One-Way Functions and a Conditional Variant of MKTP

Eric Allender
Department of Computer Science, Rutgers University, Piscataway, NJ, USA

Mahdi Cheraghchi
Department of EECS, University of Michigan, Ann Arbor, MI, USA

Dimitrios Myrisiotis
Department of Computing, Imperial College London, London, UK

Harsha Tirumala
Department of Computer Science, Rutgers University, Piscataway, NJ, USA

Ilya Volkovich
Computer Science Department, Boston College, Chestnut Hill, MA, USA

Abstract

One-way functions (OWFs) are central objects of study in cryptography and computational complexity theory. In a seminal work, Liu and Pass (FOCS 2020) proved that the average-case hardness of computing time-bounded Kolmogorov complexity is *equivalent* to the existence of OWFs. It remained an open problem to establish such an equivalence for the average-case hardness of some natural NP-complete problem. In this paper, we make progress on this question by studying a conditional variant of the Minimum KT-complexity Problem (MKTP), which we call McKTP, as follows.

1. First, we prove that if McKTP is average-case hard on a polynomial fraction of its instances, then there exist OWFs.
2. Then, we observe that McKTP is NP-complete under polynomial-time randomized reductions.
3. Finally, we prove that the existence of OWFs implies the nontrivial average-case hardness of McKTP.

Thus the existence of OWFs is inextricably linked to the average-case hardness of this NP-complete problem. In fact, building on recently-announced results of Ren and Santhanam [28], we show that McKTP is hard-on-average *if and only if* there are logspace-computable OWFs.

2012 ACM Subject Classification Theory of computation → Circuit complexity; Theory of computation → Problems, reductions and completeness; Theory of computation → Cryptographic primitives

Keywords and phrases Kolmogorov complexity, KT Complexity, Minimum KT-complexity Problem, MKTP, Conditional KT Complexity, Minimum Conditional KT-complexity Problem, McKTP, one-way functions, OWFs, average-case hardness, pseudorandom generators, PRGs, pseudorandom functions, PRFs, distinguishers, learning algorithms, NP-completeness, reductions

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2021.7

Related Version *Full Version*: `https://eccc.weizmann.ac.il/report/2021/009/`

Funding *Eric Allender*: Partially supported by NSF Grants CCF-1909216 & CCF-1909683.
Mahdi Cheraghchi: Mahdi Cheraghchi's research was partially supported by the National Science Foundation under Grant No. CCF-2006455.
Dimitrios Myrisiotis: This work was partly carried out during a visit of Dimitrios Myrisiotis to DIMACS, with support from the Special Focus on Lower Bounds in Computational Complexity funded under NSF Grant CCF-1836666.
Harsha Tirumala: Harsha Tirumala was partially supported by NSF Grant CCF-1909216 and by the Simons Collaboration on Algorithms and Geometry.

Acknowledgements We would like to thank Russell Impagliazzo for explaining his work [18] to us, and Ján Pich and Ninad Rajgopal for illuminating discussions. We thank Ján Pich for bringing his work [27] to our attention. We thank Mikito Nanashima and Hanlin Ren for helpful comments and for spotting bugs in the proofs of earlier versions of Lemma 20 and Lemma 21, respectively. In particular, we thank Hanlin Ren for asking the question of whether KT complexity would be an appropriate complexity measure to consider in the context of our work. We thank Yanyi Liu and Rafael Pass for the excellent correspondence regarding their work [20, 23, 24], and Rahul Santhanam for bringing the work by Impagliazzo and Naor [19] to our attention. Finally, we would like to thank the anonymous reviewers for their helpful feedback.

1 Introduction

One-way functions (OWFs) – that is, functions that are easy to compute but hard to invert – are objects of great importance in cryptography and computational complexity. For example, it is known that OWFs exist if and only if pseudorandom generators exist [12] and, moreover, if OWFs exist, then $\mathsf{P} \neq \mathsf{NP}$.

In this paper, we ask the following question:

> *Can the existence of OWFs be shown to be equivalent to the average-case hardness of some* NP*-complete problem?*

We take concrete steps toward giving an affirmative answer to this question, by presenting a candidate problem. Note that by Impagliazzo and Naor [19] it is known that there exists some NP-complete problem (Subset Sum) whose average-case hardness implies the existence of OWFs. However, what we attempt to do is different: We want to make concrete progress in *characterizing* OWFs by the average-case hardness of an NP-complete problem.

The importance of NP stems mainly from the fact that, for thousands of important naturally-occurring computational problems, their worst-case computational complexity is best explained by knowing that they are NP-complete. However, NP-completeness has not been as relevant for the concerns of cryptographers, who require one-way functions, which in turn require problems in NP that are hard-on-average. Liu and Pass [20] gave what is arguably the first "natural" example of a problem in NP that is hard-on-average if and only if one-way functions exist; but this problem (computing time-bounded Kolmogorov complexity, K^t) is not known to be NP-complete. Although it is not hard to modify their language to obtain an artificial NP-complete problem with the same average-case complexity (see Proposition 24), there had been no "natural" example of an NP-complete problem whose average-case complexity had been connected directly to the existence of one-way functions. Our main contribution is to present such an example.

There are different ways to define time-bounded Kolmogorov complexity; the measure KT (defined in [4]) has the property that $\mathrm{KT}(x)$ is approximately the same as the circuit complexity of the function that has x as its truth table. Thus the problem $\mathrm{MKTP} = \{(x, i) \mid \mathrm{KT}(x) \leq i\}$ has been useful [4] in studying the Minimum Circuit Size Problem $\mathrm{MCSP} = \{(f, i) \mid \mathrm{CC}(f) \leq i\}$, which has been the subject of much recent work. As with most other Kolmogorov complexity measures, $\mathrm{KT}(x)$ is defined in [4] as a special case of the conditional KT-complexity $\mathrm{KT}(x \mid y)$, where y is the empty string. Our results concern the decision problem $\mathrm{McKTP} = \{(x, y, i) \mid \mathrm{KT}(x \mid y) \leq i\}$. We show the following.

- **(a)** If McKTP is hard-on-average, then one-way functions exist (Theorem 1).
- **(b)** McKTP is NP-complete under randomized reductions (Theorem 2).
- **(c)** If one-way functions exist, then McKTP is (somewhat) hard-on-average (Theorem 4).
- **(d)** In fact, McKTP is hard-on-average if and only if logspace-computable one-way functions exist (Theorem 3 and Theorem 5).

There has been a flurry of recent activity on this topic, and it may be helpful to present the following timeline:

1. [20] is posted by Liu and Pass, proving an equivalence between the existence of OWFs and the average-case hardness of K^t complexity.
2. [6] is posted by Allender, Cheraghchi, Myrisiotis, Tirumala, and Volkovich, claiming to characterize the existence of OWFs by the average-case complexity of an NP-complete problem called Sparse Partial MCSP. This paper was retracted.
3. [5] is posted by Allender, Cheraghchi, Myrisiotis, Tirumala, and Volkovich, presenting the proofs of Item a through Item c above.
4. [21] is posted by Liu and Pass, whereby they prove that *subexponentially-hard* OWFs exist if and only if $\mathrm{M}K^t\mathrm{P}$ (a decision problem based on K^t complexity) is average-case hard for *sublinear-time non-uniform* heuristics.
5. [24] is posted by Liu and Pass, showing that one-way functions exist if and only if the EXP-complete language MKtP is hard-on-average[1] and that logspace-computable one-way functions exist if and only if the PSPACE-complete language MKSP is hard-on-average.
6. [28] is posted by Ren and Santhanam, showing that MKTP is hard-on-average if and only if logspace-computable one-way functions exist. This allows us to prove Item d above.
7. [23] is posted by Liu and Pass (which is inspired by and in part a response to [6]), showing that a conditional variant of K^t complexity is NP-complete, and is hard-on-average if and only if one-way functions exist.
8. [16] is posted by Ilango, Ren, and Santhanam, showing that one-way functions exist if and only if the undecidable problem MKP (i.e., a decision problem based on Kolmogorov complexity) is hard-on-average under a samplable distribution, and if and only if MCSP is hard-on-average under a locally-sampleable distribution.
9. [22] is posted by Liu and Pass, generalizing the results of Ilango, Ren, and Santhanam [16], whereby they show that there exists some sparse language L such that OWFs exist if and only if L is average-case hard with respect to some efficiently sampleable "high-entropy" distribution.

1.1 Prior work

An early goal in cryptographic research was to base the existence of cryptographically secure one-way functions on the worst-case complexity of some NP-complete problem. This goal remains elusive; it was shown in [2] that no black-box argument of this sort can proceed based on non-adaptive reductions. Non-adaptive worst-case-to-average-case reductions were also studied by Bogdanov and Trevisan [8], who showed that such reductions to sets in NP exist only for problems in $\mathsf{NP/poly} \cap \mathsf{coNP/poly}$. Recent work by Nanashima [26] holds open the possibility that the security of OWFs can be based on an *adaptive* black-box reduction, by first establishing a non-adaptive black-box reduction basing the existence of *auxiliary input one-way functions* on the worst-case complexity of an NP-complete problem, although this would also require non-relativizing techniques. Instead of worst-case hardness, the focus of our work is on average-case hardness assumptions. A nice survey on this area, that lays out many of the issues about one-way functions and average-case complexity, is the one by Bogdanov and Trevisan [7].

[1] This is also proved in [28], and was posted to ECCC one day later.

Hirahara and Santhanam have discussed zero-error average-case complexity of problems related to MKTP [14]. Santhanam [29] showed that a restricted type of hitting-set generator exists if and only if MCSP is zero-error average-case hard. Hirahara also proved similar results connecting the worst-case and the zero-error average-case complexity of problems related to MCSP and Kolmogorov complexity [13].

More recently, Brzuska and Couteau [9] discuss basing OWFs on average-case hardness, stating that it remains an open question to do this for the general notion of average-case hardness. They present some negative results, indicating the difficulty of establishing the existence of fine-grained one-way functions, based on the existence of average-case hardness, via black-box reductions.

There is also an important line of work (including Ajtai [1] and Micciancio and Regev [25]) basing the existence of OWFs on the *worst-case* complexity of certain problems in NP (including problems that are closely related to NP-complete problems, although they are not themselves known to be NP-complete).

1.2 Our results

In this work, we connect the existence of OWFs to the average-case hardness of computing a conditional (and NP-complete) variant of MKTP, which we term McKTP.

Initially, we prove that the average-case hardness of McKTP implies the existence of OWFs.

▶ **Theorem 1** (Informal). *OWFs exist if* McKTP *is hard-on-average on a polynomial fraction of its instances.*

We also show that McKTP is NP-complete under randomized reductions.

▶ **Theorem 2** (Informal). McKTP *is* NP*-complete under polynomial-time one-sided-error randomized reductions.*

Moreover, Theorem 1 suggests an approach for excluding Impagliazzo's *Pessiland* [17], that is, a version of our world where there are average-case hard problems in NP *and* there are no OWFs. This approach is based on the following observation. If McKTP is NP-hard under average-case reductions, then by Theorem 1 the existence of an average-case hard problem in NP would imply the existence of OWFs. Therefore proving that McKTP is NP-hard under average-case reductions excludes Pessiland.

We are able to prove a stronger version of Theorem 1, building on the work of Ren and Santhanam [28].

▶ **Theorem 3** (Informal). *Logspace-computable OWFs exist if* McKTP *is hard-on-average on a polynomial fraction of its instances.*

Finally, we prove a *weak* converse of Theorem 1, and a *strong* converse of Theorem 3.

▶ **Theorem 4** (Informal). *OWFs exist only if* McKTP *is hard-on-average on an* exponential *fraction of its instances.*

▶ **Theorem 5** (Informal). *Logspace-computable OWFs exist only if* McKTP *is hard-on-average on an* polynomial *fraction of its instances.*

By Theorem 3 and Theorem 5, we get the following corollary.

▶ **Corollary 6.** McKTP *is hard-on-average if and only if logspace-computable OWFs exist.*

1.2.1 How significant are our results?

The reader may wonder whether the hypothesis of Theorem 1 is overly strong. Is there perhaps some trivial heuristic that succeeds well on average for this NP-complete decision problem?

The input to the problem consists of a triple (x, y, θ), where the question is whether $\mathrm{KT}(x \mid y) \leq \theta$, where θ is a number bounded by $|x| + O(\log |x|)$. A simple heuristic is to accept if θ is at the high end of this range, and reject otherwise; one can augment this to accept for slightly lower values of θ if x has certain hallmarks of low complexity (such as starting or ending with a logarithmic number of zeros, or agreeing with y on those substrings). However, when inputs are chosen at random, this heuristic still seems likely to fail with constant probability if θ is close to the boundary between where the heuristic accepts and rejects. In particular, it is far from clear how to design a heuristic that would have failure probability less than, say $1/s^2$, where θ ranges over a domain of size s. In particular, it seems quite plausible that there is a constant k for which no heuristic can achieve failure probability less than $1/s^k$, which is precisely the hypothesis of Theorem 1, and is sufficient for the existence of OWFs.

Moreover, by Theorem 5, this hypothesis is in fact *equivalent* to the existence of logspace-computable OWFs, which is widely believed to hold.

By the same token, the conclusion of Theorem 4 gives a much weaker, but still non-trivial, average-case hardness condition for McKTP.

1.3 Our techniques

Our main results are Theorem 1, Theorem 2, and Theorem 4. Below we provide some intuition regarding their proofs.

1. Theorem 1 is proved by
   a. giving an average-case decision-to-search reduction for McKTP (see Lemma 20) and
   b. observing that a recent result by Liu and Pass [20], whereby they prove that the average-case hardness of a search variant of time-bounded Kolmogorov complexity K^t yields OWFs, can be adjusted to the case of McKTP as well (see Lemma 21).
   The three properties of time-bounded Kolmogorov complexity K^t, for some $t : \mathbb{N} \to \mathbb{N}$ where $t(n) \geq n$ for all $n \in \mathbb{N}$, that are used by Liu and Pass, are as follows.
      i. One can create a string of low time-bounded Kolmogorov complexity in polynomial time. This can be done by running a universal Turing machine U on some string, for polynomially-many steps, and subsequently recording the output of U.
      ii. For any string x, the possible values of its K^t complexity are polynomially-many in $|x|$. In fact, there is a $c > 0$ such that, for any function $t : \mathbb{N} \to \mathbb{N}$ such that $t(n) \geq n$ for all $n \in \mathbb{N}$, and any string x, the possible values of $K^t(x)$ are at most $|x| + c$.
      iii. The following domination property holds. Let $x^* \in \{0,1\}^n$ be a string, and $c > 0$ be as in Item 1(b)ii. Then,
   $$\Pr_{\Pi \sim \{0,1\}^{n+c}}\left[U\left(\Pi, 1^{t(n)}\right) = x^*\right] \geq \frac{1}{2^{n+c}} = \frac{2^{-n}}{2^c} \geq \frac{\mathbf{Pr}_{x \sim \{0,1\}^n}[x = x^*]}{\mathrm{poly}(n)}.$$
   As it turns out, all of these properties are satisfied even when one considers McKTP.
2. Theorem 3 is proved by use of the techniques of [28]. In particular, the proof of Theorem 1 shows that the following function is one-way, if McKTP is hard-on-average:

Given (s, t, y, Π), output the string obtained by running U on y and the length-s prefix of Π for t steps.

Ren and Santhanam observe that this function is logspace-computable if we restrict t to be $O(\log n)$. Then, crucially, they show that for most strings in the range of this function, $s + t$ is minimized when $t = O(\log n)$. These insights, combined with the the proof of the preceding theorem, suffice.

3. Theorem 2 is proved by
   a. noting that McKTP is in NP (see Lemma 11) and
   b. showing the NP-hardness of McKTP (see Corollary 34). This is done by giving a polynomial-time randomized reduction from Set Cover, which is NP-hard to approximate (see Corollary 33), to an appropriate gap version of McKTP (see Corollary 32). Note that this step closely mimics the proof of Ilango [15] for the NP-hardness of Minimum Oracle Circuit Size Problem (MOCSP).
4. Theorem 4 is proved by giving a proof of its contrapositive statement, as explained by the items below.
   a. Assume that McKTP is easy on average under the uniform distribution.
   b. By a corollary of Ilango, Loff, and Oliveira, for all $a \geq 1$, there exists a learning algorithm for $\mathsf{SIZE}[n^a]$ that works for infinitely many $n \in \mathbb{N}$.
   c. By a learner-to-distinguisher reduction, for every polynomial-time computable Boolean function family $\{f_y\}_{y \in \{0,1\}^*}$, there is a distinguisher for $\{f_y\}_{y \in \{0,1\}^*}$.
   d. By the correctness of the works by Håstad, Implagliazzo, Levin, and Luby [12], and Goldreich, Goldwasser, and Micali [11], there are no OWFs.
5. Theorem 5 is proved by giving a slight modification to the proof of [28, Lemma 4.7].

1.4 Paper organization

In Section 2 we give some background knowledge and useful facts. We prove Theorem 1 in Section 3, Theorem 3 in Section 4, and Theorem 5 in Section 5. Finally, we prove Theorem 2 in Appendix B. Theorem 4 is proved in the full version of the paper [5].

2 Preliminaries

2.1 Notation

We denote the natural numbers by $\mathbb{N}$ and the positive reals by $\mathbb{R}_{>0}$. For any $n \in \mathbb{N}$, we denote the set $\{1, \ldots, n\}$ by $[n]$. Let $x = (x_1, \ldots, x_n) \in \{0,1\}^n$ be a string of length n; we write $|x| := n$. The empty string is denoted by λ.

We denote by $\mathcal{F}_n$ the class of all Boolean functions on n variables. We identify infinite Boolean functions $f : \{0,1\}^* \to \{0,1\}$ with collections $\{f_n\}_{n \in \mathbb{N}}$, whereby $f_n : \{0,1\}^n \to \{0,1\}$ for all $n \in \mathbb{N}$.

We consider Boolean circuits over the bounded fan-in $\{\wedge_2, \vee_2, \neg\}$ basis. Given a circuit, its *size* is the number of its gates. Let $s : \mathbb{N} \to \mathbb{N}$ be a function. If we use s to upper bound the size of some circuit, then we shall call s a *size function*.

Given a Boolean function $f : \{0,1\}^n \to \{0,1\}$, the *circuit complexity of* f, denoted $\mathrm{CC}(f)$, is the size of a minimum size circuit that computes f. For a size function $s : \mathbb{N} \to \mathbb{N}$, we denote by $\mathsf{SIZE}[s(n)]$ the class of Boolean functions $f = \{f_n\}_{n \in \mathbb{N}}$, whereby $f_n : \{0,1\}^n \to \{0,1\}$ for all $n \in \mathbb{N}$, such that $\mathrm{CC}(f_n) \leq s(n)$ for all $n \in \mathbb{N}$.

In this work, we do not distinguish between Turing machines and algorithms. We say that an algorithm A is a *PPT algorithm* if A is a probabilistic polynomial-time algorithm. If A is a PPT algorithm that runs in time $p(n)$ for a polynomial p, then we denote by $A(x; r)$

the output of A on input $x \in \{0,1\}^*$ using random bits $r \in \{0,1\}^{p(|x|)}$. We say that an algorithm A is a *PPT oracle algorithm* if A is a PPT algorithm that has access to some oracle. If A is a PPT oracle algorithm that runs in time $p(n)$ for a polynomial p and has access to an oracle for a language $L \subseteq \{0,1\}^*$, then we denote by $A^L(x;r)$ the output of A^L on input $x \in \{0,1\}^*$ using random bits $r \in \{0,1\}^{p(|x|)}$.

2.2 Probability theory

We will use the following useful fact from probability theory.

▶ **Lemma 7** (Markov's inequality). *If X is a non-negative random variable with $\mu := \mathbf{E}[X]$, then for all $k > 0$ it is the case that $\mathbf{Pr}[X \geq k\mu] \leq 1/k$.*

2.3 KT complexity

2.3.1 A universal Turing machine

In what follows, we fix some *efficient* universal (oracle) Turing machine (UTM) U. Let $y, \Pi, z \in \{0,1\}^*$ and $t \in \mathbb{N}$. The notation $U^{\Pi,y}(z, 1^t)$ denotes the output of U when U runs the program Π on input z for at most t steps, given that U has extended oracle access to program Π and standard oracle access to auxiliary string y. These notions are defined as follows.

1. *Standard oracle access to auxiliary string* y means that U has a standard oracle tape T_y of $\log|y|$ cells, and that in order to read a bit y_i of y, whereby $1 \leq i \leq |y|$, the machine U has to write $i \in \{0,1\}^{\log|y|}$ on T_y and then enter a question state. In the next step, the contents of T_y are erased and replaced by a bit b such that $b = y_i$.

   One important aspect of our choice of U is that, for every auxiliary string $y \in \{0,1\}^*$ and $1 \leq i \leq \log|y|$, the oracle query $y_i \stackrel{?}{=} 1$ is such that it *requires* time $\log|y|$, and can be implemented in time $O(\log|y|)$.
2. *Extended oracle access to program* Π means that U has a tape T_Π of $|\Pi|$ cells that contains Π, and the head of T_Π has *both* the ability to jump to an indexed location $1 \leq i \leq |\Pi|$ of T_Π, namely $T_\Pi[i] = \Pi_i$, *and* to move left and right on T_Π. Note that in the former case the index i is written in a separate tape of $\log|\Pi|$ cells, specifically allocated for that purpose. (So extended oracle access implies the existence of two tapes that help facilitate the oracle query.)

The notation $U^{\Pi,y}(z)$ denotes the output of U when U runs the program Π on input z, until Π halts (if this is the case, otherwise Π runs forever), whereby U has extended oracle access to Π and standard oracle access to y.

In this work, we will assume that whenever U is given oracle access to a program Π, this access will be *extended*, and whenever U is given oracle access to an auxiliary string y, this access will be *standard*. This is mainly to avoid unnecessary complications in the proof of Theorem 2 (where it is convenient to have sequential access to Π, while requiring that each query to y uses logarithmic time) while maintaining the trivial upper bound on KT complexity (see Lemma 8) which requires oracle access to Π.

We will also assume that the output of U will either be 1 or 0, on any input.

2.3.2 Definition of KT complexity, and some properties

Given strings $x, y \in \{0,1\}^*$, we define the *KT complexity of x given y*, denoted $\mathrm{KT}(x \mid y)$, to be the minimum value of $|\Pi| + t$ over programs $\Pi \in \{0,1\}^*$ and run-time bounds $t \in \mathbb{N}$ whereby for all $1 \le i \le |x|$ it is the case that $U^{\Pi,y}(i, 1^t) = x_i$.[2] For all strings $x \in \{0,1\}^*$, we define $\mathrm{KT}(x)$ to be equal to $\mathrm{KT}(x \mid \lambda)$.

▶ **Lemma 8** ([4]). *There is a $c > 0$ such that for all $x \in \{0,1\}^*$ it is the case that $\mathrm{KT}(x)$ is at most $|x| + c \log |x|$.*

▶ **Corollary 9.** *There is a $c > 0$ such that for all $x, y \in \{0,1\}^*$ it is the case that $\mathrm{KT}(x \mid y)$ is at most $|x| + c \log |x|$.*

2.4 Minimum Conditional KT-complexity Problem, and variants

We give here formal definitions of the computational problems that we will consider in this work. These are the decision and search variants of McKTP.

▶ **Definition 10** (Decision variant). *Let $c > 0$ be as in Corollary 9. Let $n \in \mathbb{N}$ and $m : \mathbb{N} \to \mathbb{N}$. The* Minimum m-Conditional KT-complexity Problem of dimension n (McKT^{m}P of dimension n) *is defined as follows.*

- *Input: Strings $x \in \{0,1\}^n$, $y \in \{0,1\}^{m(n)}$, and a parameter $0 \le \theta \le n + c \log n$ in binary.*
- *Question: Is there a program $\Pi \in \{0,1\}^*$ and a run-time bound $t \in \mathbb{N}$ such that $U^{\Pi,y}(i, 1^t) = x_i$ for all $1 \le i \le n$, and $|\Pi| + t \le \theta$?*

The following result is a standard observation.

▶ **Lemma 11.** *For all polynomial-time computable functions $m : \mathbb{N} \to \mathbb{N}$, it is the case that* McKT^{m}P *of dimension n is in* NP.

▶ **Definition 12** (Search variant). *Let $n \in \mathbb{N}$ and $m : \mathbb{N} \to \mathbb{N}$. The* search variant of Minimum m-Conditional KT-complexity Problem of dimension n (Search McKT^{m}P of dimension n) *is defined as follows.*

- *Input: Strings $x \in \{0,1\}^n$ and $y \in \{0,1\}^{m(n)}$.*
- *Output: A program $\Pi \in \{0,1\}^*$ and a run-time bound $t \in \mathbb{N}$ in binary such that $U^{\Pi,y}(i, 1^t) = x_i$ for all $1 \le i \le n$, and the sum $|\Pi| + t$ is minimized over the choices of Π and t.*

2.5 One-way functions

In the following, a function μ is said to be *negligible* if for every polynomial p there exists a $n_0 \in \mathbb{N}$ such that for all naturals $n > n_0$ it is the case that $\mu(n) \le 1/p(n)$.

▶ **Definition 13.** *Let $f : \{0,1\}^* \to \{0,1\}^*$ be a polynomial-time computable function. We say that f is a* one-way function (OWF) *if for every PPT algorithm A there exists a negligible function μ such that for all $n \in \mathbb{N}$ it is the case that*

$$\mathop{\mathbf{Pr}}_{x \sim \{0,1\}^n, r}\left[A(1^n, f(x); r) \in f^{-1}(f(x))\right] < \mu(n)$$

where the size of r is equal to the running time of A.

[2] Originally [4], $\mathrm{KT}(x \mid y)$ was defined with the additional requirement that, for $i = |x|+1$, $U^{\Pi,y}\left(i, 1^t\right) = *$. We do not need that additional complication here, although our theorems would also hold using that definition.

We will also employ the following weaker notion of OWFs.

▶ **Definition 14.** *Let $f : \{0,1\}^* \to \{0,1\}^*$ be a polynomial-time computable function. We say that f is an* α-weak one-way function (α-weak OWF) *if for every PPT algorithm A and all sufficiently large $n \in \mathbb{N}$ it is the case that*

$$\mathbf{Pr}_{x\sim\{0,1\}^n,r}\left[A(1^n, f(x)\,;r) \in f^{-1}(f(x))\right] < 1 - \alpha(n)$$

where the size of r is equal to the running time of A. We say that f is a weak one-way function (weak OWF) *if there exists some polynomial $q > 0$ such that f is a $(1/q)$-weak OWF.*

Yao [30] proved that the existence of weak OWFs implies the existence of OWFs.

▶ **Theorem 15** ([30]). *Assume that there exists a weak one-way function. Then there exists a one-way function. (Also, if there exists a weak-one-way function computable in logspace, then there is a one-way function computable in logspace.)*

2.6 Average-case hardness/easiness

A *heuristic* H is a PPT algorithm that, on input any $x \in \{0,1\}^n$, outputs a value in $\{0,1\}$ along each computation path.

▶ **Definition 16** (Average-case hardness). *Let $\alpha : \mathbb{N} \to [0,1]$ be a failure parameter function. We say that a function $f : \{0,1\}^n \to \{0,1\}$ is* α-hard-on-average (α-HoA) *if for all heuristics H and all sufficiently large $n \in \mathbb{N}$ it is the case that*

$$\mathbf{Pr}_{x\sim\{0,1\}^n,r}[H(x;r) = f(x)] \leq 1 - \alpha(n)$$

where the size of r is equal to the running time of H.

▶ **Definition 17** (Average-case easiness). *Let $\alpha : \mathbb{N} \to [0,1]$ be a success parameter function. We say that a function $f : \{0,1\}^n \to \{0,1\}$ is* α-easy-on-average (α-EoA) *if f is not $(1-\alpha)$-hard-on-average; that is, if there exists some heuristic H such that for infinitely many $n \in \mathbb{N}$ it is the case that*

$$\mathbf{Pr}_{x\sim\{0,1\}^n,r}[H(x;r) = f(x)] > 1 - (1 - \alpha(n)) = \alpha(n)$$

where the size of r is equal to the running time of H.

Let $R \subseteq \{0,1\}^n \times \{0,1\}^*$ be a search problem. A *heuristic* H is a PPT algorithm that, on input any $x \in \{0,1\}^n$, outputs a value in $\{0,1\}^*$ along each computation path.

The notions of average-case hardness and easiness for search problems are defined in a fashion similar to that of decision problems; see Definition 16 and Definition 17.

3 OWFs from average-case hardness of McKTP

In this section, we prove the following result.

▶ **Theorem 18.** *Assume that, for some $m : \mathbb{N} \to \mathbb{N}$, $\mathrm{McKT}^m\mathrm{P}$ of dimension n is $(1/p)$-HoA for some polynomial p. Then, there exists some weak OWF.*

By Theorem 18 and Theorem 15, we get the following corollary.

▶ **Corollary 19** (Theorem 1, restated). *Assume that, for some $m : \mathbb{N} \to \mathbb{N}$, $\mathrm{McKT}^m\mathrm{P}$ of dimension n is $(1/p)$-HoA for some polynomial p. Then, there exists some OWF.*

3.1 Proof of Theorem 18

We will first require the following two lemmas.

▶ **Lemma 20.** *For all functions $m : \mathbb{N} \to \mathbb{N}$, if* $\mathrm{McKT}^m\mathrm{P}$ *is* $(1/p)$*-HoA for some polynomial* p*, then* Search $\mathrm{McKT}^m\mathrm{P}$ *is* $\left(1/p^2\right)$*-HoA.*

Proof. We will prove the contrapositive. That is, we will prove that if Search $\mathrm{McKT}^m\mathrm{P}$ is $\left(1 - 1/p^2\right)$-EoA, then $\mathrm{McKT}^m\mathrm{P}$ is $(1 - 1/p)$-EoA. In what follows, let $c > 0$ be as in Corollary 9.

Let $N' := n + m(n)$ be the size of the instances of Search $\mathrm{McKT}^m\mathrm{P}$ of dimension n. Assume that Search $\mathrm{McKT}^m\mathrm{P}$ is $\left(1 - 1/p^2\right)$-EoA. That is, assume that there exists some heuristic H' that on input a random instance $(x, y) \in \{0,1\}^n \times \{0,1\}^{m(n)}$ outputs with probability greater than $1 - 1/p(N')^2$ a program $\Pi \in \{0,1\}^*$ and a run-time bound $t \in \mathbb{N}$ (in binary) such that $U^{\Pi,y}(i, 1^t) = x_i$ for all $1 \le i \le n$, and the sum $|\Pi| + t$ is minimized over the choices of Π and t.

Given H', a heuristic H for $\mathrm{McKT}^m\mathrm{P}$ of dimension n and input size $N := n + m(n) + \log(n + c\log n)$, works as follows:

> On input strings $x \in \{0,1\}^n$ and $y \in \{0,1\}^{m(n)}$, and a size parameter $0 \le \theta \le n + c\log n$ in binary, run H' on (x, y) to get a program $\Pi \in \{0,1\}^*$ and a run-time bound $t \in \mathbb{N}$ (in binary). If Π and t are such that $U^{\Pi,y}(i, 1^t) = x_i$ for all $1 \le i \le n$ and $|\Pi| + t \le \theta$, then return YES. Else, return NO.

Note that the running time of H is polynomial in N. The success probability of H over a random instance (x, y, θ) and random bits r is

$$\begin{aligned}
&\Pr_{x,y,\theta,r}[H(x, y, \theta; r) \text{ succeeds}] \\
&\quad \ge \Pr_{x,y,\theta,r}[H(x, y, \theta; r) \text{ succeeds} \mid H'(x, y; r) \text{ succeeds}] \cdot \Pr_{x,y,r}[H'(x, y; r) \text{ succeeds}] \\
&\quad > 1 \cdot \left(1 - \frac{1}{p(N')^2}\right) = 1 - \frac{1}{p(N')^2} \ge 1 - \frac{1}{p(N)},
\end{aligned}$$

since $1/p(N')^2 \le 1/p(N)$ for all sufficiently large $n \in \mathbb{N}$, as desired.

Therefore, $\mathrm{McKT}^m\mathrm{P}$ is $(1 - 1/p)$-EoA as witnessed by H. ◀

The following is an elaboration on the seminal work by Liu and Pass [20].

▶ **Lemma 21** (Following Liu and Pass [20])**.** *Assume that, for some function $m : \mathbb{N} \to \mathbb{N}$,* Search $\mathrm{McKT}^m\mathrm{P}$ *is* $(1/p)$*-HoA for some polynomial* p*. Then, there exists some weak OWF.*

Proof. Fix some UTM U, and let $c > 0$ be as in Corollary 9. Let $n \in \mathbb{N}$ be sufficiently large and such that Search $\mathrm{McKT}^m\mathrm{P}$ of dimension n is $(1/p)$-HoA. Consider the function $f : \{0,1\}^* \to \{0,1\}^*$ defined by the mapping rule

$$(s, t, y, \Pi') \mapsto \left(s + t, U^{\Pi,y}\left(1, 1^t\right), \ldots, U^{\Pi,y}\left(n, 1^t\right), y\right),$$

where $m := m(n)$, $y \in \{0,1\}^m$, $\Pi' \in \{0,1\}^{n + c\log n}$ is a program, and $\Pi := \Pi'|_{[s]}$ is the s-bit prefix of Π'. Note that without loss of generality, $s + t \le n + c\log n$, by Corollary 9. This also implies that $s \le n + c\log n$ and $t \le n + c\log n$. For that matter, f is a function from $\{0,1\}^M$ to $\{0,1\}^N$, where $M := 2\log(n + c\log n) + m + n + c\log n$ and $N := \log(n + c\log n) + n + m$, and is computable in polynomial time.

Observe also that f is only defined over infinitely many input lengths. However, by a padding trick, f can be transformed into another function f' that is defined over all input lengths, and such that f' is a weak one-way function, given that f is [20].

We now claim that if Search McKT^{m}P is $(1/p)$-HoA, then f is a $(1/q)$-weak OWF, where q is a polynomial such that $q(n) := 2\left(n + c\log n\right)^2 n^c p(n + m(n))^3$ for all $n \in \mathbb{N}$. Towards a contradiction, assume that there exists a PPT algorithm A that inverts f with probability at least $1 - 1/q(M) \geq 1 - 1/q(n)$.

First, note that except for a fraction $1/\left(2p(n+m)\right)$ of sequences of random bits r for A, the deterministic machine A_r, given by $A_r(f(z)) := A(f(z)\,;r)$ for all $z \in \{0,1\}^M$, fails to invert f with probability at most $2p(n+m)\,/q(n)$ over a uniformly random input z. This is so, as

$$\begin{aligned}
&\Pr_r\left[\Pr_z[A_r(f(z)) \text{ fails}] > \frac{2p(n+m)}{q(n)}\right] \\
&\quad \leq \Pr_r\left[\Pr_z[A_r(f(z)) \text{ fails}] \geq 2p(n+m)\cdot \Pr_{z,r}[A_r(f(z)) \text{ fails}]\right] \\
&\quad = \Pr_r\left[\Pr_z[A(f(z)\,;r) \text{ fails}] \geq 2p(n+m)\cdot \mathop{\mathbf{E}}_r\left[\Pr_z[A(f(z)\,;r) \text{ fails}]\right]\right] \leq \frac{1}{2p(n+m)},
\end{aligned}$$

by Lemma 7. Henceforth, we will call such a sequence of random bits *good*; otherwise, we will call a sequence of random bits *bad*. Therefore, we have

$$\Pr_{z,r}[A(f(z)\,;r) \text{ fails} \mid r \text{ is good}] = \Pr_{z,r}[A_r(f(z)) \text{ fails} \mid r \text{ is good}] \leq \frac{2p(n+m)}{q(n)}.$$

We propose the following heuristic H for Search McKT^{m}P:

> On input strings $x \in \{0,1\}^n$ and $y \in \{0,1\}^m$, and using random bits r, the algorithm H runs $A(j,x,y;r)$ for all $j \in [n + c\log n]$. For each $j \in [n + c\log n]$, $A(j,x,y;r)$ returns a tuple $\left(s_j, t_j, y, \Pi'_j\right)$. Then, $H(x,y;r)$ returns a program $\Pi'_k|_{[s_k]}$ from $\left\{\Pi'_j|_{[s_j]}\right\}_{j\in[n+c\log n]}$ such that $U^{\Pi'_k|_{[s_k]},y}(i, 1^{t_k}) = x_i$ for all $1 \leq i \leq n$, and $\left|\Pi'_k|_{[s_k]}\right| + t_k = s_k + t_k$ is minimized.

We will now analyze the average-case performance of H. Fix a good sequence of random bits r, as defined above, and recall that, in this case, $\Pr_z[A_r(f(z)) \text{ fails}] \leq 2p(n+m)\,/q(n)$. Let S_r be the set of inputs (x,y) for which $H(x,y;r)$ fails, when given random bits r. Observe that, for any good r,

$$\Pr_{x,y}[H(x,y;r) \text{ fails}] = \frac{|S_r|}{2^{n+m}}.$$

Consider $(x,y) \in S_r$ and let $w_{x,y} := \mathrm{KT}(x \mid y)$ be the conditional KT-complexity of x given y. By Corollary 9, we have $w_{x,y} \leq n + c\log n$. If $H(x,y;r)$ fails, then it means that A fails to invert $(w_{x,y}, x, y)$ when given the good sequence of random bits r.

Recall that $\Pr_z[A_r(f(z)) \text{ fails}] \leq 2p(m\,(n+1))\,/q(n)$. Recall also, from the definition of f, and from the fact that $w_{x,y} \leq n + c\log n$, that

$$\Pr_z[f(z) = (w_{x,y}, x, y)] \geq \frac{1}{(n + c\log n)^2 \cdot 2^m \cdot 2^{n + c\log n}}.$$

Thus, for any good sequence r, we have

$$\frac{2p(n+m)}{q(n)} \geq \Pr_z[A_r\left(f(z)\right) \text{ fails}]$$

$$\begin{aligned}
&= \sum_{(w,x,y):A_r(w,x,y) \text{ fails}} \Pr_z[f(z) = (w,x,y)] \\
&\geq \sum_{(x,y):A_r(w_{x,y},x,y) \text{ fails}} \Pr_z[f(z) = (w_{x,y},x,y)] \\
&\geq \sum_{(x,y)\in S_r} \frac{1}{(n + c\log n)^2 \cdot 2^m \cdot 2^{n+c\log n}} \\
&= \frac{|S_r|}{2^{n+m}} \cdot \frac{1}{(n + c\log n)^2\, 2^{c\log n}} = \frac{\mathbf{Pr}_{x,y}[H(x,y;r) \text{ fails}]}{(n + c\log n)^2\, n^c}.
\end{aligned}$$

Since this holds for any good sequence r, we have that

$$\begin{aligned}
\Pr_{x,y,r}[H(x,y;r) \text{ fails} \mid r \text{ is good}] &\leq \frac{(n + c\log n)^2\, n^c 2p(n+m)}{q(n)} \\
&= \frac{(n + c\log n)^2\, n^c 2p(n+m)}{2\,(n + c\log n)^2\, n^c p(n+m)^3} \\
&= \frac{1}{p(n+m)^2} < \frac{1}{2p(n+m)},
\end{aligned}$$

since $p(n+m) > 2$ for all sufficiently large $n \in \mathbb{N}$. Therefore, H fails with probability at most

$$\Pr_{x,y,r}[H(x,y;r) \text{ fails} \mid r \text{ is good}] + \Pr_r[r \text{ is bad}] < \frac{1}{2p(n+m)} + \frac{1}{2p(n+m)} = \frac{1}{p(n+m)}.$$

This yields a contradiction. ◀

We now turn to the proof of Theorem 18.

Proof of Theorem 18. Immediate; by Lemma 20 and Lemma 21, since if p is a polynomial, then p^2 is a polynomial too. ◀

4 Logspace-computable OWFs from average-case hardness of McKTP

Now we show that, applying the insights of Ren and Santhanam [28], we can strengthen the theorems of the preceding section. We show the following.

▶ **Theorem 22.** *Assume that, for some $m : \mathbb{N} \to \mathbb{N}$, $\mathrm{McKT}^m\mathrm{P}$ of dimension n is $(1/p)$-HoA for some polynomial p. Then, there exists some weak OWF computable in logspace.*

Proof sketch. Modify the definition of f from the proof of Lemma 21, so that now f is

$$(s, t, y, \Pi') \mapsto \left(s + t, U^{\Pi,y}(1, 1^t), \ldots, U^{\Pi,y}(n, 1^t), y\right),$$

where $m := m(n)$, $y \in \{0,1\}^m$, $\Pi' \in \{0,1\}^{n+c\log n}$ is a program, $\Pi := \Pi'|_{[s]}$ is the s-bit prefix of Π', and $t \leq d\log n$ for some d. This function f is clearly computable in logspace.

Significantly, Ren and Santhanam [28, Theorem 4.1] show that, if the search version of KT is hard-on-average, then a function very similar to f is a weak one-way function. Essentially identical considerations allow us to conclude that, if Search $\mathrm{McKT}^m\mathrm{P}$ is $(1/p)$-HoA for some polynomial p, then f is a weak one-way function. The main point is that, for every y, most strings x have the property that, when $|\Pi| + t$ is minimized (where U uses description Π and run-time t to compute the bits of x), $t = O(\log n)$. The rest of the analysis is very similar to that of Lemma 21. ◀

By Theorem 22 and Theorem 15, we get the following corollary.

▶ **Corollary 23** (Theorem 3, restated). *Assume that, for some* $m : \mathbb{N} \to \mathbb{N}$, $\mathrm{McKT}^m\mathrm{P}$ *of dimension* n *is* $(1/p)$*-HoA for some polynomial* p*. Then, there exists some logspace-computable OWF.*

5 Average-case hardness of McKTP from logspace-computable OWFs: Proof of Theorem 5

Again, we appeal to the techniques of Ren and Santhanam. Ren and Santhanam [28, Theorem 4.4] show that, if there is a one-way function computable in logspace, then the problem of computing an approximation to KT complexity is hard-on-average. A nearly-identical proof shows that computing $\mathrm{KT}(x \mid y)$ is HoA. Essentially the only modification that needs to be made to the proof of [28, Theorem 4.4] arises in the proof of their Lemma 4.7, which establishes that computing KT is HoA under a condition that holds if there is a logspace-computable OWF. The proof of [28, Lemma 4.7] relies on the fact that the output of a certain pseudorandom generator has small KT complexity, whereas a random string has high KT complexity. But the output z of this generator also has small $\mathrm{KT}(z \mid y)$ for every y, whereas a random string z has $\mathrm{KT}(z \mid y)$ large for almost every y. Thus a very similar analysis shows that computing $\mathrm{KT}(x \mid y)$ is HoA, which in turn (via Lemma 20) implies that $\mathrm{McKT}^m\mathrm{P}$ is HoA.

References

1. Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing (STOC)*, pages 99–108. ACM, 1996. doi:10.1145/237814.237838.
2. Adi Akavia, Oded Goldreich, Shafi Goldwasser, and Dana Moshkovitz. On basing one-way functions on NP-hardness. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing (STOC)*, pages 701–710. ACM, 2006. See also [3].
3. Adi Akavia, Oded Goldreich, Shafi Goldwasser, and Dana Moshkovitz. Erratum for: On basing one-way functions on NP-hardness. In *Proceedings of the 42nd ACM Symposium on Theory of Computing (STOC)*, pages 795–796. ACM, 2010.
4. Eric Allender, Harry Buhrman, Michal Koucký, Dieter van Melkebeek, and Detlef Ronneburger. Power from random strings. *SIAM J. Comput.*, 35(6):1467–1493, 2006.
5. Eric Allender, Mahdi Cheraghchi, Dimitrios Myrisiotis, Harsha Tirumala, and Ilya Volkovich. One-way functions and a conditional variant of MKTP. *Electron. Colloquium Comput. Complex.*, 28:9, 2021.
6. Eric Allender, Mahdi Cheraghchi, Dimitrios Myrisiotis, Harsha Tirumala, and Ilya Volkovich. One-way functions and Partial MCSP. *Electron. Colloquium Comput. Complex.*, 28:9, 2021.
7. Andrej Bogdanov and Luca Trevisan. Average-case complexity. *Found. Trends Theor. Comput. Sci.*, 2(1), 2006.
8. Andrej Bogdanov and Luca Trevisan. On worst-case to average-case reductions for NP problems. *SIAM J. Comput.*, 36(4):1119–1159, 2006.
9. Chris Brzuska and Geoffroy Couteau. Towards fine-grained one-way functions from strong average-case hardness. *IACR Cryptol. ePrint Arch.*, 2020:1326, 2020.
10. Irit Dinur and David Steurer. Analytical approach to parallel repetition. In David B. Shmoys, editor, *Symposium on Theory of Computing (STOC)*, pages 624–633. ACM, 2014.
11. Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986.
12. Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

13 Shuichi Hirahara. Non-black-box worst-case to average-case reductions within NP. In *59th IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 247–258. IEEE Computer Society, 2018.

14 Shuichi Hirahara and Rahul Santhanam. On the average-case complexity of MCSP and its variants. In Ryan O'Donnell, editor, *32nd Computational Complexity Conference, CCC 2017, July 6-9, 2017, Riga, Latvia*, volume 79 of *LIPIcs*, pages 7:1–7:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

15 Rahul Ilango. Approaching MCSP from above and below: Hardness for a conditional variant and $\mathsf{AC}^0[p]$. In Thomas Vidick, editor, *11th Innovations in Theoretical Computer Science Conference, ITCS 2020, January 12-14, 2020, Seattle, Washington, USA*, volume 151 of *LIPIcs*, pages 34:1–34:26. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

16 Rahul Ilango, Hanlin Ren, and Rahul Santhanam. Hardness on any samplable distribution suffices: New characterizations of one-way functions by meta-complexity. *Electron. Colloquium Comput. Complex.*, 28:82, 2021.

17 Russell Impagliazzo. A personal view of average-case complexity. In *Proceedings of the Tenth Annual Structure in Complexity Theory Conference, Minneapolis, Minnesota, USA, June 19-22, 1995*, pages 134–147. IEEE Computer Society, 1995.

18 Russell Impagliazzo and Leonid A. Levin. No better ways to generate hard NP instances than picking uniformly at random. In *31st Annual Symposium on Foundations of Computer Science, St. Louis, Missouri, USA, October 22-24, 1990, Volume II*, pages 812–821. IEEE Computer Society, 1990.

19 Russell Impagliazzo and Moni Naor. Efficient cryptographic schemes provably as secure as subset sum. *J. Cryptol.*, 9(4):199–216, 1996.

20 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*, pages 1243–1254. IEEE, 2020.

21 Yanyi Liu and Rafael Pass. Cryptography from sublinear-time average-case hardness of time-bounded Kolmogorov complexity. In *Proceedings of the 53rd ACM Symposium on Theory of Computing (STOC)*. ACM, 2021.

22 Yanyi Liu and Rafael Pass. A note on one-way functions and sparse languages. *IACR Cryptol. ePrint Arch.*, 2021:890, 2021.

23 Yanyi Liu and Rafael Pass. On one-way functions from NP-complete problems. *Electron. Colloquium Comput. Complex.*, 28:59, 2021.

24 Yanyi Liu and Rafael Pass. On the possibility of basing cryptography on $\mathsf{EXP} \neq \mathsf{BPP}$. *Electron. Colloquium Comput. Complex.*, 28:56, 2021.

25 Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on Gaussian measures. *SIAM J. Comput.*, 37(1):267–302, 2007. doi:10.1137/S0097539705447360.

26 Mikito Nanashima. On basing auxiliary-input cryptography on NP-hardness via nonadaptive black-box reductions. In *12th Innovations in Theoretical Computer Science Conference (ITCS)*, volume 185 of *LIPIcs*, pages 29:1–29:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

27 Ján Pich. Learning algorithms from circuit lower bounds. *CoRR*, abs/2012.14095, 2020. arXiv:2012.14095.

28 Hanlin Ren and Rahul Santhanam. Hardness of KT characterizes parallel cryptography. *Electron. Colloquium Comput. Complex.*, 28:57, 2021.

29 Rahul Santhanam. Pseudorandomness and the Minimum Circuit Size Problem. In *11th Innovations in Theoretical Computer Science Conference (ITCS)*, volume 151 of *LIPIcs*, pages 68:1–68:26. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

30 Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 80–91. IEEE Computer Society, 1982.

A Hard-on-average problems in NP

We first introduce some useful notation. For a language $L \subseteq \{0,1\}^*$ we define its *characteristic function*, namely $f_L : \{0,1\}^* \to \{0,1\}$, to be a function given by

$$f_L(x) := \begin{cases} 1, & \text{if } x \in L, \\ 0, & \text{otherwise} \end{cases}$$

for all $x \in \{0,1\}^*$.

For sets $K, L \subseteq \{0,1\}^*$, the *disjoint union of K and L*, denoted $K \uplus L$, is the set $\{0x \mid x \in K\} \cup \{1x \mid x \in L\}$.

For a failure parameter function $\alpha : \mathbb{N} \to [0,1]$, we say that a language L is *α-hard-on-average (α-HoA)* if its characteristic function f_L is α-HoA. Similarly we define average-case easiness for languages.

We prove the following.

▶ **Proposition 24.** *Let L be a language in* NP *that is α-HoA for some failure parameter function $\alpha : \mathbb{N} \to [0,1]$. Then, the language $L^* := L \uplus \text{SAT}$ is* NP*-complete and α^*-HoA, where $\alpha^* : \mathbb{N} \to [0,1]$ is a failure parameter function such that $\alpha^*(n) := \alpha(n-1) - 1/2$ for all naturals $n \geq 2$.*

Before we prove Proposition 24, we recount the following basic observation.

▶ **Lemma 25.** NP *is closed under disjoint union.*

We now turn to the proof of Proposition 24.

Proof of Proposition 24. By Lemma 25, the language L^* is in NP since L^* is the disjoint union of $L \in \mathsf{NP}$ and $\text{SAT} \in \mathsf{NP}$.

We will now show that L^* is NP-hard, by giving a polynomial-time reduction R from SAT to L^*. For all $x \in \{0,1\}^*$, let $R(x) := 1x \in \{0,1\}^*$. We see that R is polynomial-time computable. Moreover, if $x \in \text{SAT}$, then $R(x) = 1x \in L^*$, and if $R(x) \in L^*$, then $1x \in L^*$ and so $x \in \text{SAT}$.

What is left is to prove that L^* is α^*-HoA, where $\alpha^* : \mathbb{N} \to [0,1]$ is such that $\alpha^*(n) := \alpha(n-1) - 1/2$ for all naturals $n \geq 2$. Towards a contradiction, assume that L^* is $(1-\alpha^*)$-EoA and let H^* be a heuristic that witnesses this phenomenon. We will give a heuristic H that witnesses the fact that L is $(1-\alpha)$-EoA, whereby establishing the desired contradiction. To this end, let

$$H(x) := H^*(0x)$$

for all $x \in \{0,1\}^*$. We will show that H has the desired average-case performance. Indeed,

$$\begin{aligned}
\Pr_{x \sim \{0,1\}^n}[H(x) = f_L(x)] &= \Pr_{x \sim \{0,1\}^n}[H^*(0x) = f_{L^*}(0x)] \\
&= \Pr_{y \sim \{0,1\}^{n+1}}[H^*(y) = f_{L^*}(y) \mid y_1 = 0] \\
&\geq \Pr_{y \sim \{0,1\}^{n+1}}[H^*(y) = f_{L^*}(y)] - \Pr_{y \sim \{0,1\}^{n+1}}[y_1 = 1] \\
&\geq 1 - \alpha^*(n+1) - \frac{1}{2} \\
&= 1 - \left(\alpha((n+1)-1) - \frac{1}{2}\right) - \frac{1}{2} \\
&= 1 - \alpha(n)\,.
\end{aligned}$$

◀

B McKTP is NP-complete under randomized reductions

In this section, we prove Theorem 2 by adapting Ilango's work [15].

B.1 Set Cover

We first fix some notation about Set Cover.

▶ **Definition 26.** *The* Set Cover *problem is defined as follows.*

- *Input: A tuple $(n, S_1, \dots, S_t)$ in binary, where $n \in \mathbb{N}$ and $S_1, \dots, S_t \subseteq [n]$ are sets such that $[n] \subseteq \bigcup_{i=1}^{t} S_i$.*
- *Output: The value of*

$$\min_{I \subseteq [t]} \left\{ |I| \mid [n] \subseteq \bigcup_{i \in I} S_i \right\}.$$

Dinur and Steurer [10] show that it is NP-hard to approximate Set Cover.

▶ **Theorem 27** ([10])**.** *It is* NP*-hard to approximate Set Cover by a factor of at most* $(1 - o(1)) \ln n$.

B.2 Approximation algorithms

In the following, we will adopt the following notion of an approximation algorithm.

▶ **Definition 28.** *Let Π be an optimization problem. For all instances $I \in \{0,1\}^*$ of Π, let the optimal solution of I be denoted by $\mathrm{OPT}(I) \in \mathbb{R}$. Let $\alpha > 0$. We say that a probabilistic algorithm A* approximates Π by a factor of α *if, for all instances I of Π, it is the case that*

$$\mathrm{OPT}(I) < A(I) \leq \alpha \cdot \mathrm{OPT}(I)$$

with probability at least $1 - o(1)$ over the internal randomness of A.

B.3 Proof of Theorem 2

For a string b of length m and a set $R \subseteq [m]$, let $b_{\langle R \rangle}$ be the string of length m where

$$b_{\langle R \rangle}(j) := \begin{cases} b(j)\,, & \text{if } j \in R, \\ 0, & \text{otherwise} \end{cases}$$

for all $1 \leq j \leq m$. Equivalently,

$$b_{\langle R \rangle}(j) := b(j) \wedge \mathbb{1}_{j \in R}$$

for all $j \in [m]$.

Next, we define a uniformly random partition $\mathcal{P} = (P_1, \dots, P_n)$ of $[m]$ into n parts to be such that each element $i \in [m]$ is put into P_j where $j \in [n]$ is chosen uniformly at random. It will be also useful to think of $\mathcal{P}$ as a uniformly random function $P : [m] \to [n]$.

For a partition $\mathcal{P} = (P_1, \dots, P_n)$ of $[m]$ and any set $S \subseteq [n]$, we define the *$\mathcal{P}$-lift of S*, denoted $S^{\mathcal{P}}$, to be the set

$$S^{\mathcal{P}} := \bigcup_{i \in S} P_i.$$

Following Ilango [15], we show that McKTP can be used to approximate Set Cover.

▶ **Lemma 29** (Following Ilango [15]). *Let $S_1, \ldots, S_t \subseteq [n]$ be sets that cover $[n]$. Let b be a string of length $m \geq (nt)^5$ and let $\mathcal{P} = (P_1, \ldots, P_n)$ be a uniformly random partition of $[m]$ into n parts. Define the oracle $O : \{0,1\}^{\log t} \times \{0,1\}^{\log m} \to \{0,1\}$ to be such that*

$$O(i,z) := \begin{cases} b_{\langle S_i^{\mathcal{P}} \rangle}(z), & \text{if } i \in [t], \\ 0, & \text{otherwise,} \end{cases}$$

for all $i \in [t]$ and $z \in [m]$. Let y be the truth table of O, and note that $|y| = mt$. Let ℓ be the size of an optimal cover of $[n]$ by $S_1, \ldots, S_t$. Then, we have that

1. $\mathrm{KT}(b \mid y) \leq 200\ell(\log t + \log m)$ *and*
2. $\mathrm{KT}(b \mid y) > \ell(\log t + \log m)/2$ *with high probability over the choice of b.*

Proof. We prove each item of Lemma 29 separately.

▷ Claim 30. It is the case that $\mathrm{KT}(b \mid y) \leq 200\ell(\log t + \log m)$.

Proof. Assume that an optimal set cover of size ℓ is realized by the sets $S_{i_1}, \ldots, S_{i_\ell}$. Fix some UTM U that has oracle access to y. Let $\Pi \in \{0,1\}^*$ be a program that contains in its description encodings of $i_1, \ldots, i_\ell \in \{0,1\}^t$ and operates as follows:

On input $x \in \{0,1\}^{\log m}$, compute and output $y_{(i_1,x)} \vee \cdots \vee y_{(i_\ell,x)}$.

Note that $|\Pi| \leq (\ell+2)\log t + O(1) \leq 100\ell \log t$. In what follows, let $T \in \mathbb{N}$ be a sufficiently large run-time bound such that

$$\begin{aligned} U^{\Pi,y}(x, 1^T) &:= y_{(i_1,x)} \vee \cdots \vee y_{(i_\ell,x)} \\ &= O(i_1,x) \vee \cdots \vee O(i_\ell,x) = \bigvee_{i \in [\ell]} \bigvee_{j \in S_i} b_{\langle P_j \rangle}(x) = \bigvee_{j \in [n]} b_{P_j}(x) = b(x), \end{aligned}$$

for all $x \in \{0,1\}^{\log m}$. Note that $T \leq 100\ell(\log t + \log m)$. Therefore, we have that $\mathrm{KT}(b \mid y) \leq 200\ell(\log t + \log m)$. ◁

We now turn to the lower bound. We do this by a union bound argument. Fix some oracle program $M^y(\cdot) := U^{\Pi,y}(\cdot, 1^T)$ of program Π that uses oracle y and runs in time T such that $|\Pi| + T \leq \ell(\log t + \log m)/2$. Then, as each oracle query requires time $\log t + \log m$, we can deduce that M makes at most $\ell/2 \leq n/2 \leq n$ oracle queries to y.

We will show that

$$\Pr_{b,\mathcal{P}}[M^y \text{ computes } b \text{ in time } T, \text{ and } |\Pi| + T \leq \ell(\log t + \log m)/2]$$

is exponentially small. We do this by finding a long sequence of inputs $x_1, \ldots, x_d$ on which M has not too large a chance of computing b.

We construct this list recursively, as follows. Let $x_1 := 0^{\log m}$, and let

$$Q_1 := \left\{ x \in \{0,1\}^{\log m} \mid M^y(x_1) \text{ makes a query } (i,x) \text{ to } y, \text{ for some } i \in [t] \right\}.$$

Now, for $j \geq 1$, if $\{0,1\}^{\log m} \setminus Q_j$ is non-empty, then let x_{j+1} be an element of $\{0,1\}^{\log m} \setminus Q_j$, and let

$$Q_{j+1} := Q_j \cup \left\{ x \in \{0,1\}^{\log m} \mid M^y(x_{j+1}) \text{ makes a query } (i,x) \text{ to } y, \text{ for some } i \in [t] \right\}.$$

If $\{0,1\}^{\log m} = Q_j$, then terminate the sequence. Since M makes at most n queries to y, we know that $|Q_j| \le jn$. Thus, since $|Q_d| = \left|\{0,1\}^{\log m}\right| = m$ the length of this sequence is at least m/n. That is, $d \ge m/n$.

It remains to bound the probability

$$\mathbf{Pr}[\text{for all } j \in [d],\ M^y(x_j) = b(x_j)] = \prod_{j=1}^{d} \mathbf{Pr}\left[M^y(x_j) = b(x_j) \mid \bigwedge_{k\in[j-1]} M^y(x_k) = b(x_k)\right].$$

Fix some $j \in [d]$. We will bound

$$\mathbf{Pr}\left[M^y(x_j) = b(x_j) \mid \bigwedge_{k\in[j-1]} M^y(x_k) = b(x_k)\right].$$

Let $E := \bigwedge_{k\in[j-1]} M^y(x_k) = b(x_k)$ be the event that we are conditioning on.

▷ Claim 31. It is the case that

$$\mathbf{Pr}[M^y(x_j) = b(x_j) \mid E] \le 1 - \frac{1}{2n}.$$

Proof. By construction of the sequence $x_1, \dots, x_d$, we know that on all the inputs $x_1, \dots, x_{j-1}$, the program M^y does not make an oracle call of the form (i, x_j) for any i. Thus, the only time the value of O depends on $b(x_j)$ and $P(x_j)$ is on inputs of the form (i, x_j) for some i, and since $b(x_j)$ and $P(x_j)$ are chosen independently at random, we know that $b(x_j)$ and $P(x_j)$ are still uniform random variables conditioned on E. That is,

$$\mathbf{Pr}[b(x_j) = 1 \mid E] = \frac{1}{2}$$

and

$$\mathbf{Pr}[P(x_j) = r \mid E] = \frac{1}{n}$$

for all $r \in [n]$.

Now, define O' as

$$O'(i,x) := \begin{cases} 0, & \text{if } x = x_j, \\ O(i,x), & \text{otherwise,} \end{cases}$$

and let y' be the truth table of O'. Let also $i_1, \dots, i_v$ with $v \le \ell/2$ be such that, using the modified oracle O', they are the only oracle queries $M^{y'}(x_j)$ makes that have x_j as the 2nd component of the query, so the queries are $(i_1, x_j), \dots, (i_v, x_j)$. Since $v < \ell$ there exists an element r^* that is not in $S_{i_1} \cup \dots \cup S_{i_v}$.

Moreover, observe that if $P(x_j) = r^*$, then $M^y(x_j)$ will actually make the same oracle queries (and get the same zero responses) as the modified oracle program $M^{y'}$. In this case, since $P(x_j) = r^*$ is not in $S_{i_1} \cup \dots \cup S_{i_v}$, it follows that

$$O(i_1, x_j) = \dots = O(i_v, x_j) = 0$$

regardless of the value of $b(x_j)$. Thus, the output of M^y on input x does not depend at all on the value of $b(x)$ if $P(x_j) = r^*$. Hence, the probability it correctly guesses $M^y(x) = b(x)$ is at most half when $P(x_j) = r^*$.

Since $P(x_j)$ is chosen uniformly at random, we have that $P(x_j) = r^*$ with probability $1/n$. Therefore,

$$\mathbf{Pr}[M^y(x_j) = b(x_j) \mid E] \leq 1 - \frac{1}{2n}$$

and the proof os complete. ◁

Using Claim 31, we have

$$\prod_{j=1}^{d} \mathbf{Pr}\left[M^y(x_j) = b(x_j) \mid \bigwedge_{k \in [j-1]} M^y(x_k) = b(x_k) \right] \leq \left(1 - \frac{1}{2n}\right)^d$$
$$\leq e^{-d/(2n)} \leq e^{-m/\left(2n^2\right)} \leq e^{-n^3 t^5/2}.$$

On the other hand the number of oracle programs of size at most $\ell(\log t + \log m)/2 \leq O(nt\log n)$ is at most $2^{O\left(n^2 t\right)}$. Thus, by a union bound, the probability that there exists an oracle program Π that computes any bit of b in time T, whereby $|\Pi| + T \leq \ell(\log t + \log m)/2$, is $o(1)$ as desired. ◀

Lemma 29 implies the following corollary.

▶ **Corollary 32.** *There is a polynomial-time computable function $M : \mathbb{N} \to \mathbb{N}$ such that the following hold. Given a Set Cover instance $I := (n, S_1, \ldots, S_t)$, a random b of length $N \geq (nt)^5$ and a random partition P of $[N]$ into n parts, if one constructs a string y as in Lemma 29, whereby $|y| \leq M(N)$, then* $\mathrm{KT}(b \mid y)$ *approximates Set Cover by a factor of* 400 *according to Definition 28. That is, if ℓ is the size of an optimal set cover of I and $c := \log N + \log t$, then it is the case that with probability* 1

$$\frac{2}{c} \cdot \mathrm{KT}(b \mid y) \leq 400\ell,$$

and with probability $1 - o(1)$

$$\frac{2}{c} \cdot \mathrm{KT}(b \mid y) > \ell.$$

Proof. Let $y \in \{0,1\}^*$, $n \in \mathbb{N}$, and $t \in \mathbb{N}$ be as in Lemma 29. Let $\gamma := 1/2$. Then, $\mathrm{McKT}^M\mathrm{P}$ of dimension $N := |b| \geq (nt)^5$ and $M := N^{1+\gamma} = N^{1+1/2} = N \cdot N^{1/2} \geq Nt = |y|$ is such that Lemma 29 immediately implies that

$$\ell < \frac{2}{c} \cdot \mathrm{KT}(b \mid y) \leq 400\ell,$$

where the first inequality holds with probability $1 - o(1)$ and the second one holds with probability 1. ◀

Theorem 27 and Corollary 32 yield the following corollary.

▶ **Corollary 33.** *There exists a polynomial-time computable function $m : \mathbb{N} \to \mathbb{N}$ such that* $\mathrm{McKT}^m\mathrm{P}$ *is* NP*-hard under polynomial-time randomized reductions.*

Finally, by combining Lemma 11 and Corollary 33 we get a proof of Theorem 2.

▶ **Corollary 34** (Theorem 2, restated)**.** *There exists a polynomial-time computable function $m : \mathbb{N} \to \mathbb{N}$ such that* $\mathrm{McKT}^m\mathrm{P}$ *is* NP*-complete under polynomial-time randomized reductions.*