Randomness and Intractability in Kolmogorov Complexity

Igor Carboni Oliveira
Department of Computer Science, University of Oxford, UK
igor.carboni.oliveira@cs.ox.ac.uk

Abstract

We introduce *randomized* time-bounded Kolmogorov complexity (rKt), a natural extension of Levin's notion [24] of Kolmogorov complexity. A string w of low rKt complexity can be decompressed from a short representation via a time-bounded algorithm that outputs w with high probability.

This complexity measure gives rise to a decision problem over strings: MrKtP (The Minimum rKt Problem). We explore ideas from pseudorandomness to prove that MrKtP and its variants cannot be solved in randomized quasi-polynomial time. This exhibits a natural string compression problem that is provably intractable, even for randomized computations. Our techniques also imply that there is no $n^{1-\varepsilon}$-approximate algorithm for MrKtP running in randomized quasi-polynomial time.

Complementing this lower bound, we observe connections between rKt, the power of randomness in computing, and circuit complexity. In particular, we present the first hardness magnification theorem for a natural problem that is unconditionally hard against a strong model of computation.

2012 ACM Subject Classification Theory of computation

Keywords and phrases computational complexity, randomness, circuit lower bounds, Kolmogorov complexity

Digital Object Identifier 10.4230/LIPIcs.ICALP.2019.32

Category Track A: Algorithms, Complexity and Games

Acknowledgements I am grateful to Ján Pich, Eric Allender, Ryan Williams, Shuichi Hirahara, Michal Koucký, Rahul Santhanam, and Jan Krajíček for discussions. Part of this work was completed while the author was visiting the Simons Institute for the Theory of Computing. This work was supported in part by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2014)/ERC Grant Agreement no. 615075.

1 Introduction

The Kolmogorov complexity of a string w is the length of the shortest program that prints w. This concept has found connections to a variety of topics in mathematics and computer science. Notably, Kolmogorov complexity can be used to derive Gödel's incompleteness theorems (see e.g. [12, 20, 22] and references therein), and the associated *incompressibility method* has numerous applications in areas such as graph theory, combinatorics, probability, and number theory (see [25] for a comprehensive treatment of the subject).

It is well known that computing the Kolmogorov complexity of a string is undecidable. Indeed, it is easy to see that if it were computable, then it would be possible to inspect all strings of length n and print the first string z that has complexity at least n. The resulting program provides a shorter description of z, which is contradictory.

Despite its many applications, the uncomputability of Kolmogorov complexity can render it useless in situations where an upper bound on the running time of algorithms is desirable. A time-bounded variant of Kolmogorov complexity introduced by Levin [24] has been very influential in algorithms and complexity theory (see e.g. [1, 2, 13]). In Levin's definition, the complexity of a string w takes into account not only the description length of a program generating w, but also its running time. A bit more formally, we use Kt(w) to denote the minimum over $|M| + \log t$, where M is a machine that prints w when it computes for t steps. The choice of $\log t$ in this definition can be justified by its applications in theory of computation, such as Levin's optimal universal search (see [1]).

Kolmogorov complexity and Levin complexity are important measures of the "randomness", or "information", of a string. But while the computability aspects of Kolmogorov complexity are well understood, the complexity-theoretic aspects of time-bounded Kolmogorov complexity remain mysterious. It is easy to see that $\mathsf{Kt}(w)$ can be computed in exponential time $2^{O(|w|)}$. Note however that the argument presented above for the uncomputability of Kolmogorov complexity simply does not work when one takes into account running time.

Let MKtP (The Minimum Kt Problem) denote the problem of deciding the Kt complexity of an input string. The question of whether $\mathsf{MKtP} \in \mathsf{P}$ was explicitly posed in [4]. There is evidence that the problem is hard, since under standard cryptographic assumptions it follows that $\mathsf{MKtP} \notin \mathsf{P}$. The best known upper bound on the complexity of MKtP is its inclusion in $\mathsf{E} = \mathsf{DTIME}[2^{O(n)}]$. Since it is known that $\mathsf{E} \nsubseteq \mathsf{P}$ by the deterministic time hierarchy theorem, unconditionally proving that $\mathsf{MKtP} \notin \mathsf{P}$ might be within reach of existing techniques.

In this work, we investigate time-bounded Kolmogorov complexity in the presence of *randomness*. More precisely, we consider a natural extension of Kt complexity obtained when one allows the algorithm generating the string to be randomized. The only requirement is that it generates the desired string (in some fixed time bound t) with high probability. Thus we let $\mathsf{rKt}(w)$ denote the minimum over $|M| + \log t$, where M is a *probabilistic* machine that prints w with probability at least $2/3$ when it computes for t steps.

This extension of Kt complexity is motivated from several perspectives. First, it is in line with the ubiquitous role of probabilistic algorithms in modern theoretical computer science. Second, it allows many results on time-bounded Kolmogorov complexity to be extended to the randomized setting. (For instance, it is not hard to see that if $\mathsf{SAT} \in \mathsf{BPTIME}[t]$, then every satisfiable formula ϕ admits a satisfying assignment of (conditional) rKt complexity at most $O(\log t + \log |\phi|)$.[1] This allows one to define an optimal *randomized* universal search, in the spirit of Levin's result [23].) Moreover, rKt complexity can be interpreted as an extension of Kt complexity to the *pseudodeterministic* setting (see [15] and papers citing this reference), an active research direction in algorithms and complexity. Finally, by interpreting time-bounded Kolmogorov complexity as a measure of data compression, it becomes rather natural to admit representations that can be decoded via randomized algorithms. This might allow better compression rates and faster decompression procedures.[2]

Several basic questions pose themselves: What is the computational complexity of deciding $\mathsf{rKt}(w)$? Does randomization provide better compression, in the sense that $\mathsf{rKt}(w)$ might be substantially smaller than $\mathsf{Kt}(w)$ for some strings w? How does rKt and its associated decision problem relate to the complexity of deciding MKtP?

In addition to putting forward the concept of randomized time-bounded Kolmogorov complexity, our work contains the following contributions.

[1] It is possible to use the assumption to find the lexicographic first satisfying assignment of ϕ given the description of ϕ in probabilistic time $\mathsf{poly}(t, |\phi|)$.

[2] While the definition of rKt appears to be rather natural in hindsight, to our knowledge it has not been previously considered in the literature, despite the many variants of time-bounded Kolmogorov complexity investigated in other works (see e.g. [4, 5]). Intuitively, allowing randomness in the computation is somewhat counter-intuitive, given that Kolmogorov complexity tries to capture how far from random the output string is. This may explain in part why this concept had not been identified before this work. It is worth noting that our definition is influenced by the emerging area of pseudodeterministic algorithms. Indeed, rKt is a candidate definition for the "pseudodeterministic complexity" of a string. This might explain why defining rKt is more evident at this point compared to previous works.

Our Results. In order to state our results in a general form, we let rKt_λ denote the minimum over $|M| + \log t$, where M is a probabilistic machine that prints w with probability at least λ when it computes for t steps.[3] We let $\mathsf{MrKtP}[\beta, \alpha, s]$ denote the promise problem of distinguishing whether $\mathsf{rKt}_\beta(w) \leq s(|w|)$ or $\mathsf{rKt}_\alpha(w) > s(|w|)$, where $1/2 < \alpha \leq \beta < 1$ and $s \colon \mathbb{N} \to \mathbb{N}$. The problem is easier the larger the gap between α and β, but our lower bound applies to all settings of the two parameters.

It is not hard to prove that MrKtP can be solved in randomized exponential time if $\alpha < \beta$. Our main technical result is the following *unconditional* lower bound.

▶ **Theorem 1.** *Let $1/2 < \alpha \leq \beta < 1$ and $s(n) = n^\gamma$, where $0 < \gamma < 1$. Then $\mathsf{MrKtP}[\beta, \alpha, s] \notin \mathsf{Promise\text{-}BPTIME}[n^{\mathsf{poly}(\log n)}]$.*

Note that MrKtP is a total function if $\alpha = \beta$, and that the lower bound also holds in this regime. Theorem 1 presents a natural string compression problem that is provably intractable, even with randomness. While it is known that $\mathsf{BPEXP} \not\subseteq \mathsf{BPQP}$, existing proofs of this separation and its extensions only produce artificial computational problems (see e.g. [19, 7, 14, 9] for more background). The proof of Theorem 1 employs different techniques, and the argument is robust enough to establish the hardness of several variants of the problem. We will discuss one of these extensions later in this section.

The main technique used in the proof of Theorem 1 is indirect diagonalization. The argument makes use of results from the theory of pseudorandomnenss, and relies on recent insights from the investigation of pseudodeterministic algorithms [30] and connections between learning algorithms and lower bounds [29]. While pseudorandomness has been explored in the context of Kolmogorov complexity at least since the work of [4], these new perspectives were crucial in the discovery of this unconditional lower bound.

Theorem 1 can be extended to running times that are larger than quasi-polynomial, but it is unclear how to adapt the proof to show a lower bound against randomized algorithms running in time 2^{n^ε} for a small $\varepsilon > 0$. (Similarly, it is not known if $\mathsf{BPTIME}[2^n] \subseteq \mathsf{BPTIME}[2^{n^\varepsilon}]$.) A sub-exponential lower bound is open even with respect to *deterministic* algorithms. Nevertheless, we can prove a weaker lower bound in this direction that relates the deterministic complexities of MKtP and $\mathsf{MrKtP}[\beta, \alpha, s]$. For convenience, we let MrKtP denote the problem with parameters $\beta = 3/4$, $\alpha = 2/3$, and $s(n) = n/2$.

▶ **Theorem 2.** *Either $\mathsf{MKtP} \notin \mathsf{P}$ or $\mathsf{MrKtP} \notin \mathsf{Promise\text{-}EXP}$.*

Since MrKtP can be computed in $\mathsf{Promise\text{-}BPE}$, this result shows a weakness of deterministic algorithms solving these problems. The proof of Theorem 2 combines previous results on MKtP that also rely on pseudorandomness with some observations about rKt and MrKtP.

Theorems 1 and 2 indicate that these problems are good candidates for non-uniform circuit lower bounds. In order to discuss our next result, it is convenient to introduce a variant of MrKtP. For a string $w \in \{0,1\}^n$, let $\mathsf{rKt}(w) \overset{\text{def}}{=} \mathsf{rKt}_\lambda(w)$ for $\lambda = 2/3$. For functions $s_1, s_2 \colon \mathbb{N} \to \mathbb{N}$, we let $\mathsf{Gap\text{-}MrKtP}[s_1, s_2]$ be the (promise) problem of distinguishing between $\mathsf{rKt}(w) \leq s_1(n)$ versus $\mathsf{rKt}(w) > s_2(n)$. Again, it is not hard to solve this problem in randomized exponential time if there is a certain (small) gap between $s_1(n)$ and $s_2(n)$.

We obtain the following complexity results for $\mathsf{Gap\text{-}MrKtP}$.

[3] We assume for definiteness that M is a "clocked" machine that runs in time at most t on all computation paths. This is not essential, and does not significantly affect the asymptotics of rKt.

▶ **Theorem 3.** *Let $C \geq 1$ be a sufficiently large constant. The following results hold.*

(i) *For every constructive $s\colon \mathbb{N} \to \mathbb{N}$,* $\mathsf{Gap\text{-}MrKtP}[s(n), s(n) + C\log n] \in \mathsf{Promise\text{-}BPTIME}[2^{O(n)}]$.

(ii) *For every $0 < \gamma < 1$,* $\mathsf{Gap\text{-}MrKtP}[n^{\gamma}, n/2] \notin \mathsf{Promise\text{-}BPTIME}[n^{\mathsf{poly}(\log n)}]$.

(iii) *If there is $\varepsilon > 0$ such that for every $0 < \gamma < 1$,* $\mathsf{Gap\text{-}MrKtP}[n^{\gamma}, n^{\gamma} + C\log n] \notin \mathsf{SIZE}[n^{1+\varepsilon}]$*, then* $\mathsf{Promise\text{-}BPEXP} \nsubseteq \mathsf{SIZE}[\mathsf{poly}]$.

Theorem 3 (*ii*) implies a strong *inapproximability* result for computing rKt (see [16] for a recent work where inapproximability of "complexity" plays a role). On the other hand, Theorem 3 (*iii*) proves that weak non-uniform lower bounds for Gap-MrKtP can be "magnified" (cf. [31]) to super-polynomial lower bounds for a problem in Promise-BPEXP. (Such lower bounds are only known for languages in MAEXP [10], which combines *randomness* and *nondeterminism* in the exponential-time regime.)

In contrast to previous work (cf. [28] and references therein), Theorem 3 provides the first hardness magnification theorem for a natural problem that is *provably hard* against a strong model of computation (randomized polynomial-time algorithms).[4] The proof of Theorem 3 (*ii*) is similar to the proof of Theorem 1, while part (*iii*) follows by an adaptation of a version of the result established for Gap-MKtP in [28]. Note that Theorem 3 exhibits an interesting contrast between proving uniform and non-uniform lower bounds.

Finally, we consider the relation between Kt and rKt. In other words, can we have shorter descriptions if we allow randomized decoding?[5] As a concrete example, the results in [30] imply that infinitely many prime numbers (represented as binary strings) have sub-polynomial rKt complexity. This is not known to hold with respect to Kt complexity.

We employ standard techniques to establish two results that relate rKt and Kt. The first result links the deterministic complexity of MKtP to the gap between Kt and rKt.

▶ **Theorem 4.** *If* $\mathsf{MKtP} \in \mathsf{P}$ *then there is a sequence $\{w_n\}_{n\geq 1}$ with $w_n \in \{0,1\}^n$ such that* $\mathsf{rKt}(w_n) = O(\log n)$ *and* $\mathsf{Kt}(w_n) = \Omega(n)$.

On the other hand, the next theorem (roughly) shows that Kt and rKt are linearly related for every string if and only if randomized exponential time computations can be derandomized. (We refer to [2] for similar results involving other notions of time-bounded Kolmogorov complexity.)

▶ **Theorem 5.** *The following implications hold.*

(i) *If* $\mathsf{Promise\text{-}BPE} \subseteq \mathsf{Promise\text{-}E}$*, then* $\mathsf{Kt}(w) = O(\mathsf{rKt}(w))$ *for every string w.*

(ii) *If* $\mathsf{Kt}(w) = O(\mathsf{rKt}(w))$ *for every string w, then* $\mathsf{BPE} \subseteq \mathsf{E}/O(n)$.

In particular, rKt *and* Kt *are linearly related if* E *requires exponential size boolean circuits.*

This result implies that, under the standard derandomization assumption that Promise-BPE is contained in Promise-E, the problems MrKtP and MKtP essentially coincide. Therefore, our unconditional results for MrKtP and its variants provide strong evidence that MKtP is intractable.

[4] Discussions on the feasibility of previous magnification results as an approach toward new non-uniform lower bounds relied either on conjectured separations between complexity classes or on cryptographic assumptions.

[5] Note that it is possible to recover with high probability a string w from its description in time at most $2^{O(\mathsf{rKt}(w))}$. Additionally, one can *exactly* recover w (i.e. with probability 1) by cycling through all choices of the randomness and taking a majority vote.

Related Work. Pseudodeterministic algorithms and hardness magnification are active research areas. We refer to the references in [15, 28] and to papers citing these works for more details. Quantum versions of Kolmogorov complexity have been proposed in [32, 26, 35, 8]. Before this work, unconditional lower bounds were shown for a non-deterministic formulation of Kt, where it was proved that the corresponding decision problem is in P^{NE} but not in $\mathsf{NP} \cap \mathsf{coNP}$. We refer to [5] for more information. Finally, there is a huge literature on time-bounded Kolmogorov complexity and its applications in theory of computation. A recent reference such as [3] contains pointers to many other works in the area.

Concluding Remarks. We view the unconditional lower bounds in Theorem 1 and Theorem 3 (*ii*) as a step toward understanding the hardness of computing the "complexity" of strings. Such problems are important in computer science. In particular, the conjectured security of modern cryptography implies that distinguishing "structured" strings from "random" strings (according to different measures) is hard. In this work, the complexity of a string is explored from the perspective of rKt, which is likely to be essentially equivalent to Kt complexity (as suggested by Theorem 5). Previous unconditional lower bounds on the associated decision problems applied only to strong measures, such as the non-deterministic version of Kolmogorov complexity studied in [5]. Our work is the first to show an unconditional lower bound for a notion of complexity that appears to be equivalent to Levin's seminal Kt complexity. Our techniques are also robust, and lead to a hardness of approximation result. We mention that an average-case lower bound in the sense of [17] can be proved as well.

We leave open the problem of showing an exponential lower bound on the complexity of deciding rKt complexity. Theorem 3 (*iii*) and its extensions to different circuit classes also suggest that investigating non-uniform lower bounds for this problem might be a fruitful direction.

Organization. The next section formalizes some definitions and observations mentioned above, and discusses a couple of basic facts and examples related to randomized time-bounded Kolmogorov complexity. The proofs of Theorems 1 and 2 appear in Section 3. This is followed by a sketch of the proof of Theorem 3 in Section 4. Section 5 discusses Theorems 4 and 5.

2 Preliminaries

For background in (time-bounded) Kolmogorov complexity and related topics, we refer to [25]. We fix a reasonable representation of Turing machines, and let $|M|$ denote the length of the binary encoding of a machine M. Our results are not sensitive to particular encoding choices. We assume that machines have an extra tape with random bits. We let $\boldsymbol{M_{\leq t}}$ denote the random variable that represents the content of the output tape of M when it computes for (at most) t steps over the empty string.

▶ **Definition 6** (Kt_λ Complexity)**.** *For $\lambda \in [0,1]$ and $w \in \{0,1\}^*$, we let*

$$\mathsf{Kt}_\lambda(w) = \min_{M,t}\{|M| + \lceil \log t \rceil \mid \Pr[\boldsymbol{M_{\leq t}} = w] \geq \lambda\}.$$

The randomized time-bounded Kolmogorov complexity of w is given by $\mathsf{rKt}(w) \overset{\text{def}}{=} \mathsf{Kt}_{2/3}(w)$.

As a concrete example, the main result of [30] implies that for every $\varepsilon > 0$, there is a sequence $\{p_m\}_{m \geq 1}$ of increasing prime numbers such that $\mathsf{rKt}(p_m) \leq |p_m|^\varepsilon$ for every m, where $|p_m|$ denotes the length of the binary representation of p_m. For the reader familiar with the ideas from [15] and subsequent work, the randomized time-bounded Kolmogorov complexity of a string can be seen as a measure of its "pseudodeterministic" complexity.

It is easy to see that the definition of $\mathsf{rKt}(w)$ does not change substantially if we use another threshold parameter $1/2 < \lambda < 1$.[6] The (deterministic) time-bounded Kolmogorov complexity of a string w corresponds to $\mathsf{Kt}_\lambda(w)$ for $\lambda = 1$. Note that if $\alpha \leq \beta$ then $\mathsf{Kt}_\alpha(w) \leq \mathsf{Kt}_\beta(w)$.

▶ **Definition 7** ($\mathsf{MrKtP}[\beta, \alpha, s]$). *For $0 < \alpha \leq \beta \leq 1$ and $s \colon \mathbb{N} \to \mathbb{N}$, we let $\mathsf{MrKtP}[\beta, \alpha, s]$ be the promise problem $(\mathcal{YES}_n, \mathcal{NO}_n)_{n \in \mathbb{N}}$, where*

$$\begin{aligned} \mathcal{YES}_n &= \{w \in \{0,1\}^n \mid \mathsf{Kt}_\beta(w) \leq s(n)\}, \\ \mathcal{NO}_n &= \{w \in \{0,1\}^n \mid \mathsf{Kt}_\alpha(w) > s(n)\}. \end{aligned}$$

For concreteness, we let MrKtP denote $\mathsf{MrKtP}[\beta, \alpha, s]$ for $\beta = 3/4$, $\alpha = 2/3$, and $s = n/2$.

We will tacitly assume that s is constructive in all results.

▶ **Lemma 8.** *For rationals $0 < \alpha < \beta \leq 1$ and a function $s \colon \mathbb{N} \to \mathbb{N}$, $\mathsf{MrKtP}[\beta, \alpha, s] \in \mathsf{Promise\text{-}BPE}$.*

Proof Sketch. Let $\alpha < \eta < \beta$, for a fixed rational η. For all appropriate machines M and running times t, estimate with confidence at least $1 - 2^{-\omega(n)}$ the probability that M generates w when it computes for t steps. Consider M and its time bound t to be "good" if this probability estimate is at least η. Accept w if and only if a good pair has combined complexity at most s.

The correctness of the algorithm follows by a concentration bound and a standard union bound. The upper bound on its running time uses that $\mathsf{Kt}_\lambda(w)$ is at most $O(|w|)$ for every string w and $\lambda \in [0, 1]$. ◀

Note that if $\beta = \alpha$ then $\mathsf{MrKtP}[\beta, \alpha, s]$ is a total problem. However, it is unclear if the problem is in BPE for this choice of parameters.

It is also convenient to consider a close variant of MrKtP. Recall that $\mathsf{rKt}(w) = \mathsf{Kt}_{2/3}(w)$.

▶ **Definition 9** ($\mathsf{Gap\text{-}MrKtP}[s_1, s_2]$). *Let $s_1, s_2 \colon \mathbb{N} \to \mathbb{N}$, where $s_1(n) \leq s_2(n)$ for every $n \in \mathbb{N}$. We let $\mathsf{Gap\text{-}MrKtP}[s_1, s_2]$ be the promise problem $(\mathcal{YES}_n, \mathcal{NO}_n)_{n \in \mathbb{N}}$, where*

$$\begin{aligned} \mathcal{YES}_n &= \{w \in \{0,1\}^n \mid \mathsf{rKt}(w) \leq s_1(n)\}, \\ \mathcal{NO}_n &= \{w \in \{0,1\}^n \mid \mathsf{rKt}(w) > s_2(n)\}. \end{aligned}$$

▶ **Lemma 10.** *Suppose that $s_1(n) + c \log n \leq s_2(n)$, where $c \geq 1$ is a large enough constant. Then $\mathsf{Gap\text{-}MrKtP}[s_1, s_2] \in \mathsf{Promise\text{-}BPE}$.*

Proof. Given Lemma 8, it is enough to reduce $\mathsf{Gap\text{-}MrKtP}[s_1, s_2]$ to $\mathsf{MrKtP}[2/3, 3/5, s_1]$. Clearly, the set of positive instances of both problems coincide. On the other hand, it is easy to see that $\mathsf{Kt}_{2/3}(w) \leq \mathsf{Kt}_{3/5}(w) + c \log n$ if c is a sufficiently large universal constant, by amplification of the underlying randomized algorithm. As a consequence,

$$\{w \in \{0,1\}^n \mid \mathsf{rKt}(w) > s_2(n)\} \subseteq \{w \in \{0,1\}^n \mid \mathsf{Kt}_{3/5}(w) > s_1(n)\},$$

since if $\mathsf{rKt}(w) > s_2(n)$ then $\mathsf{Kt}_{3/5}(w) > s_2(n) - c \log n$, and by assumption $s_2(n) - c \log n \geq s_1(n)$. In other words, the set of negative instances of $\mathsf{Gap\text{-}MrKtP}[s_1, s_2]$ is contained in the set of negative instances of $\mathsf{MrKtP}[2/3, 3/5, s_1]$. ◀

[6] It is not hard either to prove this claim for a constant $0 < \lambda \leq 1/2$, and we leave it as an exercise. (Hint: Use a short advice string to distinguish w from any other string that is output with probability $\geq \lambda/2$.)

▶ Remark. For simplicity of the exposition, we might abuse notation in some statements and compare a promise problem with a standard complexity class. However, in all proofs the distinction between the two cases is carefully considered.

3 The computational hardness of MrKtP

3.1 MrKtP is not in BPP

The main result established in this section is the following lower bound.

▶ **Theorem 11.** *Let $1/2 < \alpha \leq \beta < 1$ and $n^{\gamma} \leq s(n) \leq n/2$ for every large enough $n \in \mathbb{N}$, where $\gamma > 0$ is fixed but arbitrary. Then $\mathsf{MrKtP}[\beta, \alpha, s] \notin \mathsf{BPTIME}[n^{\mathsf{poly}(\log n)}]$. In other words, no randomized algorithm running in quasi-polynomial time accepts with probability $\geq 2/3$ the positive instances of $\mathsf{MrKtP}[\beta, \alpha, s]$ and rejects with probability $\geq 2/3$ the negative instances of $\mathsf{MrKtP}[\beta, \alpha, s]$.*

The proof given here requires the following results, which assume parameters α, β, and s as in Theorem 11. (We refer to [4] for applications of similar techniques.)

▶ **Lemma 12.** $\mathsf{BPE} \leq_{\mathsf{tt}}^{\mathsf{P/poly}} \mathsf{MrKtP}[\beta, \alpha, s]$. *In particular, given any sequence $\{g_n\}_{n \geq 1}$ of total boolean functions $g_n \colon \{0,1\}^n \to \{0,1\}$ that compute $\mathsf{MrKtP}[\beta, \alpha, s]$, every language in BPE can be computed by (deterministic) polynomial size oracle circuits with access to $\{g_n\}_{n \geq 1}$.*

▶ **Lemma 13.** $\mathsf{PSPACE} \subseteq \mathsf{BPP}^{\mathsf{MrKtP}[\beta,\alpha,s]}$. *More precisely, given any fixed oracle $\mathcal{O} \subseteq \{0,1\}^*$ that agrees with $\mathsf{MrKtP}[\beta, \alpha, s]$ over the relevant input strings, $\mathsf{PSPACE} \subseteq \mathsf{BPP}^{\mathcal{O}}$. Furthermore, if $\mathcal{O}$ is randomized and satisfies the promise of bounded acceptance probabilities over the inputs of $\mathsf{MrKtP}[\beta, \alpha, s]$, then the corresponding algorithm in $\mathsf{BPP}^{\mathcal{O}}$ satisfies this promise over all input strings.*

We postpone the proof of these lemmas. The next lemma is well known, and can be proved by a diagonalization argument (see e.g. [29, Corollary 2]).

▶ **Lemma 14.** *Let $s_1, s_2 \colon \mathbb{N} \to \mathbb{N}$ be space-constructible functions such that $s_2(n)^2 = o(s_1(n))$, $s_2(n) = \Omega(n)$, and $s_1(n) = 2^{o(n)}$. Then there is a language in $\mathsf{DSPACE}[s_1(n)]$ that cannot be computable by circuits of size $s_2(n)$.*

We are ready to prove Theorem 11, assuming these results.

Proof of Theorem 11. Suppose toward a contradiction that $\mathsf{MrKtP}[\beta, \alpha, s]$ can be computed in $\mathsf{BPTIME}[n^{(\log n)^a}]$, for some $a > 0$. Then, by standard non-uniform derandomization, $\mathsf{MrKtP}[\beta, \alpha, s]$ can be computed by circuits of size $O(n^{(\log n)^b})$, for some $b > 0$. It follows from Lemma 12 that every language $L \in \mathsf{BPE}$ can be computed by circuits of size $O(n^{(\log n)^{c_L}})$, for some $c_L > 0$.

Let L^* be a language given by Lemma 14 for appropriate parameters $s_1(n) = 2^{n^{o(1)}}$ and $s_2(n) = n^{(\log n)^{\omega(1)}}$. In other words, $L^* \in \mathsf{DSPACE}[s_1] \setminus \mathsf{SIZE}[s_2]$. Lemma 13 and our initial assumption imply that $\mathsf{PSPACE} \subseteq \mathsf{BPTIME}[n^{\mathsf{poly}(\log n)}]$. By a standard padding argument, we get that $L^* \in \mathsf{BPE}$. But then the upper and lower bounds on the circuit complexity of L^* are in contradiction. This completes the proof of Theorem 11. ◀

We proceed with the proofs of Lemmas 12 and 13. Given a function $f \colon \{0,1\}^* \to \{0,1\}$, we consider an associated "pseudorandom" generator G_f. (Formally, the argument employs a uniform sequence of generators, one for each $n \geq 1$.) More precisely, the generator $G_f^{\mathsf{BFNW}} \colon \{0,1\}^{n^{\varepsilon}} \to \{0,1\}^n$ can be computed in deterministic time $\exp(O(n^{\varepsilon}))$ given oracle access to f on inputs of length at most n^{ε}, and satisfies the following crucial property.

▶ **Theorem 15** (see [6, 21]). *Let $f\colon \{0,1\}^* \to \{0,1\}$ be a function, $\varepsilon > 0$ be arbitrary, and G_f^{BFNW} be the associated sequence of functions mentioned above. Moreover, let $T \subseteq \{0,1\}^*$ be an arbitrary test. If*

$$\left| \Pr_{\boldsymbol{r} \in U_n}[\boldsymbol{r} \in T] - \Pr_{\boldsymbol{x} \in U_{n^\varepsilon}}[G_f^{\mathsf{BFNW}}(\boldsymbol{x}) \in T] \right| \geq 1/n$$

for every large enough n, then there is a sequence $\{C_n\}_{n \geq 1}$ of polynomial size oracle circuits with access to T that compute f on each input length n and query T nonadaptively.

Proof of Lemma 12. Let $L \in \mathsf{BPE}$, and $\{f_n\}_{n\geq 1}$ be the corresponding sequence of boolean functions that compute L. Recall the constants $1/2 < \alpha \leq \beta < 1$ and $\gamma > 0$ from the statements of Theorem 11 and Lemma 12. Take $\varepsilon \overset{\text{def}}{=} \gamma/2$, and consider the generator G_f^{BFNW} obtained from f and ε. Moreover, let $\{g_n\}_{n\geq 1}$ be a sequence of boolean functions $g_n\colon \{0,1\}^n \to \{0,1\}$ that agree with $\mathsf{MrKtP}[\beta,\alpha,s]$ over input strings in $\mathcal{YES}_n \cup \mathcal{NO}_n$. Finally, set $T \overset{\text{def}}{=} \bigcup_{n\geq 1} g_n^{-1}(0)$.

We claim that T distinguishes the output of G_f^{BFNW} from a random n-bit string. First, for each seed $w \in \{0,1\}^{n^\varepsilon}$, $G_f^{\mathsf{BFNW}}(w)$ can be computed in time at most $\exp(O(n^\varepsilon))$ given w and oracle access to $f_1, \ldots, f_{n^\varepsilon}$. Since each function f_i for $i \leq n^\varepsilon$ can be computed in randomized time $\exp(O(n^\varepsilon))$ and with error probability at most $\exp(-n)$ by a uniform algorithm, it follows that $\mathsf{Kt}_\beta(G_f^{\mathsf{BFNW}}(w)) \leq \mathsf{Kt}_{1-o(1)}(G_f^{\mathsf{BFNW}}(w)) \leq O(n^\varepsilon) < n^\gamma \leq s(n)$, for n sufficiently large. Therefore, $G_f^{\mathsf{BFNW}}(w) \notin T$ for every $w \in \{0,1\}^{n^\varepsilon}$. On the other hand, a typical random n-bit string $\boldsymbol{r} \in U_n$ has (standard) Kolmogorov complexity $K(\boldsymbol{r}) \geq (1-o(1))n$. It is easy to see that if $\lambda > 1/2$, then $K(x) \leq \mathsf{Kt}_\lambda(x)$ for a string x. As a consequence, with high probability $\mathsf{Kt}_\alpha(\boldsymbol{r}) > n/2 \geq s(n)$, in which case we have $\boldsymbol{r} \in T$.

Since T distinguishes the generator from random, it follows from Theorem 15 that L can be computed by polynomial size oracle circuits that make non-adaptive queries to T, i.e., to the functions $\{g_n\}_{n\geq 1}$. ◀

In order to prove Lemma 13, we need a *uniform* version of Theorem 15. A result of this form was established in [18], and we discuss it in more detail now. For $\varepsilon > 0$ and a function $f\colon \{0,1\}^* \to \{0,1\}$, the generator $G_f^{\mathsf{IW}}\colon \{0,1\}^{n^\varepsilon} \to \{0,1\}^n$ is also computable in deterministic time $\exp(O(n^\varepsilon))$ with oracle access to f on inputs of size at most n^ε. In addition, it satisfies the following property.

▶ **Theorem 16** (see [18]). *Let $f\colon \{0,1\}^* \to \{0,1\}$ be a function that is both random self-reducible and downward self-reducible, $\varepsilon > 0$ be arbitrary, and G_f^{IW} be the associated sequence of functions mentioned above. Moreover, let $T \subseteq \{0,1\}^*$ be an arbitrary test. If*

$$\left| \Pr_{\boldsymbol{r} \in U_n}[\boldsymbol{r} \in T] - \Pr_{\boldsymbol{x} \in U_{n^\varepsilon}}[G_f^{\mathsf{IW}}(\boldsymbol{x}) \in T] \right| \geq 1/n$$

for every large enough n, then there is a randomized polynomial-time Turing machine with oracle access to T that on every input x outputs $f(x)$ with high probability.

▶ **Theorem 17** (see [33]). *There is a language $L_{\mathsf{TV}} \in \mathsf{DSPACE}[O(n)]$ that is* PSPACE*-hard, random self-reducible, and downward self-reducible.*

We are ready to prove Lemma 13, which completes the proof of Theorem 11.

Proof of Lemma 13 (Sketch). Let L_{TV} be the language from Theorem 17. Since this language is PSPACE-hard under polynomial-time reductions, it suffices to show that $L_{\mathsf{TV}} \in \mathsf{BPP}^{\mathsf{MrKtP}[\beta,\alpha,s]}$.

We argue as in the proof of Lemma 12. More precisely, we let $\varepsilon \stackrel{\text{def}}{=} \gamma/2$, and we instantiate the generator G^{IW}_f using the function f that computes the characteristic function of L_{TV}. If $\mathcal{O}$ is a deterministic test that agrees with $\mathsf{MrKtP}[\beta, \alpha, s]$, then a similar argument shows that every output string of the generator has randomized Kt complexity at most $s(n)$, while a random string has almost maximum complexity. The only modification here is that $f_1, \ldots, f_{n^{\varepsilon}}$ can all be computed in *deterministic* time $\exp(O(n^{\varepsilon}))$, which follows from the fact that L_{TV} is computable in linear space. Theorem 16 immediately implies that $L_{\mathsf{TV}} \in \mathsf{BPP}^{\mathcal{O}}$, as desired.

Suppose that $\mathcal{O}$ is a *randomized* procedure that accepts the positive instances of $\mathsf{MrKtP}[\beta, \alpha, s]$ with high probability, and rejects the negative instances of $\mathsf{MrKtP}[\beta, \alpha, s]$ with high probability. We make no assumptions on the acceptance probabilities of $\mathcal{O}$ over the remaining input strings. In order to establish the furthermore part in Lemma 13, it is necessary to inspect the proof of Theorem 16. The crucial observation is that the oracle $\mathcal{O}$ is only used as a distinguisher during the computation of L_{TV}, and that any procedure that distinguishes with noticeable advantage the output of the generator from a random string can be used in place of $\mathcal{O}$. (The argument sketched in the paragraph above can be used to show that the output of $\mathcal{O}$ on strings that violate the promise condition affects in a negligible way its advantage as a distinguisher.)

We also note that it is possible to reduce the analysis of the case of a randomized algorithm A as oracle to the deterministic case. By running polynomially many independent copies of A and taking a majority vote, one gets a randomized algorithm A' that is correct with probability at least $1 - 2^{-m^2}$ on every fixed string of length at most m satisfying the promise condition (think of m as n^{ℓ} for a large enough constant ℓ, where n is the input length of L_{TV}). By a union bound, randomly fixing the string in the random tape of A' provides w.h.p a deterministic oracle $\mathcal{O}$ that is correct on all strings of length at most m satisfying the promise condition. The analysis now reduces to the deterministic case.

This completes the proof of Lemma 13. ◀

Sketch of an alternate presentation via learning algorithms. Suppose that $\mathsf{MrKtP} \in \mathsf{BPP}$, i.e., there is a polynomial time randomized algorithm that is correct with high probability over inputs satisfying the promise condition. Then, by adapting ideas from [11], it is possible to prove that for every reasonable function $t \colon \mathbb{N} \to \mathbb{N}$, $\mathsf{SIZE}[t]$ can be learned in $\mathsf{BPTIME}[\mathsf{poly}(t)]$ in the model of learning with membership queries under the uniform distribution. The connection between learning algorithms and lower bounds (see [29]) now implies that, for any choice of $s(n) \leq n^{\mathsf{poly}(\log n)}$, $\mathsf{BPEXP} \not\subseteq \mathsf{SIZE}[s(n)]$. But this is in contradiction to Lemma 12 and the assumption that $\mathsf{MrKtP} \in \mathsf{BPP}$, which imply $\mathsf{BPEXP} \subseteq \mathsf{SIZE}[\mathsf{poly}(n)]$.

We remark that common to both approaches are elements from the theory of pseudorandomness, such as the use of pseudorandom generators based on [27], and ideas that go back to the work of [18] on connections between algorithms and lower bounds via random self-reducibility and downward self-reducibility. The use of [6] in the proof of Lemma 12 appears to be crucial in the arguments presented above.

3.2 Weakness of deterministic algorithms for MKtP and MrKtP

It is natural to conjecture that $\mathsf{BPEXP} \not\subseteq \mathsf{BPTIME}[2^{o(n)}]$ (a strong hierarchy theorem for randomized time) and $\mathsf{MrKtP} \notin \mathsf{BPTIME}[2^{o(n)}]$ (a nearly-optimal lower bound for MrKtP). However, it is unclear even how to show that $\mathsf{MrKtP} \notin \mathsf{DTIME}[2^{n^{o(1)}}]$. It is also open whether $\mathsf{MKtP} \in \mathsf{P}$. In this section, we place limits on the efficiency of deterministic algorithms solving these problems. We start with the following observation.

▶ **Proposition 18.** *Either* $\mathsf{EXP} \nsubseteq \mathsf{BPTIME}[2^{o(n)}]$ *or* $\mathsf{MrKtP} \notin \mathsf{EXP}$.

Proof. Suppose $\mathsf{MrKtP} = \mathsf{MrKtP}[\beta, \alpha, s]$ is in $\mathsf{DTIME}[2^{n^d}]$ for some constant d, where $\beta = 3/4$, $\alpha = 2/3$, and $s = n/2$. Let M_{MrKtP} be a Turing machine that witnesses this inclusion. Consider the following language:

$$L \overset{\text{def}}{=} \{\langle M, 1^n, w\rangle \mid M \text{ is a TM that accepts in time } \leq 2^{n^d} \\ \text{some } n\text{-bit string } y \text{ whose prefix is } w\}.$$

Note that $L \in \mathsf{EXP}$, i.e., L can be computed in deterministic time $2^{m^{O(1)}}$ on inputs of length m. Assume that $\mathsf{EXP} \subseteq \mathsf{BPTIME}[2^{o(n)}]$, and let M_L be a randomized Turing machine for L that witnesses this inclusion. It is easy to see that M_L can be used to find w.h.p. and in time $2^{o(n)}$ the lexicographic first string $z \in \{0,1\}^n$ accepted by $\overline{M_{\mathsf{MrKtP}}}$, the complement of machine M_{MrKtP} (observe that such string must exist). It follows that the triple $(M_L, M_{\mathsf{MrKtP}}, 1^n)$ can be used to give a shorter description of z. More precisely, $\mathsf{rKt}_{1-o(1)}(z) = o(n)$. On the other hand, since $\overline{M_{\mathsf{MrKtP}}}(z) = 1$ and M_{MrKtP} computes MrKtP, we must have $\mathsf{rKt}_\beta(z) > s$. These inequalities imply that $n/2 = s < \mathsf{rKt}_\beta(z) \leq \mathsf{rKt}_{1-o(1)}(z) \leq o(n)$, a contradiction. This completes the proof of Proposition 18. ◀

Additionally, we will use the following reductions.

▶ **Lemma 19** (see [4]). $\mathsf{EXP} \subseteq \mathsf{NP}^{\mathsf{MKtP}}$.

▶ **Lemma 20** (see [4]). *If* $\mathsf{MKtP} \in \mathsf{P}$ *then* $\mathsf{PSPACE} \subseteq \mathsf{ZPP}$.

As alluded to above, the next result shows hardness of deciding deterministic/randomized time-bounded Kolmogorov complexity using deterministic algorithms. (It should be contrasted with the inclusion $\mathsf{MrKtP} \in \mathsf{Promise\text{-}BPE}$ from Lemma 8.)

▶ **Theorem 21.** *Either* $\mathsf{MKtP} \notin \mathsf{P}$ *or* $\mathsf{MrKtP} \notin \mathsf{EXP}$.

Proof. Suppose $\mathsf{MKtP} \in \mathsf{P}$. Then $\mathsf{PSPACE} \subseteq \mathsf{ZPP}$ follows by Lemma 20. Moreover, Lemma 19 gives $\mathsf{EXP} \subseteq \mathsf{NP}$. Combining these two class inclusions, we get that $\mathsf{EXP} \subseteq \mathsf{BPP}$. But this implies that $\mathsf{MrKtP} \notin \mathsf{EXP}$ via Proposition 18, which is the desired result. ◀

4 Non-uniform versus randomized lower bounds for MrKtP

It is not hard to see that the proof of Theorem 11 carries over with $\mathsf{Gap\text{-}MrKtP}[s_1, s_2]$ in place of $\mathsf{MrKtP}[\beta, \alpha, s]$. We state the result here for completeness.

▶ **Theorem 22.** *Let* $\gamma > 0$ *be an arbitrarily small constant, and consider functions* $s_1, s_2 \colon \mathbb{N} \to \mathbb{N}$. *Suppose that* $n^\gamma \leq s_1(n) \leq s_2(n) \leq n/2$. *Then* $\mathsf{Gap\text{-}MrKtP}[s_1, s_2] \notin \mathsf{BPTIME}[n^{\mathsf{poly}(\log n)}]$.

Proof Sketch. The algorithm for $\mathsf{MrKtP}[\beta, \alpha, s]$ is only used as a distinguisher in the proof of Theorem 11. It is possible to check that an algorithm for $\mathsf{Gap\text{-}MrKtP}[s_1, s_2]$ works equally well as a distinguisher in the proofs of Lemmas 12 and 13. ◀

By a straightforward extension of results from [31, 28], one can show that weak *non-uniform* lower bounds for $\mathsf{Gap\text{-}MrKtP}[s_1, s_2]$ can be "magnified" to super-polynomial circuit lower bounds for some explicit problem. We state a version of the result for general boolean circuits, but the proof can be adapted to other boolean devices (similarly to [28]).

▶ **Theorem 23.** *There is a universal constant $d \geq 1$ for which the following holds. If there exists $\varepsilon > 0$ such that for every small enough $\beta > 0$ we have* $\mathsf{Gap\text{-}MrKtP}[n^\beta, n^\beta + d \log n] \notin \mathsf{SIZE}[n^{1+\varepsilon}]$*, then* $\mathsf{Promise\text{-}BPEXP} \not\subseteq \mathsf{SIZE}[\mathsf{poly}]$.

Proof Sketch. We verify that the relevant steps in the proof of [28, Theorem 1 Part 1, Section 3] carry over to $\mathsf{Gap\text{-}MrKtP}[n^\beta, n^\beta + d \log n]$, under minor modifications. (Note that N denotes the input length in [28], while here input length is denoted by n.) First, we observe that Claims 11 and 12 also work for rKt, since the error-correcting code routines are deterministic. Using a similar notation (i.e., $z = \mathsf{ECC}(w) \in \{0,1\}^m$, where $m = m(n) = O(n)$ and $n = |w|$), it follows that if $\mathsf{rKt}(w) \leq n^\beta$ then $\mathsf{rKt}(z) \leq n^\beta + c_0 \log n$, and that if $\mathsf{rKt}(w) > n^\beta + d \log n$ then $\mathsf{rKt}(z') > n^\beta + c_1 \log n$ for any $z' \in \{0,1\}^m$ that disagrees with z on at most a δ-fraction of coordinates, where $1 \leq c_0 < c_1 < d$ are constants, and we assume that d is large enough so that c_0 and c_1 are sufficiently far apart.

The crucial part of the argument is to replace the language $L \in \mathsf{EXP}$ from their Claim 13 by an appropriate problem $\Pi \in \mathsf{Promise\text{-}BPEXP}$. The input to Π is a string y encoding a tuple of the form $(m, 1^t, (i_1, b_1), \ldots, (i_r, b_r))$, where m is a positive integer represented in binary, t is a positive integer, $i_1, \ldots, i_r \in \{0,1\}^{\log m}$, $b_1, \ldots, b_r \in \{0,1\}$, and $r \in \mathbb{N}$. For $t = n^\beta$ and $r = n^{2\beta}$, we let

$$\Pi_n^{\mathsf{yes}} \overset{\text{def}}{=} \{y \mid \exists z \in \{0,1\}^m \text{ such that } \mathsf{rKt}(z) \leq t + c_0 \log n \text{ and } z_{i_1} = b_1, \ldots, z_{i_r} = b_r\}, \text{ and}$$

$$\Pi_n^{\mathsf{no}} \overset{\text{def}}{=} \{y \mid \nexists z \in \{0,1\}^m \text{ such that } \mathsf{rKt}(z) \leq t + c_1 \log n \text{ and } z_{i_1} = b_1, \ldots, z_{i_r} = b_r\}.$$

Note that these sets are disjoint, and that $\Pi \in \mathsf{Promise\text{-}BPEXP}$ by an argument analogous to the proof of Lemma 10, using that the gap between constants c_0 and c_1 is sufficiently large.

It remains to check that their Claim 14 still holds in our context. For part (a), note that if $\mathsf{rKt}(w) \leq n^\beta$ then by the discussion above $\mathsf{rKt}(z) \leq n^\beta + c_0 \log n$. Consequently, the corresponding input y generated by the randomized reduction is in Π_n^{yes} with probability 1. Similarly, for part (b) we rely on the claim that if $\mathsf{rKt}(w) > n^\beta + d \log n$ then $\mathsf{rKt}(z') > n^\beta + c_1 \log n$ for any $z' \in \{0,1\}^m$ that disagrees with z on at most a δ-fraction of coordinates. The same union bound over exponentially small probabilities shows that $y \in \Pi_n^{\mathsf{no}}$ with probability at least $\geq 1/2$.

The rest of the construction remains unaffected. ◀

5 On the relation between rKt and Kt

First, we observe that the worst-case gap between rKt and Kt over strings of length n is closely related to the derandomization of exponential time computations.[7]

▶ **Theorem 24.** *The following implications hold.*

(i) *If* $\mathsf{Promise\text{-}BPE} \subseteq \mathsf{Promise\text{-}E}$*, then* $\mathsf{Kt}(w) = O(\mathsf{rKt}(w))$ *for every string w.*

(ii) *If* $\mathsf{Kt}(w) = O(\mathsf{rKt}(w))$ *for every string w, then* $\mathsf{BPE} \subseteq \mathsf{E}/O(n)$.

In particular, rKt *and* Kt *are linearly related if* E *requires exponential size boolean circuits.*

[7] Recall that if $\mathsf{BPP} \subseteq \mathsf{P}$ then $\mathsf{BPEXP} \subseteq \mathsf{EXP}$ by translation. Consequently, derandomizing exponential time computations is not harder than derandomizing polynomial time computations. Indeed, it is not hard to prove that the derandomization of exponential time is equivalent to the derandomization of sparse languages in BPP.

Proof. We start with a proof of (i). Let $w \in \{0,1\}^n$, and suppose M and t are such that $\Pr[\boldsymbol{M}_{\leq t} = w] \geq 2/3$ and $|M| + \log t = \mathsf{rKt}(w)$. We would like to use M and the assumption that $\mathsf{Promise\text{-}BPE} \subseteq \mathsf{Promise\text{-}E}$ to upper bound the Kt complexity of w. A potential difficulty here is that the latter inclusion offers an asymptotic upper bound, while M and w are fixed objects of finite size. In order to handle this issue, we adopt a more general perspective.

Let U be a randomized universal Turing machine that simulates computations with a polynomial overhead. In other words, given the description of a randomized machine M', a time bound t' specified as a binary string, and an input string x', $U(M', t', x')$ uses its internal randomness to simulate $M'(x')$ for at most t' steps, and outputs whatever M' outputs on x'. We assume that the computation of $U(M', t', x')$ takes time at most $c(|M'| + t' + |x'|)^c$, where $c = c(U) \geq 1$ is a universal constant.

We consider a promise problem Π, defined as follows. The $\mathcal{YES}$ instances consist of tuples $(M', t', 1^{c \cdot \log t'}, i)$, where M' is the description of a randomized Turing machine, t' and i are positive integers represented in binary, and $\Pr[\text{The } i\text{-th bit of } \boldsymbol{M}'_{\leq t'} \text{ is } 1] \geq 2/3$. On the other hand, the set $\mathcal{NO}$ of negative instances of Π is defined by the condition $\Pr[\text{The } i\text{-th bit of } \boldsymbol{M}'_{\leq t'} \text{ is } 0] \geq 2/3$. Clearly, $\mathcal{YES} \cap \mathcal{NO} = \emptyset$. We claim that $\Pi \in \mathsf{Promise\text{-}BPE}$. In order to see this, given a valid input $(M', t', 1^{c \cdot \log t'}, i)$ of Π, run the randomized universal machine U on (M', t', ϵ) for t' steps, where ϵ is the empty string, and output 1 if and only if the i-th bit in the output of M' is 1. This defines a randomized machine that runs in time $O((|M'| + t')^c + i)$, which is at most exponential in its total input length. Since the randomness of U is used to simulate the randomness of M', every string in $\mathcal{YES}$ is accepted with probability at least $2/3$, while every string in $\mathcal{NO}$ is rejected with probability at least $2/3$. This shows that $\Pi \in \mathsf{Promise\text{-}BPE}$.

Under the hypothesis of (i), we obtain that Π is computed by a deterministic machine A_Π that runs in time at most 2^{Cm} on inputs of length m, where C is fixed. Now given the pair (M, t) witnessing the rKt complexity of w (a string of length n), we can use A_Π, M, t, and n to upper bound its Kt complexity. Indeed, w can be generated by the deterministic machine that runs A_Π on $(M, t, 1^{c \cdot \log t}, i)$ for each $i \in [n]$. Note that each input to A_Π satisfies the promise condition of Π, and that A_Π runs in time at most $2^{C(|M| + \log t + c \log t + \log i)}$. Therefore, $\mathsf{rKt}(w) \leq O(|M| + |A_\Pi| + \log t + \log n) + \log(O(n \cdot 2^{C(|M| + \log t + c \log t + \log n)})) = O(|M| + \log t + \log n) = O(\mathsf{rKt}(w) + \log |w|) = O(\mathsf{rKt}(w))$, where the last inequality uses that $\mathsf{rKt}(w) \geq \log |w|$ since any machine that prints w runs in time at least $|w|$.

To prove (ii), let $L \in \mathsf{BPE}$, and let M be a machine for L that runs in randomized exponential time. Define a sequence $\{w_n\}_{n \geq 1}$ of strings $w_n \in \{0,1\}^{2^{n+1}}$, where w_n encodes the output of L on all strings of length at most n. Given n as an input, by amplifying the success probability of M, we can print w_n with high probability in time $2^{O(n)}$. Consequently, $\mathsf{rKt}(w_n) = O(n)$, which is logarithmic in $|w_n|$. Under the assumption that $\mathsf{Kt}(w) = O(\mathsf{rKt}(w))$ for every string w, it follows that for every n, $\mathsf{Kt}(w_n) = O(n)$. In particular, some deterministic machine M_n with $|M_n| = O(n)$ decides L on inputs of length at most n in time $2^{O(n)}$. Now using the sequence $\{M_n\}_{n \geq 1}$ as advice and computing in the obvious way, it follows that $L \in \mathsf{E}/O(n)$. This completes the proof of (ii).

Finally, under the assumption that there is a language in E that requires circuits of size $2^{\Omega(n)}$ on every large input length, there are quick pseudorandom generators of logarithmic seed length (cf. [34]). Such generators can be used to derandomize not only $\mathsf{BPTIME}[t]$ but also $\mathsf{Promise\text{-}BPTIME}[t]$, hence it follows from (i) that rKt and Kt are linearly related. ◀

We now relate the deterministic complexity of MKtP to the gap between rKt and Kt.

▶ **Theorem 25.** *If* $\mathsf{MKtP} \in \mathsf{P}$ *then there is a sequence* $\{w_n\}_{n \geq 1}$ *with* $w_n \in \{0,1\}^n$ *such that* $\mathsf{rKt}(w_n) = O(\log n)$ *and* $\mathsf{Kt}(w_n) = \Omega(n)$.

Proof. The proof is inspired by a related idea of Schuichi Hirahara (private communication). Let $\{D_n\}_{n \geq 1}$ be a P-uniform sequence of polynomial size circuits computing MKtP_t, for a Kt complexity threshold parameter $t(n) = n/2$. The existence of such circuits follows from the hypothesis of the theorem. Now Lemma 20 implies that there is a randomized algorithm running in time polynomial in n that solves the circuit satisfiability problem for circuits of size $\mathsf{poly}(n)$ over n input variables. We can use this algorithm and self-reduction to find with high probability the lexicographic first string w_n accepted by the complement of D_n. Then, by construction, we get that $\mathsf{rKt}(w_n) = O(\log n)$ and $\mathsf{Kt}(w_n) = \Omega(n)$. ◀

References

1 Eric Allender. Applications of time-bounded Kolmogorov complexity in complexity theory. In *Kolmogorov complexity and computational complexity*, pages 4–22. Springer, 1992.

2 Eric Allender. When Worlds Collide: Derandomization, Lower Bounds, and Kolmogorov Complexity. In *Conference on Foundations of Software Technology and Theoretical Computer Science* (FSTTCS), pages 1–15, 2001. doi:10.1007/3-540-45294-X_1.

3 Eric Allender. The complexity of complexity. In *Computability and Complexity*, pages 79–94. Springer, 2017.

4 Eric Allender, Harry Buhrman, Michal Koucký, Dieter van Melkebeek, and Detlef Ronneburger. Power from Random Strings. *SIAM J. Comput.*, 35(6):1467–1493, 2006. doi:10.1137/050628994.

5 Eric Allender, Michal Koucký, Detlef Ronneburger, and Sambuddha Roy. The pervasive reach of resource-bounded Kolmogorov complexity in computational complexity theory. *J. Comput. Syst. Sci.*, 77(1):14–40, 2011. doi:10.1016/j.jcss.2010.06.004.

6 László Babai, Lance Fortnow, Noam Nisan, and Avi Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3:307–318, 1993. doi:10.1007/BF01275486.

7 Boaz Barak. A Probabilistic-Time Hierarchy Theorem for "Slightly Non-uniform" Algorithms. In *International Workshop on Randomization and Approximation Techniques* (RANDOM), pages 194–208, 2002. doi:10.1007/3-540-45726-7_16.

8 André Berthiaume, Wim van Dam, and Sophie Laplante. Quantum Kolmogorov Complexity. *J. Comput. Syst. Sci.*, 63(2):201–221, 2001. doi:10.1006/jcss.2001.1765.

9 Harry Buhrman, Lance Fortnow, and Rahul Santhanam. Unconditional Lower Bounds against Advice. In *International Colloquium on Automata, Languages and Programming* (ICALP), pages 195–209, 2009. doi:10.1007/978-3-642-02927-1_18.

10 Harry Buhrman, Lance Fortnow, and Thomas Thierauf. Nonrelativizing Separations. In *Conference on Computational Complexity* (CCC), pages 8–12, 1998. doi:10.1109/CCC.1998.694585.

11 Marco L. Carmosino, Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova. Learning Algorithms from Natural Proofs. In *Conference on Computational Complexity* (CCC), pages 10:1–10:24, 2016. doi:10.4230/LIPIcs.CCC.2016.10.

12 Gregory J. Chaitin. Information-Theoretic Limitations of Formal Systems. *J. ACM*, 21(3):403–424, 1974. doi:10.1145/321832.321839.

13 Lance Fortnow. Kolmogorov complexity and computational complexity. *Complexity of Computations and Proofs. Quaderni di Matematica*, 13, 2004.

14 Lance Fortnow, Rahul Santhanam, and Luca Trevisan. Hierarchies for semantic classes. In *Symposium on Theory of Computing* (STOC), pages 348–355, 2005. doi:10.1145/1060590.1060642.

15 Eran Gat and Shafi Goldwasser. Probabilistic Search Algorithms with Unique Answers and Their Cryptographic Applications. *Electronic Colloquium on Computational Complexity* (ECCC), 18:136, 2011.

16 Shuichi Hirahara. Non-Black-Box Worst-Case to Average-Case Reductions within NP. In *Symposium on Foundations of Computer Science* (FOCS), pages 247–258, 2018. doi:10.1109/FOCS.2018.00032.

17 Shuichi Hirahara and Rahul Santhanam. On the Average-Case Complexity of MCSP and Its Variants. In *Computational Complexity Conference* (CCC), pages 7:1–7:20, 2017. doi:10.4230/LIPIcs.CCC.2017.7.

18 Russell Impagliazzo and Avi Wigderson. Randomness vs Time: Derandomization under a Uniform Assumption. *J. Comput. Syst. Sci.*, 63(4):672–688, 2001. doi:10.1006/jcss.2001.1780.

19 Marek Karpinski and Rutger Verbeek. On the Monte Carlo Space Constructible Functions and Seperation Results for Probabilistic Complexity Classes. *Inf. Comput.*, 75(2):178–189, 1987. doi:10.1016/0890-5401(87)90057-5.

20 Makoto Kikuchi. Kolmogorov complexity and the second incompleteness theorem. *Archive for Mathematical Logic*, 36(6):437–443, 1997.

21 Adam R. Klivans and Dieter van Melkebeek. Graph Nonisomorphism Has Subexponential Size Proofs Unless the Polynomial-Time Hierarchy Collapses. *SIAM J. Comput.*, 31(5):1501–1526, 2002. doi:10.1137/S0097539700389652.

22 Shira Kritchman and Ran Raz. The surprise examination paradox and the second incompleteness theorem. *Notices of the AMS*, 57(11):1454–1458, 2010.

23 Leonid A. Levin. Universal sequential search problems. *Problemy Peredachi Informatsii*, 9(3):115–116, 1973.

24 Leonid A. Levin. Randomness Conservation Inequalities; Information and Independence in Mathematical Theories. *Information and Control*, 61(1):15–37, 1984. doi:10.1016/S0019-9958(84)80060-1.

25 Ming Li and Paul M. B. Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications*. Texts in Computer Science. Springer, 2008. doi:10.1007/978-0-387-49820-1.

26 Caterina E. Mora and Hans J. Briegel. Algorithmic Complexity and Entanglement of Quantum States. *Phys. Rev. Lett.*, 95:200503, 2005. doi:10.1103/PhysRevLett.95.200503.

27 Noam Nisan and Avi Wigderson. Hardness vs Randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994. doi:10.1016/S0022-0000(05)80043-1.

28 Igor Carboni Oliveira, Ján Pich, and Rahul Santhanam. Hardness magnification near state-of-the-art lower bounds. *Electronic Colloquium on Computational Complexity* (ECCC), 25:158, 2018. URL: https://eccc.weizmann.ac.il/report/2018/158.

29 Igor Carboni Oliveira and Rahul Santhanam. Conspiracies Between Learning Algorithms, Circuit Lower Bounds, and Pseudorandomness. In *Computational Complexity Conference* (CCC), pages 18:1–18:49, 2017. doi:10.4230/LIPIcs.CCC.2017.18.

30 Igor Carboni Oliveira and Rahul Santhanam. Pseudodeterministic constructions in subexponential time. In *Symposium on Theory of Computing* (STOC), pages 665–677, 2017. doi:10.1145/3055399.3055500.

31 Igor Carboni Oliveira and Rahul Santhanam. Hardness Magnification for Natural Problems. In *Symposium on Foundations of Computer Science* (FOCS), pages 65–76, 2018. doi:10.1109/FOCS.2018.00016.

32 Karl Svozil. Quantum Algorithmic Information Theory. *J. UCS*, 2(5):311–346, 1996. doi:10.3217/jucs-002-05-0311.

33 Luca Trevisan and Salil P. Vadhan. Pseudorandomness and Average-Case Complexity Via Uniform Reductions. *Computational Complexity*, 16(4):331–364, 2007. doi:10.1007/s00037-007-0233-x.

34 Christopher Umans. Pseudo-random generators for all hardnesses. *J. Comput. Syst. Sci.*, 67(2):419–440, 2003. doi:10.1016/S0022-0000(03)00046-1.

35 Paul M. B. Vitányi. Quantum Kolmogorov complexity based on classical descriptions. *IEEE Trans. Information Theory*, 47(6):2464–2479, 2001. doi:10.1109/18.945258.