions to randomized computation, see [LO22].

Subsequent works have provided simplifications to the original proof of [Hir21] and extensions of the result [GK22, GKLO22, CHV22]. All of these developments continue to rely on meta-complexity. We refer to [Hir22a] for a comprehensive treatment of the meta-complexity approach to average-case complexity.

Motivated by these developments and the intriguing role of meta-complexity, we consider the following natural question:

*Is Meta-Complexity formally necessary for understanding the relation between
average-case and worst-case complexity?*

As our main contribution, we give a positive answer to this question. We show that excluding PH Heuristica (Equation (1)) is equivalent to proving a *gap reduction* between two worst-case meta-computational tasks. This gives a precise sense in which meta-complexity is inevitable: any proof excluding PH Heuristica can be converted into a proof of the gap reduction from the standard gap meta-complexity problem to the mild-gap meta-complexity problem; conversely, any proof of this gap reduction excludes PH Heuristica.

¹In other words, the implication $\text{DistPH} \subseteq \text{AvgBPP} \implies \text{PH} \subseteq \text{BPP}$ follows from the implication $\text{DistNP} \subseteq \text{AvgBPP} \implies \text{NP} \subseteq \text{BPP}$.

1.2 Results

We now state our results in more detail.²

Average-Case vs Worst-Case Complexity as Meta-Complexity. We use $\text{pK}^{t,\mathcal{O}}(x)$ to denote probabilistic t -time bounded Kolmogorov complexity with oracle access to $\mathcal{O}$. Informally, $\text{Gap-MINpKT}^{\mathcal{O}}$ asks one to distinguish strings with small $\text{pK}^{t,\mathcal{O}}$ complexity from strings whose $\text{pK}^{\text{poly}(t),\mathcal{O}}$ complexity is larger by a logarithmic term. This is the standard gap setting for several time-bounded Kolmogorov complexity estimation problems in meta-complexity. In contrast, the mild version $\text{Mild-Gap-MINpKT}^{\mathcal{O}}$ asks for the analogous distinction when the larger time bound is only $t + \text{poly}(n)$ rather than $\text{poly}(t)$. We refer to [Section 2](#) for the formal notions. [Figure 1](#) informally describes the input instances of the two gap problems and their inclusion relations.

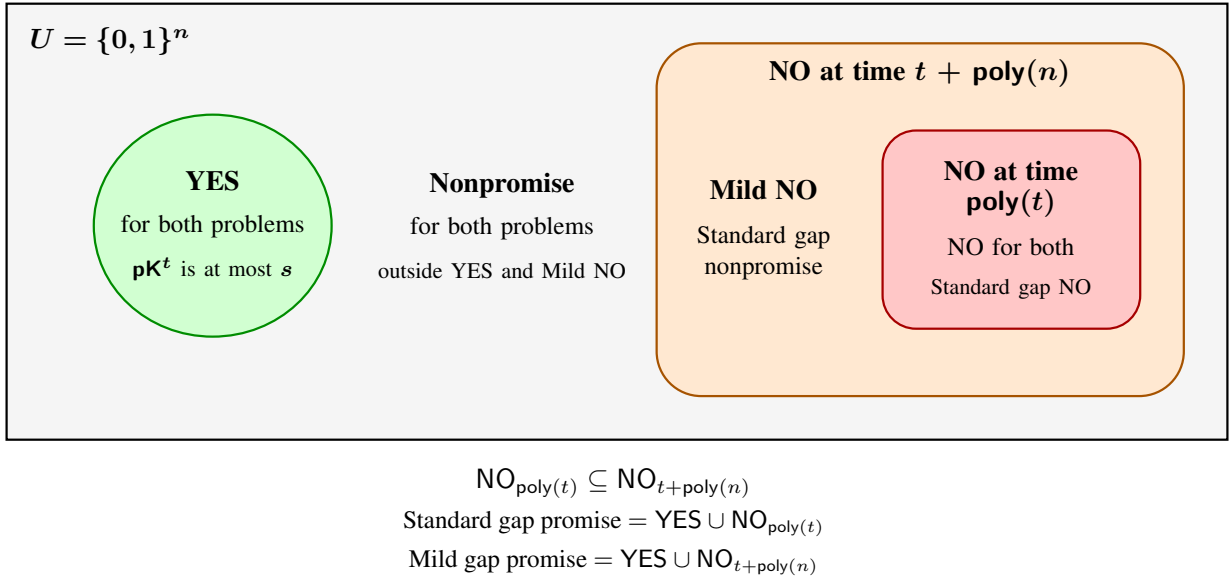


Figure 1: Instance sets for the standard and mild gap problems for fixed description length bound s and time bound t . Under $\text{poly}(t) \geq t + \text{poly}(n)$, monotonicity of time-bounded probabilistic Kolmogorov complexity gives $\text{pK}^{t+\text{poly}(n),\mathcal{O}}(x) \geq \text{pK}^{\text{poly}(t),\mathcal{O}}(x)$. Consequently, $\text{NO}_{\text{poly}(t)} \subseteq \text{NO}_{t+\text{poly}(n)}$.

For an oracle-dependent problem A such as Gap-MINpKT or Mild-Gap-MINpKT , the notation $A^{\text{PH}} \in \text{prBPP}$ means that the corresponding promise problem is in randomized polynomial time for every oracle language $\mathcal{O} \in \text{PH}$.

Theorem 1.1 (Main Equivalence). *The following statements are equivalent:*

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

Note that the second item refers to the *worst-case* complexities of the corresponding meta-computational problems. Thus, the average-case versus worst-case complexity of PH is equivalent to a comparison between two approximation regimes for probabilistic time-bounded Kolmogorov complexity with PH oracle access. Roughly speaking, the standard gap problem captures average-case complexity, while the mild gap problem

²A preliminary version of a subset of these results appeared as part of the unpublished ECCC preprint [\[GHL⁺25\]](#).

captures worst-case complexity. Establishing a gap reduction, namely that an efficient algorithm for the former yields an efficient algorithm for the latter, is exactly what is needed to rule out PH Heuristica.³ The proof of [Theorem 1.1](#) appears in [Section 3](#).

There is a deterministic analogue of the characterization: replacing $\text{pK}^{t,\text{PH}}$ by deterministic $\text{K}^{t,\text{PH}}$ and replacing randomized algorithms by deterministic algorithms gives the corresponding equivalence between

$$\text{DistPH} \subseteq \text{AvgP} \implies \text{PH} \subseteq \text{P}$$

and the implication from the deterministic standard gap version of MINKT^{PH} to its deterministic mild gap version. We refer to [Section 4](#) for the details.

Eliminating the Gap in the Average-Case Setting. In our next result, we show that the implication in [Theorem 1.1](#) (Item 2) can be established in the average-case setting. Indeed, we are able to prove in this case that the easiness of $\text{Gap-MINpKT}^{\text{PH}}$ implies the easiness of $\text{MINpKT}^{\text{PH}}$, where there is *no gap* in the running time when comparing positive and negative instances. We consider both the zero-error setting of AvgBPP and the two-sided error setting of HeurBPP .

Theorem 1.2 (Average-Case Gap Reductions). *The following two equivalences hold:*

- “ $\text{Gap-MINpKT}^{\text{PH}} \in \text{AvgBPP}$ ” $\iff$ “ $\text{MINpKT}^{\text{PH}} \in \text{AvgBPP}$ ”.
- “ $\text{Gap-MINpKT}^{\text{PH}} \in \text{HeurBPP}$ ” $\iff$ “ $\text{MINpKT}^{\text{PH}} \in \text{HeurBPP}$ ”.

In other words, the time gap between instances can be eliminated in these average-case settings. The necessary definitions are presented in [Section 2.5](#), while the proof of [Theorem 1.2](#) appears in [Section 5](#).

Limits of Relativizing Techniques. The natural next step is to obtain a gap reduction in the worst case, which by [Theorem 1.1](#) is both necessary and sufficient to exclude PH Heuristica. We finish with a proof that the implication between the two meta computational tasks in [Theorem 1.1](#) (Item 2) cannot be established by relativizing techniques alone.

Theorem 1.3 (Relativization Barrier). *There is an oracle $\mathcal{O}$ such that*

$$\text{Gap-MINpKT}^{\text{PH}^{\mathcal{O}},\mathcal{O}} \in \text{prBPP}^{\mathcal{O}} \quad \text{but} \quad \text{Mild-Gap-MINpKT}^{\text{PH}^{\mathcal{O}},\mathcal{O}} \notin \text{prBPP}^{\mathcal{O}}.$$

Thus, in a relativized world, the standard gap problem can be easy even while the mild gap problem remains hard. We refer to [Section 6](#) for the proof of this result and related discussions.

1.3 Techniques

In this section, we provide an overview of our proofs and techniques.

Main Equivalence. The proof of [Theorem 1.1](#) builds heavily on the approaches and techniques from [\[Hir20\]](#) and [\[HLO26\]](#). The result follows from two correspondences between *structural complexity* and *meta-complexity*. As explained below, the average-case easiness of PH is captured by the complexity of the standard gap problem, where a negative instance is allowed to use a much larger time bound $\text{poly}(t)$. In contrast, the worst-case easiness of PH is captured by the complexity of the mild gap problem, where a negative instance only receives an additive increase of the form $t + \text{poly}(n)$. [Figure 2](#) provides an overview.

³For completeness, we spell out here a reduction that is sufficient for excluding PH Heuristica. Fix an arbitrary oracle $\mathcal{O} \in \text{PH}$. Exhibit a polynomial p and an oracle $\mathcal{O}' \in \text{PH}$ such that, for every large enough polynomial q , separating $\text{pK}^{t,\mathcal{O}}(x) \leq s$ vs $\text{pK}^{t+p(|x|),\mathcal{O}}(x) \geq s + \log p(|x|)$ reduces in randomized polynomial time to separating $\text{pK}^{t,\mathcal{O}'}(x) \leq s$ vs $\text{pK}^{q(t),\mathcal{O}'}(x) \geq s + \log q(|x|)$.

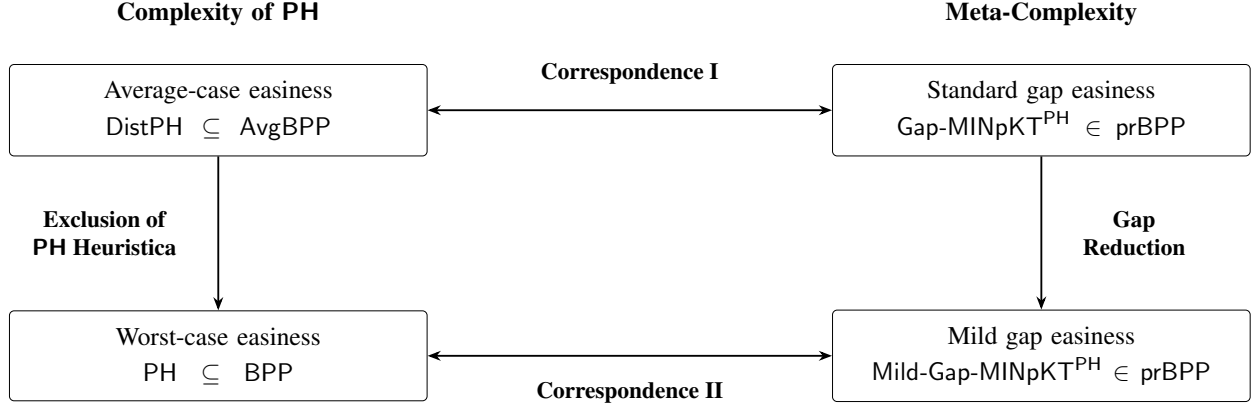


Figure 2: Relating the average-case and worst-case complexities of the polynomial hierarchy (PH) is equivalent to a question from meta-complexity: Is there a gap reduction for computing pK^{PH} ?

The first correspondence identifies average-case easiness of PH with easiness of the standard gap problem:

$$\text{DistPH} \subseteq \text{AvgBPP} \iff \text{Gap-MINpKT}^{\text{PH}} \in \text{prBPP}.$$

This correspondence is a randomized extension of Hirahara’s characterization of $\text{DistPH} \subseteq \text{AvgP}$ by deterministic worst-case meta-complexity [Hir20]. We note that in the deterministic setting, a substantial part of the argument from [Hir20] is the construction of an explicit pseudorandom generator from the assumed easiness of $\text{Gap-MINKT}^{\text{PH}}$. This is used not just to conclude $\text{DistPH} \subseteq \text{AvgP}$ but also as a key technical element of the proof, which manipulates the (deterministic) time-bounded Kolmogorov complexity measure $\text{K}^{t,\text{PH}}$. By working directly with probabilistic Kolmogorov complexity $\text{pK}^{t,\text{PH}}$ and with randomized procedures, we are able to bypass this specific pseudorandom generator construction.⁴ Moreover, we provide an induction-based proof that differs somewhat from, and simplifies, the argument of [Hir20]. We briefly outline the proof below.

The forward direction follows from a worst-case-to-average-case reduction for estimating time-bounded Kolmogorov complexity. If every language in PH is easy on every samplable distribution, then $\text{MINKT}^{\mathcal{O}}$ is easy on average for every $\mathcal{O} \in \text{PH}$. This average-case easiness can then be upgraded to a worst-case algorithm for $\text{Gap-MINpKT}^{\mathcal{O}}$ using standard techniques from meta-complexity.

We now discuss the backward direction. Fix $k_0 \geq 1$ and a language $\mathcal{O}$ that is complete for $\Sigma_{k_0}^{\text{P}}$. Assuming that $\text{Gap-MINKT}^{\mathcal{O}} \in \text{prBPP}$, our goal is to decide any language in Σ_k^{P} , for $k \leq k_0$, on input x in time roughly

$$2^{\text{pK}^{\text{poly}(n),\mathcal{O}}(x) - \text{K}(x)}. \quad (3)$$

Since $\text{K}(x)$ is close to n for a typical $x \in \{0,1\}^n$, we have $\text{pK}^{\text{poly}(n),\mathcal{O}}(x) - \text{K}(x) \leq O(\log n)$ on average. Consequently, Equation (3) yields an average-case polynomial-time algorithm for PH.

To establish Equation (3), standard techniques from meta-complexity imply that the assumption $\text{Gap-MINKT}^{\mathcal{O}} \in \text{prBPP}$ yields a symmetry-of-information statement of the following form: for every $x, y \in \{0,1\}^*$ and every t ,

$$\text{pK}^{\text{poly}(t)}(y \mid x) \lesssim \text{pK}^{t,\mathcal{O}}(x, y) - \text{pK}^{\text{poly}(t),\mathcal{O}}(x).$$

Now let x be an instance of the Σ_k^{P} language that we wish to decide, and, if x is a YES instance, let y^* denote the lexicographically smallest witness corresponding to the first existential quantifier. Taking t to be

⁴As in [Hir20], the remaining steps of the proof still rely on techniques from computational pseudorandomness and on the reconstruction procedure of another pseudorandom generator.

a sufficiently large polynomial, the symmetry-of-information inequality gives

$$\begin{aligned} \text{pK}^{\text{poly}(t)}(y^* \mid x) &\lesssim \text{pK}^{t, \mathcal{O}}(x, y^*) - \text{pK}^{\text{poly}(t), \mathcal{O}}(x) =: s \\ &\lesssim \text{pK}^{2t, \mathcal{O}}(x) - \text{K}(x). \end{aligned}$$

The second inequality above uses the fact that y^* can be generated efficiently from x given oracle access to $\mathcal{O}$, since $\mathcal{O}$ is complete for a sufficiently high level of PH.

A natural next step is therefore to enumerate all programs of length at most s , run them on input x , and consider their outputs as candidate witnesses y . The difficulty is that checking whether a candidate y is a valid witness requires solving a Π_{k-1}^P problem on the instance (x, y) . At first sight, one might try to invoke the inductive hypothesis directly. However, doing so would only give a running time bounded by

$$2^{\text{pK}^{\text{poly}(n), \mathcal{O}}(x, y) - \text{K}(x, y)} \lesssim 2^{\text{pK}^{\text{poly}(n), \mathcal{O}}(x, y) - \text{K}(x)},$$

whereas we need the overall running time to remain within

$$2^{\text{pK}^{\text{poly}(n), \mathcal{O}}(x) - \text{K}(x)}.$$

For an arbitrary candidate y , there is no reason for $\text{pK}^{\text{poly}(n), \mathcal{O}}(x, y)$ to be comparable to $\text{pK}^{\text{poly}(n), \mathcal{O}}(x)$.

The key observation is that we only need to guarantee successful verification of y^* , when such a witness exists. For this particular witness, we have

$$\text{pK}^{\text{poly}(n), \mathcal{O}}(x, y^*) \lesssim \text{pK}^{\text{poly}(n), \mathcal{O}}(x),$$

because y^* can be recovered from x in polynomial time using oracle access to $\mathcal{O}$. Thus, before invoking the inductive-hypothesis algorithm on a candidate y , we can first test whether

$$\text{pK}^{\text{poly}(n), \mathcal{O}}(x, y) \lesssim \text{pK}^{\text{poly}(n), \mathcal{O}}(x).$$

Since $\text{Gap-MINKT}^{\text{PH}} \in \text{prBPP}$, this test can be implemented efficiently (up to the usual approximation slack). Candidates that fail the test are discarded. Crucially, y^* passes the test, while every candidate that passes can be verified using the inductive hypothesis within time

$$2^{\text{pK}^{\text{poly}(n), \mathcal{O}}(x) - \text{K}(x)}$$

up to the suppressed polynomial factors. This gives the desired inductive running-time bound.

As alluded to above, the second correspondence identifies the worst-case easiness of PH with the easiness of the mild gap problem:

$$\text{PH} \subseteq \text{BPP} \iff \text{Mild-Gap-MINpKT}^{\text{PH}} \in \text{prBPP}.$$

The easy direction works as follows: it is possible to show that the mild gap problem has an Arthur-Merlin protocol relative to the oracle, and this allows us to prove that if $\text{PH} \subseteq \text{BPP}$ then $\text{Mild-Gap-MINpKT}^{\text{PH}} \in \text{prBPP}$.

The hard direction relies on the witness compression framework of [HLO26]. Suppose $\text{Mild-Gap-MINpKT}^{\text{SAT}}$ is easy. Since $\text{PH} \subseteq \text{BPP}$ follows from $\text{NP} \subseteq \text{BPP}$, it is enough to establish the latter inclusion. Crucially, we derive from the easiness assumption the following consequence: For every NP verifier, every yes instance x , and every witness y for x ,

$$\text{pK}^{\text{poly}(n)}(y \mid x) \leq \log |W_x| + O(\log n), \tag{4}$$

where W_x is the witness set of x . Therefore, by choosing a random short program and random coins, one samples an accepting witness with inverse polynomial probability. Repetition then gives a BPP-algorithm for every NP-language, as desired.

An important ingredient in the proof of Equation (4) in [HLO26] is that probabilistic nondeterministic Kolmogorov complexity (pnK^t) admits a *language compression theorem*: witnesses for an NP instance can be described using essentially the logarithm of the number of witnesses. Our first observation is that $\text{pK}^{t,\text{SAT}}$ is strong enough to inherit this compression property, since SAT oracle access can simulate the nondeterministic choices appearing in pnK^t with only polynomial overhead. Note that this is not sufficient to prove Equation (4), where we need an upper bound on pK^t instead of $\text{pK}^{t,\text{SAT}}$.

The remaining task is precisely to remove the SAT oracle from the description of y given x . The key ingredient here is that the assumption $\text{Mild-Gap-MINpKT}^{\text{PH}} \in \text{prBPP}$ provides an ordinary probabilistic polynomial-time algorithm A for testing whether a string has low $\text{pK}^{t,\text{SAT}}$ complexity. Although the complexity measure tested by A allows its programs to query SAT, the algorithm A itself has no oracle access. Consequently, if A can be used as a distinguisher in a reconstruction argument, the resulting reconstruction procedure is also an ordinary probabilistic procedure. It therefore yields a bound on the unrelativized complexity $\text{pK}^{\text{poly}(t)}(y \mid x)$, rather than merely on $\text{pK}^{\text{poly}(t),\text{SAT}}(y \mid x)$. This is the mechanism by which the proof removes the SAT oracle.

Concretely, we compare the SAT-oracle pK complexity of (x, y) in two ways. The upper bound describes x using a SAT oracle program and then describes y given x using another such program:

$$\text{pK}^{t+\text{poly}(n),\text{SAT}}(x, y) \leq \text{pK}^{t,\text{SAT}}(x) + \text{pK}^{\text{poly}(n),\text{SAT}}(y \mid x) + O(\log t).$$

For the lower bound, fix k slightly below the ordinary complexity $\text{pK}^{\text{poly}(t)}(y \mid x)$ and compare the distributions $(x, \mathcal{U}_{mk+k})$ and $(x, \text{DP}_k(y, \mathcal{U}_{mk}))$, where DP (Direct Product) is a pseudorandom generator with suitable pseudorandomness and reconstruction properties. Weak symmetry of information implies that a uniformly random extension of x usually has high SAT relative complexity, so A rejects it at an appropriate threshold. If A behaved noticeably differently on the generated extension, the reconstruction property of the direct product generator would turn A into an ordinary probabilistic description of y given x of length at most $k + O(\log t)$, contradicting the choice of k . Thus, A must also reject some generated extension. With some additional work, this can be used to prove the following inequality:

$$\text{pK}^{t+\text{poly}(n),\text{SAT}}(x, y) + O(\log t) \geq \text{pK}^{\text{poly}(t)}(y \mid x) + \text{pK}^{\text{poly}(t),\text{SAT}}(x).$$

The important asymmetry is that the term involving x remains SAT relative, while the term involving $y \mid x$ is now unrelativized.

Combining the upper and lower bounds then rearranging gives

$$\text{pK}^{\text{poly}(t)}(y \mid x) \leq \text{pK}^{\text{poly}(n),\text{SAT}}(y \mid x) + \left(\text{pK}^{t,\text{SAT}}(x) - \text{pK}^{\text{poly}(t),\text{SAT}}(x) \right) + O(\log t).$$

Thus, the SAT oracle has been removed from the description of y , at the cost of a *computational depth term* measuring the decrease in the SAT relative complexity of x between the two time bounds t and $\text{poly}(t)$.

To eliminate this remaining cost, we use a relativized extension of the lifted measure $\ell\text{-pK}_{\lambda,\gamma}^t$ from [HLO26]. Crucially, this measure admits a more efficient form of symmetry of information, allowing the two relevant time bounds to differ by only an additive $\text{poly}(n)$ amount. The assumed algorithm for $\text{Mild-Gap-MINpKT}^{\text{SAT}}$ lifts to the corresponding meta-computational problem for the relativized $\ell\text{-pK}$ variant considered here, and a standard telescoping argument now finds a polynomially bounded time scale at which the lifted complexity of x changes by only $O(1)$. At this time scale, the two SAT relative terms involving x cancel up to logarithmic overhead, leaving

$$\text{pK}^{\text{poly}(n)}(y \mid x) \leq \text{pK}^{\text{poly}(n),\text{SAT}}(y \mid x) + O(\log n).$$

Combining this oracle removal inequality with language compression for $\text{pK}^{t,\text{SAT}}$ is sufficient to prove Equation (4).

Finally, the two correspondences described above yield Theorem 1.1.

Equivalence of Gap and No-Gap Hardness in the Average-Case Setting. Here we describe the ideas behind the proof of Theorem 1.2. It is not hard to see that the tractability of $\text{MINpKT}^{\text{PH}}$ implies that of $\text{Gap-MINpKT}^{\text{PH}}$. Therefore, it suffices to prove the converse implication in each item.

We first consider the zero-error setting of AvgBPP. The starting point is the observation that standard worst-case-to-average-case reduction techniques for meta-complexity problems, developed in [Hir18, GKLO22], allow us to obtain that if $\text{Gap-MINpKT}^{\text{PH}}$ admits a zero-error average-case algorithm, then it also admits a worst-case algorithm. In other words, $\text{Gap-MINpKT}^{\text{PH}} \in \text{prBPP}$. Then by Correspondence I (illustrated in Figure 1), this yields $\text{DistPH} \subseteq \text{AvgBPP}$. Finally, we observe that the non-gap problem, $\text{MINpKT}^{\text{PH}}$, is decidable in (promise) $\text{AM}^{\text{PH}} = \text{PH}$. Consequently, $\text{MINpKT}^{\text{PH}}$ also admits a zero-error average-case algorithm.

We now turn to the two-sided-error setting of HeurBPP. First, it is worth noting that existing work implicitly establishes that the (two-sided-error) average-case tractability of Gap-MINpKT and MINpKT are equivalent. Indeed, [IRS22] showed that the (two-sided-error) average-case hardness of approximating the time-unbounded Kolmogorov complexity K is equivalent to the existence of one-way functions. Their proof can be adapted to pK^t to show that the (two-sided-error) average-case hardness of Gap-MINpKT is equivalent to the existence of one-way functions (and hence to the existence of pseudorandom generators [HILL99]).⁵ On the other hand, [LP23] established that the (two-sided-error) average-case hardness of MINpKT is also equivalent to the existence of one-way functions.

It is therefore natural to try to extend these equivalences to the setting with PH oracles by considering PH-computable one-way functions. However, there are a couple of obstacles. First, it is unclear how to adapt the ideas of [IRS22] to show that if $\text{Gap-MINpKT}^{\text{PH}}$ is easy on average, then no PH-computable pseudorandom generators (equivalently, PH-computable one-way functions) exist. Ideally, one would use an average-case algorithm for $\text{Gap-MINpKT}^{\text{PH}}$ to break the security of such a generator, since every output of a PH-computable generator has small time-bounded Kolmogorov complexity with PH oracles. The issue is that a PH-computable pseudorandom generator induces a distribution that is polynomial-time samplable only with access to a PH oracle, whereas our assumption asserts only that $\text{Gap-MINpKT}^{\text{PH}}$ is easy over polynomial-time samplable distributions without such oracle access. Second, the techniques of [LP23] yield only the following relativized statement: for every oracle $\mathcal{O}$ (for example, $\mathcal{O} \in \text{PH}$), if there are no one-way functions computable with oracle access to $\mathcal{O}$ that are secure against probabilistic polynomial-time algorithms *with oracle access* to $\mathcal{O}$, then $\text{MINpKT}^{\mathcal{O}} \in \text{HeurBPP}^{\mathcal{O}}$. Thus, these results do not yield the desired equivalence between the average-case tractability of $\text{Gap-MINpKT}^{\text{PH}}$ and that of $\text{MINpKT}^{\text{PH}}$ via PH-computable one-way functions.

Instead, we take a different route based on [HS22]. Specifically, [HS22] proved that

$$\text{DistPH} \not\subseteq \text{HeurBPP}$$

is equivalent to the existence of PH-computable pseudorandom generators that are secure against probabilistic polynomial-time algorithms. Moreover, the pseudorandom generators obtained there satisfy a certain entropy-preserving property that allows us to break them using an average-case algorithm

⁵In fact, here we will only use the implication that if one can efficiently approximate pK^t -complexity, then pseudorandom generators do not exist. The intuition is that such an approximation would allow us to break any candidate pseudorandom generator: the output of a generator has small time-bounded Kolmogorov complexity, roughly bounded by the length of its seed, whereas a truly random string has high complexity. For the converse direction, the result in [IRS22] relies on the coding theorem for K -complexity. The same argument extends to pK^t by applying the optimal coding theorem for pK^t -complexity established in [LOZ22].

for $\text{Gap-MINpKT}^{\text{PH}}$ over the uniform distribution. Consequently, the (two-sided-error) average-case tractability of $\text{Gap-MINpKT}^{\text{PH}}$ implies that

$$\text{DistPH} \subseteq \text{HeurBPP}.$$

Again, since the non-gap problem $\text{MINpKT}^{\text{PH}}$ is decidable in (promise) $\text{AM}^{\text{PH}} = \text{PH}$, it follows that $\text{MINpKT}^{\text{PH}}$ also admits a two-sided-error average-case algorithm.

One technical point omitted from the discussion above is that the characterization of [HS22] only concerns pseudorandom generators with arbitrary logarithmic stretch. A minor extension of their result is nevertheless sufficient for our purposes, because it is enough to establish the desired connection with the tractability of $\text{Gap}_\rho\text{-MINpKT}^{\text{PH}}$, where ρ is any fixed polynomial (see Claim 5.6 for details).

Relativization Barrier. Finally, we briefly comment on the proof of Theorem 1.3. The relativization barrier is proved by applying the same correspondences described above inside a suitable oracle world. Impagliazzo [Imp11], followed by Hirahara and Nanashima [HN21], give an oracle $\mathcal{O}$ for which $\text{DistPH}^{\mathcal{O}} \subseteq \text{AvgBPP}^{\mathcal{O}}$ but $\text{PH}^{\mathcal{O}} \not\subseteq \text{BPP}^{\mathcal{O}}$. The first property feeds into the *relativized* average-case correspondence, giving $\text{Gap-MINpKT}^{\text{PH}^{\mathcal{O}}, \mathcal{O}} \in \text{prBPP}^{\mathcal{O}}$. On the other hand, if the corresponding mild gap problem were also easy, the *relativized* worst-case correspondence would imply $\text{NP}^{\mathcal{O}} \subseteq \text{BPP}^{\mathcal{O}}$ and hence $\text{PH}^{\mathcal{O}} \subseteq \text{BPP}^{\mathcal{O}}$, contradicting the oracle separation. Most of the work goes into the verification that the techniques employed to establish these correspondences relativize.

1.4 Related Work

The modern theory of average-case complexity for distributional problems was initiated by Levin [Lev86] and developed into the framework surveyed by Bogdanov and Trevisan [BT06]. Impagliazzo's five worlds [Imp95] placed the average-case versus worst-case question at the center of complexity theory and cryptography. The particular question relevant here is the exclusion of PH Heuristica, namely whether $\text{DistPH} \subseteq \text{AvgBPP}$ implies $\text{PH} \subseteq \text{BPP}$.

Classical barrier results explain why this question has resisted the most direct reduction based approaches. Feigenbaum and Fortnow [FF93] showed that strong forms of random self reducibility for complete sets would imply unexpected collapses of the polynomial hierarchy. Bogdanov and Trevisan [BT03] extended this barrier to nonadaptive self correction with respect to samplable distributions: if an NP complete problem had such a self corrector, then $\text{coNP} \subseteq \text{NP/poly}$ and the polynomial hierarchy collapses. Thus, black box nonadaptive reductions cannot by themselves base average-case hardness in NP on worst case hardness of an NP-complete problem, except under unlikely collapses. Gutfreund, Shaltiel, and Ta Shma [GSTS07] obtained a positive but weaker connection: if $\text{NP} \not\subseteq \text{BPP}$, then for every fixed algorithm there is a polynomial time samplable distribution on which it errors often. The caveat is that the hard distribution may depend on the algorithm, so this does not produce a single distributional problem hard for all efficient algorithms.

Hirahara developed the meta-complexity approach to this question through a series of works (see, e.g., [Hir22a]). The first step was a non-black-box worst-case to average-case reduction for time-bounded Kolmogorov complexity [Hir18]. Hirahara then characterized $\text{DistPH} \subseteq \text{AvgP}$ by the worst-case easiness of $\text{Gap-MINKT}^{\text{PH}}$ [Hir20], and later proved that average-case easiness of PH yields worst case algorithms for PH running in time $2^{O(n/\log n)}$ [Hir21]. Goldberg, Kabanets, Lu, and Oliveira [GKLO22] lifted the original result to the probabilistic setting and simplified the proof using probabilistic Kolmogorov complexity.

Other works have refined and extended this paradigm. Chen, Hirahara, and Vafa [CHV22] obtained average-case hardness consequences for NP and PH from fine-grained worst-case hardness assumptions. Hirahara [Hir22c] related the exclusion of Heuristica to symmetry of information and hardness of approximating conditional time-bounded Kolmogorov complexity.

Our work is different in that the main theorem offers an *equivalence* for the implication $\text{DistPH} \subseteq \text{AvgBPP} \Rightarrow \text{PH} \subseteq \text{BPP}$, rather than only a sufficient condition. Moreover, the equivalence is between this complexity-theoretic implication and a reduction between two meta-computational approximation regimes, the standard gap and mild gap versions of $\text{MINpKT}^{\text{PH}}$.

Relativization has also played an important role in understanding what meta-complexity methods can and cannot prove. Impagliazzo [Imp11] constructed relativized separations between worst-case and average-case complexity for NP, and Hirahara and Nanashima [HN21] studied worst-case learning in relativized Heuristica. Ren and Santhanam [RS22] initiated a systematic study of relativization in meta-complexity. Among other oracle worlds, they separate decision and search versions of MCSP, show that different MCSP thresholds can lead to different complexities, that Kt can become efficiently approximable, and that robust reductions from DistNP to Gap-MINKT face oracle barriers. Hirahara, Lu, and Ren [HLR23] later introduced bounded relativization as a finer way to classify proof techniques that may go beyond ordinary relativization while still relativizing against restricted oracle classes.

In contrast, our relativization theorem deals with a delicate regime of parameters and constructs an oracle world where the standard gap version of $\text{MINpKT}^{\text{PH}}$ is easy while the mild gap version remains hard. In this sense, the oracle result isolates the exact nonrelativizing step needed by our main equivalence: transforming average-case easiness into worst-case easiness of the standard gap problem is relativizing, but upgrading the standard gap regime to the mild gap regime cannot be done by relativizing arguments alone.

Nonrelativizing techniques have also played an important role in establishing hardness results for meta-computational problems. Hirahara [Hir22b] employed ideas from algorithmic information theory to overcome a relativization barrier for showing hardness of learning programs and, in the same work, proved NP-hardness under randomized polynomial-time reductions for the partial function variant MCSP* and related problems such as MKTP* and MINKT*. The proof uses conditional Kolmogorov complexity and pseudo-random generators to quantify the information available to a small program, providing a complementary example of nonrelativizing techniques in meta-complexity.

A separate line of work has sought to reduce the *compression gap* between the description length thresholds in the yes and no cases of meta-computational promise problems. Beginning with Hirahara’s non-black-box worst-case to average-case reduction for time-bounded Kolmogorov complexity, subsequent results progressively reduced this loss [Hir18, Hir21, GKLO22]. Most recently, Liu and Pass [LP26] eliminated the compression gap entirely in the large threshold regime, where the description length threshold is close to the input length, obtaining a worst-case to average-case reduction for the exact problem under a standard derandomization assumption. This progress concerns the gap between description lengths. By contrast, reducing the separate *time gap* between the running time bounds in the two promise cases appears to have received less attention. Investigation of changes in the time bound appear in the fine-grained reductions of Chen, Hirahara, and Vafa [CHV22] and in the recent witness compression framework of Hu, Lu, and Oliveira [HLO26]. Our work shows that the *time gap* is fundamental to our understanding of average-case complexity.

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

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

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

2.1 Computational Models

We consider multi-tape Turing machines over the alphabet $\Sigma = \{0, 1, \perp\}$. In addition to the usual input tape and work tapes, in some contexts we allow a machine to access three additional tapes: one holding an auxiliary string, one holding a non-deterministic guess, and one holding the randomness. We assume that the machine has *sequential access* (i.e., no random access) to the randomness tape and the non-deterministic tape.⁶

For a machine M , time bound t , and string x , we let $K_M^t(x)$ 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 $K_U^{C \cdot t \log t}(x) \leq K_{U'}^t(x) + C$. 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 $\tilde{O}(|M|, t)$ 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 valid encoding of a machine and $\alpha \in \{0, 1\}^*$ is an “advice” string. The output of Π on $(x; w; r)$ (which corresponds to the auxiliary input, the non-deterministic guess, and the randomness), denoted $\Pi(x; w; r)$, is given by $M(\alpha; x; w; r)$.

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

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

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

where $\langle M \rangle$ is an encoding of the machine M ,⁷ and s_1 (resp. s_2) is the binary representation of the integer specifying the length of $\langle M \rangle$ (resp. α), with each bit duplicated. The size of Π , denoted $|\Pi|$, is the total bit length of this description.

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

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

A Computational Model with Efficient Composition. The above computational model suffices for most of our results, except for [Theorem 1.1](#). As in the main result of [HLO26], [Theorem 1.1](#) requires a computational model that possesses an additional natural efficiency property called “efficient composition”. Specifically, this property states that there exists a polynomial p such that the following holds. For any $x, y, z, r \in \{0, 1\}^*$, if:

⁶To 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.

⁷As noted above, we assume a fixed standard encoding for Turing machines.

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

then there is a program of size $s_1 + s_2 + \log p(|x| + |y| + |z|)$ that outputs z on input $(x; r)$ in time $t_1 + p(t_2) + p(|x| + |y| + |z|)$. (As discussed in detail in [HLO26], this property is typically implicitly assumed in the literature on time-bounded Kolmogorov complexity.)

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

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

where each M_i is a valid encoding of a machine and $\alpha_i \in \{0, 1\}^*$. The output of Π on $(x; r)$ (which corresponds to the auxiliary input and the randomness), denoted $\Pi(x; r)$, is obtained as follows:⁸

- Let y_1 be the output of $M_1(\alpha_1; x; r)$. Let t_1 be the number of steps $M_1(\alpha_1; x; r)$ takes to produce y_1 .
- Let y_2 be the output of $M_2(\alpha_2; x \circ y_1; r_{[t_1+1:|r|]})$, and let t_2 be the corresponding running time.
- ⋮
- Let y_ℓ be the output of $M_\ell(\alpha_\ell; x \circ y_{\ell-1}; r_{[1+\sum_{j<\ell} t_j:|r|]})$, and let t_ℓ be the corresponding running time.
- Output y_ℓ .

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

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

It can be shown that this definition of a program satisfies the efficient composition property (see [HLO26, Proposition 57]). The definition can be extended to programs that have oracle access to a language $\mathcal{O}$ in the natural way.

2.2 Time-Bounded Kolmogorov Complexity

We record only the Kolmogorov notions used in the main results and their proofs.

Definition 2.1 (K^t). For $x, y, r \in \{0, 1\}^*$, $t \in \mathbb{N}$, and an oracle $\mathcal{O}$, define

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

We write $K^{t, \mathcal{O}}(x \mid y)$ for $K^{t, \mathcal{O}}(x \mid y; \epsilon)$ and $K^{t, \mathcal{O}}(x)$ for $K^{t, \mathcal{O}}(x \mid \epsilon)$. We write $K(x)$ for the corresponding unbounded plain Kolmogorov complexity. If oracle queries are non adaptive, we write $K^{t, \parallel \mathcal{O}}$.

Definition 2.2 (pK^t). For $x, y \in \{0, 1\}^*$, $\lambda \in [0, 1]$, $t \in \mathbb{N}$, and an oracle $\mathcal{O}$, define

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

We omit $\mathcal{O}$ when it is empty and omit λ when $\lambda = 2/3$.

⁸For a string z of length ℓ and a subset $S \subseteq [\ell]$, where $[\ell] = \{1, 2, \dots, \ell\}$, we let z_S denote the substring of z obtained by concatenating the bits of z at indices in S . For $a < b$, we let $[a : b]$ denote the set $\{a, a+1, \dots, b\}$.

Definition 2.3 (nK^t). For $x, y, r \in \{0, 1\}^*$ and $t \in \mathbb{N}$, define

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

We write $\text{nK}^t(x \mid y)$ for $\text{nK}^t(x \mid y; \epsilon)$.

Definition 2.4 (pnK^t). For $x, y \in \{0, 1\}^*$, $\lambda \in [0, 1]$, and $t \in \mathbb{N}$, define

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

We omit λ when $\lambda = 2/3$.

2.3 Meta-Complexity

We now define the promise problems that appear in the theorem statements. The deterministic problems for the deterministic analogue are included in [Theorem 4.1](#).

Definition 2.5 (MINKT). For an oracle $\mathcal{O}$, we define the promise language $\text{MINKT}^\mathcal{O} \triangleq (\text{YES}, \text{NO})$ as

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

Definition 2.6 (Gap-MINKT). For an oracle $\mathcal{O}$ and $\rho: \mathbb{N} \rightarrow \mathbb{N}$, we define the promise language

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

as

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

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

Definition 2.7 (Mild-Gap-MINKT). For an oracle $\mathcal{O}$ and a function $\rho: \mathbb{N} \rightarrow \mathbb{N}$, we define the promise language $\text{Mild-Gap}_\rho\text{-MINKT}^\mathcal{O} \triangleq (\text{YES}, \text{NO})$ as

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

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

Definition 2.8 (MINpKT). For an oracle $\mathcal{O}$, we define the promise language $\text{MINpKT}^\mathcal{O} \triangleq (\text{YES}, \text{NO})$ as

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

Definition 2.9 (Gap-MINpKT). For an oracle $\mathcal{O}$ and $\rho: \mathbb{N} \rightarrow \mathbb{N}$, we define the promise language

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

as

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

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

Definition 2.10 (Mild-Gap-MINpKT). For an oracle $\mathcal{O}$ and a function $\rho: \mathbb{N} \rightarrow \mathbb{N}$, we define the promise language $\text{Mild-Gap}_\rho\text{-MINpKT}^\mathcal{O} \triangleq (\text{YES}, \text{NO})$ as

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

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

For all the promise problems

$$\Pi \in \{\text{Gap}_\rho\text{-MINKT}, \text{Mild-Gap}_\rho\text{-MINKT}, \text{Gap}_\rho\text{-MINpKT}, \text{Mild-Gap}_\rho\text{-MINpKT}\},$$

we adopt the following convention. For complexity classes $\mathcal{C}$ and $\mathcal{D}$, we say that $\Pi^\mathcal{D} \in \mathcal{C}$ if, for every language $L \in \mathcal{D}$, there exists a constant $c > 0$ such that $\Pi^L \in \mathcal{C}$ with gap parameter $\rho(n) = n^c$. In particular, note that ρ may depend on the choice of L .

2.4 Useful Tools

Here, we state some useful tools about Kolmogorov complexity.

Lemma 2.11 (Standard Counting Argument). For every $t \in \mathbb{N}$, $\mu \in \{\text{K}, \text{K}^t, \text{nK}^t\}$, and every $n, \gamma \in \mathbb{N}$, we have

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

Also, for every $t \in \mathbb{N}$, $\lambda \in (0, 1)$, $\mu \in \{\text{pK}_\lambda^t, \text{pnK}_\lambda^t\}$, and every $n, \gamma \in \mathbb{N}$, we have

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

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 less than $n - \gamma$ is at most $2^{n-\gamma} - 1$. Therefore,

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

The deterministic and nondeterministic time bounded cases follow similarly, since each program of length less than $n - \gamma$ contributes at most one output string.

For $\mu = \text{pK}_\lambda^t, \text{pnK}_\lambda^t$, 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 x if $M(r)$ outputs x . For a right vertex x to have pK_λ^t -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 $2^{n - \gamma} / \lambda$. $\square$

Lemma 2.12 (Following [GKLO22, Lemma 18]). *Let L be any computable language. There exists a constant $c > 0$ such that for any $x \in \{0, 1\}^*$, time bound $t \in \mathbb{N}$, $\text{K}(x \mid t) \leq \text{pK}^{t, L}(x) + c \cdot \log |x|$.*

Theorem 2.13 (Language Compression for pnK^t [HLO26]). *There is a polynomial $p_{\text{LC}}(\cdot)$ such that for every set $A := \{A_n \subseteq \{0, 1\}^n\}_{n \in \mathbb{N}}$, and for all $n \in \mathbb{N}$ and $x \in A_n$,*

$$\text{pnK}^{p_{\text{LC}}(n), A_n}(x) \leq \log |A_n| + \log p_{\text{LC}}(n).$$

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

Definition 2.14 (Direct Product Generator (DPG) [Hir21]). *For $n, k \in \mathbb{N}$, the k -wise direct product generator is the function*

$$\text{DP}_k: \{0, 1\}^n \times \{0, 1\}^{nk} \rightarrow \{0, 1\}^{nk+k},$$

defined by

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

Lemma 2.15 (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 $\text{DP}_k(x; \mathcal{U}_{nk})$ from $\mathcal{U}_{nk+k}$. Then*

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

A SAT Oracle Simulation Lemma.

Lemma 2.16. *There exists a constant $c > 0$ such that for any strings $x, y \in \{0, 1\}^*$ and $t \in \mathbb{N}$,*

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

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 $k := \text{nK}^t(x \mid y)$, 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 $\text{poly}(|x| + |y| + t)$. Also, by the fact that Π nondeterministically outputs x on input y in time t , it can correctly recover x . Thus, for $t' = \text{poly}(|y| + t)$, we obtain

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

as desired. $\square$

Lemma 2.17. *For every oracle $\mathcal{O}$ and every language L that is hard for $\text{NP}^{\mathcal{O}}$ under deterministic polynomial-time reductions, there exists a constant $c > 0$ such that, for all $x, y \in \{0, 1\}^*$ and $t \in \mathbb{N}$,*

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

Proof. The proof follows the same idea as Lemma 2.16. Let $k = \text{pnK}^{t, \mathcal{O}}(x \mid y)$, 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 $L_0 \in \text{NP}^{\mathcal{O}}$ be the following language:

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

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 $\text{poly}(t)$ 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 $\text{NP}^{\mathcal{O}}$ -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 $\text{poly}(t)$.

Overall, we have

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

as desired. □

2.5 Average-Case Complexity

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

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

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

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

1. for every $x \in \text{Support}(\mathcal{D}_n)$,

$$A(x, 1^n, 1^k) \in \{L(x), \perp\},$$

2. and

$$\Pr_{x \sim \mathcal{D}_n} [A(x, 1^n, 1^k) \neq \perp] \geq 1 - \frac{1}{k}.$$

We let AvgP denote the class of distributional problems that admit an errorless heuristic scheme.

The above definition naturally extends to randomized algorithms. A distributional problem $(L, \mathcal{D})$ is said to admit a *randomized errorless heuristic scheme* if there exists a probabilistic polynomial-time algorithm A such that, for every $n, k \in \mathbb{N}$,

1. for every $x \in \text{Support}(\mathcal{D}_n)$,

$$\Pr_A \left[A(x, 1^n, 1^k) \in \{L(x), \perp\} \right] \geq \frac{4}{5},$$

2. and

$$\Pr_{x \sim \mathcal{D}_n} \left[\Pr_A \left[A(x, 1^n, 1^k) = \perp \right] < \frac{1}{5} \right] \geq 1 - \frac{1}{k}.$$

We let AvgBPP denote the class of distributional problems that admit randomized errorless heuristic schemes.

The above definitions require the heuristic algorithm to be errorless, allowing it only to output $\perp$ instead of an incorrect answer. We can relax this requirement by permitting the algorithm to err on both yes- and no-instances.

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

$$\Pr_{x \sim \mathcal{D}_n} \left[\Pr_A \left[A(x, 1^n, 1^k) = L(x) \right] > \frac{3}{5} \right] \geq 1 - \frac{1}{k}.$$

We let HeurBPP denote the class of distributional problems that admit randomized two-sided-error heuristic schemes.

We will need the following result.

Theorem 2.18 ([IL90]). *Let A be any oracle. If $(\text{NP}^A, \mathcal{U}) \subseteq \text{AvgBPP}$, then for every distribution $\mathcal{D} \in \text{PSAMP}$ and $L \in \text{NP}^A$, we have $(L, \mathcal{D}) \in \text{AvgBPP}$.*

Notions of Average-Case Tractability for Meta-Complexity Problems. Let A be an algorithm, $\Pi := (\text{YES}, \text{NO})$ be a promise problem, and $z \in \{0, 1\}^*$, we write “ A decides Π on z ” if the following holds.

$$A(z) = \begin{cases} 1 & \text{if } z \in \text{YES}, \\ 0 & \text{if } z \in \text{NO}, \\ \text{either 0 or 1} & \text{otherwise.} \end{cases}$$

Definition 2.19 (“Gap-MINpKT^{PH} $\in$ HeurBPP”). *Let O be an oracle and let $\rho: \mathbb{N} \rightarrow \mathbb{N}$. We say that $\text{Gap}_\rho\text{-MINpKT}^O$ is easy on average with two-sided error if for every polynomial-time samplable distribution family $\{\mathcal{D}_n\}_n$ supported over $\{0, 1\}^n$, there exist a PPT algorithm A such that for every $n, s, t, a, b, k \in \mathbb{N}$,*

$$\Pr_{x \sim \mathcal{D}_n} \left[\Pr_A \left[A(-; 1^k) \text{ decides } \text{Gap}_\rho\text{-MINpKT}^O \text{ on } (x, 1^s, 1^t, 1^a, 1^b) \right] > \frac{3}{5} \right] \geq 1 - \frac{1}{k}.$$

We say that $\text{Gap-MINpKT}^{\text{PH}}$ is easy on average with two-sided error, denoted as “Gap-MINpKT^{PH} $\in$ HeurBPP”, if for every $O \in \text{PH}$, there exist a polynomial ρ such that $\text{Gap}_\rho\text{-MINpKT}^O$ is easy on average with two-sided error.

In the following zero error setting, we further require that the algorithm will never output an incorrect answer on both the positive and negative instances (except with small probability over the internal randomness of the algorithm), while it can output $\perp$ to indicate that it does not know the correct answer.

Definition 2.20 (“Gap-MINpKT^{PH} $\in$ AvgBPP”). *Let O be an oracle and let $\rho: \mathbb{N} \rightarrow \mathbb{N}$. We say that $\text{Gap}_\rho\text{-MINpKT}^O$ is easy on average with zero error if for every polynomial-time samplable distribution family $\{\mathcal{D}_n\}_n$ supported over $\{0, 1\}^n$, there exist a PPT algorithm A such that for all $n, s, t, a, b, k \in \mathbb{N}$,*

$$\Pr_{x \sim \mathcal{D}_n} \left[\Pr_A \left[A(-; 1^k) \text{ decides } \text{Gap}_\rho\text{-MINpKT}^O \text{ on } (x, 1^s, 1^t, 1^a, 1^b) \right] > \frac{3}{5} \right] \geq 1 - \frac{1}{k}.$$

and for every $x \in \text{Support}(\mathcal{D}_n)$,

$$\Pr_A \left[A(-; 1^k) \text{ decides } \text{Gap}_\rho\text{-MINpKT}^O \text{ on } (x, 1^s, 1^t, 1^a, 1^b) \text{ OR } A(x, 1^s, 1^t, 1^a, 1^b; 1^k) = \perp \right] \geq \frac{4}{5}.$$

We say that $\text{Gap-MINpKT}^{\text{PH}}$ is easy on average with zero error, denoted as “Gap-MINpKT^{PH} $\in$ AvgBPP”, if for every $\mathcal{O} \in \text{PH}$, there exist a polynomial ρ such that $\text{Gap}_\rho\text{-MINpKT}^{\mathcal{O}}$ is easy on average with zero error.

It is straightforward to extend the above definitions to Mild-Gap-MINpKT^{PH} and MINpKT^{PH}.

3 Proof of the Equivalence Theorem

In this section, we prove [Theorem 1.1](#), which is restated below.

Theorem 1.1 (Main Equivalence). *The following statements are equivalent:*

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

[Theorem 1.1](#) follows directly from the following two theorems using that $\text{NP} \subseteq \text{BPP}$ implies $\text{PH} \subseteq \text{BPP}$.

Theorem 3.1. *The following holds.*

$$\text{NP} \subseteq \text{BPP} \iff \text{Mild-Gap-MINpKT}^{\text{PH}} \in \text{prBPP}.$$

Theorem 3.2. *The following holds.*

$$\text{DistPH} \subseteq \text{AvgBPP} \iff \text{Gap-MINpKT}^{\text{PH}} \in \text{prBPP}.$$

In the remainder of this section, we show [Theorem 3.1](#) and [Theorem 3.2](#). We prove [Theorem 3.1](#) in [Section 3.1](#). The forward direction of [Theorem 3.2](#) is established by [Lemma 3.11](#), which is proved in [Section 3.2](#). The backward direction follows from [Lemma 3.13](#), with the proof provided in [Section 3.3](#).

3.1 Characterizing Easiness of NP via Easiness of Mild-Gap-MINpKT^{PH}

In this section, we prove [Theorem 3.1](#). The theorem follows directly from [Lemma 3.3](#) and [Lemma 3.4](#), which are proved in the subsections below.

3.1.1 The Easy Direction

Lemma 3.3. *If $\text{NP} \subseteq \text{BPP}$, then $\text{Mild-Gap-MINpKT}^{\text{PH}} \in \text{prBPP}$.*

Proof. It is easy to see that, for any oracle $\mathcal{O}$, we have $\text{Mild-Gap-MINpKT}^{\mathcal{O}} \in \text{prAM}^{\mathcal{O}}$.⁹ Also, if $\text{NP} \subseteq \text{BPP}$, then $\text{PH} \subseteq \text{BPP}$. It follows that $\text{prAM}^{\text{PH}} \subseteq \text{prAM}^{\text{BPP}} = \text{prAM} \subseteq \text{prBPP}^{\text{NP}} \subseteq \text{prBPP}^{\text{BPP}} = \text{prBPP}$. $\square$

3.1.2 The Hard Direction

Lemma 3.4. *If $\text{Mild-Gap-MINpKT}^{\text{SAT}} \in \text{prBPP}$, then $\text{NP} \subseteq \text{BPP}$.*

The proof of Lemma 3.4 follows the approach in [HLO26], which shows that if $\text{MINnKT} \in \text{BPP}$, then $\text{NP} \subseteq \text{BPP}$. Here, we sketch the proof and highlight the key steps and modifications.

We begin by describing the approach at a high level. Let $L \in \text{NP}$, and let $x \in \{0, 1\}^n$ be an input instance. Our goal is to find a witness for x . To this end, suppose $x \in L$, and let A_x denote the set of L -witnesses for x . We aim to establish the following: for every $y \in A_x$,

$$\text{pK}^{\text{poly}(n)}(y \mid x) \leq \log |A_x| + O(\log n). \quad (5)$$

The above is a consequence of a result known as *language compression* (for pK^t), which states that every witness $y \in A_x$ can be compressed to approximately $\log |A_x|$ bits.

Note that if $\text{pK}^{\text{poly}(n)}(y \mid x) \leq k$, then by the definition of pK^t , if we randomly sample a string $r \sim \{0, 1\}^{\text{poly}(n)}$ and a program $\Pi \sim \{0, 1\}^{\leq k}$, and output $\Pi(x; r)$, we obtain y with probability at least $1/(2^k \cdot \text{poly}(n))$. Then, by Equation (5), for every $y \in A_x$, the above procedure samples y with probability at least $1/(|A_x| \cdot \text{poly}(n))$. Since this holds for each of the $|A_x|$ elements in A_x , it follows that we obtain some string in A_x with probability at least $1/\text{poly}(n)$. By repeating this procedure, we can obtain a witness for x with high probability.

It is not known whether pK^t admits a language compression property as stated in Equation (5). However, it was shown in [HLO26] that its nondeterministic variant, pnK^t , indeed admits such a property. Since $\text{pK}^{t, \text{SAT}}$ is a more powerful notion than pnK^t , it follows that $\text{pK}^{t, \text{SAT}}$ also admits this language compression property. In particular, we get that for every witness $y \in A_x$,

$$\begin{aligned} \text{pK}^{\text{poly}(n), \text{SAT}}(y \mid x) &\leq \text{pnK}^{\text{poly}(n)}(y \mid x) + O(\log n) && \text{(by Lemma 2.17)} \\ &\leq \text{pnK}^{\text{poly}(n), A_x}(y) + O(\log n) \\ &\leq \log |A_x| + O(\log n). && \text{(by Theorem 2.13)} \end{aligned}$$

Thus, it suffices to show that if $\text{Mild-Gap-MINpKT}^{\text{SAT}} \in \text{prBPP}$, then for every $x, y \in \{0, 1\}^n$ and every t that is a sufficiently large polynomial in n ,

$$\text{pK}^{\text{poly}(t)}(y \mid x) \leq \text{pK}^{\text{poly}(n), \text{SAT}}(y \mid x) + O(\log t). \quad (6)$$

We first note that, following previous work, one can establish a weaker form of Equation (6), which states that

$$\text{pK}^{\text{poly}(t)}(y \mid x) \leq \text{pK}^{\text{poly}(n), \text{SAT}}(y \mid x) + \text{pK}^{t, \text{SAT}}(x) - \text{pK}^{\text{poly}(t), \text{SAT}}(x) + O(\log t). \quad (7)$$

Note that this weaker bound is not sufficient for our purposes, since the term $\text{pK}^{t, \text{SAT}}(x) - \text{pK}^{\text{poly}(t), \text{SAT}}(x)$, known as the (time-bounded) computational depth, can be large for some x . Nevertheless, we sketch how Equation (7) can be shown under the assumption that $\text{Mild-Gap-MINpKT}^{\text{SAT}}$ is easy.

⁹This is true even for $\text{MINpKT}^{\mathcal{O}}$.

The idea is to show both an upper bound and a lower bound for the quantity

$$\text{pK}^{t+\text{poly}(n),\text{SAT}}(x, y).$$

Roughly speaking, we aim to show that:

- **(Upper bound)** $\text{pK}^{t+\text{poly}(n),\text{SAT}}(x, y) \lesssim \text{pK}^{\text{poly}(n),\text{SAT}}(y \mid x) + \text{pK}^{t,\text{SAT}}(x).$
- **(Lower bound)** $\text{pK}^{t+\text{poly}(n),\text{SAT}}(x, y) \gtrsim \text{pK}^{\text{poly}(t)}(y \mid x) + \text{pK}^{\text{poly}(t),\text{SAT}}(x).$

It is easy to combine the two inequalities above to obtain [Equation \(7\)](#).

The upper bound is straightforward. The lower bound can be established under the assumption that Mild-Gap-MINpKT^{SAT} is easy. For simplicity, let us assume that we have a deterministic polynomial-time algorithm A that given $(z, 1^s, 1^t)$ decides whether $\text{pK}^{t,\text{SAT}}(z) \leq s$.

Let $\text{DP}_k: \{0, 1\}^n \times \{0, 1\}^{nk} \rightarrow \{0, 1\}^{nk+k}$ be the direct product generator defined in [Definition 2.14](#), where

$$k := \text{pK}^{t^c}(y \mid x) - c \cdot \log t,$$

for some large constant $c > 1$ to be specified later. We then consider the following two distributions:

$$\begin{aligned} \mathcal{D}_1 &:= (x, \text{DP}_k(y, \mathcal{U}_{nk})), \\ \mathcal{D}_2 &:= (x, \mathcal{U}_{nk+k}). \end{aligned}$$

The goal is to show that $\mathcal{D}_1$ and $\mathcal{D}_2$ are indistinguishable under the algorithm $A(-, 1^s, 1^{t+n^d})$ for some s and constant d to be specified later. The reason for doing this is that, assuming Mild-Gap-MINpKT^{SAT} is easy, one can show a weak time-bounded symmetry of information [[Hir21](#)] (see also [[Ila23](#)]), which states that, with high probability over $w \sim \{0, 1\}^{nk+k}$, we have

$$\text{pK}^{t+n^d,\text{SAT}}(x, w) \geq \text{pK}^{(t+n^d)^b,\text{SAT}}(x) + nk + k - b \log t,$$

where $b \geq 0$ is a constant. Therefore, with high probability (over $\mathcal{D}_2$), $A(\mathcal{D}_2, 1^s, 1^{t+n^d}) = 0$ for $s := \text{pK}^{(t+n^d)^b,\text{SAT}}(x) + nk + k - b \log t$.

Now if $\mathcal{D}_1$ is indeed indistinguishable from $\mathcal{D}_2$, then with high probability (over $\mathcal{D}_1$), we also have $A(\mathcal{D}_1, 1^s, 1^{t+n^d}) = 0$, which implies that there exists some $z \in \{0, 1\}^{nk}$ such that

$$\text{pK}^{t+n^d,\text{SAT}}(x, \text{DP}_k(y, z)) \geq s.$$

Also, since $\text{DP}_k(y, z)$ is efficiently computable given y and z , by letting d be a sufficiently large constant, we have

$$\text{pK}^{t+n^d,\text{SAT}}(x, \text{DP}_k(y, z)) \leq \text{pK}^{t+\text{poly}(n),\text{SAT}}(x, y) + nk + O(\log t).$$

Combining the above two inequalities, we get

$$\begin{aligned} \text{pK}^{t+\text{poly}(n),\text{SAT}}(x, y) &\geq s - nk - O(\log t) \\ &= \text{pK}^{\text{poly}(t),\text{SAT}}(x) + k - O(\log t) \\ &= \text{pK}^{\text{poly}(t),\text{SAT}}(x) + \text{pK}^{\text{poly}(t)}(y \mid x) - O(\log t), \end{aligned}$$

which gives the desired lower bound.

Now, let us explain why $\mathcal{D}_1$ and $\mathcal{D}_2$ are indistinguishable. Indeed, if $A(-, 1^s, 1^{t+n^d})$ could distinguish $\mathcal{D}_1$ from $\mathcal{D}_2$, then by the fact that A is polynomial-time computable and by the reconstruction property of DP_k ([Lemma 2.15](#)), we would obtain

$$\text{pK}^{\text{poly}(t)}(y \mid x) \leq k + O(\log t),$$

which contradicts our choice of k by letting c be a sufficiently large constant.

Next, let us return to [Equation \(7\)](#). It turns out that if we could show a more fine-grained form:

$$\text{pK}^{\text{poly}(t)}(y \mid x) \leq \text{pK}^{\text{poly}(n), \text{SAT}}(y \mid x) + \text{pK}^{t, \text{SAT}}(x) - \text{pK}^{t+p(n), \text{SAT}}(x) + O(\log n), \quad (8)$$

where $p(n)$ is a polynomial independent of t , then we could aim to conclude the proof by showing that

$$\text{pK}^{t, \text{SAT}}(x) - \text{pK}^{t+p(n), \text{SAT}}(x) = O(\log n).$$

While this may not hold for a particular value of t , one can argue, via a telescoping sum trick, that such a bound holds for some t' that is not too large. More concretely, for any fixed time bound t , let $t_i = t + i \cdot p(n)$ for $i \in \{0, 1, \dots, n\}$. Note that $\text{pK}^{t_0, \text{SAT}}(x) - \text{pK}^{t_n, \text{SAT}}(x) \leq 2n$, since any n -bit string x can be described using at most $2n$ bits. Therefore,

$$\sum_{i=0}^{n-1} (\text{pK}^{t_i, \text{SAT}}(x) - \text{pK}^{t_{i+1}, \text{SAT}}(x)) = \text{pK}^{t_0, \text{SAT}}(x) - \text{pK}^{t_n, \text{SAT}}(x) \leq 2n,$$

which implies that there exists an index i such that $\text{pK}^{t_i, \text{SAT}}(x) - \text{pK}^{t_{i+1}, \text{SAT}}(x) \leq 2$. In particular, for some time bound $t' \leq t + n \cdot p(n)$, we have $\text{pK}^{t', \text{SAT}}(x) - \text{pK}^{t'+p(n), \text{SAT}}(x) = O(\log n)$, which would be sufficient for our purpose.

Unfortunately, it is not clear how to prove [Equation \(8\)](#) exactly. As illustrated earlier, a key difficulty lies in the use of weak symmetry of information, which introduces a time overhead of the form $\text{poly}(t)$ rather than an *additive* overhead of the form $t + p(n)$.

Here, we follow the approach developed in [\[HLO26\]](#) which incorporates the ideas described above and suffices to prove [Lemma 3.4](#). A key ingredient is the following new notion of time-bounded Kolmogorov complexity.

Definition 3.5 ($\ell\text{-pK}_{\lambda, \gamma}^t$). For $t, \ell \in \mathbb{N}$, $\lambda, \gamma \in (0, 1)$, an oracle $\mathcal{O}$, and a string $x \in \{0, 1\}^*$, we define

$$\ell\text{-pK}_{\lambda, \gamma}^{t, \mathcal{O}}(x) \triangleq \min \left\{ s \in \mathbb{Z} \mid \Pr_{\hat{r} \sim \mathcal{U}_\ell} \left[\text{pK}_{\lambda}^{t, \mathcal{O}}(x \circ \hat{r}) \leq s + \ell \right] \geq \gamma \right\}.$$

It can be shown that $\ell\text{-pK}_{\lambda, \gamma}^t$ is a well-defined measure that satisfies several basic properties one would expect from a Kolmogorov complexity measure; see [\[HLO26, Section 4.2.1\]](#). Moreover, if the mild-gap version of approximating pK^t is easy, then so is approximating $\ell\text{-pK}_{\lambda, \gamma}^t$.

Definition 3.6 (Mild-Gap-MIN $\ell\text{-pKT}$). For an oracle $\mathcal{O}$ and $\rho: \mathbb{N} \rightarrow \mathbb{N}$, we define the promise language $\text{Mild-Gap}_\rho\text{-MIN}\ell\text{-pKT}^\mathcal{O} \triangleq (\text{YES}, \text{NO})$ as

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

We say that $\text{Mild-Gap-MIN}\ell\text{-pKT}^\mathcal{O} \in \text{prBPP}$ if there is a constant $g > 0$ such that $\text{Mild-Gap}_\rho\text{-MIN}\ell\text{-pKT}^\mathcal{O} \in \text{prBPP}$ with $\rho(n) = n^g$.

Lemma 3.7. For every oracle $\mathcal{O}$, if $\text{Mild-Gap-MINpKT}^\mathcal{O} \in \text{prBPP}$, then $\text{Mild-Gap-MIN}\ell\text{-pKT}^\mathcal{O} \in \text{prBPP}$.

Proof. The proof is adapted from that of [\[HLO26, Lemma 69\]](#), which shows a similar result for the case of pnK^t and $\ell\text{-pnK}^t$ without the mild gap. For simplicity, we assume $\mathcal{O} = \emptyset$; it is straightforward to verify that the proof also holds for any oracle $\mathcal{O}$.

Let A be a probabilistic polynomial-time algorithm that decides Mild-Gap $_\rho$ -MINpKT with error at most $1/3$, for some polynomial ρ . Let A^m denote the algorithm that runs A m times and returns the majority vote. By the Chernoff bound, the error of A^m is bounded by $2^{-m/18}$.

Suppose we are given an instance

$$z := (x \in \{0, 1\}^n, 1^s, 1^t, 1^\ell, 1^a, 1^b, 1^c, 1^d)$$

of Mild-Gap $_\rho$ -MIN ℓ -pKT.

On the one hand, if z is a yes-instance, then $\ell\text{-pK}_{b/a, d/c}^t(x) \leq s$, and by the definition of $\ell\text{-pK}$, we have

$$\Pr_{\hat{r} \sim \mathcal{U}_\ell} \left[\text{pK}_{b/a}^t(x \circ \hat{r}) \leq s + \ell \right] \geq \frac{d}{c}.$$

Then by the correctness of A , we get

$$\Pr_{\hat{r} \sim \mathcal{U}_\ell} \left[\Pr_A \left[A^{\lceil 54 \log c \rceil}(x \circ \hat{r}, 1^{s+\ell}, 1^t, 1^a, 1^b) = 1 \right] \geq 1 - \frac{1}{c^3} \right] \geq \frac{d}{c},$$

which, by a union bound, yields

$$\Pr_{\hat{r} \sim \mathcal{U}_{\ell, A}} \left[A^{\lceil 54 \log c \rceil}(x \circ \hat{r}, 1^{s+\ell}, 1^t, 1^a, 1^b) = 1 \right] \geq \frac{d}{c} - \frac{1}{c^3}. \quad (9)$$

On the other hand, if z is a no-instance, then

$$\ell\text{-pK}_{(b-1)/a, (d-1)/c}^{t+\rho(n+\ell+a+b+c+d)}(x) \geq s + \log \rho(t + \ell + a + b + c + d),$$

and hence

$$\Pr_{\hat{r} \sim \mathcal{U}_\ell} \left[\text{pK}_{(b-1)/a}^{t+\rho(n+\ell+a+b+c+d)}(x \circ \hat{r}) < s + \ell + \log \rho(t + \ell + a + b + c + d) \right] < \frac{d}{c} - \frac{1}{c}.$$

Note that the above implies

$$\Pr_{\hat{r} \sim \mathcal{U}_\ell} \left[\text{pK}_{(b-1)/a}^{t+\rho(n+\ell+a+b)}(x \circ \hat{r}) < s + \ell + \log \rho(t + \ell + a + b) \right] < \frac{d}{c} - \frac{1}{c}.$$

Again, by the correctness of A , the above implies

$$\Pr_{\hat{r} \sim \mathcal{U}_\ell} \left[\Pr_A \left[A^{\lceil 54 \log c \rceil}(x \circ \hat{r}, 1^{s+\ell}, 1^t, 1^a, 1^b) = 1 \right] > \frac{1}{c^3} \right] < \frac{d}{c} - \frac{1}{c},$$

which, by an averaging argument, gives

$$\Pr_{\hat{r} \sim \mathcal{U}_{\ell, A}} \left[A^{\lceil 54 \log c \rceil}(x \circ \hat{r}, 1^{s+\ell}, 1^t, 1^a, 1^b) = 1 \right] < \frac{d}{c} - \frac{1}{c} + \frac{1}{c^3}. \quad (10)$$

Comparing Equation (9) and Equation (10), we obtain a polynomial-time algorithm that distinguishes yes- and no-instances with a gap of at least $1/c - 2/c^3 \geq 1/(2c)$. Such a gap can be amplified by repeating the algorithm $\text{poly}(c)$ times. $\square$

Next, assume that Mild-Gap-MIN ℓ -pKT $^{\text{SAT}} \in \text{prBPP}$. We will show a result that informally states the following: for every $x, y \in \{0, 1\}^n$ and for appropriate settings of parameters ℓ and t , we have

$$\text{pK}^{\text{poly}(t)}(y \mid x) \lesssim \text{pK}^{\text{poly}(n), \text{SAT}}(y \mid x) + \ell\text{-pK}^{t, \text{SAT}}(x) - (\ell + \text{poly}(n))\text{-pK}^{t+\text{poly}(n), \text{SAT}}(x).$$

Observe the similarity between the above and Equation (8). The precise technical lemma is as follows:

Lemma 3.8. Suppose $\text{Mild-Gap-MIN}\ell\text{-pKT}^{\text{SAT}} \in \text{prBPP}$. Then there exist a polynomial p and an integer $N_0 \in \mathbb{N}^+$ such that the following holds. For any $n, m, \tau, \ell, t, b, d \in \mathbb{N}^+$ and any strings $x \in \{0, 1\}^n$ and $y \in \{0, 1\}^m$ satisfying the following constraints:

- $2/3 < b/n^3, d/n^3 < 1$,
- $N_0 \leq n, m \leq \tau$,
- $\tau \leq \ell \leq \tau^4$,
- $t \geq 2p(\tau)$,

we have

$$\text{pK}^{p(t)}(y \mid x) \leq \text{pK}_{1-1/n^3}^{\tau, \text{SAT}}(y \mid x) + \ell \cdot \text{pK}_{b/n^3, (d+1)/n^3}^{t-p(\tau), \text{SAT}}(x) - (\ell + \tau^3) \cdot \text{pK}_{(b-1)/n^3, (d-2)/n^3}^{t+p(\tau), \text{SAT}}(x) + \log p(t).$$

Proof Sketch. The proof closely follows that of [HLO26, Lemma 74], which establishes a similar result for the measure $\ell\text{-pnK}$. The main difference is that the former assumes the easiness of the non-gap version, whereas here we assume the easiness of the mild-gap version. We provide a proof sketch below.

Let $x \in \{0, 1\}^n$ and $y \in \{0, 1\}^m$. Similar to the earlier discussion, the idea is to show both upper and lower bounds for the quantity

$$\ell\text{-pK}_{b/n^3, d/n^3}^{t, \text{SAT}}(x, y).$$

Upper Bound.

Claim 3.9. There exists a polynomial p_1 such that

$$\ell\text{-pK}_{b/n^3, d/n^3}^{t, \text{SAT}}(x, y) \leq \text{pK}_{1-1/n^3}^{\tau, \text{SAT}}(y \mid x) + \ell \cdot \text{pK}_{b/n^3, (d+1)/n^3}^{t-p_1(\tau), \text{SAT}}(x) + \log p_1(\tau).$$

Proof Sketch of Claim 3.9. Intuitively, the claim states that if there is a program that outputs x in time $t - \text{poly}(\tau)$, and a program that outputs y given x in time τ , then one can obtain a program that outputs (x, y) in time t . This intuition was shown to hold for $\ell\text{-pK}$ in [HLO26]. See the proof of [HLO26, Lemma 72] for details. $\diamond$

Lower Bound.

Claim 3.10. There exists a polynomial p_2 such that

$$\ell\text{-pK}_{b/n^3, d/n^3}^{t, \text{SAT}}(x, y) \geq (\ell + \tau^3) \cdot \text{pK}_{(b-1)/n^3, (d-2)/n^3}^{t+p_2(\tau), \text{SAT}}(x) + \text{pK}^{p_2(t)}(y \mid x) - \log p_2(t).$$

Proof of Claim 3.10. Let $\text{DP}_k: \{0, 1\}^m \times \{0, 1\}^{mk} \rightarrow \{0, 1\}^{mk+k}$ be the direct product generator (Definition 2.14), where

$$k := \text{pK}^{q(t)}(y \mid x) - \log q(t),$$

and q is a sufficiently large polynomial specified later.¹⁰

As in the previous case, we consider two distributions:

$$\begin{aligned} \mathcal{D}_1 &:= (x, \text{DP}_k(y, \mathcal{U}_{mk})), \\ \mathcal{D}_2 &:= (x, \mathcal{U}_{mk+k}). \end{aligned}$$

¹⁰We may assume without loss of generality that $k > 0$. Indeed, if $\text{pK}^{q(t)}(y \mid x) \leq O(\log t)$, then the claim follows trivially from standard monotonicity properties of $\ell\text{-pK}^t$ (see [HLO26, Lemmas 66 and 67]).

We aim to show that $\mathcal{D}_1$ and $\mathcal{D}_2$ are indistinguishable, and that with high probability over an element from $\mathcal{D}_2$, its complexity is large. To show the latter, we again use weak symmetry of information. However, the key difference here is that we will be able to use a *highly efficient* weak symmetry of information for the notion of ℓ -pK. More specifically, we have the following: For any $n, m', k, t', \ell \in \mathbb{N}^+$, any $\lambda, \gamma, \alpha \in (0, 1)$ satisfying $\alpha < \gamma$, and any string $x \in \{0, 1\}^n$, we have

$$\Pr_{z \sim \mathcal{U}_{m'}} \left[\ell\text{-pK}_{\lambda, \gamma}^{t', \text{SAT}}(x \circ z) \geq (\ell + m')\text{-pK}_{\lambda, \gamma - \alpha}^{t', \text{SAT}}(x) + m' \right] \geq \alpha. \quad (11)$$

The proof of this result appears in [HLO26, Lemma 71].

Now, suppose there exists a (randomized) polynomial-time algorithm A that solves Mild-Gap $_{\rho}$ -MIN ℓ -pKT^{SAT} for some polynomial ρ . Without loss of generality, by applying standard error reduction, we may assume that A has error probability at most $\delta := 1/(100n^4)$ on every promise instance. Let

$$t' := t + q(\tau) + \rho(\tau^5).$$

and

$$s := \left((\ell + mk + k)\text{-pK}_{(b-1)/n^3, (d-2)/n^3}^{t', \text{SAT}}(x) + mk + k \right) - \log \rho(t^4).$$

We get

$$\begin{aligned} & \Pr_{w \sim \mathcal{U}_{mk+k}} \left[\ell\text{-pK}_{(b-1)/n^3, (d-1)/n^3}^{t+q(\tau)+\rho(n+mk+k+\ell+b+d+2n^3), \text{SAT}}(x \circ w) \geq s + \log \rho(t + q(\tau) + \ell + b + d + 2n^3) \right] \\ & \geq \Pr \left[\ell\text{-pK}_{(b-1)/n^3, (d-1)/n^3}^{t+q(\tau)+\rho(\tau^5), \text{SAT}}(x \circ w) \geq s + \log \rho(t^4) \right] \\ & = \Pr \left[\ell\text{-pK}_{(b-1)/n^3, (d-1)/n^3}^{t', \text{SAT}}(x \circ w) \geq (\ell + mk + k)\text{-pK}_{(b-1)/n^3, (d-2)/n^3}^{t', \text{SAT}}(x) + mk + k \right] \\ & \geq \frac{1}{n^3}. \end{aligned} \quad (\text{by Equation (11) with } \alpha = 1/n^3 \text{ and } \lambda = (b-1)/n^3)$$

In fact, for the first inequality, we need the notion ℓ -pK^t to be monotone in t , meaning that ℓ -pK^t decreases as t increases (see [HLO26, Lemma 66]).

Note that the above implies that, with probability at least $1/n^3$ over $(x, w) \sim \mathcal{D}_2$, A (with probability at least $1 - \delta$ over its internal randomness) outputs 0 on $(x \circ w, 1^s, 1^{t+q(\tau)}, 1^\ell, 1^{n^3}, 1^b, 1^{n^3}, 1^d)$. This yields

$$\Pr_{(x, w) \sim \mathcal{D}_2, A} \left[A(x \circ w, 1^s, 1^{t+q(\tau)}, 1^\ell, 1^{n^3}, 1^b, 1^{n^3}, 1^d) = 0 \right] \geq (1 - \delta) \cdot \frac{1}{n^3}.$$

For the sake of contradiction, suppose the procedure $A(-, 1^s, 1^{t+q(\tau)}, 1^\ell, 1^{n^3}, 1^b, 1^{n^3}, 1^d)$ can $(1/n^4)$ -distinguish $\mathcal{D}_1$ and $\mathcal{D}_2$, i.e.,

$$\Pr_{z \sim \mathcal{U}_{mk}, A} \left[A(x \circ \text{DP}_k(y, z), 1^s, 1^{t+q(\tau)}, 1^\ell, 1^{n^3}, 1^b, 1^{n^3}, 1^d) = 0 \right] < (1 - \delta) \cdot \frac{1}{n^3} - \frac{1}{n^4}. \quad (12)$$

Then, by the fact that A is a polynomial-time algorithm and by the reconstruction property of DP_k (Lemma 2.15; see also [HLO26, Corollary 24]), we would obtain

$$\text{pK}^{\text{poly}(t)}(y | x) \leq k + O(\log t),$$

which contradicts our choice of k , provided that q is chosen to be a sufficiently large polynomial.

In other words, the quantity in Equation (12) is at least $(1 - \delta) \cdot n^{-3} - n^{-4} > \delta$. It follows that there exists $z \in \{0, 1\}^{mk}$ such that

$$\ell\text{-pK}_{b/n^3, d/n^3}^{t+q(\tau), \text{SAT}}(x, \text{DP}_k(y, z)) \geq s. \quad (13)$$

Moreover, it can also be shown (see [HLO26, Lemma 70]) that, for q a sufficiently large polynomial,

$$\ell\text{-pK}_{b/n^3, d/n^3}^{t+q(\tau), \text{SAT}}(x, \text{DP}_k(y, z)) \leq \ell\text{-pK}_{b/n^3, d/n^3}^{t, \text{SAT}}(x, y) + mk + \log q(\tau). \quad (14)$$

Intuitively, this is because $\text{DP}_k(y, z)$ is efficiently computable given y and the seed z .

It follows from Equation (13) and Equation (14) that

$$\begin{aligned} \ell\text{-pK}_{b/n^3, d/n^3}^{t, \text{SAT}}(x, y) &\geq s - mk - \log q(\tau) \\ &= (\ell + mk + k) - \text{pK}_{(b-1)/n^3, (d-2)/n^3}^{t', \text{SAT}}(x) + k - \log \rho(t^4) - \log q(\tau) \\ &= (\ell + mk + k) - \text{pK}_{(b-1)/n^3, (d-2)/n^3}^{t+q(\tau)+\rho(\tau^5), \text{SAT}}(x) + \text{pK}^{q(t)}(y \mid x) \\ &\quad - \log q(t) - \log \rho(t^4) - \log q(\tau) \\ &\geq (\ell + \tau^3) - \text{pK}_{(b-1)/n^3, (d-2)/n^3}^{t+q(\tau^5)+\rho(\tau^5), \text{SAT}}(x) + \text{pK}^{q(t)}(y \mid x) \\ &\quad - \log q(t) - \log \rho(t^4) - \log q(\tau^5), \end{aligned}$$

where the last inequality uses the monotonicity of $\ell\text{-pK}^t$ in ℓ (see [HLO26, Lemma 67]), which roughly states that for any sufficiently large polynomial q , we have that for every $\ell < \ell'$, $x \in \{0, 1\}^*$ and $t' \in \mathbb{N}$,

$$\ell'\text{-pK}^{t'+q(|x|+|y|+\ell')}, \text{SAT}(x) \leq \ell\text{-pK}^{t', \text{SAT}}(x) + \log q(|x| + |y| + \ell').$$

Finally, we can conclude that there exists a polynomial p_2 such that

$$\ell\text{-pK}_{b/n^3, d/n^3}^{t, \text{SAT}}(x, y) \geq (\ell + \tau^3) - \text{pK}_{(b-1)/n^3, (d-2)/n^3}^{t+p_2(\tau), \text{SAT}}(x) + \text{pK}^{p_2(t)}(y \mid x) - \log p_2(t).$$

This completes the proof of the lower bound. $\diamond$

Putting It All Together. By combining the upper and lower bounds (Claim 3.9 and Claim 3.10) and rearranging, we obtain

$$\begin{aligned} \text{pK}^{p_2(t)}(y \mid x) &\leq \text{pK}_{1-1/n^3}^{\tau, \text{SAT}}(y \mid x) + \ell\text{-pK}_{b/n^3, (d+1)/n^3}^{t-p_1(\tau), \text{SAT}}(x) - (\ell + \tau^3) - \text{pK}_{(b-1)/n^3, (d-2)/n^3}^{t+p_2(\tau), \text{SAT}}(x) \\ &\quad + \log p_1(\tau) + \log p_2(t), \end{aligned}$$

which, by the monotonicity of $\ell\text{-pK}^t$ with respect to t , implies the existence of a polynomial p such that

$$\text{pK}^{p(t)}(y \mid x) \leq \text{pK}_{1-1/n^3}^{\tau, \text{SAT}}(y \mid x) + \ell\text{-pK}_{b/n^3, (d+1)/n^3}^{t-p(\tau), \text{SAT}}(x) - (\ell + \tau^3) - \text{pK}_{(b-1)/n^3, (d-2)/n^3}^{t+p(\tau), \text{SAT}}(x) + \log p(t),$$

as desired. $\square$

Proof Sketch of Lemma 3.4. The proof is essentially the same as that of [HLO26, Lemma 75]. We sketch the high-level ideas here and refer the reader to [HLO26] for full details.

Suppose $\text{Mild-Gap-MINpKT}^{\text{SAT}} \in \text{prBPP}$, which, by Lemma 3.7, implies $\text{Mild-Gap-MIN}\ell\text{-pKT}^{\text{SAT}} \in \text{prBPP}$. Given Lemma 3.8, we can apply a slightly more sophisticated telescoping sum argument than described earlier and show that for every $x \in \{0, 1\}^n$ and any τ that is a large enough polynomial in n , there exist t, ℓ, b, d that are at most $\text{poly}(n)$ such that

$$\ell\text{-pK}_{b/n^3, (d+1)/n^3}^{t-p(\tau), \text{SAT}}(x) - (\ell + \tau^3) - \text{pK}_{(b-1)/n^3, (d-2)/n^3}^{t+p(\tau), \text{SAT}}(x) + \log p(t)$$

is at most $O(\log n)$. Again, this can be achieved because the increase in the time bounds between the two quantities is only an *additive* $\text{poly}(n)$ term. As a result, for every x and y , we obtain

$$\text{pK}^{\text{poly}(n)}(y \mid x) \leq \text{pK}_{1-1/n^3}^{\tau, \text{SAT}}(y \mid x) + O(\log n). \quad (15)$$

Also, as mentioned earlier, $\text{pK}^{t, \text{SAT}}$ admits the language compression property (see [HLO26, Theorem 4]), that is, for every set A_x whose membership can be decided in polynomial time given x , there exists polynomial τ such that

$$\text{pK}_{1-1/n^3}^{\tau, \text{SAT}}(y \mid x) \leq \log |A_x| + O(\log n). \quad (16)$$

Now let $L \in \text{NP}$ and consider any $x \in \{0, 1\}^n$ that is a yes-instance of L . Let A_x be the set of L -witnesses for x . Then Equations (15) and (16) imply that for every $y \in A_x$,

$$\text{pK}^{p(n)}(y \mid x) \leq \log |A_x| + O(\log n).$$

As described earlier, this enables an efficient sampling procedure which, given x , outputs some element in A_x with non-trivial probability $1/\text{poly}(n)$. Repeating this procedure sufficiently many times then yields a witness for x with high probability. $\square$

3.2 Worst-Case Easiness of $\text{Gap-MINpKT}^{\text{PH}}$ from Average-Case Easiness of PH

Lemma 3.11. *If $\text{DistPH} \subseteq \text{AvgBPP}$, then $\text{Gap-MINpKT}^{\text{PH}} \in \text{prBPP}$.*

To show Lemma 3.11, we need the following.

Lemma 3.12. *For every oracle $\mathcal{O}$, if $(\text{MINKT}^{\mathcal{O}}, \text{PSAMP}) \subseteq \text{AvgBPP}$, then $\text{Gap-MINpKT}^{\mathcal{O}} \in \text{prBPP}$.*

Proof Sketch. The proof is similar to that of [HLO26, Proposition 58], using techniques from worst-case-to-average-case reductions developed by [Hir18]. $\square$

Proof of Lemma 3.11. Suppose $\text{DistPH} \subseteq \text{AvgBPP}$. Then for every oracle $\mathcal{O} \in \text{PH}$, $(\text{MINKT}^{\mathcal{O}}, \text{PSAMP}) \in \text{AvgBPP}$, since $\text{MINKT}^{\mathcal{O}} \in \text{PH}$. This, by Lemma 3.12, implies that $\text{Gap-MINpKT}^{\mathcal{O}} \in \text{prBPP}$. It follows that $\text{Gap-MINpKT}^{\text{PH}} \in \text{prBPP}$. $\square$

3.3 Average-Case Easiness of PH from Worst-Case Easiness of $\text{Gap-MINpKT}^{\text{PH}}$

In this subsection, we prove the following which shows the forward direction of Theorem 1.1.

Lemma 3.13. *If $\text{Gap-MINpKT}^{\text{PH}} \in \text{prBPP}$, then $\text{DistPH} \subseteq \text{AvgBPP}$.*

We begin with some technical tools.

3.3.1 Technical Tools

The following is the main technical lemma for this subsection.

Lemma 3.14. *Let $k_0 \in \mathbb{N}$, and let $\mathcal{O}$ be a $\Sigma_{k_0}^{\text{P}}$ -complete language. If there exist polynomials ρ and τ such that $\text{Gap}_{\rho}\text{-MINpKT}^{\mathcal{O}} \in \text{prBPTIME}[\tau(n)]$, then for every $k \leq k_0$, $L \in \Sigma_k^{\text{P}}$, and polynomial q , there exists a randomized algorithm A such that for every $\ell \in \mathbb{N}$ and $x \in \{0, 1\}^n$, with probability at least $1 - 2^{-\ell}$ (over the internal randomness of A) both of the following hold.*

1. $A(x, 1^\ell)$ runs in time

$$2^{O(\text{pK}^{q(n), \mathcal{O}}(x) - \text{K}(x))} \cdot \text{poly}(n, \ell),$$

where the constants hidden in $O(\cdot)$ and $\text{poly}(\cdot)$ depend only on $\mathcal{O}$, ρ , τ , k , L , and q .

2. $A(x, 1^\ell) = 1$ if and only if $x \in L$.

The rest of this subsection is devoted to proving [Lemma 3.14](#). For this, we need the following additional tools.

Lemma 3.15 (Symmetry of Information for pK^t ; see, e.g., [\[HKLO24, Lemma 36\]](#)). *Let $\mathcal{O}$ be any oracle. If $\text{Gap-MINpKT}^\mathcal{O} \in \text{prBPP}$, then there exist polynomials p_{Sol} and p_0 such that for all sufficiently large $x, y \in \{0, 1\}^*$ and every $t \geq p_0(|x| + |y|)$,*

$$\text{pK}^{p_{\text{Sol}}(t)}(y \mid x) \leq \text{pK}^t(x, y) - \text{pK}^{p_{\text{Sol}}(t)}(x) + \log p_{\text{Sol}}(t).$$

Proof Sketch. To get the desired symmetry of information property, it suffices have an efficient algorithm that accepts the set of strings with low K^{poly} -complexity while rejecting a large fraction of random strings. Note that such an algorithm follows from $\text{Gap-MINpKT}^\mathcal{O} \in \text{prBPP}$ for any $\mathcal{O}$. $\square$

Lemma 3.16. *Let $\mathcal{O}$ be any oracle. If $\text{Gap-MINpKT}^\mathcal{O} \in \text{prBPP}$, then there exists a polynomial p_1 such that for all $z \in \{0, 1\}^*$ and $t \in \mathbb{N}$,*

$$\text{pK}^{p_1(t)}(z) \leq \text{pK}^{t, \mathcal{O}}(z) + \log p_1(t).$$

Proof Sketch. The proof can be easily adapted from that of [\[HLO26, Lemma 56\]](#). $\square$

We are now ready to prove [Lemma 3.14](#).

Proof of Lemma 3.14. Let $k_0 \in \mathbb{N}$ and let $\mathcal{O}$ be a $\Sigma_{k_0}^{\text{P}}$ -complete language. Suppose $\text{Gap}_\rho\text{-MINpKT}^\mathcal{O} \in \text{prBPTIME}[\tau(n)]$ for some polynomials ρ and τ . The proof is by induction on k .

The base case for $k = 0$ is trivial.

For the induction step, suppose the claimed conclusion holds for $k - 1$, where $1 \leq k \leq k_0$. Let $L \in \Sigma_k^{\text{P}}$ and let V be a corresponding verifier. That is, for $x \in \{0, 1\}^n$, $x \in L$ if and only if

$$\exists y_1 \in \{0, 1\}^{v(n)}, \forall y_2 \in \{0, 1\}^{v(n)}, \dots, Q^k y_k \in \{0, 1\}^{v(n)}, V(x, y_1, y_2, \dots, y_k) = 1,$$

where v is a polynomial and $Q^k \in \{\exists, \forall\}$. Let q be a polynomial.

To decide L , we aim to find a witness for a given instance, if one exists. Consider a yes-instance $x \in \{0, 1\}^n$ of L . Let y_1^* be the lexicographically first witness for x , i.e., y_1^* is the lexicographically smallest string in $\{0, 1\}^{v(n)}$ that satisfies

$$\forall y_2 \in \{0, 1\}^{v(n)}, \dots, Q^k y_k \in \{0, 1\}^{v(n)}, V(x, y_1, y_2, \dots, y_k) = 1,$$

where $Q^k \in \{\exists, \forall\}$. Let t be a sufficiently large polynomial specified later. We first observe the following.

Claim 3.17. *There exists a polynomial p such that*

$$\text{pK}^{p(t(n))}(y_1^* \mid x) \leq \text{pK}^{t(n), \mathcal{O}}(x) - \text{pK}^{p(t(n)), \mathcal{O}}(x) + \log p(t(n)).$$

Proof of Claim 3.17. Note that by Lemma 3.15 and the assumption that Gap-MINpKT^O is easy, we get symmetry of information for pK^t. Now let p_1 be the polynomial from Lemma 3.16 and let $t := t(n)$. We have

$$\begin{aligned}
& \text{pK}^{p_{\text{Sol}}(p_1(t+\text{poly}(n)))}(y_1^* \mid x) \\
& \leq \text{pK}^{p_1(t+\text{poly}(n))}(x, y_1^*) - \text{pK}^{p_{\text{Sol}}(p_1(t+\text{poly}(n)))}(x) + \log p_{\text{Sol}}(p_1(t + \text{poly}(n))) \\
& \leq \text{pK}^{t+\text{poly}(n), \mathcal{O}}(x, y_1^*) - \text{pK}^{p_{\text{Sol}}(p_1(t+\text{poly}(n)))}(x) + \log p_{\text{Sol}}^2(p_1(t + \text{poly}(n))) \quad (\text{by Lemma 3.16}) \\
& \leq \text{pK}^{\text{poly}(n), \mathcal{O}}(y_1^* \mid x) + \text{pK}^{t, \mathcal{O}}(x) - \text{pK}^{p_{\text{Sol}}(p_1(t+\text{poly}(n)))}(x) + \log p_{\text{Sol}}^3(p_1(t + \text{poly}(n))) \\
& \leq \text{pK}^{\text{poly}(n), \mathcal{O}}(y_1^* \mid x) + \text{pK}^{t, \mathcal{O}}(x) - \text{pK}^{p_{\text{Sol}}(p_1(t+\text{poly}(n))), \mathcal{O}}(x) + \log p_{\text{Sol}}^3(p_1(t + \text{poly}(n))).
\end{aligned}$$

Let p_2 be a sufficiently large polynomial, the above yields

$$\text{pK}^{p_2(t)}(y_1^* \mid x) \leq \text{pK}^{\text{poly}(n), \mathcal{O}}(y_1^* \mid x) + \text{pK}^{t, \mathcal{O}}(x) - \text{pK}^{p_2(t), \mathcal{O}}(x) + \log p_2(t),$$

Now note that $\text{pK}^{\text{poly}(n), \mathcal{O}}(y_1^* \mid x) \leq O(1)$ since given x and oracle access to $\mathcal{O}$, one can recover the lexicographically first witness for x in time $\text{poly}(n)$. Therefore, we get

$$\text{pK}^{p_2(t)}(y_1^* \mid x) \leq O(1) + \text{pK}^{t, \mathcal{O}}(x) - \text{pK}^{p_2(t), \mathcal{O}}(x) + \log p_2(t).$$

The claims follows by choosing p to be a sufficiently large polynomial. $\diamond$

We will generate a set of candidate strings that is guaranteed to include a witness for x (specifically, y_1^*) if $x \in L$. Given this, it suffices to solve the task of checking whether a given string is a witness for x . This is achieved using the language $L' \in \Sigma_{k-1}^P$ defined as follows.

$$(x, y_1) \in L' \text{ if and only if } \exists y_2, \dots, Q^k y_k, V(x, y_1, y_2, \dots, y_k) = 0, \text{ where } Q^k \in \{\exists, \forall\}.$$

Let $q' > q$ be a sufficiently large polynomial specified later. We have the following.

Claim 3.18. *There exists a randomized algorithm A' such that every $\ell' \in \mathbb{N}$, $x \in \{0, 1\}^n$ and $y_1 \in \{0, 1\}^{v(n)}$, with probability at least $1 - 2^{-\ell'}$ (over the internal randomness of A') both of the following hold.*

1. $A'((x, y_1), 1^{\ell'})$ runs in time

$$2^{O(\text{pK}^{q'(n+v(n)), \mathcal{O}}(x, y_1) - \text{K}(x, y_1))} \cdot \text{poly}(\ell', n).$$

2. $A'((x, y_1), 1^{\ell'}) = 1$ if and only if $(x, y_1) \in L'$.

Proof of Claim 3.18. This follows directly from our induction hypothesis. $\diamond$

In order to generate a set of “good” candidate witnesses, we will also need the following subroutines.

Claim 3.19. *There exists a randomized polynomial-time algorithm Approx-depth such that, for every $x \in \{0, 1\}^n$ and $\ell' \in \mathbb{N}$, Approx-depth(x, ℓ') outputs, with probability at least $1 - 2^{-\ell'}$, an integer s satisfying*

$$\text{pK}^{t(n), \mathcal{O}}(x) - \text{pK}^{p(t(n)), \mathcal{O}}(x) \leq s \leq \text{pK}^{\rho^{-1}(t(n)), \mathcal{O}}(x) - \text{pK}^{\rho(p(t(n))), \mathcal{O}}(x) + \log t(n) + \log \rho(p(t(n))),$$

where p is the polynomial defined in Claim 3.17.

Proof of Claim 3.19. Let Approx be a randomized polynomial-time algorithm that satisfies the following: Given $z \in \{0, 1\}^m$, $1^{t'}$ and 1^κ , $\text{Approx}(z, 1^{t'}, 1^\kappa)$ outputs, with probability at least $1 - 2^{-\kappa}$, an integer s_0 such that

$$\text{pK}^{\rho(t'), \mathcal{O}}(z) - \log \rho(t') \leq s_0 \leq \text{pK}^{t', \mathcal{O}}(z).$$

It is easy to see that such an algorithm Approx can be obtained from the assumption that $\text{Gap}_\rho\text{-MINpKT}^\mathcal{O} \in \text{prBPP}$ and using success amplification techniques. (See, e.g., [GKLO22, Lemma 28].)

Let $t := t(n)$. By running $\text{Approx}(x, 1^{\rho^{-1}(t)}, 1^{\ell'+1})$, with probability at least $1 - 2^{-\ell'}/2$, we get an integer s_1 such that

$$\text{pK}^{t, \mathcal{O}}(x) - \log t \leq s_1 \leq \text{pK}^{\rho^{-1}(t), \mathcal{O}}(x).$$

Similarly, by running $\text{Approx}(x, 1^{p(t)}, 1^{\ell'+1})$, with probability at least $1 - 2^{-\ell'}/2$, we get some integer s_2 such that

$$\text{pK}^{\rho(p(t)), \mathcal{O}}(x) - \log \rho(p(t)) \leq s_2 \leq \text{pK}^{p(t), \mathcal{O}}(x).$$

Then by a union bound, with probability at least $1 - 2^{-\ell'}$, we have

$$s_1 - s_2 \leq \text{pK}^{\rho^{-1}(t), \mathcal{O}}(x) - \left(\text{pK}^{\rho(p(t)), \mathcal{O}}(x) - \log \rho(p(t)) \right) \quad (17)$$

and

$$s_1 - s_2 \geq (\text{pK}^{t, \mathcal{O}}(x) - \log(t)) - \text{pK}^{p(t), \mathcal{O}}(x). \quad (18)$$

We can then output

$$s := s_1 - s_2 + \log(t).$$

It is easy to verify that s satisfies the stated condition in the claim. $\diamond$

Claim 3.20. *There exist a randomized polynomial-time algorithm Valid and a constant C such that for every $(x, y_1, 1^{\ell'}) \in \{0, 1\}^n \times \{0, 1\}^{v(n)} \times \mathbb{N}$, with probability at least $1 - 2^{-\ell'}$ (over the internal randomness of Valid) both of the following hold:*

1. *If $y_1 = y_1^*$, then $\text{Valid}(x, y_1, 1^{\ell'}) = 1$.*
2. *If $\text{Valid}(x, y_1, 1^{\ell'}) = 1$, then it holds that*

$$\text{pK}^{q'(n+v(n)), \mathcal{O}}(x, y_1) \leq \text{pK}^{q(n), \mathcal{O}}(x) + \log q'(n+v(n)) + \log \rho(q(n)) + C \cdot \log n.$$

Proof of Claim 3.20. Again, let Approx be a randomized polynomial-time algorithm that satisfies the following: Given $z \in \{0, 1\}^m$, $1^{t'}$ and 1^κ , $\text{Approx}(z, 1^{t'}, 1^\kappa)$ outputs an integer s such that

$$\text{pK}^{\rho(t'), \mathcal{O}}(z) - \log \rho(t') \leq s \leq \text{pK}^{t', \mathcal{O}}(z),$$

with probability at least $1 - 2^{-\kappa}$, where ρ is the polynomial such that $\text{Gap}_\rho\text{-MINpKT}^\mathcal{O} \in \text{prBPP}$.

Let Valid be the following algorithm, where C is a constant to be determined later.

Given $(x, y_1, 1^{\ell'})$, let

$$\theta := \text{Approx}\left(x, 1^{q(n)}, 1^{\ell'+1}\right),$$

and

$$\mu := \text{Approx}\left((x, y_1), 1^{\rho^{-1}(q'(n+v(n)))}, 1^{\ell'+1}\right).$$

We accept if and only if $\mu \leq \theta + \log \rho(q(n)) + C \cdot \log n$.

We argue that the above algorithm satisfies the two conditions stated in the claim. Indeed, by the correctness of Approx and a union bound, we have that, with probability at least $1 - 2^{-\ell'}$, both the following two hold.

$$\mathsf{pK}^{\rho(q(n)), \mathcal{O}}(x) - \log \rho(q(n)) \leq \theta \leq \mathsf{pK}^{q(n), \mathcal{O}}(x),$$

and

$$\mathsf{pK}^{q'(n+v(n)), \mathcal{O}}(x, y_1) - \log q'(n + v(n)) \leq \mu \leq \mathsf{pK}^{\rho^{-1}(q'(n+v(n))), \mathcal{O}}(x, y_1).$$

Now to see that $\text{Valid}(x, y_1^*, 1^\ell) = 1$, observe that in this case

$$\begin{aligned} \mu &\leq \mathsf{pK}^{\rho^{-1}(q'(n+v(n))), \mathcal{O}}(x, y_1^*) \\ &\leq \mathsf{pK}^{\rho(q(n)), \mathcal{O}}(x) + C \cdot \log n \\ &\leq \theta + \log \rho(q(n)) + C \cdot \log n, \end{aligned}$$

where the second inequality uses the facts that q' is a sufficiently large polynomial and that given oracle access to $\mathcal{O}$, one can recover y_1^* in polynomial time given x with a program of length $C \cdot \log n$, for some constant C .

For the second condition, if $\text{Valid}(x, y_1, 1^\ell) = 1$, then it implies that

$$\begin{aligned} \mathsf{pK}^{q'(n+v(n)), \mathcal{O}}(x, y_1) - \log q'(n + v(n)) &\leq \mu \\ &\leq \theta + \log \rho(q(n)) + C \cdot \log n \\ &\leq \mathsf{pK}^{q(n), \mathcal{O}}(x) + \log \rho(q(n)) + C \cdot \log n. \end{aligned}$$

This completes the proof. $\diamond$

We are now ready to describe our algorithm for deciding L .

Algorithm 1 An algorithm for deciding L

```

1: procedure  $A(x, 1^\ell)$ 
2:    $n := |x|$ .
3:    $s := \text{Approx-depth}(x, 1^{\ell+3})$ .
4:    $s' := s + \log p(t(n))$ .
5:
6:   for  $i \in [\ell]$  do
7:      $r :=$  a uniformly random string in  $\{0, 1\}^{p(t(n))}$ .
8:     for  $\Pi \in \{0, 1\}^{\leq s'}$  do
9:        $y_1 :=$  the output of  $\Pi(x; r)$  after running  $p(t(n))$  steps.
10:      if  $|y_1| = v(n)$  and  $\text{Valid}(x, y_1, 1^{\ell+\log(\ell)+s'+5}) = 1$  then
11:        if  $A'(x, y_1, 1^{\ell+\log(\ell)+s'+5}) = 0$  then
12:          Output True.
13:   Output False.

```

Correctness. We argue that the algorithm A correctly decides L . Fix $x \in \{0, 1\}^n$ and $\ell \in \mathbb{N}$.

First, we assume that the (randomized) algorithm Approx-depth outputs a correct answer for x —that is, it satisfies the condition stated in Claim 3.19. Note that this occurs with probability at least $1 - 2^{-\ell}/8$, due to the choice of the second parameter in the invocation of Approx-depth. We also assume that both algorithms, Valid and A' , succeed (meaning that the conditions in Claim 3.20 and Claim 3.18, respectively, are satisfied)

in all of their at most $\ell \cdot 2^{s'+1}$ executions. By a union bound and the parameter choices in their invocations, this occurs with probability at least $1 - 2^{-\ell}/8$.

Now consider a single execution of the for loop at Line 6 (i.e., Lines 7–12). Suppose $x \notin L$. Note that, in this case, by the definition of the language L' and the fact that A' correctly decides L' (Claim 3.18), the condition at Line 11 will not be satisfied for any y_1 . Therefore, Line 12 will never be reached in this case.

On the other hand, if $x \in L$, then by Claim 3.17, with probability at least $2/3$ over $r \sim \{0, 1\}^{p(t(n))}$, the multiset

$$\mathcal{S} := \left\{ y_1 \mid y_1 \text{ is the output of } \Pi(x; r) \text{ after } p(t(n)) \text{ steps for some } \Pi \in \{0, 1\}^{\leq s'} \right\}, \quad (19)$$

generated by Line 9, contains y_1^* , which is the lexicographically first witness for x . To see this, observe that if the algorithm Approx-depth outputs a correct answer s for x , then we have

$$\begin{aligned} \text{pK}^{p(t(n))}(y_1^* \mid x) &\leq \text{pK}^{t(n), \mathcal{O}}(x) - \text{pK}^{p(t(n)), \mathcal{O}}(x) + \log p(t(n)) && \text{(by Claim 3.17)} \\ &\leq s + \log p(t(n)) && \text{(by Claim 3.19)} \\ &\leq s'. \end{aligned}$$

Consider the case when $y_1 = y_1^*$. It follows from Claim 3.20 (Item 1) that the condition at Line 10 is satisfied. Also, since y_1^* is a witness for x , by the definition of L' , we have $(x, y_1^*) \notin L'$, and hence, by the fact that A' decides L' (Claim 3.18), the condition at Line 11 will be satisfied. Therefore, True will be output in this case.

Since the above process is executed ℓ times in the for loop, the probability that it never outputs True when $x \in L$ is at most $1/3^\ell$.

To conclude, with probability at least

$$1 - \frac{1}{8 \cdot 2^\ell} - \frac{1}{8 \cdot 2^\ell} - \frac{1}{3^\ell} > 1 - \frac{1}{2^\ell},$$

the algorithm will correctly decide whether $x \in L$.

Running Time. We now analyze the running time of the algorithm. Again, we assume that the (randomized) algorithms Approx-depth, Valid, and A' succeed in all their executions, which occurs with probability at least $1 - 2^{-\ell}$.

Consider a single execution of the for loop at Line 6. Define the following sub-multiset of $\mathcal{S}$ (where $\mathcal{S}$ is defined in Equation (19)):

$$\mathcal{S}' := \{y_1 \in \mathcal{S} \mid y_1 \text{ satisfies the condition at Line 10}\}.$$

Now, it is easy to see that a single execution (Lines 7–12) takes time at most

$$2^{s'+1} \cdot \text{poly}(t(n)) + \sum_{y_1 \in \mathcal{S}'} \text{Runtime}\left(A'(x, y_1, 1^{\ell'})\right), \quad (20)$$

where $\ell' = \ell + \log(\ell) + s' + 5$.

Note that for every $y_1 \in \mathcal{S}'$, it follows from Claim 3.20 (Item 2) that

$$\text{pK}^{q'(n+v(n)), \mathcal{O}}(x, y_1) \leq \text{pK}^{q(n), \mathcal{O}}(x) + \log q'(n + v(n)) + \log \rho(q(n)) + C \cdot \log n. \quad (21)$$

Therefore, for every such y_1 , we have

$$\begin{aligned}
\text{Runtime}\left(A'(x, y_1, 1^{\ell'})\right) &\leq \exp\left(O\left(\text{pK}^{q'(n+v(n)), \mathcal{O}}(x, y_1) - \text{K}(x, y_1)\right)\right) \cdot \text{poly}(n, \ell) \quad (\text{by Claim 3.18}) \\
&\leq \exp\left(O\left(\text{pK}^{q(n), \mathcal{O}}(x) - \text{K}(x, y_1)\right)\right) \cdot \text{poly}(n, \ell) \quad (\text{by Equation (21)}) \\
&\leq \exp\left(O\left(\text{pK}^{q(n), \mathcal{O}}(x) - \text{K}(x)\right)\right) \cdot \text{poly}(n, \ell)
\end{aligned}$$

It follows that the quantity in Equation (20) is at most

$$\begin{aligned}
&2^{s'+1} \cdot \text{poly}(t(n)) + 2^{s'+1} \cdot \exp\left(O\left(\text{pK}^{q(n), \mathcal{O}}(x) - \text{K}(x)\right)\right) \cdot \text{poly}(n, \ell) \quad (\text{by } |S'| \leq 2^{s'+1}) \\
&\leq \exp(s') \cdot \exp\left(O\left(\text{pK}^{q(n), \mathcal{O}}(x) - \text{K}(x)\right)\right) \cdot \text{poly}(n, \ell) \\
&\leq \exp\left(\text{pK}^{\rho^{-1}(t(n)), \mathcal{O}}(x) - \text{pK}^{\rho(p(t(n))), \mathcal{O}}(x) + \log t(n) + \log \rho(p(t(n))) + \log p(t(n))\right) \\
&\quad \cdot \exp\left(O\left(\text{pK}^{q(n), \mathcal{O}}(x) - \text{K}(x)\right)\right) \cdot \text{poly}(n, \ell) \quad (\text{by Claim 3.19}) \\
&\leq \exp\left(\text{pK}^{\rho^{-1}(t(n)), \mathcal{O}}(x) - \text{K}(x) + O(\log n)\right) \cdot \exp\left(O\left(\text{pK}^{q(n), \mathcal{O}}(x) - \text{K}(x)\right)\right) \cdot \text{poly}(n, \ell) \\
&\quad (\text{by Lemma 2.12}) \\
&\leq \exp\left(\text{pK}^{q(n), \mathcal{O}}(x) - \text{K}(x)\right) \cdot \exp\left(O\left(\text{pK}^{q(n), \mathcal{O}}(x) - \text{K}(x)\right)\right) \cdot \text{poly}(n, \ell) \\
&\leq \exp\left(O\left(\text{pK}^{q(n), \mathcal{O}}(x) - \text{K}(x)\right)\right) \cdot \text{poly}(n, \ell),
\end{aligned}$$

where the second last inequality holds by letting $t(\cdot) \geq \rho(q(\cdot))$. This completes the proof. $\square$

3.3.2 Proof of Lemma 3.13

We are now ready to show Lemma 3.13.

Proof of Lemma 3.13. Assume $\text{Gap-MINpKT}^{\text{PH}} \in \text{prBPP}$. Let $k \in \mathbb{N}$ and $L \in \Sigma_k^{\text{P}}$.

Given Theorem 2.18, it suffices to show how to decide L on average over the uniform distribution. The idea is to obtain a heuristic scheme by using the worst-case algorithm from Lemma 3.14 for deciding L , along with the fact that $\text{pK}^{t, \text{PH}}(x) - \text{K}(x)$ is small for a uniformly random $x \in \{0, 1\}^n$. Details follow.

Let $\mathcal{O}$ be a Σ_k^{P} -complete language, and let A be the algorithm from Lemma 3.14 that satisfies the following. Given $\ell \in \mathbb{N}$ and $x \in \{0, 1\}^n$, with probability at least $1 - 2^{-\ell}$, $A(x, 1^\ell)$ runs in time

$$2^{O(\text{pK}^{q(n), \mathcal{O}}(x) - \text{K}(x))} \cdot \text{poly}(n, \ell)$$

and decides whether $x \in L$, where q is a polynomial satisfying that for all $x \in \{0, 1\}^*$, $\text{pK}^{q(|x|), \mathcal{O}}(x) \leq |x| + O(\log |x|)$.

We now describe a heuristic scheme for solving L over the uniform distribution. Let B be the following algorithm:

On input $(x, 1^n, 1^\kappa)$, where $|x| = n$, set cutoff $:= p(n, \kappa)$, where p is a polynomial specified later. Run $A(x, 1^3)$ for cutoff steps. If it halts, output the returned answer; otherwise output $\perp$.

We argue the correctness of B . By the fact that for every $x \in \{0, 1\}^n$, $A(x, 1^3)$ only errs with probability at most $2^{-3} = 1/8$ (over the internal randomness of A), we get that

$$\Pr_B[B(x, 1^n, 1^\kappa) \in \{L(x), \perp\}] \geq \frac{4}{5}.$$

It remains to show that

$$\Pr_{x \sim \{0,1\}^n} \left[\Pr_B[B(x, 1^n, 1^\kappa) = \perp] < \frac{1}{5} \right] \geq 1 - \frac{1}{\kappa}.$$

Note that by construction, it is sufficient to show that with probability at least $1 - 1/\kappa$ over $x \sim \{0, 1\}^n$, we have

$$\Pr_A[\text{Runtime}(A(x, 1^3)) \leq \text{cutoff}] > \frac{4}{5}.$$

First of all, for every $x \in \{0, 1\}^n$, we have that with probability at least $7/8$ (which is greater than $4/5$) over its internal randomness, $A(x, 1^3)$ runs in time

$$2^{O(\text{pK}^{q(n), \mathcal{O}}(x) - \text{K}(x))} \cdot \text{poly}(n). \quad (22)$$

Firstly, for every $x \in \{0, 1\}^n$, we have

$$\text{pK}^{q(n), \mathcal{O}}(x) \leq n + O(\log n).$$

Secondly, by [Lemma 2.11](#), with probability at least $1 - 1/\kappa$ over $x \sim \{0, 1\}^n$,

$$\text{K}(x) \geq n - \log \kappa.$$

It follows that with probability at least $1 - 1/\kappa$ over $x \sim \{0, 1\}^n$, the runtime given by [Equation \(22\)](#) is at most

$$\exp(n + O(\log n) - n + \log \kappa) \cdot \text{poly}(n) \leq \text{poly}(n, \kappa).$$

By letting p be a sufficiently large polynomial, the above is at most cutoff. This completes the proof. $\square$

4 The Deterministic Case

Theorem 4.1. *The following statements are equivalent:*

1. (Exclusion of PH-Heuristica). $\text{DistPH} \subseteq \text{AvgP} \implies \text{PH} = \text{P}$.
2. (Meta-Complexity of K^{PH}). $\text{Gap-MINKT}^{\text{PH}} \in \text{P} \implies \text{Mild-Gap-MINKT}^{\text{PH}} \in \text{P}$.

As in the proof of [Theorem 1.1](#), [Theorem 4.1](#) follows from the following two results.

Theorem 4.2 ([\[Hir20\]](#)). *The following holds:*

$$\text{DistPH} \subseteq \text{AvgP} \iff \text{Gap-MINKT}^{\text{PH}} \in \text{P}.$$

Theorem 4.3. *The following holds:*

$$\text{NP} = \text{P} \iff \text{Mild-Gap-MINKT}^{\text{PH}} \in \text{P}.$$

To prove [Theorem 4.3](#), we first establish the following.

Lemma 4.4. *If $\text{Mild-Gap-MINKT}^{\text{SAT}} \in \text{BPP}$, then $\text{NP} \subseteq \text{BPP}$.*

Proof Sketch. The proof is essentially the same as that of [Lemma 3.4](#). It suffices to consider the notion $\ell\text{-K}^{t, \text{SAT}}$ in place of $\ell\text{-pK}^{t, \text{SAT}}$. See also the proof of [\[HLO26, Theorem 34\]](#) for details. $\square$

Proof of Theorem 4.3. The forward direction is immediate since $\text{Mild-Gap-MINKT}^{\text{PH}}$ lies in PH. Note that $\text{PH} = \text{P}$ if $\text{NP} = \text{P}$.

For the backward direction, first note that Lemma 4.4 implies the following.

$$\text{Mild-Gap-MINKT}^{\text{PH}} \in \text{P} \implies \text{NP} \subseteq \text{BPP}.$$

Moreover, as mentioned in Section 1.3, [Hir20] shows that if $\text{Gap-MINKT}^{\text{PH}} \in \text{P}$, then $\text{BPP} = \text{P}$ (see [Hir20, Theorem 1.17]). It follows that if $\text{Mild-Gap-MINKT}^{\text{PH}} \in \text{P}$, then we have both $\text{NP} \subseteq \text{BPP}$ and $\text{BPP} = \text{P}$, which together imply $\text{NP} = \text{P}$, as desired. $\square$

5 Equivalence of Gap and No-Gap Hardness in the Average-Case Setting

In this section, we show that, in the average-case setting (both zero-error and two-sided-error), the hardness of the standard gap version of $\text{MINpKT}^{\text{PH}}$ is equivalent to that of its no-gap variant.

5.1 The Zero-Error Case

We first prove the zero-error case.

Theorem 5.1. *The following holds:*

$$\text{“Gap-MINpKT}^{\text{PH}} \in \text{AvgBPP} \text{”} \iff \text{“MINpKT}^{\text{PH}} \in \text{AvgBPP} \text{”}.$$

We will need the following worst-case to average-case reduction result for approximating pK^t .

Theorem 5.2 (Following [Hir18, GKLO22]). *Let $\mathcal{O}$ be any oracle. If “Gap-MINpKT $^{\mathcal{O}}$ $\in$ AvgBPP”, then $\text{Gap-MINpKT}^{\mathcal{O}} \in \text{prBPP}$.*

We are now ready to show Theorem 5.1.

Proof of Theorem 5.1. The reverse implication is immediate. We therefore prove the forward implication.

Assume that

$$\text{“Gap-MINpKT}^{\text{PH}} \in \text{AvgBPP} \text{”}.$$

By Theorem 5.2, this implies

$$\text{Gap-MINpKT}^{\text{PH}} \in \text{prBPP}.$$

Then, by Lemma 3.13, we obtain

$$\text{DistPH} \subseteq \text{AvgBPP}.$$

Consequently, it suffices to establish the following claim.

Claim 5.3. *If $\text{DistPH} \subseteq \text{AvgBPP}$, then “MINpKT $^{\text{PH}}$ $\in$ AvgBPP”.*

Proof of Claim 5.3. Assume $\text{DistPH} \subseteq \text{AvgBPP}$. Fix an oracle $\mathcal{O} \in \text{PH}$ and let $\{\mathcal{D}_n\}$ be any polynomial-time samplable distribution family. We show that $\text{MINpKT}^{\mathcal{O}}$ is solvable with zero error on average with respect to $\{\mathcal{D}_n\}$.

First, we observe that $\text{MINpKT}^{\mathcal{O}} \in \text{promise-AM}^{\text{PH}}$. Consider the following protocol on input $(x, 1^s, 1^t, 1^a, 1^b)$:

The verifier samples a random string $w \in \{0, 1\}^t$ and sends it to the prover. The prover responds with an oracle program $M_w^{(\cdot)}$. The verifier then simulates $M_w^{\mathcal{O}}(w)$ for at most t steps and accepts iff

$$|M_w| \leq s \quad \text{and} \quad M_w^{\mathcal{O}}(w) = x.$$

If $\text{pK}_{b/a}^{t,O}(x) \leq s$, then there exists a strategy for the prover that causes the verifier to accept with probability at least b/a . Conversely, if $\text{pK}_{(b-1)/a}^{t,O}(x) > s$, then every prover causes acceptance with probability less than $(b-1)/a$. Standard error reduction yields a promise- AM^{PH} protocol for MINpKT^O .

Since $\text{promise-AM}^{\text{PH}} \subseteq \text{promise-PH}$, it follows that $\text{MINpKT}^O \in \text{promise-PH}$. Hence there exists a PH predicate B that correctly decides MINpKT^O on the promise inputs.

Now consider the family of distributions $\mathcal{D}'_{\langle n,s,t,a,b \rangle}$, where for each triple $\langle n, s, t, a, b \rangle$, we sample $x \sim \mathcal{D}_n$ and output $(x, 1^s, 1^t, 1^a, 1^b)$.

By the assumption $\text{DistPH} \subseteq \text{AvgBPP}$, there exists a zero-error heuristic scheme A that solves B under $\{\mathcal{D}'_{\langle n,s,t,a,b \rangle}\}$. It is easy to see that A is also a zero-error heuristic scheme for MINpKT^O . $\diamond$

The theorem follows immediately from the claim. $\square$

5.2 The Two-Sided-Error Case

Here we show that the hardness of the standard gap version of $\text{MINpKT}^{\text{PH}}$ is equivalent to that of its no-gap variant in the two-sided-error average-case setting.

Theorem 5.4. *The following holds:*

$$\text{“Gap-MINpKT}^{\text{PH}} \in \text{HeurBPP”} \iff \text{“MINpKT}^{\text{PH}} \in \text{HeurBPP”}.$$

We will need the following result.

Theorem 5.5 ([HS22]). *The following statements are equivalent.*¹¹

1. $\text{DistPH} \not\subseteq \text{HeurBPP}$.
2. *There exists a language $f \in \text{PH}$ and a constant $d > 0$ such that, for every constant $c > 0$, there exists a pseudorandom generator*

$$G = \{G_n : \{0, 1\}^n \rightarrow \{0, 1\}^{n+c \log n}\}_{n \in \mathbb{N}}$$

satisfying the following conditions:

- *G is infinitely often secure against probabilistic polynomial-time algorithms. That is, for every probabilistic polynomial-time algorithm A and every polynomial p , there exist infinitely many n such that*

$$\left| \Pr_{r \sim \{0,1\}^n} [A(G_n(r)) = 1] - \Pr_{x \sim \{0,1\}^{n+c \log n}} [A(x) = 1] \right| \leq \frac{1}{p(n)}.$$

- *For all sufficiently large n , G_n is computable in time n^d with oracle access to f .*
- *G is min-entropy-preserving in the sense that for every $n \in \mathbb{N}$ and $y \in \text{Support}(G_n)$, $\Pr_{r \sim \{0,1\}^n} [G_n(r) = y] = 2^{-n}$, i.e., $H_\infty(G_n(\mathcal{U}_n)) = n$.*¹²

We are now ready to prove [Theorem 5.4](#).

¹¹This formulation differs slightly from the original statement in [HS22]. More precisely, here we use the same language f for every choice of c . It is not hard to verify that this version of the statement follows from the proofs in that paper.

¹²In fact, the PRG is based on the direct product generator in which the seed is part of the output; in particular, the PRG is seed-extending.

Proof of Theorem 5.4. Since the backward direction is immediate, we proceed to prove the forward direction.

Assume that “Gap-MINpKT^{PH} ∈ HeurBPP”. We first show that infinitely-often secure PH-computable pseudorandom generators with arbitrary logarithmic stretch do not exist.

Claim 5.6. *If Gap-MINpKT^{PH} ∈ HeurBPP, then Item 2 in Theorem 5.5 is false.*

Proof of Claim 5.6. In the following, we always consider the Gap-MINpKT problem with input parameters $a = 3$ and $b = 2$, and omit these parameters from the notation.

For the sake of contradiction, suppose that both “Gap-MINpKT^{PH} ∈ HeurBPP” and Item 2 of Theorem 5.5 hold. Let

- $f \in \text{PH}$ be the language guaranteed by Item 2 of Theorem 5.5;
- ρ be a polynomial such that $\text{Gap}_\rho\text{-MINpKT}^f \in \text{HeurBPP}$;
- $\alpha > \rho$ be a polynomial specified later, and let

$$G = \{G_n : \{0, 1\}^n \rightarrow \{0, 1\}^{m(n)}\}$$

be the pseudorandom generator guaranteed by Item 2 of Theorem 5.5, where

$$m(n) := n + \log \alpha(n),$$

and for all sufficiently large n , G_n is computable in time $t(n) := n^d$ for some constant $d > 0$ that is independent of α ;

- $\mathcal{D} = \{D_{m(n)}\}$ where each $D_{m(n)}$ samples from $G_n(\mathcal{U}_n)$.

Next, starting from an algorithm witnessing the easiness of $\text{Gap}_\rho\text{-MINpKT}^f$ and applying standard error-reduction techniques, we obtain a probabilistic polynomial-time algorithm A such that¹³

$$\Pr_{x \sim \{0,1\}^{m(n)}, A} \left[A \text{ decides } \text{Gap}_\rho\text{-MINpKT}^f \text{ on } (x, 1^{s(n)}, 1^{t(n)}) \right] \geq 1 - \frac{1}{\alpha(n)^2}, \quad (23)$$

where $s(n) := n + \gamma \log n$ for some absolute constant $\gamma > 0$ specified later, and $t(n)$ is specified as above.

We will construct from A an efficient algorithm that distinguishes $G_n(\mathcal{U}_n)$ from $\mathcal{U}_{m(n)}$ for every sufficiently large n . Fix such an n for the remainder of the proof.

First, note that for every $y \in \text{Support}(D_{m(n)})$, we have $D_{m(n)}(y) \leq 2^{-n} = 2^{-m} \cdot \alpha(n)$. In other words, $\mathcal{U}_{m(n)}$ dominates $D_{m(n)}$, and therefore

$$\begin{aligned} & \Pr_{y \sim D_{m(n)}, A} \left[A \text{ fails to decide } \text{Gap}_\rho\text{-MINpKT}^f \text{ on } (y, 1^{s(n)}, 1^{t(n)}) \right] \\ & \leq \Pr_{x \sim \mathcal{U}_{m(n)}, A} \left[A \text{ fails to decide } \text{Gap}_\rho\text{-MINpKT}^f \text{ on } (x, 1^{s(n)}, 1^{t(n)}) \right] \cdot \alpha(n) \\ & \leq \frac{1}{\alpha(n)^2} \cdot \alpha(n) \quad (\text{by Equation (23)}) \\ & = \frac{1}{\alpha(n)}. \end{aligned} \quad (24)$$

¹³We fix the probability threshold b/a to $2/3$ and omit these input parameters in the subsequent discussion.

Also, we have for every $r \in \{0, 1\}^n$,

$$\text{pK}^{t(n),f}(G_n(r)) \leq n + O(\log n) \leq s(n).$$

Hence every such $(G_n(r), 1^{s(n)}, 1^{t(n)})$ is a YES instance of $\text{Gap}_\rho\text{-MINpKT}^f$. It follows that

$$\begin{aligned} 1 - \frac{1}{\alpha(n)} &\leq \Pr_{y \sim D_{m(n),A}} \left[A \text{ decides } \text{Gap}_\rho\text{-MINpKT}^f \text{ on } (y, 1^{s(n)}, 1^{t(n)}) \right] && \text{(by Equation (24))} \\ &= \Pr_{r \sim \{0,1\}^n, A} \left[A(G_n(r), 1^{s(n)}, 1^{t(n)}) = 1 \right]. \end{aligned} \quad (25)$$

On the other hand, by [Lemma 2.11](#), with probability except $1/n$ over $x \sim \mathcal{U}_{m(n)}$,

$$\text{pK}_{1/3}^{\rho(t(n)+5),f}(x) \geq m(n) - O(\log n) > s(n) + \log \rho(t(n) + 5),$$

where the final inequality holds provided that α is chosen to be a sufficiently large polynomial. That is, with probability except $1/n$ over $x \sim \mathcal{U}_{m(n)}$, $(x, 1^{s(n)}, 1^{t(n)})$ is a NO instance of $\text{Gap}_\rho\text{-MINpKT}^f$. It follows from the correctness of A that

$$\Pr_{x \sim \mathcal{U}_{m(n),A}} \left[A(x, 1^{s(n)}, 1^{t(n)}) = 0 \right] \geq 1 - \frac{1}{n} - \frac{1}{\alpha(n)^2}. \quad (26)$$

Comparing [Equations \(26\) and \(25\)](#), we obtain a probabilistic polynomial-time algorithm that distinguishes $G_n(\mathcal{U}_n)$ from $\mathcal{U}_{m(n)}$ for every sufficiently large n , a contradiction. $\diamond$

It follows from [Claim 5.6](#) and [Theorem 5.5](#) that $\text{DistPH} \subseteq \text{HeurBPP}$. Thus, it suffices to establish the following claim.

Claim 5.7. *If $\text{DistPH} \subseteq \text{HeurBPP}$, then “ $\text{MINpKT}^{\text{PH}} \in \text{HeurBPP}$ ”.*

Proof of Claim 5.7. The proof is essentially identical to that of [Claim 5.3](#). $\diamond$

The theorem now follows immediately from [Claim 5.7](#) and the preceding discussion. $\square$

6 A Relativization Barrier

In this section, we exhibit an oracle world where $\text{Gap-MINpKT}^{\text{PH}}$ is easy while $\text{Mild-Gap-MINpKT}^{\text{PH}}$ is hard. To make the statement that we aim to prove precise, we first explain the relativization of these computational problems. As usual, this is obtained by providing oracle access to all machines involved in the computation.

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

$$\text{pK}_\lambda^{t,L,\mathcal{O}}(x) \triangleq \min \left\{ s \in \mathbb{N} \mid \Pr_{r \sim \{0,1\}^t} \left[\text{K}^{t,L,\mathcal{O}}(x \mid r) \leq s \right] \geq \lambda \right\},$$

where $\text{K}^{t,L,\mathcal{O}}(x \mid r)$ denotes the conditional t -time bounded Kolmogorov complexity of x with oracle access to both L and $\mathcal{O}$.

For a function $\rho: \mathbb{N} \rightarrow \mathbb{N}$ and oracles L and $\mathcal{O}$, let $\text{Gap}_\rho\text{-MINpKT}^{L,\mathcal{O}}$ be the following promise problem: Given $(x, 1^s, 1^t, 1^a, 1^b)$, where $x \in \{0, 1\}^*$ and $s, t, a, b \in \mathbb{N}$, decide whether $\text{pK}_{b/a}^{t,L,\mathcal{O}}(x) \leq s$ (“yes” case) or $\text{pK}_{(b-1)/a}^{\rho(t+a+b),L,\mathcal{O}}(x) \geq s + \log \rho(t + a + b)$ (“no” case).

Let $\mathcal{O}$ be an oracle. We say that $\text{Gap-MINpKT}^{\text{PH}^\mathcal{O}, \mathcal{O}} \in \text{prBPP}^\mathcal{O}$ if for every language $L \in \text{PH}^\mathcal{O}$, there is a polynomial ρ and an oracle probabilistic polynomial-time algorithm that solves $\text{Gap}_\rho\text{-MINpKT}^{L, \mathcal{O}}$ when given oracle access to $\mathcal{O}$.

Analogously, we consider the collection of problems $\text{Mild-Gap-MINpKT}^{\text{PH}^\mathcal{O}, \mathcal{O}}$, for each $L \in \text{PH}^\mathcal{O}$. Each problem $\text{Mild-Gap}_\rho\text{-MINpKT}^{L, \mathcal{O}}$ is similar to $\text{Gap}_\rho\text{-MINpKT}^{L, \mathcal{O}}$ except for that in the no case, we have $\text{pK}_{(b-1)/a}^{t+\rho(|x|+a+b), L, \mathcal{O}}(x) \geq s + \log \rho(t+a+b)$ instead of $\text{pK}_{(b-1)/a}^{\rho(t+a+b), L, \mathcal{O}}(x) \geq s + \log \rho(t+a+b)$.

The statement $\text{Mild-Gap-MINpKT}^{\text{PH}^\mathcal{O}, \mathcal{O}} \notin \text{prBPP}^\mathcal{O}$ means that there is a language $L \in \text{PH}^\mathcal{O}$ such that for no constant $c \geq 1$ and polynomial $\rho(n) = c \cdot n^c$ we have $\text{Mild-Gap}_\rho\text{-MINpKT}^{L, \mathcal{O}} \in \text{prBPP}^\mathcal{O}$.

Theorem 6.1 (Oracle world where $\text{Gap-MINpKT}^{\text{PH}}$ is easy while $\text{Mild-Gap-MINpKT}^{\text{PH}}$ is hard).

There is an oracle $\mathcal{O}$ such that $\text{Gap-MINpKT}^{\text{PH}^\mathcal{O}, \mathcal{O}} \in \text{prBPP}^\mathcal{O}$ but $\text{Mild-Gap-MINpKT}^{\text{PH}^\mathcal{O}, \mathcal{O}} \notin \text{prBPP}^\mathcal{O}$.

Recall that, for an oracle $\mathcal{O}$, $\text{DistPH}^\mathcal{O}$ denotes the set of all distributional problems $(L, \mathcal{D})$ where $L \in \text{PH}^\mathcal{O}$ and $\mathcal{D} \in \text{PSAMP}^\mathcal{O}$. The proof of [Theorem 6.1](#) relies on the following oracle separation, which is an immediate consequence of [\[Imp11, HN21\]](#).

Theorem 6.2 ([\[Imp11, HN21\]](#)). *There is an oracle $\mathcal{O} \subseteq \{0, 1\}^*$ such that $\text{DistPH}^\mathcal{O} \subseteq \text{AvgBPP}^\mathcal{O}$ and $\text{PH}^\mathcal{O} \not\subseteq \text{BPP}^\mathcal{O}$.*

We are now ready to prove [Theorem 6.1](#).

Proof Sketch of Theorem 6.1. Since the argument is straightforward given [Theorem 6.2](#) and the proof of [Theorem 1.1](#), we only provide a sketch here. We consider the oracle $\mathcal{O} \subseteq \{0, 1\}^*$ provided by [Theorem 6.2](#), and verify that it satisfies the required properties in [Theorem 6.1](#).

1) Proof that $\text{Gap-MINpKT}^{\text{PH}^\mathcal{O}, \mathcal{O}} \in \text{prBPP}^\mathcal{O}$. By assumption, $\text{DistPH}^\mathcal{O} \subseteq \text{AvgBPP}^\mathcal{O}$. Let $L \in \text{PH}^\mathcal{O}$. We need to prove that there is a polynomial ρ and an oracle probabilistic polynomial-time algorithm that solves $\text{Gap}_\rho\text{-MINpKT}^{L, \mathcal{O}}$ when given oracle access to $\mathcal{O}$. We rely on the following lemma.¹⁴

Lemma 6.3 (Relativized form of [Lemma 3.12](#)). *For all oracles $L_1, L_2 \subseteq \{0, 1\}^*$, if $(\text{MINKT}^{L_1, L_2}, \text{PSAMP}) \in \text{AvgBPP}^{L_2}$, then $\text{Gap}_\rho\text{-MINpKT}^{L_1, L_2} \in \text{prBPP}^{L_2}$ for some polynomial ρ .*

Assuming [Lemma 6.3](#), we proceed as follows. We apply the lemma with $L_1 = L$ and $L_2 = \mathcal{O}$, which simplifies the assumption to $(\text{MINKT}^{L, \mathcal{O}}, \text{PSAMP}^\mathcal{O}) \subseteq \text{AvgBPP}^\mathcal{O}$. Since it is not hard to see that $\text{MINKT}^{L, \mathcal{O}} \in \text{PH}^\mathcal{O}$, the assumption of [Lemma 6.3](#) holds (using that $\text{DistPH}^\mathcal{O} \subseteq \text{AvgBPP}^\mathcal{O}$). Therefore, it follows that $\text{Gap}_\rho\text{-MINpKT}^{L, \mathcal{O}} \in \text{prBPP}^\mathcal{O}$ for some polynomial ρ , as desired.

It remains to prove [Lemma 6.3](#). The proof is a simple adaptation of that of [\[HLO26, Proposition 58\]](#) to the relativized setting. Since it requires no new idea, we omit the details.

2) Proof that $\text{Mild-Gap-MINpKT}^{\text{PH}^\mathcal{O}, \mathcal{O}} \notin \text{prBPP}^\mathcal{O}$. Recall that, by construction, $\text{PH}^\mathcal{O} \not\subseteq \text{BPP}^\mathcal{O}$. Next, we use this separation to argue that $\text{Mild-Gap-MINpKT}^{\text{PH}^\mathcal{O}, \mathcal{O}} \notin \text{prBPP}^\mathcal{O}$. In more detail, we establish the contrapositive, i.e., if $\text{Mild-Gap-MINpKT}^{\text{PH}^\mathcal{O}, \mathcal{O}} \in \text{prBPP}^\mathcal{O}$ then $\text{PH}^\mathcal{O} \subseteq \text{BPP}^\mathcal{O}$. Since $\text{NP}^\mathcal{O} \subseteq \text{BPP}^\mathcal{O}$ yields $\text{PH}^\mathcal{O} \subseteq \text{BPP}^\mathcal{O}$, it suffices to argue the assumption implies that $\text{NP}^\mathcal{O} \subseteq \text{BPP}^\mathcal{O}$.

To achieve this, it is sufficient to establish the following lemma.

Lemma 6.4 (Relativized form of [Lemma 3.4](#)). *For every $\mathcal{O} \subseteq \{0, 1\}^*$, if $\text{Mild-Gap-MINpKT}^{\text{PH}^\mathcal{O}, \mathcal{O}} \in \text{prBPP}^\mathcal{O}$, then $\text{NP}^\mathcal{O} \subseteq \text{BPP}^\mathcal{O}$.*

¹⁴ Analogously to the definitions introduced above, the measure $\text{K}^{t, L_1, L_2}(x)$ and the corresponding meta-computational problem MINKT^{L_1, L_2} are defined in the natural way, i.e., by considering programs that have oracle access to both L_1 and L_2 .

It is straightforward to check that the proof of [Lemma 3.4](#) extends to [Lemma 6.4](#). The key point is that all techniques employed in the proof relativize. We omit the details. $\square$

References

- [BT03] Andrej Bogdanov and Luca Trevisan. On worst-case to average-case reductions for NP problems. In *Symposium on Foundations of Computer Science*, pages 308–317, 2003.
- [BT06] Andrej Bogdanov and Luca Trevisan. Average case complexity. *Foundations and Trends in Theoretical Computer Science*, 2(1), 2006.
- [CHV22] Lijie Chen, Shuichi Hirahara, and Neekon Vafa. Average-case hardness of NP and PH from worst-case fine-grained assumptions. In *Innovations in Theoretical Computer Science Conference*, pages 45:1–45:16, 2022.
- [FF93] Joan Feigenbaum and Lance Fortnow. Random-self-reducibility of complete sets. *SIAM Journal on Computing*, 22(5):994–1005, 1993.
- [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.*, TR22-007, 2022.
- [GKLO22] Halley Goldberg, Valentine Kabanets, Zhenjian Lu, and Igor C. Oliveira. Probabilistic Kolmogorov complexity with applications to average case complexity. In *Computational Complexity Conference*, pages 16:1–16:60, 2022.
- [GSTS07] Dan Gutfreund, Ronen Shaltiel, and Amnon Ta-Shma. If NP languages are hard on the worst-case, then it is easy to find their hard instances. *Computational Complexity*, 16(4):412–441, 2007.
- [HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.
- [Hir18] Shuichi Hirahara. Non black box worst case to average case reductions within NP. In *Symposium on Foundations of Computer Science*, 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*, pages 50–60, 2020.
- [Hir21] Shuichi Hirahara. Average case hardness of NP from exponential worst case hardness assumptions. In *Symposium on Theory of Computing*, pages 292–302, 2021.
- [Hir22a] Shuichi Hirahara. Meta-computational average-case complexity: A new paradigm toward excluding Heuristica. *Bull. EATCS*, 136, 2022.
- [Hir22b] Shuichi Hirahara. NP-hardness of learning programs and partial MCSP. In *Symposium on Foundations of Computer Science*, pages 968–979, 2022.

- [Hir22c] Shuichi Hirahara. Symmetry of information from meta-complexity. In *Computational Complexity Conference*, pages 26:1–26:41, 2022.
- [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*, pages 29:1–29:56, 2024.
- [HLO26] Jinqiao Hu, Zhenjian Lu, and Igor C. Oliveira. Hardness of computing nondeterministic Kolmogorov complexity. In *Computational Complexity Conference*, 2026.
- [HLR23] Shuichi Hirahara, Zhenjian Lu, and Hanlin Ren. Bounded relativization. In *Computational Complexity Conference*, pages 6:1–6:45, 2023.
- [HN21] Shuichi Hirahara and Mikito Nanashima. On worst-case learning in relativized Heuristica. In *Symposium on Foundations of Computer Science*, pages 751–758, 2021.
- [HS22] Shuichi Hirahara and Rahul Santhanam. Excluding PH pessiland. In *Innovations in Theoretical Computer Science Conference*, pages 85:1–85:25, 2022.
- [IL90] Russell Impagliazzo and Leonid A. Levin. No better ways to generate hard NP instances than picking uniformly at random. In *Symposium on Foundations of Computer Science*, pages 812–821, 1990.
- [Ila23] Rahul Ilango. SAT reduces to the minimum circuit size problem with a random oracle. In *Symposium on Foundations of Computer Science*, pages 733–742, 2023.
- [Imp95] Russell Impagliazzo. A personal view of average case complexity. In *Proceedings of the Tenth Annual Structure in Complexity Theory Conference*, pages 134–147, 1995.
- [Imp11] Russell Impagliazzo. Relativized separations of worst-case and average-case complexities for NP. In *Conference on Computational Complexity*, pages 104–114, 2011.
- [IRS22] Rahul Ilango, Hanlin Ren, and Rahul Santhanam. Robustness of average-case meta-complexity via pseudorandomness. In *Symposium on Theory of Computing*, pages 1575–1583, 2022.
- [Lev86] Leonid A. Levin. Average case complete problems. *SIAM Journal on Computing*, 15(1):285–286, 1986.
- [LO22] Zhenjian Lu and Igor C. Oliveira. Theory and applications of probabilistic Kolmogorov complexity. *Bull. EATCS*, 137, 2022.
- [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*, pages 92:1–92:14, 2022.
- [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*, pages 645–673, 2023.
- [LP26] Yanyi Liu and Rafael Pass. One-way functions and boundary hardness of randomized time-bounded Kolmogorov complexity. In *Innovations in Theoretical Computer Science*, 2026.
- [LV19] Ming Li and Paul M. B. Vitanyi. *An Introduction to Kolmogorov Complexity and Its Applications*. Springer, 4 edition, 2019.

- [RS22] Hanlin Ren and Rahul Santhanam. A relativization perspective on meta-complexity. In *International Symposium on Theoretical Aspects of Computer Science*, pages 54:1–54:13, 2022.
- [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>.