One-Way Functions and the Hardness of (Probabilistic) Time-Bounded Kolmogorov Complexity w.r.t. Samplable Distributions

Yanyi Liu[1(✉)] and Rafael Pass[2]

[1] Cornell Tech, New York, USA
yl2866@cornell.edu
[2] Tel-Aviv University & Cornell Tech, Tel Aviv, Israel
rafaelp@tau.ac.il

Abstract. Consider the recently introduced notion of *probabilistic time-bounded Kolmogorov Complexity*, pK^t (Goldberg et al., CCC'22), and let $\mathsf{MpK}^t\mathsf{P}$ denote the language of pairs (x, k) such that $pK^t(x) \leq k$. We show the equivalence of the following:

- $\mathsf{MpK}^{\mathsf{poly}}\mathsf{P}$ is (mildly) hard-on-average w.r.t. *any* samplable distribution $\mathcal{D}$;
- $\mathsf{MpK}^{\mathsf{poly}}\mathsf{P}$ is (mildly) hard-on-average w.r.t. the *uniform* distribution;
- existence of one-way functions.

As far as we know, this yields the first natural class of problems where hardness with respect to any samplable distribution is equivalent to hardness with respect to the uniform distribution.

Under standard derandomization assumptions, we can show the same result also w.r.t. the standard notion of time-bounded Kolmogorov complexity, K^t.

1 Introduction

A *one-way function* [5] (OWF) is a function f that can be efficiently computed (in polynomial time), yet no probabilistic polynomial-time (PPT) algorithm can invert f with inverse polynomial probability for infinitely many input lengths n. Whether one-way functions exist is unequivocally the most important open problem in Cryptography (and arguably the most importantly open problem in

Y. Liu—Work done while visiting the Simons Institute during the Meta-complexity program. Supported by a JP Morgan fellowship.
R. Pass—Supported in part by NSF Award CNS 2149305, NSF Award CNS-2128519, NSF Award RI-1703846, AFOSR Award FA9550-18-1-0267, FA9550-23-1-0312, a JP Morgan Faculty Award, the Algorand Centres of Excellence programme managed by Algorand Foundation, and DARPA under Agreement No. HR00110C0086. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Government, DARPA or the Algorand Foundation.

© International Association for Cryptologic Research 2023
H. Handschuh and A. Lysyanskaya (Eds.): CRYPTO 2023, LNCS 14082, pp. 645–673, 2023.
https://doi.org/10.1007/978-3-031-38545-2_21

the theory of computation, see e.g., [19]): OWFs are both necessary [16] and sufficient for many of the most central cryptographic primitives and protocols (e.g., pseudorandom generators [2,12], pseudorandom functions [8], private-key encryption [9], digital signatures [28], commitment schemes [26], identification protocols [6], coin-flipping protocols [1], and more). These primitives and protocols are often referred to as *private-key primitives*, or "Minicrypt" primitives [14] as they exclude the notable task of public-key encryption [5,27]. Additionally, as observed by Impagliazzo [10,14], the existence of a OWF is equivalent to the existence of polynomial-time method for sampling hard *solved* instances for an NP language (i.e., hard instances together with their witnesses).

The Win-Win Paradigm, and OWFs from Average-case Hardness of NP? A central problem in the theory of Cryptography is whether the existence of OWFs can be based on some simple complexity-theoretic assumptions. Ideally, we would want an assumption that leads to a *win-win* scenario: either we have secure OWFs (and thus can securely implement all primitives in Minicrypt), or we get some algorithmic breakthroughs that are useful to society/the pursuit of knowledge etc. The ideal win-win scenario would be to get a construction of OWF based on worst-case hardness of NP (i.e., on the assumption that $\mathsf{NP} \not\subseteq \mathsf{BPP}$)—the question of whether this is possible goes back to the original work by Diffie and Hellman [5] and is sometimes referred to as the "holy-grail" of Cryptography. A slightly less ambitious goal that still would yield a very strong win-win scenario would be to base the existence of OWF on the existence of an NP language that is *average-case hard* w.r.t. to some samplable distribution $\mathcal{D}$. (Note that the existence of OWF trivially implies this assumption.) If such a reduction were to be obtained (or in Impagliazzo's language, if we rule out "Pessiland"—a world where NP is hard on average but OWFs do not exist.), then either OWF exists, or we can solve all NP problems "in practice", whenever the instances are sampled by an "efficient world". Unfortunately, also obtaining such a reduction has remained an open problem for 5 decades:

> *Does the average-case hardness (w.r.t. some efficiently samplable distribution) of some language in* NP *imply the existence of OWFs?*

There has, however, been some recent progress towards this question based on connections between OWFs and Kolmogorov Complexity.

On OWFs and Kolmogorov Complexity. The notion of *Kolmogorov complexity* (K-complexity), introduced by Solomonoff [31], Kolmogorov [18] and Chaitin [4], provides an elegant method for measuring the amount of "randomness" in individual strings: The K-complexity of a string is the length of the shortest program (to be run on some fixed universal Turing machine U) that outputs the string x; the notion of *$t(\cdot)$-time-bounded Kolmogorov Complexity (K^t-complexity)* [11,17,18,30,32] considers a time-bounded version of this problem: $K^t(x)$ is defined as the length of the shortest program that outputs the string x within time $t(|x|)$.

A recent result by Liu and Pass [21] shows that "mild" average-case hardness[1] of the time-bounded Kolmogorov complexity problem (when the time-bound is some polynomial) is *equivalent* to the existence of OWFs. Additionally, [23] demonstrates that the same type of result also holds for the, so-called, *conditional time-bounded Kolmogorov Complexity problem* [20,24,32,34] (where $K^t(x|z)$ is defined as the length of the shortest program that within time $t(|x|)$ outputs x having access to z) that they also show is NP-complete. The problem, however, is that it is not known whether the problem is *average-case* complete with respect to the *uniform distribution*. In other words, if NP is average-case hard (with respect to some samplable distribution), then the (conditional) time-bounded Kolmogorov complexity problem is hard for *some* efficiently samplable distribution (by its NP-completeness), but the characterization of OWFs considers hardness of the problem with respect to the *uniform* distribution.

Hardness w.r.t. Samplable or the Uniform Distribution. Thus, resolving the above central open problem (of basing OWF on average-case hardness of NP) is *equivalent* to showing that average-case hardness of the time-bounded Kolmogorov complexity problem with respect to *any* samplable distribution implies average-case hardness with respect to the uniform distribution. More generally, we may ask:

> *For what classes of problems does average-case hardness with respect to* **any samplable distributions** *imply average-case hardness with respect to the* **uniform distribution***?*

Our focus here will be on time-bounded Kolmogorov complexity-style problem due to their connection with cryptography. As mentioned, showing this for the particular conditional time-bounded Kolmogorov complexity problem is equivalent to basing OWF on the average-case hardness of NP (i.e., ruling out Pessiland). But showing this for just the "plain" time-bounded Kolmogorov complexity problem would also yield a very natural win-win scenario: while there are many important applications to solving the time-bounded Kolmorogov complexity (e.g., optimal file-compression, inductive reasoning in science, optimal machine learning etc.[2]), we typically do not care much about solving it on random instances, but rather instances with structure. If one can base OWF on the hardness of this problem with respect to any samplable distribution, we would get non-existence of OWF implies that the K^t-complexity can be "solved in practice".

An elegant step in this direction was recently taken by Ilango, Ren and Santhanam [13]; they show that the existence of OWFs is equivalent to average-case hardness of a *Gap* version—with a $\omega(\log n)$ gap—of the Kolmogorov complexity

[1] By "mild" average-case hardness, we here mean that no PPT algorithm is able to solve the problem with probability $1-\frac{1}{p(n)}$ on inputs of length n, for some polynomial $p(\cdot)$.

[2] Typically, one would actually like to solve a *search version* of this problem, where one not only finds the time-bounded K-complexity of a string but also the program that "witnesses" this complexity; as we shall see our results actually consider this.

problem w.r.t. any efficiently samplable distribution; Liu and Pass [22] extend this result to show that it suffices to assume that it is hard to *approximate* K-complexity within a term of $\omega(\log n)$ with respect to any samplable distribution. [13] also show that under standard derandomization assumption, it suffices to assume average-case hardness also of a Gap version (again with $\omega(\log n)$ gap) also of the time-bounded Kolmogorov complexity problem, and thus for a problem in NP, w.r.t. any samplable distribution. These results thus show that one can characterize OWFs through average-case hardness of some natural gap/approximation problem with respect to any samplable distribution. The problem, however, is that they all work in a gap/approximation regime where the problem is provably easy under the uniform distribution—It is trivial to provide a $\omega(\log n)$ approximation of K or K^t w.r.t. the uniform distribution: simply output the length of the string! Indeed, as these result show, in this regime, the sampler for the hard distribution must it self be a OWF. So in a sense, these result do not give us any insight into how to build a OWF "from scratch".

As far as we know, the only result that we are away of showing that hardness with respect to any samplable distributions implies hardness with respect to the uniform distribution is the seminal result of Impagliazzo and Levin [15]; their result however only shows average-case hardness of NP with respect to some samplable distribution implies average-case hardness of some (artificial) specially-constructed NP language with respect to the uniform distribution. As far as we known, no such reductions are not known for any "natural" classes of languages.

1.1 Our Results

Roughly speaking, our main result shows that for a *probabilistic version* of K^t, average-case hardness with respect to any samplable distribution (of the exact, as opposed to the approximate) problem is equivalent to the average-case hardness with respect to the uniform distribution, which in turn is equivalent to the existence of OWFs. This notion, called *probabilistic* K^t (denoted pK^t) was recently introduced by Goldberg et al [7]. Roughly speaking, this notion measures the length of the shortest program that outputs a string x if we get access to a random string (think of it as K^t in the "Common Random String" (CRS) model). More formally, let

$$pK^t_\delta(x) = \min\{w \in \mathbb{N} \mid \Pr[r \leftarrow \{0,1\}^{t(|x|)} : K^t(x \mid r) \leq w] \geq \delta\}$$

and let $\mathsf{MpK}^t\mathsf{P}$ denote the promise problem $(\Pi_{\mathsf{YES}}, \Pi_{\mathsf{NO}})$ where Π_{YES} consists of (x,k), $|k| = \lceil \log |x| \rceil$, $pK^t_{2/3}(x) \leq k$ and Π_{NO} consists of (x,k), $|k| = \lceil \log |x| \rceil$, $pK^t_{1/3}(x) > k$.

Our main result shows:

Theorem 11 *The following are equivalent:*

- *There exists an efficiently samplable distribution $\mathcal{D}$ such $\mathsf{MpK}^t\mathsf{P}$ is mildly hard-on-average on $\mathcal{D}$ for some (or every) sufficiently large polynomial $t(\cdot)$;*

- $\mathsf{MpK}^t\mathsf{P}$ *is mildly hard-on-average on the uniform distribution for some (or every) sufficiently large polynomial* $t(\cdot)$
- *OWF exists.*

In fact, our formal proof is even stronger; we show that it suffices to assume hardness of a search version of the pK^t problem where given any x sampled from an efficient distribution $\mathcal{D}$, and a random CRS r, the goal is to find the shortest program that generates x given r. This yields a strong and natural win-win scenario:

> *Either OWF exist, or we can (with probability* $1 - 1/\mathsf{poly}(n)$*) find the best way to compress any efficiently sampled string* x*, in the presence of a CRS.*

We highlight that such compression is not just useful in its own; if ascribe to Occam's razor (i.e., "rule of parsimony": that the simplest way to explain a phenomena is preferable to a more complex), then solving this search version of K^t (even in the presence of a CRS), yields a powerful tool for scientific discovery.

We next turn to considering the "standard" K^t problem; let $\mathsf{MK}^t\mathsf{P}$ denote the language of pairs of (x, k), $|k| = \lceil \log |x| \rceil$, $K^t(x) \leq k$. We show that under standard derandomization assumptions (used to show that $\mathsf{AM} \subset \mathsf{NP}$), hardness of $\mathsf{MK}^{\mathsf{poly}}\mathsf{P}$ w.r.t. some samplable distribution is equivalent to hardness w.r.t. the uniform distribution (which by [21] is equivalent to OWF).

Theorem 12. *Assume that* $\mathsf{E} \not\subseteq \mathsf{ioNSIZE}[2^{\Omega(n)}]$*. Then, the following are equivalent:*

- *There exists an efficiently samplable distribution* $\mathcal{D}$ *such that* $\mathsf{MK}^t\mathsf{P}$ *is mildly hard-on-average on* $\mathcal{D}$ *for some (or every) sufficiently large polynomial* $t(\cdot)$*;*
- $\mathsf{MK}^t\mathsf{P}$ *is mildly hard-on-average on the uniform distribution for some (or every) sufficiently large polynomial* $t(\cdot)$*;*
- *OWF exists.*

Again, we can strengthen the result and base it on the hardness of solving the search version of the K^t problem.

This final results is related to the recent result by [13], that shows equivalence of *infinitely-often* OWFs and the io-average-case hardness of a Gap K^t problem (with a $\omega(\log n)$ gap) under a derandomization assumption. First, their result does not extend to handle also "standard" OWFs (on the other hand, it uses a weaker derandomization assumptions).[3] More significantly, our result weakens the assumption to only require hardness of solving the *exact* (as opposed to Gap/approximate) version of the K^t problem. This difference is significant, and the results are different on a qualitative level: K^t seemingly is hard to compute on essentially any "well-spread" distribution (and in particular on the uniform distribution), but it seems very hard to (unconditionally) find a distribution on which it is hard to approximate within $\omega(\log n)$. Indeed, as mentioned above,

[3] It would seem that we can also use a weaker derandomization assumption in case we only want to deduce io-OWFs; we defer the details to the full version.

the proof in [13] essentially show that the Gap problem can only be hard on a samplable distribution $\mathcal{D}$ if the sampling procedure for the distribution itself is a OWF.

1.2 Proof Overview

We here provide some intuitions behind the proofs of Theorems 11 and 12. We will show that (1) hardness with respect to any samplable distribution implies OWF, and (2) OWFs imply hardness with respect to the uniform distribution. Step (2) will actually follow mostly using the techniques from [21]—they pass through the notion of an "entropy-preserving PRG" constructed in [21] from OWFs, and we next observe that just as [21] showed that such PRGs imply (mild) average-case hardness of $\mathsf{MK}^t\mathsf{P}$, we can also show (dealing just with some minor technical details) that they also imply mild average-case hardness of $\mathsf{MpK}^t\mathsf{P}$.

We here focus on (1); for simplicity of notation, let us start by considering the standard K^t problem. We aim to construct a OWF assuming K^t is mildly hard-on-average to compute with respect to some samplable distribution. Towards doing this, let us first recall the high-level idea behind the construction of [21], that was based on the average-case hardness of computing K^t with respect to the *uniform* distribution.

The LP20 OWF. [21] actually only constructs a so-called *weak* OWF[4]; a (strong) OWF can be be obtained by relying on Yao's hardness amplification theorem [33]. Their construction proceeds as follows. Let c be a constant such that every string x can be output by a program of length $|x| + c$ (running on the fixed Universal Turing machine U). Consider the function $f(\ell||\Pi')$, where ℓ is a bitstring of length $\log(n + c)$ and Π' is a bitstring of length $n + c$, that lets Π be the first ℓ bits of Π', and outputs $\ell||y$ where y is the output generated by running the program Π[5] for $t(n)$ steps.

We aim to show that if f can be inverted with high probability—significantly higher than $1 - 1/n$—then K^t-complexity of *random strings* $z \in \{0,1\}^n$ can be computed with high probability. The heuristic $\mathcal{H}$, given a string z, simply tries to invert f on $\ell||z$ for all $\ell \in [n + c]$, and outputs the smallest ℓ for which inversion succeeds.[6]

The key idea for arguing that this works is that for every string z with K^t-complexity w, there exists some program Π_z of length w that outputs it; furthermore, by our assumption on c, $w \leq n + c$. We thus have that $f(\mathcal{U}_{n+c+\log(n+c)})$ will output $w||z$ with probability at least

[4] Recall that an efficiently computable function f is a weak OWF if there exists some polynomial $q > 0$ such that f cannot be efficiently inverted with probability better than $1 - \frac{1}{q(n)}$ for sufficiently large n.

[5] Formally, the program/description Π is an encoding of a pair (M, w) where M is a Turing machine and w is some input, and we evaluate $M(w)$ on the Universal Turing machine U.

[6] Or, in case, we also want solve the search problem, we also output the ℓ-bit truncation of the program Π' output by the inverter.

$$\frac{1}{n+c} \cdot 2^{-w} \geq \frac{1}{n+c} \cdot 2^{-(n+c)} = \frac{2^{-n}}{O(n)}$$

(we need to pick the right length, and next the right program). So, if the heuristic fails with probability δ, then the one-way function inverter must fail with probability at least $\frac{\delta}{O(n)}$, which leads to the conclusion that δ must be small (as we assumed the inverter fails with probability significantly smaller than $\frac{1}{n}$).

Dealing with Samplable Distributions: Step 1. Our main insight is that the above proof idea actually works to solve K^t not only on the uniform distribution but in fact also on any distribution D that samples any string x with probability upperbounded by $\frac{\mathsf{poly}(n)}{2^{K^t(x)}}$—we refer to such a distribution as being *polynomially bounded by* K^t. To see why this holds, consider again a string z with K^t complexity w. As before, $f(\mathcal{U}_{n+c+\log(n+c)})$ will output $w||z$ with probability at least

$$\frac{1}{n+c} \cdot 2^{-w}$$

Given that this string z is sampled by D with probability $\leq \mathsf{poly}(n)2^{-w}$, we again have that if the heuristic fails for a set of string z with probability mass δ, then the OWF inverter must fail with probability $\delta/\mathsf{poly}(n)$ (since pointwise, the probabilities in the OWFs experiment "dominate" the probabilities assigned by D, except for a polynomial factor.) This concludes that the LP20 OWF actually is secure even if we simply assume that K^t is hard for any distribution that is polynomially bounded by K^t. (Note that it directly follows that the uniform distribution is polynomially bounded by K^t, by the observation that $K^t(x) \leq |x| + c$; so this condition already trivially generalizes the condition from [21].)

Dealing with Samplable Distributions: Step 2. In the second step of the proof we aim to show that if we take an efficiently samplable distribution, then it must be polynomially bounded by K^t. (As we shall discuss shortly, we will not quite be able to do this, but either turning to pK^t, or using derandomization, will help. But let's postpone this for a moment.)

The intuition for why this ought to be true is the following. Consider some efficient sampler D that is able to sample an element x s.t. $K^t(x) = w$ with probability $n^{\omega(1)} \cdot 2^{-w}$. We can have at most $2^w/n^{\omega(1)}$ such elements so intuitively, we can compress x into $\log(2^{w-\omega(1)\log n}) = w - \omega(1)\log n$ bits, which seems like a contradiction. The problem, however is that we cannot efficiently recover x from the list of these strings.

However, the very recently-established Coding Theorem of Lu, Oliviera and Zimand [25] essentially shows how to do this exactly with the caveat that we need to consider pK^t as opposed to just K^t. Their proof uses quite heavy machinery. As an independent contribution, we here provide an elementary proof.

In particular, to efficiently recover x, what if we had access to a universal hash function H, provided as a CRS? As we shall argue, we can indeed find a short ($< w$) representation of x that can be efficiently decoded. Let ℓ denote the number of random bits used by the sampler, and let S denote the set of random tapes that

map x; note that $|S| \geq 2^{\ell} \cdot n^{\omega(1)} \cdot 2^{-w} = 2^{\ell - w - \omega(1)\log n}$. If we apply H to each of these random tapes, truncate the answer to $\log |S| - O(1)$ bits, then it follows from the Chebyshev's inequality that with some large constant probability, there will exist some random tape leading to x that gets mapped to the all 0 string. Furthermore, by the same Chebyshev's inequality-based argument, there are at most $2^{\ell - \log |S|} = 2^{w - \omega(1)\log n}$ strings in total that get mapped to the all 0 string. We can finally rely on the fact that universal hash functions can be constructed using a linear mapping, and we can leverage this structure to efficiently index each of these pre-images to the all 0 string of the hash function. Essentially, we can simply use a basis for the kernel of the matrix describing the hash function. Since the space contains $2^{w - \omega(1)\log n}$ strings, we will have $w - \omega(1)\log n$ basis vectors, and each such string in the space can thus be specified by a binary vector of length $w - \omega(1)\log n$ bits.

This still does not contradict the assumption that $K^t(x) = w$ since the above compression uses an external hash function. However, if we instead switch to using pK^t, then we do get a contradiction. This, of course, requires redoing also Step 1 w.r.t to pK^t, which introduces some additional technicalities but we can essentially proceed in the same way.

Finally, we remark that if we rely on derandomization assumptions, we can actually derandomize the hashfunction and actually prove Step 2 also for K^t. (In fact, proving step 2 for K^t can be obtained as a direct corollary step 2 w.r.t. pK^t and a result from [7] that relates pK^t and K^t under derandomization assumption; for completeness, also provide a a simple direct proof based on derandomizing the hashfunction.)

2 Preliminaries

2.1 One-Way Functions

We recall the definition of one-way functions [5]. Roughly speaking, a function f is one-way if it is polynomial-time computable, but hard to invert for PPT attackers.

Definition 21. *Let $f : \{0,1\}^* \to \{0,1\}^*$ be a polynomial-time computable function. f is said to be a* one-way function (OWF) *if for every* PPT *algorithm $\mathcal{A}$, there exists a negligible function μ such that for all $n \in \mathbb{N}$,*

$$\Pr[x \leftarrow \{0,1\}^n; y = f(x) : A(1^n, y) \in f^{-1}(f(x))] \leq \mu(n)$$

We may also consider a weaker notion of a *weak one-way function* [33], where we only require all PPT attackers to fail with probability noticeably bounded away from 1:

Definition 22. *Let $f : \{0,1\}^* \to \{0,1\}^*$ be a polynomial-time computable function. f is said to be an α-*weak one-way function (α-weak OWF) *if for every* PPT *algorithm $\mathcal{A}$, for all sufficiently large $n \in N$,*

$$\Pr[x \leftarrow \{0,1\}^n; y = f(x) : A(1^n, y) \in f^{-1}(f(x))] < 1 - \alpha(n)$$

We say that f is simply a weak one-way function (weak OWF) *if there exists some polynomial $q > 0$ such that f is a $\frac{1}{q(\cdot)}$-weak OWF.*

Yao's hardness amplification theorem [33] shows that any weak OWF can be turned into a (strong) OWF.

Theorem 23 ([33]). *Assume there exists a weak one-way function. Then there exists a one-way function.*

2.2 Time-Bounded Kolmogorov Complexity

We introduce the notion of time-bounded conditional Kolmogorov complexity. Roughly speaking, the *t-time-bounded Kolmogorov complexity*, $K^t(x \mid z)$, of a string $x \in \{0,1\}^*$ conditioned on a string $z \in \{0,1\}^*$ is the length of the shortest program $\Pi = (M, y)$ such that $\Pi(z)$ outputs x in $t(|x|)$ steps.

Formally, fix some universal RAM machine U (with only polynomial overhead), and let $t(\cdot)$ be a running time bound. For any string $x, z \in \{0,1\}^*$, we define

$$K^t(x \mid z) = \min\{w \in \mathbb{N} \mid \exists \Pi \in \{0,1\}^w, U(\Pi(z), 1^{t(|x|)}) = x\}$$

When z is an empty string, we simply denote the quantity by $K^t(x)$. We consider RAM machines (as in [7,23]) since it allows z to be as long as (or even longer than) the running time of the machine Π.

Very recently, Goldberg et al [7] introduced a probabilistic variant of time-bounded Kolmogorov complexity, denoted as pK^t. Let us recall the notion here. Roughly speaking, in the probabilistic version, the program is allowed to be picked after a uniform random string. And a string will have small pK^t-complexity if a short program exists over a large fraction of random strings. We proceed to the formal definition. Let $\delta(n)$ be a probability threshold function. For any string $x \in \{0,1\}^*$, the δ-probabilistic t-bounded Kolmogorov complexity of x [7], $pK^t_\delta(x)$, is defined to be

$$pK^t_\delta(x) = \min\{w \in \mathbb{N} \mid \Pr[r \leftarrow \{0,1\}^{t(|x|)} : K^t(x \mid r) \leq w] \geq \delta(n)\}$$

We usually consider δ as being a constant. We omit the subscript δ if $\delta = 2/3$.

We rely on the following decisional/search problems about time-bound Kolmogorov complexity (and its probabilistic variant).

Decisional. We turn to defining the decisional version of the minimum time-bounded Kolmogorov complexity problem. Let $\mathsf{MK^tP}$ denote the language of pairs of (x,k), $|k| = \lceil \log |x| \rceil$, $K^t(x) \leq k$. For its probabilistic version, let $\mathsf{MpK^tP}$ denote the promise problem $(\Pi_{\mathsf{YES}}, \Pi_{\mathsf{NO}})$ where Π_{YES} consists of (x,k), $|k| = \lceil \log |x| \rceil$, $pK^t_{2/3}(x) \leq k$ and Π_{NO} consists of (x,k), $|k| = \lceil \log |x| \rceil$, $pK^t_{1/3}(x) > k$.

Search. We will rely on the search version of the minimum time-bounded Kolmogorov complexity problem. In our search problem, an instance is a single string $x \in \{0,1\}^*$ (as opposed to a pair of string x and threshold k, as in the

decisional version). A witness of a string x is the shortest program that outputs x within $t(|x|)$ steps. We turn to the formal definition. Let Search-K^t denote the binary relation $R_{\mathsf{search}\text{-}K^t} \subseteq \{0,1\}^n \times \{0,1\}^*$ where $(x, \Pi) \in R_{\mathsf{search}\text{-}K^t}$ iff $|\Pi| = K^t(x)$, and $U(\Pi, 1^{t(|x|)}) = x$.

We will also define the search version of the minimum conditional time-bounded Kolmogorov complexity problem. In this problem, an instance is a pair of a target string x and an auxiliary input z. And its witness is just the "K^t-witness" of x conditioned on z. More formally, let Search-cK^t denote the binary relation $R_{\mathsf{search}\text{-}cK^t} \subseteq \{0,1\}^{n+t(n)} \times \{0,1\}^*$ where $((x,z), \Pi) \in R_{\mathsf{search}\text{-}cK^t}$ iff $z \in \{0,1\}^{t(|x|)}, |\Pi| = K^t(x \,|\, z)$, and $U(\Pi(z), 1^{t(|x|)}) = x$.

Finally, we recall two useful properties with respect to K^t: (1) The K^t-complexity of x is always bounded by its length (plus a universal constant); and (2) random strings will have high K^t-complexity with high probability. We notice that these two properties are also satisfied if we focus on its probabilistic variant pK^t.

Fact 24 ([7]). *The following statements hold.*

1. *There exists a constant c such that for all polynomials $t(n) \geq n$, all functions $\delta(n) \leq 1$, for all strings $x \in \{0,1\}^*$ it holds that $pK^t_\delta(x) \leq K^t(x) \leq |x| + c$.*
2. *For any $n \in \mathbb{N}, m < n$, $\Pr[x \leftarrow \{0,1\}^n : pK^t_\delta(x) \leq m] \geq 1 - \frac{1}{\delta(n) 2^{n-m+1}}$*

2.3 Average-Case Complexity

We turn to defining what it means for complexity problems to be average-case hard (for PPT algorithms). We will be considering problems that are only defined on some input lengths (such as $\mathsf{MK^tP}$). We say that a language L is *defined over inputs lengths $s(\cdot)$* if $L \subseteq \cup_{n\in\mathbb{N}}\{0,1\}^{s(n)}$. (For promise problems or search problems, this can also be done analogously.) For concreteness, note that $\mathsf{MK^tP}$ is defined on input lengths $s(n) = n + \lceil \log n \rceil$.

We will also consider ensembles that are only defined on some input lengths. We say that $\mathcal{D} = \{D_n\}_{n\in\mathbb{N}}$ is an *ensemble* defined over input lengths $s(\cdot)$ if for all $n \in \mathbb{N}$, D_n is a probability distribution over $\{0,1\}^{s(n)}$. (In this work, we will only consider ensembles that are defined only over $s(n) = n + \lceil \log n \rceil$.) We say that an ensemble $\mathcal{D} = \{D_n\}_{n\in\mathbb{N}}$ is *samplable* if there exists a probabilistic polynomial-time Turing machine S such that $S(1^n)$ samples D_n; we use the notation $S(1^n; r)$ to denote the algorithm S with randomness fixed to r. We say that $\mathcal{D}$ is $t_D(\cdot)$-time samplable if for all $n \in \mathbb{N}$, $S(1^n)$ terminates in $t_D(n)$ steps. One example of an ensemble defined over input lengths $s(\cdot)$ is the uniform distribution, which samples each $x \in \{0,1\}^{s(n)}$ with equal probability for each $n \in \mathbb{N}$.

Definition 25 (Average-case Complexity). *We say that a problem P defined over input lengths $s(\cdot)$ is* mildly hard-on-average (mildly HoA) with respect to an ensemble $\mathcal{D}$ *(also defined over input lengths $s(\cdot)$) if there exists a polynomial p such that for all PPT heuristic $\mathcal{H}$, for all sufficiently large $n \in \mathbb{N}$,*

$$\Pr[x \leftarrow D_n : \mathcal{H}(x) \textit{ fails to solve } P \textit{ on } x] \geq \frac{1}{p(n)}$$

where

- *if P is a language L, $\mathcal{H}(x)$ fails to solve L on x iff $\mathcal{H}(x) \neq L(x)$;*
- *if P is a promise problem $\Pi = (\Pi_{\mathsf{YES}}, \Pi_{\mathsf{NO}})$, $\mathcal{H}(x)$ fails to solve Π on x iff $x \in \Pi_{\mathsf{YES}} \wedge \mathcal{H}(x) = 0$ or $x \in \Pi_{\mathsf{NO}} \wedge \mathcal{H}(x) = 1$;*
- *if P is a search problem $R \subseteq \{0,1\}^* \times \{0,1\}^*$, $\mathcal{H}(x)$ fails to solve R on x iff $(x, \mathcal{H}(x)) \notin R$;*

We next show some search-to-decision reductions for the Kolmogorov complexity problems we considered. These reductions are easy to see in the worst-case setting, we now prove them in the average-case setting. Let $\mathcal{D}$ be a distribution ensemble for $\mathsf{MK}^t\mathsf{P}$ (or $\mathsf{MpK}^t\mathsf{P}$). Recall that D_n will sample a pair of a string $x \in \{0,1\}^n$ and a threshold $k \in \{0,1\}^{\lceil \log n \rceil}$. We consider the *projected distribution*, $\mathcal{D}'$, of $\mathcal{D}$, where each D'_n is just D_n but only samples x from D_n.

Lemma 26. *Let t be a polynomial.*

- *If there exists an ensemble $\mathcal{D}$ under which $\mathsf{MpK}^t\mathsf{P}$ is mildly HoA, then Search-cK^t is mildly HoA w.r.t. $(\mathcal{D}', \mathcal{U})$ where $\mathcal{D}'$ is the projected distribution of $\mathcal{D}$.*
- *If there exists an ensemble $\mathcal{D}$ under which $\mathsf{MK}^t\mathsf{P}$ is mildly HoA, then Search-K^t is mildly HoA w.r.t. the projected distribution of $\mathcal{D}$.*

Proof. We sketch the proof for the former statement, and the latter statement will follow from essentially the same proof with minor adjustments.

For any polynomial p, we will show that if there is an algorithm $\mathcal{A}$ that solves Search-cK^t with probability at least $1 - \frac{1}{2p(n)^2}$ over $(\mathcal{D}', \mathcal{U})$ for infinitely many n, then there exists an algorithm $\mathcal{H}$ that decides $\mathsf{MpK}^t\mathsf{P}$ with probability $\geq 1 - \frac{1}{p(n)}$ w.r.t. $\mathcal{D}$ for infinitely many n. Fix some n on which $\mathcal{A}$ succeeds over $x \leftarrow D'_n$, $r \leftarrow \{0,1\}^{t(n)}$.

Consider the algorithm $\mathcal{H}$ that acts as follows. On input $(x,k) \leftarrow D_n$, $\mathcal{H}$ repeats the following procedure for at least n times. In each iteration, $\mathcal{H}$ samples random $r \leftarrow \{0,1\}^{t(n)}$, and invokes $\mathcal{A}(x,r)$. Finally, $\mathcal{H}$ outputs 1 if in at least a half fraction of iterations, $\mathcal{A}$ returns a program of length at most k.

By a standard averaging argument, with probability at least $1 - \frac{1}{2p(n)}$ over $x \leftarrow D'_n$, $\mathcal{A}(x,r)$ will output a $K^t(x \mid r)$-witness with probability at least $1 - \frac{1}{p(n)}$. We refer to such x as being "good". We argue that on input a good x, $\mathcal{H}(x)$ fails to decide $\mathsf{MpK}^t\mathsf{P} = (\Pi_{\mathsf{YES}}, \Pi_{\mathsf{NO}})$ with probability at most $\frac{1}{2p(n)}$: If $(x,k) \in \Pi_{\mathsf{YES}}$, it follows that $K^t(x \mid r) \leq k$ with probability at least $2/3$, and $\mathcal{A}(x,r)$ will output a program with length $\leq k$ with probability $2/3 - \frac{1}{p(n)}$. By a standard Chernoff-type argument, if follows that $\mathcal{H}(x)$ will output 0 with probability at most $\frac{1}{2p(n)}$. If $(x,k) \in \Pi_{\mathsf{NO}}$, the same argument can be made and $\mathcal{H}(x)$ will output 1 with probability at most $\frac{1}{2p(n)}$. Combining this with the fact that x is "bad" with probability at most $\frac{1}{2p(n)}$, we conclude that the heuristic $\mathcal{H}$ fails with probability at most $\frac{1}{p(n)}$. Finally, note that this holds for infinitely many n, which is a contradiction.

3 Theorems

We state our main results in this section.

Theorem 31. *There exists a polynomial γ such that the following are equivalent:*

1. *The existence of a $t_D(\cdot)$-time samplable ensemble $\mathcal{D}$, a polynomial $t(\cdot)$, $t(n) \geq \gamma(t_D(n))$ such that Search-cK^t is mildly hard-on-average w.r.t. $(\mathcal{D},\mathcal{U})$.*
2. *The existence of a $t_D(\cdot)$-time samplable ensemble $\mathcal{D}$, a polynomial $t(\cdot)$, $t(n) \geq \gamma(t_D(n))$ such that $\mathsf{MpK}^t\mathsf{P}$ is mildly hard-on-average w.r.t. $\mathcal{D}$.*
3. *The existence of one-way functions.*
4. *For all polynomials $t(n)$, $t(n) > (1+\varepsilon)n$ for some $\varepsilon > 0$, $\mathsf{MpK}^t\mathsf{P}$ is mildly hard-on-average w.r.t. the uniform distribution.*

Proof. (2) $\Rightarrow$ (1) is proved in Lemma 26. The implication (1) $\Rightarrow$ (3) follows from Theorem 41 (stated and proved in Sect. 4). By Theorem 51 (stated and proved in Sect. 5), (3) implies (4). Finally, (4) trivially implies (2).

Theorem 32. *Assume that $\mathsf{E} \not\subseteq \mathsf{ioNSIZE}[2^{\Omega(n)}]$. There exists a polynomial γ such that the following are equivalent.*

1. *The existence of a $t_D(\cdot)$-time samplable ensemble $\mathcal{D}$, and a polynomial $t(\cdot)$, $t(n) \geq \gamma(t_D(n))$, such that Search-K^t is mildly hard-on-average w.r.t. $\mathcal{D}$.*
2. *The existence of one-way functions.*
3. *For all polynomials $t(n)$, $t(n) > (1+\varepsilon)n$ for some $\varepsilon > 0$, $\mathsf{MK}^t\mathsf{P}$ is mildly hard-on-average w.r.t. the uniform distribution.*

Proof. (1) $\Rightarrow$ (2) follows from Theorem 42 (stated and proved in Sect. 4). By Theorem 52, (2) implies (3). Finally, the implication (3) $\Rightarrow$ (1) is proved in Lemma 26.

4 OWFs from Hardness of $\mathsf{MpK}^{\mathsf{poly}}\mathsf{P}$ w.r.t. Any Samplable Distribution

In this section, we show that if there exists a samplable distribution under which Search-cK^{poly} is mildly hard-on-average, then one-way functions exist. In addition, this result can be extend to assuming mild average-case hardness of Search-K^{poly} under a derandomization assumption. Note that by Lemma 26, hardness of $\mathsf{MpK}^{\mathsf{poly}}\mathsf{P}$ (resp $\mathsf{MK}^{\mathsf{poly}}\mathsf{P}$) implies hardness of Search-cK^{poly} (resp Search-K^t). Therefore, we obtained OWFs assuming mild average-case hardness of $\mathsf{MpK}^{\mathsf{poly}}\mathsf{P}$ (resp of $\mathsf{MK}^{\mathsf{poly}}\mathsf{P}$).

Theorem 41. *There exists a polynomial γ such that the following holds. Assume that there exist a $t_D(\cdot)$-time samplable ensemble $\mathcal{D}$, and a polynomial $t(\cdot)$, $t(n) \geq \gamma(t_D(n))$, such that Search-cK^t is mildly HoA w.r.t. $(\mathcal{D},\mathcal{U})$. Then, one-way functions exist.*

Proof. This theorem follows from Lemma 43 (stated and proved in Sect. 4.1) and Lemma 46 (stated and proved in Sect. 4.2).

Theorem 42. *Assume that* $\mathsf{E} \not\subseteq \mathsf{ioNSIZE}[2^{\Omega(n)}]$. *There exists a polynomial* γ *such that the following holds. Assume that there exist a* $t_D(\cdot)$*-time samplable ensemble* $\mathcal{D}$*, and a polynomial* $t(\cdot)$*,* $t(n) \geq \gamma(t_D(n))$*, such that Search-*K^t *is mildly HoA w.r.t.* $\mathcal{D}$*. Then, one-way functions exist.*

Proof. This theorem follows from Lemma 44 (stated and proved in Sect. 4.1) and Lemma 48 (stated and proved in Sect. 4.2).

The above theorems are proved in two steps. We first show that if we only consider distributions that are "polynomially bounded by the complexity measure" (defined blow), we can deduce OWFs from average-case hardness of Search-cK^{poly} (if the complexity measure is cK^{poly}) or Search-K^{poly} (if K^{poly}). Then we show that any samplable distribution will be polynomial bounded by pK^{poly}, and by K^{poly} under derandomization assumption.

We turn to defining what it means for a distribution to be polynomially bounded by a complexity measure. For a distribution ensemble $\mathcal{D}$, we say that $\mathcal{D}$ is *polynomially bounded* by pK^t (resp by K^t) if there exists a polynomial $\delta(\cdot)$ such that for all string $x \in \{0,1\}^*$, $n = |x|$, $\Pr[x' \leftarrow D_n : x' = x] \leq \delta(n)2^{-pK^t(x)}$ (resp $\leq \delta(n)2^{-K^t(x)}$).

4.1 When $\mathcal{D}$ Is Polynomially Bounded

Lemma 43. *Assume that there exist a polynomial* t *and an ensemble* $\mathcal{D}$ *such that* $\mathcal{D}$ *is polynomially bounded by* pK^t *and Search-*cK^t *is mildly HoA w.r.t.* $\mathcal{D}$*. Then, weak one-way functions exist.*

Proof. Let c be the constant from Fact 24, and t be the polynomial as in the lemma statement. We consider the function $f : \{0,1\}^{\lceil \log(n+c) \rceil + n + c + t(n)} \to \{0,1\}^*$, which takes an input $\ell || \Pi' || r$ where $|\ell| = \lceil \log(n+c) \rceil$, $|\Pi'| = n + c$ and $|r| = t(n)$, outputs

$$f(\ell || \Pi' || r) = \ell || r || U(\Pi(r), 1^{t(n)})$$

where Π is a prefix of Π' and Π is of length ℓ (where the bit-string ℓ is interpreted as an integer $\in [n+c]$).

This function is only defined over some input lengths, but by an easy padding trick, it can be transformed into a function f' defined over all input lengths, such that if f is weakly one-way (over the restricted input lengths), then f' will be weakly one-way (over all input lengths): $f'(x')$ simply truncates its input x' (as little as possible) so that the (truncated) input x now becomes of length $n' = \lceil \log(n+c) \rceil + n + c + t(n)$ for some n and outputs $f(x)$. This will decrease the input length by a polynomial factor (since t is a polynomial) so the padding trick can be applied here.

We now show that f is a weak OWF (over the restricted input length) assuming that Search-cK^t is mildly HoA w.r.t. $(\mathcal{D},\mathcal{U})$. Since the search problem is mildly HoA and the distribution $\mathcal{D}$ is polynomially bounded by $pK^t_{1-2^{-n}}$, let p be the polynomial in the mild average-case hardness and δ be the polynomial in the bound where pK^t bounds $\mathcal{D}$. Let $q(n) = 16n\delta(n)p(n)^2$. We assume for contradiction that f is not $\frac{1}{q}$-weak one-way. (In the later proof, although the input length of f we consider is $m = \lceil \log(n+c) \rceil + n + c + t(n)$ for some n, we will view n as a "security parameter" and analyze the one-wayness of f on input length m with respect to n. Since n and m are polynomially related, we can still conclude that f is weak one-way.) Then, there exists a PPT attacker $\mathcal{A}$ that inverts f with probability at least $1 - \frac{1}{q(n)}$ on infinitely many n. We will use $\mathcal{A}$ to solve the cK^t search problem over $(\mathcal{D},\mathcal{U})$ with probability at least $1 - \frac{1}{p(n)}$ (and thus a contradiction).

Our search algorithm, $\mathcal{H}$, proceeds as follows. On input $z \leftarrow D_n, r \leftarrow \mathcal{U}_{t(n)}$, the search algorithm enumerates over all possible $i \in [n+c]$, and for each i, $\mathcal{H}$ will invoke the attacker $\mathcal{A}$ to invert f on the string $i||r||z$. And $\mathcal{H}$ will also check if the inversion succeeds. If the inverter succeeds, it will output a pre-image of $i||r||z$. By the definition of f, this string will be of the form $i||\Pi'||r$, and from which we can obtain a program Π of length i. Finally, $\mathcal{H}$ outputs the shortest program it obtains.

We turn to proving that $\mathcal{H}$ is a good search algorithm that succeeds with probability $1 - \frac{1}{p(n)}$. Fix some $n \in \mathbb{N}$ such that the inverter $\mathcal{A}$ succeeds on security parameter n. It is helpful here to introduce what it means for a string z to be "good": Let $\alpha = \frac{1}{2p(n)}$. For each string $z \in \{0,1\}^n$, $r \in \{0,1\}^{t(n)}$, we let $w_{z,r} = K^t(z \,|\, r)$ denote the length of the shortest program that outputs z on input r within $t(n)$ steps. We refer to a string $z \in \{0,1\}^n$ as being *good* if

$$\Pr[r \leftarrow \{0,1\}^{t(n)} : \mathcal{A}(w_{z,r}||r||z) \text{ succeeds}] \geq 1 - \alpha(n)$$

where the probability is also taken over the internal randomness of $\mathcal{A}$, and the inverter $\mathcal{A}$ succeeds on $w_{z,r}||r||z$ if it returns a valid pre-image. Notice that by our choice of $w_{z,r}$, it is guaranteed that a pre-image must exist. Let G denote the set of all good strings $z \in \{0,1\}^n$. We first claim that heuristic $\mathcal{H}$ will succeeds with high probability on any good z.

Claim 1. *For any $z \in \{0,1\}^n$, if z is good, then $\mathcal{H}(z,r)$ fails with probability at most $\frac{1}{2p(n)}$ over random $r \leftarrow \{0,1\}^{t(n)}$.*

Proof. Notice that for any z, r, if the inverter succeeds in inverting f on $w_{z,r}||r||z$, it will obtain a program Π of length $w_{z,r}$ that on input r, outputs z within $t(n)$ steps. By our choice of $w_{z,r}$, this will be the shortest such program, $\mathcal{H}(z,r)$ will finally output it as a witness. Therefore, if z is good, then $\mathcal{H}(z,r)$ will output a valid $K^t(z \,|\, r)$-witness with probability at least $1 - \alpha(n) = 1 - \frac{1}{2p(n)}$ over r.

On the other hand, since our inverter $\mathcal{A}$ succeeds with high probability, there should be only "a few" bad strings. We assume for contradiction that the total

probability weight of bad strings (w.r.t. D_n) is $\geq \frac{1}{2p(n)}$. That is,

$$\sum_{z \notin G} \Pr[z' \leftarrow D_n : z' = z] \geq \frac{1}{2p(n)}$$

Recall that the distribution $\mathcal{D}$ is polynomial bounded by pK^t. So for any string $z \notin G$, the probability that z is sampled by D_n is at most $\delta(n)2^{-pK^t(z)}$. It follows that

$$\sum_{z \notin G} \delta(n)2^{-pK^t(z)} \geq \sum_{z \notin G} \Pr[z' \leftarrow D_n : z' = z] \geq \frac{1}{2p(n)} \tag{1}$$

However, z will be sampled with probability at least

$$\frac{1}{2^{t(n)}} \sum_{r \in \{0,1\}^{t(n)}} \frac{1}{n+c} \frac{1}{2^{w_{z,r}}}$$

in the one-way function experiment, since for each randomness r, there exists a program of length $w_{z,r} = K^t(z \mid r)$ that outputs z (on input r) within $t(n)$ steps, and the OWF will output z if the program is picked. By the definition of pK^t, it follows that z will be sampled with probability at least

$$\frac{1}{n+c}\left(2^{-pK^t(z)} \cdot 2/3\right) \geq \frac{1}{2(n+c)} 2^{-pK^t(z)} \geq \frac{1}{4n} 2^{-pK^t(z)}$$

When a bad string z is sampled, the inverter $\mathcal{A}$ will fail with probability at least $\frac{1}{\alpha(n)}$. Thus, $\mathcal{A}$ will fail with probability at least

$$\sum_{z \notin G} \frac{1}{4n\alpha(n)} 2^{-pK^t(z)}$$

By Eq. 1, this is at least

$$\frac{1}{4n\alpha(n)} \cdot \frac{1}{2p(n)\delta(n)} > \frac{1}{q(n)}$$

which is a contradiction (since $\mathcal{A}$ is a good inverter). We thus conclude that the total probability weight of bad strings (w.r.t. D_n) is $\leq \frac{1}{2p(n)}$. Finally, by a Union bound (also taking into account good strings) and Claim 1, the probability that the heuristic fails w.r.t. D_n is at most $\frac{1}{p(n)}$.

Combining the above argument with the fact that the attacker $\mathcal{A}$ succeeds on infinitely many input lengths n, we conclude that $\mathcal{H}$ fails with probability at most $\frac{1}{p(n)}$ on infinitely many n, which is a contradiction.

We can prove that Lemma 43 also holds for K^t analogously.

Lemma 44. *Assume that there exist a polynomial t and an ensemble $\mathcal{D}$ such that $\mathcal{D}$ is polynomially bounded by K^t and Search-K^t is mildly HoA w.r.t. $\mathcal{D}$. Then, weak one-way functions exist.*

Proof. This lemma will be proved using similar ideas in the proof of Lemma 43, so the proof is presented based on the proof of Lemma 43. We assume familiarity of the proof of Lemma 43.

We will consider roughly the same OWF construction as in Lemma 43, except that we no longer need to sample a random string r in the construct. We consider the function $f : \{0,1\}^{\lceil \log(n+c) \rceil + n + c} \to \{0,1\}^*$, which takes an input $\ell || \Pi'$ where $|\ell| = \lceil \log(n+c) \rceil$, and $|\Pi'| = n + c$, outputs

$$f(\ell || \Pi') = \ell || U(\Pi(r), 1^{t(n)})$$

where Π is a prefix of Π' and Π is of length ℓ (where the bit-string ℓ is interpreted as an integer $\in [n+c]$).

This function is also only defined over some input lengths, but it suffices to show that f is a weak OWF over input lengths on which f is well defined. As showed in Lemma 43, by using a padding trick, we can transform this function f to a standard OWF.

In this lemma, we will show that f is a weak OWF (over the restricted input length) assuming that K^t is mildly HoA to search w.r.t. $\mathcal{D}$. Since Search-K^t is mildly HoA and $\mathcal{D}$ is polynomially bounded by K^t, let p be the polynomial in the mild average-case hardness and δ be the polynomial in the bound where K^t bounds $\mathcal{D}$. Let $q(n) = 8n\delta(n)p(n)^2$. We assume for contradiction that f is not $\frac{1}{q}$-weak one-way. (In the later proof, although the input length of f we consider is $m = \lceil \log(n+c) \rceil + n + c + t(n)$ for some n, we will view n as a "security parameter", as in Lemma 43.) Then, there exists a PPT attacker $\mathcal{A}$ that inverts f with probability at least $1 - \frac{1}{q(n)}$ on infinitely many n. We will use $\mathcal{A}$ to solve the Search-K^t problem over $\mathcal{D}$ with probability at least $1 - \frac{1}{p(n)}$ (and thus a contradiction).

Our search algorithm for K^t, $\mathcal{H}$, proceeds as follows. On input $z \leftarrow D_n$, $\mathcal{H}$ will enumerate over all possible $i \in [n+c]$, and for each i, the heuristic will invoke the attacker $\mathcal{A}$ to invert f on the string $i||z$. And $\mathcal{H}$ will also check if the inversion succeeds. If the inverter succeeds, it will output a pre-image of $i||z$. By the definition of f, this string will be of the form $i||\Pi'$, and from which we can obtain a program Π of length i. Finally, $\mathcal{H}$ outputs the shortest program it obtains.

We turn to proving that $\mathcal{H}$ is a good search algorithm that succeeds with probability $1 - \frac{1}{p(n)}$. Fix some $n \in \mathbb{N}$ such that the inverter $\mathcal{A}$ succeeds on security parameter n. As in Lemma 43, it is also helpful here to introduce what it means for a string z to be "good": Let $\alpha(n) = \frac{1}{2p(n)}$. For each string $z \in \{0,1\}^n$, we let $w_z = K^t(z)$ denote the length of the shortest program that outputs z within $t(n)$ steps. We refer to a string $z \in \{0,1\}^n$ as being *good* if

$$\Pr[\mathcal{A}(w_z || z) \text{ succeeds}] \geq 1 - \alpha(n)$$

where the probability is taken over the internal randomness of $\mathcal{A}$. Notice that by our choice of w_z, it is guaranteed that a pre-image must exist. Let G denote the

set of all good strings $z \in \{0,1\}^n$. We first claim that heuristic $\mathcal{H}$ will succeeds with high probability on any good z (similar to the proof of Lemma 43).

Claim 2. *For any $z \in \{0,1\}^n$, if z is good, $\mathcal{H}(z)$ fails with probability at most $\frac{1}{2p(n)}$.*

Proof. Note that if the inverter $\mathcal{A}$ succeeds on $w_z||z$, it will obtain a program Π which is a K^t-witness of z – Π will output z within time $t(n)$ and it's of length $K^t(z)$. Therefore, $\mathcal{H}(z)$ will eventually output this program. This implies that if z is good, $\mathcal{H}(z)$ will find a correct witness with probability at least $1 - \alpha(n) = 1 - \frac{1}{2p(n)}$

On the other hand, since our inverter $\mathcal{A}$ succeeds with high probability, there should be only "a few" bad strings. We assume for contradiction that the total probability weight of bad strings (w.r.t. D_n) is $\geq \frac{1}{2p(n)}$. That is,

$$\sum_{z \notin G} \Pr[z' \leftarrow D_n : z' = z] \geq \frac{1}{2p(n)}$$

Recall that the distribution $\mathcal{D}$ is polynomial bounded by K^t. So for any string $z \notin G$, the probability that z is sampled by D_n is at most $\delta(n)2^{-K^t(z)}$. It follows that

$$\sum_{z \notin G} \delta(n)2^{-K^t(z)} \geq \sum_{z \notin G} \Pr[z' \leftarrow D_n : z' = z] \geq \frac{1}{2p(n)} \tag{2}$$

However, z will be sampled with probability at least

$$\frac{1}{n+c}\frac{1}{2^{w_z}} = \frac{1}{n+c}\frac{1}{2^{K^t(z)}}$$

in the one-way function experiment, since there exists a program of length $w_z = K^t(z)$ that outputs z within $t(n)$ steps, and the OWF will output z if the program is picked. When a bad string z is sampled, the inverter $\mathcal{A}$ will fail with probability at least $\frac{1}{\alpha(n)}$. Thus, $\mathcal{A}$ will fail with probability at least

$$\sum_{z \notin G} \frac{1}{2n\alpha(n)} 2^{-K^t(z)}$$

By Eq. 2, this is at least

$$\frac{1}{2n\alpha(n)} \cdot \frac{1}{2p(n)\delta(n)} > \frac{1}{q(n)}$$

which is a contradiction (since $\mathcal{A}$ is a good inverter). We thus conclude that the total probability weight of bad strings (w.r.t. D_n) is $\leq \frac{1}{2p(n)}$. Finally, by a Union bound (to take into account good strings) and Claim 2, the probability that the heuristic fails w.r.t. D_n is at most $\frac{1}{p(n)}$.

4.2 Bounding Any Samplable Distribution

We proceed to showing that samplable distributions are bounded by the complexity measures we consider. We first focus our attention to pK^{poly}. Towards this, let us recall the Coding Theorem for pK^{poly}.

Theorem 45. ([25, Theorem 30]) *There exists a polynomial γ such that for any t_D-time samplable ensemble $\mathcal{D}$, any string $x \in \{0,1\}^n$ such that D_n samples x with probability $\delta > 0$, for any polynomial t such that $t(n) \geq \gamma(t_D(n))$, it holds that*

$$pK^t(x) \leq \log(1/\delta) + O(\log t_D(n))$$

The coding theorem roughly says that if a (samplable) distribution assign too much weight to an individual string, we will be able to find a short description of that string. We next use the coding theorem to show that any samplable distribution can be bounded by pK^{poly}.

Lemma 46. *There exists a polynomial γ such that for all polynomial t, any t_D-time samplable ensemble $\mathcal{D}$ is polynomially bounded by pK^t if $t(n) \geq \gamma(t_D(n))$.*

Proof. Let γ be the polynomial in Theorem 45. It follows that for any t_D-time $\mathcal{D}$, any $x, \delta = \Pr[D_n = x]$, and any polynomial $t(n) \geq \gamma(t_D(n))$, it holds that

$$1/\delta \cdot t_D(n)^{O(1)} \geq 2^{pK^t(x)}$$

which implies that $\delta \leq t_D(n)^{O(1)} \cdot 2^{-pK^t(x)}$. Notice that $t_D(n)^{O(1)}$ is a polynomial, and the above equation holds for any x in the support of $\mathcal{D}$, we thus conclude that $\mathcal{D}$ is polynomially bounded by pK^t.

The above proof relies on Theorem 45, whose proof uses tools from complexity theory (unconditional PRGs that fool constant-depth circuits). As a result of independent interest, in Appendix A, we present an alternative direct proof of Lemma 46 (which implicitly also proves Theorem 45) using only elementary machinery (in particular, simply universal hash functions).

We move on to considering K^{poly}. As observed by [7], K^{poly} is at most pK^{poly} up to an additive $O(\log n)$ factor under a derandomization assumption.

Proposition 47 ([7, Proposition 66])**.** *Assume that $\mathsf{E} \not\subseteq \mathsf{ioNSIZE}[2^{\Omega(n)}]$. It holds that there exists a polynomial p such that for every polynomial t, t', $t'(n) \geq p(t(n))$, for every $x \in \{0,1\}^n$, it holds that*

$$K^{t'}(x) \leq pK^t(x) + \log(t(n))$$

Combining the above Proposition and Lemma 46, we prove that any samplable distribution will also be bounded by K^{poly}. (In addition, we also give another proof of this statement (see the Lemma below for a formal version) in Appendix A without using Proposition 47, for concreteness.)

Lemma 48. *Assume that* $\mathsf{E} \not\subseteq \mathsf{ioNSIZE}[2^{\Omega(n)}]$. *There exists a polynomial* γ' *such that for all polynomial* t', *any* t_D*-time samplable ensemble* $\mathcal{D}$ *is polynomially bounded by* $K^{t'}$ *if* $t'(n) \geq \gamma'(t_D(n))$.

Proof. The proof of Lemma 48 relies on the proof of Lemma 46, and we refer the reader to the proof of Lemma 46 for notations used in this proof. Let p be the polynomial in Proposition 47. Recall that the proof of Lemma 46 showed that $\delta \leq t_D(n)^{O(1)} \cdot 2^{-pK^t(x)}$. We now consider any polynomial t' such that $t'(n) \geq p(t(n))$. By Proposition 47, it follows that

$$\delta \leq t_D(n)^{O(1)} \cdot 2^{-pK^t(x)} \leq t_D(n)^{O(1)} \cdot 2^{-K^{t'}(x)+\log(t(n))} = t_D(n)^{O(1)} t(n) \cdot 2^{-K^{t'}(x)}$$

Thus, we conclude that $\mathcal{D}$ is polynomially bounded by $K^{t'}$.

5 Hardness of $\mathsf{MpK}^{\mathsf{poly}}\mathsf{P}$ w.r.t. Uniform from OWFs

We here show that that for every polynomial $t(n) \geq 1.1n$, the existence of OWFs implies mild average-case hardness of $\mathsf{MpK}^{\mathsf{poly}}\mathsf{P}$ and mild average-case hardness of $\mathsf{MK}^{\mathsf{poly}}\mathsf{P}$. Our proof closely follows the proof in [21] with only minor modifications to deal with the fact that we now consider the probabilistic variant of Kolmogorov complexity and we focus on languages/promise problems.

Theorem 51. *Assume that one-way functions exist. Then,* $\mathsf{MpK}^t\mathsf{P}$ *is mildly hard-on-average with respect to the uniform distribution.*

Proof. This theorem follows from Theorem 55 and Theorem 56.

Theorem 52. *Assume that one-way functions exist. Then,* $\mathsf{MK}^t\mathsf{P}$ *is mildly hard-on-average with respect to the uniform distribution.*

Proof. This theorem follows from Theorem 55 and Theorem 57.

5.1 Some Additional Preliminaries

Let us first recall some additional standard preliminaries.

Computational Indistinguishability. We recall the definition of (computational) indistinguishability [9].

Definition 53. *Two ensembles* $\{A_n\}_{n\in\mathbb{N}}$ *and* $\{B_n\}_{n\in\mathbb{N}}$ *are said to be* $\mu(\cdot)$- indistinguishable, *if for every probabilistic machine* D *(the "distinguisher") whose running time is polynomial in the length of its first input, there exists some* $n_0 \in \mathbb{N}$ *so that for every* $n \geq n_0$*:*

$$|\Pr[D(1^n, A_n) = 1] - \Pr[D(1^n, B_n) = 1]| < \mu(n)$$

We say that are $\{A_n\}_{n\in\mathbb{N}}$ *and* $\{B_n\}_{n\in\mathbb{N}}$ *simply* indistinguishable *if they are* $\frac{1}{p(\cdot)}$*-indistinguishable for every polynomial* $p(\cdot)$.

Statistical Distance and Entropy. For any two random variables X and Y defined over some set $\mathcal{V}$, we let $\mathsf{SD}(X,Y) = \frac{1}{2}\sum_{v\in\mathcal{V}} |\Pr[X=v] - \Pr[Y=v]|$ denote the *statistical distance* between X and Y. For a random variable X, let $H(X) = \mathsf{E}[\log \frac{1}{\Pr[X=x]}]$ denote the (Shannon) entropy of X, and let $H_\infty(X) = \min_{x\in Supp(X)} \log \frac{1}{\Pr[X=x]}$ denote the *min-entropy* of X.

5.2 Entropy-Preserving PRGs

Liu and Pass [21] defined a notion of a *conditionally-secure entropy-preserving pseudorandom generator (cond EP-PRG)*. Roughly speaking, a cond EP-PRG is a function where the output is indistinguishable from the uniform distribution and also preserves the entropy in the input only when *conditioned on* some event E.

Definition 54. *An efficiently computable function $G : \{0,1\}^n \to \{0,1\}^{n+\gamma\log n}$ is a $\mu(\cdot)$-*conditionally secure entropy-preserving pseudorandom generator (μ-cond EP-PRG) *if there exist a sequence of events $= \{E_n\}_{n\in\mathbb{N}}$ and a constant α (referred to as the* entropy-loss constant*) such that the following conditions hold:*

- **(pseudorandomness):** *$\{G(\mathcal{U}_n \mid E_n)\}_{n\in\mathbb{N}}$ and $\{\mathcal{U}_{n+\gamma\log n}\}_{n\in\mathbb{N}}$ are $\mu(n)$-indistinguishable;*
- **(entropy-preserving):** *For all sufficiently large $n \in \mathbb{N}$, $H(G(\mathcal{U}_n \mid E_n)) \geq n - \alpha\log n$.*

We say that G has *rate-1 efficiency* if its running time on inputs of length n is bounded by $n + O(n^\varepsilon)$ for some constant $\varepsilon < 1$.

Theorem 55. ([21]) *Assume that OWFs exist. Then, for every $\gamma > 1$, there exists a rate-1 efficient μ-cond-EP PRG $G_\gamma : \{0,1\}^n \to \{0,1\}^{n+\gamma\log n}$, where $\mu = \frac{1}{n^2}$.*

Though in [21], running time was counted in terms of execution on Turing machines, as noted in [23], the PRG is also rate-1 efficient when run on a RAM.

5.3 Hardness of $\mathsf{MpK}^t\mathsf{P}$ and $\mathsf{MK}^t\mathsf{P}$ from Cond EP-PRG

Theorem 56. *Assume that for some $\gamma \geq 4$, there exists a rate-1 efficient μ-cond EP-PRG $G : \{0,1\}^n \to \{0,1\}^{n+\gamma\log n}$ where $\mu(n) = 1/n^2$. Then, for every $\epsilon > 0$, all $t(n) \geq (1+\epsilon)n$, $\mathsf{MpK}^t\mathsf{P}$ is mildly HoA w.r.t. the uniform distribution.*

Proof. The proof follows exactly the same structure as the proof [21] with only minor adjustments to deal with the fact that we now consider probabilistic Kolmogorov complexity and $\mathsf{MpK}^t\mathsf{P}$, a promise problem. Essentially, the key observation is that random strings have high probabilistic Kolmogorov complexity, and due to this observation, essentially the proof in [21] can still be applied. We proceed to the full details.

Let $\gamma \geq 4$, and let $G' : \{0,1\}^n \to \{0,1\}^{m'(n)}$ where $m'(n) = n + \gamma \log n$ be a rate-1 efficient μ-cond EP-PRG, where $\mu = 1/n^2$. For any constant c, let $G^c(x)$ be a function that computes $G'(x)$ and truncates the last c bits. It directly follows that G^c is also a rate-1 efficient μ-cond EP-PRG (since G' is so). Consider any $\varepsilon > 0$ and any polynomial $t(n) \geq (1+\varepsilon)n$ and let $p(n) = 2n^{2(\alpha+\gamma+1)}$.

Assume for contradiction that there exists some PPT $\mathcal{H}(x,k)$ that decides $\mathsf{MpK}^t\mathsf{P}$ with probability $1 - \frac{1}{p(m)}$ over random $x \in \{0,1\}^m, k \in \{0,1\}^{\lceil \log m \rceil}$ for infinitely many $m \in \mathbb{N}$. Since $m'(n+1) - m'(n) \leq \gamma + 1$, there must exist some constant $c \leq \gamma + 1$ such that $\mathcal{H}$ succeeds (to decide $\mathsf{MpK}^t\mathsf{P}$) with probability $1 - \frac{1}{p(m)}$ for infinitely many m of the form $m = m(n) = n + \gamma \log n - c$. Let $G(x) = G^c(x)$; recall that G is a rate-1 efficient μ-cond EP-PRG (trivially, since G^c is so), and let $\alpha, \{E_n\}$, respectively, be the entropy loss constant and sequence of events, associated with it.

We next show that $\mathcal{H}$ can be used to break the cond EP-PRG G. Towards this, note that a random string still has high pK^t-complexity with high probability: for $m = m(n)$, by Fact 24, we have,

$$\Pr_{x \in \{0,1\}^m}[pK^t_{1/3}(x) > m - \frac{\gamma}{2}\log n] \geq 1 - \frac{3}{n^{\gamma/2}}, \tag{3}$$

However, any string output by G, must have "low" pK^t complexity : For every sufficiently large $n, m = m(n)$, we have that,

$$\Pr_{x \in \{0,1\}^n}[pK^t_1(G(x)) > m - \frac{\gamma}{2}\log n] = 0, \tag{4}$$

since for every string $r \in \{0,1\}^{t(m)}$, $G(x)$ can be produced by a program Π with the seed x of length n and the code of G (of constant length) hardwired in it (and the string r is skipped). The running time of Π is bounded by $t(m)$ for all sufficiently large n (since G is rate-1 efficient) , so $K^t(G(x)) = n + O(1) \leq m - \gamma/2 \log n$ for sufficiently large n (since recall that $\gamma \geq 4$).

Based on these observations, we now construct a PPT distinguisher $\mathcal{A}$ breaking G. On input $1^n, x$, where $x \in \{0,1\}^{m(n)}$, $\mathcal{A}(1^n, x)$ picks $k = m - \frac{\gamma}{2}\log n$. $\mathcal{A}$ outputs 1 if $\mathcal{H}(x,k)$ outputs 1 and 0 otherwise. Fix some n, $m = m(n)$ for which $\mathcal{H}$ succeeds to decide $\mathsf{MpK}^t\mathsf{P}$ with probability $\frac{1}{p(m)}$. The following two claims conclude that $\mathcal{A}$ distinguishes $\mathcal{U}_{m(n)}$ and $G(\mathcal{U}_n \mid E_n)$ with probability at least $\frac{1}{n^2}$.

Claim 3. *$\mathcal{A}(1^n, \mathcal{U}_m)$ outputs 0 with probability at least $1 - \frac{4}{n^{\gamma/2}}$.*

Proof. Note that $\mathcal{A}(1^n, x)$ will output 0 if (1) x is a string with $pK^t_{1/3}$-complexity larger than $m - \gamma/2 \log n$ and (2) $\mathcal{H}$ succeeds on input (x,k). (Note that if (1)

holds, (x, k) is guaranteed to be a NO instance in $\mathsf{MpK}^t\mathsf{P}$.) Thus,

$$\begin{aligned}
&\Pr[\mathcal{A}(1^n, x) = 0] \\
&\geq \Pr[pK^t_{1/3}(x) > m - \gamma/2 \log n \wedge \mathcal{H} \text{ succeeds on } (x, k)] \\
&\geq 1 - \Pr[pK^t_{1/3}(x) \leq m - \gamma/2 \log n] - \Pr[\mathcal{H} \text{ fails on } (x, k)] \\
&\geq 1 - \frac{3}{n^{\gamma/2}} - \frac{1}{p(m)} \\
&\geq 1 - \frac{4}{n^{\gamma/2}}.
\end{aligned}$$

where the probability is over a random $x \leftarrow \mathcal{U}_m$, $k \leftarrow \lceil \log m \rceil$ and the randomness of $\mathcal{A}$ and $\mathcal{H}$.

Claim 4. *$\mathcal{A}(1^n, G(\mathcal{U}_n \mid E_n))$ outputs 0 with probability at most $1 - \frac{1}{n} + \frac{2}{n^2}$*

Proof. Recall that by assumption, $\mathcal{H}(x, k)$ fails to decide whether $(x, k) \in \mathsf{MpK}^t\mathsf{P}$ for a random $x \in \{0,1\}^m, k \in \{0,1\}^{\lceil \log m \rceil}$ with probability at most $\frac{1}{p(m)}$.

By an averaging argument, for at least a $1 - \frac{1}{n^2}$ fraction of random tapes r for $\mathcal{H}$, the deterministic machine $\mathcal{H}_r$ fails to decide $\mathsf{MpK}^t\mathsf{P}$ with probability at most $\frac{n^2}{p(m)}$. Fix some "good" randomness r such that $\mathcal{H}_r$ decides $\mathsf{MpK}^t\mathsf{P}$ with probability at least $1 - \frac{n^2}{p(m)}$.

We next analyze the success probability of $\mathcal{A}_r$. Assume for contradiction that A_r outputs 1 with probability at least $1 - \frac{1}{n} + \frac{1}{n^{\alpha+\gamma}}$ on input $G(\mathcal{U}_n \mid E_n)$. Recall that (1) the entropy of $G(\mathcal{U}_n \mid E_n)$ is at least $n - \alpha \log n$ and (2) the quantity $-\log \Pr[G(\mathcal{U}_n \mid E_n) = y]$ is upper bounded by n for all $y \in G(\mathcal{U}_n \mid E_n)$. By an averaging argument, with probability at least $\frac{1}{n}$, a random $y \in G(\mathcal{U}_n \mid E_n)$ will satisfy

$$-\log \Pr[G(\mathcal{U}_n \mid E_n) = y] \geq (n - \alpha \log n) - 1.$$

We refer to an output y satisfying the above condition as being "good" and other y's as being "bad". Let $S = \{y \in G(\mathcal{U}_n \mid E_n) : \mathcal{A}_r(1^n, y) = 0 \wedge y \text{ is good}\}$, and let $S' = \{y \in G(\mathcal{U}_n \mid E_n) : \mathcal{A}_r(1^n, y) = 0 \wedge y \text{ is bad}\}$. Since

$$\Pr[\mathcal{A}_r(1^n, G(\mathcal{U}_n \mid E_n)) = 0] = \Pr[G(\mathcal{U}_n \mid E_n) \in S] + \Pr[G(\mathcal{U}_n \mid E_n) \in S'],$$

and $\Pr[G(\mathcal{U}_n \mid E_n) \in S']$ is at most the probability that $G(\mathcal{U}_n \mid E_n)$ is "bad" (which as argued above is at most $1 - \frac{1}{n}$), we have that

$$\Pr[G(\mathcal{U}_n \mid E_n) \in S] \geq \left(1 - \frac{1}{n} + \frac{1}{n^{\alpha+\gamma}}\right) - \left(1 - \frac{1}{n}\right) = \frac{1}{n^{\alpha+\gamma}}.$$

Furthermore, since for every $y \in S$, $\Pr[G(\mathcal{U}_n \mid E_n) = y] \leq 2^{-n+\alpha \log n+1}$, we also have,

$$\Pr[G(\mathcal{U}_n \mid E_n) \in S] \leq |S| 2^{-n+\alpha \log n+1}$$

So,

$$|S| \geq \frac{2^{n-\alpha \log n-1}}{n^{\alpha+\gamma}} = 2^{n-(2\alpha+\gamma)\log n-1}$$

However, for any $y \in G(\mathcal{U}_n \mid E_n)$, if $\mathcal{A}_r(1^n, y)$ outputs 0, then by Eq. 4, $pK_1^t(y) \leq m-\gamma/2 \log n = k$ (and therefore a YES instance in $\mathsf{MpK}^t\mathsf{P}$), so $\mathcal{H}_r$ fails to decide $\mathsf{MpK}^t\mathsf{P}$ on input $(y, m-\gamma/2\log n)$.

Thus, the probability that $\mathcal{H}_r$ fails (to decide $\mathsf{MpK}^t\mathsf{P}$) on a random input (y,k) (where y and k are uniformly sampled in $\{0,1\}^m$ and $\{0,1\}^{\lceil \log m \rceil}$) is at least

$$|S|/2^{m+\lceil \log m \rceil} = \frac{2^{n-(2\alpha+\gamma)\log n-1}}{2^{n+\gamma \log n+\lceil \log m \rceil}} \geq \frac{1}{2n^{2(\alpha+\gamma+1)}}$$

which contradicts the fact that $\mathcal{H}_r$ fails to decide $\mathsf{MpK}^t\mathsf{P}$ with probability at most $\frac{n^2}{p(m)} < \frac{1}{2n^{2(\alpha+\gamma+1)}}$ (since $n < m$).

We conclude that for every good randomness r, $\mathcal{A}_r$ outputs 0 with probability at most $1 - \frac{1}{n} + \frac{1}{n^{\alpha+\gamma}}$. Finally, by union bound (and since a random tape is bad with probability $\leq \frac{1}{n^2}$), we have that the probability that $\mathcal{A}(G(\mathcal{U}_n \mid E_n))$ outputs 1 is at most

$$\frac{1}{n^2} + \left(1 - \frac{1}{n} + \frac{1}{n^{\alpha+\gamma}}\right) \leq 1 - \frac{1}{n} + \frac{2}{n^2},$$

since $\gamma \geq 2$.

We conclude, recalling that $\gamma \geq 4$, that $\mathcal{A}$ distinguishes $\mathcal{U}_m$ and $G(\mathcal{U}_n \mid E_n)$ with probability of at least

$$\left(1 - \frac{4}{n^{\gamma/2}}\right) - \left(1 - \frac{1}{n} + \frac{2}{n^2}\right) \geq \left(1 - \frac{4}{n^2}\right) - \left(1 - \frac{1}{n} + \frac{2}{n^2}\right) = \frac{1}{n} - \frac{6}{n^2} \geq \frac{1}{n^2}$$

for infinitely many $n \in \mathbb{N}$.

Theorem 57. *Assume that for some $\gamma \geq 4$, there exists a rate-1 efficient μ-cond EP-PRG $G : \{0,1\}^n \to \{0,1\}^{n+\gamma \log n}$ where $\mu(n) = 1/n^2$. Then, for every $\epsilon > 0$, all $t(n) \geq (1+\epsilon)n$, $\mathsf{MK}^t\mathsf{P}$ is mildly HoA w.r.t. the uniform distribution.*

Proof. The proof of Theorem 56 can also prove this theorem by replacing pK^t with K^t, and $\mathsf{MpK}^t\mathsf{P}$ with $\mathsf{MK}^t\mathsf{P}$.

A An Alternative Proof of Lemma 46 and Lemma 48

As mentioned in Sect. 4.2, we provide direct proofs for Lemma 46 and Lemma 48. Let us start by a reminder of the statement of Lemma 46.

Lemma A1. (Lemma 46, restated). *There exists a polynomial γ such that for all polynomial t, any t_D-time samplable ensemble $\mathcal{D}$ is polynomially bounded by pK^t if $t(n) \geq \gamma(t_D(n))$.*

We recall the notion of a universal hash function [3].

Definition A2. *Let $\mathcal{H}_m^\ell$ be a family of functions where $m < \ell$ and each function $h \in \mathcal{H}_m^\ell$ maps $\{0,1\}^\ell$ to $\{0,1\}^m$. We say that $\mathcal{H}_m^\ell$ is a* universal hash family *if (i) the functions $h_\sigma \in \mathcal{H}_m^\ell$ can be described by a string σ of ℓ^c bits where c is a universal constant that does not depend on ℓ; (ii) for all $x \neq x' \in \{0,1\}^\ell$, and for all $y, y' \in \{0,1\}^m$*

$$\Pr[h_\sigma \leftarrow \mathcal{H}_m^\ell : h_\sigma(x) = y \ \text{ and } \ h_\sigma(x') = y'] = 2^{-2m}$$

We will rely on the following properties of universal hash functions.

Proposition A3. *Let $\ell \in \mathbb{N}$, $S \subseteq \{0,1\}^\ell$ be a set, $\mathcal{H}_m^\ell$ be a universal hash family such that $m \leq \log |S|$. The following statements hold:*

- *With probability at least $1 - 2^{-\log|S|+m+3}$ over $h_\sigma \leftarrow \mathcal{H}_m^\ell$, there exists $s \in S$ such that $h_\sigma(s) = 0^m$.*
- *With probability at least $1 - 2^{-\ell+m+3}$ over $h_\sigma \leftarrow \mathcal{H}_m^\ell$, $|h_\sigma^{-1}(0^m)| \leq 2 \cdot 2^{\ell-m}$.*

For completeness, we provide the proof of Proposition A3 here.

Proof. We first prove the former statement. We consider picking a random hash function $h_\sigma \leftarrow \mathcal{H}_m^\ell$. For each element $s \in S$, let X_s denote the random variable such that $X_s = 1$ iff $h_\sigma(s) = 0^m$. Let X denote the random variable $X = \sum_{s \in S} X_s$. Note that $\mathsf{E}[X] = |S|/2^m$ and the variance of X is

$$\mathsf{V}(X) = \mathsf{E}[X^2 - \mathsf{E}[X]^2] = |S|(\frac{1}{2^m} - \frac{1}{2^{2m}}) \leq \mathsf{E}[X]$$

since $\mathcal{H}_m^\ell$ is a universal hash family and all $s_1, s_2 \in S$, X_{s_1} and X_{s_2} are independent. Therefore the variance of X is very small and we can apply Chebyshev's Inequality to show that

$$\begin{aligned}\Pr[X = 0] &\leq \Pr[|X - \mathsf{E}[X]| \geq \mathsf{E}[X] - 1] \\ &\leq \Pr[|X - \mathsf{E}[X]| \geq (\sqrt{\mathsf{V}(X)}/2)\sqrt{\mathsf{V}(X)}] \\ &\leq \frac{1}{\mathsf{V}(X)/4} \leq 2^{-\log|S|+m+3}\end{aligned}$$

So we conclude that with probability at least $1 - \Pr[X = 0] \geq 1 - 2^{-\log|S|+m+3}$, there exists $s \in S$ such that $h_\sigma(s) = 0^m$.

The latter statement follows from essentially the same proof. For each element $z \in \{0,1\}^\ell$, let Y_z denote the random variable such that $Y_z = 1$ iff $h_\sigma(z) = 0^m$. Let Y denote the random variable $Y = \sum_{z \in \{0,1\}^\ell} Y_z$. Note that $\mathsf{E}[Y] = 2^\ell/2^m$ and the variance of Y is

$$\mathsf{V}(Y) = \mathsf{E}[Y^2 - \mathsf{E}[Y]^2] = 2^\ell(\frac{1}{2^m} - \frac{1}{2^{2m}})$$

since $\mathcal{H}_m^\ell$ is a universal hash family and all $z_1, z_2 \in \{0,1\}^\ell$, Y_{z_1} and Y_{z_2} are independent.. Notice that by Chebyshev's inequality,

$$\Pr[Y \geq 2 \cdot 2^{\ell-m}] \leq \Pr[|Y - \mathsf{E}[Y]| \geq (\sqrt{\mathsf{V}(Y)}/2)\sqrt{\mathsf{V}(Y)}] \leq \frac{1}{\mathsf{V}(Y)/4} \leq 2^{-\ell+m+3}$$

So we conclude that with probability at least $1 - 2^{-\ell+m+3}$, $|h_\sigma^{-1}(0^m)| \leq 2 \cdot 2^{\ell-m}$.

We turn to introducing the linear universal hash family construction [3].

Proposition A4 ([3]). *Let $\ell, m \in \mathbb{N}, m < \ell$. For each $\sigma \in \{0,1\}^{\ell m+m}$, define h_σ to be the function such that for each $x \in \{0,1\}^\ell$, $h_\sigma(x) = Ax + b$ where $\sigma = (A, b)$, A is a binary matrix of $m \times \ell$, and b is a binary vector of length m. Let $\mathcal{H}_m^\ell = \{h_\sigma \,|\, \sigma \in \{0,1\}^{\ell m+m}\}$.*

Then, it holds that $\mathcal{H}_m^\ell$ is a universal hash family.

We are now ready to prove Lemma A1.

Proof. Consider any polynomial t, and any t_D-time samplable ensemble $\mathcal{D}$. Let M be the PPT sampler such that $M(1^n, r)$ uses $r \in \{0,1\}^{t_D(n)}$ as random coins and samples D_n for each $n \in \mathbb{N}$.

We will show that $\mathcal{D}$ is polynomially bounded by $pK^t_{1-2^{-n}}$. Consider any string $x \in \{0,1\}^*$, $n = |x|$. Let $\ell = t_D(n)$ be the length of the random tape of M. Let $p_x = \Pr[r \leftarrow \{0,1\}^\ell, x' = M(1^n, r) : x' = x]$ denote the probability mass of x in D_n. Our goal is to show that there exists a polynomial δ such that $p_x \leq \delta(n)2^{-pK^t(x)}$ holds for all x.

Let $S = \{r \in \{0,1\}^\ell : M(1^n, r) = x\}$ be the set of random tapes on which M will output x. (Note that $|S| = 2^\ell p_x$.) Let $m = \lceil \log |S| \rceil - 5$. Let $\mathcal{H}_m^\ell$ be the universal hash family defined in Proposition A4.

For any hash function $h_\sigma \in \mathcal{H}_m^\ell$, we refer to a hash function h_σ as being *good* if (1) $\exists s \in S$, $h_\sigma(s) = 0^m$ and (2) $|h^{-1}(0^m)| \leq 2 \cdot 2^{\ell-m}$. We first claim that with high probability over $h_\sigma \leftarrow \mathcal{H}_m^\ell$, h_σ will be good.

Claim 5. *h_σ is good with probability at least $1/2$ over $h_\sigma \leftarrow \mathcal{H}_m^\ell$.*

Proof. By Proposition A3 and a Union Bound, a random h_σ is good with probability at least $1 - 2^{-\log |S|+m+3} - 2^{-\ell+m-3} \geq \frac{1}{2}$.

We next claim that given a good hash function h_σ, there exists a short program of size roughly $\log |S|$ that produce the string x.

Claim 6. *For any good hash function $h_\sigma \in \mathcal{H}_m^\ell$, there exists a program Π of length at most*

$$O(\log \ell) + \lceil \log 1/p_x \rceil$$

that, given h_σ as input, outputs the string x within time $O(\ell^3)$.

Proof. Since h_σ is good, and let s be an string $\in S$ such that $h_\sigma(s) = 0^m$. Note that if s can be produced using a short program, x can be generated by running $M(1^n, s)$, which adds $|M| = c$ bits to the description and can be done in time $t_D(n)$.

Finally, we show how to produce s using linear algebra. Recall that the hash function $h_\sigma(x)$ is defined to be $Ax + b$ where $\sigma = (A, b)$, A, b are a binary matrix and a binary vector. We can use the Gaussian Elimination algorithm to find an vector $v \in \{0,1\}^\ell$ such that $Av + b = 0^m$ and a basis $(b_1, \ldots, b_d)$ for the kernel of A. Note that each $y \in h^{-1}(0^m)$ can be represented by a d-bit

coordinate vector (under the basis $(b_1, \ldots, b_d)$ and with respect to the offset vector v). So $d \leq \ell - m + 1$ and s can also be represented a coordinate vector of $\ell - m + 1$ bits (and let e denote this vector). We then use this fact to construct a program Π with length $\leq 4 \log \ell + \ell - m + O(1)$ bits to produce the string x. Π has the integers n, ℓ, the coordinate vector, and the code of M hardcoded ($\leq 4 \log \ell + \ell - m + 1 + O(1)$ bits). On input a hash function description σ, it computes v and $(b_1, \ldots, b_d)$ using Gaussian Elimination and Gram Schmidt, and computes $s = \sum_{i \in [d]} b_i \cdot e_i + v$. Finally, Π outputs $M_1(1^n, s)$. Notice that Π runs in time $O(\ell^3) + t_D(n) = O(t_D(n)^3) \leq t(n)$. Also notice that Π can be described by $4 \log \ell + \ell - m + 1 + O(1)$, and we fix c to be the constant such that Π can be described using $4 \log \ell + \ell - m + c$ bits.

Finally, we are ready to show that $p_x \leq \delta(n) 2^{-pK^t(x)}$. Towards this, we will prove that

$$pK^t(x) \leq O(\log \ell) + \lceil \log 1/p_x \rceil$$

and the aforementioned inequality will follow if we set $\delta(n) = \ell^{O(1)} = t_D(n)^{O(1)}$ to be a large polynomial. Consider any random string $r \in \{0,1\}^{2n(\ell m + m)}$, and we view r as $r = \sigma_1 || \sigma_2 || \ldots || \sigma_{2n}$ where each σ_i is a description of a random hash function $h_{\sigma_i} \leftarrow \mathcal{H}^\ell_m$. By Claim 5, with probability at least $1 - 2^{-2n} \geq 2/3$, there exists $i \in [2n]$ such that h_{σ_i} is a good hash function. By Claim 6, there exists a program Π' that on input h_{σ_i}, outputs the string x. Thus, let Π be a program with the index i and Π' hardcoded, and Π on input r simply outputs $\Pi'(h_{\sigma_i})$. Note that Π can be implemented using $O(\log \ell) + \lceil \log 1/p_x \rceil$ bits, and it terminates within time $O(\ell^3)$. By picking $\gamma(n) = O(n^3)$, it follows that Π runs in $O(\ell^3) \leq \gamma(\ell) \leq \gamma(t_D(n)) \leq t(n)$.

We below prove Lemma 48. We first recall the statement.

Lemma A5 (Lemma 48, restated). *Assume that* $\mathsf{E} \not\subseteq \mathsf{ioNSIZE}[2^{\Omega(n)}]$. *There exists a polynomial* γ' *such that for all polynomial* t', *any* t_D*-time samplable ensemble* $\mathcal{D}$ *is polynomially bounded by* $K^{t'}$ *if* $t'(n) \geq \gamma'(t_D(n))$.

Proof. The idea behind our proof is to derandomize the hash function used in Lemma A1. Therefore, this proof will rely on the proof of Lemma A1 heavily and we assume familiarity of Lemma A1.

Let $t, t_D, \mathcal{D}, M, x, n, \ell, p_x, S, m, \mathcal{H}^\ell_m, h_\sigma$ be as in Lemma A1. The proof of Lemma A1 shows that (1) with probability at least 0.5, a random hash function is "good" (as defined in Lemma A1) and (2) if a hash function is good, there exists a small program Π that produces x on input the hash function within time $\mathsf{poly}(\ell)$. Note that for the purpose of derandomization, the probability 0.5 is good enough for us and we don't need the parallel repetition performed in the end of Lemma A1.

Towards derandomizing h_σ, we first show that whether h_σ is good can be verified by a non-deterministic circuit. Consider a non-deterministic circuit N_x with the string x hardcoded in it. Recall that a hash function h_σ is good if (1) $\exists s \in S$ such that $h_\sigma(s) = 0^m$ and (2) $|h^{-1}(0^m)| \leq 2 \cdot 2^{\ell - m}$. (1) can be checked by guessing a string $w \in \{0,1\}^\ell$, verifying if $h_\sigma(w) = 0^m$ and if $M(1^n, w)$

outputs x. (2) can be checked deterministically. Recall that h_σ is a linear hash function defined to be $h_\sigma(x) = Ax + b$ where $\sigma = (A, b)$. By facts in linear algebra, $|h_\sigma^{-1}(0^m)| = 2^d$ where d is the dimension of the kernel for A. And d can be computed using Gram Schmidt. Therefore, we can implement a non-deterministic circuit N_x such that $N_x(\sigma) = 1$ iff h_σ is good (so N_x accepts with probability at least 0.5), and N_x is of size $\leq O(|\sigma|^2)$.

Shaltiel and Umas [29] showed that under the assumption that $\mathsf{E} \not\subseteq \mathsf{ioNSIZE}[2^{\Omega(n)}]$, there exists a PRG $G : \{0,1\}^{O(\log l)} \to \{0,1\}^l$ running in time $\mathsf{poly}(l)$ such that for all $l \in \mathbb{N}$, for all non-deterministic circuits C of size $\leq O(l^2)$, it holds that

$$|\Pr[C(G(\mathcal{U}_{O(\log l)})) = 1] - \Pr[C(\mathcal{U}_l) = 1]| \leq \frac{1}{6}$$

It follows that G also fools N_x. Thus, there exists a seed $z \in \{0,1\}^{O(\log |\sigma|)}$ such that $h_{G(z)}$ is a good hash function.

We are now ready to show that x has a deterministic short description. We consider a program Π with the seed z hardcoded in it. Π first compute $\sigma = G(z)$. Since h_σ is a good hash function, as shown in the proof of Lemma 46, there exists a program Π' of length at most

$$O(\log \ell) + \lceil \log 1/p_x \rceil$$

that produces the string x on input the hash function description σ within time $O(\ell^3)$. Π also hardcodes the program Π', and Π just runs it on σ to obtain x. Note that Π's running time is bounded by the PRG's running time ($\leq \mathsf{poly}|\sigma|$) plus the running time of Π' ($\leq O(\ell^3)$). So there exists a polynomial γ' such that Π runs in time $\gamma'(\ell) \leq \gamma'(t_D(n))$. Consider any polynomial $t'(n) \geq \gamma'(t_D(n))$. It follows that

$$K^{t'}(x) \leq |z| + O(1) + O(\log \ell) + \lceil \log 1/p_x \rceil \leq -\lceil \log 1/p_x \rceil + O(\log n)$$

which implies that there exists a polynomial δ such that for all x, $p_x \leq \delta(n) 2^{-K^{t'}(x)}$. Note that this holds for any t_D-time ensemble if $t'(n) \geq \gamma'(t_D(n))$, which concludes our proof.

References

1. Blum, M.: Coin flipping by telephone - A protocol for solving impossible problems. In: COMPCON1982, Digest of Papers, Twenty-Fourth IEEE Computer Society International Conference, San Francisco, California, USA, 22–25 February 1982, pp. 133–137. IEEE Computer Society (1982)
2. Blum, M., Micali, S.: How to generate cryptographically strong sequences of pseudo-random bits. SIAM J. Comput. **13**(4), 850–864 (1984)
3. Carter, L., Wegman, M.: Universal classes of hash functions. J. Comput. Syst. Sci. **18**(2), 143–154 (1979)
4. Chaitin, G.J.: On the simplicity and speed of programs for computing infinite sets of natural numbers. J. ACM **16**(3), 407–422 (1969)

5. Diffie, W., Hellman, M.: New directions in cryptography. IEEE Trans. Inf. Theory **22**(6), 644–654 (1976)
6. Feige, U., Shamir, A.: Witness indistinguishable and witness hiding protocols. In: STOC 1990, pp. 416–426 (1990). http://doi.acm.org/10.1145/100216.100272
7. Goldberg, H., Kabanets, V., Lu, Z., Oliveira, I.C.: Probabilistic Kolmogorov complexity with applications to average-case complexity. In: 37th Computational Complexity Conference (CCC 2022). Schloss Dagstuhl-Leibniz-Zentrum für Informatik (2022)
8. Goldreich, O., Goldwasser, S., Micali, S.: On the cryptographic applications of random functions. In: CRYPTO, pp. 276–288 (1984)
9. Goldwasser, S., Micali, S.: Probabilistic encryption. J. Comput. Syst. Sci. **28**(2), 270–299 (1984)
10. Gurevich, Y.: The challenger-solver game: variations on the theme of p=np. In: Logic in Computer Science Column, The Bulletin of EATCS (1989)
11. Hartmanis, J.: Generalized Kolmogorov complexity and the structure of feasible computations. In: 24th Annual Symposium on Foundations of Computer Science (SFCS 1983), pp. 439–445 (1983). https://doi.org/10.1109/SFCS.1983.21
12. Håstad, J., Impagliazzo, R., Levin, L.A., Luby, M.: A pseudorandom generator from any one-way function. SIAM J. Comput. **28**(4), 1364–1396 (1999)
13. Ilango, R., Ren, H., Santhanam, R.: Robustness of average-case meta-complexity via pseudorandomness. In: Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing, pp. 1575–1583 (2022)
14. Impagliazzo, R.: A personal view of average-case complexity. In: Structure in Complexity Theory 1995, pp. 134–147 (1995)
15. Impagliazzo, R., LA, L.: No better ways to generate hard np instances than picking uniformly at random. In: Proceedings [1990] 31st Annual Symposium on Foundations of Computer Science, pp. 812–821. IEEE (1990)
16. Impagliazzo, R., Luby, M.: One-way functions are essential for complexity based cryptography (extended abstract). In: 30th Annual Symposium on Foundations of Computer Science, Research Triangle Park, North Carolina, USA, 30 October - 1 November 1989, pp. 230–235 (1989)
17. Ko, K.: On the notion of infinite pseudorandom sequences. Theor. Comput. Sci. **48**(3), 9–33 (1986). https://doi.org/10.1016/0304-3975(86)90081-2
18. Kolmogorov, A.N.: Three approaches to the quantitative definition of information. Int. J. Comput. Math. **2**(1–4), 157–168 (1968)
19. Levin, L.A.: The tale of one-way functions. Prob. Inf. Trans. **39**(1), 92–103 (2003). https://doi.org/10.1023/A:1023634616182
20. Levin, L.A.: Universal search problems (Russian), translated into English by BA Trakhtenbrot in [32]. Prob. Inf. Transmission **9**(3), 265–266 (1973)
21. Liu, Y., Pass, R.: On one-way functions and Kolmogorov complexity. In: 61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, 16–19 November 2020, pp. 1243–1254. IEEE (2020)
22. Liu, Y., Pass, R.: A note on one-way functions and sparse languages. Cryptology ePrint Archive (2021)
23. Liu, Y., Pass, R.: On one-way functions from np-complete problems. In: Proceedings of the 37th Computational Complexity Conference, pp. 1–24 (2022)
24. Longpré, L., Mocas, S.: Symmetry of information and one-way functions. In: Hsu, W.-L., Lee, R.C.T. (eds.) ISA 1991. LNCS, vol. 557, pp. 308–315. Springer, Heidelberg (1991). https://doi.org/10.1007/3-540-54945-5_75

25. Lu, Z., Oliveira, I.C., Zimand, M.: Optimal coding theorems in time-bounded Kolmogorov complexity. In: 49th International Colloquium on Automata, Languages, and Programming (ICALP 2022). Schloss Dagstuhl-Leibniz-Zentrum für Informatik (2022)
26. Naor, M.: Bit commitment using pseudorandomness. J. Cryptol. **4**(2), 151–158 (1991). https://doi.org/10.1007/BF00196774
27. Rivest, R.L., Shamir, A., Adleman, L.M.: A method for obtaining digital signatures and public-key cryptosystems (reprint). Commun. ACM **26**(1), 96–99 (1983). https://doi.org/10.1145/357980.358017
28. Rompel, J.: One-way functions are necessary and sufficient for secure signatures. In: STOC, pp. 387–394 (1990)
29. Shaltiel, R., Umans, C.: Simple extractors for all min-entropies and a new pseudorandom generator. J. ACM (JACM) **52**(2), 172–216 (2005)
30. Sipser, M.: A complexity theoretic approach to randomness. In: Proceedings of the 15th Annual ACM Symposium on Theory of Computing, 25–27 April 1983, Boston, Massachusetts, USA, pp. 330–335. ACM (1983)
31. Solomonoff, R.: A formal theory of inductive inference, part I. Inf. Control **7**(1), 1–22 (1964). https://doi.org/10.1016/S0019-9958(64)90223-2
32. Trakhtenbrot, B.A.: A survey of Russian approaches to Perebor (brute-force searches) algorithms. Annal. History Comput. **6**(4), 384–400 (1984)
33. Yao, A.C.: Theory and applications of trapdoor functions (extended abstract). In: 23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3–5 November 1982, pp. 80–91 (1982)
34. Zvonkin, A.K., Levin, L.A.: The complexity of finite objects and the development of the concepts of information and randomness by means of the theory of algorithms. Russ. Math. Surv. **25**(6), 83–124 (1970). https://doi.org/10.1070/RM1970v025n06ABEH001269