Regularization of Low Error PCPs and an Application to MCSP

Shuichi Hirahara
National Institute of Informatics, Tokyo, Japan

Dana Moshkovitz
Department of Computer Science, University of Texas at Austin, TX, USA

Abstract

In a *regular* PCP the verifier queries each proof symbol in the same number of tests. This number is called the *degree* of the proof, and it is at least $1/(sq)$ where s is the soundness error and q is the number of queries. It is incredibly useful to have regularity and reduced degree in PCP. There is an expander-based transformation by Papadimitriou and Yannakakis that transforms any PCP with a constant number of queries and constant soundness error to a regular PCP with constant degree. There are also transformations for low error projection and unique PCPs. Other PCPs are constructed especially to be regular. In this work we show how to regularize and reduce degree of PCPs with a possibly large number of queries and low soundness error.

As an application, we prove NP-hardness of an unweighted variant of the collective minimum monotone satisfying assignment problem, which was introduced by Hirahara (FOCS'22) to prove NP-hardness of MCSP* (the partial function variant of the Minimum Circuit Size Problem) under randomized reductions. We present a simplified proof and sufficient conditions under which MCSP* is NP-hard under the standard notion of reduction: MCSP* is NP-hard under deterministic polynomial-time many-one reductions if there exists a function in E that satisfies certain direct sum properties.

2012 ACM Subject Classification Theory of computation → Computational complexity and cryptography

Keywords and phrases PCP theorem, regularization, Minimum Circuit Size Problem

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2023.39

Funding *Shuichi Hirahara*: Supported by JST, PRESTO Grant Number JPMJPR2024, Japan.
Dana Moshkovitz: Supported by the National Science Foundation under grants number 2200956 and 2312573.

Acknowledgements We are thankful to Dean Doron for discussions about explicit construction of dispersers.

1 Introduction

1.1 Regularization of Low Error PCPs

In a Probabilistically Checkable Proof (PCP) the verifier uses randomness to pick a small number of queries to its proof. A correct proof is typically accepted, whereas a proof of an incorrect statement is typically rejected. PCPs found many surprising applications over the years in areas like hardness of approximation [14, 15], cryptography [30], complexity theoretic lower bounds [49, 2, 9], quantum computation [25] and metric embeddings [29].

It is often desirable, both for the construction of PCPs and for their applications, that the PCP is *regular*[1], that is, the verifier queries each proof symbol on the same number of tests. This number is called the *degree*. Some PCPs are naturally regular or can be made regular

[1] We remark that regular PCPs were called *smooth* PCPs in a few forks following the definition of smooth locally decodable codes [27]. In PCP the term "smooth PCP" was also used with a completely different meaning [21]. Hence, we will use the term "regularity" and not "smoothness".

with some effort (see, e.g., [38] that constructed a regular PCP from scratch). In contrast, other constructions are inherently non regular, typically because the proof consists of different parts with different roles. Most constructions, and especially algebraic constructions, do not naturally have small degree. We focus on regularization and degree reduction, i.e., on transformations that take *any* PCP verifier and create a similar PCP verifier that is regular and has small degree.

Note that for a regular PCP with degree d, number of queries q and soundness error s we have $s > 1/(dq)$. The reason is that fraction $1/(dq)$ of the randomness strings have disjoint queries. Since each verifier test is satisfiable on its own, it is always possible to satisfy fraction $1/(dq)$ of the tests. Thus, the degree has to be at least $1/(sq)$. As q is typically constant or only slightly super-constant, one desires d that is about $1/s$.

For soundness error s close to 1 and a small number of queries q, Papadimitriou and Yannakakis [37] showed how to regularize the query graph and make the degree constant. This was extended to the case of low soundness error and unique or projection games ($q = 2$) that is especially important for hardness of approximation [28, 34, 11].

We regularize and decrease degree of PCPs of low soundness error s and a general number q of queries:

▶ **Theorem 1** (Regularization and degree reduction). *Let V be a PCP verifier that uses r random bits to make q queries to a proof over alphabet Σ and has completeness error c and soundness error s, where $s \leq \min\{1/(eq), 1/|\Sigma|^a\}$ for e the basis of the natural logarithm and a constant $0 < a \leq 1$. Then V can be efficiently transformed into a new PCP verifier V' whose query graph is bi-regular and where proof symbols have degree $d = q\mathsf{poly}(1/s)$. The verifier V' uses $r + O(\log(1/s))$ random bits to make $\mathsf{poly}(q)$ queries to a proof over alphabet Σ. It has completeness error c and soundness error $O(s^{\Omega(1)})$.*

We can apply our theorem to the PCPs with the lowest error known today:
For a large constant number of queries:

▶ **Corollary 2** (follows from [10, 12] and Theorem 1). *For any $\beta > 0$, for any $s \geq 2^{-(\log n)^{1-\beta}}$, for every $L \in \mathsf{NP}$, there exists a regular PCP verifier V for L that uses r random bits to make q queries to a proof over alphabet Σ and has perfect completeness and soundness error s, where $r = O(\log n)$, $|\Sigma| = \mathsf{poly}(1/s)$, and $q = \mathsf{poly}(1/\beta)$. The degree is $d = \mathsf{poly}(1/s)$.*

For a super-constant number of queries:

▶ **Corollary 3** (follows from [12] and Theorem 1). *For every $L \in \mathsf{NP}$, there exists a regular PCP verifier V for L that uses r random bits to make q queries to a proof over alphabet Σ and has perfect completeness and soundness error s, where $r = O(\log n)$, $s = 1/\mathsf{poly}(n)$, $|\Sigma| = n^{1/(\log\log n)^{O(1)}}$, and $q = (\log\log n)^{O(1)}$. The degree is $d = \mathsf{poly}(n)$.*

1.2 Regularization Technique

Next we describe the Papadimitriou–Yannakakis transformation and subsequent work, as well as explain the difficulty with a larger number of queries and low soundness error. The idea of Papadimitriou and Yannakakis was to replace each proof symbol with several "copies" of the symbol depending on the symbol's original degree. Whenever the verifier wishes to query the symbol, it queries one of the copies instead, so all copies have low degree. The verifier also checks equality between copies by placing an *expander* of low degree on the set of copies, and picking a random edge from the expander. Overall, the verifier makes an original test with probability $\frac{1}{2}$ and an equality check with probability $\frac{1}{2}$.

This construction is only for the case of large soundness error $s > \frac{1}{2}$, because half of the tests can be satisfied even in the soundness case. Why is this construction only for a small number of queries? Because for an original test *all* q copies queried must be consistent with some global proof π. For the Papadimitrious–Yannakakis verifier this happens if the soundness error is sufficiently larger than $1 - \frac{1}{q}$, so we can tolerate a union bound over the q queries.

To allow for lower soundness error one can combine equality tests with original tests, however the natural way to do it requires $\mathsf{poly}(q, 1/s)$ *queries* in order to ensure that except with probability s the labels to every copy queried are consistent with the majority alphabet symbol for the query. Unfortunately, $1/s$ is large when the soundness error s small. In particular, $1/s$ is often much larger than the number of queries q, so one would get a PCP with a much worse number of queries than the number of queries one started with.

An exception is known for "robust" PCPs[2], for which an increase in the number of queries as described above is acceptable, since robust PCPs can always be converted to PCPs with two queries [11]. Indeed, the transformation outlined above, combining the original test with equality tests is equivalent to the regularization and degree reduction of [34]. Alas, the robust PCPs of lowest error [34, 13] have alphabet size $\exp(1/s)$ instead of $\mathsf{poly}(1/s)$. In particular, to keep the alphabet size polynomial in n for a robust PCP the soundness error has to be at least logarithmically small in n.

We show how to regularize and reduce the degree of general, non robust, PCPs while maintaining $\mathsf{poly}(q)$ queries and $O(s^{\Omega(1)})$ soundness error. The degree becomes $\mathsf{poly}(q, 1/s)$ (recall that degree $1/(qs)$ is needed).

Our construction is similar in spirit to what was described above: we introduce copies for each of the original proof symbols. For each original query the verifier makes queries to several of its copies, checking equality on the copies as well as checking the original test. Surprisingly, we show that only $\mathsf{poly}(q)$ queries chosen according to a disperser suffice. Our main insight is that since the soundness error s of the original PCP is small, it suffices to have a list decoding of size about $\sqrt[q]{1/s}$ for each one of the q queries. There is only probability about $(\sqrt[q]{s})^q = s$ that q copies of an original proof symbol all fall outside the list decoding.

1.3 An Application to MCSP

We expect that our regularization would have many applications in future. Here, we present a specific application of regularization to the Minimum Circuit Size Problem (MCSP) [26].

MCSP is the decision problem that asks to decide whether there exists a circuit of size s that computes a given function $f\colon \{0,1\}^n \to \{0,1\}$, given the truth table of f and a size parameter $s \in \mathbb{N}$. It is easy to see that MCSP $\in \mathsf{NP}$, but it is a long-standing open question whether MCSP is NP-hard. In fact, Levin [32] delayed his publication on the theory of NP-completeness because he hoped to prove NP-hardness of MCSP. In his seminal paper [32], he proved NP-completeness of DNF-MCSP*, i.e, the partial function variant of the Minimum DNF Formula Size Problem. In addition to its historical aspect, MCSP has connections to many areas of theoretical computer science, including learning theory [7], average-case complexity [17], circuit complexity [26, 36, 8], and cryptography [42, 33]. Recently, NP-hardness of the partial function variant of MCSP, denoted by MCSP*, was resolved under randomized polynomial-time reductions [18]. Here, MCSP* is the problem of deciding if there exists a circuit of size s that computes a given *partial* function f on input $x \in f^{-1}(\{0,1\})$, given the truth table of $f\colon \{0,1\}^n \to \{0,1,*\}$ and $s \in \mathbb{N}$ as input.

[2] In a robust PCP [5], in the soundness case, only s fraction of tests are even s-*close* (in Hamming distance) to satisfying. This notion is equivalent to projection PCP [11].

The starting point of the NP-hardness reduction of [18] is the problem called *Collective Minimum Monotone Satisfying Assignment Problem* (CMMSA). The input of CMMSA consists of a collection of monotone formulas of size Δ over n variables, where $\Delta \ll n$, and the task is decide whether there exists an assignment for the n variables with small weight that satisfies as many formulas as possible. In [18], the *weighted* version of CMMSA in which each variable has its own weight is shown to be NP-hard to approximate within a factor of $\Delta^{\Omega(1)}$ using low-error PCP systems. The reason why variables are assigned weights comes from the fact that low-error PCP systems may not be regular. Using our regularization for low-error PCPs, we prove NP-hardness of the unweighted version of CMMSA. This enables us to present a simpler proof of NP-hardness of MCSP*.

Using the simplified proof, we investigate whether MCSP* is NP-hard under the standard notion of reduction. The NP-hardness of MCSP* shown in [18] differs from the standard notion of NP-hardness in that the reduction is *randomized*. The usage of randomized reductions is in some sense necessary because of the connection to a circuit lower bound for explicit functions: A line of work [26, 20, 19, 41] shows that NP-hardness of MCSP* under deterministic reductions implies breakthrough separations, such as $\mathsf{EXP} \neq \mathsf{ZPP}$ or $\mathsf{EXP} \not\subseteq \mathsf{P/poly}$. Thus, proving NP-hardness of MCSP* under deterministic reductions is at least as difficult as the central open problems in complexity theory. In fact, the hardness of certain variants of MCSP under deterministic reductions characterizes some circuit lower bounds for explicit functions [1].

We present sufficient conditions under which the reduction of [18] can be derandomized: MCSP* is NP-hard under deterministic polynomial-time many-one reductions if there exists a family $f = \left\{f_{k,n} \colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}\right\}_{k,n\in\mathbb{N}} \in \mathsf{E} = \mathsf{DTIME}[2^{O(n)}]$ of functions with certain "direct sum" properties. Roughly speaking, for some parameter $\sigma \geq 2^{\Omega(n)}$, we require that (i) $f_{k,n}(i,\text{-})$ can be computed by a circuit of size σ for every $i \in \{0,1\}^k$, and that (ii) for any set $B \subseteq \{0,1\}^k$, the size of any circuit that computes $f_{k,n}(i,\text{-})$ for every $i \in B$ on average is at least $\gtrsim |B| \cdot \sigma$. Although the actual assumption is somewhat stronger, it can be shown that a random function satisfies the direct sum properties with high probability. The original proof of [18] heavily relies on Kolmogorov complexity, and it is unclear what property of random functions is used. Our contribution is to identify the direct sum properties that are sufficient for the proof to go through, by giving a simplified proof that does not rely on Kolmogorov complexity.

We mention that, in general, any randomized polynomial-time reduction for a problem in NP can be derandomized to a deterministic polynomial-time *nonadaptive* reduction that makes several queries, under the assumption that $\mathsf{E} = \mathsf{DTIME}[2^{O(n)}]$ cannot be computed by non-deterministic circuits of size $2^{\Omega(n)}$. This follows from the theory of pseudorandom generators secure against non-deterministic circuits [31, 43]. We also mention that Ilango [22] showed that MCSP* is hard under the Exponential-Time Hypothesis, which provides an *exponential-time* reduction from SAT to MCSP*. Here, we aim at obtaining NP-hardness of MCSP* under deterministic *polynomial-time many-one* reductions.

2 Regularization and Degree Reduction For General PCPs

2.1 Preliminaries

2.1.1 Expanders and Dispersers

We will use an explicit construction of expanders obtained by powering an explicit constant degree expander. For a constant degree expander one can use the construction of [39] based on the zig zag product. The same paper discusses powering as well. The parameters one can get are given in this lemma (for a proof see the appendix of [34]):

▶ **Lemma 4** (Explicit construction of expanders). *There is a constant $\alpha < 1$ and a function $\Delta : \mathbb{N} \to \mathbb{N}^+$ with $\Delta(D) = \Theta(D)$, such that given two natural numbers n and D, one can find in time polynomial in n and in D an undirected (multi-)graph $G = (V, E)$ with $|V| = n$, which is $\Delta(D)$-regular and whose adjacency matrix has second largest eigenvalue (in absolute value) $\lambda \leq (\Delta(D))^\alpha$.*

We will use expander random walk as a hitter (see, e.g., Theorem 4.7 in [47]):

▶ **Lemma 5** (Expander random walk hitting property). *Let $G = (V, E)$ be a Δ-regular undirected (multi-)graph, whose adjacency matrix has second largest eigenvalue (in absolute value) $\lambda\Delta$. Then, for any set $B \subseteq V$ of fraction $\mu = |B| / |V|$, the probability that a random walk $v_1, \ldots, v_t$ in G satisfies $v_i \in B$ for $i = 1, \ldots, t$ is at most $(\mu + \lambda)^t$.*

Lemma 4 and Lemma 5 give an explicit construction of a *disperser*:

▶ **Definition 6** (Disperser/hitter). *A (δ, ε)-disperser graph is a bi-regular bipartite graph $G = (U, V, E)$ such that for every set $B \subseteq V$ of fraction at most ε, for at most δ fraction of the $u \in U$ it holds that all of u's neighbors in G are in B.*

▶ **Corollary 7** (Explicit disperser). *For any $q \geq 1$ and $0 < \varepsilon < 1$, for any $N \geq (1/\varepsilon)^{q+1}$ there is an explicit construction of a $(e\varepsilon^q, \varepsilon)$-disperser $G = ([N], [M], E)$ with N-degree q and M-degree $q \cdot \mathsf{poly}(1/\varepsilon^q)$.*

Proof. Let $G = ([M], E)$ for $M = N/\Delta^q$ be an explicit expander of degree $\Delta = \mathsf{poly}(1/\varepsilon)$ and second eigenvalue $(\varepsilon/q)\Delta$ as given by Lemma 4. Let N correspond to length q walks in G. In the disperser each walk is connected to the q vertices it contains. Let $B \subseteq [M]$ of fraction ε. By Lemma 5, the fraction of walks that fall completely in B is $(\varepsilon+\varepsilon/q)^q = \varepsilon^q(1+1/q)^q \leq e\varepsilon^q$. ◀

2.1.2 PCP Verifiers and Their Parameters

▶ **Definition 8** (PCP verifier). *A PCP verifier for a language L is a procedure that on input x uses r bits of randomness to make q queries to a proof π of length m over alphabet Σ. The verifier satisfies the following:*

- Completeness: *If $x \in L$ then there exists π that the verifier accepts with probability at least c.*
- Soundness: *If $x \notin L$ then for all π the verifier accepts with probability at most s.*

Typically, if n is the input size one considers r that is logarithmic in n, so the answers to all the 2^r tests would make an NP witness if $x \in L$. The number of queries is typically a constant independent of n or slightly super constant; ideally $q = 2$. We have $s \geq 1/|\Sigma|^q$, since a random proof would satisfy at least this fraction of verifier tests. Thus, $|\Sigma|$ is ideally close to $1/s^{1/q}$. The completeness c is often 1. The soundness error s is as small as possible. Sometimes one considers a constant $s < 1$, and sometimes sub-constant s is desired. Note that $s \geq 2^{-r}$. Ideally one could hope for s that is exponentially small in r and polynomially small in n. However, it is currently a known open problem ("The Sliding Scale Conjecture" [4]) whether polynomially small error can be achieved simultaneously with constant number of queries. The state-of-the-art PCP with soundness error $s = 1/n$ has $q = \mathsf{poly}(\log \log n)$ queries [12].

▶ **Definition 9.** *The query graph Q_V of a PCP verifier V that uses r random bits to make q queries to a proof of length m is the bipartite graph that has the 2^r randomness strings of the verifier on one side and the m proof symbols of the proof π on the other side. Connect each randomness string to the q queries the verifier makes on this randomness string.*

2.2 Our Regularization and Its Analysis

Assume a PCP verifier V that uses r random bits to make q queries to a proof π over alphabet Σ and has completeness c and soundness s, where $s \leq 1/(eq), 1/|\Sigma|^a$ for a constant $0 < a \leq 1$ (e is the basis of the natural logarithm). We will construct a new, similar, PCP verifier V' with a bi-regular query graph of small degree as specified in Theorem 1.

Let $A = 1/s$. First, duplicate each of the 2^r tests A times, so each of the degrees is at least A. This causes r to grow to $r + O(\log(1/s))$ and does not change the other parameters. For $l = A, A+1, \ldots, 2^r \cdot A$ consider an explicit disperser $G_l = ([l], [m_l], E_l)$ as guaranteed by Corollary 7 for $N = l$ vertices, $[l]$-degree $q' = \lceil 6q/a \rceil$, and $\varepsilon = s^{1/(2q)}$. Note that $m_l < l$. If V queries the i'th symbol in the proof in $d(i)$ verifier tests, then replace the i'th symbol with $m_{d(i)}$ new symbols $\mathsf{symbol}(i, j)$, $1 \leq j \leq m_{d(i)}$ that are supposed to be copies of the i'th symbol. That is, in the proof for V' in the completeness case all those copies are assigned the same label from Σ as the one assigned to the i'th symbol in the completeness proof of V.

The verifier V' picks a uniformly random test of the verifier V. For every symbol i that the test queries, if the test is the t'th test on which the i'th symbol is queried ($1 \leq t \leq d(i)$), the verifier V' queries instead the q' copies of the i'th symbol that correspond to the $G_{d(i)}$ neighborhood of t. The verifier V' checks equality between the copies in addition to the original test. Overall the number of queries that V' makes is $O(q^2)$, and the degree of every proof symbol is the same $\mathsf{poly}(q, 1/s)$. This step does not change the number of random bits the verifier uses. The alphabet of the proof is the same as the alphabet of the proof of V. The completeness error c of V' is the same as the completeness error c of V.

It remains to prove soundness. Suppose that we have a proof for V' that V' accepts with probability larger than $2\sqrt{s}$.

Let $L = 1/\varepsilon$ and assume for simplicity that L is a natural number (otherwise, round it). For every original proof symbol i, consider the L labels from the alphabet Σ that repeat in the proof of V' in the largest number of copies $\mathsf{symbol}(i, j)$. We call a label from Σ "bad" for i if it is not one of those L. We call a copy "bad" for i if its label is bad for i. Note that a bad label repeats in at most ε fraction of copies.

For any original query i, consider a uniform choice of a V test among the $d(i)$ tests that query it, as well as the q' corresponding queries of V' to copies $\mathsf{symbol}(i, j)$. By the disperser property, for every bad label $\sigma \in \Sigma$ for i, the probability that V' accepts and queries copies labeled σ is at most $e\varepsilon^{q'} \leq es^{3/a} \leq s/(q|\Sigma|)$, where the last inequality used the low soundness error of V. Consider a uniform test of V', which induces a uniform test of V that makes q original queries. By a union bound over the q queries and $|\Sigma|$ possible bad labels for them, the probability that V' accepts yet for one of the q queries it queries a bad copy is at most s.

Hence, with probability larger than $2\sqrt{s} - s \geq \sqrt{s}$ over a choice of a uniform V' test, the verifier accepts and for none of the q original queries it queries a bad copy. Consider the following proof for V: for every proof symbol pick uniformly at random one of its L labels. The probability that this assignment is accepted is larger than $\sqrt{s} \cdot (1/L)^q = s$.

3 Application: NP-Hardness of Partial MCSP

As an application, we simplify the proof of NP-hardness of MCSP* and present two sufficient conditions under which the randomized reductions of [18] can be derandomized. The first sufficient condition is that E cannot be computed by 2^{cn}-time algorithms with $2^n - 2^{n/2}$ bits of advice for a sufficiently large constant c. This condition is essentially equivalent to the statement that there exists a polynomial-time algorithm that, on input 1^N, outputs a string

of length N whose time-bounded Kolmogorov complexity is at least $N - \sqrt{N}$ [40].[3] The second sufficient condition is weaker and is that there exists a function in E that satisfies "direct sum" properties.

To define the direct sum properties formally, we introduce the notion of *oracle-sum circuit*, which generalizes a standard circuit. An oracle-sum circuit consists of a pair (C, D) of an oracle circuit C and a circuit D. The oracle-sum circuit computes a function f such that $f(x) = C^D(x)$, i.e., the function computed by the D-oracle circuit C. Abusing the notation, we identify (C, D) with the function computed by (C, D). The size of an oracle-sum circuit is measured by $|C| + |D|$, where $|C|$ and $|D|$ denote the number of wires in C and D, respectively. Note that it is possible to simulate an oracle-sum circuit (C, D) by a circuit of size $O(|C| \cdot |D|)$ by having $|C|$ copies of the circuit D. The main difference between an oracle-sum circuit and a standard circuit lies in how we measure their size.

For a function $f\colon \{0,1\}^n \to \{0,1\}$, let $f^m\colon (\{0,1\}^n)^m \to \{0,1\}^m$ denote the m-wise direct product of f_i, i.e., the function defined as $f^m(x_1, \dots, x_m) := (f(x_1), \dots, f(x_m))$. We now provide the formal definition of direct sum properties.

▶ **Definition 10.** *For a function $\sigma\colon \mathbb{N}^2 \to \mathbb{N}$, a family $f = \left\{f_{k,n}\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}\right\}_{k,n\in\mathbb{N}}$ of functions is said to have* σ-direct sum properties *if the following hold for some constant $\delta > 0$ and all sufficiently large constant c.*

1. *For every $B \subseteq \{0,1\}^k$, there exists a circuit C of size $|B| \cdot \sigma(k, n)$ such that C computes the m-wise direct product of f_i for every $i \in B$, i.e., $(f_i^m \mid i \in B)$, where $m := n^c$. Here, $f_i(x) := f_{k,n}(i, x)$.*
2. *For every $B \subseteq \{0,1\}^k$ and every oracle-sum circuit C of size at most $|B| \cdot \sigma(k, n) \cdot n^{-1/c}$, there exists $i \in B$ such that $\Pr_{x\sim\{0,1\}^n}[C(i, x) = f_i(x)] \leq 1 - \delta$.*

Roughly speaking, a function with σ-direct sum properties satisfies that the circuit complexity of computing $(f_i \mid i \in B)$ is approximately equal to $\sigma \cdot |B|$ for every $B \subseteq \{0,1\}^k$. Definition 10 is stronger than this in the following respects: (i) Item 1 states that for every $i \in B$, not only each f_i is computable by a circuit of size σ, but also the m-wise direct product of f_i is computable by a circuit of size σ. In particular, computing f_i^m is as easy as computing f_i, which means that the strong direct sum property for f_i *fails* to hold. (ii) Item 2 is a strong direct sum property for $(f_i \mid i \in B)$, and states that *oracle-sum* circuits of size $\lesssim |B| \cdot \sigma$ cannot compute $(f_i \mid i \in B)$ *on average*.

The formal definition of MCSP* is as follows.

▶ **Definition 11.** *For a partial function $f\colon \{0,1\}^n \to \{0,1,*\}$, let $\mathrm{CC}^*(f)$ denote the minimum number of the wires in a circuit C such that $C(x) = f(x)$ for every $x \in \{0,1\}^n$.* Partial MCSP (MCSP*) *is defined as the language that consists of (f, s) such that $\mathrm{CC}^*(f) \leq s$, where f is encoded as a binary string of length $2^{\Theta(n)}$.*

We now state the main result of this section.

▶ **Theorem 12.** *Assume that either*

1. *for every constant $c > 0$, there exists a language in $\mathsf{E} \setminus \mathsf{i.o.DTIME}[2^{cn}]/(2^n - 2^{n/2})$, or*
2. *there exists a family $f = \{f_{k,n}\}_{k\leq n}$ functions computable in time $2^{O(n)}$ with σ-direct sum properties, where $\sigma(k, n) \geq 2^{\gamma n}$ for some constant $\gamma > 0$.*

Then, MCSP* *is* NP*-hard under deterministic polynomial-time many-one reductions.*

We first show that the first condition implies the second condition in Theorem 12. Item 1 of Definition 10 follows from Uhlig's theorem:

[3] We mention that this can be optimized to $N - N^{1-\epsilon}$ for any constant $\epsilon > 0$.

▶ **Lemma 13** ([45, 46]; see also [48]). *Let $r\colon \mathbb{N} \to \mathbb{N}$ be a function such that $r(n) = 2^{o(n/\log n)}$. Then, for all large $n \in \mathbb{N}$, for any function $f\colon \{0,1\}^n \to \{0,1\}$, there exists a circuit of size $O(2^n/n)$ that computes $f^{r(n)}\colon (\{0,1\}^n)^{r(n)} \to \{0,1\}^{r(n)}$.*

▶ **Proposition 14.** *There exists a constant $c > 0$ such that if $f\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}$ cannot be computed by any algorithm running in time 2^{cn} with $2^{n+k} - 2^{n-1}$ bits of advice, then f satisfies the σ-direct sum property for $\sigma = \Theta(2^n/n)$.*

Proof. To see the first property of Definition 10, for each $i \in B$, by Lemma 13, there exists a circuit C_i of size $O(2^n/n)$ that computes the m-wise direct product f_i^m of f_i. By combining the circuits $(C_i \mid i \in B)$ for all $i \in B$, we obtain a circuit of size $|B| \cdot O(2^n/n)$ that computes f_i^m for every $i \in B$.

Let $\mathrm{K}^t(x)$ denote the t-time bounded Kolmogorov complexity of x, i.e., the length of a shortest program that prints x in time t. Then, the assumption implies that $\mathrm{K}^{2^{cn}}(f) \geq 2^{n+k} - 2^{n-1}$.

To see the second property, we first claim that for every $B \subseteq [n]$, the time-bounded Kolmogorov complexity of $f_B := (f_i \mid i \in B)$ is at least $|B| \cdot 2^{n-2}$. Since f can be described by a description for $(f_i \mid i \in B)$, the set B, and $(f_i \mid i \in \{0,1\}^k \setminus B)$, we have

$$2^{n+k} - 2^{n-1} \leq \mathrm{K}^t(f) \leq \mathrm{K}^{t'}(f_B) + |B| \cdot O(k) + (2^k - |B|) \cdot 2^n$$

for some $t, t' \leq 2^{O(n)}$. It follows that $\mathrm{K}^t(f_B) \geq |B| \cdot (2^n - 2^{n-1} - O(k)) \geq |B| \cdot 2^{n-2}$. We now claim that any small oracle-sum circuit fails to approximate $(f_i \mid i \in B)$. Let C be an oracle-sum circuit of size s such that $\Pr_{x \sim \{0,1\}^n}[C(i,x) = f_i(x)] \geq 1 - \delta$ for every $i \in B$. For each $i \in B$, the set of inputs x such that $C(i,x) \neq f_i(x)$ can be described by $\log \sum_{k \leq \delta 2^n} \binom{2^n}{k} \leq \mathrm{H}_2(\delta) 2^n \leq 2^{n-3}$ bits, where the last inequality holds for a sufficiently small constant $\delta > 0$. Since C can be described by $O(s \log s)$ bits, $(f_i \mid i \in B)$ can be described by $O(s \log s) + |B| \cdot 2^{n-3}$. Thus, we obtain $|B| \cdot 2^{n-3} \leq O(s \log s)$, which implies $s \geq \Omega(|B| \cdot 2^n/n)$. ◀

Since a random function has high Kolmogorov complexity, the proof of Proposition 14 also shows that a random function satisfies $\Theta(2^n/n)$-direct sum properties with high probability.

3.1 Collective Minimum Monotone Satisfying Assignment Problem

In [18], *Collective Minimum Monotone Satisfying Assignment* (CMMSA) was introduced and shown to be NP-hard to approximate. Using the regularization for low-error PCPs, we show that the same hardness of approximation can be proved for the unweighted version of CMMSA.

For an assignment $\alpha\colon [n] \to \{0,1\}$, let $w(\alpha)$ denote the Hamming weight $\sum_{i=1}^n \alpha(i)$ of α. For a formula φ, let $\varphi(\alpha) \in \{0,1\}$ denote the output of φ when the variables are assigned by α.

▶ **Definition 15** ([18]). *The* Collective Minimum Satisfying Assignment problem (CMMSA) with gap $g \in \mathbb{N}$ and soundness $\epsilon > 0$ *is the following problem. The input consists of a collection $\Phi = \{\varphi_1, \ldots, \varphi_m\}$ of monotone formulas over the set $[n]$ of variables and a threshold parameter $s \in \mathbb{N}$. The task is to distinguish the following two cases.*

Yes: *There exists an assignment $\alpha\colon [n] \to \{0,1\}$ such that*

$$w(\alpha) \leq s \quad \text{and} \quad \Pr_{\varphi \sim \Phi}[\varphi(\alpha) = 1] = 1.$$

No: *For every assignment* $\alpha\colon [n] \to \{0,1\}$*, if* $w(\alpha) \le g \cdot s$*, then*

$$\Pr_{\varphi\sim\Phi}[\varphi(\alpha) = 1] < \epsilon.$$

The degree *of* Φ *is defined to be* $\max_{\varphi\in\Phi} |\varphi|$*, where* $|\varphi|$ *denotes the number of the literals in the formula* φ*. The size of an instance of* CMMSA *is measured by the number* n *of input variables.*

▶ **Theorem 16.** *For any constant* $\beta > 0$*, there exists a constant* $\alpha > 0$ *such that for every parameter* $\Delta\colon \mathbb{N} \to \mathbb{N}$ *such that* $\omega(1) \le \Delta(n) \le 2^{(\log n)^{1-\beta}}$ *for all large* $n \in \mathbb{N}$*, it is* NP*-hard under polynomial-time many-one reductions to compute* CMMSA *with gap* $\Delta(n)^{\alpha}$*, degree* $\Delta(n)$*, and soundness* $\Delta(n)^{-\alpha}$ *on a collection* Φ *of monotone DNF formulas over* n *variables.*

The proof of Theorem 16 is essentially the same with [18], except that we use the regularized PCP system of Corollary 2.

Proof of Theorem 16. The PCP theorem of Corollary 2 can be stated in terms of MaxCSP as follows: Let $\Psi = \{C_1, \dots, C_m\}$ be the set of constraints over n variables on the alphabet Σ. Here, for any internal randomness $j \in \{0,1\}^{O(\log n)}$ of a PCP verifier, there is a constraint C_j. Each constraint C_j depends on $D = O(1/\beta)$ variables. The size of the alphabet Σ is at most $\mathsf{poly}(1/\delta)$, where δ is the soundness error. Let $C_j^{-1}(1)$ denote the set of assignments to the variables in C_j that cause C_j to accept. Here, an assignment r to the variables in C_j is a function $r\colon \mathrm{dom}(r) \to \Sigma$, where $\mathrm{dom}(r) \subseteq [n]$ denotes the set of variables in C_j.

Given the MaxCSP instance Ψ over Σ, we reduce it to an instance (Φ, s) of CMMSA as follows: Each variable of Φ is indexed by $(x, a) \in [n] \times \Sigma$ and is denoted by $L_{x,a}$. Informally, $L_{x,a} = 1$ indicates that the variable x in the original CSP instance Ψ is assigned to $a \in \Sigma$. For each $j \in [m]$, construct a monotone DNF formula φ_j defined as

$$\varphi_j(L) := \bigvee_{r\in C_j^{-1}(1)} \bigwedge_{x\in\mathrm{dom}(r)} L_{x,r(x)}.$$

The threshold s is defined to be n.

We prove the correctness of the reduction. Assume that the CSP instance Ψ is satisfied by an assignment $\alpha\colon [n] \to \Sigma$. Then, we set $L_{x,\alpha(x)} := 1$ and $L_{x,y} := 0$ for every $y \in \Sigma \setminus \{\alpha(x)\}$. The weight of the assignment $L\colon [n] \times \Sigma \to \{0,1\}$ is $w(L) = n$. By the perfect completeness, we have $C_j(\alpha) = 1$ for every $j \in [m]$; thus, α satisfies every formula in Φ. It follows that (Φ, s) is a Yes instance of CMMSA.

Next, assume that any assignment to Ψ can satisfy at most a δ-fraction of constraints in Ψ. Assume that there exists an assignment $L\colon [n] \times \Sigma \to \{0,1\}$ such that $w(L) = g \cdot n$ and

$$\Pr_{j\sim[m]}[\varphi_j(L) = 1] \ge \epsilon, \tag{1}$$

where $\epsilon > 0$ is a parameter to be chosen later. We claim that g must be large. For each variable $x \in [n]$ of Ψ, let $A(x) := \{a \in \Sigma \mid L_{x,a} = 1\}$. Since $gn = w(L) = \sum_{x\in[n]} |A(x)|$, we have $\mathbf{E}_{x\sim[n]}[|A(x)|] = g$. We say that $x \in [n]$ is *bad* if $|A(x)| \ge 2gD/\epsilon$. By Markov's inequality, the probability that x is bad is at most $\epsilon/2D$. Since the PCP system is bi-regular, the uniform distribution $x \sim [n]$ is identical to the following distribution: First choose $j \sim [m]$, and then choose x uniformly at random from the variables in C_j. We say that C_j is bad if C_j contains some bad variable x. Thus, the probability, over $j \sim [m]$, that C_j is bad is at most $\epsilon/2$. Combining this with Equation (1), we obtain that

$$\Pr_{C\sim\Psi}\left[\exists r \in C^{-1}(1), \forall x \in \mathrm{dom}(r),\ |A(x)| \le \frac{2gD}{\epsilon} \text{ and } r(x) \in A(x)\right] \ge \epsilon - \frac{\epsilon}{2} = \frac{\epsilon}{2}.$$

Now, we construct a random assignment $\alpha\colon [n] \to \Sigma$ as follows: For each $x \in [n]$, pick $a \sim A(x) \subseteq \Sigma$ uniformly and randomly and define $\alpha(x) := a$. Under the event that $r \in C^{-1}(1)$, $|A(x)| \leq \frac{2gD}{\epsilon}$, and $r(x) \in A(x)$ for every $x \in \mathrm{dom}(r)$, we have $C(\alpha) = 1$ if $\alpha(x) = r(x)$ for every $x \in \mathrm{dom}(r)$, which happens with probability at least $\left(\frac{\epsilon}{2gD}\right)^D$. It follows that

$$\delta \geq \Pr_{\substack{C \sim \Psi \\ \alpha}}[C(\alpha) = 1] \geq \frac{\epsilon}{2} \cdot \left(\frac{\epsilon}{2gD}\right)^D,$$

which implies that $g \geq \Omega\left(\epsilon^2 \cdot \delta^{-\frac{1}{D}}\right) \geq \Omega\left(\delta^{-\frac{1}{2D}}\right)$, where the last inequality holds by setting $\epsilon := \delta^{\frac{1}{4D}}$. The number of the literals in $\varphi_j \in \Phi$ is at most $|C_j^{-1}(1)| \cdot D \leq |\Sigma|^D \cdot D \leq \delta^{-O(D)}$. Given a parameter Δ, we choose $\delta := \Delta^{-\Omega(1/D)}$ so that the degree of Φ is at most Δ. Then, the gap g is at least $\Omega\left(\delta^{-\frac{1}{2D}}\right) \geq \Delta^{\Omega(1/D^2)}$. Moreover, the soundness ϵ is at least $\delta^{\frac{1}{4D}} \geq \Delta^{-\Omega(1/D^2)}$. ◀

3.2 Technical Tools

We review the three technical tools used in [18]. The first tool is a secret sharing scheme.

▶ **Definition 17** (Secret Sharing Scheme [3]). *A* secret sharing scheme *for $\mathcal{A} \subseteq 2^{[n]}$ is a pair* $(\mathrm{Share}, \mathrm{Rec})$ *of a randomized algorithm* Share *and a deterministic algorithm* Rec *with the following properties:*

Correctness: *For every authorized set $T \in \mathcal{A}$ and for every bit $b \in \{0,1\}$, the output of* $\mathrm{Share}(b)$ *is a sequence $(s_1, \ldots, s_n)$ of n strings that satisfies* $\mathrm{Rec}(T, s_T) = b$ *with probability* 1 *over the internal randomness of* $\mathrm{Share}(b)$.

Privacy: *For every unauthorized set $T \notin \mathcal{A}$ and for every random variable b on $\{0,1\}$, the random variables b and $\mathrm{Share}(b)_T$ are statistically independent.*

For a monotone formula φ, let $\mathcal{A}_\varphi$ denote the access structure such that $T \in \mathcal{A}_\varphi$ if and only if the indicator function of $T \subseteq [n]$ satisfies φ.

▶ **Lemma 18** ([24, 6]). *Let $\mathcal{A} = \{\mathcal{A}_\varphi\}_\varphi$ be the family of access structures $\mathcal{A}_\varphi$ represented by monotone formulas φ. Then, there exists a pair of a randomized polynomial-time algorithm* Share *and a deterministic polynomial-time algorithm* Rec *such that for every monotone formula φ, the pair* $(\mathrm{Share}(\varphi, \text{-}), \mathrm{Rec}(\varphi, \text{-}))$ *is a secret sharing scheme for the access structure $\mathcal{A}_\varphi$. Moreover, the length $|s_i|$ of each share s_i is at most the number $|\varphi|$ of the literals in the formula φ.*

The second tool is the Nisan–Wigderson pseudorandom generator construction.

▶ **Proposition 19** ([35, 44]). *For any sufficiently large parameters $\ell, m, \rho \in \mathbb{N}$ with $m \leq 2^\ell$, there exists a "design" $S_1, \ldots, S_m \subseteq [d]$ such that for every $i \in [m]$,*

1. *$|S_i| = \ell$, $d = O(\exp(\ell/\rho) \cdot \ell^2/\rho)$, and*
2. *$|S_i \cap S_j| \leq \rho$ for every $j \in [m] \setminus \{i\}$*

Moreover, such a family can be constructed in time $\mathsf{poly}(2^d, m)$.

▶ **Definition 20** (The Nisan–Wigderson pseudorandom generator construction [35]). *Let $\mathcal{S} = (S_1, \ldots, S_m)$ be a family of ℓ-sized subsets of $[d]$. For a function $f\colon \{0,1\}^\ell \to \{0,1\}$, we define a function*

$$\mathrm{NW}_{\mathcal{S}}\colon \{0,1\}^{2^\ell} \times \{0,1\}^d \to \{0,1\}^m$$

as

$$\mathrm{NW}_{\mathcal{S}}(f;z) := (f(z_{S_1}),\dots,f(z_{S_m})) \in \{0,1\}^m,$$

where $z_{S_i} \in \{0,1\}^\ell$ *denotes the string obtained by concatenating all the bits of* $z \in \{0,1\}^d$ *indexed by* $S_i \subseteq [d]$.

The third tool is a derandomized version of Yao's XOR lemma.

▶ **Lemma 21** ([23, 16]; see also [18]). *For any constant* $\gamma > 0$*, there exist a constant* $c \in \mathbb{N}$ *and a procedure* Amp *that takes a function* $f\colon \{0,1\}^n \to \{0,1\}$ *and parameters* $\epsilon, \delta \in (0,1/2)$ *as input, and returns a function* $\mathrm{Amp}^f = \mathrm{Amp}^f_{\epsilon,\delta}\colon \{0,1\}^{cn} \to \{0,1\}$ *that satisfies the following properties:*

1. *For every circuit* D *that computes* Amp^f *on a* $(1/2+\epsilon)$*-fraction of inputs, there exists an oracle circuit* C *of size* $2^{\gamma n} \cdot \mathsf{poly}(1/\epsilon\delta)$ *such that* C^D *computes* f *on a* $(1-\delta)$*-fraction of inputs.*
2. *There is a nonadaptive* f*-oracle circuit of size* $\mathsf{poly}(n/\epsilon\delta)$ *and depth* $O(\log(n/\epsilon\delta))$ *that computes* Amp^f *by making* $O(1/\epsilon\delta)$ *queries to* f*.*
3. Amp^f *can be computed in time* $\mathsf{poly}(2^n, n/\epsilon\delta)$ *given the truth table of* f *and the parameters as input.*

3.3 Proof of NP-hardness of MCSP*

We are ready to present a proof of Theorem 12. As shown in Proposition 14, the first condition is stronger than the second condition. Thus, it suffices to show NP-hardness of MCSP* under the second condition that there exists a family $F = \{F_{k,n}\}_{k\le n}$ of functions with σ-direct sum properties, where $\sigma(k,n) \ge 2^{\gamma n}$. Let $\delta > 0$ be the constant of Definition 10.

We present a reduction from CMMSA with degree Δ and soundness ϵ_0 to MCSP*, where $\Delta := (\log n)^{1/2}$ and $\epsilon_0 < 1/4$. Let (Φ,θ) be an instance of CMMSA, where $\Phi = \{\varphi_1,\dots,\varphi_\nu\}$ is a degree-Δ collection of monotone formulas over the set $[n]$ of input variables. For each $j \in [\nu]$, let V_j denote the set $\left\{v^j_1,\cdots,v^j_m\right\} \subseteq [n]$ of the variables of φ_j. Here, $m \le \Delta$ is the number of variables on which φ_j depends for every $j \in [\nu]$. We may assume without loss of generality that m does not depend on j and n is a power of 2.

Let Amp be the hardness amplification procedure of Lemma 21 for $\epsilon := \epsilon_0/2\Delta m$, and let $c \ge 1$ be the constant of Lemma 21. Let $\lambda = n^{O(1)}$ be a sufficiently large parameter. We define $\ell := c\log\lambda$.

Let $f := F_{\log n,\log\lambda}\colon \{0,1\}^{\log n} \times \{0,1\}^{\log\lambda} \to \{0,1\}$, where we identify $\{0,1\}^{\log n}$ with $[n]$. For each $k \in [n]$, let $f_k\colon \{0,1\}^{\log\lambda} \to \{0,1\}$ be the function such that $f_k(x) = f(k,x)$ for every $x \in \{0,1\}^{\log\lambda}$. Let $\widehat{f_k} := \mathrm{Amp}^{f_k}\colon \{0,1\}^\ell \to \{0,1\}$ denote the hardness-amplified version of the function f_k.

We construct a partial function

$$g\colon \{0,1\}^{O(\log\nu)} \times \{0,1\}^d \times (\{0,1\}^\Delta)^m \to \{0,1,*\},$$

which is the output of the reduction, as follows. Let $\mathcal{S} = (S_1,\dots,S_{m\Delta})$ be the collection of ℓ-sized subsets of $[d]$ from Proposition 19, where $d = O(\ell)$ and $\rho := \gamma\log\lambda$. Let $\mathcal{S}_i := (S_{(i-1)\Delta+1},\dots,S_{(i-1)\Delta+\Delta})$ for every $i \in [m]$. g takes $x = (j,z,\xi_1,\dots,\xi_m) \in \{0,1\}^{O(\log\nu)} \times \{0,1\}^d \times (\{0,1\}^\Delta)^m$ as input. Define $s_i := \xi_i \oplus \mathrm{NW}_{\mathcal{S}_i}\left(\widehat{f_{v^j_i}};z\right)$ for every $i \in [m]$, where $\oplus$ denotes the bit-wise XOR. Then, we check if $(s_1,\dots,s_m)$ can be obtained by running $\mathrm{Share}(\varphi_j,b)$ for some secret $b \in \{0,1\}$ and some internal randomness of Share. (Here, Share

is the randomized algorithm of Lemma 18.) If not, we define $g(x) := *$; otherwise, we define $g(x) := b$. Observe that $|x| = O(\log \nu + d + \Delta m) = O(\log n + \Delta^2) = O(\log n)$.

We prove the correctness of the reduction that maps (Φ, θ) to an instance $(g, 2\theta\sigma)$ of MCSP* in the following two claims.

▷ Claim 22. If (Φ, θ) is a Yes instance of CMMSA, then $\mathrm{CC}^*(g) \leq \theta\sigma + \mathsf{poly}(n \log \lambda/\epsilon\delta)$, where $\sigma := \sigma(\log n, \log \lambda)$.

Since $\mathsf{poly}(n \log \lambda/\epsilon\delta) \leq 2^{\gamma \log \lambda} \leq \sigma$ for a sufficiently large λ, it follows that $\mathrm{CC}^*(g) \leq 2\theta\sigma$ in the Yes case.

Proof of Claim 22. Let $\alpha\colon [n] \to \{0,1\}$ be an assignment of weight θ that satisfies every formula in Φ. Define $T := \alpha^{-1}(1)$. We construct a circuit C that computes the partial function g. Let $x = (j, z, \xi_1, \ldots, \xi_m)$ be an input to C. First, for each $k \in T$, the circuit C computes strings $(y_1^k, \ldots, y_\Delta^k) \in \left(\{0,1\}^\ell\right)^\Delta$ such that if there exists $i \in [m]$ such that $k = v_i^j$, then $y_p^k = z_{S_{(i-1)\Delta+p}}$ for every $p \in [\Delta]$. Then, for each $k \in T$, the circuit C computes $(\widehat{f_k}(y_1^k), \ldots, \widehat{f_k}(y_\Delta^k))$ from $(y_1^k, \ldots, y_\Delta^k)$. By Lemma 21, each $\widehat{f_k}(y_p^k)$ can be computed by $O(1/\epsilon\delta)$ nonadaptive queries to f; thus, the tuple $\left(\widehat{f_k}(y_p^k) \;\middle|\; p \in [\Delta]\right)$ can be computed by $O(\Delta/\epsilon\delta)$ nonadaptive queries to f_k. By the direct sum property of F, the nonadaptive queries to f_k for every $k \in T$ can be simulated by a circuit of size $|T| \cdot \sigma \leq \theta \cdot \sigma$. Finally, C computes $s_i := (\widehat{f_k}(y_1^k), \ldots, \widehat{f_k}(y_\Delta^k)) \oplus \xi_i$ for every k and i such that $k = v_i^j$. Then, C outputs $b = \mathrm{Rec}(\varphi_j, V_j \cap T, s_{V_j \cap T})$. Overall, the size of the circuit is

$$\theta \cdot \sigma + \mathsf{poly}(n\Delta \log \lambda/\epsilon\delta). \qquad \lhd$$

Let $\eta > 0$ be a constant such that CMMSA is NP-hard to approximate to within a factor of $4(\log \lambda)^\eta$. For a No instance (Φ, θ) of CMMSA, we claim that $\mathrm{CC}^*(g)$ is large.

▷ Claim 23. Assume that

$$\Pr_{j \sim [\nu]}[\varphi_j(\alpha) = 1] \leq \epsilon_0$$

for every assignment α of weight $4\theta \cdot (\log \lambda)^\eta$. Then,

$$\mathrm{CC}^*(g) > 2\theta\sigma.$$

Proof of Claim 23. Let C be an arbitrary circuit of size $2\theta\sigma$. We prove that C cannot compute g on average with respect to some distribution.

We say that C *knows* $k \in [n]$ if there exists an oracle circuit S of size t such that S^C computes f_k on a $(1-\delta)$-fraction of inputs, where $t := 2^{\gamma \log \lambda} \cdot \mathsf{poly}(m\Delta/\epsilon\delta)$. Let B be the set of $k \in [n]$ such that C knows k.

For every $j \in [\nu]$ and every $i \in \{0, \ldots, m\}$, we consider the hybrid distribution H_i^j defined by the following sampling procedure: Choose a secret $b \sim \{0,1\}$ randomly. Let $(s_1, \ldots, s_m) := \mathrm{Share}(\varphi_j, b)$. Define

$$x := \left(j, z, Y_1, \ldots, Y_i, Y'_{i+1}, \ldots, Y'_m\right),$$

where $Y_a := \mathrm{NW}_{\mathcal{S}_a}\left(\widehat{f_{v_a^j}}; z\right) \oplus s_a$ for every $a \in [m]$ and $Y'_a := Y_a$ if $v_a^j \in B$ and $Y'_a \sim \{0,1\}^\Delta$ otherwise. Output (x, b).

Fix any $j \in [\nu]$ and $i \in [m]$. We claim that

$$\left| \Pr_{(x,b) \sim H_{i-1}^j}[C(x) = b] - \Pr_{(x,b) \sim H_i^j}[C(x) = b] \right| \leq \epsilon\Delta. \tag{2}$$

Assume, towards a contradiction, that this does not hold. The only difference between H^j_{i-1} and H^j_i is that the i-th coordinate is Y'_i in the former and is Y_i in the latter. Let $k := v^j_i$. If $k \in B$, it is evident that the two distributions are identical, in which case we are done. Thus, assume $k \notin B$. In this case, Y'_i is the uniform distribution. We use a standard security proof of the Nisan–Wigderson generator to construct a C-oracle circuit S^C of size t that computes f_k, which contradicts $k \notin B$. Specifically, we use a hybrid argument in which each bit of Y_i is replaced with Y'_i. Then, there exists a bit position a of Y_i that can be distinguished from the uniform distribution. We fix $z_{[m\Delta]\setminus S_a}$, b, the randomness of $\mathrm{Share}(\varphi_j, b)$ so that the distinguishing probability is preserved. Since $|z_{S_a} \cap z_{S_{a'}}| \le \rho$ for every $a' \neq a$, given z_{S_a}, one can compute $\left(\widehat{f_k}(z_{S_{a'}}) \;\middle|\; a' \in [m\Delta] \setminus \{a\}\right)$ using a circuit of size $O(m\Delta 2^\rho \rho) \le \lambda^\gamma \cdot \mathsf{poly}(m\Delta)$. By Yao's next bit predictor, we obtain a C-oracle circuit that computes $\widehat{f_k}(z_{S_a})$ on a $(1/2+\epsilon)$-fraction of inputs. By Lemma 21, we obtain a C-oracle circuit S^C that computes f_k on a $(1-\delta)$-fraction of inputs. The size of S is at most $\lambda^\gamma \cdot \mathsf{poly}(m\Delta/\epsilon\delta) \le t$, which implies $k \in B$. However, this contradicts $k \notin B$.

It follows from Equation (2) that

$$\left| \Pr_{(x,b)\sim H^j_0}[C(x) = b] - \Pr_{(x,b)\sim H^j_m}[C(x) = b] \right| \le \epsilon \Delta m.$$

Observe that $g(x) = b$ for every (x, b) in the support of H^j_m. Thus, we have

$$\Pr_{(x,b)\sim H^j_m}[C(x) = b] = \Pr_{(x,b)\sim H^j_m}[C(x) = g(x)]$$

Let $\alpha\colon [n] \to \{0,1\}$ be a function such that $\alpha(k) = 1$ iff $k \in B$. In the distribution of $(x, b) \sim H^j_0$, only the shares in B are included in x. Thus, if $\varphi_j(\alpha) = 0$, then by the privacy of the secret sharing scheme, b and x are statistically independent, in which case we have

$$\Pr_{(x,b)\sim H^j_0}[C(x) = b] = \frac{1}{2}.$$

We claim that the size of B is small. In order to use the direct sum property of F, we construct a small oracle-sum circuit (S, C) that approximates f_k for every $k \in B$. For each $k \in B$, let S_k be the oracle circuit S_k of size t such that S^C_k computes f_k on a $(1-\delta)$-fraction of inputs. We define an oracle circuit S^C as follows: Given $k \in B$ and $x \in \{0,1\}^n$ as input, the circuit outputs $S^C_k(x)$. The size of S is at most $(1 + o(1)) \cdot |B| \cdot t$. The size of the oracle-sum circuit (C, S) is at most $|S| + |C|$. By the direct sum property of F, we obtain $|B| \cdot \sigma \cdot (\log \lambda)^{-\eta} \le |S| + |C| \le (1 + o(1)) \cdot |B| \cdot t + |C|$. Since $t \ll \sigma \cdot (\log \lambda)^{-\eta}$, we obtain $|B| \le (1 + o(1)) \cdot |C| \cdot (\log \lambda)^\eta / \sigma \le \theta \cdot 4(\log \lambda)^\eta$. By the assumption, we have $\Pr_j[\varphi_j(\alpha) = 1] \le \epsilon_0$.

Choose $j \sim [\nu]$ randomly. Then, we obtain

$$\begin{aligned}\Pr_{j\sim[\nu],(x,b)\sim H^j_m}[C(x) = g(x)] &\le \Pr_j[\varphi_j(\alpha) = 1] + \Pr_{j,(x,b)}[C(x) = g(x) \mid \varphi_j(\alpha) = 0] \\ &\le \epsilon_0 + \frac{1}{2} + \epsilon\Delta m \le \frac{1}{2} + 2\epsilon_0 < 1.\end{aligned}$$

◁

4 Open Problems

Can we show NP-hardness of MCSP* under circuit lower bound assumptions, such as the assumption that E cannot be computed by non-deterministic circuits of size $2^{\epsilon n}$ for some constant $\epsilon > 0$? Using a pseudorandom generator secure against non-deterministic

algorithms [31, 43], one can generate, in time $\mathsf{poly}(N)$, functions $f_1, \ldots, f_N \colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}$ such that f_i satisfies the $\Theta(2^n/n)$-direct sum properties for most $i \in [N]$, where $N = 2^{O(n+k)}$. However, it remains open whether a *single* function f with direct sum properties can be obtained.

It is interesting to see whether there exists any candidate for a function with σ-direct sum properties, where $2^{\Omega(n)} \leq \sigma \ll 2^n/n$. In this regime, Uhlig's theorem (Lemma 13) cannot be used, so new insights would be required to answer this question.

The original motivation of the regularization was to show NP-hardness of learning sparse parities by small programs, which was raised as an open problem in [18]. Unfortunately, it turned out that regularization is not sufficient for resolving this question. It remains open whether learning sparse parities by small programs is NP-hard.

References

1 E. Allender and S. Hirahara. New Insights on the (Non-)Hardness of Circuit Minimization and Related Problems. *TOCT*, 11(4):27:1–27:27, 2019. doi:10.1145/3349616.

2 J. Alman and L. Chen. Efficient construction of rigid matrices using an NP oracle. In *Proc. 60th IEEE Symp. on Foundations of Computer Science*, pages 1034–1055, 2019.

3 A. Beimel. Secret-Sharing Schemes: A Survey. In *The Third International Workshop on Coding and Cryptology (IWCC)*, pages 11–46, 2011. doi:10.1007/978-3-642-20901-7_2.

4 M. Bellare, S. Goldwasser, C. Lund, and A. Russell. Efficient probabilistically checkable proofs and applications to approximations. In *Proc. 25th ACM Symp. on Theory of Computing*, pages 294–304, 1993.

5 E. Ben-Sasson, O. Goldreich, P. Harsha, M. Sudan, and S. Vadhan. Robust PCPs of proximity, shorter PCPs, and applications to coding. *SIAM Journal on Computing*, 36(4):889–974, 2006.

6 J. C. Benaloh and J. Leichter. Generalized Secret Sharing and Monotone Functions. In *Proceedings of the International Cryptology Conference (CRYPTO)*, pages 27–35, 1988. doi:10.1007/0-387-34799-2_3.

7 M. L. Carmosino, R. Impagliazzo, V. Kabanets, and A. Kolokolova. Learning Algorithms from Natural Proofs. In *Proceedings of the Conference on Computational Complexity (CCC)*, pages 10:1–10:24, 2016. doi:10.4230/LIPIcs.CCC.2016.10.

8 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)*, pages 70:1–70:48, 2020. doi:10.4230/LIPIcs.ITCS.2020.70.

9 J. Cook and D. Moshkovitz. Tighter MA/1 circuit lower bounds from verifier efficient PCPs for PSPACE. Technical Report TR22-014, ECCC, 2022.

10 I. Dinur, E. Fischer, G. Kindler, R. Raz, and S. Safra. PCP characterizations of NP: Toward a polynomially-small error-probability. *Computational Complexity*, 20(3):413–504, 2011.

11 I. Dinur and P. Harsha. Composition of low-error 2-query PCPs using decodable PCPs. In *Proc. 50th IEEE Symp. on Foundations of Computer Science*, pages 472–481, 2009.

12 I. Dinur, P. Harsha, and G. Kindler. Polynomially low error pcps with polyloglog n queries via modular composition. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *Proc. 47th ACM Symp. on Theory of Computing*, pages 267–276. ACM, 2015.

13 I. Dinur and D. Steurer. Analytical approach to parallel repetition. In *Proc. 46th ACM Symp. on Theory of Computing*, 2014.

14 U. Feige, S. Goldwasser, L. Lovasz, S. Safra, and M. Szegedy. Interactive proofs and the hardness of approximating cliques. *Journal of the ACM*, 43(2):268–292, 1996.

15 J. Håstad. Some optimal inapproximability results. *Journal of the ACM*, 48(4):798–859, 2001.

16 A. Healy, S. P. Vadhan, and E. Viola. Using Nondeterminism to Amplify Hardness. *SIAM J. Comput.*, 35(4):903–931, 2006. doi:10.1137/S0097539705447281.

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

18 S. Hirahara. NP-Hardness of Learning Programs and Partial MCSP. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 968–979, 2022. doi:10.1109/FOCS54457.2022.00095.

19 S. Hirahara and O. Watanabe. Limits of Minimum Circuit Size Problem as Oracle. In *Proceedings of the Conference on Computational Complexity (CCC)*, pages 18:1–18:20, 2016. doi:10.4230/LIPIcs.CCC.2016.18.

20 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)*, pages 236–245, 2015. doi:10.4230/LIPIcs.FSTTCS.2015.236.

21 J. Holmerin and S. Khot. A new PCP outer verifier with applications to homogeneous linear equations and max-bisection. In *Proc. 36th ACM Symp. on Theory of Computing*, pages 11–20, 2004.

22 R. Ilango. Constant Depth Formula and Partial Function Versions of MCSP are Hard. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 424–433, 2020. doi:10.1109/FOCS46700.2020.00047.

23 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)*, pages 220–229, 1997. doi:10.1145/258533.258590.

24 M. Ito, A. Saito, and T. Nishizeki. Multiple Assignment Scheme for Sharing Secret. *J. Cryptol.*, 6(1):15–20, 1993. doi:10.1007/BF02620229.

25 Z. Ji, A. Natarajan, T. Vidick, J. Wright, and H. Yuen. MIP* = RE. Submitted, 2020.

26 V. Kabanets and J. Cai. Circuit minimization problem. In *Proceedings of the Symposium on Theory of Computing (STOC)*, pages 73–79, 2000. doi:10.1145/335305.335314.

27 J. Katz and L. Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *Proc. 32nd ACM Symp. on Theory of Computing*, pages 80–86, 2000.

28 S. Khot and O. Regev. Vertex cover might be hard to approximate to within 2-epsilon. *Journal of Computer and System Sciences*, 74(3):335–349, 2008.

29 S. Khot and N. K. Vishnoi. The unique games conjecture, integrality gap for cut problems and embeddability of negative type metrics into l_1. In *Proc. 46th IEEE Symp. on Foundations of Computer Science*, pages 53–62, 2005.

30 J. Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract. In *Proc. 24th ACM Symp. on Theory of Computing*, pages 723–732, 1992.

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

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

33 Y. Liu and R. Pass. On One-way Functions and Kolmogorov Complexity. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 1243–1254, 2020. doi:10.1109/FOCS46700.2020.00118.

34 D. Moshkovitz and R. Raz. Two query PCP with sub-constant error. *Journal of the ACM*, 57(5), 2010.

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

36 I. C. Oliveira and R. Santhanam. Hardness Magnification for Natural Problems. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 65–76, 2018.

37 C. Papadimitriou and M. Yannakakis. Optimization, approximation and complexity classes. *Journal of Computer and System Sciences*, 43:425–440, 1991.

38 O. Paradise. Smooth and strong pcps. *Comput. Complex.*, 30(1):1, 2021.

39 O. Reingold, S. P. Vadhan, and A. Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders and extractors. *Annals of Mathematics*, 155(1):157–187, 2002.

40 H. Ren, R. Santhanam, and Z. Wang. On the Range Avoidance Problem for Circuits. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 640–650, 2022. doi:10.1109/FOCS54457.2022.00067.

41 M. Saks and R. Santhanam. Circuit Lower Bounds from NP-Hardness of MCSP Under Turing Reductions. In *Proceedings of the Computational Complexity Conference (CCC)*, pages 26:1–26:13, 2020. doi:10.4230/LIPIcs.CCC.2020.26.

42 R. Santhanam. Pseudorandomness and the Minimum Circuit Size Problem. In *Proceedings of the Innovations in Theoretical Computer Science Conference (ITCS)*, pages 68:1–68:26, 2020. doi:10.4230/LIPIcs.ITCS.2020.68.

43 R. Shaltiel and C. Umans. Pseudorandomness for Approximate Counting and Sampling. *Computational Complexity*, 15(4):298–341, 2006. doi:10.1007/s00037-007-0218-9.

44 L. Trevisan. Extractors and pseudorandom generators. *J. ACM*, 48(4):860–879, 2001. doi:10.1145/502090.502099.

45 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*, 15(6):558–562, 1974.

46 D. Uhlig. Networks Computing Boolean Functions for Multiple Input Values. In *Poceedings of the London Mathematical Society Symposium on Boolean Function Complexity*, pages 165–173, USA, 1992. Cambridge University Press.

47 S. P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1-3):1–336, 2012.

48 I. Wegener. *The complexity of Boolean functions*. Wiley-Teubner, 1987. URL: http://ls2-www.cs.uni-dortmund.de/monographs/bluebook/.

49 R. Williams. Improving exhaustive search implies superpolynomial lower bounds. *SIAM Journal on Computing*, 42:231–240, 2010.