Electronic Colloquium on Computational Complexity, Report No. 89 (2025)

Chain Rules for Time-Bounded Kolmogorov Complexity

Valentine Kabanets* Antonina Kolokolova†

July 9, 2025

Abstract

Time-bounded conditional Kolmogorov complexity of a string x given y, $\mathsf{K}^t(x \mid y)$, is the length of a shortest program that, given y, prints x within t steps. The Chain Rule for conditional K^t with error e is the following hypothesis: there is a constant $c \in \mathbb{N}$ such that, for any strings $y, x_1, \ldots, x_\ell \in \{0,1\}^*$, for any $\ell \in \mathbb{N}$, and all sufficiently large time bounds t,

$$\mathsf{K}^t(x_1, \ldots, x_\ell \mid y) \geq \sum_{i=1}^{\ell} \mathsf{K}^{t^c}(x_i \mid y, x_1, \ldots, x_{i-1}) - \ell \cdot O(\log t) - e(N,t),$$

where $N = \sum_{i=1}^{\ell} |x_i|$.

We pinpoint the complexity assumptions *equivalent* to Chain Rules for conditional K^t, and the probabilistic variant pK^t, where $\mathsf{pK}^t(x \mid y) \leq s$ iff $\mathsf{K}^t(x \mid y, r) \leq s$ for at least $2/3$ of random strings $r \in \{0,1\}^t$.

- Chain Rule for conditional K^t with error $e(N,t) \leq o(N)$ is *equivalent* to the conjunction of the following two statements:
  1. $\mathsf{E} \not\subset \mathsf{io\text{-}SIZE}[2^{o(n)}]$, and
  2. $\mathsf{GapMcK^tP} \in \mathsf{promise\text{-}P}$, where $\mathsf{GapMcK^tP}$ is a promise problem to distinguish between inputs $(x, y, 1^s)$ with $\mathsf{K}^t(x \mid y) \leq s$ and those with $\mathsf{K}^{\mathsf{poly}(t)}(x \mid y) > s + o(|x|)$.
- Chain Rule for conditional pK^t with error $e(N,t) \leq o(N)$ is *equivalent* to $\mathsf{GapMcpK^tP} \in \mathsf{promise\text{-}BPP}$, for the analog of $\mathsf{GapMcK^tP}$ for conditional pK^t.

These are the first exact complexity characterizations for natural versions of Chain Rules for time-bounded Kolmogorov complexity.

Assuming $\mathsf{GapMcK^tP}$ is NP-hard (which is true under cryptographic assumptions [HIR23]), the equivalence above would simplify to "the Chain Rule for conditional K^t with error $e(N,t) \leq o(N)$ holds iff $\mathsf{NP} = \mathsf{P}$". That is, under a plausible NP-hardness assumption for $\mathsf{GapMcK^tP}$, we would get that proving $\mathsf{P} \neq \mathsf{NP}$ is equivalent to disproving the Chain Rule for conditional K^t.

Among other results, we present a natural $\mathsf{promise\text{-}BPP}$-complete problem based on the problem of approximating $\mathsf{pK}^t(x \mid y)$ for short inputs x with $|x| \leq \log t$, and give some algorithmic consequences if $\mathsf{GapMcpK^tP}$ were easy.

*School of Computing Science, Simon Fraser University, Burnaby, BC, Canada; kabanets@sfu.ca
†Department of Computer Science, Memorial University of Newfoundland, NL, Canada; kol@mun.ca

ISSN 1433-8092

Contents

1 Introduction

Kolmogorov complexity defines an algorithmic measure of information in a given string x, denoted $\mathsf{K}(x)$, as the length of a shortest program that prints x. The conditional version $\mathsf{K}(x \mid y)$ is the length of a shortest program that, given y, prints x.

One of the fundamental properties of the Kolmogorov complexity measure K, discovered by Kolmogorov and Levin (see [ZL70]), is the Chain Rule, saying that for any binary strings x, y, and z,

$$\mathsf{K}(x, y \mid z) \approx \mathsf{K}(x \mid z) + \mathsf{K}(y \mid z, x), \tag{1}$$

where $\approx$ hides an additive error term, at most logarithmic in the total length of all strings. That is, the most efficient way to describe a pair (x, y) given z is to describe one of them given z, and then describe the other one assuming the knowledge of z and the first one.

When $z = \epsilon$ (the empty string), this Chain Rule is also called *Symmetry of Information (SoI)* for K, since it implies

$$\mathsf{K}(x) - \mathsf{K}(x \mid y) \approx \mathsf{K}(y) - \mathsf{K}(y \mid x),$$

reminding of the classical symmetry of information equality $H(X) - H(X \mid Y) = H(Y) - H(Y \mid X)$, where $H(X)$ is Shannon's entropy of a random variable X.

SoI for K is universally regarded as one of the most beautiful and useful results in classical Kolmogorov complexity.

Symmetry of Information for Time-Bounded Kolmogorov Complexity. Of particular interest to computational complexity is the variant of *time-bounded* Kolmogorov complexity measure K^t, where $\mathsf{K}^t(x)$ refers to the length of a shortest program that prints x within t time steps. An obvious question is whether the time-bounded measure K^t satisfies SoI. Since the upper bound $\mathsf{K}^{2t}(x, y) \lessapprox \mathsf{K}^t(x) + \mathsf{K}^t(y \mid x)$ is straightforward, SoI for K^t is defined as the following hypothesis:

$$\mathsf{K}^t(x, y) \geq \mathsf{K}^{p(t)}(x) + \mathsf{K}^{p(t)}(y \mid x) - O(\log t), \tag{SoI for K^t}$$

for some polynomial p, and any sufficiently large time bound t.

According to Levin [Lev03], Kolmogorov was interested in the SoI for K^t question, since he believed it might be useful in the quest to resolve open problems in computational complexity. In particular, Kolmogorov suggested in the late 1960's that disproving SoI for K^t might be a good approach to proving that $\mathsf{P} \neq \mathsf{NP}$. [1]

However, it was only in 1995 that Longpre and Watanabe [LW95] proved that $\mathsf{P} = \mathsf{NP}$ implies SoI for K^t. Thus, Kolmogorov's suggestion to disprove SoI fof K^t in order to prove $\mathsf{P} \neq \mathsf{NP}$ was sound. Since then, it remains an open question if disproving SoI for K^t is also *necessary* for proving $\mathsf{P} \neq \mathsf{NP}$. More generally, the open question is to come up with a *natural complexity statement* (like $\mathsf{P} = \mathsf{NP}$) that would be *equivalent* to SoI for K^t.

[1] Kolmogorov's intuition was, perhaps, that his proof of the chain rule for the time-unbounded K used an algorithm doing a *brute-force search* over exponentially many possibilities, and it was not clear to him how such exhaustive search could be avoided.

Multi-String Chain Rules. Applying the Chain Rule in (1) inductively, we easily get the following version of the chain rule for multiple strings: For any $x_1, \ldots, x_\ell, y \in \{0,1\}^*$,

$$\mathsf{K}(x_1, \ldots, x_\ell \mid y) \approx \sum_{i=1}^{\ell} \mathsf{K}(x_i \mid y, x_1, \ldots, x_{i-1}), \tag{2}$$

where $\approx$ hides an additive error term at most $\ell \cdot O(\log(\sum_{i=1}^{\ell} |x_i| + |y|))$.

An analogous version for the time-bounded case is the following hypothesis:

$$\mathsf{K}^t(x_1, \ldots, x_\ell \mid y) \geq \sum_{i=1}^{\ell} \mathsf{K}^{p(t)}(x_i \mid y, x_1, \ldots, x_{i-1}) - \ell \cdot O(\log t) - e(N), \qquad \text{(Chain Rule for } \mathsf{cK}^t\text{)}$$

where p is some polynomial that is the *same* for any number ℓ of strings $x_1, \ldots, x_\ell$, y is any string, t is a sufficiently large polynomial time bound in the total string length $N + |y|$ for $N = \sum_{i=1}^{\ell} |x_i|$, and $e(N)$ is an error term.

Note that the interesting setting of parameters for this Chain Rule is when ℓ is *super-constant* (for example, $\ell = o(N/\log t)$). For super-constant ℓ, unlike in the time-unbounded case of K, SoI for K^t (the chain rule for two strings) does not seem to imply the multi-string chain rule by induction on ℓ. [2] Thus, in the time-bounded setting, the multi-string Chain Rule appears to be *more powerful* than SoI.

This, potentially more powerful, version of the chain rule for conditional K^t is our main subject of study. For this chain rule, we do get an *exact complexity characterization*, as we explain next.

1.1 Results

Chain rule for conditional K^t. First we need to define the problem $\mathsf{Gap}_{\tau,o(n)}\mathsf{McK}^{\mathsf{t}}\mathsf{P}$. This is a promise-problem for conditional $\mathsf{K}^t(x \mid y)$, where one needs to distinguish between inputs $(x, y, 1^s)$ with $\mathsf{K}^t(x \mid y) \leq s$ and those with $\mathsf{K}^{\tau(t)}(x \mid y) > s + o(|x|)$, for some polynomial τ.

Theorem 1.1 (Case of conditional K^t, informal)**.** *Chain Rule for conditional K^t with multiple strings, with error $e(N) \leq o(N)$, is* equivalent *to the conjunction*

$$\mathsf{E} \not\subset \mathsf{io\text{-}SIZE}[2^{o(n)}] \;\; \& \;\; \mathsf{Gap}_{\tau,o(n)}\mathsf{McK}^{\mathsf{t}}\mathsf{P} \in \mathsf{promise\text{-}P}.$$

The problem $\mathsf{Gap}_{\tau,o(n)}\mathsf{McK}^{\mathsf{t}}\mathsf{P}$ is likely to be NP-hard. For example, [HIR23] show that this problem is NP-hard under randomized polynomial-time reductions, assuming the existence of subexponentially secure witness encryption[3] for NP.

If we suppose that $\mathsf{Gap}_{\tau,o(n)}\mathsf{McK}^{\mathsf{t}}\mathsf{P}$ is indeed NP-hard, our equivalence in Theorem 1.1 would simplify to the following: "Chain Rule for conditional K^t with multiple strings, with error $e(N) \leq o(N)$, is *equivalent* to $\mathsf{P} = \mathsf{NP}$". This would show that disproving the Chain Rule for K^t is also *necessary* for proving $\mathsf{P} \neq \mathsf{NP}$, validating Kolmogorov's intuition to the fullest!

[2] The reason is a polynomial blowup in the time bounds for K^t on the right-hand side of SoI after each inductive step, resulting in super-polynomial in t time bounds on the right-hand side of the resulting chain rule when ℓ is super-constant.

[3] We note that secure witness encryption for NP, introduced in [Gar+13], may exist in either the world where there are one-way functions, or the world where $\mathsf{P} = \mathsf{NP}$.

Chain rule for conditional pK^t. In our equivalence for the chain rule for K^t above, we have a derandomization assumption of exponential circuit complexity for a problem in exponential time. This circuit lower bound assumption may be dropped if we consider a probabilistic variant of time-bounded Kolmogorov complexity pK^t instead of K^t. This probabilistic measure can be defined as follows: $\mathsf{pK}^t(x \mid y) \leq s$ if $\mathsf{K}^t(x \mid y, r) \leq s$ for at least $2/3$ of uniformly random strings $r \in \{0,1\}^t$.

We need the problem $\mathsf{Gap}_{\tau,o(n)}\mathsf{McpK^tP}$, which is a promise problem for conditional $\mathsf{pK}^t(x \mid y)$, defined analogously to $\mathsf{Gap}_{\tau,o(n)}\mathsf{McK^tP}$ above.

Theorem 1.2 (Case of conditional pK^t, informal)**.** *Chain Rule for conditional pK^t with multiple strings, with error $e(N) \leq o(N)$, is* equivalent *to $\mathsf{Gap}_{\tau,o(n)}\mathsf{McpK^tP} \in \mathsf{promise\text{-}BPP}$.*

Theorems 1.1 and 1.2 (and related ones) are formally stated and proved in Section 6.

The complexity of computing pK^t. While it is likely that the problem $\mathsf{Gap}_{\tau,o(n)}\mathsf{McpK^tP}$ considered above may be NP-hard, we do not have any proof yet. A natural approach to exploring the computational power of a problem is to assume that it is easy and see if any interesting algorithmic consequences would follow. In that spirit, we show the following.

Theorem 1.3 (Consequences of easiness of $\mathsf{GapMcpK^tP}$, informal)**.** *If, for some polynomial τ, $\mathsf{Gap}_{\tau,o(n)}\mathsf{McpK^tP} \in \mathsf{promise\text{-}P}$, then*

1. *$\mathsf{E} \not\subset \mathsf{io\text{-}SIZE}[2^{o(n)}]$, and*
2. *there is a deterministic polynomial-time algorithm for a version of Range Avoidance, where one is given a circuit $C\colon \{0,1\}^n \to \{0,1\}^{2n}$, and needs to find a $y \in \{0,1\}^{2n}$ such that $C(x) \neq y$ for all $x \in \{0,1\}^n$.*

Recall that deciding if $\mathsf{pK}^t(x \mid y) \leq s$ is equivalent to deciding if

$$\Pr_{r \in \{0,1\}^t}\left[\mathsf{K}^t(x \mid r, y) \leq s\right] \geq 2/3. \tag{3}$$

We consider a variant of this problem where (i) the input x is very short, $|x| \leq \log t$, and (ii) we just need to approximate the probability in (3) up to an additive error $1/8$. It is not hard to see that under these two restrictions, the resulting problem to approximate $\Pr_{r \in \{0,1\}^t}\left[\mathsf{K}^t(x \mid r, y) \leq s\right]$ is in $\mathsf{promise\text{-}BPP}$. (For any such short x, we can compute $\mathsf{K}^t(x \mid r, y)$ in deterministic $\mathsf{poly}(t)$ time by brute force. By randomly sampling enough r's, we can approximate the required probability in randomized time $\mathsf{poly}(t)$.) It turns out that this problem is $\mathsf{promise\text{-}BPP}$-complete (under polynomial-time Turing reductions).

Theorem 1.4 ($\mathsf{promise\text{-}BPP}$-complete problem)**.** *The problem to estimate, for given $x, y \in \{0,1\}^*$ and $s, t \in \mathbb{N}$ such that $|x| \leq \log t$, the probability*

$$\Pr_{r \in \{0,1\}^t}\left[\mathsf{K}^t(x \mid r, y) \leq s\right],$$

to within an additive error at most $1/8$, is $\mathsf{promise\text{-}BPP}$-complete.

The proof of Theorem 1.4 also yields a simple alternative proof of the recent result by [LPT24] showing that if one could derandomize Yao's "distinguisher-to-predictor" transformation, then $\mathsf{promise\text{-}BPP} = \mathsf{promise\text{-}P}$.

Theorems 1.3 and 1.4 are formally stated and proved in Section 7.

1.2 Techniques

To prove Theorems 1.1 and 1.2, we use the concept of a *natural property* for (conditional) K^t. A natural property for conditional K^t on n-bit strings with *usefulness* $s(n,t)$ is a predicate $\mathcal{P}$ such that: for all $x \in \{0,1\}^n$ and $y \in \{0,1\}^*$, if $\mathsf{K}^t(x \mid y) \leq s(n,t)$, then $\mathcal{P}(x,y,1^t) = 1$; and for all $y \in \{0,1\}^*$, $\mathcal{P}(x,y,1^t) = 0$ for at least $1/2$ of uniformly random $x \in \{0,1\}^n$. Thus, in a certain sense, a natural property distinguishes $s(n,t)$-easy n-bit strings (given y) from uniformly random ones. A natural property for conditional pK^t is defined analogously.

We first show that a chain rule for conditional pK^t (conditional K^t) is equivalent to the existence of an efficiently computable natural property for conditional K^t (and, in the case of the chain rule for conditional K^t, also the derandomization assumption that $\mathsf{E} \not\subset \mathsf{io\text{-}SIZE}[2^{o(n)}]$).

We then give "worst-case to average-case reductions" showing that computing a natural property for conditional K^t is equivalent to computing a corresponding gap version of $\mathsf{McpK^tP}$ (or, $\mathsf{McK^tP}$, under the assumption that $\mathsf{E} \not\subset \mathsf{io\text{-}SIZE}[2^{o(n)}]$). This implies the equivalence between chain rules and the worst-case complexity of the corresponding gap problems for $\mathsf{McpK^tP}$ ($\mathsf{McK^tP}$).

We give more details next.

Chain Rule from Natural Property. Here we sketch a proof that a natural property for conditional K^t implies a chain rule for conditional pK^t; the case of the chain rule for conditional K^t is proved similarly (using the derandomization assumption that $\mathsf{E} \not\subset \mathsf{io\text{-}SIZE}[2^{o(n)}]$).

While a natural property for conditional K^t is sufficient for all our purposes, let us assume for simplicity that we have a natural property for conditional pK^t. Using the fact that $\mathsf{pK}^t(x \mid y) \leq s$ is equivalent to $\mathsf{K}^t(x \mid y, r) \leq s$ for at least $2/3$ of random r's, we can adapt the argument sketched below to work with a natural property for conditional K^t instead; see Section 3 for details.

The proof of the Chain Rule for conditional pK^t from a natural property for conditional pK^t is a generalization of the proof argument from [Hir22; GK22; Gol+22]. It uses tools from pseudorandomness (e.g., a hybrid argument) as well as the list-decoding algorithm for Hadamard codes of [GL89]. The latter is used to define the Hadamard Code Direct Product Generator $\mathsf{DP}^x_k \colon \{0,1\}^{nk} \to \{0,1\}^k$ [Hir21]. For $z = (z^1, \ldots, z^k)$ with each $z^i \in \{0,1\}^n$, we define

$$\mathsf{DP}^x_k(z) = \langle z^1, x\rangle \ldots \langle z^k, x\rangle,$$

where $\langle z^i, x\rangle$ denotes the inner product of z^i and x modulo 2.

It can be shown that DP^x_k "encodes" any given string $x \in \{0,1\}^n$ into a distribution over k-bit strings (over all seeds z) with the following property: if this distribution can be distinguished from the uniform distribution by some efficient algorithm (distinguisher) D (with possibly some advice $\alpha \in \{0,1\}^*$), then $\mathsf{pK}^t(x \mid \alpha) \lessapprox k$. It follows that if we set k to be just slightly less than $\mathsf{pK}^t(x \mid \alpha)$, then DP^x_k will be a pseudorandom generator.

To prove the conditional pK^t Chain Rule for strings $x_1, \ldots, x_\ell, y$, we consider the concatenation of ℓ generators

$$\mathsf{DP}^{x_1}_{k_1}(z_1) \circ \cdots \circ \mathsf{DP}^{x_\ell}_{k_\ell}(z_\ell)$$

on independent seeds $z_1, \ldots, z_\ell$, where each $k_i \approx \mathsf{pK}^{t'}(x_i \mid y, x_1, \ldots, x_{i-1})$, for some $t' = \mathsf{poly}(t)$. By a hybrid argument, this choice of k_i's ensures that the concatenation of ℓ generators is a pseudorandom generator against efficient distinguishers with advice y. This means that a natural property for $\mathsf{pK}^t(- \mid y)$ is fooled by the generator. It follows that, for some choice of the seeds

$z_1, \ldots, z_\ell$, we have by the usefulness of the natural property that

$$\begin{aligned}\mathsf{pK}^t(\mathsf{DP}^{x_1}_{k_1}(z_1) \circ \cdots \circ \mathsf{DP}^{x_\ell}_{k_\ell}(z_\ell) \mid y, z_1, \ldots, z_\ell) &\gtrapprox \sum_{i=1}^{\ell} k_i \\ &\approx \sum_{i=1}^{\ell} \mathsf{pK}^{t'}(x_i \mid y, x_1, \ldots, x_{i-1}).\end{aligned}$$

On the other hand, for every choice of seeds $z_1, \ldots, z_\ell$ we have by the definition of DP^x_k that

$$\begin{aligned}\mathsf{pK}^t(\mathsf{DP}^{x_1}_{k_1}(z_1) \circ \cdots \circ \mathsf{DP}^{x_\ell}_{k_\ell}(z_\ell) \mid y, z_1, \ldots, z_\ell) &\lessapprox \mathsf{pK}^t(x_1, \ldots, x_\ell \mid y, z_1, \ldots, z_\ell) \\ &\leq \mathsf{pK}^t(x_1, \ldots, x_\ell \mid y)\end{aligned}$$

Combining the two inequalities above, we conclude the required Chain Rule:

$$\mathsf{pK}^t(x_1, \ldots, x_\ell \mid y) \gtrapprox \sum_{i=1}^{\ell} \mathsf{pK}^{t'}(x_i \mid y, x_1, \ldots, x_{i-1}).$$

Natural Property from Chain Rule. To derive a natural property for conditional pK^t from the Chain Rule for conditional pK^t, we proceed as follows. Given strings x, y, partition $x \in \{0,1\}^n$ into $\ell = n/(c \log t)$ strings $x_1, \ldots, x_\ell$ of length $c \log t$ each. If $\mathsf{pK}^t(x \mid y) \leq s(n)$, for some $s(n) < n$, then by the Chain Rule, so is the sum of conditional Kolmogorov complexities

$$\sum_{i=1}^{\ell} \mathsf{pK}^{\mathsf{poly}(t)}(x_i \mid y, x_1, \ldots, x_{i-1}),$$

where for simplicity of the presentation we ignore some additive error terms. Hence, by averaging, there is at least one $1 \leq i \leq \ell$ such that

$$\mathsf{pK}^{\mathsf{poly}(t)}(x_i \mid y, x_1, \ldots, x_{i-1}) \leq s(n)/\ell. \tag{4}$$

Since each x_i is of length $O(\log t)$, we can approximate the conditional time-bounded Kolmogorov complexity of x_i in Eq. (4) in randomized time $\mathsf{poly}(n, t)$.

The resulting randomized algorithm will accept (with high probability) all "easy" strings x. By a counting argument, we can also show that this algorithm is likely to reject at least $1/2$ of uniformly random strings x. Thus we get a natural property for pK^t, which is computable by a randomized polynomial-time algorithm. See Section 4 for details.

Natural Property for Conditional K^t vs. $\mathsf{GapMcK^tP}$. Using the idea of the "worst-case to average-case" reduction of [Hir18], combined with the use of DP generators as in [Hir20b; Hir20a], we can show an equivalence between the existence of an efficiently computable natural property for conditional K^t (with usefulness $n - o(n)$) and the existence of an efficient algorithm for solving $\mathsf{Gap}_{\tau, o(n)}\mathsf{McK^tP}$, for some polynomial τ (under the circuit lower bound assumption that $\mathsf{E} \not\subset \mathsf{io\text{-}SIZE}[2^{o(n)}]$). Such an equivalence allows us to relate the chain rules to the existence of efficient algorithms for solving $\mathsf{GapMcK^tP}$, yielding Theorem 1.1. (A similar argument yields also

Theorem 1.2, without using the circuit lower bound assumption.) We sketch the main idea next; see Section 2.4 for details.

The main idea used in the proof that a natural property implies a chain rule (sketched above) is that a given string x can be compressed to about k bits if the DP generator DP^x_k can be broken by some $\mathsf{poly}(t)$-time distinguisher.

Since the conditional K^t complexity of the output of the generator $\mathsf{DP}^x_k(z)$ (given a seed z and an auxiliary string y) is always at most $\mathsf{K}^t(x \mid y)$, a natural property for conditional K^t (with usefulness at least $\mathsf{K}^t(x \mid y)$) will always accept the outputs $\mathsf{DP}^x_k(z)$, for all z and y, no matter what k is.

In the other direction, if this natural property accepts $\mathsf{DP}^x_k(z)$, given z and y, with high probability (say, at least 0.6) over random z, then it distinguishes the outputs of $\mathsf{DP}^x_k(z)$ from the uniform distribution (because a natural property accepts at most $1/2$ of random strings). Thus it breaks the generator DP^x_k. It follows that $\mathsf{K}^{\mathsf{poly}(t)}(x \mid y) \leq k$.

These arguments can be used to imply that, for $k \approx s$, estimating the acceptance probability of the natural property on $(\mathsf{DP}^x_k(z), y)$, over random z, allows one to distinguish between pairs of strings (x, y) with $\mathsf{K}^t(x \mid y) \leq s$ and those with $\mathsf{K}^{\mathsf{poly}(t)}(x \mid y) \gg s$: the former pairs of strings will be accepted with probability 1, while the latter ones with probability at most 0.6.

Incompresible Strings from Chain Rules. Chain Rule for K^t implies an efficient algorithm to construct, for any $n, t \in \mathbb{N}$, for $n \leq t \leq 2^{o(n)}$, an *incompressible string* $z \in \{0,1\}^n$ with $\mathsf{K}^t(z) \geq n/2$, in time $\mathsf{poly}(n, t)$.

To find the required incompressible $z \in \{0,1\}^n$, for given $n, t \in \mathbb{N}$, proceed as follows.

> Set $t' = p(t)$, for the polynomial p from the Chain Rule for K^t. For a constant $c > 0$ to be determined, set $m = c \log t$ and $\ell = n/m$. Find, one after the other, by brute force, strings $w_1, w_2, \ldots, w_\ell \in \{0,1\}^m$ so that, for each $1 \leq i \leq \ell$, $\mathsf{K}^{t'}(w_i \mid w_1, \ldots, w_{i-1}) \geq m$. Output $z = w_1 w_2 \ldots w_\ell \in \{0,1\}^n$.

For the analysis, observe that such strings $w_1, \ldots, w_\ell$ exist by a counting argument, and each can be found by brute force in time $2^{O(m)} \cdot \mathsf{poly}(t') \leq \mathsf{poly}(t)$. By the Chain Rule for K^t,

$$\mathsf{K}^t(z_1, \ldots, z_\ell) \geq \ell \cdot m - \ell \cdot O(\log t) = (n/c \log t) \cdot (c \log t - O(\log t)),$$

which is at least $n/2$, for a sufficiently large constant $c > 0$. Hence $z = w_1 \ldots w_\ell$ is the required incompressible string, and it is constructed in time $\ell \cdot \mathsf{poly}(t) \leq \mathsf{poly}(n, t)$.

The constructed incompressible string z (for large enough polynomial t) can be shown to require large circuit size as well. Thus we get that a chain rule for conditional K^t also yields the circuit lower bound $\mathsf{E} \not\subset \mathsf{io\text{-}SIZE}[2^{o(n)}]$, completing the proof of the forward direction of Theorem 1.1. See Section 5 for details.

Similar reasoning also proves Item (1) of Theorem 1.3. The idea is that the assumption of Theorem 1.3 yields a chain rule for conditional pK^t, by Theorem 1.2. The latter can be used to construct an incompressible string, relative to the conditional pK^t complexity measure, piece by piece, as we constructed the string $z = w_1 \ldots w_\ell$ above. Here, when constructing each w_i of high conditional pK^t complexity, we use an assumed deterministic polynomial-time algorithm $\mathcal{B}$ for $\mathsf{GapMcpK^tP}$ to choose the lexicographically first log-length string rejected by $\mathcal{B}$.

The proof of Item (2) of Theorem 1.3 also follows since there is a close connection (in fact, equivalence) between solving the Range Avoidance problem and efficiently constructing a string x

of near-maximum conditional Kolmogorov complexity. The idea is that every string $x \in \{0,1\}^m$ in the range of a given circuit $C\colon \{0,1\}^n \to \{0,1\}^{2n}$ has $\mathsf{K}^t(x \mid C) \leq n < 2n$, for some $t = \mathsf{poly}(|C|)$. Thus any string $z \in \{0,1\}^{2n}$ with $\mathsf{K}^t(z \mid C) \approx 2n$ will be a solution to the Range Avoidance problem for the circuit C. See Section 7.4 for details.

Promise-BPP completeness. To prove Theorem 1.4, consider the generator

$$G^{x_1,\ldots,x_n}(z) := \mathsf{DP}_1^{x_1}(z) \circ \cdots \circ \mathsf{DP}_1^{x_n}(z),$$

where each $x_i \in \{0,1\}^m$ for $m = c \log n$ for some large constant $c > 1$, and also $z \in \{0,1\}^m$. That is, we use the same seed z for all n DP generators, with each generator $\mathsf{DP}_1^{x_i}(z)$ outputting just one bit (the inner product modulo 2 of z and x_i). So $G^{x_1,\ldots,x_n}(z)$ outputs n bits, using a seed z of $c \log n$ bits. We will explain how to choose the strings $x_1, \ldots, x_n$ to make this generator "good".

We want to use this $G^{x_1,\ldots,x_n}(z)$ to approximate the acceptance probability of any given circuit $C\colon \{0,1\}^n \to \{0,1\}$, to within an additive error $1/8$. This approximation problem is known to be promise-BPP-complete.

Suppose that $G^{x_1,\ldots,x_n}(z)$ fails to approximate the acceptance probability of some $C\colon \{0,1\}^n \to \{0,1\}$ to within additive error $1/8$. Then this circuit C is a distinguisher between $G^{x_1,\ldots,x_n}(z)$ and the uniform distribution over $\{0,1\}^n$. Using an argument similar to that in the "Chain Rule from Natural Property" paragraph above, we can show that for some $1 \leq i \leq n$,

$$\mathsf{pK}^{\mathsf{poly}(|C|)}(x_i \mid C, x_1, \ldots, x_{i-1}) < c' \log n,$$

for some constant $c' > 1$, independent of the constant c.

It follows that if we are able to select $x_1, \ldots, x_n$ so that, for each $1 \leq i \leq n$,

$$\mathsf{pK}^{\mathsf{poly}(|C|)}(x_i \mid C, x_1, \ldots, x_{i-1}) > c' \log n,$$

the resulting generator $G^{x_1,\ldots,x_n}(z)$ will "fool" this circuit C. (Such generators fooling a specific given circuit C are called *targeted* generators by [Gol11].)

Since $\mathsf{pK}^t(x \mid w) \leq s$ iff $\mathsf{K}^t(x \mid w, r) \leq s$ for at least $2/3$ of random r's, we can show that access to an oracle that approximates the probability

$$\Pr_{r \in \{0,1\}^t}\left[\mathsf{K}^t(x \mid w, r) \leq s\right]$$

allows us to construct the required strings $x_1, \ldots, x_n$, one by one, by trying, for each $1 \leq i \leq n$, all possible candidate strings for x_i in $\{0,1\}^m$. See Section 7.1 and Section 7.2 for more details. For the proof that derandomizing Yao's "distinguisher-to-predictor" construction would imply promise-BPP = promise-P, see Section 7.3.

1.3 Related Work

SoI for Time-Bounded Kolmogorov Complexity. Some of the first formal connections between SoI for K^t and computational complexity and cryptography were discovered in the 1990s. It was shown that SoI for K^t implies that there are no one-way functions [LM93; LW95], and that SoI for K^t is implied by the assumption $\mathsf{P} = \mathsf{NP}$ [LW95].

The gap between the necessary and sufficient conditions for SoI for K^t has been narrowed recently. Hirahara [Hir22] and, independently, Goldberg and Kabanets [GK22] improved the result

of [LW95] to show that SoI for K^t is implied by $\mathsf{DistNP} \subseteq \mathsf{AvgP}$, an average-case (errorless) version of the assumption that $\mathsf{NP} = \mathsf{P}$. The argument was extended by [Gol+22] to the case of SoI for pK^t, under the assumption that $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$.

Thus, SoI for K^t is sandwiched between two average-case assumptions: that any one-way function candidate can be efficiently inverted on average, and that NP is easy on average. Hirahara [Hir22] makes further progress toward closing the gap. He shows that, assuming E requires exponential-size circuits, SoI for K^t is sandwiched between the assumptions that there exists an *errorless* average-case heuristic scheme and that there exists an *error-prone* average-case heuristic scheme, both for computing K^t. Still, exact complexity-theoretic characterization of SoI for K^t is missing.

Lee and Romashchenko [LR05] proved that SoI for K^t implies that $\mathsf{EXP} \neq \mathsf{BPP}$. Perifel [Per07] showed that a certain version of SoI for conditional K^t implies that $\mathsf{EXP} \not\subset \mathsf{P/poly}$. The latter paper seems to be the first to study the Chain Rule hypothesis for conditional K^t, showing how to derive a weak version of the Chain Rule for K^t from a variant SoI hypothesis for K^t.

Unconditionally, Ronneburger [Ron04] proved that SoI does *not* hold for Levin's version of the time-bounded Kolmogorov complexity measure Kt.

Recently, SoI for K^t has been an important tool for, among others, worst-case to average-case reductions for problems in the polynomial-time hierarchy, computational learning, meta-complexity, and cryptography [Hir20b; Hir21; HN21; Hir22; GK22; CHV22; Gol+22; Hir+23; Ila23].

For the probabilistic version of time-bounded Kolmogorov complexity pK^t, [Hir+23] shows that a certain *average-case* version of SoI for pK^t (over polynomial-time samplable distributions) is equivalent to the non-existence of one-way functions. Note that this result is also an equivalence between a variant of SoI and a complexity (cryptography) assumption. The difference from the equivalences proved in our paper, in particular, our Theorem 1.2, is that we consider the *worst-case* version of a chain rule ("multi-string version of SoI") for conditional pK^t, and show it is equivalent to the *worst-case* complexity assumption about approximating conditional pK^t.

Complexity of computing Time-Bounded Kolmogorov Complexity measures. Hirahara [Hir18] shows worst-case to average-case reduction for $\mathsf{GapMINKT}$. For the conditional Kolmogorov complexity $\mathsf{K}^t(x \mid y)$, the corresponding minimization problem $\mathsf{McK^tP}$ is known to be NP-hard (under randomized reductions) in the *sublinear* time-bound regime when $t \ll |y|$ [Hir22; LP22b]; a sublinear-time version of computing conditional $\mathsf{pK}^t(x \mid y)$ is also NP-hard [LS24]. The problem $\mathsf{McK^tP}$ is shown to be NP-hard also for $t \geq |y|$, but only under the additional cryptographic assumption of the existence of secure witness encryption [HIR23]. Hirahara [Hir20b] shows that conditional $\mathsf{K}^{t,\mathsf{SAT}}$ (where the decoding Turing machine also has oracle access to SAT) is NP-hard to compute. For a random oracle $\mathcal{O}$, the oracle version $\mathsf{MK}^{t,\mathcal{O}}\mathsf{P}$ is also known to be NP-hard [Ila23]; see [Ila23] and the references therein for more information about the ongoing quest to prove NP-hardness of various meta-complexity problems such as MCSP (Minimum Circuit Size Problem) and $\mathsf{MK}^t\mathsf{P}$ (Minimum K^t complexity Problem).

While the worst-case complexity of $\mathsf{MK}^t\mathsf{P}$ is yet unknown, its average-case complexity (in the error-prone setting of average-case complexity) has been characterized by Liu and Pass [LP20] who showed that $\mathsf{MK}^t\mathsf{P}$ is hard on average if and only if one-way functions exist; this equivalence was later extended also to the case of $\mathsf{McK^tP}$ [LP22b]. More equivalences between one-way functions and meta-complexity are given in [IRS22; Hir23; LS24].

Natural properties. The concept of a *natural property* for circuit complexity was introduced by Razborov and Rudich [RR97] in the context of trying to understand the limitations of current proof techniques for showing strong circuit lower bounds. Roughly, a natural property for circuit size $S(n)$ is a predicate on inputs of size $N = 2^n$ that accepts all truth tables of n-input Boolean functions of circuit complexity at most $S(n)$ (all "easy" strings), and rejects at least $1/2$ of random N-bit strings; the first condition (of accepting "easy" strings) is called *usefulness*. It was shown in [RR97] that an efficiently computable natural property (of appropriate usefulness) yields an efficient algorithm for inverting well on average any given candidate one-way function, and so the existence of one-way functions would imply the non-existence of efficiently computable natural properties.

The concept became highly influential, and inspired a lot of research on circuit complexity, derandomization, computational learning, proof complexity, and meta-complexity, to name just a few. For instance, in the context of computational learning, the power of natural properties was explored in [Car+16; Car+17; OS17; Bin+22; Kar24]; in [GK23], stronger learning results were obtained from the natural properties for time-bounded Kolmogorov complexity such as K^t and KT. Natural properties were shown to imply various circuit lower bounds in, e.g., [KC00; IKW02; IKV23]. In [Hir+24], an assumed natural property for K^t and the additional assumption that $\mathsf{E} \not\subset \mathsf{io\text{-}SIZE}[2^{o(n)}]$, give efficient average-case algorithms for finding programs of size $\mathsf{K}^t(x)$ that generate x within time t (i.e., K^t *witnesses*), for strings x coming from any efficiently samplable distribution.

Range Avoidance Problem. The Range Avoidance Problem is related to the dual weak pigeonhole principle studied in the context of bounded arithmetic and propositional proof complexity [Kra01; Kra04; Jer04; Jeř07]. Motivated to identify natural search problems in the polynomial-time hierarchy, [Kle+21] studied the Range Avoidance Problem, for the case of circuits $C\colon \{0,1\}^n \to \{0,1\}^m$ with $m \geq n+1$, under the name 1-Empty. Korten [Kor21] showed that a polynomial-time algorithm for the Range Avoidance Problem would imply polynomial-time constructions for many important combinatorial objects (e.g., Ramsey graphs, extractors, rigid matrices, etc.). The Range Avoidance Problem for restricted circuit classes (and for different stretch regimes) was studied by [RSW22; GLW22; Gaj+23]. It is shown in [ILW23] that, under certain cryptographic assumptions, a polynomial-time algorithm for the Range Avoidance Problem (even for a polynomial stretch regime) would imply that $\mathsf{NP} = \mathsf{coNP}$; this result provides some evidence that the Range Avoidance problem may be intractable.

2 Preliminaries

2.1 Kolmogorov Complexity

Let U be a Turing machine. For $t \in \mathbb{N}$ and $x, y \in \{0,1\}^*$, we define *t-time-bounded Kolmogorov complexity of x* given y (with respect to U) as

$$\mathsf{K}^t_U(x \mid y) = \min_{p \in \{0,1\}^*} \Big\{ |p| \;\mid\; U(p,y) \text{ outputs } x \text{ in at most } t \text{ steps} \Big\}.$$

We assume that the string y is given on a separate input tape. As usual, we fix U to be a time-optimal machine [LV19], i.e., a universal machine that is almost as fast and length efficient as any other universal machine, and drop the index U when referring to time-bounded Kolmogorov

complexity measures. We use $\mathsf{K}(x \mid y)$ to denote the (time-unbounded) Kolmogorov complexity of x given y. When $y = \epsilon$ (i.e., y is the empty string), we drop the conditioning on y, getting the definition of the (time-bounded) Kolmogorov complexity of x.

Next we recall the definitions of the randomized time-bounded Kolmogorov complexity pK^t; see [LO22] for more information on randomized time-bounded Kolmogorov complexity variants and their applications.

For $\lambda \in [0,1]$ and $t \in \mathbb{N}$, we define *t-time-bounded probabilistic Kolmogorov complexity of x* given y as

$$\mathsf{pK}^t_\lambda(x \mid y) = \min\left\{k \;\middle|\; \mathbf{Pr}_{r\sim\{0,1\}^t}[\exists p \in \{0,1\}^k, U(p,r,y) \text{ outputs } x \text{ in at most } t \text{ steps}] \geq \lambda\right\}.$$

Equivalently, $\mathsf{pK}^t_\lambda(x \mid y) \leq s$ if and only if

$$\Pr_{r\in\{0,1\}^t}\left[\mathsf{K}^t(x \mid r, y) \leq s\right] \geq \lambda.$$

We assume that the random string r is given on a separate input tape. For simplicity, we omit λ when $\lambda = 2/3$.

Lemma 2.1 (Probabilistic Incompressibility [Gol+22])**.** *For any string $y \in \{0,1\}^*$, time bound $t \in \mathbb{N}$ (including $t = \infty$), $0 < \lambda \leq 1$, and $k \in \mathbb{N}$, we have*

$$\mathbf{Pr}_{x\in\{0,1\}^n}\left[\mathsf{pK}^t_\lambda(x \mid y) \leq n - k\right] \leq \frac{(2/\lambda)}{2^k}.$$

2.2 Generators

Let $\mathcal{U}_n$ denote the uniform distribution over n-bit strings. For a generator $G\colon \{0,1\}^\ell \to \{0,1\}^n$, a (randomized) algorithm $D\colon \{0,1\}^n \to \{0,1\}$, and $0 < \varepsilon \leq 1$, we say that D *ε-distinguishes* $G(\mathcal{U}_\ell)$ from $\mathcal{U}_n$ if

$$\left|\mathbf{Pr}_{z\in\{0,1\}^\ell, D}[D(G(z)) = 1] - \mathbf{Pr}_{y\in\{0,1\}^n, D}[D(y) = 1]\right| \geq \varepsilon.$$

Otherwise, we say that the generator G *ε-fools* D. We call a generator G *an ε-pseudorandom generator (PRG)* for a class $\mathcal{C}$ of Boolean functions, if G ε-fools every $D \in \mathcal{C}$.

Theorem 2.2 ([NW94; IW97])**.** *Assume $\mathsf{E} \not\subset \mathsf{io\text{-}SIZE}[2^{o(n)}]$. Then, for any size parameter $s(n)$ and error ε, there is an explicit construction of a generator $G\colon \{0,1\}^\ell \to \{0,1\}^n$ that is an ε-PRG for the class of n-input Boolean circuits of size at most $s(n)$, with the seed length $\ell \leq O(\log(s(n)/\varepsilon))$, that can be evaluated on any given seed in time $\mathsf{poly}(s(n)/\varepsilon)$.*

Definition 2.3 (Hadamard Direct Product Generator (DP) [Hir21])**.** *For $k, n \in \mathbb{N}$ and any given $x \in \{0,1\}^n$, we define the k-*wise direct product of the Hadamard encoding of *x to be the generator*

$$\mathsf{DP}^x_k\colon \{0,1\}^{nk} \to \{0,1\}^k$$

such that

$$\mathsf{DP}^x_k(z_1, \ldots, z_k) := (x \cdot z_1, \ldots, x \cdot z_k),$$

where each $z_i \in \{0,1\}^n$, for $1 \leq i \leq k$, and $x \cdot z$ denotes the inner product of x and z modulo 2. We define the strong version of the DP Generator as

$$\mathsf{s\text{-}DP}^x_k\colon \{0,1\}^{nk} \to \{0,1\}^{nk+k}$$

where $\mathsf{s\text{-}DP}^x_k(z_1,\ldots,z_k) = z_1,\ldots,z_k,\mathsf{DP}^x_k(z_1,\ldots,z_k)$. [4]

Lemma 2.4 (K^t Reconstruction [Hir21])**.** *Assume* $\mathsf{E} \not\subseteq \mathsf{io\text{-}SIZE}[2^{o(n)}]$. *For* $\varepsilon > 0$, $x \in \{0,1\}^n$, *and* $k \in \mathbb{N}$ *satisfying* $k \le 2n$, *let* D *be a randomized algorithm that takes an advice string* β *and runs in time* t_D *such that* D ε*-distinguishes* $\mathsf{s\text{-}DP}^x_k(\mathcal{U}_{nk})$ *from* $\mathcal{U}_{nk+k}$. *Then there is a polynomial* p_{DP} *such that*

$$\mathsf{K}^{p_{\mathsf{DP}}(nt_D/\varepsilon)}(x \mid \beta) \le k + \log p_{\mathsf{DP}}(nt_D/\varepsilon).$$

Lemma 2.5 (pK^t Reconstruction [Gol+22])**.** *For* $\varepsilon > 0$, $x \in \{0,1\}^n$, *and* $k \in \mathbb{N}$ *satisfying* $k \le 2n$, *let* D *be a randomized algorithm that takes an advice string* β *and runs in time* t_D *such that* D ε*-distinguishes* $\mathsf{s\text{-}DP}^x_k(\mathcal{U}_{nk})$ *from* $\mathcal{U}_{nk+k}$. *Then there is a polynomial* p_{DP} *such that*

$$\mathsf{pK}^{\widetilde{O}(t_D)\cdot p_{\mathsf{DP}}(n/\varepsilon)}(x \mid \beta) \le k + O(\log(n/\varepsilon)).$$

Remark 2.6. *An extra dependence on* t_D *in the Kolmogorov description size of* x *in Lemma 2.4 is due to the derandomization of a distingsuisher* D *and the Goldreich-Levin list-decoding algorithm for Hadamard codes [GL89], using the PRG from Theorem 2.2; some seeds of this generator, of size* $O(\log(nt_D/\varepsilon))$, *are added to the Kolmogorov description of* x.

2.3 Natural Properties

We recall the definition of a natural property for circuit size. For a truth table $x \in \{0,1\}^N$, for $N = 2^n$, we denote by $\mathsf{size}(x)$ the size of a smallest Boolean circuit that computes the n-input Boolean function with the truth table x.

Definition 2.7 (Natural Property for Circuit Size [RR97])**.** *A* natural property for circuit size *for truth tables of length* $N = 2^n$ *with* usefulness $s(N) < N$ *is a predicate* $\mathcal{R}\colon \{0,1\}^N \to \{0,1\}$ *such that*

1. *for all* $x \in \{0,1\}^N$, *if* $\mathsf{size}(x) \le s(N)$, *then* $\mathcal{R}(x) = 1$, *and*
2. $\mathbf{Pr}_{x\in\{0,1\}^N}[\mathcal{R}(x) = 0] \ge 1/2$.

Such a natural property is called BPP-computable *if there is a randomized polynomial-time algorithm* $\mathcal{A}$ *such that*

1. *for all* $x \in \{0,1\}^N$, *if* $\mathsf{size}(x) \le s(N)$, *then* $\mathbf{Pr}_{\mathcal{A}}[\mathcal{A}(x) = 1] \ge 0.9$, *and*
2. $\mathbf{Pr}_{x\in\{0,1\}^N}[\mathbf{Pr}_{\mathcal{A}}[\mathcal{A}(x) = 0] \ge 0.9] \ge 1/2$.

We need the following result of [IKV23]. We use a common definition of the class BPP with advice, where we say that $L \in \mathsf{BPP}/a(n)$, for some advice size $a(n)$ for input length n, if there is a probabilistic polynomial-time Turing machine M such that, for inputs x of length n and for some *good* advice $\alpha \in \{0,1\}^{a(n)}$, $M(x,\alpha)$ either accepts or rejects every given $x \in \{0,1\}^n$ with probability at least $2/3$ over its internal randomness. Note that there is no such acceptance probability guarantee for any different advice string $\alpha' \neq \alpha$.

[4] In [Hir21], the output of the DP generator was defined to also include the seed $z^1,\ldots,z^k$. We call that version of the DP generator a *strong* DP Generator.

Theorem 2.8 ([IKV23])**.** *Let $\mathcal{R}$ be a natural property for circuit size for truth tables of length $N = 2^n$ with usefulness $s(N) \geq N^{\varepsilon}$, for some $0 < \varepsilon < 1$. Then*

1. $\mathsf{ZPEXP}^{\mathcal{R}} \not\subset \mathsf{P/poly}$*, and*
2. $\mathsf{ZPP}^{\mathcal{R}}/1 \not\subset \mathsf{SIZE}[n^k]$ *and* $\mathsf{promise\text{-}ZPP}^{\mathcal{R}} \not\subset \mathsf{SIZE}[n^k]$ *for all $k \in \mathbb{N}$.*

Replacing a natural property oracle $\mathcal{R}$ with an efficient randomized version, we immediately get from Theorem 2.8 the following.

Corollary 2.9 (implicit in [IKV23])**.** *Suppose there is a* BPP*-computable natural property for circuit size for truth tables of length $N = 2^n$ with usefulness $s(N) \geq N^{\varepsilon}$, for some $0 < \varepsilon < 1$. Then*

1. $\mathsf{BPEXP} \not\subset \mathsf{P/poly}$*, and*
2. $\mathsf{BPP}/1 \not\subset \mathsf{SIZE}[n^k]$ *and* $\mathsf{promise\text{-}BPP} \not\subset \mathsf{SIZE}[n^k]$ *for all $k \in \mathbb{N}$.*

Proof. A natural oracle $\mathcal{R}$ is only used as a distinguisher between a distribution over "easy" strings and the uniform distribution. A BPP-computable natural property gives rise to a randomized such distinguisher, which is not required to satisfy the BPP-promise (of rejecting or accepting with high probability) on all inputs. The base ZPEXP algorithm will run a randomized algorithm for a BPP-computable natural property to simulate its oracle access to $\mathcal{R}$, becoming a BPEXP algorithm in item (1), and a BPP/1 or promise-BPP algorithm in item (2). □

We define a natural property for K^t by analogy with the natural property for circuit size above.

Definition 2.10 (Natural Property for K^t)**.** *A* P-computable natural property *for K^t on n-bit inputs, with* usefulness *$s(n,t)$, for some $s(n,t) < n$, is a polynomial-time algorithm $\mathcal{A}$ satisfying the following: For some polynomial p, we have for all sufficiently large $n \in \mathbb{N}$ and $t \geq p(n)$ that*

1. *for all $x \in \{0,1\}^n$, if $\mathsf{K}^t(x) \leq s(n,t)$, then $\mathcal{A}(x, 1^t) = 1$, and*
2. $\mathbf{Pr}_{x \in \{0,1\}^n}\left[\mathcal{A}(x, 1^t) = 0\right] \geq 1/2$.

Such a natural property is said to be BPP-computable *if there is a randomized polynomial-time algorithm $\mathcal{A}$ such that, for some polynomial p, we have for all large $n \in \mathbb{N}$ and $t \geq p(n)$ that*

1. *for all $x \in \{0,1\}^n$, if $\mathsf{K}^t(x) \leq s(n,t)$, then $\mathbf{Pr}_{\mathcal{A}}[\mathcal{A}(x, 1^t) = 1] \geq 0.9$, and*
2. $\mathbf{Pr}_{x \in \{0,1\}^n}\left[\mathbf{Pr}_{\mathcal{A}}[\mathcal{A}(x, 1^t) = 0] \geq 0.9\right] \geq 1/2$.

We also define a natural property for *conditional* K^t as follows.

Definition 2.11 (Natural Property for Conditional K^t)**.** *A* natural property for conditional K^t on n-bit strings with usefulness *$s(n,t)$ is a predicate $\mathcal{P}\colon \{0,1\}^* \times \{0,1\}^* \times 1^* \to \{0,1\}$ satisfying the following: For some polynomial p, we have for all large $n, m \in \mathbb{N}$ and $t \geq p(n+m)$ that*

1. *for all $x \in \{0,1\}^n$ and $y \in \{0,1\}^m$, if $\mathsf{K}^t(x \mid y) \leq s(n,t)$, then $\mathcal{P}(x, y, 1^t) = 1$, and*
2. *for all $y \in \{0,1\}^m$, $\mathbf{Pr}_{x \in \{0,1\}^n}[\mathcal{P}(x, y, 1^t) = 0] \geq 1/2$.*

A BPP*-computable natural property for conditional K^t is defined similarly to that for K^t above.*

Below we consider natural properties with usefulness $n - \delta(n,t)$, for monotone non-decreasing functions $\delta\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$. When $\delta(n,t)$ depends only on one of its inputs, we omit the mention of the other input, and think of δ as a function on one input only.

2.4 Approximating Time-Bounded Kolmogorov Complexity

$\mathsf{MK}^t\mathsf{P}$ (also denoted as MINKT) is the problem to determine if $\mathsf{K}^t(x) \leq s$, for given positive integers t and s. We recall the definition of the gap version of $\mathsf{MK}^t\mathsf{P}$.

Definition 2.12 ([Ko91; Hir20a])**.** *For a polynomial τ and a function $\delta\colon \mathbb{N}\times\mathbb{N}\to\mathbb{N}$, $\mathsf{Gap}_{\tau,\delta}\mathsf{MK}^t\mathsf{P}$ is the following promise problem:*

$$\Pi_{\text{YES}} = \{(x, 1^s, 1^t) \mid \mathsf{K}^t(x) \leq s\},$$
$$\Pi_{\text{NO}} = \{(x, 1^s, 1^t) \mid \mathsf{K}^{\tau(t)}(x) > s + \delta(|x|, t)\}.$$

Next we consider the Minimum Conditional Time-Bounded Kolmogorov Complexity Problem, denoted $\mathsf{McK}^\mathsf{t}\mathsf{P}$, where

$$\mathsf{McK}^\mathsf{t}\mathsf{P} = \{(x, y, 1^s, 1^t) \mid \mathsf{K}^t(x \mid y) \leq s\}.$$

Definition 2.13. *For a polynomial τ and a function $\delta\colon \mathbb{N}\times\mathbb{N}\to\mathbb{N}$, define $\mathsf{Gap}_{\tau,\delta}\mathsf{McK}^\mathsf{t}\mathsf{P}$ as the following promise problem:*

$$\Pi_{\text{YES}} = \{(x, y, 1^s, 1^t) \mid \mathsf{K}^t(x \mid y) \leq s\},$$
$$\Pi_{\text{NO}} = \{(x, y, 1^s, 1^t) \mid \mathsf{K}^{\tau(t)}(x \mid y) > s + \delta(|x|, t)\}.$$

Lemma 2.14. $\mathsf{Gap}_{\tau,\delta}\mathsf{McK}^\mathsf{t}\mathsf{P} \in \mathsf{promise\text{-}P}$ *iff there is a polynomial-time algorithm $\tilde{K}$ such that, for all $x \in \{0,1\}^n$ and $y \in \{0,1\}^m$ and every large enough integer t,*

$$\mathsf{K}^{\tau(t)}(x \mid y) - \delta(n, t) \leq \tilde{K}(x, y, 1^t) \leq \mathsf{K}^t(x \mid y). \tag{5}$$

Proof. In the forward direction, define $\tilde{K}(x, y, 1^t)$ to be the smallest integer $0 \leq s \leq 2|x|$ such that the assumed polynomial-time algorithm for $\mathsf{promise\text{-}Gap}_{\tau,\delta}\mathsf{McK}^\mathsf{t}\mathsf{P}$ outputs "yes" on input $(x, y, 1^s, 1^t)$. Since for this s, we have $(x, y, 1^s, 1^t) \notin \Pi_{\text{NO}}$, we conclude the required lower bound on $\tilde{K}(x, y, 1^t)$ in (5). For the required upper bound on $\tilde{K}(x, y, 1^t)$ in (5), observe that we will get a "yes" for some $s \leq \mathsf{K}^t(x \mid y)$, because we definitely get a "yes" when $s = \mathsf{K}^t(x \mid y)$.

In the reverse direction, define an algorithm for $\mathsf{Gap}_{\tau,\delta}\mathsf{McK}^\mathsf{t}\mathsf{P}$ as follows: "On input $(x, y, 1^s, 1^t)$, accept iff $\tilde{K}(x, y, 1^t) \leq s$." By the definition of $\tilde{K}$, this algorithm will accept all instances in Π_{YES}, and reject all instances in Π_{NO}. □

A natural property for conditional K^t is closely related to $\mathsf{GapMcK}^\mathsf{t}\mathsf{P}$.

Lemma 2.15. *If, for some polynomial τ and a function $\delta\colon \mathbb{N}\to\mathbb{N}$, $\mathsf{Gap}_{\tau,\delta}\mathsf{McK}^\mathsf{t}\mathsf{P} \in \mathsf{promise\text{-}P}$, then there is a P-computable natural property for conditional $\mathsf{K}^t(x \mid y)$ on n-bit strings x with usefulness $s(n, t) = n - \delta(n, t) - 2$.*

Proof. Let $\mathcal{K}$ be a polynomial-time algorithm for $\mathsf{Gap}_{\tau,\delta}\mathsf{McK}^\mathsf{t}\mathsf{P}$. For $s(n, t) = n - \delta(n, t) - 2$, define an algorithm $\mathcal{A}$ as follows: "On input $(x, y, 1^t)$, accept iff $\mathcal{K}(x, y, 1^{s(|x|,t)}, 1^t)$ accepts."

For correctness, observe that all $x \in \{0,1\}^n$ and $y \in \{0,1\}^m$ with $\mathsf{K}^t(x \mid y) \leq s(n, t)$ are accepted because $(x, y, 1^{s(|x|,t)}, 1^t) \in \Pi_{\text{YES}}$. For every y, we have by counting that, for at least $1/2$ of random $x \in \{0,1\}^n$, $\mathsf{K}(x \mid y) > n - 2$. Hence, for these random x, $\mathsf{K}^{\tau(t)}(x \mid y) \geq \mathsf{K}(x \mid y) > n - 2 = s(n, t) + \delta(n, t)$, and so $(x, y, 1^{s(|x|,t)}, 1^t) \in \Pi_{\text{NO}}$. □

Lemma 2.16. *Assume that* $\mathsf{E} \not\subset \mathsf{io\text{-}SIZE}[2^{o(n)}]$. *If there is a* P*-computable natural property for conditional* $\mathsf{K}^t(x \mid y)$ *on* n*-bit strings* x *with usefulness* $s(n,t) = n - \delta(n,t)$*, then for some polynomial* τ*, we have* $\mathsf{Gap}_{\tau,\delta'}\mathsf{McK^tP} \in \mathsf{promise\text{-}P}$*, where* $\delta'(n,t) = \delta(2n,2t) + O(\log(nt))$.

Proof. For this "worst-case to average-case" reduction, we rely on the ideas from [Hir18; Hir20a; Hir20b]. Let $\mathcal{A}(x,y,1^t)$ be a polynomial-time $p(n,t)$ algorithm for the assumed natural property for $\mathsf{K}^t(x \mid y)$ for n-bit strings x. For given $x \in \{0,1\}^n$ and $y \in \{0,1\}^m$, consider the generator

$$\mathsf{DP}^x_k \colon \{0,1\}^{nk} \to \{0,1\}^k$$

from Definition 2.3, for some $0 \le k \le 2n$ to be determined. We have, for any t larger than some polynomial in n, and for any seed $z \in \{0,1\}^{nk}$, that

$$\mathsf{K}^{2t}(\mathsf{DP}^x_k(z) \mid z, y) \le \mathsf{K}^t(x \mid y) + c \log n, \tag{6}$$

for some constant $c > 0$. For any given $\sigma \in \mathbb{N}$, if $\mathsf{K}^t(x \mid y) \le \sigma$, then for $k = \sigma + c\log n + \delta(2n,2t)$, we have by (6) that

$$\begin{aligned} \mathsf{K}^{2t}(\mathsf{DP}^x_k(z) \mid z, y) &\le \sigma + c \log n \\ &\le k - \delta(k, 2t) \\ &= s(k, 2t). \end{aligned}$$

So, $\mathcal{A}(\mathsf{DP}^x_k(z), z \circ y, 1^{2t})$ accepts with probability 1 over $z \in \{0,1\}^{nk}$ (by the definition of a natural property with usefulness $s(k,2t)$).

On the other hand, for every y and z, $\mathcal{A}(w, z \circ y, 1^{2t})$ rejects at least $1/2$ of random $w \in \{0,1\}^k$. Hence, $\mathcal{A}(w, z \circ y, 1^{2t})$ rejects with probability at least $1/2$ over uniformly random independent $w \in \{0,1\}^k$ and $z \in \{0,1\}^{nk}$.

If it were the case that $\mathcal{A}(\mathsf{DP}^x_k(z), z \circ y, 1^{2t})$ accepts with probability at least 0.6 over $z \in \{0,1\}^{nk}$, then $\mathcal{A}(-, y, 1^{2t})$ would be a (0.1)-distinguisher between $\mathsf{s\text{-}DP}^x_k$ and $\mathcal{U}_{nk+k}$. Hence, by Lemma 2.4, we would get that

$$\begin{aligned} \mathsf{K}^{p(t)}(x \mid y) &\le k + O(\log(nt)) \\ &= \sigma + c \log n + \delta(2n,2t) + O(\log(nt)) \\ &= \sigma + \delta(2n,2t) + O(\log(nt)) \\ &= \sigma + \delta'(n,t), \end{aligned}$$

for some polynomial p. This means that $(x, y, 1^\sigma, 1^t)$ is not in Π_{NO} of $\mathsf{Gap}_{\tau,\delta'}\mathsf{McK^tP}$ for $\tau = p$. In other words, for every instance $(x, y, 1^\sigma, 1^t)$ in Π_{NO} of $\mathsf{Gap}_{\tau,\delta'}\mathsf{McK^tP}$, $\mathcal{A}(\mathsf{DP}^x_k(z), z \circ y, 1^{2t})$ accepts with probability less than 0.6 over random $z \in \{0,1\}^{nk}$.

We conclude that the following algorithm correctly solves $\mathsf{Gap}_{\tau,\delta'}\mathsf{McK^tP}$:

> On input $(x, y, 1^\sigma, 1^t)$, set $k = \sigma + c\log n + \delta(2n,2t)$. Estimate, to within an additive error 0.1, the probability over $z \in \{0,1\}^{nk}$ that $\mathcal{A}(\mathsf{DP}^x_k(z), z \circ y, 1^{2t})$ accepts. If the estimated probability is at least 0.9, then accept; otherwise, reject.

Using the PRG of Theorem 2.2, we can make the algorithm above run in deterministic polynomial time, which concludes the proof. □

Combining Lemma 2.15 and Lemma 2.16, we immediately get the following.

Corollary 2.17. *Assume* $\mathsf{E} \not\subset \mathsf{io\text{-}SIZE}[2^{o(n)}]$. *The following are equivalent:*

- *For any* $C > 1$, *there is a polynomial* τ *such that* $\mathsf{Gap}_{\tau,\delta}\mathsf{McK^tP} \in \mathsf{promise\text{-}P}$, *for* $\delta(n,t) \le n/C$.
- *For any* $D > 1$, *there is a* P*-computable natural property for conditional* $\mathsf{K}^t(x \mid y)$ *on* n*-bit strings* x *with usefulness* $s(n,t) = n - n/D$.

We can similarly define the gap version of the minimum conditional pK^t measure, $\mathsf{Gap}_{\tau,\delta}\mathsf{McpK^tP}$.

Definition 2.18. *For a polynomial* τ *and a function* $\delta\colon \mathbb{N}\times\mathbb{N}\to\mathbb{N}$, *define* $\mathsf{Gap}_{\tau,\delta}\mathsf{McpK^tP}$ *as the following promise problem:*

$$\Pi_{\text{YES}} = \{(x,y,1^s,1^t) \mid \mathsf{pK}^t(x \mid y) \le s\},$$
$$\Pi_{\text{NO}} = \{(x,y,1^s,1^t) \mid \mathsf{pK}^{\tau(t)}(x \mid y) > s + \delta(|x|,t)\}.$$

We get the following.

Corollary 2.19. *The following are equivalent:*

- *For any* $C > 1$, *there is a polynomial* τ *such that* $\mathsf{Gap}_{\tau,\delta}\mathsf{McpK^tP} \in \mathsf{promise\text{-}BPP}$, *for* $\delta(n,t) \le n/C$.
- *For any constant* $D > 1$, *there is a* BPP*-computable natural property for conditional* $\mathsf{K}^t(x \mid y)$ *on* n*-bit strings* x *with usefulness* $s(n,t) = n - n/D$.

Proof sketch. By similar arguments, we can prove analogs of Lemma 2.15 and Lemma 2.16 for the case of $\mathsf{Gap}_{\tau,\delta}\mathsf{McpK^tP}$ and BPP-computable natural properties for conditional K^t. For the analog of Lemma 2.16, we do not assume $\mathsf{E} \not\subset \mathsf{io\text{-}SIZE}[2^{o(n)}]$, and use Lemma 2.5. □

2.5 Range Avoidance

Definition 2.20 (Avoid)**.** *The Range Avoidance Problem [Kle+21; Kor21] is the following search problem: Given a circuit* $C\colon \{0,1\}^n \to \{0,1\}^m$, *for some* $m > n$, *find a string* $z \in \{0,1\}^m$ *such that, for all* $x \in \{0,1\}^n$, $C(x) \neq z$. *Let* Avoid *be the Range Avoidance problem for circuits* $C\colon \{0,1\}^n \to \{0,1\}^m$ *with* $m > (1+\varepsilon)n$ *for some (arbitrarily small) constant* $\varepsilon > 0$ *(e.g., think* $m > (1.01)\cdot n$*).*

Remark 2.21. *The complexity of the Range Avoidance Problem appears to be sensitive to the assumed stretch of circuits* $C\colon \{0,1\}^n \to \{0,1\}^m$. *[Kle+21; Kor21] considered the version with* $m \ge n+1$, *and showed that* $m = n+1$ *and* $m = 2n$ *are equivalent under* P^{NP}*-reductions. Such an equivalence is not known under* P*-reductions. A similar issue arises also in the context of bounded arithmetic where one seems to need different variants of the dual weak pigeonhole principle axiom for different systems of randomized polynomial-time reasoning (cf. [Jeř07]). In the present paper, we only consider the version of Range Avoidance with at least linear stretch.*

For every $y \in \{C(x) \mid x \in \{0,1\}^n\}$, we can reconstruct y if we know some pre-image $x \in \{0,1\}^n$ such that $C(x) = y$. Therefore, $\mathsf{K}^t(y \mid C) \le n$, for $t = p(|C|)$, where p is the polynomial bound on the time required to evaluate a given circuit C on a given input. Hence, to solve Avoid for a given circuit C, it suffices to find a string of high conditional Kolmogorov complexity $\mathsf{K}^t(- \mid C)$.

In fact, as observed in [RSW22], the two tasks are polynomial-time equivalent: one can solve Avoid in polynomial time iff one can find "incompressible" strings for conditional K^t.[5]

Actually, finding a string of high conditional pK^t complexity allows us to solve a natural generalization of the Range Avoidance problem, which we term Collective Range Avoidance (CRA).

Definition 2.22 (Collective Range Avoidance (CRA))**.** *The* 1/3-Collective Range Avoidance (CRA) *problem is the following search problem: Given a circuit* $C\colon \{0,1\}^\ell \times \{0,1\}^n \to \{0,1\}^m$ *for* $m > (1+\varepsilon)n$ *for some constant* $\varepsilon > 0$*, find a string* $z \in \{0,1\}^m$ *such that, for at least* $1/3$ *of strings* $r \in \{0,1\}^\ell$*, the string* z *is not in the range of the circuit* $C_r\colon \{0,1\}^n \to \{0,1\}^m$ *where* $C_r(x) = C(r,x)$*.*

Note that the usual Range Avoidance problem is a special case of the CRA problem defined above when $\ell = 0$.

Consider any string $y \in \{0,1\}^m$ that is in the range of at least $2/3$ of circuits $C_r\colon \{0,1\}^n \to \{0,1\}^m$, over uniformly random $r \in \{0,1\}^\ell$. Then, with probability at least $2/3$ over $r \in \{0,1\}^\ell$, there is a string $x_r \in \{0,1\}^n$ such that $C(r,x_r) = y$. This implies that $\mathsf{pK}^t(y \mid C) \leq n$, for some $t = \mathsf{poly}(|C|)$ (the time needed to evaluate the circuit C on a given input). Thus, to solve 1/3-CRA problem for a given circuit $C\colon \{0,1\}^\ell \times \{0,1\}^n \to \{0,1\}^m$, it suffices to find a string $z \in \{0,1\}^m$ with $\mathsf{pK}^t(z \mid C) > n$, for some polynomial $t = \mathsf{poly}(|C|)$.

For completeness, we state the following observation; the proof similar to the one in [RSW22] for the case of unary Range Avoidance.

Lemma 2.23. *The search problem* (1/3)-CRA *is in polynomial time if and only if, for every constant* $0 < \gamma < 1$*, there is a polynomial-time algorithm that, given* 1^m*,* 1^t*, and* $y \in \{0,1\}^\ell$*, finds a string* $z \in \{0,1\}^m$ *with* $\mathsf{pK}^t(z \mid y) \geq (1-\gamma)\cdot m$*.*

Proof. In the forward direction, given 1^m, 1^t, $y \in \{0,1\}^\ell$, and $0 < \gamma < 1$, set $n = \lceil \alpha \cdot m \rceil$, for some constant $0 < \alpha < 1$ to be determined, define a circuit $C_y\colon \{0,1\}^t \times \{0,1\}^n \to \{0,1\}^m$, which simulates the following Turing machine:

> On inputs $r \in \{0,1\}^t$ and $x \in \{0,1\}^n$, decompose $x = uv$ where $|u| = \log n$. Let $0 \leq d < n$ be the integer encoded by the binary string u. Let w be the last $\min\{d, n - \log n\}$ bits of x. Run the universal Turing machine $U(w,r,y)$ for at most t steps, and output the binary string it outputs, padded or trimmed to be exactly of length m.

Note that C_y is of size $\mathsf{poly}(n,\ell,t)$. Applying an assumed polynomial-time algorithm for 1/3-CRA to C_y, we get a string $z \in \{0,1\}^m$ such that, for all sufficiently large $m \in \mathbb{N}$,

$$\begin{aligned}\mathsf{pK}^t(z \mid y) &> n - \log n \\ &\geq \alpha \cdot m - \log m \\ &\geq (1-\gamma)\cdot m,\end{aligned}$$

if we set $\alpha = 1 - (\gamma/2)$.

In the reverse direction, For a given circuit $C\colon \{0,1\}^\ell \times \{0,1\}^n \to \{0,1\}^m$, with $m > (1+\varepsilon)n$ for some constant $\varepsilon > 0$, consider any string $z \in \{0,1\}^m$ with

$$\mathsf{pK}^t(z \mid C) \geq (1-\nu)m,$$

[5] Actually, [RSW22] observes this equivalence for the case of unary Avoid and K^t, but it immediately generalizes to the case of Avoid and conditional K^t.

where $t = p(|C|)$ for some polynomial p, and $\nu = \varepsilon/(1+\varepsilon)$. By our choice of parameters, we get that $\mathsf{pK}^t(z \mid C) > n$. Hence, by our earlier observation, the string z is a solution to 1/3-CRA, and by our assumption, such a z can be found in polynomial time. □

3 Chain Rules from Natural Properties

3.1 Case of K^t

Theorem 3.1 (Chain Rule for K^t)**.** *Suppose that*

- $\mathsf{E} \not\subset \mathsf{io\text{-}SIZE}[2^{o(n)}]$*, and*
- *there is a* P*-computable natural property* $\mathcal{A}(w, 1^t)$ *for* K^t *on* n*-bit inputs* w *with usefulness* $s(n,t) = n - \delta(n,t)$*.*

Then there exist constants $c_0, c_1 \in \mathbb{N}$ *such that, for all sufficiently large* $x_1, \ldots, x_\ell \in \{0,1\}^*$ *of lengths* $n_1, \ldots, n_\ell$*, respectively, for any* $\ell \in \mathbb{N}$*, and for every* $t \geq N^{c_0}$*, where* $N = \sum_{i=1}^{\ell} n_i$*, we have*

$$\mathsf{K}^t(x_1, \ldots, x_\ell) \geq \sum_{i=1}^{\ell} \mathsf{K}^{t^{c_1}}(x_i \mid x_1, \ldots, x_{i-1}) - \ell \cdot O(\log t) - \delta(2(N + n_0^2), 2t), \tag{7}$$

where $n_0 = \max\{n_i \mid 1 \leq i \leq \ell\}$*.*

Proof. We will apply a DP generator to each of the ℓ strings $x_1, \ldots, x_\ell$, with the parameters $k_i \approx \mathsf{K}^{\mathsf{poly}(t)}(x_i \mid x_1, \ldots, x_{i-1})$ to be determined. We will use the same seed for each of these ℓ generators.

Note that each x_i of length $n_i \leq n_0$ has Kolmogorov complexity at most $n_i + O(1) \leq 2n_0$. So it suffices to have a common DP seed z of length at most $\max_i\{k_i n_i\} \leq 2n_0^2$. Each of the ℓ DP generators will use a prefix of the same z of the length required by the particular DP generator's parameters n_i and k_i. We provide more details next.

For $z \in \{0,1\}^{2n_0^2}$, and for $k_1, \ldots, k_\ell \geq 0$ to be determined, consider the following generator

$$G(z) := z \circ \mathsf{DP}_{k_1}^{x_1}(z_1) \circ \cdots \circ \mathsf{DP}_{k_\ell}^{x_\ell}(z_\ell),$$

where each z_i is the prefix of z of length $n_i \cdot k_i$, for all $1 \leq i \leq \ell$.

Note that the total output length of G is

$$M := |z| + \sum_{i=1}^{\ell} k_i \leq |z| + 2N = 2(n_0^2 + N),$$

since each $k_i \leq 2n_i$.

First, we show how to choose the parameters k_i, $1 \leq i \leq \ell$ to make G a (1/4)-pseudorandom generator against time $\tau(2(n_0^2+N), 2t)$-time uniform algorithms, where $\tau(n,t)$ is the polynomial runtime of the assumed natural property $\mathcal{A}$ for K^t on n-bit strings.

We will use a hybrid argument. Suppose there is a $\tau(2(n_0^2+N), 2t)$-time distinguisher D between the uniform distribution and the output of the generator G, with a distinguishing probability at

least 1/4. Define hybrid distributions $\mathcal{D}_0, \ldots, \mathcal{D}_\ell$ so that, for each $0 \le i \le \ell$, and the uniform distribution Z over $\{0,1\}^{2n_0^2}$,

$$\mathcal{D}_i := Z \circ \mathsf{DP}^{x_1}_{k_1}(Z_1) \circ \cdots \circ \mathsf{DP}^{x_i}_{k_i}(Z_i) \circ \mathcal{U}_{k_{i+1}} \circ \cdots \circ \mathcal{U}_{k_\ell},$$

where each Z_i is the prefix of Z of length $k_i n_i$.

By assumption, D is a (1/4)-distinguisher between $\mathcal{D}_0$ and $\mathcal{D}_\ell$. By a hybrid argument, there must exist an index $1 \le i \le \ell$ such that D is a $(1/(4\ell))$-distinguisher between $\mathcal{D}_{i-1}$ and $\mathcal{D}_i$. Note that D can be used to define a randomized distinguisher D' between $\mathsf{s\text{-}DP}^{x_i}_{k_i}(z_i)$, for $z_i \in \mathcal{U}_{|z_i|}$, and $\mathcal{U}_{|z_i|+k_i}$, given as advice $x_1, \ldots, x_{i-1}$:

> The distinguisher D' on inputs z_i and $w \in \{0,1\}^{k_i}$, will sample a uniformly random string $\alpha \in \{0,1\}^{2n_0^2 - |z_i|}$ to define $z = z_i \circ \alpha$ of length $2n_0^2$, which will be placed in the position of Z in $\mathcal{D}_i$. The string w will be placed in the ith DP position of $\mathcal{D}_i$. Then D' will use its advice and randomness to sample from the hybrid distribution $\mathcal{D}_i$ in the remaining positions. Finally, D' will simulate D on the resulting tuple of strings, accepting iff D accepts.

The runtime of the randomized distinguisher D' is that of D plus $O(M)$, which is $O(\tau(2(n_0^2 + N), 2t))$. By Lemma 2.4 we get that, for a sufficiently large c_0 such that $t \ge N^{c_0}$, and for $q(t) = p_{\mathsf{DP}}(\mathsf{poly}(t))$,

$$\mathsf{K}^{q(t)}(x_i \mid x_1, \ldots, x_{i-1}) \le k_i + \log q(t). \tag{8}$$

Set

$$k_i = \max\{0, \mathsf{K}^{q(t)}(x_i \mid x_1, \ldots, x_{i-1}) - \log q(t) - 1\} \tag{9}$$

so that the inequality in (8) cannot hold. (Note that if $k_i = 0$, then $\mathcal{D}_{i-1}$ and $\mathcal{D}_i$ are identical, and hence indistinguishable by any algorithm D.) It follows that for these k_i's, the generator G is 1/4-pseudorandom against $\tau(2(n_0^2 + N), 2t)$-time uniform algorithms.

Set $s := M - \delta(M, 2t)$. By the definition of a natural property, $\mathcal{A}(-, 1^{2t})$ rejects with probability at least 1/2 on the uniform distribution $\mathcal{D}_0$. Hence, by the (1/4)-pseudorandomness of G, $\mathcal{A}(-, 1^{2t})$ rejects with probability at least $1/2 - 1/4 = 1/4$ on the outputs of G.

By averaging, there exists a string $z \in \{0,1\}^{2n_0^2}$ such that

$$\mathcal{A}(z \circ \mathsf{DP}^{x_1}_{k_1}(z_1) \circ \cdots \circ \mathsf{DP}^{x_\ell}_{k_\ell}(z_\ell), 1^{2t}) = 0,$$

where, as before, each z_i is the prefix of z of length $k_i n_i$. By the definition of a natural property (and recalling the definition of $s = M - \delta(M, 2t)$), we get that

$$\mathsf{K}^{2t}\left(z \circ \mathsf{DP}^{x_1}_{k_1}(z_1) \circ \cdots \circ \mathsf{DP}^{x_\ell}_{k_\ell}(z_\ell)\right) > s = \sum_{i=1}^{\ell} k_i + |z| - \delta(M, 2t). \tag{10}$$

On the other hand, for every seed $z \in \{0,1\}^{2n_0^2}$, we have

$$\mathsf{K}^{2t}\left(z \circ \mathsf{DP}^{x_1}_{k_1}(z_1) \circ \cdots \circ \mathsf{DP}^{x_\ell}_{k_\ell}(z_\ell)\right) \le \mathsf{K}^t(x_1, \ldots, x_\ell) + |z| + c \cdot \ell \cdot \log(N/\ell), \tag{11}$$

where the last term is to take into account encoding of all n_i's and k_i's, which can be done using at most $c \cdot \sum_{i=1}^{\ell} \log n_i$ bits, for some constant $c > 0$; using the AM-GM inequality, we have $\sum_{i=1}^{\ell} \log n_i \le \ell \cdot \log(N/\ell)$.

Combining (10) and (11), and recalling the definition of k_i's in (9), we get that

$$\mathsf{K}^t(x_1,\ldots,x_\ell) + |z| + c\cdot\ell\cdot\log(N/\ell) \geq \sum_{i=1}^{\ell}\Big(\mathsf{K}^{q(t)}(x_i \mid x_1,\ldots,x_{i-1}) - \log q(t) - 1\Big) + |z| - \delta(M,2t),$$

which implies the required lower bound in (7). □

Remark 3.2. *If we assume a natural property with $\delta(2(n_0^2+N),2t) \leq \ell\cdot O(\log t)$ in Theorem 3.1, then we get an optimal chain rule statement for ℓ strings with an overall additive error $\ell\cdot O(\log t)$.*

3.2 Case of Conditional K^t

Here we generalize Theorem 3.1 to the case of conditional K^t complexity, by using a natural property for conditional K^t. Note that we get slightly better error parameters for the conditional Chain Rule below compared to the unconditional Chain Rule of Theorem 3.1.

Theorem 3.3 (Conditional Chain Rule for K^t)**.** *Suppose that*

- $\mathsf{E} \not\subset \mathsf{io\text{-}SIZE}[2^{o(n)}]$*, and*
- *there is a P-computable natural property $\mathcal{A}(w,z,1^t)$ for conditional K^t on n-bit inputs w, conditioned on $z\in\{0,1\}^*$, with usefulness $s(n,t) = n - \delta(n,t)$.*

Then there exist constants $c_0, c_1 \in \mathbb{N}$ such that for all sufficiently large $y, x_1,\ldots,x_\ell \in \{0,1\}^*$, for any $\ell\in\mathbb{N}$, and for every $t \geq M^{c_0}$, where $M = |y| + \sum_{i=1}^{\ell}|x_i|$, we have

$$\mathsf{K}^t(x_1,\ldots,x_\ell \mid y) \geq \sum_{i=1}^{\ell}\mathsf{K}^{t^{c_1}}(x_i \mid y, x_1,\ldots,x_{i-1}) - \ell\cdot O(\log t) - \delta(2N,2t), \tag{12}$$

where $N = \sum_{i=1}^{\ell}|x_i|$.

Proof. The proof is similar to that of Theorem 3.1, with conditioning on y added and some modifications. The main difference is that we can afford to use independent seeds for different DP generators now because we can "hide" all these seeds in the conditioned string of the assumed natural property $\mathcal{A}$ for conditional K^t. This allows us to make the error of the Chain Rule for conditional K^t to be $\delta(2N,2t)$, independent of the maximum length of the strings $x_1,\ldots,x_\ell$ (in contrast to Theorem 3.1). We provide the details next.

For given strings $x_1,\ldots,x_\ell$ of lengths $n_1,\ldots,n_\ell$, respectively, for a string y of length m, and for the parameters $k_1,\ldots,k_\ell$ to be determined, we consider the following generator

$$G(z_1,\ldots,z_\ell) = \mathsf{DP}^{x_1}_{k_1}(z_1)\circ\cdots\circ\mathsf{DP}^{x_\ell}_{k_\ell}(z_\ell),$$

where $|z_i| = n_i\cdot k_i$. Note that the output of this generator G is

$$M := \sum_{i=1}^{\ell} k_i \leq 2N,$$

since each $k_i \leq 2n_i$.

For every given sequence of seeds $z_1, \ldots, z_\ell$, we have

$$\mathsf{K}^{2t}\left(\mathsf{DP}^{x_1}_{k_1}(z_1) \circ \cdots \circ \mathsf{DP}^{x_\ell}_{k_\ell}(z_\ell) \mid y, z_1, \ldots, z_\ell\right) \leq \mathsf{K}^t(x_1, \ldots, x_\ell \mid y) + c \cdot \ell \cdot \log(N/\ell), \tag{13}$$

where the term $c \cdot \ell \cdot \log(N/\ell)$ accounts for the encoding of the input lengths $n_1, \ldots, n_\ell$.

We argue that, for uniformly random z_i's, $G(z_1, \ldots, z_\ell)$ is $(1/4)$-pseudorandom with respect to the distinguisher $\mathcal{A}(-, y \circ z_1 \circ \cdots \circ z_\ell, 1^{2t})$. Consider the hybrid distributions

$$\mathcal{D}_i = \mathsf{DP}^{x_1}_{k_1}(Z_1) \circ \cdots \circ \mathsf{DP}^{x_i}_{k_i}(Z_i) \circ \mathcal{U}_{k_{i+1}} \circ \cdots \circ \mathcal{U}_{k_\ell} \circ y \circ Z_1 \circ \cdots \circ Z_\ell,$$

for independent uniform distributions $Z_1, \ldots, Z_\ell$ over strings of lengths $n_1k_1, \ldots, n_\ell k_\ell$, respectively. Suppose that $\mathcal{A}(-, 1^{2t})$ is a $(1/4)$-distinguisher between $\mathcal{D}_0$ and $\mathcal{D}_\ell$. By a hybrid argument, there must exist some $1 \leq i \leq \ell$ such that $\mathcal{A}(-, 1^{2t})$ is a $(1/(4\ell))$-distinguisher between $\mathcal{D}_{i-1}$ and $\mathcal{D}_i$.

We can use this fact to get a randomized distinguisher D that, given advice $y, x_1, \ldots, x_{i-1}$, will $(1/(4\ell))$-distinguish between $\mathsf{s}\text{-}\mathsf{DP}^{x_i}_{k_i}(\mathcal{U}_{k_i n_i})$ and $\mathcal{U}_{k_i n_i + k_i}$. The distinguisher D is as follows:

> The distinguisher D, on input $z_i \in \{0,1\}^{n_i k_i}$ and $w_i \in \{0,1\}^{k_i}$, will randomly sample strings z_j, for $1 \leq j \leq \ell$ with $j \neq i$, will compute (using its advice $x_1, \ldots, x_{i-1}$) the strings $\mathsf{DP}^{x_1}_{k_1}(z_1), \ldots, \mathsf{DP}^{x_{i-1}}_{k_{i-1}}(z_{i-1})$, and will simulate
>
> $$\mathcal{A}(\mathsf{DP}^{x_1}_{k_1}(z_1), \ldots, \mathsf{DP}^{x_{i-1}}_{k_{i-1}}(z_{i-1}), w_i, \mathcal{U}_{k_{i+1}}, \ldots, \mathcal{U}_{k_\ell}, y, z_1, \ldots, z_\ell, 1^{2t}),$$
>
> accepting iff $\mathcal{A}$ accepts.

By Lemma 2.4 we get that, for a sufficiently large c_0 such that $t \geq M^{c_0}$, and for $q(t) = p_{\mathsf{DP}}(\mathsf{poly}(t))$,

$$\mathsf{K}^{q(t)}(x_i \mid y, x_1, \ldots, x_{i-1}) \leq k_i + \log q(t). \tag{14}$$

Set

$$k_i = \max\{0, \mathsf{K}^{q(t)}(x_i \mid y, x_1, \ldots, x_{i-1}) - \log q(t) - 1\} \tag{15}$$

so that the inequality in (14) cannot hold. (Note that if $k_i = 0$, then $\mathcal{D}_{i-1}$ and $\mathcal{D}_i$ are identical, and hence indistinguishable by any algorithm.) We get for these k_is that

$$\mathcal{A}(G(z_1, \ldots, z_\ell), y \circ z_1 \circ \cdots \circ z_\ell, 1^{2t}) = 0$$

with probability at least $1/2 - 1/4 = 1/4$, since $\mathcal{A}(w, z, 1^t)$, as a natural property for conditional K^t, must reject at least $1/2$ of random inputs w. By averaging over z_i's and by the usefulness property of $\mathcal{A}$, we get for some $z_1, \ldots, z_\ell$ that

$$\begin{aligned}\mathsf{K}^{2t}(G(z_1, \ldots, z_\ell) \mid y, z_1, \ldots, z_\ell) &\geq M - \delta(M, 2t) \\ &\geq \sum_{i=1}^{\ell} k_i - \delta(2N, 2t).\end{aligned}$$

Combining this inequality with (13) and (15) above, we get the claimed chain rule. □

3.3 Case of Conditional pK^t

Here we prove that a chain rule for conditional probabilistic Kolmogorov complexity pK^t follows from a BPP-computable natural property for conditional K^t. The difference from Theorem 3.3 above is that (1) no circuit lower bound for E (derandomization assumption) is needed, and (2) a natural property for conditional K^t can be BPP-computable (rather than P-computable).[6]

Theorem 3.4 (Chain Rule for Conditional pK^t)**.** *Suppose that there is a BPP-computable natural property $\mathcal{A}$ for conditional K^t on n-bit inputs with usefulness $s(n,t) = n - \delta(n,t)$. Then there exist constants $c_0, c_1 \in \mathbb{N}$ such that for all sufficiently large $x_1, \ldots, x_\ell \in \{0,1\}^*$, for any $\ell \in \mathbb{N}$, or every $y \in \{0,1\}^*$, and for every $t \geq (N + |y|)^{c_0}$, where $N = \sum_{i=1}^{\ell} |x_i|$, we have*

$$\mathsf{pK}^t(x_1, \ldots, x_\ell \mid y) \geq \sum_{i=1}^{\ell} \mathsf{pK}^{t^{c_1}}(x_i \mid y, x_1, \ldots, x_{i-1}) - \ell \cdot O(\log N) - \delta(2N, 2t). \tag{16}$$

Proof. The proof is similar to that of Theorem 3.3. Let $n_i = |x_i|$, for all $1 \leq i \leq \ell$. For $k_1, \ldots, k_\ell$ to be determined, consider the concatenation of ℓ generators

$$\mathsf{DP}^{x_1}_{k_1}(z_1) \circ \cdots \circ \mathsf{DP}^{x_\ell}_{k_\ell}(z_\ell),$$

where $|z_i| = n_i \cdot k_i$ for all $1 \leq i \leq \ell$. Note that the total output length of this concatenation of DP generators is

$$M := \sum_{i=1}^{\ell} k_i \leq 2N,$$

assuming that each $k_i \leq 2n_i$.

For every given sequence of seeds $z_1, \ldots, z_\ell$, we have

$$\mathsf{pK}^{2t}\left(\mathsf{DP}^{x_1}_{k_1}(z_1) \circ \cdots \circ \mathsf{DP}^{x_\ell}_{k_\ell}(z_\ell) \mid y, z_1, \ldots, z_\ell\right) \leq \mathsf{pK}^t(x_1, \ldots, x_\ell \mid y) + O(\ell \cdot \log(N/\ell)) =: \sigma.$$

By the definition of pK^t, this means that, for at least $2/3$ of random strings $r \in \{0,1\}^{2t}$,

$$\mathsf{K}^{2t}\left(\mathsf{DP}^{x_1}_{k_1}(z_1) \circ \cdots \circ \mathsf{DP}^{x_\ell}_{k_\ell}(z_\ell) \mid r, y, z_1, \ldots, z_\ell\right) \leq \sigma. \tag{17}$$

Set $s = M - \delta(M, 2t)$. By the definition of a BPP-computable natural property for conditional K^t with usefulness s, we have for every sequence of $z_1, \ldots, z_\ell$, every r, and every y that $\mathcal{A}(-, 1^{2t})$ rejects on the distribution

$$\mathcal{U}_{k_1} \circ \cdots \circ \mathcal{U}_{k_\ell} \circ r \circ y \circ z_1 \circ \cdots \circ z_\ell,$$

with probability at least $(1/2) \cdot 0.9 = 0.45$ (where the probability is over both the distribution of uniformly random inputs and the internal randomness of $\mathcal{A}$). Hence, it rejects with probabilty at least 0.45 on the distribution

$$\mathcal{U}_{k_1} \circ \cdots \circ \mathcal{U}_{k_\ell} \circ R \circ y \circ Z_1 \circ \cdots \circ Z_\ell,$$

[6] Assuming a BPP-computable natural property for K^t, rather than conditional K^t, one can get a chain rule for pK^t, albeit with worse error parameters. Since these error parameters are insufficient for getting an equivalence between the chain rule and a natural property, we do not state this result here.

where R is the uniform disribution over strings of length $|r|$, and each Z_i is uniform over strings of length $|z_i|$ (independend of all other distributions).

Next we will argue that $\sigma > s$. Towards a contradition, suppose that $\sigma \leq s$. Then, by (17), we have that $\mathcal{A}(-, 1^{2t})$ accepts with probability at least $0.9 \cdot (2/3) = 0.6$ on the distribution

$$\mathsf{DP}^{x_1}_{k_1}(Z_1) \circ \cdots \circ \mathsf{DP}^{x_\ell}_{k_\ell}(Z_\ell) \circ R \circ y \circ Z_1 \circ \cdots \circ Z_\ell,$$

where the probability is both over the input distribution and the internal randomness of $\mathcal{A}$.

Define hybrid distributions $\mathcal{D}_0, \ldots, \mathcal{D}_\ell$ so that, for each $0 \leq i \leq \ell$,

$$\mathcal{D}_i := \mathsf{DP}^{x_1}_{k_1}(Z_1) \circ \cdots \circ \mathsf{DP}^{x_i}_{k_i}(Z_i) \circ \mathcal{U}_{k_{i+1}} \circ \cdots \circ \mathcal{U}_{k_\ell} \circ R \circ y \circ Z_1 \circ \cdots \circ Z_\ell.$$

Since $\mathcal{A}(-, 1^{2t})$ distinguishes between $\mathcal{D}_0$ and $\mathcal{D}_\ell$ with the distinguishing probability at least $0.6 - 0.55 = 0.05$, we get by the hybrid argument that, for some $1 \leq i \leq \ell$, $\mathcal{A}(-, 1^{2t})$ distinguishes between $\mathcal{D}_{i-1}$ and $\mathcal{D}_i$ with the distinguishing probability at least $0.05/\ell$. By Lemma 2.5 we get that, for a sufficiently large c_0 such that $t \geq N^{c_0}$, and for $q(t) = p_{\mathsf{DP}}(\mathsf{poly}(t))$,

$$\mathsf{pK}^{q(t)}(x_i \mid y, x_1, \ldots, x_{i-1}) \leq k_i + O(\log N). \tag{18}$$

Set

$$k_i = \max\{0, \mathsf{pK}^{q(t)}(x_i \mid y, x_1, \ldots, x_{i-1}) - O(\log N) - 1\} \tag{19}$$

for all $1 \leq i \leq \ell$, so that (18) cannot hold. Therefore, for this setting of k_i's, we conclude that $\sigma > s$. This means that, by the definitions of σ, s, and k_i's,

$$\begin{aligned}
\sigma &= \mathsf{pK}^t(x_1, \ldots, x_\ell \mid y) + O(\ell \cdot \log(N/\ell)) \\
&> s \\
&= M - \delta(M, 2t) \\
&= \sum_{i=1}^{\ell} k_i - \delta(M, 2t) \\
&\geq \sum_{i=1}^{\ell} \Big(\mathsf{pK}^{q(t)}(x_i \mid y, x_1, \ldots, x_{i-1}) - O(\log N) - 1 \Big) - \delta(2N, 2t),
\end{aligned}$$

which implies the required inequality (16). □

4 Natural Properties from Chain Rules

4.1 Case of K^t and Conditional K^t

Theorem 4.1. *Assume that, for every constant $C > 1$, the Chain Rule for K^t as in Eq. (7) of Theorem 3.1 holds for $\delta(N, t) \leq N/C$. Then, for every constant $D \geq 1$, there is a P-computable natural property $\mathcal{A}$ for K^t on n-bit inputs with usefulness $s(n, t) = n - n/D$.*

Proof. Given $x \in \{0,1\}^n$, partition x into $1 \leq \ell \leq n$ strings $x_1, \ldots, x_\ell$ of length n/ℓ each, for some $\ell = n/(c \log t)$, for a constant $c > 0$ to be chosen later. Note that

$$\mathsf{K}^{2t}(x_1, \ldots, x_\ell) \leq \mathsf{K}^t(x) + O(\log n).$$

On the other hand, by the assumed Chain Rule, we have, for sufficiently large $t \geq \mathsf{poly}(n)$, that

$$\mathsf{K}^{2t}(x_1, \ldots, x_\ell) \geq \sum_{i=1}^{\ell} \mathsf{K}^{\mathsf{poly}(t)}(x_i \mid x_1, \ldots, x_{i-1}) - \ell \cdot O(\log t) - \delta(2((n/\ell)^2 + n), 2t).$$

Assume that $\mathsf{K}^t(x) \leq s(n,t)$. Then

$$\sum_{i=1}^{\ell} \mathsf{K}^{\mathsf{poly}(t)}(x_i \mid x_1, \ldots, x_{i-1}) \leq s(n,t) + \ell \cdot O(\log t) + \delta(2((n/\ell)^2 + n), 2t).$$

Hence, by averaging, there is some $1 \leq i \leq \ell$ such that

$$\mathsf{K}^{\mathsf{poly}(t)}(x_i \mid x_1, \ldots, x_{i-1}) \leq \left(s(n,t) + \ell \cdot O(\log t) + \delta(2((n/\ell)^2 + n), 2t)\right)/\ell \;=: S. \tag{20}$$

Note that, by brute force, for each $1 \leq i \leq \ell$, we can compute $\mathsf{K}^{\mathsf{poly}(t)}(x_i \mid x_1, \ldots, x_{i-1})$ in time $2^{n/\ell} \cdot \mathsf{poly}(t)$, which is at most $\mathsf{poly}(t)$ for $\ell = n/(c \log t)$, for a constant $c > 0$ to be determined. Consider the following decision algorithm $\mathcal{A}$:

> On input $x \in \{0,1\}^n$ and 1^t, output 1 iff there is some $1 \leq i \leq \ell$ such that Eq. (20) is satisfied.

Clearly, $\mathcal{A}$ is a polynomial-time algorithm. It accepts every string $x \in \{0,1\}^n$ with $\mathsf{K}^t(x) \leq s(n,t)$. We need to argue that $\mathcal{A}$ will reject at least $1/2$ of random input strings $z \in \{0,1\}^n$, for an appropriately chosen $s(n,t)$.

Suppose that $s(n,t)$ is such that

$$S \leq (n/\ell) - 2\log \ell, \tag{21}$$

for S defined in (20). By a simple counting argument, we have, for any fixed string w,

$$\mathbf{Pr}_{y \in \{0,1\}^{n/\ell}}\left[\mathsf{K}(y \mid w) \leq (n/\ell) - 2\log \ell\right] \leq 2/\ell^2.$$

Imagine picking a uniformly random $z \in \{0,1\}^n$ in stages, where we first pick $z_1 \in \{0,1\}^{n/\ell}$, then $z_2 \in \{0,1\}^{n/\ell}$, and so on until $z_\ell \in \{0,1\}^{n/\ell}$, and finally output $z = z_1 \circ z_2 \circ \cdots \circ z_\ell$. By the union bound, the probability that at least one of $z_1, \ldots, z_\ell \in \{0,1\}^{n/\ell}$ has

$$K(z_i \mid z_1, \ldots, z_{i-1}) \leq (n/\ell) - 2\log \ell$$

is at most $\ell \cdot (2/\ell^2) \leq 2/\ell$. It follows that for $s(n,t)$ satisfying (21) above, we have

$$\mathbf{Pr}_{z \in \{0,1\}^n}[\mathcal{A}(z, 1^t) = 1] \leq 2/\ell,$$

which is at most $1/2$ for $\ell \geq 4$.

It remains to see which values $s(n,t)$ will satisfy (21). We get that $s(n,t)$ must satisfy the following:

$$s(n,t) \leq n - 2\ell \cdot \log \ell - \ell \cdot O(\log t) - \delta(2((n/\ell)^2 + n), 2t).$$

We will set $s(n,t)$ equal to the expression above. Recalling that $\ell = n/(c \log t)$, we get for any given constant $d > 0$ (by choosing c a large enough constant) that

$$\begin{aligned} s(n,t) &= n - (2n \log \ell)/(c \log t) - n \cdot O(\log t)/(c \log t) - \delta(2((n/\ell)^2 + n), 2t) \\ &\geq n - n/d - \delta(3n, 2t) \\ &\geq n - n/d - (3n)/C \\ &= n - n/D, \end{aligned}$$

for $C = 3d$ and $d = 2D$. The theorem follows. □

Theorem 4.2. *Assume that, for every constant $C > 1$, the Chain Rule for conditional K^t as in Eq. (12) of Theorem 3.3 holds for $\delta(N,t) \leq N/C$. Then, for every constant $D \geq 1$, there is a P-computable natural property $\mathcal{A}(x, y, 1^t)$ for conditonal K^t on n-bit inputs x, conditioned on y, with usefulness $s(n,t) = n - n/D$.*

Proof. The proof is analogous to that of Theorem 4.1 above, with a given string y added as a conditioned string everywhere in the proof. □

4.2 Case of Conditional pK^t

Theorem 4.3. *Assume that, for every constant $C > 1$, the Chain Rule for conditional pK^t as in Eq. (16) of Theorem 3.4 holds for $\delta(N,t) \leq N/C$. Then, for every constant $D \geq 1$, there is a BPP-computable natural property $\mathcal{A}$ for conditional K^t on n-bit inputs with usefulness $s(n,t) = n - n/D$.*

Proof. The proof is similar to that of Theorem 4.1, appropriately adapted to the case of conditional pK^t. Given $x \in \{0,1\}^n$, partition x into $1 \leq \ell \leq n$ strings $x_1, \ldots, x_\ell$ of length n/ℓ each, for some ℓ to be chosen later. Note that, for every $y \in \{0,1\}^*$,

$$\mathsf{pK}^{2t}(x_1, \ldots, x_\ell \mid y) \leq \mathsf{pK}^t(x \mid y) + O(\log n).$$

On the other hand, by the assumed Chain Rule, we have, for sufficiently large $t \geq \mathsf{poly}(n)$, that

$$\mathsf{pK}^{2t}(x_1, \ldots, x_\ell \mid y) \geq \sum_{i=1}^{\ell} \mathsf{pK}^{\mathsf{poly}(t)}(x_i \mid y, x_1, \ldots, x_{i-1}) - \ell \cdot O(\log n) - \delta(2n, 2t).$$

Assume that $\mathsf{K}^t(x \mid y) \leq s(n,t)$. Then $\mathsf{pK}^t(x \mid y) \leq \mathsf{K}^t(x \mid y) \leq s(n,t)$, and so

$$\sum_{i=1}^{\ell} \mathsf{pK}^{\mathsf{poly}(t)}(x_i \mid y, x_1, \ldots, x_{i-1}) \leq s(n,t) + \ell \cdot O(\log n) + \delta(2n, 2t).$$

Hence, by averaging, there is some $1 \leq i \leq \ell$ such that

$$\mathsf{pK}^{\mathsf{poly}(t)}(x_i \mid y, x_1, \ldots, x_{i-1}) \leq \left(s(n,t) + \ell \cdot O(\log n) + \delta(2n, 2t)\right)/\ell \;=: S. \tag{22}$$

By brute force, for each $1 \leq i \leq \ell$, we can approximate $\mathsf{pK}^{\mathsf{poly}(t)}(x_i \mid y, x_1, \ldots, x_{i-1})$ (by random sampling) in randomized time $2^{n/\ell} \cdot \mathsf{poly}(t)$, which is at most $\mathsf{poly}(t)$ for $\ell = n/(c \log t)$, for some sufficiently large constant $c > 0$ to be determined. We provide the details next.

Consider the following randomized decision algorithm $\mathcal{A}$:

On input $x \in \{0,1\}^n$, $y \in \{0,1\}^*$, and 1^t, output 1 iff there is some $1 \le i \le \ell$ such that Eq. (22) is approximately satisfied. More precisely, output 1 if, for some $1 \le i \le \ell$, for each of at least 0.6 fraction of randomly sampled n strings $r \in \{0,1\}^{t'}$, for $t' = \mathsf{poly}(t)$, there is a string $w_r \in \{0,1\}^{\le S}$ such that $U(w_r, r, y, x_1, \ldots, x_{i-1})$ outputs x_i within t' steps.

Clearly, $\mathcal{A}$ is a randomized polynomial-time algorithm. It accepts every string $x \in \{0,1\}^n$ with $\mathsf{K}^t(x \mid y) \le s(n,t)$, with high probability over its internal randomness (which can be argued by the Chernoff bounds). We need to argue that, for every y, $\mathcal{A}(-, y, 1^t)$ will reject at least $1/2$ of random input strings $z \in \{0,1\}^n$, for an appropriately chosen $s(n,t)$.

Suppose that $s(n,t)$ is such that

$$S \le (n/\ell) - 2\log \ell, \tag{23}$$

for S defined in (22). By Lemma 2.1, we have, for any fixed string w,

$$\mathbf{Pr}_{a \in \{0,1\}^{n/\ell}} \left[\mathsf{pK}^{t'}_{0.55}(a \mid w) \le (n/\ell) - 2\log \ell \right] \le 4/\ell^2.$$

Imagine picking a uniformly random $z \in \{0,1\}^n$ in stages, where we first pick $z_1 \in \{0,1\}^{n/\ell}$, then $z_2 \in \{0,1\}^{n/\ell}$, and so on until $z_\ell \in \{0,1\}^{n/\ell}$, and finally output $z = z_1 \circ z_2 \circ \cdots \circ z_\ell$. By the union bound, we have with probability at least $1 - 4/\ell \ge 1/2$ over random $z = z_1 \ldots z_\ell \in \{0,1\}^n$ that, for every one of $z_1, \ldots, z_\ell \in \{0,1\}^{n/\ell}$, it holds that

$$\mathsf{pK}^{t'}_{0.55}(z_i \mid y, z_1, \ldots, z_{i-1}) > (n/\ell) - 2\log \ell.$$

Consider any such $z = z_1 \ldots z_\ell \in \{0,1\}^n$. For each $1 \le i \le \ell$, there are fewer than 0.55 fraction of "good" random strings $r \in \{0,1\}^{t'}$ for which there is a description $w_r \in \{0,1\}^{\le S}$ such that $U(w_r, r, y, z_1, \ldots, z_{i-1})$ outputs z_i within t' steps. By the Chernoff bounds, algorithm $\mathcal{A}$ will see at least 0.6 fraction of "good" random strings r in its sample of size n with probability at most $2^{-\Omega(n)}$. By the union bound, the probability that $\mathcal{A}$ will accept for at least one $1 \le i \le \ell$ is at most $\ell \cdot 2^{-\Omega(n)}$. Hence, the probability over the internal randomness of $\mathcal{A}$ that $\mathcal{A}(z, y, 1^t) = 0$ is at least $1 - \ell \cdot 2^{-\Omega(n)} \ge 0.9$, for all sufficiently large n.

It remains to see which values $s(n,t)$ will satisfy (23). We get that $s(n,t)$ must satisfy the following:

$$s(n,t) \le n - 2\ell \cdot \log \ell - \ell \cdot O(\log n) - \delta(2n, 2t).$$

Recalling that $\ell = n/(c \log t)$, we set, for any given constant $D > 0$ (by choosing c large enough),

$$\begin{aligned} s(n,t) &= n - (2n \log \ell)/(c \log t) - n \cdot O(\log n)/(c \log t) - \delta(2n, 2t) \\ &\ge n - n/(2D) - \delta(2n, 2t) \\ &= n - n/(2D) - (2n)/C \\ &\le n - n/D, \end{aligned}$$

for $C = 4D$. □

5 Circuit Lower Bounds from Chain Rules

5.1 Case of K^t

Here we get strongly exponential, almost everywhere, circuit lower bounds from a Chain Rule for K^t, using an idea from [Per07]; see also [All+06] for a similar argument in the setting of *time-unbounded* Kolmogorov complexity.[7] We will need the following lemma.

Lemma 5.1. *Let $x \in \{0,1\}^N$, for $N = 2^n$, be such that, for for $t \geq N^3$, we have*

$$\mathsf{K}^t(x) \geq (0.9) \cdot N. \tag{24}$$

Then x, viewed as a truth table of an n-variate Boolean function, requires circuit size at least $2^n/(c \cdot n)$, for a sufficiently large constant $c \in \mathbb{N}$.

Proof. Suppose towards a contradiction that x can be computed by a circuit of size $\sigma < N/(c \cdot n)$. Such a circuit can be described using

$$\sigma' = \sigma \cdot d \cdot \log \sigma < \frac{N \cdot d \cdot n}{c \cdot n}$$

bits, for some constant $d \in \mathbb{N}$. We can make $\sigma' < N/2$ by choosing $c = 2d$. Given the description of this circuit, one can compute the entire string x bit by bit, by evaluating the circuit on all possible n-bit inputs. This can be done in time $2^n \cdot (\sigma')^2 \leq N^3$. Hence, for $t \geq N^3$, we have that $\mathsf{K}^t(x) < N/2$, contradicting (24). □

Theorem 5.2. *Assume that, for any constant $D > 1$, the Chain Rule for K^t as in Eq. (7) of Theorem 3.1 holds for $\delta(N,t) \leq N/D$. Then*

$$\mathsf{E} \not\subset \mathsf{io\text{-}SIZE}[o(2^n/n)].$$

Proof. Let $N = 2^n$, $m = c \cdot n$ for some constant $c > 0$ to be determined, and $\ell = N/m$. Let $N^{c_0} \leq t = N^{c_1}$, for some constant $c_1 \geq 3$. Let $t' = \mathsf{poly}(t)$ be the polynomial time bound on the right-hand side of the Chain Rule for K^t.

By a brute-force algorithm, we construct $A_1 \in \{0,1\}^m$ so that

$$\mathsf{K}^{t'}(A_1) \geq m. \tag{25}$$

Note that by a counting argument, such a string A_1 must exist. We find the lexicographically first such string by enumerating all m-bit strings $a \in \{0,1\}^m$, and checking for each if some $w \in \{0,1\}^{\leq m}$ exists such that $U(w)$ outputs a within t' steps; if yes, we skip over to the next $a \in \{0,1\}^m$. It takes time $2^{O(n)}$ to find A_1.

Similarly, given the found string A_1, we next find (again by brute force, in time $2^{O(n)}$) a string $A_2 \in \{0,1\}^m$ such that

$$\mathsf{K}^{t'}(A_2 \mid A_1) \geq m.$$

Such a string must exist by a counting argument. To find it, we proceed similarly to the case of A_1, but now simulating $U(w, A_1)$ for t' steps.

[7] More precisely, [All+06] re-prove a result of [Buh+05] that one can efficiently deterministically construct a string of high Kolmogorov complexity, given oracle access to the set of Kolmogorov-random strings. The proof relies on the Symmetry of Information for time-unbounded Kolmogorov complexity.

We continue this way to define $A_1, A_2, \ldots, A_\ell$ so that, for all $1 \le i \le \ell$,

$$\mathsf{K}^{t'}(A_i \mid A_1, \ldots, A_{i-1}) \ge m. \tag{26}$$

Next, by the assumed Chain Rule, we get from Eq. (26) that

$$\begin{aligned}\mathsf{K}^t(A_1, \ldots, A_\ell) &\ge \sum_{i=1}^{\ell} \mathsf{K}^{t'}(A_i \mid A_1, \ldots, A_{i-1}) - \ell \cdot O(\log t) - \delta(2(m^2+N), 2t) \\ &\ge \ell \cdot m - \ell \cdot O(c_1 n) - 4N/D \\ &= \frac{2^n}{cn} \cdot (cn - O(c_1 n) - 4cn/D)\,. \end{aligned} \tag{27}$$

Choose the constants c and D large enough so that $(cn - O(c_1 n) - 4cn/D) \ge (0.9) \cdot cn$. Then $B = A_1 \ldots A_\ell \in \{0,1\}^{2^n}$ requires circuit size $\Omega(2^n/n)$ by Lemma 5.1.

Finally, observe that each A_i, for $1 \le i \le \ell$, is computable in time $2^{O(n)}$. Hence, the truth table $B \in \{0,1\}^{2^n}$ is computable in time $2^{O(n)}$. Collecting such B's over all input lengths n, we get a language $L \in \mathsf{E}$ that, almost everywhere, requires circuit size $\Omega(2^n/n)$. □

Remark 5.3. *Under the assumption that a variant of the two-string Conditional Chain Rule for* K^t *holds, Perifel [Per07] proved that* $\mathsf{EXP} \not\subset \mathsf{P/poly}$. *One reason that [Per07] obtained only a superpolynomial rather than exponential circuit lower bound for* EXP *is that it used this two-string Conditional Chain Rule to derive an ℓ-string Chain Rule for* K^t*, for a super-constant ℓ, suffering significant blowups in the time bounds for the* K^t *complexities involved. This precluded the use of sufficiently large values of ℓ in the derived Chain Rule. In contrast, we start with an optimal ℓ-string Chain Rule for* K^t *for an arbitrary $\ell \in \mathbb{N}$, and so we can choose a sufficiently large value of ℓ to get exponential circuit lower bounds.*

5.2 Case of pK^t

We get circuit lower bounds for randomized complexity classes from the assumption that a Chain Rule for pK^t holds.

Lemma 5.4. *A* BPP*-computable natural property for* K^{n^3} *on n-bit inputs with usefulness $s(n) \ge (0.9) \cdot n$ yields a* BPP*-computable natural property for circuit size on truth tables of length $N = 2^n$ with usefulness $s(N) \ge N^\varepsilon$, for some $0 < \varepsilon < 1$.*

Proof. Let $\mathcal{A}$ be a BPP-computable natural property for K^{n^3} on n-bit strings with usefulness $s(n) \ge (0.9) \cdot n$. Define $\mathcal{A}' \colon \{0,1\}^N \to \{0,1\}$ as follows:

On input $x \in \{0,1\}^N$, $\mathcal{A}'(x) = \mathcal{A}(x, 1^{N^3})$.

By definition, $\mathcal{A}'$ is a randomized $\mathsf{poly}(N)$-time algorithm. For at least $1/2$ of random strings $w \in \{0,1\}^N$, $\mathcal{A}'(w)$ rejects with probability at least 0.9 because so does $\mathcal{A}(w)$. For $x \in \{0,1\}^N$ with $\mathsf{size}(x) \le 2^{(0.9)\cdot n}$, we get by Lemma 5.1 that $\mathsf{K}^{N^3}(x) < (0.9) \cdot N$, for all sufficiently large $n \in \mathbb{N}$. Hence, $\mathcal{A}(x, 1^{N^3})$ accepts such an x with probability at least 0.9, and therefore so does $\mathcal{A}'(x)$. □

Theorem 5.5. *Assume the Chain Rule for conditional* pK^t *as in Eq. (16) of Theorem 3.4, for $\delta(n,t) \le n/c$, for a sufficiently large $c > 1$. Then*

- $\mathsf{BPEXP} \not\subset \mathsf{P/poly}$, *and,*
- $\mathsf{BPP}/1 \not\subset \mathsf{SIZE}[n^k]$, *and* $\mathsf{promise\text{-}BPP} \not\subset \mathsf{SIZE}[n^k]$ *for any fixed* $k \geq 1$.

Proof. By our assumption and by Theorem 4.3, we get that there is a BPP-computable natural property $\mathcal{A}$ for K^{n^3} on n-bit strings with usefulness $s(n) = (0.9) \cdot n$. By Lemma 5.4 and Corollary 2.9, the theorem follows. □

6 Proofs of Equivalences for Chain Rules

Here we prove the main equivalences stated in the introduction.

6.1 Case of K^t

Theorem 6.1 (Equivalence for K^t)**.** *Consider the following assumptions:*

CKT-LB*:* $\mathsf{E} \not\subset \mathsf{io\text{-}SIZE}[2^{o(n)}]$,

PROP-K^t*: for any constant* $C > 1$*, there is a* P*-computable natural property* $\mathcal{P}$ *for* K^t *with usefulness* $s(n,t) = n - n/C$*, and*

CHAIN-K^t*: for any costant* $D > 1$*, for some polynomial* p*, for any* $\ell \in \mathbb{N}$*, for any* $x_1, \ldots, x_\ell \in \{0,1\}^*$ *with* $N = \sum_{i=1}^{\ell} |x_i|$ *and each* $|x_i| \leq O(\sqrt{N})$*, and all sufficiently large* t *(at least polynomial in* N*), we have*

$$\mathsf{K}^t(x_1, \ldots, x_\ell) \geq \sum_{i=1}^{\ell} \mathsf{K}^{p(t)}(x_i \mid x_1, \ldots, x_{i-1}) - \ell \cdot O(\log t) - N/D.$$

We have that

$$\text{CKT-LB} \ \& \ \text{PROP-}\mathsf{K}^t \ \Leftrightarrow \ \text{CHAIN-}\mathsf{K}^t.$$

Proof. The direction CKT-LB&PROP-$\mathsf{K}^t \Rightarrow$ CHAIN-K^t follows by Theorem 3.1. That CHAIN-$\mathsf{K}^t \Rightarrow$ CKT-LB follows by Theorem 5.2. Finally, the direction CHAIN-$\mathsf{K}^t \Rightarrow$ PROP-K^t follows by Theorem 4.1. □

6.2 Case of Conditional K^t

Theorem 6.2 (Equivalence for Conditional K^t)**.** *Consider the following assumptions:*

CKT-LB*:* $\mathsf{E} \not\subset \mathsf{io\text{-}SIZE}[2^{o(n)}]$,

PROP-cK^t*: for any constant* $C > 1$*, there is a* P*-computable natural property* $\mathcal{P}$ *for conditional* K^t *with usefulness* $s(n,t) = n - n/C$*,*

GAP-$\mathsf{McK^tP}$*: for any constant* $C' > 1$*, there is a polynomial* τ *such that* $\mathsf{Gap}_{\tau,\delta'}\mathsf{McK^tP} \in \mathsf{promise\text{-}P}$*, where* $\delta'(n,t) = n/C'$*, and*

CHAIN-cK^t: *for any constant $D > 1$, for some polynomial p, for any $\ell \in \mathbb{N}$, for any $x_1, \ldots, x_\ell \in \{0,1\}^*$ with $N = \sum_{i=1}^{\ell} |x_i|$, and any $y \in \{0,1\}^*$, and all sufficiently large t (at least polynomial in $N + |y|$), we have*

$$\mathsf{K}^t(x_1, \ldots, x_\ell \mid y) \geq \sum_{i=1}^{\ell} \mathsf{K}^{p(t)}(x_i \mid y, x_1, \ldots, x_{i-1}) - \ell \cdot O(\log t) - N/D.$$

We have that

$$\text{CKT-LB} \;\&\; \text{PROP-}\mathsf{cK}^t \;\Leftrightarrow\; \text{CHAIN-}\mathsf{cK}^t \;\Leftrightarrow\; \text{CKT-LB} \;\&\; \text{GAP-}\mathsf{McK^tP}.$$

Proof. The direction CKT-LB & PROP-cK^t $\Rightarrow$ CHAIN-cK^t follows by Theorem 3.3. The direction CHAIN-cK^t $\Rightarrow$ CKT-LB follows by Theorem 5.2. The direction CHAIN-cK^t $\Rightarrow$ PROP-cK^t follows by Theorem 4.2. Finally, the equivalence CKT-LB & PROP-cK^t $\Leftrightarrow$ CKT-LB & GAP-$\mathsf{McK^tP}$ is by Corollary 2.17. □

6.3 Case of Conditional pK^t

Theorem 6.3 (Equivalence for Conditional pK^t)**.** *The following are equivalent:*

PROP-cK^t: *For any constant $C > 1$, there is a BPP-computable natural property $\mathcal{P}$ for conditional K^t with usefulness $s(n,t) = n - n/C$.*

GAP-$\mathsf{McpK^tP}$: *For any constant $C' > 1$, there is a polynomial τ such that $\mathsf{Gap}_{\tau,\delta}\mathsf{McpK^tP} \in \mathsf{promise\text{-}BPP}$, for $\delta(n,t) \leq n/C'$.*

CHAIN-cpK^t: *For any constant $D > 1$, for some polynomial p, for any $\ell \in \mathbb{N}$, for any $x_1, \ldots, x_\ell \in \{0,1\}^*$ and any $y \in \{0,1\}^*$, and all sufficiently large t (at least polynomial in the total length of all strings), we have*

$$\mathsf{pK}^t(x_1, \ldots, x_\ell \mid y) \geq \sum_{i=1}^{\ell} \mathsf{pK}^{p(t)}(x_i \mid y, x_1, \ldots, x_{i-1}) - \ell \cdot O(\log t) - N/D, \tag{28}$$

where $N = \sum_{i=1}^{\ell} |x_i|$.

Proof. The direction PROP-cK^t $\Rightarrow$ CHAIN-cpK^t is by Theorem 3.4. The direction CHAIN-cpK^t $\Rightarrow$ PROP-cK^t is by Theorem 4.3. The equivalence PROP-cK^t $\Leftrightarrow$ GAP-$\mathsf{McpK^tP}$ is by Corollary 2.19. □

6.4 If a Gap-Version of Conditional K^t Were NP-Hard

In [HIR23], it is shown under cryptographic assumptions that a certain gap-version of computing conditional K^t is NP-hard.

Theorem 6.4 ([HIR23])**.** *Assume subexponentially-secure witness encryption for NP. Then the following promise-problem is NP-hard under randomized polynomial-time (black-box) reductions: For $x, y \in \{0,1\}^*$, with $|x| = n$ and $|y| = \mathsf{poly}(n)$, and some polynomial t, distinguish between*

- $\Pi_{\text{YES}} = \{(x,y) \mid \mathsf{K}^{t(n)}(x \mid y) \leq \sqrt{n}\}$, *and*

- $\Pi_{\text{NO}} = \{(x, y) \mid \mathsf{K}^{2^{n^2}}(x \mid y) \geq n - O(1)\}$.

Note that if, for $\delta(n, t) = n/C$, for some large constant $c > 1$, and for some polynomial τ, it would be the case that $\mathsf{Gap}_{\tau,\delta}\mathsf{McK^tP} \in \mathsf{promise\text{-}BPP}$, then we would be able to solve the promise-problem from Theorem 6.4 above in $\mathsf{promise\text{-}BPP}$ as well. Thus, it is plausible that, for any large $c > 1$ and any polynomial τ, the problem $\mathsf{Gap}_{\tau,n/c}\mathsf{McK^tP}$ may in fact be NP-hard (under randomized polynomial-time reductions). This would dramatically simplify the equivalence for the Chain Rule for conditional K^t in Theorem 6.2, as we show next.

Theorem 6.5. *Suppose that, for any large $c > 1$ and any polynomial τ, the problem $\mathsf{Gap}_{\tau,n/c}\mathsf{McK^tP}$ is NP-hard. Then the following are equivalent:*

- $\mathsf{NP} = \mathsf{P}$, *and*
- *for any constant $D > 1$, for some polynomial p, for any $\ell \in \mathbb{N}$, for any $x_1, \ldots, x_\ell \in \{0,1\}^*$ with $N = \sum_{i=1}^{\ell} |x_i|$, and any $y \in \{0,1\}^*$, and all sufficiently large t (at least polynomial in $N + |y|$), we have*

$$\mathsf{K}^t(x_1, \ldots, x_\ell \mid y) \geq \sum_{i=1}^{\ell} \mathsf{K}^{p(t)}(x_i \mid y, x_1, \ldots, x_{i-1}) - \ell \cdot O(\log t) - N/D.$$

Proof. In the forward direction, $\mathsf{NP} = \mathsf{P}$ implies $\mathsf{E} \not\subset \mathsf{io\text{-}SIZE}[2^{o(n)}]$ [Kan82], as well as a P-computable natural property for conditional K^t with usefulness $s(n, t) = n - o(n)$. Hence we can apply Theorem 6.2 to get the Chain Rule for conditional K^t, as required.

In the reverse direction, by Theorem 6.2, we get from the assumed Chain Rule for conditional K^t, for any constant $C' > 1$ and some polynomial τ, a $\mathsf{promise\text{-}P}$ algorithm for $\mathsf{Gap}_{\tau,n/C'}\mathsf{McK^tP}$. The latter problem was assumed to be NP-hard (under randomized reductions). Hence, we conclude that $\mathsf{NP} \subseteq \mathsf{BPP}$. Since the Chain Rule also implies that $\mathsf{E} \not\subset \mathsf{io\text{-}SIZE}[2^{o(n)}]$, we get that $\mathsf{BPP} = \mathsf{P}$ (by Theorem 2.2). □

7 Applications

7.1 Sparse Natural Property for Conditional pK^t

Definition 7.1 (Sparse natural property for conditional pK^t)**.** *A natural property for conditional pK^t with usefulness $s(n, t)$ is a predicate $\mathcal{P} \colon \{0,1\}^* \times \{0,1\}^* \times 1^* \to \{0,1\}$ such that, for some polynomial p, we have for all large $n, m \in \mathbb{N}$ and $t \geq p(n, m)$ that*

1. *for all $x \in \{0,1\}^n$ and $y \in \{0,1\}^m$, if $\mathsf{pK}^t(x \mid y) \leq s(n, t)$, then $\mathcal{P}(x, y, 1^t) = 1$, and*
2. *for every $y \in \{0,1\}^m$, $\mathbf{Pr}_{x \in \{0,1\}^n}\left[\mathcal{P}(x, y, 1^t) = 0\right] \geq 1/2$.*

When $n \leq \log t$, we call such a property sparse.

Lemma 7.2. *For every $s(n, t) \leq n - 3$, a natural property for conditional $\mathsf{pK}^t(x \mid y)$, for $x \in \{0,1\}^n$ and $y \in \{0,1\}^m$, with usefulness $s(n, t)$ is $\mathsf{BPTIME}(\mathsf{poly}(2^n, m, t))$-computable. Hence, a sparse natural property for conditional pK^t with usefulness $s(n, t)$ is BPP-computable.*

Proof. For given $x, y \in \{0,1\}^*$, take at random strings $r_1, \ldots, r_n \in \{0,1\}^t$. For each r_i, compute $\mathsf{K}^t(x \mid r_i, y)$ by "brute force" in time $\mathsf{poly}(2^{|x|}, t, |y|)$. If the fraction of r_i's with $\mathsf{K}^t(x \mid r_i, y) \leq s(n,t)$ is at least 0.6, then accept; otherwise, reject.

Note that if x and y are such that $\mathsf{pK}^t(x \mid y) \leq s(n,t)$, then the algorithm above accepts with probability at least $1 - 2^{-\Omega(n)}$, by Chernoff bounds.

By Lemma 2.1, for any string $y \in \{0,1\}^*$ and any time bound t, the fraction of strings $x \in \{0,1\}^n$ with $\mathsf{pK}^t_{1/2}(x \mid y) \leq n-3$ is at most $1/2$. It means that for each of at least $1/2$ of strings $x \in \{0,1\}^n$, there are at most $1/2$ of strings $r \in \{0,1\}^t$ such that $\mathsf{K}^t(x \mid r, y) \leq s(n,t)$. By Chernoff bounds, getting a fraction of 0.6 such r's in a random sample of size n is at most $2^{-\Omega(n)}$. Thus our randomized algorithm will reject with high probability on each of at least $1/2$ of random strings $x \in \{0,1\}^n$. □

Theorem 7.3. $\mathsf{promise\text{-}BPP} \subseteq \mathsf{promise\text{-}P}^{\mathcal{A}}$ *for any sparse natural property* $\mathcal{A}$ *for conditional* pK^t *with usefulness* $s(n,t) \geq \Omega(n)$*. Hence, if for some* $s(n,t) \geq \Omega(n)$*, there is a* P*-computable sparse natural property* $\mathcal{A}$ *for conditional* pK^t *with usefulness* $s(n,t)$*, then* $\mathsf{promise\text{-}BPP} = \mathsf{promise\text{-}P}$*.*

Proof. For simplicity, assume that we have a P-computable sparse natural property $\mathcal{P}$ with usefulness $s(n,t) \geq n/2$; a similar argument would work for $s(n,t) \geq \Omega(n)$. Under our assumption, we will show how to solve the canonical $\mathsf{promise\text{-}BPP}$ complete problem CAPP: given a Boolean circuit $C\colon \{0,1\}^n \to \{0,1\}$ of size at most n, estimate its acceptance probability $\Pr_{z \in \{0,1\}^n}[C(x) = 1]$ to within an additive constant error, say $1/8$; see, e.g., [Vad12].

For strings $x_1, \ldots, x_n \in \{0,1\}^{10 \log n}$ to be determined, consider the following generator from $10 \log n$ to n bits: for $r \in \{0,1\}^{10 \log n}$,

$$G^{x_1,\ldots,x_n}(r) = x_1 \cdot r, \ldots, x_n \cdot r,$$

where $x \cdot y$ denotes the inner product modulo 2 of binary strings x and y.

Suppose some circuit $C\colon \{0,1\}^n \to \{0,1\}$ (of size at most n) distinguishes between the outputs $G^{x_1,\ldots,x_n}(r)$, for uniformly random seeds $r \in \{0,1\}^{10 \log}$, and the uniform distribution $\mathcal{U}_n$, with the distinguishing probability at least $1/8$. (In other words, the generator G fails to solve CAPP on the circuit C.) We will argue this implies the following.

Claim 7.4. *There exists an* $1 \leq i \leq n$ *such that*

$$\mathsf{pK}^t(x_i \mid C, x_1, \ldots, x_{i-1}) \leq 4 \log n,$$

for some $t \leq \mathsf{poly}(n)$*.*

Proof of Claim. This can be deduced from the pK^t Reconstruction lemma (Lemma 2.5), but we will give a self-contained argument, which will be useful for us later.

First, by a hybrid argument (see, e.g., [Vad12]), there must exist an index $1 \leq i \leq n$ such that C distinguishes between

$$x_1 \cdot r, \ldots, x_{i-1} \cdot r, x_i \cdot r, \mathcal{U}_{n-i}$$

and

$$x_1 \cdot r, \ldots, x_{i-1} \cdot r, \mathcal{U}_1, \mathcal{U}_{n-i},$$

for a random $r \in \{0,1\}^{10 \log n}$, with the distinguishing probability at least $1/(8n)$.

Next, by Yao's distinguisher-to-predictor reduction, we get a randomized predictor algorithm C' that on input r computes $x_i \cdot r$ with probability at least $1/2 + 1/(8n)$ over r and its internal randomness, given (as advice) the bits $x_1 \cdot r, \ldots, x_{i-1} \cdot r$. This $C'(r)$ samples at random a bit $b \in \{0,1\}$ and a string $w \in \{0,1\}^{n-i}$, and simulates $C(x_1 \cdot r, \ldots, x_{i-1} \cdot r, b, w)$, outputting b if C accepts, and $1-b$ if C rejects.

By an averaging argument, with probability at least $1/(16n)$ over fixing the internal randomness b, w, we get a deterministic circuit $C''_{b,w}(r)$ that computes $x_i \cdot r$ with probability at least $1/2+1/(16n)$ over r, when given as advice $x_1, \ldots, x_{i-1}$ (so that it can compute the required bits $x_1 \cdot r, \ldots, x_{i-1} \cdot r$). Call b, w *good* if $C''_{b,w}(r)$ is correct with probability at least $1/2 + 1/(16n)$ in computing $x_i \cdot r$.

Next, for given b, w, we build a list of all $z \in \{0,1\}^{10 \log n}$ such that

$$\Pr_r[C''_{b,w}(r) = z \cdot r] \geq 1/2 + 1/(16n). \tag{29}$$

We do so by enumerating all strings z, computing the probability in (29) for each z, and keeping only those z that satisfy (29). This can be done in deterministic time $\mathsf{poly}(n)$, say at most n^{22}. By the well-known combinatorial bound on the list size of Hadamard error-correcting codes, the size of such a list of z's is at most $(8n)^2$.[8]

For good b, w, such a list of z's must contain the string x_i, which can be identified by its index on the list. Since the list is of size at most $64n^2$, such an index is of binary length at most $2 \log n + 6$.

If we repeatedly sample random b, w for $32n$ rounds, at least one of b, w will be good with probability at least $1 - (1 - 1/(16n))^{32n} \geq 1 - e^{-2} \geq 2/3$. Assuming this happened, the description of x_i will consist of about $\log n + 5$ bits to specify the succefull round, plus $2 \log n + 6$ bits to specify x_i on the list produced in that round. It follows that $\mathsf{pK}^t(x_i \mid C, x_1, \ldots, x_{i-1}) \leq 4 \log n$, for $t = \mathsf{poly}(n)$, say $t \leq n^{24}$. □

By the claim, if we have $x_1, \ldots, x_n$ with $\mathsf{pK}^t(x_i \mid C, x_1, \ldots, x_{i-1}) > 5 \log n$ for all $1 \leq i \leq n$, then $G^{x_1,\ldots,x_n}$ will fool the circuit C. That is, we get a PRG for a given circuit C, a *targeted* PRG in the terminology of [Gol11].

We can find such x_i's one by one, starting with x_1, by enumerating over all strings $z \in \{0,1\}^{10 \log n}$, until we find the first one such that $\mathcal{A}(z, C \circ x_1 \circ \cdots \circ x_{i-1}, 1^t) = 0$, for $t = n^{24}$. By the assumption on the natural property algorithm $\mathcal{A}$, each such z has the required conditional pK^t complexity greater than $5 \log n$. To find such $x_1, \ldots, x_n$ takes time at most $\mathsf{poly}(n)$. It follows that we can solve CAPP in polynomial time, and so $\mathsf{promise\text{-}BPP} = \mathsf{promise\text{-}P}$.

Finally, observe that to find the required strings $x_1, \ldots, x_n$, we just need oracle access to the natural property $\mathcal{A}$. So we get an (adaptive) polynomial-time Turing reduction from CAPP to any sparse natural property for conditional pK^t with usefulness $s(n,t) \geq \Omega(n)$, as required. □

Remark 7.5. *It is interesting to contrast the result in Theorem 7.3 with the result of [Buh+05] (see also [All+06]) that* $\mathsf{BPP} \subseteq \mathsf{P}^{\mathcal{K}}$*, where* $\mathcal{K}$ *is a natural property for time-unbounded Kolmoogorov complexity* K*, with usefulness* $s(n) \geq \Omega(n)$*. Both results are proved by using the oracle adaptively, for a polynomial number of steps. The crucial difference is that, in their case, the oracle* $\mathcal{K}$ *is uncomputable, whereas in our case, a required sparse natural property* $\mathcal{A}$ *is* BPP*-computable.*

[8] Each such z corresponds to a Fourier coefficient of the Boolean function $C''_{b,w}(r)$ of weight at least $1/(8n)$. By Parseval's identity, the sum of the squares of all Fourier coefficients of a given Boolean function is 1. Hence, the number of "heavy" z's is at most $(8n)^2$.

7.2 pK^t-Based promise-BPP Complete Problem

The sparse natural property for conditional pK^t considered above is not a promise-problem (as it has no well-defined set of No-instances). We can easily modify it to be a promise-BPP problem as follows.

Definition 7.6 (Promise-BPP problem for estimating conditional pK^t on short inputs)**.** *Consider the following* Gap-pK^t*-problem. For $x, y \in \{0,1\}^*$ and $t \in \mathbb{N}$, with $t = \mathsf{poly}(|y|)$ and $|x| \leq \log t$, we have the following Yes and No instances:*

- $\Pi_{\text{YES}} = \{(x, y, 1^t) \mid \mathsf{pK}^t_{2/3}(x \mid y) \leq |x|/2\}$
- $\Pi_{\text{NO}} = \{(x, y, 1^t) \mid \mathsf{pK}^t_{1/2}(x \mid y) > |x| - 3\}$

Theorem 7.7. *The* Gap-pK^t*-problem of Definition 7.6 above is* promise-BPP*-complete (under polynomial-time Turing reductions).*

Proof. That this problem is in promise-BPP was already argued in the proof of Lemma 7.2 above. The promise-BPP-hardness for this problem follows from Theorem 7.3. □

We also get that the following problem to approximate the conditional K^t complexity over random conditioned strings is promise-BPP-complete.

Corollary 7.8. *The problem to estimate, for given $x, y \in \{0,1\}^*$ and $s, t \in \mathbb{N}$ such that $|x| \leq \log t$, the following probability*

$$\Pr_{r \in \{0,1\}^t}\left[\mathsf{K}^t(x \mid r, y) \leq s\right]$$

to within an additive error less than $1/6$ is promise-BPP*-complete (under polynomial-time Turing reductions).*

Proof. This is immediate by Theorem 7.7 and the definition of pK^t_λ. □

Remark 7.9. *We note that the work by Liu and Pass [LP22a] also studies derandomization of* promise-BPP *through the lens of (Levin's version) of Kolmogorov complexity. Their main result is that* promise-BPP = promise-P *iff a certain gap-version of computing conditional Levin's complexity $\mathsf{Kt}(x \mid y)$ is almost-everywhere worst-case hard (for almost all strings $y \in \{0,1\}^*$) with respect to probabilistic polynomial-time algorithms. In contrast, we characterize* promise-BPP = promise-P *in terms of worst-case* easiness *(being in* promise-P*) of a* promise-BPP *computable version of conditional $\mathsf{pK}^t(x \mid y)$ problem (for logarithmically short inputs x).*

7.3 On Derandomizing Yao's Distinguisher-to-Predictor Transformation

Here we observe that Theorem 7.3 can be used to give an alternative proof of (one direction) of the recent result by Li, Pyne, and Tell [LPT24]. One of their main results is that derandomizing Yao's distinguisher-to-predictor transformation (of the kind described in our proof of Theorem 7.3) is *equivalent* to proving promise-BPP = promise-P. We will show the following.

Lemma 7.10 ([LPT24])**.** *If Yao's distinguisher-to-predictor transformation can be made deterministic polynomial time, then* promise-BPP = promise-P*.*

Proof. Given a Boolean circuit $C\colon \{0,1\}^n \to \{0,1\}$ of size n, we will try to fool it with the generator $G^{x_1,\ldots,x_n}\colon \{0,1\}^{10\log n} \to \{0,1\}^n$ as in Theorem 7.3. Assuming that C is *not* fooled by this G, we follow the proof of Theorem 7.3 to get that, for some $1 \le i \le n$,

$$\mathsf{K}^t(x_i \mid C, x_1, \ldots, x_{i-1}) \le 4\log n,$$

rather than $\mathsf{pK}^t(x_i \mid C, x_1, \ldots, x_{i-1}) \le 4\log n$ of Claim 7.4. The reason is that the *only randomized step* in the proof of Claim 7.4 was Yao's distinguisher-to-predictor transformation. Under our current assumption that this step can be made in deterministic polynomial time, we get a bound on the *deterministic* conditional K^t description of string x_i.

Thus, to fool C, we just need to find n strings $x_1, \ldots, x_n \in \{0,1\}^{10\log n}$ such that, for each $1 \le i \le n$,

$$\mathsf{K}^t(x_i \mid C, x_1, \ldots, x_{i-1}) > 4\log n.$$

This can be accomplished in deterministic polynomial time, using a brute-force algorithm that, for each $i = 1, \ldots, n$, looks for the lexicographically first string in $\{0,1\}^{10\log n}$ of large conditional K^t complexity, for $t \le n^{24}$. We conclude that $\mathsf{CAPP} \in \mathsf{promise\text{-}P}$. □

7.4 If Conditional pK^t Were Easy

As noted earlier in Section 6.4, it is plausible (and true under cryptographic assumptions) that a gap-version of computing conditional $\mathsf{K}^t(x \mid y)$ might be NP-hard. It is reasonable to assume that a gap-version of computing conditional $\mathsf{pK}^t(x \mid y)$ might also be NP-hard. However, given the lack of actual proofs of NP-hardness for these problems, we may ask ourselves what interesting algorithmic consequences would follow if we were to assume the *easisess* of these problems instead.

Ideally, assuming such easiness, we would like to conclude that SAT is also easy (thereby essentially establishing the NP-hardness of the corresponding gap-problem). We cannot achieve this yet, but we can show the following.

Theorem 7.11. *Suppose that, for $\delta(n,t) \le \varepsilon n$, for any constant $\varepsilon > 0$, there is a polynomial τ such that $\mathsf{Gap}_{\tau,\delta}\mathsf{McpK^tP} \in \mathsf{promise\text{-}P}$. Then we have*

1. *$\mathsf{E} \not\subset \mathsf{io\text{-}SIZE}[2^{o(n)}]$, and*
2. *$(1/3)\text{-}\mathsf{CRA} \in \mathsf{FP}$.*

Proof. We give two proofs. The first one will use the chain rule.

Under our assumption, we get by Corollary 2.19 the existence of a BPP-computable (actually, P-computable) natural property for conditional $\mathsf{K}^t(x \mid y)$ for $x \in \{0,1\}^n$ with usefulness $s(n,t) = n - \gamma n$, for any sufficiently small $\gamma > 0$ (dependent on ε).

Next, by Theorem 3.4, we get a chain rule for conditional pK^t: for some polynomial p, for any $\ell \in \mathbb{N}$, for any $x_1, \ldots, x_\ell \in \{0,1\}^*$ and any $y \in \{0,1\}^*$, and all sufficiently large t (at least polynomial in the total length of all strings), we have

$$\mathsf{pK}^t(x_1, \ldots, x_\ell \mid y) \ge \sum_{i=1}^{\ell} \mathsf{pK}^{p(t)}(x_i \mid y, x_1, \ldots, x_{i-1}) - \ell \cdot O(\log t) - N/D, \tag{30}$$

where $N = \sum_{i=1}^{\ell} |x_i|$ and $D > 1$ is any sufficiently large constant (dependent on γ).

By setting $\ell = N/(c \log t)$ and $|x_1| = \cdots = |x_\ell| = N/\ell = c \log t$, for a sufficiently large constant $c > 0$, we can use the chain rule above to construct, in deterministic polynomial time, a string $x = x_1, \ldots, x_\ell \in \{0,1\}^N$ of complexity $\mathsf{pK}^t(x \mid y) \geq (1-\tau)N$, for any given string $y \in \{0,1\}^*$ and any constant $\tau > 0$. We do so by constructing each $x_1, \ldots, x_\ell$ in turn, by brute-forcing over all possible strings in $\{0,1\}^{c \log t}$ and taking the first one of high conditional $\mathsf{pK}^{p(t)}$ complexity; the latter can be done in deterministic polynomial time by our assumption that $\mathsf{Gap}_{\tau,\delta}\mathsf{McpK}^{\mathsf{t}}\mathsf{P} \in \mathsf{promise\text{-}P}$.

It follows that we can construct in polynomial time a string $a \in \{0,1\}^{2^n}$ of $\mathsf{K}^{2^{O(n)}}(a) \geq \Omega(2^n)$, and hence a requires circuit size at least $\Omega(2^n/n)$. This implies item (1).

Similarly, constructing in polynomial time a string $b \in \{0,1\}^n$ with high conditional complexity $\mathsf{pK}^t(b \mid y)$ is equivalent to solving the range avoidance problem (1/3)-CRA by Lemma 2.23. This implies item (2).

The second proof. The assumption that $\mathsf{Gap}_{\tau,\delta}\mathsf{McpK}^{\mathsf{t}}\mathsf{P} \in \mathsf{promise\text{-}P}$ implies that there is a P-computable natural property for conditional pK^t with usefulness $s(n,t) \geq n/2$. By Theorem 7.3, we conclude that $\mathsf{promise\text{-}BPP} = \mathsf{promise\text{-}P}$. By the standard "search-to-decision" reduction for $\mathsf{promise\text{-}BPP}$ (see, e.g., [Vad12]), we can construct in deterministic polynomial time a string $x \in \{0,1\}^n$ that is accepted by a given circuit $C\colon \{0,1\}^n \to \{0,1\}$, provided C accepts a significant fraction of n-bit strings (e.g., at least $1/2$). In our case, we use the assumed polynomial-time algorithm A for $\mathsf{Gap}_{\tau,\delta}\mathsf{McpK}^{\mathsf{t}}\mathsf{P}$ to look for a string $x \in \{0,1\}^n$ such that $A(x, y, 1^s, 1^t)$ rejects, for given $y \in \{0,1\}^*$ and parameters s and t. By choosing s and t appropriately, we can find strings of high (conditional) pK^t complexity in deterministic polynomial time. This suffices for items (1) and (2), as noted above. □

8 Concluding Remarks

The complexity of (vairants of) K^t appears to be the right notion in the context of (variant) chain rules for K^t. By [Hir+23] and [LP20], the average-case SoI for pK^t over polynomial-time samplable distributions is equivalent to the (error-prone) average-case easiness of (a variant of) $\mathsf{MK}^t\mathsf{P}$. By [LS24], it is also equivalent to computing a certain promise-version for conditional pK^t. By [LP24], it is also equivalent to the worst-case easiness of a certain (somewhat complicated) version of the promise-problem $\mathsf{GapMK}^t\mathsf{P}$ (where the inputs are restricted to be of "shallow computational depth"). Our work shows that the worst-case (multi-string) chain rule for conditional pK^t is equivalent to $\mathsf{Gap}_{\mathsf{poly},o(n)}\mathsf{McpK}^{\mathsf{t}}\mathsf{P} \in \mathsf{promise\text{-}BPP}$, and the chain rule for conditional K^t is equivalent to $\mathsf{Gap}_{\mathsf{poly},o(n)}\mathsf{McK}^{\mathsf{t}}\mathsf{P} \in \mathsf{promise\text{-}P}$ (and exponential circuit lower bounds for E).

A very interesting open question is whether $\mathsf{Gap}_{\mathsf{poly},o(n)}\mathsf{McK}^{\mathsf{t}}\mathsf{P}$ can be shown NP-hard unconditionally (without any cryptographic assumptions). Such a result would provide an extremely clean equivalence: the chain rule for conditional K^t is equivalent to $\mathsf{P} = \mathsf{NP}$.

Another open question is to understand what complexity assumption would be equivalent to the worst-case SoI, or chain rule on a constant number of strings, for (conditional) K^t or pK^t. One specific technical challenge is to try to derive a polynomial-time computable natural property, say for K^t, from the assumed SoI for K^t. It is possible to get a $2^{O(n/\log n)}$ computable natural property from SoI for K^t (see Appendix B), but it is not clear how to get time, say $2^{O(\sqrt{n})}$.

References

[All+06] Eric Allender, Harry Buhrman, Michal Koucký, Dieter van Melkebeek, and Detlef Ronneburger. "Power from Random Strings". In: *SIAM J. Comput.* 35.6 (2006), pp. 1467–1493. DOI: 10.1137/050628994.

[Bin+22] Eric Binnendyk, Marco Carmosino, Antonina Kolokolova, R. Ramyaa, and Manuel Sabin. "Learning with Distributional Inverters". In: *International Conference on Algorithmic Learning Theory, 29 March - 1 April 2022, Paris, France*. Ed. by Sanjoy Dasgupta and Nika Haghtalab. Vol. 167. Proceedings of Machine Learning Research. PMLR, 2022, pp. 90–106.

[Buh+05] Harry Buhrman, Lance Fortnow, Ilan Newman, and Nikolai K. Vereshchagin. "Increasing Kolmogorov Complexity". In: *STACS 2005, 22nd Annual Symposium on Theoretical Aspects of Computer Science, Stuttgart, Germany, February 24-26, 2005, Proceedings*. Ed. by Volker Diekert and Bruno Durand. Vol. 3404. Lecture Notes in Computer Science. Springer, 2005, pp. 412–421. DOI: 10.1007/978-3-540-31856-9_34.

[Car+16] Marco L. Carmosino, Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova. "Learning Algorithms from Natural Proofs". In: *Conference on Computational Complexity* (CCC). 2016, 10:1–10:24.

[Car+17] Marco L. Carmosino, Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova. "Agnostic Learning from Tolerant Natural Proofs". In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2017, August 16-18, 2017, Berkeley, CA, USA*. Ed. by Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala. Vol. 81. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017, 35:1–35:19. DOI: 10.4230/LIPICS.APPROX-RANDOM.2017.35.

[CHV22] Lijie Chen, Shuichi Hirahara, and Neekon Vafa. "Average-Case Hardness of NP and PH from Worst-Case Fine-Grained Assumptions". In: *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, January 31 - February 3, 2022, Berkeley, CA, USA*. Ed. by Mark Braverman. Vol. 215. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 45:1–45:16. DOI: 10.4230/LIPIcs.ITCS.2022.45.

[Gaj+23] Karthik Gajulapalli, Alexander Golovnev, Satyajeet Nagargoje, and Sidhant Saraogi. "Range Avoidance for Constant Depth Circuits: Hardness and Algorithms". In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2023, September 11-13, 2023, Atlanta, Georgia, USA*. Ed. by Nicole Megow and Adam D. Smith. Vol. 275. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023, 65:1–65:18. DOI: 10.4230/LIPICS.APPROX/RANDOM.2023.65.

[Gar+13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. "Witness encryption and its applications". In: *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*. Ed. by Dan Boneh, Tim Roughgarden, and Joan Feigenbaum. ACM, 2013, pp. 467–476. DOI: 10.1145/2488608.2488667.

[GK22] Halley Goldberg and Valentine Kabanets. "A Simpler Proof of the Worst-Case to Average-Case Reduction for Polynomial Hierarchy via Symmetry of Information". In: *Electron. Colloquium Comput. Complex.* TR22-007 (2022). ECCC: TR22-007.

[GK23] Halley Goldberg and Valentine Kabanets. "Improved Learning from Kolmogorov Complexity". In: *38th Computational Complexity Conference, CCC 2023, July 17-20, 2023, Warwick, UK*. Ed. by Amnon Ta-Shma. Vol. 264. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023, 12:1–12:29. DOI: 10.4230/LIPICS.CCC.2023.12.

[GL89] Oded Goldreich and Leonid A. Levin. "A Hard-Core Predicate for all One-Way Functions". In: *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washington, USA*. Ed. by David S. Johnson. ACM, 1989, pp. 25–32. DOI: 10.1145/73007.73010.

[GLW22] Venkatesan Guruswami, Xin Lyu, and Xiuhan Wang. "Range Avoidance for Low-Depth Circuits and Connections to Pseudorandomness". In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2022, September 19-21, 2022, University of Illinois, Urbana-Champaign, USA (Virtual Conference)*. Ed. by Amit Chakrabarti and Chaitanya Swamy. Vol. 245. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 20:1–20:21. DOI: 10.4230/LIPICS.APPROX/RANDOM.2022.20.

[Gol+22] Halley Goldberg, Valentine Kabanets, Zhenjian Lu, and Igor C. Oliveira. "Probabilistic Kolmogorov Complexity with Applications to Average-Case Complexity". In: *37th Computational Complexity Conference, CCC 2022, July 20-23, 2022, Philadelphia, PA, USA*. Ed. by Shachar Lovett. Vol. 234. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 16:1–16:60. DOI: 10.4230/LIPICS.CCC.2022.16.

[Gol11] Oded Goldreich. "In a World of P=BPP". In: *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation - In Collaboration with Lidor Avigad, Mihir Bellare, Zvika Brakerski, Shafi Goldwasser, Shai Halevi, Tali Kaufman, Leonid Levin, Noam Nisan, Dana Ron, Madhu Sudan, Luca Trevisan, Salil Vadhan, Avi Wigderson, David Zuckerman*. Ed. by Oded Goldreich. Vol. 6650. Lecture Notes in Computer Science. Springer, 2011, pp. 191–232. DOI: 10.1007/978-3-642-22670-0_20.

[Hås+99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. "A Pseudorandom Generator from any One-way Function". In: *SIAM J. Comput.* 28.4 (1999), pp. 1364–1396. DOI: 10.1137/S0097539793244708.

[Hir+23] Shuichi Hirahara, Rahul Ilango, Zhenjian Lu, Mikito Nanashima, and Igor C. Oliveira. "A Duality between One-Way Functions and Average-Case Symmetry of Information". In: *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*. Ed. by Barna Saha and Rocco A. Servedio. ACM, 2023, pp. 1039–1050. DOI: 10.1145/3564246.3585138.

[Hir+24] Shuichi Hirahara, Valentine Kabanets, Zhenjian Lu, and Igor C. Oliveira. "Exact Search-To-Decision Reductions for Time-Bounded Kolmogorov Complexity". In: *39th Computational Complexity Conference, CCC 2024, July 22-25, 2024, Ann Arbor, MI, USA*. Ed. by Rahul Santhanam. Vol. 300. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024, 29:1–29:56. DOI: 10.4230/LIPICS.CCC.2024.29.

[Hir18] Shuichi Hirahara. "Non-Black-Box Worst-Case to Average-Case Reductions within NP". In: *Symposium on Foundations of Computer Science* (FOCS). 2018, pp. 247–258.

[Hir20a] Shuichi Hirahara. "Non-Disjoint Promise Problems from Meta-Computational View of Pseudorandom Generator Constructions". In: *35th Computational Complexity Conference, CCC 2020, July 28-31, 2020, Saarbrücken, Germany (Virtual Conference)*. Ed. by Shubhangi Saraf. Vol. 169. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020, 20:1–20:47. DOI: 10.4230/LIPICS.CCC.2020.20.

[Hir20b] Shuichi Hirahara. "Unexpected hardness results for Kolmogorov complexity under uniform reductions". In: *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*. Ed. by Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy. ACM, 2020, pp. 1038–1051. DOI: 10.1145/3357713.3384251.

[Hir21] Shuichi Hirahara. "Average-case hardness of NP from exponential worst-case hardness assumptions". In: *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*. Ed. by Samir Khuller and Virginia Vassilevska Williams. ACM, 2021, pp. 292–302. DOI: 10.1145/3406325.3451065.

[Hir22] Shuichi Hirahara. "Symmetry of Information from Meta-Complexity". In: *Computational Complexity Conference* (CCC). 2022, 26:1–26:41. DOI: 10.4230/LIPIcs.CCC.2022.26.

[HIR23] Yizhi Huang, Rahul Ilango, and Hanlin Ren. "NP-Hardness of Approximating Meta-Complexity: A Cryptographic Approach". In: *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*. Ed. by Barna Saha and Rocco A. Servedio. ACM, 2023, pp. 1067–1075. DOI: 10.1145/3564246.3585154.

[Hir23] Shuichi Hirahara. "Capturing One-Way Functions via NP-Hardness of Meta-Complexity". In: *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*. Ed. by Barna Saha and Rocco A. Servedio. ACM, 2023, pp. 1027–1038. DOI: 10.1145/3564246.3585130.

[HN21] Shuichi Hirahara and Mikito Nanashima. "On Worst-Case Learning in Relativized Heuristica". In: *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*. IEEE, 2021, pp. 751–758. DOI: 10.1109/FOCS52979.2021.00078.

[IKV23] Russell Impagliazzo, Valentine Kabanets, and Ilya Volkovich. "The Power of Natural Properties as Oracles". In: *Comput. Complex.* 32.2 (2023), p. 6. DOI: 10.1007/S00037-023-00241-0.

[IKW02] Russell Impagliazzo, Valentine Kabanets, and Avi Wigderson. "In Search of an Easy Witness: Exponential Time vs. Probabilistic Polynomial Time". In: *J. Comput. Syst. Sci.* 65.4 (2002), pp. 672–694. DOI: 10.1016/S0022-0000(02)00024-7.

[Ila23] Rahul Ilango. "SAT Reduces to the Minimum Circuit Size Problem with a Random Oracle". In: *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*. IEEE, 2023, pp. 733–742. DOI: 10.1109/FOCS57990.2023.00048.

[ILW23] Rahul Ilango, Jiatu Li, and R. Ryan Williams. "Indistinguishability Obfuscation, Range Avoidance, and Bounded Arithmetic". In: *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*. Ed. by Barna Saha and Rocco A. Servedio. ACM, 2023, pp. 1076–1089. DOI: 10.1145/3564246.3585187.

[IRS22] Rahul Ilango, Hanlin Ren, and Rahul Santhanam. "Robustness of average-case meta-complexity via pseudorandomness". In: *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 - 24, 2022*. Ed. by Stefano Leonardi and Anupam Gupta. ACM, 2022, pp. 1575–1583. DOI: 10.1145/3519935.3520051.

[IW97] Russell Impagliazzo and Avi Wigderson. "$P = BPP$ if E Requires Exponential Circuits: Derandomizing the XOR Lemma". In: *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997*. Ed. by Frank Thomson Leighton and Peter W. Shor. ACM, 1997, pp. 220–229. DOI: 10.1145/258533.258590.

[Jer04] Emil Jerábek. "Dual weak pigeonhole principle, Boolean complexity, and derandomization". In: *Ann. Pure Appl. Log.* 129.1-3 (2004), pp. 1–37. DOI: 10.1016/J.APAL.2003.12.003.

[Jeř07] Emil Jeřábek. "On independence of variants of the weak pigeonhole principle". In: *J. Log. Comput.* 17.3 (2007), pp. 587–604. DOI: 10.1093/logcom/exm017.

[Kan82] Ravi Kannan. "Circuit-Size Lower Bounds and Non-Reducibility to Sparse Sets". In: *Inf. Control.* 55.1-3 (1982), pp. 40–56. DOI: 10.1016/S0019-9958(82)90382-5.

[Kar24] Ari Karchmer. "Distributional PAC-Learning from Nisan's Natural Proofs". In: *15th Innovations in Theoretical Computer Science Conference, ITCS 2024, January 30 to February 2, 2024, Berkeley, CA, USA*. Ed. by Venkatesan Guruswami. Vol. 287. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024, 68:1–68:23. DOI: 10.4230/LIPICS.ITCS.2024.68.

[KC00] Valentine Kabanets and Jin-yi Cai. "Circuit minimization problem". In: *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA*. Ed. by F. Frances Yao and Eugene M. Luks. ACM, 2000, pp. 73–79. DOI: 10.1145/335305.335314.

[Kle+21] Robert Kleinberg, Oliver Korten, Daniel Mitropolsky, and Christos H. Papadimitriou. "Total Functions in the Polynomial Hierarchy". In: *12th Innovations in Theoretical Computer Science Conference, ITCS 2021, January 6-8, 2021, Virtual Conference*. Ed. by James R. Lee. Vol. 185. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021, 44:1–44:18. DOI: 10.4230/LIPICS.ITCS.2021.44.

[Ko91] Ker-I Ko. "On the Complexity of Learning Minimum Time-Bounded Turing Machines". In: *SIAM J. Comput.* 20.5 (1991), pp. 962–986. DOI: 10.1137/0220059.

[Kor21] Oliver Korten. "The Hardest Explicit Construction". In: *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*. IEEE, 2021, pp. 433–444. DOI: 10.1109/FOCS52979.2021.00051.

[Kra01] Jan Krajícek. "Tautologies from pseudo-random generators". In: *Bull. Symb. Log.* 7.2 (2001), pp. 197–212. DOI: 10.2307/2687774.

[Kra04] Jan Krajícek. "Dual weak pigeonhole principle, pseudo-surjective functions, and provability of circuit lower bounds". In: *J. Symb. Log.* 69.1 (2004), pp. 265–286. DOI: 10.2178/JSL/1080938841.

[Lev03] Leonid A. Levin. "The Tale of One-Way Functions". In: *Probl. Inf. Transm.* 39.1 (2003), pp. 92–103. DOI: 10.1023/A%3A1023634616182.

[LM93] Luc Longpré and Sarah Mocas. "Symmetry of Information and One-Way Functions". In: *Inf. Process. Lett.* 46.2 (1993), pp. 95–100. DOI: 10.1016/0020-0190(93)90204-M.

[LO22] Zhenjian Lu and Igor C. Oliveira. "Theory and Applications of Probabilistic Kolmogorov Complexity". In: *Bull. EATCS* 137 (2022).

[LP20] Yanyi Liu and Rafael Pass. "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*. 2020, pp. 1243–1254. DOI: 10.1109/FOCS46700.2020.00118.

[LP22a] Yanyi Liu and Rafael Pass. "Characterizing Derandomization Through Hardness of Levin-Kolmogorov Complexity". In: *37th Computational Complexity Conference, CCC 2022, July 20-23, 2022, Philadelphia, PA, USA*. Ed. by Shachar Lovett. Vol. 234. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 35:1–35:17. DOI: 10.4230/LIPICS.CCC.2022.35.

[LP22b] Yanyi Liu and Rafael Pass. "On One-Way Functions from NP-Complete Problems". In: *37th Computational Complexity Conference, CCC 2022, July 20-23, 2022, Philadelphia, PA, USA*. Ed. by Shachar Lovett. Vol. 234. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 36:1–36:24. DOI: 10.4230/LIPICS.CCC.2022.36.

[LP24] Yanyi Liu and Rafael Pass. "On One-Way Functions, the Worst-Case Hardness of Time-Bounded Kolmogorov Complexity, and Computational Depth". In: *Theory of Cryptography - 22nd International Conference, TCC 2024, Milan, Italy, December 2-6, 2024, Proceedings, Part I*. Ed. by Elette Boyle and Mohammad Mahmoody. Vol. 15364. Lecture Notes in Computer Science. Springer, 2024, pp. 222–252. DOI: 10.1007/978-3-031-78011-0_8.

[LPT24] Jiatu Li, Edward Pyne, and Roei Tell. "Distinguishing, Predicting, and Certifying: On the Long Reach of Partial Notions of Pseudorandomness". In: *65th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2024, Chicago, IL, USA, October 27-30, 2024*. IEEE, 2024, pp. 1–13. DOI: 10.1109/FOCS61266.2024.00095.

[LR05] Troy Lee and Andrei E. Romashchenko. "Resource bounded symmetry of information revisited". In: *Theor. Comput. Sci.* 345.2-3 (2005), pp. 386–405. DOI: 10.1016/J.TCS.2005.07.017.

[LS24] Zhenjian Lu and Rahul Santhanam. "Impagliazzo's Worlds Through the Lens of Conditional Kolmogorov Complexity". In: *51st International Colloquium on Automata, Languages, and Programming, ICALP 2024, July 8-12, 2024, Tallinn, Estonia*. Ed. by Karl Bringmann, Martin Grohe, Gabriele Puppis, and Ola Svensson. Vol. 297. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024, 110:1–110:17. DOI: 10.4230/LIPICS.ICALP.2024.110.

[LV19] Ming Li and Paul M. B. Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications, 4th Edition*. Texts in Computer Science. Springer, 2019. DOI: 10.1007/978-3-030-11298-1.

[LW95] Luc Longpré and Osamu Watanabe. "On Symmetry of Information and Polynomial Time Invertibility". In: *Inf. Comput.* 121.1 (1995), pp. 14–22. DOI: 10.1006/inco.1995.1120.

[NW94] Noam Nisan and Avi Wigderson. "Hardness vs Randomness". In: *J. Comput. Syst. Sci.* 49.2 (1994), pp. 149–167. DOI: 10.1016/S0022-0000(05)80043-1.

[OS17] Igor C. Oliveira and Rahul Santhanam. "Conspiracies Between Learning Algorithms, Circuit Lower Bounds, and Pseudorandomness". In: *32nd Computational Complexity Conference, CCC 2017, July 6-9, 2017, Riga, Latvia*. Ed. by Ryan O'Donnell. Vol. 79. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017, 18:1–18:49. DOI: 10.4230/LIPICS.CCC.2017.18.

[Per07] Sylvain Perifel. "Symmetry of Information and Nonuniform Lower Bounds". In: *Computer Science - Theory and Applications, Second International Symposium on Computer Science in Russia, CSR 2007, Ekaterinburg, Russia, September 3-7, 2007, Proceedings*. Ed. by Volker Diekert, Mikhail V. Volkov, and Andrei Voronkov. Vol. 4649. Lecture Notes in Computer Science. Springer, 2007, pp. 315–327. DOI: 10.1007/978-3-540-74510-5_32.

[Ron04] Detlef Ronneburger. "Kolmogorov Complexity and Derandomization". PhD thesis. Rutgers University, 2004.

[RR97] Alexander A. Razborov and Steven Rudich. "Natural Proofs". In: *J. Comput. Syst. Sci.* 55.1 (1997), pp. 24–35. DOI: 10.1006/JCSS.1997.1494.

[RSW22] Hanlin Ren, Rahul Santhanam, and Zhikun Wang. "On the Range Avoidance Problem for Circuits". In: *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*. IEEE, 2022, pp. 640–650. DOI: 10.1109/FOCS54457.2022.00067.

[Vad12] Salil P. Vadhan. "Pseudorandomness". In: *Found. Trends Theor. Comput. Sci.* 7.1-3 (2012), pp. 1–336. DOI: 10.1561/0400000010.

[ZL70] A.K. Zvonkin and L.A. Levin. "The Complexity of Finite Objects and the Development of the Concepts of Information and Randomness by means of the Theory of Algorithms". In: *Russian Mathematical Surveys* 25 (1970), pp. 83–124.

A Inverting One-Way Functions from a Chain Rule

Longpré and Watanabe [LW95] proved that a chain rule for K^t, for two strings (Symmetry of Information for K^t) implies that no one-way functions exist. That is, every one-way function candidate $f\colon \{0,1\}^* \to \{0,1\}^*$ (which is length-preserving) can be inverted by a randomized algorithm, with high probability over uniformly random inputs x to f. Using Theorem 4.1 and Theorem 5.2, we get the following version of their result.

Theorem A.1. *Assume the Chain Rule for K^t as in Eq. (7) of Theorem 3.1, for some $\delta(n) \leq o(n^{1/3})$. Then any function family f of length-preserving functions $f = \{f_n\colon \{0,1\}^n \to \{0,1\}^n\}$,*

computable in polynomial time, can be inverted in deterministic polynomial time on almost all inputs.

Proof Sketch. By [Hås+99], one-way functions exist if and only if there are secure PRGs with polynomial stretch, i.e., polynomial-time computable $G\colon \{0,1\}^{n^\varepsilon} \to \{0,1\}^n$, for some $0 < \varepsilon < 1$, such that no randomized polynomial-time algorithm can distinguish the outputs of G from the uniform distribution with a distinguishing probability at least $1/\mathsf{poly}(n)$, for some polynomial $\mathsf{poly}(n)$. Note that, for every seed $z \in \{0,1\}^{n^\varepsilon}$, we have $\mathsf{K}^t(G(z)) \leq |z| + O(1)$, for t being the polynomial runtime of the generator G. Hence, a polynomial-time computable natural property for K^t on n-bit strings with usefulness $s(n) = n^\varepsilon$ would be a distinguisher for G.

By Theorem 4.1, we get from our assumption the existence of a polynomial-time computable natural property with even larger usefulness $s(n) = 0.9 \cdot n$. Thus, we can break any candidate PRG G. Hence, by [Hås+99], we get a randomized polynomial-time algorithm A for inverting a candidate one-way function family f that the PRG G was based on. This algorithm A succeeds (with high probability over its internal randomness) on almost all inputs $y = f(x)$ for uniformly random $x \in \{0,1\}^n$.

Note that such a randomized inverting algorithm A for f can also always check in polynomial time whether it has succeeded at getting an inverse z of f on a given input $y = f(x)$ (by checking if $f(z) = y$). Hence, we can derandomize this randomized inverter A in polynomial time, using a Nisan-Wigderson-style PRG (from $\{0,1\}^{O(\log n)}$ to $\{0,1\}^{\mathsf{poly}(n)}$ bits) that exists under the assumption that $\mathsf{E} \not\subset \mathsf{io\text{-}SIZE}[2^{o(n)}]$ [NW94; IW97]; see Theorem 2.2. The latter assumption holds in our case by Theorem 5.2. The theorem follows. □

We also get an analogue of Theorem A.1 from the Chain Rule for pK^t as in in Eq. (16) of Theorem 3.4, except we end up with a randomized inverting algorithm rather than a deterministic one.

B Natural property from SoI

Borrowing some ideas from [LW95] and [Hir21] (see also [Hir22]), we show how to achieve usefulness $s(n) = n - 2$, in time $2^{O(n/\log n)}$, from SoI.

We need the folowing lemma.

Lemma B.1 (Computational Depth Upper Bound [Hir21])**.** *For every $\varepsilon > 0$, every non-decreasing polynomials q_{dpt} and p_{dpt}, and every large enough $x \in \{0,1\}^n$, there exists a time bound t^* such that $q_{\mathsf{dpt}}(n) \leq t^* \leq 2^{n^\varepsilon}$, and*

$$\mathsf{K}^{t^*}(x) - \mathsf{K}^{p_{\mathsf{dpt}}(t^*)}(x) \leq O\left(n/\log n\right).$$

Theorem B.2. *Assume the Chain Rule for K^t as in Eq. (7) of Theorem 3.1, for some $\delta(n,t) \leq O(n^{1/2}/\log n)$. Let t be any sufficiently large polynomial in n. Then there is a $2^{O(n/\log n)}$-time computable natural property $\mathcal{A}$ for K^t on n-bit inputs with usefulness $s(n) = n - 2$.*

Proof. For any $x \in \{0,1\}^n$ and any time bound $\tau \in \mathbb{N}$, let $y_\tau \in \{0,1\}^{\mathsf{K}^\tau(x)}$ be the lexicographically first string of length $\mathsf{K}^\tau(x)$ such that the universal TM $U(y)$ outputs x within τ steps; that is, y_τ is the lexicographically first $\mathsf{K}^\tau(x)$-witness.

By the 2-string Chain Rule (Symmetry of Information) for strings x and y_t, we have for some polynomial p that, for any large τ,

$$\begin{aligned}
\mathsf{K}^{p(\tau)}(y_\tau \mid x) &\leq \mathsf{K}^{2\tau}(x, y_\tau) - \mathsf{K}^{p(\tau)}(x) + O(\log \tau) + O(n/\log n) \\
&\leq \mathsf{K}^{\tau}(y_\tau) - \mathsf{K}^{p(\tau)}(x) + O(\log \tau) + O(n/\log n) && (y_\tau \text{ determines } x \text{ in } \tau \text{ steps}) \\
&\leq |y_\tau| - \mathsf{K}^{p(\tau)}(x) + O(\log \tau) + O(n/\log n) && (\text{since } \mathsf{K}^t(y_\tau) \leq |y_\tau| + O(1)) \\
&= \mathsf{K}^{\tau}(x) - \mathsf{K}^{p(\tau)}(x) + O(\log \tau) + O(n/\log n). && (\text{since } |y_\tau| = \mathsf{K}^\tau(x))
\end{aligned}$$

By Lemma B.1, for every $\varepsilon > 0$, there exists a time bound $t \leq t^* \leq 2^{n^\varepsilon}$ such that

$$\mathsf{K}^{t^*}(x) - \mathsf{K}^{p(t^*)}(x) \leq O(n/\log n).$$

Hence, by the above, we get for $\tau = t^*$ that

$$\mathsf{K}^{p(t^*)}(y_{t^*} \mid x) \leq O(n/\log n).$$

It follows that, by exhaustive search in time $2^{O(n/\log n)}$, we can find a list of strings that contains the $\mathsf{K}^{t^*}(x)$-witness y_{t^*}. Let y be the smallest-length string on the list such that $U(y)$ prints out x within t' time steps, for $t \leq t' \leq 2^{n^\varepsilon}$. By the choice of y, the definition of y_{t^*}, and since $t^* \geq t$, we have that

$$|y| \leq |y_{t^*}| = \mathsf{K}^{t^*}(x) \leq \mathsf{K}^t(x). \tag{31}$$

On the other hand, since y is a time-bounded Kolmogorov description of x, we also have

$$K(x) \leq |y|. \tag{32}$$

Consider the following algorithm $\mathcal{A}$:

> On input $x \in \{0,1\}^n$ and 1^t, try all strings $w \in \{0,1\}^{O(n/\log n)}$, running $U(w)$ for at most 2^{n^ε} steps on each. Collect all those outputs y of $U(w)$ such that $U(y)$ prints x within 2^{n^ε} steps. Let d be the length of the shortest such y. If $d \leq s(n)$, then accept; otherwise, reject.

Observe that $\mathcal{A}$ runs in time $2^{O(n/\log n)}$. For correctness, if $\mathsf{K}^t(x) \leq s(n)$, then by (31), $\mathcal{A}(x, 1^t)$ will accept. On the other hand, by a simple counting argument, with probability at least $1 - 2^{-c+1}$ over uniformly random $x \in \{0,1\}^n$, we have that $\mathsf{K}(x) > n - c$. Hence, by (32), $\mathcal{A}(x, 1^t)$ will reject at least $1/2$ random strings $x \in \{0,1\}^n$, for $s(n) = n - 2$. □

ECCC ISSN 1433-8092
https://eccc.weizmann.ac.il