NP-Hardness of Learning Programs and Partial MCSP

Shuichi Hirahara
Principles of Informatics Research Division
National Institute of Informatics
Tokyo, Japan
s_hirahara@nii.ac.jp

***Abstract*—A long-standing open question in computational learning theory is to prove NP-hardness of learning efficient programs, the setting of which is in between proper learning and improper learning. Ko (COLT'90, SICOMP'91) explicitly raised this open question and demonstrated its difficulty by proving that there exists no relativizing proof of NP-hardness of learning programs. In this paper, we overcome Ko's relativization barrier and prove NP-hardness of learning programs under randomized polynomial-time many-one reductions. Our result is provably non-relativizing, and comes somewhat close to the parameter range of improper learning: We observe that mildly improving our inapproximability factor is sufficient to exclude Heuristica, i.e., show the equivalence between average-case and worst-case complexities of NP.**

We also make progress on another long-standing open question of showing NP-hardness of the Minimum Circuit Size Problem (MCSP). We prove NP-hardness of the partial function variant of MCSP as well as other meta-computational problems, such as the problems MKTP^* and MINKT^* of computing the time-bounded Kolmogorov complexity of a given partial string, under randomized polynomial-time reductions.

Our proofs are algorithmic information (a.k.a. Kolmogorov complexity) theoretic. We utilize black-box pseudorandom generator constructions, such as the Nisan–Wigderson generator, as a one-time encryption scheme secure against a program which "does not know" a random function. Our key technical contribution is to quantify the "knowledge" of a program by using conditional Kolmogorov complexity and show that no small program can know many random functions.

***Index Terms*—minimum circuit size problem, Kolmogorov complexity, PAC learning, pseudorandomness**

I. Introduction

The two main results established in this paper are the NP-hardness of MINLT [2] and the partial function variant of MCSP [3], both of which are of significant importance in computational learning theory and meta-complexity theory. In the following two subsections, we present the backgrounds of the two problems and our results.

A. PAC Learning

Ever since Valiant [4] introduced the notion of PAC learning, classification of its complexity has been a fundamental and central question in computational learning theory. PAC learning is parameterized by a *concept class* $\mathcal{C}$ and a *hypothesis class* $\mathcal{H}$. Informally, a class $\mathcal{C}$ is said to be *PAC learnable by* $\mathcal{H}$ if there exists an efficient algorithm L such that for every distribution $\mathcal{D}$ and for every concept $c \in \mathcal{C}$, given sufficiently many random samples $(x_1, c(x_1)), \ldots, (x_m, c(x_m))$, where each x_i is drawn from $\mathcal{D}$ independently, the learning algorithm L outputs a hypothesis $h \in \mathcal{H}$ such that $\Pr_{x\sim\mathcal{D}}[h(x) = c(x)] \geq 1 - \delta$ for a given parameter δ. A fundamental theorem in computational learning theory [5]–[7] states that, for a sufficiently large hypothesis class $\mathcal{H}$, PAC learning is equivalent to Occam learning, which can be formulated as a search problem in NP.[1] An outstanding question is whether PAC learning of linear-sized circuits by polynomial-sized circuits is at least as hard as solving NP, which would establish the "NP-completeness" of PAC learning.

NP-hardness of PAC learning has been proved in the case of *proper learning*, i.e., when $\mathcal{C} = \mathcal{H}$. Pitt and Valiant [8] proved NP-hardness of learning k-term DNF by k-term DNF. This was extended to the NP-hardness of learning linear-sized DNF formulas by polynomial-sized disjunction of half-spaces [9]; i.e., $\mathcal{C} = \{\text{DNF formulas}\}$ and $\mathcal{H} = \mathrm{OR} \circ \{\text{halfspaces}\}$. Note that, as the hypothesis class $\mathcal{H}$ becomes larger, it becomes increasingly harder to prove NP-hardness. For example, consider the class NC^1 of fan-in-2 circuits of logarithmic depth. Intuitively, PAC learning of NC^1 by NC^1 appears to be much harder than PAC learning of DNFs by DNFs since NC^1 is larger than the class of DNF formulas. However, NP-hardness of learning linear-sized NC^1 by polynomial-sized NC^1 is currently unknown.

Contrary to proper learning, *improper learning* does not involve any other restriction on the hypothesis class $\mathcal{H}$ except the requirement that $\mathcal{H}$ must be evaluated in polynomial time. As noted in the seminal work of Valiant [4], it is possible to prove hardness of improper PAC learning under cryptographic assumptions. A recent exciting line of research (e.g., [10]–[12]) based on Daniely, Linial, and Shalev-Shwartz [13] has demonstrated that specific average-case hardness assumptions of NP already imply hardness of improper PAC learning. However, there is a fundamental obstacle that prevents us from proving NP-hardness of improper PAC learning. Applebaum, Barak, and Xiao [14] showed that NP-hardness of improper

This work was supported by JST, PRESTO Grant Number JPMJPR2024, Japan. The full version of this paper is available on ECCC [1].

[1]The task of Occam learning is to output a short description of a hypothesis that is consistent with given samples $(x_1, c(x_1)), \ldots, (x_m, c(x_m))$ for an unknown concept c.

2575-8454/22/$31.00 ©2022 IEEE
DOI 10.1109/FOCS54457.2022.00095

PAC learning cannot be proved using nonadaptive reductions unless the polynomial hierarchy collapses. They also proved that NP-hardness of improper PAC learning excludes Pessiland [15] from Impagliazzo's five worlds, i.e., it implies the equivalence between the existence of one-way functions and average-case hardness of NP. Moreover, a partial converse to [13] was recently proved by Hirahara and Nanashima [16]: PAC learning with respect to distributions samplable by polynomial-size circuits is feasible under the assumption that NP is easy on average. Thus, the complexity of improper PAC learning is intimately related to the average-case complexity of NP. In particular, proving NP-hardness of PAC learning (with respect to P/poly samplable distributions) also excludes Heuristica from Impagliazzo's five worlds [15], i.e., it establishes the equivalence between worst-case and average-case complexities of NP. These previous works highlight the importance and, simultaneously, the difficulty of proving NP-hardness of improper PAC learning.

Ko [2] raised the question of classifying the complexity of learning efficient programs, i.e., PAC learning by $\mathcal{H}$, where the hypothesis class $\mathcal{H}$ is the class of efficient programs. Arguably, this is the most general hypothesis class: By the fundamental principle of Kolmogorov complexity [17], the Kolmogorov complexity of a string x, i.e., the length of a shortest program that prints x, is a lower bound for the length of any decodable encoding of the string x up to an $O(1)$ additive term. Similarly, Ko observed that representing a function using a program is the most succinct way of representing it using any algorithm up to an $O(1)$ additive term. In particular, programs can represent functions more succinctly than circuits.

Ko formulated the task of learning efficient programs by introducing a problem called MINLT, which is the decision version of Occam learning for efficient programs. The input of MINLT consists of $((x_1, b_1), \ldots, (x_m, b_m); 1^t, 1^s)$, where $x_i \in \{0,1\}^n$ and $b_i \in \{0,1\}$ for some $n \in \mathbb{N}$, and the objective is to decide whether there exists a t-time program M of size s that is *consistent* with the given samples $(x_1, b_1), \ldots, (x_m, b_m)$, i.e., $M(x_i) = b_i$ for every i. Since the complexity of MINLT "appears very difficult to classify precisely" [2], Ko presented a formal evidence for this statement by proving that there exists no relativizing proof that establishes the NP-hardness of MINLT. A *relativizing proof*—a notion introduced by Baker, Gill, and Solovay [18] to argue the difficulty of resolving the P versus NP question—is a proof of a complexity-theoretic statement that can be generalized to statements in the presence of arbitrary oracles. Although there are several non-relativizing proof techniques (see, e.g., [19], [20]), the vast majority of complexity-theoretic proofs are relativizing; thus, proving a non-relativizing statement is highly challenging and of major importance in complexity theory. Indeed, we are not aware of any previous non-relativizing result in complexity theory for which relativization barriers were presented three decades ago.[2]

In this paper, we overcome Ko's relativization barrier and resolve the long-standing open problem of proving the NP-hardness of MINLT.

Theorem I.1. *Under randomized polynomial-time one-query reductions, it is* NP*-hard to distinguish the following cases, given as input a size parameter $s \in \mathbb{N}$ and a distribution $\mathcal{E}$ such that* $\mathrm{supp}(\mathcal{E}) \subseteq \{0,1\}^n \times \{0,1\}$ *for some* $n \in \mathbb{N}$.[3]

Yes: *There exists a polynomial-time program M of size s such that*

$$\Pr_{(x,b)\sim\mathcal{E}}[M(x) = b] = 1.$$

Moreover, M computes a linear function over $\mathrm{GF}(2)$.

No: *For any program M of size $s \cdot n^\epsilon$,*

$$\Pr_{(x,b)\sim\mathcal{E}}[M(x) = b] \leq \frac{1}{2} + 2^{-n^{1-\delta}}$$

Here, $\delta > 0$ is an arbitrary constant and $\epsilon = 1/(\log\log n)^{O(1)}$.

A couple of remarks are in order. Firstly, the problem considered in Theorem I.1 can be reduced to MINLT by drawing m samples $(x_1, b_1), \ldots, (x_m, b_m)$ from the distribution $\mathcal{E}$ for $m = O(s)$; thus, the theorem also shows the NP-hardness of MINLT under randomized reductions as an immediate corollary. Our result is *provably* non-relativizing. Although Ko [2] stated his relativization barrier for deterministic reductions, it can be extended to randomized reductions. See the full version for the details. Secondly, Theorem I.1 refers to the decision version of PAC learning of a concept class $\mathcal{C}$ by a hypothesis class $\mathcal{H}$, where $\mathcal{C}$ denotes the class of efficient programs of size s (that compute a linear function) and $\mathcal{H}$ denotes the class of *time-unbounded* programs of size $s \cdot n^\epsilon$. Note that the decision version is reducible to the standard search version, as long as $\mathcal{H}$ can be evaluated in polynomial time; thus, another corollary of Theorem I.1 is the NP-hardness of PAC learning of $\mathcal{C}$ by $\mathcal{H}'$, where $\mathcal{H}' \subseteq \mathcal{H}$ denotes the class of polynomial-time programs of size $s \cdot n^\epsilon$. Let us emphasize that, in the NO case, even *time-unbounded* programs fail to output b on input x for a random sample (x, b) drawn from $\mathcal{E}$. Although finding one hypothesis that is consistent with $\mathcal{E}$ is easy using Gaussian elimination, Theorem I.1 implies that deciding whether such a hypothesis can be compressed as a small time-unbounded program is NP-hard. Thirdly, the probability $\frac{1}{2} + 2^{-n^{1-\delta}}$ in the NO case is close to the optimal, as a trivial hypothesis that always outputs either 0 or 1 agrees with the samples from $\mathcal{E}$ with probability at least $\frac{1}{2}$. Although several papers have been published on the NP-hardness of learning with large

[2] As discussed in [20, Section 9], there are several non-relativizing statements that exploit the subtleties in defining oracle access mechanism. These include statements on small depth circuits and bounded-space Turing machines. Whether such results are "truly" non-relativizing is a controversial question. In contrast, Theorem I.1 can be naturally relativized using the standard definition of oracle Turing machines.

[3] A distribution $\mathcal{E}$ is represented by a circuit C such that the output $C(r)$ of C over a uniformly random string $r \sim \{0,1\}^{|C|}$ is identical to $\mathcal{E}$. The support of $\mathcal{E}$ is denoted by $\mathrm{supp}(\mathcal{E})$.

error (e.g., [9], [21]–[25]), most results establish NP-hardness of learning with error $\frac{1}{2} - \epsilon$ for a constant $\epsilon > 0$; we are not aware of any previous result that achieves the exponentially small correlation bound of $2^{-n^{1-\delta}}$.

Theorem I.1 comes somewhat close to the range of parameters for which proof techniques on improper PAC learning can be applied. For example, using the proof techniques of [16], we observe that improving the inapproximability factor n^{ϵ} to $1.01n$ is sufficient to exclude Heuristica; see the full version for further details.

B. Meta-Complexity

A problem closely related to PAC learning is the *Minimum Circuit Size Problem* [3] (MCSP). The input of MCSP consists of a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$ represented as the truth table of length 2^n as well as a size parameter $s \in \mathbb{N}$, and the task is to decide whether there exists a circuit of size s that computes f. The study of MCSP dates back to the 1960s [26]. Although it is easy to see that MCSP $\in$ NP, it is a long-standing open problem to prove NP-hardness of MCSP—indeed, it was reported in [27] that Levin [28] delayed the publication of his seminal work on the theory of NP-completeness because he hoped to prove the NP-completeness of MCSP.

More generally, MCSP is an example of *meta-computational problems*. Meta-complexity refers to the computational complexity of problems that themselves consider complexity. The aforementioned work of Ko [2] introduced the problem MINKT of deciding whether a given string x can be printed by a t-time program of size s, given $(x, 1^t, 1^s)$ as input. Similarly, MKTP is the problem of deciding whether each bit of a given string x can be computed by a t-time program of size s for some (t, s) such that $t + s \leq \theta$, given (x, θ) as input [29]. All of the aforementioned problems are meta-computational—MCSP considers the circuit complexity of a given string; MINKT considers the time complexity of printing a given string (i.e., time-bounded Kolmogorov complexity); MKTP considers the trade-off between the time complexity and the size complexity of printing a given string. Technically, MKTP is often considered to be a convenient surrogate for MCSP, for which many theorems that are not known to hold for MCSP can be proved (e.g., [30], [31]). Meta-complexity has recently received significant attention because of its connection to diverse areas of theoretical computer science, including learning theory [16], [32], average-case complexity [33], [34], cryptography [35]–[37], and circuit lower bounds [38], [39]; see the survey of Allender [40] for a broad overview.

Although none of the meta-computational problems mentioned above has been shown to be NP-hard, there has been recent substantial progress to prove the NP-hardness of meta-computational problems. For restricted circuit classes $\mathfrak{C} \in \{\mathrm{DNF}, \mathrm{DNF} \circ \mathrm{XOR}, \mathsf{AC}^0 \text{ formulas}\}$, the corresponding versions of MCSP (denoted by $\mathfrak{C}$-MCSP) were shown to be NP-hard [41]–[44]. More recently, Ilango [45] proved that the formula variant of MCSP is hard under the Exponential-Time Hypothesis (ETH) [46].

A well-trodden path in the proofs of NP-hardness of variants of MCSP consists of the following two steps. First, the NP-hardness of the partial function variants of MCSP, which are often denoted by MCSP*, is proved. The input of MCSP* consists of a partial function $f\colon \{0,1\}^n \to \{0,1,*\}$ (encoded as a string of length $2^{O(n)}$) and a size parameter $s \in \mathbb{N}$, and the task is to decide whether there exists a circuit of size s that computes $f(x)$ on input x such that $f(x) \neq *$. Second, the partial variants of MCSP are reduced to MCSP. For example, the NP-hardness of DNF-MCSP presented in [42] is proved by composing two reductions—a reduction from NP to DNF-MCSP* and a reduction from DNF-MCSP* to DNF-MCSP. Other works [43], [45] follow the same paradigm. Thus, proving the NP-hardness of partial variants of meta-computational problems serves as a milestone towards proving the NP-hardness of the total versions of meta-computational problems. In fact, Levin's seminal paper [28] presented six NP-complete problems; the second problem shown to be NP-complete was DNF-MCSP*. Since the introduction of the theory of NP-completeness [28], [47], extension of Levin's NP-completeness result to MCSP* has been a long-standing open problem.

The difficulty of proving the NP-hardness of MCSP is closely related to the inability of proving explicit circuit lower bounds. Kabanets and Cai [3] proved that if MCSP is NP-hard under deterministic "natural" reductions, then EXP $\not\subseteq$ P/poly.[4] Their proof techniques are applicable to MCSP* as well. Since EXP $\not\subseteq$ P/poly is a major open question in complexity theory, this suggests that proving NP-hardness of MCSP* would be quite difficult, at least under deterministic reductions. A subsequent line of work improved the barrier result of [3] to more general types of reductions, such as "non-natural" reductions [48], nonadaptive deterministic reductions [49], and adaptive deterministic reductions [50], [51]. The intuition behind the aforementioned results is that if there is a many-one reduction from SAT to MCSP*, then, given an unsatisfiable formula as input, it must produce the truth table of a function f with high circuit complexity, which implies a circuit lower bound for EXP.

To avoid this barrier, one may be tempted to consider randomized reductions. A randomized reduction can easily produce a function with high circuit complexity, as a random function has high circuit complexity with high probability. Thus, the barriers mentioned above do not apply to randomized reductions. Unfortunately, Hirahara and Watanabe [50] presented some evidence for the difficulty of proving the NP-hardness of MCSP even under randomized reductions. They observed that most reductions to MCSP are *oracle-independent*, i.e., they can be generalized to the Minimum A-Oracle Circuit Size Problem (denoted by MCSP^A) for every oracle A. Then, they proved that there is no one-query

[4] A *natural reduction to* MCSP is a reduction such that the size parameter in the output of the reduction depends only on the input length.

randomized polynomial-time reduction from NP to MCSP^A for some oracle A unless the polynomial hierarchy collapses.

Another approach for avoiding the barriers is to use exponential-time reductions. Ilango [44] recently bypassed the barrier of [3] and proved ETH-hardness of MCSP^* under deterministic reductions. The intuition behind this result is that lower bounds against $O(n)$-size circuits can be proved using gate elimination techniques; since it is already known that there exists an explicit function that cannot be computed by a circuit of size cn for a constant c, the barrier of [3] does not apply. Such proof techniques based on deterministic reductions, however, are unlikely to be extended to NP-hardness of MCSP^*, as the barriers of [3] come into play. Moreover, it would be extremely difficult to prove the NP-hardness of MKTP^* and MINKT^* under deterministic reductions, as there are few techniques to prove a lower bound of the time-bounded Kolmogorov complexity of an explicit string. Here, MKTP^* and MINKT^* are the partial variants of MKTP and MINKT, which ask the minimum of the time-bounded Kolmogorov complexity of y over all the strings $y \in \{0,1\}^*$ that are *consistent with*[5] a given partial string $x \in \{0,1,*\}^*$.

Using the techniques used to prove the NP-hardness of PAC learning, we circumvent the barrier of [3] by means of randomized reductions and prove the NP-hardness of partial variants of several meta-computational problems.

Theorem I.2. *Under randomized polynomial-time one-query reductions, it is* NP*-hard to distinguish the following two cases, given a partial function $f\colon \{0,1\}^n \to \{0,1,*\}$ (encoded as a string over $\{0,1,*\}$ of length 2^n), a size parameter $s \in \mathbb{N}$, and a distribution $\mathcal{D}$ over $f^{-1}(\{0,1\})$:*

Yes: *There exist an s-time[6] program M of size s and a circuit C of size $\frac{s}{\log s}$ and depth $O(\log s)$ such that*

$$\Pr_{x\sim\mathcal{D}}[M(x) = f(x)] = 1 \text{ and } \Pr_{x\sim\mathcal{D}}[C(x) = f(x)] = 1.$$

No: *For every program M of size $s \cdot n^\epsilon$ and for every circuit C of size $\frac{s}{\log s} \cdot n^\epsilon$,*

$$\Pr_{x\sim\mathcal{D}}[M(x) = f(x)] \le \frac{1}{2} + n^{-\epsilon} \text{ and}$$
$$\Pr_{x\sim\mathcal{D}}[C(x) = f(x)] \le \frac{1}{2} + n^{-\epsilon}.$$

Here, $\epsilon > 0$ is a universal constant. In particular, MCSP^**,* NC^1-MCSP^**,* MKTP^**, and* MINKT^* *are* NP*-hard under randomized polynomial-time reductions. Moreover, these problems are* NP*-hard to approximate within a factor of $(\log N)^\epsilon$ on inputs of length N.*

Our proofs are inspired by a line of research [42], [43], [52]–[56] that developed "top-down" approaches to prove the NP-hardness of MCSP.[7] Theorem I.2 is proved using techniques that are fundamentally different from previous results of [44] and significantly improves them from the following perspectives:

1) The time complexity of a reduction is improved from exponential to polynomial.
2) The hardness of approximation is proved in Theorem I.2, unlike in [52].
3) The NP-hardness of MKTP^* and MINKT^* is proved.
4) In addition, by slightly modifying the NP-hardness reduction of MCSP^*,[8] we also prove the NP-hardness of the average-case variant of MCSP called AveMCSP [36], which asks the *average-case* circuit complexity of a given *total* function $f\colon \{0,1\}^n \to \{0,1\}$ with respect to the uniform distribution.

The size parameter s in Theorem I.2 is exponential in the input length n (i.e, $s = 2^{\Theta(n)}$), which is inevitable unless NP can be solved in randomized sub-exponential time.[9] The strong circuit lower bound of the NO case would be surprising—it indicates that no circuit of exponential size can compute the function f. Given our poor knowledge of circuit lower bounds for explicit functions,[10] naïvely, one would expect any proof of NP-hardness of MCSP^* to require a complicated analysis of exponential-sized circuits. This is indeed the primary difficulty that has hindered the proof of NP-hardness of MCSP^* (and formalized as the barrier of [3] in the case of deterministic reductions). Surprisingly, we prove the circuit lower bound using *almost nothing* about circuits—the circuit lower bound simply follows from the fact that no program of size s can approximate f. Since a circuit of size s' can be encoded as a binary string of length $O(s' \log s')$, we obtain a circuit lower bound of $s' \ge \Omega(s/\log s)$. In contrast, the construction of the circuit C in the YES case is more complicated.

All the previous NP-hardness results for $\mathfrak{C}$-MCSP are proved by different reductions for each circuit class $\mathfrak{C}$. Ideally, one would like to prove NP-hardness of $\mathfrak{C}$-MCSP for any sufficiently large circuit class $\mathfrak{C}$ by a single reduction. Unfortunately, this is not possible—the aforementioned work of [50] proves that MCSP is not reducible to MCSP^A for some oracle A (unless MCSP is easy), despite the fact that A-oracle circuits are more powerful than circuits. Surprisingly, Theorem I.2 indicates that the same is not true for partial variants of MCSP. Our reduction is, in fact, oracle-independent in the sense that it also proves the NP-hardness of MKTP^{*A} for every oracle A.[11] This indicates that the negative result of

[5]We say that a string $y \in \{0,1\}^n$ is *consistent with* a partial string $x \in \{0,1,*\}^n$ if every bit of x is equal to either $*$ or the bit of y at the corresponding position.

[6]Here, the time complexity of M is measured as in KT-complexity; that is, we assume that a program is given oracle access to the description of M.

[7]Specifically, based on his NP-hardness result of a conditional variant of MCSP, Ilango [52] proposed an approach to prove the hardness of MCSP "from above", as opposed to "bottom-up" approaches of [42]–[45], which attempt to prove the NP-hardness of $\mathfrak{C}$-MCSP for larger and larger classes $\mathfrak{C}$. Theorem I.2 realizes such a top-down approach successfully.

[8]Specifically, we replace $f(x) = *$ with a uniformly random bit $f(x) \sim \{0,1\}$ in the NP-hardness proof of MCSP^*.

[9]This follows from the fact that MCSP^* with size parameter $s(n)$ can be solved in time $2^{O(s(n)\log s(n))}$ by a brute-force search.

[10]For example, proving $\mathsf{E}^{\mathsf{NP}} \not\subseteq \mathsf{SIZE}(O(n))$ is an important open problem. A function $f \in \mathsf{E}^{\mathsf{NP}}$ is said to be *explicit*.

[11]For a technical reason, our reduction may not prove the NP-hardness of MCSP^{*A}.

[50] for MCSP is unlikely to be extended to the partial variant MCSP*, despite that the negative result of [3] can easily be extended to MCSP*.

Theorem I.2 is proved by optimizing the reduction of Theorem I.1. Note that the NP-hardness reduction of Theorem I.2 takes an NP instance of length N and produces the truth table of a partial function $f\colon \{0,1\}^n \to \{0,1,*\}$, which is of length $2^{O(n)}$, in time $N^{O(1)}$. Thus, we must have $n = O(\log N)$. A high-level idea used in the proof of Theorem I.2 is that the input length n of the distribution $\mathcal{E}$ in Theorem I.1 can be optimized to be as small as $O(\log N)$.

II. An Overview of Our Proofs

In this section, we outline our proofs. At a very high level, our technical contribution is to develop an *algorithmic information (Kolmogorov complexity) theoretic* proof technique for showing lower bounds of the size of programs. We assume familiarity with the notions of Kolmogorov complexity and secret sharing scheme.

Notation: $[n]$ denotes $\{1,\dots,n\}$. For $s_1,\dots,s_n$ and a subset $T \subseteq [n]$, let s_T denote $(s_{i_1},\dots,s_{i_m})$, where $T = \{i_1 < \cdots < i_m\}$.

A. NP-Hardness of Learning Programs

We present an overview of the proof of Theorem I.1, which shows NP-hardness of learning programs. We reduce the Minimum Monotone Satisfying Assignment (MMSA) problem [57] to the problem of learning programs. The input of MMSA consists of a monotone formula φ on n variables and a parameter $\theta \in \mathbb{N}$. The task is to decide whether there exists a satisfying assignment $\alpha\colon [n] \to \{0,1\}$ of φ such that $\sum_{i=1}^n \alpha(i) \le \theta$. Approximation of MMSA to within a factor of $n^{1/(\log\log n)^{O(1)}}$ is known to be NP-hard [58], [59].

To reduce MMSA to the learning problem, we use a secret sharing scheme $(\mathrm{Share}(\varphi,\text{-}), \mathrm{Rec}(\varphi,\text{-}))$ for a monotone formula φ. We say that a subset $T \subseteq [n]$ is *authorized* if the characteristic function $\chi_T\colon [n] \to \{0,1\}$ of T satisfies the formula φ. The secret sharing scheme allows a secret $b \in \{0,1\}$ to be shared among n parties so that any authorized set of parties can reconstruct the secret, whereas no unauthorized set of parties can obtain any information of the secret. At a high level, our reduction produces a distribution $\mathcal{E}$, from which (x,b) is sampled as follows. A random secret $b \sim \{0,1\}$ is shared as $\mathrm{Share}(\varphi,b) =: (s_1,\dots,s_n)$, where each s_i denotes the share given to the i-th party, and then each share s_i is "hidden" in an input x. Our main technical contribution is to develop a way of hiding the shares in an input x so that large programs can read the hidden shares from x, whereas small programs cannot read many hidden shares.

A key tool for hiding shares in an input is a (black-box) pseudorandom generator construction [60]. There are many pseudorandom generator constructions in the literature, such as the Nisan–Wigderson pseudorandom generator [61]. Although most pseudorandom generator constructions can be used for our purpose,[12] for the sake of simplicity, we use a simple pseudorandom generator construction called a *k-wise direct product generator* $\mathrm{DP}_k\colon \{0,1\}^\lambda \times \{0,1\}^{\lambda\times k} \to \{0,1\}^{\lambda k+k}$ [63], which is defined as follows. Given a string $f \in \{0,1\}^\lambda$ (regarded as a row vector over $\mathrm{GF}(2)$) and a $\lambda \times k$ matrix z over $\mathrm{GF}(2)$, the output $\mathrm{DP}_k(f;z)$ is defined to be $(z, f\cdot z)$, where $f\cdot z$ denotes the multiplication of a vector f and a matrix z over $\mathrm{GF}(2)$. The pseudorandom generator construction DP_k satisfies the following "reconstruction" property [64]: If there exists a function D that ϵ-distinguishes the output distribution $\mathrm{DP}_k(f;\text{-})$ from the uniform distribution, i.e.,

$$\left| \Pr_{z\sim\{0,1\}^{\lambda k}}[D(\mathrm{DP}_k(f;z))=1] - \Pr_{w\sim\{0,1\}^{\lambda k+k}}[D(w)=1] \right| \ge \epsilon,$$

then

$$\mathrm{K}^D(f) \le k + O(\log(\lambda k/\epsilon)). \tag{1}$$

In other words, $\mathrm{DP}_k(f;\text{-})\colon \{0,1\}^{\lambda k} \to \{0,1\}^{\lambda k+k}$ is a pseudorandom generator secure against an algorithm D if f is a "hard" function in the sense that f cannot be described using approximately k bits of information and oracle access to D. The pseudorandom generator $\mathrm{DP}_k(f;\text{-})$ can be regarded as a one-time encryption scheme that is secure against any algorithm D that "does not know" f in the sense that $\mathrm{K}^D(f) \gg k$. Besides its simplicity, a useful property of $\mathrm{DP}_k(f;\text{-})$ is that $\mathrm{DP}_k(f;\text{-})$ is a linear function for any fixed f, which enables us to show the linearity property of the YES case in Theorem I.1.

We now present the details of the reduction from MMSA to the learning problem. Let (φ,θ) be an instance of MMSA. We choose n strings $f_1,\dots,f_n \sim \{0,1\}^\lambda$ uniformly at random. Each string f_i is associated with the i-th variable of φ. Using the strings $f_1,\dots,f_n$, we define an example distribution $\mathcal{E} = \mathcal{E}(f_1,\dots,f_n)$ as follows. To sample an example $(x,b) \in \{0,1\}^m \times \{0,1\}$ distributed according to $\mathcal{E}$, we first choose a secret $b \sim \{0,1\}$ randomly. The secret b is shared among n parties using the secret sharing scheme for φ; let $(s_1,\dots,s_n) := \mathrm{Share}(\varphi,b)$ be the set of shares and let $k \in \mathbb{N}$ be the length of each share. The idea is to hide these shares in the input x so that any algorithm that "knows" f_i can read the i-th share and any algorithm that "does not know" f_i cannot read the i-th share. To this end, we let $\xi_i := (z_i, (f_i\cdot z_i)\oplus s_i) = \mathrm{DP}_k(f_i;z_i)\oplus(0^{\lambda k}, s_i)$ for every $i \in [n]$, where $z_i \sim \{0,1\}^{\lambda k}$. Here, "$\oplus$" denotes the bitwise XOR. Then, we define $x := (\xi_1,\dots,\xi_n)$. This completes the description of how to sample (x,b) from the example distribution $\mathcal{E}$. In fact, this is a complete description of the reduction of Theorem I.1. The difficulty lies in the proof of the correctness of the reduction, which we sketch below.

It is not hard to see the completeness of the reduction. Assume that there exists a satisfying assingment α of φ such that $\sum_{i=1}^n \alpha(i) \le \theta$. Let T denote the set of indices $i \in [n]$ such that $\alpha(i) = 1$. The fact that the set T is authorized

[12] One property of a pseudorandom generator that is required for our reduction is *seed-extending* [62], i.e., a pseudorandom generator is secure even if the seed is included in the output. This property is used in the proof of the completeness of our reduction.

motivates us to define the following program M: The program M takes $\{f_i \mid i \in T\}$ as hard-wired input. Given an input $x = (\xi_1, \ldots, \xi_n)$, the program M lets $(z_i, \eta_i) := \xi_i$, defines $s_i := (f_i \cdot z_i) \oplus \eta_i$, reconstructs the secret b using the reconstruction procedure $\mathrm{Rec}(\varphi, \text{-})$ for the shares $\{s_i \mid i \in T\}$, and outputs $b \in \{0,1\}$. Since T is authorized, it is guaranteed that the secret $b \in \{0,1\}$ can be reconstructed in this way. The size of M is approximately bounded by $\sum_{i \in T} |f_i| = |T| \cdot \lambda \leq \theta\lambda$.

To prove the soundness of the reduction, we first clarify the condition under which the randomized reduction is successful. The condition is that $\mathrm{K}(f_T) \gtrsim |T| \cdot \lambda$ for every T, which can be proved to happen with high probability over the random choice of $f_1, \ldots, f_n$ by a simple counting argument. (Throughout this section, an approximate inequality $a \lesssim b$ can be understood as $a \leq (1 + o(1)) \cdot b$, where $o(1)$ approaches 0 as the parameter λ increases.) Now, assuming that there exists no authorized set T of size θ, we claim that no program M of size $\theta\lambda/2$ can output b on input x with high probability over $(x, b) \sim \mathcal{E}$. The key technical lemma, which we call an *algorithmic information extraction lemma*, is informally stated as follows.

Lemma II.1 (informal)**.** *Let $f_1, \ldots, f_n \in \{0,1\}^\lambda$ be strings such that $\mathrm{K}(f_T) \gtrsim |T| \cdot \lambda$ for every $T \subseteq [n]$, where λ is sufficiently large. Let M be a program of size $|M|$. Then, there exists a set $B \subseteq [n]$ such that $|B| \lesssim |M|/\lambda$ and*

$$\begin{aligned}&\Pr\bigl[M(X_B, U_{[n]\setminus B}) = 1\bigr]\\ \approx\ &\Pr\bigl[M(X_B, X_{[n]\setminus B}) = 1\bigr],\end{aligned}$$

where X_i is the random variable identical to $\mathrm{DP}_k(f_i; z_i)$ for a random choice of $z_i \sim \{0,1\}^{nk}$ and U_i is identical to the uniform distribution over $\{0,1\}^{\lambda k + k}$.

This lemma demonstrates that a small set B can be "extracted" from a program M such that $\mathrm{DP}_k(f_i; \text{-})$ appears to be pseudorandom against M for every $i \in [n] \setminus B$. In particular, for every $i \in [n] \setminus B$, the program M cannot read s_i from $\xi_i = \mathrm{DP}_k(f_i; z_i) \oplus s'_i$, where s'_i denotes $(0^{\lambda k}, s_i)$. Moreover, we have $|B| \lesssim |M|/\lambda \leq \theta/2$, which implies that B is not authorized. By the privacy of the secret sharing scheme, the shares s_B and the secret $b \sim \{0,1\}$ are statistically independent. Thus, we obtain

$$\begin{aligned}&\Pr\Bigl[M(X_B \oplus s'_B, U_{[n]\setminus B} \oplus s'_{[n]\setminus B}) = b\Bigr]\\ =\ &\Pr\bigl[M(X_B \oplus s'_B, U_{[n]\setminus B}) = b\bigr]\\ =\ &\frac{1}{2}.\end{aligned}$$

It follows from the algorithmic information extraction lemma that[13]

$$\begin{aligned}&\Pr_{(x,b)\sim\mathcal{E}}[M(x) = b]\\ =\ &\Pr\Bigl[M(X_B \oplus s'_B, X_{[n]\setminus B} \oplus s'_{[n]\setminus B}) = b\Bigr]\\ \approx\ &\Pr\Bigl[M(X_B \oplus s'_B, U_{[n]\setminus B} \oplus s'_{[n]\setminus B}) = b\Bigr]\\ =\ &\frac{1}{2},\end{aligned}$$

as desired.

It remains to prove the algorithmic information extraction lemma. We formalize the notion that M "knows" f_i based on conditional Kolmogorov complexity. We say that M *knows* f_i if

$$\mathrm{K}(f_i \mid M) \leq \theta$$

for a threshold $\theta := 2nk$. The intuition behind this definition is that if M contains a lot of information about f_i, it should be possible to describe f_i using a few bits of information.[14] Now, we define B to be the set of indices $i \in [n]$ such that M knows f_i. It can be shown that $|B|$ is small:

$$\begin{aligned}&|B| \cdot \lambda - |M|\\ \lesssim\ &\mathrm{K}(f_B) - |M|\\ \lesssim\ &\mathrm{K}(f_B \mid M)\\ \lesssim\ &\sum_{i \in B} \mathrm{K}(f_i \mid M)\\ \leq\ &|B| \cdot \theta, \qquad (2)\end{aligned}$$

where the first inequality follows from the assumption, the second inequality holds by the definition of conditional Kolmogorov complexity, the third inequality holds because $f_B = (f_i \mid i \in B)$ can be described by programs describing f_i for all $i \in B$, and the final inequality holds by the definition of B. If we choose a sufficiently large λ such that $\theta = o(\lambda)$ (e.g., $\lambda := \theta^2$), we obtain

$$|B| \cdot \lambda \cdot (1 - o(1)) \lesssim |M|,$$

which implies $|B| \lesssim |M|/\lambda$, as desired. We now prove that M cannot distinguish $X_i = \mathrm{DP}_k(f_i; z_i)$ from U_i for every $i \in [n] \setminus B$. This can be proved using a standard (but careful) hybrid argument. To illustrate the idea, let us assume that $B = \{2\}$ and $n = 2$. In this case, we must prove that

$$\begin{aligned}&\Pr_{z_1, z_2}[M(\mathrm{DP}_k(f_1; z_1), \mathrm{DP}_k(f_2; z_2)) = 1]\\ \approx\ &\Pr_{U_1, z_2}[M(U_1, \mathrm{DP}_k(f_2; z_2)) = 1].\end{aligned}$$

Assume, towards a contradiction, that this approximate equality fails. Our goal is to prove $\mathrm{K}(f_1 \mid M) \leq \theta$, which

[13]The argument is not precise, especially because we need to argue that the set B does not depend on the secret b and the shares $s_{[n]}$. To address this issue, the algorithmic information extraction lemma will be stated for a program that takes an advice string α, which includes the information about the secret and the shares; see the full version for details.

[14]In terms of the mutual information $\mathrm{I}(f_i : M) := \mathrm{K}(f_i) - \mathrm{K}(f_i \mid M)$ between f_i and M, the condition that M knows f_i means that $\mathrm{I}(f_i : M) \approx \mathrm{K}(f_i)$.

contradicts that M does not know f_1 (i.e., $1 \notin B$). We use the reconstruction property of DP_k. However, it is problematic to apply the property of Eq. (1) naively: Applying Eq. (1), we would get $\mathrm{K}^D(f_1) \lesssim k$, where D is a function that outputs $M(w, \mathrm{DP}_k(f_2; z_2))$ on input w and random bits z_2. Since D can be simulated using M and f_2, we obtain

$$\mathrm{K}(f_1 \mid M, f_2) \lesssim \mathrm{K}^D(f_1) \lesssim k.$$

We need to remove f_2 from the condition of $\mathrm{K}(f_1 \mid M, f_2)$. To this end, we observe that D is, in fact, a randomized algorithm that takes an a-bit advice string as input, which depends on randomness for some small $a \in \mathbb{N}$ (independent of λ). The notion of such advice was introduced by Trevisan and Vadhan [60]. Specifically, D takes w and random bits z_2 as inputs and outputs $M(w, \mathrm{DP}_k(f_2; z_2)) = M(w, (z_2, f_2 \cdot z_2))$—this function can be computed using the k-bit advice string $f_2 \cdot z_2 \in \{0,1\}^k$. Thus, we do not require the full description of $f_2 \in \{0,1\}^\lambda$ to compute D. Using a pseudorandom generator (computable by an exhaustive search), we observe that the reconstruction property of Eq. (1) actually yields

$$\mathrm{K}(f_1 \mid M) \lesssim k + a$$

for any randomized algorithm M that takes a bits of Trevisan–Vadhan advice. In our case, we have $a \leq (n-1) \cdot k$; thus, we can prove that $\mathrm{K}(f_1 \mid M) \lesssim nk \leq \theta/2$. This implies that M knows f_1, which contradicts $1 \notin B$.

B. NP-Hardness of Partial Variants of Meta-computatioal Problems

We now update the aforementioned proof to a proof of Theorem I.2, i.e., the NP-hardness of the partial function variants of meta-computational problems, such as NC^1-$\mathrm{MCSP}^*, \mathrm{MCSP}^*, \mathrm{MKTP}^*$, and MINKT^*. Specifically, our goal is to reduce an instance of an NP-complete problem to an example distribution $\mathcal{E}$ such that $\mathrm{supp}(\mathcal{E}) \subseteq \{0,1\}^{O(\log n)} \times \{0,1\}$, where n denotes the length of an instance of the original NP-complete problem. Enumerating all the elements in the support $\mathrm{supp}(\mathcal{E})$ of the distribution $\mathcal{E}$, we obtain a partial function $f\colon \{0,1\}^{O(\log n)} \to \{0,1,*\}$, which yields a reduction to partial variants of meta-computational problems. Note that the reduction of Theorem I.1 produces a partial function $f\colon \{0,1\}^{n^{O(1)}} \to \{0,1,*\}$. We need to reduce the input length of f exponentially.

We begin by inspecting the NP-hardness reduction to the MMSA problem. Dinur and Safra [58] presented a generic approximation-preserving reduction from any MaxCSP (Constraint Satisfaction Problem) to MMSA. An instance of MaxCSP consists of a set $\Psi = \{C_1, \dots, C_m\}$ of constraints over n variables taking values in an alphabet Σ. Each constraint C_j depends on D variables, where $D = O(1)$. The reduction of [58] reduces such an instance to a depth-3 monotone formula φ such that $\varphi = \bigwedge_{j \in [m]} \varphi_j$, where each depth-2 subformula φ_j "verifies" that the constraint C_j is satisfied. Since each constraint C_j depends on a constant number D of variables, each subformula φ_j also depends on a small number of variables. The fundamental idea behind reducing the input length in our reduction is to exploit this "locality" of φ_j.

Now, we introduce the Collective Minimum (Weight) Monotone Satisfying Assignment (CMMSA) problem, which generalizes the MMSA problem. An instance of CMMSA consists of a collection $\Phi = \{\varphi_1, \dots, \varphi_m\}$ of monotone DNF formulas over n variables, a weight function $w\colon [n] \to \mathbb{N}$, and a threshold θ. The task is to distinguish (1) the YES case, in which there exists an assignment $\alpha\colon [n] \to \{0,1\}$ such that the weight $\sum_{i=1}^n \alpha(i) \cdot w(i)$ is at most θ and $\Pr_{\varphi \sim \Phi}[\varphi(\alpha) = 1] = 1$ from (2) the NO case, in which for every assignment of weight $\Delta^\epsilon \cdot \theta$, $\Pr_{\varphi \sim \Phi}[\varphi(\alpha) = 1] < \Delta^{-\epsilon}$, where $\epsilon > 0$ is a constant and Δ is a parameter such that the number of literals in the DNF formula φ_j is at most Δ for every $j \in [m]$. Using a PCP system [65] developed in the research line on the Sliding Scale Conjecture [66], we observe that CMMSA is NP-hard for some constant $\epsilon > 0$ and for a parameter $\Delta := (\log n)^{1/2}$ by applying the reduction of [58]. We note that the weight function w in CMMSA is used for a technical reason; thus, for simplicity, we assume that $w \equiv 1$ in this proof overview.

Now, our goal is to reduce CMMSA to MINKT^*. To this end, we exploit the "locality" of CMMSA: Each formula φ_j in Φ depends on at most Δ variables out of n variables. The reduction is similar to the reduction discussed in Theorem I.1. For random strings $f_1, \dots, f_n \sim \{0,1\}^\lambda$, we define a distribution $\mathcal{E} = \mathcal{E}(f_1, \dots, f_n)$ as follows: A secret $b \sim \{0,1\}$ and $\varphi_j \sim \Phi$ are chosen randomly. Using the secret sharing scheme for φ_j, the secret b is shared among at most Δ parties $v_1, \dots, v_\Delta \in [n]$. Let $(s_1, \dots, s_\Delta) = \mathrm{Share}(\varphi_j, b)$ be the shares distributed to the parties $v_1, \dots, v_\Delta$, respectively. We define an input x to be $(j, \mathrm{DP}_k(f_{v_1}; z_1) \oplus s'_1, \dots, \mathrm{DP}_k(f_{v_\Delta}; z_\Delta) \oplus s'_\Delta)$, where $s'_i := (0^{\lambda k}, s_i)$. The output of the distribution $\mathcal{E}$ is defined to be (x, b). Just as in the proof of Theorem I.1, it is possible to prove the following statements: If there is an assignment $\alpha\colon [n] \to \{0,1\}$ that satisfies all the formulas φ_j in Φ, then there exists a program of size $\lesssim \sum_{i=1}^n \alpha(i) \cdot \lambda$ (that takes $\{f_i \mid \alpha(i) = 1\}$ as hard-wired input) that computes b on input x for every $(x, b) \in \mathrm{supp}(\mathcal{E})$. If there is no assignment of small weight that satisfies a $\Delta^{-\epsilon}$-fraction of Φ, then there is no small program that computes b on input x with probability $1/2 + 2\Delta^{-\epsilon}$ over a random choice of $(x, b) \sim \mathcal{E}$.

The only issue is that the length of the seeds $z_1, \dots, z_\Delta$ is large, which prevents us from obtaining $|x| = O(\log n)$. The length of each z_i is λk, which is always greater than the length of f_i. This is because DP_k is instantiated with Hadamard encoding, which maps a string f_i of length λ to a string of length 2^λ. Instead, if we instantiate a k-wise direct product generator with an error correcting code that maps a string of length λ to a string of length $\lambda^{O(1)}$, the length of seed z_i can be reduced to $k \cdot O(\log \lambda)$ (as in [63]). This optimization is still not sufficient to guarantee $|x| = O(\log n)$, as the seeds $z_1, \dots, z_\Delta$ are independent. To further reduce the seed length, we use the Nisan–Wigderson pseudorandom generator construction [61]. Nisan and Wigderson [61] developed a method of generating correlated seeds $z_{S_1}, \dots, z_{S_\Delta}$ from a short seed z and used it

to construct a pseudorandom generator with short seed length based on an average-case hard function. Using the Nisan–Wigderson pseudorandom generator construction NW, we construct the input x as follows: $(j, z, \mathrm{NW}(\mathrm{Enc}(f_{v_1}); z_{S_1}) \oplus s_1, \ldots, \mathrm{NW}(\mathrm{Enc}(f_{v_\Delta}); z_{S_\Delta}) \oplus s_\Delta)$, where $\mathrm{Enc}\colon \{0,1\}^\lambda \to \{0,1\}^{\lambda^{O(1)}}$ is an error-correcting code. It can be proved that the input length $|x|$ is $O(\log n)$. This completes the overview of the proof of NP-hardness of MINKT^*.

Proving the NP-hardness of MCSP^* requires additional ideas. We need to argue that there exists a small circuit C that takes $x = (j, z, \mathrm{NW}(\mathrm{Enc}(f_{v_1}); z_{S_1}) \oplus s_1, \ldots, \mathrm{NW}(\mathrm{Enc}(f_{v_\Delta}); z_{S_\Delta}) \oplus s_\Delta)$ as input, reads shares from the input, and reconstructs the secret. This can be achieved using a $\mathrm{poly}(\lambda)$-time algorithm by computing $\mathrm{Enc}(f_{v_i})$ from a hard-wired input f_{v_i}. However, this is not necessarily possible for a circuit. The string $f_{v_i} \in \{0,1\}^\lambda$ must be hard-wired in a circuit using $O(\lambda/\log\lambda)$ gates. Then, the bits of $\mathrm{Enc}(f_{v_i})$ specified by z_{S_i} must be computed from such embedded gates using $O(\lambda/\log\lambda)$ gates. This does not seem to be possible, as each bit of $\mathrm{Enc}(f_{v_i})$ depends on almost all bits of f_{v_i} in order for Enc to be a good error-correcting code; thus, simply reading such bits amounts to $O(\lambda)$ gates, which is too large. One may be tempted to let Enc be the identity function to avoid such a computation, but this does not work because it significantly weakens the reconstruction property of NW. In general, there are two conflicting requirements on Enc(-):

1) It must be ensured that each bit of $\mathrm{Enc}(f)$ can be computed by reading a small number of bits from f.
2) Enc(-) must be a good list-decodable error-correcting code, i.e., given a string that agrees with $\mathrm{Enc}(f)$ on a $(1/2+\epsilon)$-fraction of bits for a small parameter $\epsilon > 0$, f must be identified based on a small number of advice bits.

In other words, a "locally-encodable and list-decodable" error-correcting code is required. We observe that the derandomized hardness amplification theorem of Impagliazzo and Wigderson [67] can be regarded as such a code: They showed that any function $f\colon \{0,1\}^{\log\lambda} \to \{0,1\}$ can be converted into a function $\widehat{f}\colon \{0,1\}^{O(\log\lambda)} \to \{0,1\}$ such that given a small circuit that computes $\widehat{f}$ on a $(1/2+\epsilon)$-fraction of inputs, there exists a small circuit that computes f on almost all inputs. Letting $\mathrm{Enc}(f)$ be $\widehat{f}$, we observe that the two properties are satisfied. This enables us to prove the NP-hardness of MKTP^*.

Proving NP-hardness of MCSP^* requires an additional (and the last) ingredient. In this case, we need to ensure that $\widehat{f}$ can be computed by a circuit of size $O(\lambda/\log\lambda)$ for a random function $f\colon \{0,1\}^{\log\lambda} \to \{0,1\}$. To this end, we employ the theorem of Uhlig [68], [69], which states that for any function $f\colon \{0,1\}^{\log\lambda} \to \{0,1\}$, the r-wise direct product $f^r\colon \left(\{0,1\}^{\log\lambda}\right)^r \to \{0,1\}^r$ can be computed by a circuit of size $O(\lambda/\log\lambda)$ for $r = \lambda^{o(1/\log\log\lambda)}$. Since $\widehat{f}$ can be locally computed using the output of f on at most r inputs, we obtain a circuit of size $O(\lambda/\log\lambda)$ that computes $\widehat{f}$.

III. Open Problems

We expect the results proved in this paper to motivate several fruitful research directions, as described below.

MCSP: A major open problem is to prove NP-hardness of MCSP. Following the two-step paradigm of proving the NP-hardness of variants of MCSP, it suffices to present a reduction from MCSP^* to MCSP. We expect that this task now becomes much easier than it was previously, because the reduction of Theorem I.2 creates a large gap between the YES instances and the NO instances. Thus, the reduction of the approximate and partial variant of MCSP of Theorem I.2 to MCSP is a promising research problem.

It is also interesting to see if MCSP^* for AC^0 circuits is NP-hard. In general, our results do not necessarily prove NP-hardness of MCSP^* for circuit classes with unbounded fan-in gates. The reason is that an $O(1)$-fan-in circuit of size s can be encoded as a binary string of $O(s\log s)$, whereas the binary encoding of an unbounded-fan-in circuit of size s can be as large as s^2. Generalization of our results to unbounded-fan-in circuits is an interesting direction of research.

Heuristica: Another major open question is to improve the inapproximability factor n^ϵ of Theorem I.1 to $1.01n$. This has a significant implication for Impagliazzo's five worlds—in particular, it excludes Heuristica. We expect that this requires a "non-black-box" reduction technique that exploits the efficiency of a program in the NO case. Indeed, the literature on the limits of black-box reductions [70]–[74] suggests that no nonadaptive randomized reduction can reduce NP to a black-box oracle that solves NP on average. Our proof techniques are algorithmic information theoretic and do not exploit the efficiency of programs in the NO case. Thus, we expect that our reduction techniques are subject to such a barrier.

We note, however, that excluding Heuristica is achievable in principle by combining current proof techniques: Hirahara [33], [34] developed *non-black-box* but *relativizing* reduction techniques (which are, thus, subject to the relativization barrier of Impagliazzo [75]; see also [16]); we developed *black-box* but *non-relativizing* reduction techniques. What remains is to combine these proof techniques to overcome black-box and relativization barriers *simultaneously*. This can be achieved by reducing the approximate problem of MINKT^* (Theorem I.2) to an approximate problem $\mathrm{GapMINKT}$ of MINKT, which is known to be in P in Heuristica [33], [64].

A less challenging but intriguing open problem is to improve the inapproximability factor n^ϵ to $n^{1-o(1)}$, which could be achievable by combining sophisticated PCP machinery with the proof techniques developed in this paper.

Computational Learning Theory: Our proof techniques would have applications to the problem of learning parities, which is one of central research topics in computational learning theory (see, e.g., [76], [77] and references therein). Learning k-sparse parities by k-juntas is known to be W[1]-hard [77]. We strongly conjecture that Theorem I.1 can be improved to NP-hardness of learning s-sparse parities by programs of size $s\cdot n^{1/(\log\log n)^{O(1)}}$, which would significantly

improve the result of [77]. In fact, the only missing piece is the following question regarding PCPs: Can the PCP system of Dinur, Harsha, and Kindler [59] be made smooth in the sense that every coordinate of a proof is queried with equal probability? We expect that the proof techniques developed in, e.g., [78] that make PCP systems smooth might be used to resolve this question affirmatively.

More broadly, our results open up the possibility of proving the NP-hardness of PAC learning of various concept classes by small programs. There are many non-proper PAC learners in the literature (e.g., [32], [76], [79], to name a few). Can we give evidence that such PAC learners cannot produce small programs, using our proof techniques of showing NP-hardness?

Acknowledgment

I thank Leonid Levin for clarifying that the second problem in [28] is DNF-MCSP* and anonymous referees for helpful comments.

References

[1] S. Hirahara, "NP-Hardness of Learning Programs and Partial MCSP," *Electronic Colloquium on Computational Complexity (ECCC)*, no. 119, 2022.

[2] K.-I. Ko, "On the Complexity of Learning Minimum Time-Bounded Turing Machines," *SIAM J. Comput.*, vol. 20, no. 5, pp. 962–986, 1991. DOI: 10.1137/0220059.

[3] V. Kabanets and J.-y. Cai, "Circuit minimization problem," in *Proceedings of the Symposium on Theory of Computing (STOC)*, 2000, pp. 73–79. DOI: 10.1145/335305.335314.

[4] L. G. Valiant, "A Theory of the Learnable," *Commun. ACM*, vol. 27, no. 11, pp. 1134–1142, 1984. DOI: 10.1145/1968.1972.

[5] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth, "Occam's Razor," *Inf. Process. Lett.*, vol. 24, no. 6, pp. 377–380, 1987. DOI: 10.1016/0020-0190(87)90114-1.

[6] R. A. Board and L. Pitt, "On the Necessity of Occam Algorithms," *Theor. Comput. Sci.*, vol. 100, no. 1, pp. 157–184, 1992. DOI: 10.1016/0304-3975(92)90367-O.

[7] R. E. Schapire, "The Strength of Weak Learnability," *Mach. Learn.*, vol. 5, pp. 197–227, 1990. DOI: 10.1007/BF00116037.

[8] L. Pitt and L. G. Valiant, "Computational limitations on learning from examples," *J. ACM*, vol. 35, no. 4, pp. 965–984, 1988. DOI: 10.1145/48014.63140.

[9] M. Alekhnovich, M. Braverman, V. Feldman, A. R. Klivans, and T. Pitassi, "The complexity of properly learning simple concept classes," *J. Comput. Syst. Sci.*, vol. 74, no. 1, pp. 16–34, 2008. DOI: 10.1016/j.jcss.2007.04.011.

[10] A. Daniely and S. Shalev-Shwartz, "Complexity Theoretic Limitations on Learning DNF's," in *Proceedings of the Conference on Learning Theory (COLT)*, 2016, pp. 815–830.

[11] S. P. Vadhan, "On Learning vs. Refutation," in *Proceedings of the Conference on Learning Theory (COLT)*, 2017, pp. 1835–1848.

[12] A. Daniely and G. Vardi, "From Local Pseudorandom Generators to Hardness of Learning," in *Proceedings of the Conference on Learning Theory (COLT)*, 2021, pp. 1358–1394.

[13] A. Daniely, N. Linial, and S. Shalev-Shwartz, "From average case complexity to improper learning complexity," in *Proceedings of the Symposium on Theory of Computing (STOC)*, 2014, pp. 441–448. DOI: 10.1145/2591796.2591820.

[14] B. Applebaum, B. Barak, and D. Xiao, "On Basing Lower-Bounds for Learning on Worst-Case Assumptions," in *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, 2008, pp. 211–220. DOI: 10.1109/FOCS.2008.35.

[15] R. Impagliazzo, "A Personal View of Average-Case Complexity," in *Proceedings of the Structure in Complexity Theory Conference*, 1995, pp. 134–147. DOI: 10.1109/SCT.1995.514853.

[16] S. Hirahara and M. Nanashima, "On Worst-Case Learning in Relativized Heuristica," in *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, 2021, pp. 751–758. DOI: 10.1109/FOCS52979.2021.00078.

[17] A. N. Kolmogorov, "Three approaches to the quantitative definition of information," *Problems of information transmission*, vol. 1, no. 1, pp. 1–7, 1965.

[18] T. P. Baker, J. Gill, and R. Solovay, "Relativizations of the P =? NP Question," *SIAM J. Comput.*, vol. 4, no. 4, pp. 431–442, 1975. DOI: 10.1137/0204037.

[19] H. Buhrman, L. Fortnow, and T. Thierauf, "Nonrelativizing Separations," in *Proceedings of the Conference on Computational Complexity (CCC)*, 1998, pp. 8–12. DOI: 10.1109/CCC.1998.694585.

[20] S. Aaronson and A. Wigderson, "Algebrization: A New Barrier in Complexity Theory," *TOCT*, vol. 1, no. 1, 2:1–2:54, 2009. DOI: 10.1145/1490270.1490272.

[21] S. Khot and R. Saket, "Hardness of Minimizing and Learning DNF Expressions," in *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, 2008, pp. 231–240. DOI: 10.1109/FOCS.2008.37.

[22] P. Gopalan, S. Khot, and R. Saket, "Hardness of Reconstructing Multivariate Polynomials over Finite Fields," *SIAM J. Comput.*, vol. 39, no. 6, pp. 2598–2621, 2010. DOI: 10.1137/070705258.

[23] S. Khot and R. Saket, "On the hardness of learning intersections of two halfspaces," *J. Comput. Syst. Sci.*, vol. 77, no. 1, pp. 129–141, 2011. DOI: 10.1016/j.jcss.2010.06.010.

[24] V. Feldman, P. Gopalan, S. Khot, and A. K. Ponnuswami, "On Agnostic Learning of Parities, Monomials, and Halfspaces," *SIAM J. Comput.*, vol. 39, no. 2, pp. 606–645, 2009. DOI: 10.1137/070684914.

[25] V. Feldman, V. Guruswami, P. Raghavendra, and Y. Wu, "Agnostic Learning of Monomials by Halfspaces Is Hard," *SIAM J. Comput.*, vol. 41, no. 6, pp. 1558–1590, 2012. DOI: 10.1137/120865094.

[26] B. A. Trakhtenbrot, "A Survey of Russian Approaches to Perebor (Brute-Force Searches) Algorithms," *IEEE Annals of the History of Computing*, vol. 6, no. 4, pp. 384–400, 1984. DOI: 10.1109/MAHC.1984.10036.

[27] E. Allender, M. Koucký, D. Ronneburger, and S. Roy, "The pervasive reach of resource-bounded Kolmogorov complexity in computational complexity theory," *J. Comput. Syst. Sci.*, vol. 77, no. 1, pp. 14–40, 2011. DOI: 10.1016/j.jcss.2010.06.004.

[28] L. A. Levin, "Universal sequential search problems," *Problemy Peredachi Informatsii*, vol. 9, no. 3, pp. 115–116, 1973.

[29] E. Allender, H. Buhrman, M. Koucký, D. van Melkebeek, and D. Ronneburger, "Power from Random Strings," *SIAM J. Comput.*, vol. 35, no. 6, pp. 1467–1493, 2006. DOI: 10.1137/050628994.

[30] S. Hirahara and R. Santhanam, "On the Average-Case Complexity of MCSP and Its Variants," in *Proceedings of the Computational Complexity Conference (CCC)*, 2017, 7:1–7:20. DOI: 10.4230/LIPIcs.CCC.2017.7.

[31] E. Allender and S. Hirahara, "New Insights on the (Non-)Hardness of Circuit Minimization and Related Problems," *TOCT*, vol. 11, no. 4, 27:1–27:27, 2019. DOI: 10.1145/3349616.

[32] M. L. Carmosino, R. Impagliazzo, V. Kabanets, and A. Kolokolova, "Learning Algorithms from Natural Proofs," in *Proceedings of the Conference on Computational Complexity (CCC)*, 2016, 10:1–10:24. DOI: 10.4230/LIPIcs.CCC.2016.10.

[33] S. Hirahara, "Non-Black-Box Worst-Case to Average-Case Reductions within NP," in *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, 2018, pp. 247–258. DOI: 10.1109/FOCS.2018.00032.

[34] ——, "Average-case hardness of NP from exponential worst-case hardness assumptions," in *Proceedings of the Symposium on Theory of Computing (STOC)*, 2021, pp. 292–302. DOI: 10.1145/3406325.3451065.

[35] R. Impagliazzo and L. A. Levin, "No Better Ways to Generate Hard NP Instances than Picking Uniformly at Random," in *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, 1990, pp. 812–821. DOI: 10.1109/FSCS.1990.89604.

[36] R. Santhanam, "Pseudorandomness and the Minimum Circuit Size Problem," in *Proceedings of the Innovations in Theoretical Computer Science Conference (ITCS)*, 2020, 68:1–68:26. DOI: 10.4230/LIPIcs.ITCS.2020.68.

[37] Y. Liu and R. Pass, "On One-way Functions and Kolmogorov Complexity," in *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, 2020, pp. 1243–1254.

[38] I. C. Oliveira and R. Santhanam, "Hardness Magnification for Natural Problems," in *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, 2018, pp. 65–76.

[39] L. Chen, S. Hirahara, I. C. Oliveira, J. Pich, N. Rajgopal, and R. Santhanam, "Beyond Natural Proofs: Hardness Magnification and Locality," in *Proceedings of the Innovations in Theoretical Computer Science Conference (ITCS)*, 2020, 70:1–70:48. DOI: 10.4230/LIPIcs.ITCS.2020.70.

[40] E. Allender, "Vaughan Jones, Kolmogorov Complexity, and the new complexity landscape around circuit minimization," *New Zealand journal of mathematics*, vol. 52, pp. 585–604, 2021.

[41] W. J. Masek, "Some NP-complete set covering problems," *Unpublished manuscript*, 1979.

[42] E. Allender, L. Hellerstein, P. McCabe, T. Pitassi, and M. E. Saks, "Minimizing Disjunctive Normal Form Formulas and AC^0 Circuits Given a Truth Table," *SIAM J. Comput.*, vol. 38, no. 1, pp. 63–84, 2008. DOI: 10.1137/060664537.

[43] S. Hirahara, I. C. Oliveira, and R. Santhanam, "NP-hardness of Minimum Circuit Size Problem for OR-AND-MOD Circuits," in *Proceedings of the Computational Complexity Conference (CCC)*, 2018, 5:1–5:31. DOI: 10.4230/LIPIcs.CCC.2018.5.

[44] R. Ilango, "Constant Depth Formula and Partial Function Versions of MCSP are Hard," in *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, 2020, pp. 424–433.

[45] ——, "The Minimum Formula Size Problem is (ETH) Hard," in *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, 2021, pp. 427–432. DOI: 10.1109/FOCS52979.2021.00050.

[46] R. Impagliazzo, R. Paturi, and F. Zane, "Which Problems Have Strongly Exponential Complexity?" *J. Comput. Syst. Sci.*, vol. 63, no. 4, pp. 512–530, 2001. DOI: 10.1006/jcss.2001.1774.

[47] S. A. Cook, "The Complexity of Theorem-Proving Procedures," in *Proceedings of the Symposium on Theory of Computing (STOC)*, 1971, pp. 151–158. DOI: 10.1145/800157.805047.

[48] C. D. Murray and R. R. Williams, "On the (Non) NP-Hardness of Computing Circuit Complexity," *Theory of Computing*, vol. 13, no. 1, pp. 1–22, 2017. DOI: 10.4086/toc.2017.v013a004.

[49] J. M. Hitchcock and A. Pavan, "On the NP-Completeness of the Minimum Circuit Size Problem," in *Proceedings of the Conference on Foundation of Software Technology and Theoretical Computer Science (FSTTCS)*, 2015, pp. 236–245. DOI: 10.4230/LIPIcs.FSTTCS.2015.236.

[50] S. Hirahara and O. Watanabe, "Limits of Minimum Circuit Size Problem as Oracle," in *Proceedings of the Conference on Computational Complexity (CCC)*, 2016, 18:1–18:20. DOI: 10.4230/LIPIcs.CCC.2016.18.

[51] M. Saks and R. Santhanam, "Circuit Lower Bounds from NP-Hardness of MCSP Under Turing Reductions," in *Proceedings of the Computational Complexity Conference (CCC)*, 2020, 26:1–26:13. DOI: 10.4230/LIPIcs.CCC.2020.26.

[52] R. Ilango, "Approaching MCSP from Above and Below: Hardness for a Conditional Variant and $\mathsf{AC}^0[p]$," in *Proceedings of the Innovations in Theoretical Computer Science Conference (ITCS)*, 2020, 34:1–34:26. DOI: 10.4230/LIPIcs.ITCS.2020.34.

[53] R. Ilango, B. Loff, and I. C. Oliveira, "NP-Hardness of Circuit Minimization for Multi-Output Functions," in *Proceedings of the Computational Complexity Conference (CCC)*, 2020, 22:1–22:36. DOI: 10.4230/LIPIcs.CCC.2020.22.

[54] E. Allender, M. Cheraghchi, D. Myrisiotis, H. Tirumala, and I. Volkovich, "One-Way Functions and a Conditional Variant of MKTP," in *Proceedings of the Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, 2021, 7:1–7:19. DOI: 10.4230/LIPIcs.FSTTCS.2021.7.

[55] Y. Liu and R. Pass, "On One-Way Functions from NP-Complete Problems," in *Proceedings of the Computational Complexity Conference (CCC)*, 2022, 36:1–36:24. DOI: 10.4230/LIPIcs.CCC.2022.36.

[56] S. Hirahara, "Symmetry of Information from Meta-Complexity," in *Proceedings of the Computational Complexity Conference (CCC)*, 2022, 26:1–26:41. DOI: 10.4230/LIPIcs.CCC.2022.26.

[57] M. Alekhnovich, S. R. Buss, S. Moran, and T. Pitassi, "Minimum Propositional Proof Length Is NP-Hard to Linearly Approximate," *J. Symb. Log.*, vol. 66, no. 1, pp. 171–191, 2001. DOI: 10.2307/2694916.

[58] I. Dinur and S. Safra, "On the hardness of approximating label-cover," *Inf. Process. Lett.*, vol. 89, no. 5, pp. 247–254, 2004. DOI: 10.1016/j.ipl.2003.11.007.

[59] I. Dinur, P. Harsha, and G. Kindler, "Polynomially Low Error PCPs with polyloglog n Queries via Modular Composition," in *Proceedings of the Symposium on Theory of Computing (STOC)*, 2015, pp. 267–276. DOI: 10.1145/2746539.2746630.

[60] L. Trevisan and S. P. Vadhan, "Pseudorandomness and Average-Case Complexity Via Uniform Reductions," *Computational Complexity*, vol. 16, no. 4, pp. 331–364, 2007. DOI: 10.1007/s00037-007-0233-x.

[61] N. Nisan and A. Wigderson, "Hardness vs Randomness," *J. Comput. Syst. Sci.*, vol. 49, no. 2, pp. 149–167, 1994. DOI: 10.1016/S0022-0000(05)80043-1.

[62] J. Kinne, D. van Melkebeek, and R. Shaltiel, "Pseudorandom Generators, Typically-Correct Derandomization, and Circuit Lower Bounds," *Computational Complexity*, vol. 21, no. 1, pp. 3–61, 2012. DOI: 10.1007/s00037-011-0019-z.

[63] S. Hirahara, "Unexpected hardness results for Kolmogorov complexity under uniform reductions," in *Proceedings of the Symposium on Theory of Computing (STOC)*, 2020, pp. 1038–1051. DOI: 10.1145/3357713.3384251.

[64] ——, "Non-Disjoint Promise Problems from Meta-Computational View of Pseudorandom Generator Constructions," in *Proceedings of the Computational Complexity Conference (CCC)*, 2020, 20:1–20:47. DOI: 10.4230/LIPIcs.CCC.2020.20.

[65] I. Dinur, E. Fischer, G. Kindler, R. Raz, and S. Safra, "PCP Characterizations of NP: Toward a Polynomially-Small Error-Probability," *Comput. Complex.*, vol. 20, no. 3, pp. 413–504, 2011. DOI: 10.1007/s00037-011-0014-4.

[66] M. Bellare, S. Goldwasser, C. Lund, and A. Russell, "Efficient probabilistic checkable proofs and applications to approximation," in *Proceedings of the Symposium on Theory of Computing (STOC)*, 1994, p. 820. DOI: 10.1145/195058.195467.

[67] R. Impagliazzo and A. Wigderson, "P = BPP if E Requires Exponential Circuits: Derandomizing the XOR Lemma," in *Proceedings of the Symposium on the Theory of Computing (STOC)*, 1997, pp. 220–229. DOI: 10.1145/258533.258590.

[68] D. Uhlig, "On the synthesis of self-correcting schemes from functional elements with a small number of reliable elements," *Mathematical Notes of the Academy of Sciences of the USSR*, vol. 15, no. 6, pp. 558–562, 1974.

[69] ——, "Networks Computing Boolean Functions for Multiple Input Values," in *Poceedings of the London Mathematical Society Symposium on Boolean Function Complexity*, London, United Kingdom: Cambridge University Press, 1992, pp. 165–173, ISBN: 0521408261.

[70] J. Feigenbaum and L. Fortnow, "Random-Self-Reducibility of Complete Sets," *SIAM J. Comput.*, vol. 22, no. 5, pp. 994–1005, 1993. DOI: 10.1137/0222061.

[71] A. Bogdanov and L. Trevisan, "On Worst-Case to Average-Case Reductions for NP Problems," *SIAM J. Comput.*, vol. 36, no. 4, pp. 1119–1159, 2006. DOI: 10.1137/S0097539705446974.

[72] A. Akavia, O. Goldreich, S. Goldwasser, and D. Moshkovitz, "On basing one-way functions on NP-hardness," in *Proceedings of the Symposium on Theory of Computing (STOC)*, 2006, pp. 701–710. DOI: 10.1145/1132516.1132614.

[73] A. Bogdanov and C. Brzuska, "On Basing Size-Verifiable One-Way Functions on NP-Hardness," in *Proceedings of the Theory of Cryptography Conference (TCC)*, 2015, pp. 1–6. DOI: 10.1007/978-3-662-46494-6_1.

[74] S. Hirahara and O. Watanabe, "On Nonadaptive Security Reductions of Hitting Set Generators," in *Proceedings of the Approximation, Randomization, and Combinatorial Optimization (APPROX/RANDOM)*, 2020, 15:1–15:14. DOI: 10.4230/LIPIcs.APPROX/RANDOM.2020.15.

[75] R. Impagliazzo, "Relativized Separations of Worst-Case and Average-Case Complexities for NP," in *Proceedings of the Conference on Computational Complexity (CCC)*, 2011, pp. 104–114. DOI: 10.1109/CCC.2011.34.

[76] A. T. Kalai, Y. Mansour, and E. Verbin, "On agnostic boosting and parity learning," in *Proceedings of the Symposium on Theory of Computing (STOC)*, 2008, pp. 629–638. DOI: 10.1145/1374376.1374466.

[77] A. Bhattacharyya, A. Gadekar, S. Ghoshal, and R. Saket, "On the Hardness of Learning Sparse Parities," in *Proceedings of the European Symposium on Algorithms (ESA)*, 2016, 11:1–11:17. DOI: 10.4230/LIPIcs.ESA.2016.11.

[78] A. Bhangale, P. Harsha, O. Paradise, and A. Tal, "Rigid Matrices From Rectangular PCPs," *Electron. Colloquium Comput. Complex.*, p. 75, 2020.

[79] N. Linial, Y. Mansour, and N. Nisan, "Constant Depth Circuits, Fourier Transform, and Learnability," *J. ACM*, vol. 40, no. 3, pp. 607–620, 1993. DOI: 10.1145/174130.174138.