DOI:10.1145/3587167

To view the accompanying Technical Perspective, visit doi.acm.org/10.1145/3587166

Toward Basing Cryptography on the Hardness of EXP

By Yanyi Liu and Rafael Pass

Abstract

Let $Kt(x)$ denote the Levin-Kolmogorov Complexity of the string x, and let MKtP denote the language of pairs (x, k) having the property that $Kt(x) \leq k$. We demonstrate that:

- **$\mathsf{MKtP} \notin \mathsf{Heur}_{\mathsf{neg}}\mathsf{BPP}$ (i.e., MKtP is *two-sided error* mildly average-case hard) iff infinitely-often OWFs exist.**
- **$\mathsf{MKtP} \notin \mathsf{Avg}_{\mathsf{neg}}\mathsf{BPP}$ (i.e., MKtP is *errorless* mildly average-case hard) iff $\mathsf{EXP} \neq \mathsf{BPP}$.**

Taken together, these results show that the only "gap" toward getting (infinitely-often) OWFs from the assumption that $\mathsf{EXP} \neq \mathsf{BPP}$ is the seemingly "minor" technical gap between two-sided error and errorless average-case hardness of the MKtP problem.

1. INTRODUCTION

A *one-way function*[6] (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 important open problem in the theory of computation, see e.g., Levin[19]): OWFs are both necessary[12] and sufficient for many of the most central cryptographic primitives and protocols (e.g., pseudorandom generators,[2, 11] pseudorandom functions,[7] private-key encryption,[8] etc.)

While many candidate constructions of OWFs are known, the question of whether OWFs can be based on some "standard" complexity-theoretic assumption is mostly wide open.

In this work, we focus on the question of whether cryptography can be based on the "super-weak" complexity-theoretic assumption that computations cannot be exponentially speedup using randomness:

> *Can the existence of OWFs be based on the assumption that* $\mathsf{EXP} \neq \mathsf{BPP}$*?*

While we are not able to provide a full positive answer to this problem (which as we shall see later on, would imply that $\mathsf{NP} \neq \mathsf{P}$), we are able to show that the task of basing OWFs on the assumption that $\mathsf{EXP} \neq \mathsf{BPP}$ is equivalent to a seemingly minor technical problem regarding different notions of average-case hardness w.r.t. Levin's notion of Kolmogorov complexity.[18] Toward explaining our main result, let us first review some recent connections between cryptography and Kolmogorov complexity.

1.1. On OWFs and Kolmogorov complexity

What makes the string 12121212121212121 less "random" than 60484850668340357492? The notion of *Kolmogorov complexity* (K-complexity), introduced by Solomonoff,[26] Kolmogorov,[16] and Chaitin,[5] 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. From a computational point of view, however, this notion is unappealing as there is no efficiency requirement on the program. The notion of *$t(\cdot)$-time-bounded Kolmogorov Complexity (K^t-complexity)* overcomes this issue: $K^t(x)$ is defined as the length of the shortest program that outputs the string x within time $t(|x|)$. As surveyed by Trakhtenbrot,[27] the problem of efficiently determining the K^t-complexity for $t(n) = \mathsf{poly}(n)$ has been studied in the Soviet Union since the 60s as a candidate for a problem that requires "brute-force search." The modern complexity-theoretic study of this problem goes back to Sipser,[24] Ko,[15] and Hartmanis[9] from the early 1980s.

A very recent result by Liu and Pass[20] shows that "mild" average-case hardness of the time-bounded Kolmogorov complexity problem (when the time bound is some polynomial) is *equivalent* to the existence of OWFs. In this work, we will extend their work to consider a different variant of the notion of "resource-bounded" Kolmogorov complexity due to Levin.[18] The central advantage of doing so will be that we will be able to base OWFs on the average-case hardness of a problem that is average-case complete for EXP! The only reason that this result falls short of basing OWF on $\mathsf{EXP} \neq \mathsf{BPP}$ is that the notion of average-case

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 all polynomials $p(\cdot)$

The original version of this paper—which contains additional results—is entitled "On the Possibility of Basing Cryptography on $\mathsf{EXP} \neq \mathsf{BPP}$" and was published in *Proceedings of the 2021 International Cryptography Conference*. Here, we focus only on providing an outline of the main results.

hardness in the EXP-completeness result is slightly different from the notion of average-case hardness for the "OWF-completeness" result. However, "morally," this result can be interpreted as an indication that the existence of OWFs is equivalent to EXP ≠ BPP (since trivially, the existence of OWFs implies that EXP ≠ BPP).

1.2. Levin-Kolmogorov complexity

While the definition of time-bounded Kolmogorov complexity, K^t, is simple and clean, as noted by Leonid Levin[18] in 1973, an annoying aspect of this notion is that it needs to be parametrized by the time-bound t. To overcome this issue, Levin proposed an elegant "non-parametrized" version of Kolmogorov complexity that directly incorporates the running time as a *cost*. To capture the idea that polynomial-time computations are "cheap," Levin's definition only charges logarithmically for running time. More precisely, let the Levin-Kolmogorov complexity of the string, $Kt(x)$, be defined as follows:

$$Kt(x) = \min_{\Pi \in \{0,1\}^*, t \in \mathbb{N}} \left\{ |\Pi| + \lceil \log t \rceil : U(\Pi, 1^t) = x \right\},$$

where U is a universal Turing machine, and we let $U(\prod, 1^t)$ denote the output of the program $\prod$ after t steps. Note that, just like the standard notion of Kolmogorov complexity, $Kt(x)$ is bounded by $|x| + O(1)$—we can simply consider a program that has the string x hard-coded and directly halts.

Let MKtP denote the decisional Levin-Kolmogorov complexity problem; namely, the language of pairs (x, k) where $k \in \{0, 1\}^{\lfloor \log |x| \rfloor}$ having the property that $Kt(x) \leq k$. MKtP no longer seems to be in NP, as there may be strings x that can be described by a short program $\prod$ (with description size, e.g., $n/10$) but with a "largish" running time (e.g., $2^{n/10}$); the resulting string x thus would have small Kt-complexity ($n/5$), yet verifying that the witness program $\prod$ indeed outputs x would require executing it which would take exponential time. In fact, Allender et al.[1] show that MKtP actually is EXP-complete w.r.t. P/poly reductions; in other words, MKtP ∈ P/poly if and only if EXP ⊆ P/poly.

We will be studying (mild) average-case hardness of the MKtP problem, and consider two standard (see e.g., Bogdanov[3]) notions of average-case tractability for a language L with respect to the uniform distribution over instances:

- **two-sided error average-case heuristics:** We say that $L \in \mathsf{Heur}_{\mathsf{neg}}\mathsf{BPP}$ if for every polynomial $p(\cdot)$, there exists some PPT heuristic $\mathcal{H}$ that decides L (w.r.t. uniform n-bit strings) with probability $1 - \frac{1}{p(n)}$.
- **errorless average-case heuristics:** We say that $L \in \mathsf{Avg}_{\mathsf{neg}}\mathsf{BPP}$ if for every polynomial $p(\cdot)$, there exists some PPT heuristic $\mathcal{H}$ such that (a) for every instance x, with probability 0.9, $\mathcal{H}(x)$ either outputs $L(x)$ or $\perp$, and (b), $\mathcal{H}(x)$ outputs $\perp$ with probability at most $\frac{1}{p(n)}$ given uniform n-bit strings x.

In other words, the difference between an errorless and a two-sided error heuristic $\mathcal{H}$ is that an errorless heuristic needs to (with probability 0.9 over its own randomness but not the instance x) output either $\perp$ (for "I don't know") or the correct answer $L(x)$, whereas a two-sided error heuristic may simply make mistakes without "knowing it".

To better understand the class $\mathsf{Avg}_{\mathsf{neg}}\mathsf{BPP}$, it may be useful to compare it to the class $\mathsf{Avg}_{\mathsf{neg}}\mathsf{P}$ (languages solvable by deterministic errorless heuristics): $L \in \mathsf{Avg}_{\mathsf{neg}}\mathsf{P}$ if for every polynomial $p(\cdot)$, there exists some deterministic polynomial-time heuristic $\mathcal{H}$ such that (a) for every input x, $\mathcal{H}(x)$ outputs either $L(x)$ or $\perp$, and (b) the probability over uniform n-bit inputs x that $\mathcal{H}$ outputs $\perp$ is bounded by $\frac{1}{p(n)}$. In other words, the *only* way an errorless heuristic may make a "mistake" is by saying $\perp$ ("I don't know"); if it ever outputs a non-$\perp$ response, this response needs to be correct. (Compare this to a two-sided error heuristic that only makes mistakes with a small probability, but we do not know when they happen.) $\mathsf{Avg}_{\mathsf{neg}}\mathsf{BPP}$ is simply the natural "BPP-analog" of $\mathsf{Avg}_{\mathsf{neg}}\mathsf{P}$ where the heuristic is allowed to be randomized.

1.3. Our results

Two-sided error average-case hardness of MKtP: Our first result shows that the characterization of Liu and Pass[20] can be extended to work also w.r.t. MKtP. More precisely,

THEOREM 1.1. $\mathsf{MKtP} \notin \mathsf{Heur}_{\mathsf{neg}}\mathsf{BPP}$ *iff infinitely-often OWFs exist.*

We highlight that whereas[20] characterized "standard" OWF, the above theorem only characterizes *infinitely-often* OWFs—that is, functions that are hard to invert for infinitely many inputs lengths (as opposed to all input lengths). The reason for this is that[20] considered an "almost-everywhere" notion of average-case hardness of K^t, whereas the statement $\mathsf{MKtP} \notin \mathsf{Heur}_{\mathsf{neg}}\mathsf{BPP}$ only considers an infinitely-often notion of average-case hardness. (As we demonstrate in the full version, we can also obtain a characterization of standard "almost-everywhere" OWFs by assuming that MKtP is "almost-everywhere" mildly average-case hard, but for simplicity, in the this paper, we focus our attention on the more standard complexity-theoretic setting of infinitely-often hardness.)

On a high level, the proof of Theorem 1.1 follows the same structure as the characterization of Liu and Pass.[20] The key obstacle to deal with is that since MKtP is not known to be in NP, there may not exist some polynomial time bound that bounds the running time of a program $\prod$ that "witnesses" the Kt-complexity of a string x; this is a serious issue as the OWF construction in Liu and Pass[20] requires knowing such a running time bound (and indeed, the running time of the OWF depends on it). To overcome this issue, we rely on a new insight about Levin-Kolmogorov complexity.

We say that the program $\prod$ is a *Kt-witness* for the string x if $\prod$ generates x within t steps while minimizing $|\prod| + \log t$ among all other programs (i.e., $\prod$ is a witness for the Kt-complexity of x). The crucial observation (see Fact 3.1) is

that for every $0 < \varepsilon < 1$, except for an ε fraction of n-bit strings x, x has a Kt-witness $\prod$ that runs in time $O(\frac{1}{\varepsilon})$. That is, "most" strings have a Kt-witness that has a "short" running time. To see this, recall that as aforesaid, for every string x, $Kt(x) \leq |x| + O(1)$; thus, every string $x \in \{0,1\}^n$ with a Kt-witness $\prod$ with running time exceeding $O(\frac{1}{\varepsilon})$, must satisfy that $|\prod| + \log O(\frac{1}{\varepsilon}) \leq kt(x) \leq n + O(1)$, so $|\prod| \leq n + O(1) - \log(\frac{O(1)}{\varepsilon}) = n + O(1) + \log \varepsilon$. Since the length of $\prod$ is bounded by $n + O(1) + \log \varepsilon$, it follows that we can have at most $O(\varepsilon)2^n$ strings x where the Kt-witness for x has a "long" running time.

We can next use this observation to consider a more computationally tractable version of Kt-complexity where we cut off the machine's running time after $\frac{1}{\varepsilon}$ steps (where ε is selected as an appropriate polynomial) and next follow a similar paradigm as in Liu and Pass[20]

Errorless average-case hardness of MKtP. We next show how to extend the result of Allender et al.[1] to show that MKtP is not just EXP complete in the worst case, but also EXP-average-case complete; furthermore, we are able to show completeness w.r.t. BPP (as opposed to P/poly) reductions. We highlight, however, that completeness is shown in a "non-black box" way (whereas[1] presented a P/poly truth table reduction). By non-black box, we here mean that we are not able to show how to use any algorithm that solves MKtP (on average) as an oracle (i.e., as a black box) to decide EXP (in probabilistic polynomial time); rather, we directly show that if MKtP $\in$ Avg$_{neg}$BPP, then EXP $\subseteq$ BPP.

THEOREM 1.2. MKtP $\notin$ Avg$_{neg}$BPP *iff* EXP $\neq$ BPP.

Theorem 1.2 follows a similar structure as the EXP-completeness results of Allender et al.[1] Roughly speaking, Allender et al. observe that by the result of Nisan and Wigderson[21] and Impagliazzo and Wigderson,[13] the assumption that EXP $\not\subseteq$ P/poly implies the existence of a (subexponential-time computable) pseudorandom generator that fools polynomial-size circuits. But using a Kt-oracle, it is easy to break the PRG (as outputs of the PRG have small Kt-complexity since its running time is "small"). We first observe that the same approach can be extended to show that MKtP is (errorless) average-case hard w.r.t. polynomial-size circuits (under the assumption that EXP $\not\subseteq$ P/poly). We next show that if we instead rely on a PRG construction of Impagliazzo and Wigderson,[14] it suffices to rely on the assumption that EXP $\neq$ BPP to show average-case hardness of MKtP w.r.t. PPT algorithms.

Interpreting Thm 1.1 and Thm 1.2. By combining Theorem 1.1 and Theorem 1.2, we get that the only "gap" toward getting (infinitely-often) one-way functions from the assumption that EXP $\neq$ BPP is the seemingly "minor" technical gap between two-sided error and errorless average-case hardness of the MKtP problem (i.e., proving MKtP $\notin$ Avg$_{neg}$BPP $\Rightarrow$ MKtP $\notin$ Heur$_{neg}$BPP). Furthermore, note that this "gap" *fully characterizes* the possibility of basing (infinitely-often) OWFs on the assumption that EXP $\neq$ BPP: Any proof that EXP $\neq$ BPP implies infinitely-often OWFs also shows the implication MKtP $\notin$ Avg$_{neg}$BPP $\Rightarrow$ MKtP $\notin$ Heur$_{neg}$BPP.

As a corollary of Theorem 1.1 and Theorem 1.2, we next demonstrate that the implication MKtP $\notin$ Avg$_{neg}$BPP $\Rightarrow$ MKtP $\notin$ Heur$_{neg}$BPP implies that NP $\neq$ P.

THEOREM 1.3. *If* MKtP $\notin$ Avg$_{neg}$BPP $\Rightarrow$ MKtP $\notin$ Heur$_{neg}$BPP, *then* NP $\neq$ P.

This result can be interpreted in two ways. The pessimistic way is that closing this gap between two-sided error, and errorless, heuristics will be very hard. The optimistic way, however, is to view it as a new and algorithmic approach toward proving that NP $\neq$ P: To demonstrate that NP $\neq$ P, it suffices to demonstrate that MKtP can be solved by an errorless heuristic, given access to a two-sided error heuristic for the same problem.

Concurrent work. A concurrent and independent work by Ren and Santhanam[23] presents related but orthogonal characterizations of MKtP. Both works essentially show an equivalence between mild average-case hardness of MKtP and the existence of OWFs; we next show that errorless average-case hardness of MKtP is equivalent to EXP $\neq$ BPP, whereas they instead consider an incomparable notion of two-sided error hardness with a "tiny" error and show that such average-case hardness of MKtP w.r.t. non-uniform polynomial-time adversaries is equivalent to the assumption that EXP $\notin$ P/poly.

2. PRELIMINARIES

We assume familiarity with basic concepts and computational classes such as Turing machines, polynomial-time algorithms, probabilistic polynomial-time (PPT) algorithms, NP, EXP, BPP, and P/poly. A function μ is said to be *negligible* if for every polynomial $p(\cdot)$ there exists some n_0 such that for all $n > n_0$, $\mu(n) \leq \frac{1}{p(n)}$. A *probability ensemble* is a sequence of random variables $A = \{A_n\}_{n \in \mathbb{N}}$. We let $\mathcal{U}_n$ denote the uniform distribution over $\{0,1\}^n$. Given a string x, we let $[x]_j$ denote the first j bits of x.

One-way functions. We recall the definition of one-way functions.[6] Roughly speaking, a function f is one way if it is polynomial-time computable, but hard to invert for PPT attackers. The standard cryptographic definition of a one-way function requires that for every PPT attacker A, there exists some negligible function $\mu(\cdot)$ such that A only succeeds in inverting the function with probability $\mu(n)$ for *all* input lengths n. (That is, hardness holds "almost-everywhere.") We will also consider a weaker notion of an *infinitely-often* one-way function,[22] which only requires that the success probability is bounded by $\mu(n)$ for *infinitely many* input lengths n. (That is, hardness only holds "infinitely-often," analogously to complexity-theoretic notions of hardness).

This non-black box aspect of our results stems from its use of Impagliazzo and Wigderson.[14]

Definition 2.1. *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 $n \in \mathbb{N}$,*

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

f is said to be an infinitely-often one-way function (ioOWF) *if the above condition holds for infinitely many $n \in \mathbb{N}$ (as opposed to all).*

We may also consider a weaker notion of a *weak one-way function*,[29] where we only require all PPT attackers to fail with probability noticeably bounded away from 1:

Definition 2.2. *Let $f: \{0,1\}^* \to \{0,1\}^*$ be a polynomial-time computable function. f is said to be a* 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. f is said to be an* weak infinitely-often one-way function (weak ioOWF) *if the above condition holds for infinitely many $n \in \mathbb{N}$ (as opposed to all).*

Yao's hardness amplification theorem[29] shows that any weak (io) OWF can be turned into a "strong" (io) OWF.

Levin-Kolmogorov complexity. Let U be some fixed universal Turing machine that can emulate any Turing machine M with polynomial overhead. Given a description $\prod \in \{0,1\}^*$ which encodes a pair (M, w) where M is a (single-tape) Turing machine and $w \in \{0,1\}^*$ is an input, let $U(\prod, 1^t)$ denote the output of $M(w)$, when emulated on U for t steps. Note that (by assumption that U only has polynomial overhead) $U(\prod, 1^t)$ can be computed in time $\mathsf{poly}(|\prod|, t)$. We turn to defining Levin's notion of Kolmogorov complexity[18]:

$$Kt(x) = \min_{\Pi \in \{0,1\}^*, t \in \mathbb{N}} \{|\Pi| + \lceil \log t \rceil : U(\Pi, 1^t) = x\}.$$

Let MKtP denote the decisional Levin-Kolmogorov complexity problem; namely, the language of pairs (x, k) where $k \in \{0,1\}^{\lceil \log |x| \rceil}$ having the property that $Kt(x) \leq k$. As is well known, we can always produce a string by hardwiring the string in (the tape of) a machine that does nothing and just halts, which yields the following central fact about (Levin)-Kolmogorov complexity.

Fact 2.1 ([25]). *There exists a constant c such that for every $x \in \{0,1\}^*$ it holds that $Kt(x) \leq |x| + c$.*

Average-case complexity. We will consider average-case complexity of languages L with respect to the *uniform* distribution of instances. Let $\mathsf{Heur}_{\mathsf{neg}}\mathsf{BPP}$ denote the class of languages that can be decided by PPT heuristics that only make mistakes on an inverse polynomial fraction of instances. More formally:

Definition 2.3 ($\mathsf{Heur}_{\mathsf{neg}}\mathsf{BPP}$). *For a decision problem $L \subset \{0,1\}^*$, we say that $L \in \mathsf{Heur}_{\mathsf{neg}}\mathsf{BPP}$ if for all polynomial $p(\cdot)$, there exists a probabilistic polynomial-time heuristic $\mathcal{H}$, such that for all sufficiently large n,*

$$\Pr[x \leftarrow \{0,1\}^n : \mathcal{H}(x) = L(x)] \geq 1 - \frac{1}{p(n)}.$$

We will refer to languages in $\mathsf{Heur}_{\mathsf{neg}}\mathsf{BPP}$ as languages that admit *two-sided error* heuristics. We will also consider a more restrictive type of *errorless* heuristics $\mathcal{H}$: for every instance x, with probability 0.9 (over the randomness of only $\mathcal{H}$), $\mathcal{H}(x)$ either outputs $L(x)$ or $\perp$ (for "I don't know"). More formally:

Definition 2.4 ($\mathsf{Avg}_{\mathsf{neg}}\mathsf{BPP}$). *For a decision problem $L \subset \{0,1\}^*$, we say that $L \in \mathsf{Avg}_{\mathsf{neg}}\mathsf{BPP}$ if for all polynomial $p(\cdot)$, there exists a probabilistic polynomial-time heuristic $\mathcal{H}$, such that for all sufficiently large n, for every $x \in \{0,1\}^n$,*

$$\Pr[\mathcal{H}(x) \in \{L(x), \perp\}] \geq 0.9,$$

and

$$\Pr[x \leftarrow \{0,1\}^n : \mathcal{H}(x) = \perp] \leq \frac{1}{p(n)}.$$

We will refer to languages in $\mathsf{Avg}_{\mathsf{neg}}\mathsf{BPP}$ as languages that admit *errorless* heuristics. As explained in the introduction, to better understand the class $\mathsf{Avg}_{\mathsf{neg}}\mathsf{BPP}$, it may be useful to compare it to the class $\mathsf{Avg}_{\mathsf{neg}}\mathsf{P}$ (languages solvable by *deterministic* errorless heuristics): $L \in \mathsf{Avg}_{\mathsf{neg}}\mathsf{P}$ if for every polynomial $p(\cdot)$, there exists some deterministic polynomial-time heuristic $\mathcal{H}$ such that (a) for every input x, $\mathcal{H}(x)$ outputs either $L(x)$ or $\perp$, and (b) the probability over uniform n-bit inputs x that $\mathcal{H}$ outputs $\perp$ is bounded by $\frac{1}{p(n)}$. In other words, the *only* way an errorless heuristic may make a "mistake" is by saying $\perp$ ("I don't know"), whereas for a two-sided error heuristic we do not know when mistakes happen. $\mathsf{Avg}_{\mathsf{neg}}\mathsf{BPP}$ is simply the natural "BPP-analog" of $\mathsf{Avg}_{\mathsf{neg}}\mathsf{P}$ where the heuristic is allowed to be randomized.

Computational indistinguishability. We recall the definition of (computational) indistinguishability[8] along with its infinitely-often variant.

Definition 2.5. *Two ensembles $\{A_n\}_{n \in \mathbb{N}}$ and $\{B_n\}_{n \in \mathbb{N}}$ are said to be* $\varepsilon(\cdot)$-indistinguishable, *if for every* PPT *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 $n \geq n_0$:*

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

We say that $\{A_n\}_{n \in \mathbb{N}}$ and $\{B_n\}_{n \in \mathbb{N}}$ are infintely-often

We remark that the constant 0.9 can be made arbitrarily small—any constants bounded away from $\frac{2}{3}$ works as we can amplify it using a standard Chernoff-type argument.

$\varepsilon(\cdot)$-indistinguishable (io-ε-indistinguishable) *if the above condition holds for infinitely many* $n \in \mathbb{N}$ *(as opposed to all sufficiently large ones)*.

Pseudorandom generators. We recall the standard definition of pseudorandom generators (PRGs)[2] and its infinitely-often variant.

Definition 2.6. *Let* $g : \{0,1\}^n \to \{0,1\}^{m(n)}$ *be a polynomial-time computable function. g is said to be a* $\varepsilon(\cdot)$-pseudorandom generator (ε-PRG) *if for any* PPT *algorithm* $\mathcal{A}$ *(whose running time is polynomial in the length of its first input), for all sufficiently large n,*

$$|\Pr[x \leftarrow \{0,1\}^n : \mathcal{A}(1^n, g(x)) = 1] - \Pr[y \leftarrow \{0,1\}^{m(n)} : \mathcal{A}(1^n, y) = 1]| < \varepsilon(n).$$

g is said to be an infinitely-often $\varepsilon(\cdot)$-pseudorandom generator (io-ε-PRG) *if the above condition holds for infinitely many* $n \in \mathbb{N}$ *(as opposed to all).*

Although the standard cryptographic definition of a PRG g requires that g runs in polynomial time, when used for the other purposes (e.g., for derandomizing BPP), we allow the PRG g to have an exponential running time.[28] We refer to such PRGs (resp. ioPRGs) as *inefficient* PRGs (resp. *inefficient* ioPRGs).

Conditionally entropy-preserving PRGs. Liu and Pass[20] introduced variant of a PRG referred to as an *entropy-preserving* pseudorandom generator (EP-PRG). Roughly speaking, an EP-PRG is a pseudorandom generator that expands n-bits to $n + O(\log n)$ bits, having the property that the output of the PRG is not only pseudorandom, but also preserves the entropy of the input (i.e., the seed): The Shannon entropy of the output is $n - O(\log n)$.[20] did not manage to construct an EP-PRG from OWFs, but rather constructed a relaxed form of an EP-PRG, called a *conditionally-secure* entropy-preserving PRG (condEP-PRG), which relaxes both the pseudorandomness and entropy-preserving properties of the PRG, to hold only conditioned on some event E. We will here consider also an infinitely-often variant:

Definition 2.7. *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-condEP-PRG) *if there exist a sequence of events* $= \{E_n\}_{n\in\mathbb{N}}$ *and a constant* $\propto$ *(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* $n \in \mathbb{N}$, $H(G(\mathcal{U}_n \mid E_n)) \geq n - \propto \log n$.

G is referred to as an $\mu(\cdot)$-conditionally secure entropy-preserving infinitely-often pseudorandom generator (μ-condEP-ioPRG) *if it satisfies the above definition except that we replace* $\mu(n)$*-indistinguishability with io-*$\mu(n)$*-indistinguishability.*

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 $\varepsilon < 1$. We recall that the existence of rate-1 efficient condEP-PRGs can be based on the existence of OWFs, and that the same theorem holds in the infinitely-often setting.

Theorem 2.8 (Implicit in Liu and Pass[20]). *Assume that OWFs (resp. ioOWFs) exist. Then, for every* $\gamma > 1$, *there exists a rate-1 efficient* μ*-condEP-PRG (resp.* μ*-condEP-ioPRG)* $G_\gamma : \{0,1\}^n \to \{0,1\}^{n+\gamma \log n}$, *where* $\mu = \frac{1}{n^2}$.

3. CHARACTERIZING OWFS

In this section, we prove our main characterization of OWFs through two-sided error average-case hardness of MKtP.

Theorem 3.1. MKtP $\notin$ Heur$_{\text{neg}}$BPP *iff infinitely-often OWFs exist.*

We remark that, in full version, we also characterize "standard" (as opposed to infinitely-often) OWFs through (almost-everywhere) mild average-case hardness of MKtP.

Below, we prove each direction of Theorem 3.1 separately (in Theorem 3.2 and Theorem 3.3).

OWFs from Avg-case hardness of MKtP. We first show that if weak ioOWFs do not exists, then we can compute the Kt-complexity of random strings with high probability (and thus, MKtP is in Heur$_{\text{neg}}$BPP). On a high-level, we will be using the same proof approach as in Liu and Pass[20] One immediate obstacle to relying on the proof in Liu and Pass[20] is that it relies on the fact that the program $\prod$ (which we refer to as the "witness") that certifies the time-bounded Kolmogorov complexity K^t of a string x has some *a priori polynomial* running time, namely $t(\cdot)$; this polynomial bound gets translated into the running time of the constructed OWF. Unfortunately, this fact no longer holds when it comes to Kt-complexity: We say that the program $\prod$ is a Kt-*witness* for the string x if $\prod$ generates x within t steps while minimizing $|\prod| + \log t$ among all other programs (i.e., $\prod$ is a witness for the Kt-complexity of x). Note that given a Kt-witness of a string x, there is no *a priori* polynomial time bound on the running time of $\prod$, since only the *logarithm* of the running time gets included in the complexity measure. For instance, it could be that the Kt-witness is a program $\prod$ of length $n/10$ that requires running time $2^{n/10}$, for a total Kt-complexity of $n/5$. Nevertheless, the crucial observation we make is that *for most strings x*, the running time of the Kt-witness actually is small: For every $0 < \varepsilon < 1$, except for an ε fraction of n-bit strings x, x has a Kt-witness $\prod$ that runs in time $O(\frac{1}{\varepsilon})$. More formally:

Fact 3.1. *For all* $n \in \mathbb{N}$, $0 < \varepsilon < 1$, *there exists* $1 - \varepsilon$ *fraction of strings* $x \in \{0,1\}^n$ *such that there exist a Turing machine* $\prod_x$ *and a running time parameter* t_x *satisfying* $U(\prod_x, 1^{t_x}) = x$, $|\prod_x| + \lceil \log t_x \rceil = Kt(x)$, *and* $t_x \leq 2^c/\varepsilon$ *(where c is as in Fact 2.1).*

Proof: Consider some $n \in \mathbb{N}$, $0 < \varepsilon < 1$, and some set $S \subset \{0,1\}^n$ such that $|S| > \varepsilon 2^n$. For any string $x \in \{0,1\}^n$, let $(\prod_x, t_x)$ be

a pair of strings such that $U(\prod_x, 1^{t_x}) = x$ and $|\prod_x| + \lceil \log t_x \rceil = Kt(x)$; that is, $(\prod_x, t_x)$ is the optimal compression for x. Note that for any $x \in \{0,1\}^n$, such $(\prod_x, t_x)$ always exists due to Fact 2.1. Let c be the constant from Fact 2.1.

We assume for contradiction that for any $x \in S$, $t_x > 2^c/\varepsilon$. Note that by Fact 2.1, it holds that $Kt(x) \le |x| + c$. Thus, $|\prod_x| = Kt(x) - \lceil \log t_x \rceil \le n + c - \lceil \log 2^c/\varepsilon \rceil \le n - \log 1/\varepsilon$. Consider the set $Z = \{\prod_x : x \in S\}$ of all (descriptions of) Turing machines $\prod_x$. Since $|\prod_x| \le n - \log 1/\varepsilon$, it follows that $|Z| \le 2^{n-\log 1/\varepsilon} = \varepsilon 2^n$. However, for each machine $\prod$ in Z, it could produce only a single string in S. So $|Z| \ge |S| > \varepsilon 2^n$, which is a contradiction. ■

We now show how to adapt the proof in Liu and Pass[20] by relying on the above fact.

THEOREM 3.2. *If* $\mathsf{MKtP} \notin \mathsf{Heur}_{\mathsf{neg}}\mathsf{BPP}$, *then there exists a weak ioOWF (and thus also an ioOWF).*

Proof: We start with the assumption that $\mathsf{MKtP} \notin \mathsf{Heur}_{\mathsf{neg}}\mathsf{BPP}$; that is, there exists a polynomial $p(\cdot)$ such that for all PPT heuristics $\mathcal{H}'$ and infinitely many n,

$$\Pr\left[x \leftarrow \{0,1\}^n, k \leftarrow \{0,1\}^{\lceil \log n \rceil} : \mathcal{H}'(x,k) = \mathsf{MKtP}(x,k)\right] < 1 - \frac{1}{p(n)}.$$

Let c be the constant from Fact 2.1. Consider the function $f: \{0,1\}^{n+c+\lceil \log(n+c) \rceil} \to \{0,1\}^*$, which given an input $\ell || \prod'$ where $|\ell| = \lceil \log(n+c) \rceil$ and $|\prod'| = n + c$, outputs $\ell + \lceil \log t \rceil \,||U(\prod, 1^t)$ where $\prod$ is the ℓ-bit prefix of $\prod'$, t is the (smallest) integer $\le 2^{c+2}p(n)$ such that $\prod$ (when interpreted as a Turing machine) halts in step t. (If $\prod$ does not halt in $2^{c+2}p(n)$ steps, f picks $t = 2^{c+2}p(n)$.) That is,

$$f(\ell || \prod') = \ell + \lceil \log t \rceil || U(\prod, 1^t).$$

Observe that f 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 $m = n + c + \lceil \log(n+c) \rceil$ for some n and outputs $f(x)$.

We now show that f is a $\frac{1}{q(\cdot)}$-weak ioOWF where $q(n) = 2^{2c+4}np(n)^2$, which concludes the proof of the theorem. Assume for contradiction that f is not a $\frac{1}{q(\cdot)}$-weak ioOWF; that is, there exists some PPT attacker $\mathcal{A}$ that inverts f with probability at least $1 - \frac{1}{q(n)} \le 1 - \frac{1}{q(m)}$ for all sufficiently large input lengths $m = n + c + \lceil \log(n+c) \rceil$. We first claim that we can use $\mathcal{A}$ to construct a PPT heuristic $\mathcal{H}^*$ such that

$$\Pr[x \leftarrow \{0,1\}^n : \mathcal{H}^*(x) = Kt(x)] \ge 1 - \frac{1}{p(n)}.$$

If this is true, consider the heuristic $\mathcal{H}$ which given a string $x \in \{0,1\}^n$ and a size parameter $k \in \{0,1\}^{\lceil \log n \rceil}$, outputs 1 if $\mathcal{H}^*(x) \le k$, and outputs 0 otherwise. Note that if $\mathcal{H}^*$ succeeds on some string x, $\mathcal{H}$ will also succeed. Thus,

$$\Pr\left[x \leftarrow \{0,1\}^n, k \leftarrow \{0,1\}^{\lceil \log n \rceil} : \mathcal{H}(x,k) = \mathsf{MKtP}(x,k)\right] \ge 1 - \frac{1}{p(n)},$$

which is a contradiction.

It remains to construct the heuristic $\mathcal{H}^*$ that computes $Kt(x)$ with high probability over random inputs $x \in \{0,1\}^n$, using $\mathcal{A}$. By an averaging argument, except for a fraction $\frac{1}{2p(n)}$ of random tapes r for $\mathcal{A}$, the *deterministic* machine $\mathcal{A}_r$ (i.e., machine $\mathcal{A}$ with randomness fixed to r) fails to invert f with probability at most $\frac{2p(n)}{q(n)}$. Consider some such "good" randomness r for which $\mathcal{A}_r$ succeeds to invert f with probability $1 - \frac{2p(n)}{q(n)}$.

On input $x \in \{0,1\}^n$, our heuristic $\mathcal{H}_r^*$ runs $\mathcal{A}_r(i||x)$ for all $i \in [n+c]$ where i is represented as a $\lceil \log(n+c) \rceil$-bit string and outputs the smallest i where the inversion on $(i||x)$ succeeds. Let $\varepsilon = \frac{1}{4p(n)}$, and S be the set of strings $x \in \{0,1\}^n$ for which $\mathcal{H}_r^*(x)$ fails to compute $Kt(x)$ and x satisfies the requirements in Fact 3.1. Note that the probability that a random $x \in \{0,1\}^n$ does not satisfy the requirements in Fact 3.1 is at most ε. Thus, $\mathcal{H}_r^*$ fails with probability at most (by a union bound)

$$\mathsf{fail}_r \le \varepsilon + \frac{|S|}{2^n}.$$

Consider any string $x \in S$ and let $w = Kt(x)$ be its Kt-complexity. Note that x satisfies the requirements in Fact 3.1; that is, there exist a Turing machine $\prod_x$ and a running time parameter t_x such that $U(\prod_x, 1^{t_x}) = x$, $|\prod_x| + \lceil \log t_x \rceil = Kt(x)$, and $t_x \le 2^c/\varepsilon = 2^{c+2}p(n)$. By Fact 2.1, we have that $|\prod_x| \le w \le n + c$. Thus, for all strings $(\ell || \prod') \in \{0,1\}^{n+c+\lceil \log(n+c) \rceil}$ such that $\ell = |\prod_x|$, $[\prod']_{|\ell|} = \prod_x$, it holds that $f(\ell || \prod') = (w||x)$. Since $\mathcal{H}_r^*(x)$ fails to compute $Kt(x)$, $\mathcal{A}_r$ must fail to invert $(w||x)$. But, since $|\prod_x| \le n + c$, the output $(w||x)$ is sampled with probability at least

$$\frac{1}{n+c} \cdot \frac{1}{2^{|\prod_x|}} \ge \frac{1}{n+c} \frac{1}{2^{n+c}} \ge \frac{1}{n2^{2c+1}} \cdot \frac{1}{2^n}$$

in the one-way function experiment, so $\mathcal{A}_r$ must fail with probability at least

$$|S| \cdot \frac{1}{n2^{2c+1}} \cdot \frac{1}{2^n} = \frac{1}{n2^{2c+1}} \cdot \frac{|S|}{2^n} \ge \frac{\mathsf{fail}_r - \varepsilon}{n2^{2c+1}}$$

which by assumption (that $\mathcal{A}_r$ is a good inverter) is at most that $\frac{2p(n)}{q(n)}$. We thus conclude that

$$\mathsf{fail}_r \le \frac{2^{2c+2}np(n)}{q(n)} + \varepsilon$$

Finally, by a union bound, we have that $\mathcal{H}^*$ (using a uniform random tape r) fails in computing Kt with probability at most

We note that the choice of $(\prod_x, t_x)$ for some x is not unique. Our argument holds if any such $(\prod_x, t_x)$ is chosen.

$$\frac{1}{2p(n)}+\frac{2^{2c+2}np(n)}{q(n)}+\varepsilon=\frac{1}{p(n)}$$

Thus, $\mathcal{H}^*$ computes Kt with probability $1-\frac{1}{p(n)}$ for all sufficiently large $n\in\mathbb{N}$, which is a contradiction. ■

Avg-case Hardness of MKtP from OWFs. We additionally show the converse of Theorem 3.2:

THEOREM 3.3. *If ioOWFs exist, then* $\mathsf{MKtP}\notin\mathsf{Heur}_{\mathsf{neg}}\mathsf{BPP}$.

Theorem 3.3 follows immediately from Theorem 2.8 and the following theorem:

THEOREM 3.4. *Assume that for some* $\gamma\geq 4$, *there exists a rate-1 efficient* μ*-condEP-ioPRG* $G:\{0,1\}^n\to\{0,1\}^{n+\gamma\log n}$ *where* $\mu(n)=1/n^2$. *Then,* $\mathsf{MKtP}\notin\mathsf{Heur}_{\mathsf{neg}}\mathsf{BPP}$.

The proof of Theorem 3.4 closely follows the proof in Liu and Pass[20] and relies only on relatively minor modifications to observe that the properties used of the time-bounded Kolmogorov complexity function K^t actually also hold for Kt—namely that random strings have "high" Kt-complexity, whereas outputs of a PRG have "low" Kt-complexity. We refer the reader to the full version for the actual proof.

4. CHARACTERIZING EXP

In this section, we will prove the following theorem:

THEOREM 4.1. $\mathsf{EXP}\neq\mathsf{BPP}$ *if and only if* $\mathsf{MKtP}\notin\mathsf{Avg}_{\mathsf{neg}}\mathsf{BPP}$.

Roughly speaking, the above theorem is proved in two steps:

- We first observe that, assuming $\mathsf{EXP}\neq\mathsf{BPP}$, there exists an (inefficient, infinitely-often) pseudorandom generator[14] that maps a n^ε-bit seed to a n-bit string in time $O(2^{n^\gamma})$ (for some $0<\varepsilon,\gamma<1$).
- We will next show that an errorless heuristic for MKtP can be used to break such PRGs (since the Kt-complexity of the output of the PRG is at most $n^\varepsilon+n^\gamma+O(1)\leq n-1$), which is a contradiction and concludes the proof.

Recall that Impagliazzo and Wigderson[14] showed that BPP can be derandomized (on average) in subexponential time by assuming $\mathsf{EXP}\neq\mathsf{BPP}$. The central technical contribution in their work can be stated as proving the existence of an inefficient PRG assuming $\mathsf{EXP}\neq\mathsf{BPP}$:

THEOREM 4.2 ([14], [28], THEOREM 3.9]). *Assume that* $\mathsf{EXP}\neq\mathsf{BPP}$. *Then, for all* $\varepsilon>0$, *there exists an inefficient io-*$\frac{1}{10}$*-PRG* $G:\{0,1\}^{n^\varepsilon}\to\{0,1\}^n$ *that runs in time* $2^{O(n^\varepsilon)}$.

We note that the proof in Impagliazzo and Wigderson[14] is non-black box. In particular, it does not show how to solve EXP in probabilistic polynomial-time having black box access to an attacker that breaks the PRG.

It remains to show that if there exists an (inefficient) io-PRG $G:\{0,1)^{n^\varepsilon}\to$ ® $\{0,1)^n$ with running time $O(2^{n^\gamma})$ (for some $0<\varepsilon,\gamma<1$), then $\mathsf{MKtP}\notin\mathsf{Avg}_{\mathsf{neg}}\mathsf{BPP}$. We recall that a string's Kt-complexity is the minimal sum of (1) the description length of a Turing machine that prints the string and (2) the logarithm of its running time. Note that the output of G could be printed by a machine with the code of G (of constant length) and the seed (of length n^ε) hardwired in it within $O(2^{n^\gamma})$ time. Thus, strings output by G have Kt-complexity less than or equal to $O(1)+n^\varepsilon+n^\gamma\leq n-1$. On the other hand, random strings have high Kt-complexity (e.g., $>n-1$) with high probability (e.g., $\geq\frac{1}{2}$). It follows that an errorless heuristic for MKtP can be used to break G. Let us highlight why it is important that we have an *errorless* heuristic (as opposed to a two-sided error heuristic): while a two-sided error heuristic would still work well on random strings, we do not have any guarantees on its success probability given pseudorandom strings (as they are sparse); an errorless heuristics, however, will either correctly decide those strings, or output $\perp$ (in which case, we can also guess that the string is pseudorandom).

We proceed to a formal statement of the theorem, and its proof.

THEOREM 4.3. *Assume that there exist constants* $0<\varepsilon,\gamma<1$ *and an inefficient io-*$\frac{1}{10}$*-PRG* $G:\{0,1\}^{n^\varepsilon}\to\{0,1\}^n$ *with running time* $O(2^{n^\gamma})$. *Then,* $\mathsf{MKtP}\notin\mathsf{Avg}_{\mathsf{neg}}\mathsf{BPP}$.

Proof: We assume for contradiction that $\mathsf{MKtP}\in\mathsf{Avg}_{\mathsf{neg}}\mathsf{BPP}$, which in turn implies that there exists an errorless PPT heuristic $\mathcal{H}$ such that for all sufficiently large n, every $x\in\{0,1\}^n$ and $k\in\{0,1\}^{\lceil\log n\rceil}$,

$$\Pr[\mathcal{H}(x,k)\in\{\mathsf{MKtP}(x,k),\perp\}]\geq 0.9, \tag{1}$$

and

$$\Pr[x\leftarrow\{0,1\}^n,k\leftarrow\{0,1\}^{\lceil\log n\rceil}:\mathcal{H}(x,k)=\perp]\leq\frac{1}{2n^2}\cdot$$

Fix some sufficiently large n, and let $k=n-1$. It follows by an averaging argument that

$$\Pr[x\leftarrow\{0,1\}^n:\mathcal{H}(x,n-1)=\perp]\leq\frac{1}{2n^2}\cdot 2^{\lceil\log n\rceil}\leq\frac{1}{n}. \tag{2}$$

We next show that we can use $\mathcal{H}$ to break the PRG G. On input $x\in\{0,1\}^n$, our distinguisher $\mathcal{A}(1^{n^\varepsilon},x)$ outputs 1 if $\mathcal{H}(x,n-1)=1$ or $\mathcal{H}(x,n-1)=\perp$. $\mathcal{A}$ outputs 0 if and only if $\mathcal{H}(x,n-1)=0$. The following two claims conclude that $\mathcal{A}$ distinguishes $\mathcal{U}_n$ and $G(\mathcal{U}_{n^\varepsilon})$ with probability at least 0.2.

CLAIM 1. $\mathcal{A}(1^{n^\varepsilon},\mathcal{U}_n)$ *will output 0 with probability at least* $0.4-\frac{1}{n}$.

Proof: Note that the probability that a random string $x\in\{0,1\}^n$ is of Kt-complexity at most $n-1$ is at most $\frac{2^{n-1}}{2^n}=\frac{1}{2}$ (since the total number of machines with description length $\leq n-1$ is 2^{n-1}). And the probability that $\mathcal{H}(x,n-1)$ outputs $\perp$ is at most $\frac{1}{n}$ (over random $x\in\{0,1\}^n$) by Equation 2. In addition, the probability that $\mathcal{H}(x,n-1)$ fails to output either $\mathsf{MKtP}(x,n-1)$ or $\perp$ is at most 0.1 by Equation 1. Thus, by a union bound,

$$\begin{aligned}
&\Pr[\,\mathcal{A}(1^{n^\varepsilon}, \mathcal{U}_n) = 0\,] \\
&\geq 1 - \Pr[Kt(\mathcal{U}_n) \leq n-1] - \Pr[\mathcal{H}(\mathcal{U}_n, n-1) = \perp] \\
&\quad - \Pr\left[\mathcal{H}(\mathcal{U}_n, n-1) \text{ fails}\right] \\
&\geq 1 - \frac{1}{2} - \frac{1}{n} - 0.1 \\
&= 0.4 - \frac{1}{n}.
\end{aligned}$$

■

Claim 2. $\mathcal{A}(1^{n^\varepsilon}, G(\mathcal{U}_{n^\varepsilon}))$ *will output 0 with probability at most* 0.1.

Proof: We first show that for all $z \in \{0,1\}^{n^\varepsilon}$, $Kt(G(z)) \leq n^\varepsilon + n^\gamma + O(1) \leq n-1$. Note that the string $G(z)$ could be produced by a machine with the code of G (of length $O(1)$) and the seed z (of length n^ε) in time $O(2^{n^\gamma})$ (which adds $\log O(2^{n^\gamma}) = n^\gamma + O(1)$ to its Kt-complexity). In addition, recall that $\mathcal{H}$ is a probabilistic errorless heuristics. Thus, $\mathcal{H}(G(z), n-1)$ will output 0 with probability at most 0.1 (by Equation 1), and the claim follows. ■

This conclude the proof of Theorem 4.3. ■

We are now ready to conclude the proof of Theorem 4.1.

Proof: [of Theorem 4.1] We show each direction separately:

- To show that EXP ≠ BPP ⇒ MKtP ∉ Avg_{neg}BPP, assume that EXP ≠ BPP and let $\varepsilon = \frac{1}{3}$, and $\gamma = \frac{1}{2}$. By Theorem 4.2, there exists an $io\text{-}\frac{1}{10}\text{-}PRG\ G: \{0,1\}^{n^\varepsilon} \to \{0,1\}^n$ with running time $2^{O(n^\varepsilon)} \leq O(2^{n^\gamma})$. We conclude by Theorem 4.3 that MKtP ∉ Avg_{neg}BPP.
- To show that MKtP ∉ Avg_{neg}BPP ⇒ EXP ≠ BPP, assume that MKtP ∉ Avg_{neg}BPP; this trivially implies that MKtP ∉ BPP. We observe that MKtP ∈ EXP as by Fact 2.1, $Kt(x) \leq |x| + O(1)$ and thus the running time for a Kt-witness, $\prod$, for x is bounded by $2^{|x|+O(1)}$. Thus, EXP ⊈ BPP, which in particular means that EXP ≠ BPP.

■

5. CONCLUSIONS AND BARRIERS

Recall that in Theorem 4.1, we showed that if we assume that EXP ≠ BPP, then MKtP is hard-on-average for errorless heuristics. Furthermore, in Theorem 3.2, we showed that if MKtP is hard-on-average for two-sided error heuristics, then (infinitely-often) one-way functions exist. Combining the two theorems together, we have that the implication MKtP ∉ Avg_{neg}BPP ⇒ MKtP ∉ Heur_{neg}BPP fully characterizes when we can base the existence of (infinitely-often) one-way functions on EXP ≠ BPP. Formally,

Theorem 5.1. MKtP ∉ Avg_{neg}BPP ⇒ MKtP ∉ Heur_{neg}BPP *holds iff* EXP ≠ BPP *implies the existence of ioOWFs.*

Perhaps surprisingly, we observe that the implication by itself (without any assumptions) implies that NP ≠ P:

Theorem 5.2. *If it holds that* MKtP ∉ Avg_{neg}BPP ⇒ MKtP ∉ Heur_{neg}BPP, *then* NP ≠ P.

Proof: Assume for contradiction that MKtP ∉ Avg_{neg}BPP ⇒ MKtP ∉ Heur_{neg}BPP holds, yet NP = P. Recall that BPP ⊆ NP^{NP},[24, 17] so it follows that P = BPP, and thus by the time hierarchy Theorem,[10] EXP ≠ BPP. Then, by Theorem 4.1, MKtP ∉ Avg_{neg}BPP. It follows from our assumption that MKtP ∉ Avg_{neg}BPP ⇒ MKtP ∉ Heur_{neg}BPP and from Theorem 5.1 that ioOWFs exist, which contradicts the assumption that NP = P. ■

We remark that the above theorem could be strengthened to show even that NP is average-case hard (w.r.t. deterministic errorless heuristics), since Buhrman, Fortnow, and Pavan[4] have showed that unless this is the case, P = BPP, which suffices to complete the rest of the proof.

The pessimistic way to interpret Theorem 5.2 is that closing the gap between two-sided error, and errorless, heuristics for MKtP will be very hard as it requires proving that NP ≠ P. The optimistic way to interpret it, however, is as a new and algorithmic approach toward proving that NP ≠ P: To demonstrate that NP ≠ P, it suffices to demonstrate that MKtP can be solved by an errorless heuristic, given access to a two-sided error heuristic for the same problem. Additionally, note that approach also does not "overshoot" the NP vs. P problem by too much. In fact, any proof of the existence of infinitely often one-way functions needs to also show this implication since by Theorem 3.3, the existence of ioOWFs implies MKtP ∉ Heur_{neg}BPP, which in turn implies that the implication trivially holds.

Acknowledgments

This work is supported in part by NSF Award SATC-1704788, NSF Award RI-1703846, AFOSR Award FA9550-18-1-0267, and a JP Morgan Faculty Award. This material is based upon work supported by 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 U.S. Government or DARPA. We are very grateful to Salil Vadhan for helpful discussions about the PRG construction of Impagliazzo and Wigderson.[14] The first author also wishes to thank Hanlin Ren for helpful discussions about Levin's notion of Kolmogorov Complexity. Finally, we are grateful to the CRYPTO'21 PC for their helpful comments (Y.V. and R.P.). C

References

1. Allender, E., Buhrman, H., Koucký, M., Van Melkebeek, D., Ronneburger, D. Power from random strings. *SIAM J. Comput 35*, 6 (2006), 1467–1493.
2. Blum, M., Micali, S. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM J. Comput 13*, 4 (1984), 850–864.
3. Bogdanov, A., Trevisan, L. Average-case complexity. Manuscript, 2008. http://arxiv.org/abs/cs.CC/0606037.
4. Buhrman, H., Fortnow, L., Pavan, A. Some results on derandomization. In *Annual Symposium on Theoretical Aspects of Computer Science*. Springer, 2003, 212–222.
5. Chaitin, G.J. On the simplicity and speed of programs for computing infinite sets of natural numbers. *J. ACM 16*, 3 (1969), 407–422.
6. Diffie, W., Hellman, M. New directions in cryptography. *IEEE Trans. Inf. Theory 22*, 6 (1976), 644–654.
7. Goldreich, O., Goldwasser, S., Micali, S. On the cryptographic applications of random functions. In *CRYPTO*. 1984, 276–288.
8. Goldwasser, S., Micali, S. Probabilistic encryption. *J. Comput. Syst. Sci 28*, 2 (1984), 270–299.
9. Hartmanis, J. Generalized kolmogorov complexity and the structure of

feasible computations. In *24th Annual Symposium on Foundations of Computer Science (sfcs 1983)*. Nov 1983, 439–445.
10. Hartmanis, J., Stearns, R.E. On the computational complexity of algorithms. *Trans. Amer. Math. Soc 117*, 1965, 285–306.
11. Håstad, J., Impagliazzo, R., Levin, L.A., Luby, M. A pseudorandom generator from any one-way function. *SIAM J. Comput. 28*, 4 (1999), 364–1396.
12. 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*. 1989, 230–235.
13. Impagliazzo, R., Wigderson, A. *P = BPP* if *e* requires exponential circuits: Derandomizing the xor lemma. In *STOC '97*. 1997, 220–229.
14. Impagliazzo, R., Wigderson, A. Randomness vs. time: de-randomization under a uniform assumption. In *Proceedings 39th Annual Symposium on Foundations of Computer Science (Cat. No. 98CB36280)*. IEEE, 1998, 734–743.
15. Ko, K. On the notion of infinite pseudorandom sequences. *Theor. Comput. Sci 48*, 3 (1986), 9–33.
16. Kolmogorov, A.N. Three approaches to the quantitative definition of information. *Int. J. Comput. Math. 2* 1-4 (1968), 157–168.
17. Lautemann, C. BPP and the polynomial hierarchy. *Inf. Process. Lett 17*, 4 (1983), 215–217.
18. Levin, L.A. Universal search problems (russian), translated into English by BA Trakhtenbrot in [27]. *Prob. Inf. Transm 9*, 3 (1973), 265–266.
19. Levin, L.A. The tale of one-way functions. *Prob. Inf. Transm 39*, 1 (2003), 92–103.
20. 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, November 16-19, 2020*. IEEE, 2020, 1243–1254.
21. Nisan, N., Wigderson, A. Hardness vs randomness. *J. Comput. Syst. Sci 49*, 2 (1994), 149–167.
22. Ostrovsky, R., Wigderson, A. One-way fuctions are essential for non-trivial zero-knowledge. In *ISTCS*, 1993, 3–17.
23. Ren, H., Santhanam, R. Hardness of KT characterizes parallel cryptography. *Electron. Colloquium Comput. Complex 28*, 57 (2021).
24. 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*. ACM, 1983, 330–335.
25. Sipser, M. Introduction to the theory of computation. *ACM Sigact News 27*, 1 (1996), 27–29.
26. Solomonoff, R. A formal theory of inductive inference. Part i. *Inf. Control 7*, 1 (1964), 1–22.
27. Trakhtenbrot, B.A. A survey of Russian approaches to perebor (brute-force searches) algorithms. *Ann. Hist. Comput. 6*, 4 (1984), 384–400.
28. Trevisan, L., Vadhan, S. Pseudorandomness and average-case complexity via uniform reductions. In *Proceedings 17th IEEE Annual Conference on Computational Complexity*. IEEE Computer Society, 2002, 0129–0129.
29. 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*. 1982, 80–91.

Yanyi Liu (yl2866@cornell.edu), Cornell Tech, New York, NY, USA.

Rafael Pass (rafael@cs.cornell.edu), Cornell Tech, New York, NY, and Tel-Aviv University, Israel.