Range Avoidance, Remote Point, and Hard Partial Truth Table via Satisfying-Pairs Algorithms

Yeyuan Chen
yychen9961@gmail.com
Xi'an Jiaotong University
Xi'an, China

Yizhi Huang
huangyizhi01@gmail.com
IIIS, Tsinghua University
Beijing, China

Jiatu Li
ljt714285@gmail.com
IIIS, Tsinghua University
Beijing, China

Hanlin Ren
h4n1in.r3n@gmail.com
University of Oxford
Oxford, United Kingdom

ABSTRACT

The *range avoidance problem*, denoted as $\mathscr{C}$-Avoid, asks to find a non-output of a given $\mathscr{C}$-circuit $C : \{0,1\}^n \to \{0,1\}^\ell$ with stretch $\ell > n$. This problem has recently received much attention in complexity theory for its connections with circuit lower bounds and other explicit construction problems. Inspired by the Algorithmic Method for circuit lower bounds, Ren, Santhanam, and Wang (FOCS'22) established a framework to design $\mathsf{FP}^{\mathsf{NP}}$ algorithms for $\mathscr{C}$-Avoid via *slightly non-trivial* data structures related to $\mathscr{C}$. However, a major drawback of their approach is the lack of unconditional results even for $\mathscr{C} = \mathsf{AC}^0$.

In this work, we present the first unconditional $\mathsf{FP}^{\mathsf{NP}}$ algorithm for ACC^0-Avoid. Indeed, we obtain $\mathsf{FP}^{\mathsf{NP}}$ algorithms for the following stronger problems:

(ACC^0-Remote-Point). Given $C : \{0,1\}^n \to \{0,1\}^\ell$ for some $\ell = \text{quasi-poly}(n)$ such that each output bit of C is computed by a quasi-poly(n)-size $\mathsf{AC}^0[m]$ circuit, we can find some $y \in \{0,1\}^\ell$ in $\mathsf{FP}^{\mathsf{NP}}$ such that for every $x \in \{0,1\}^n$, the relative Hamming distance between y and $C(x)$ is at least $1/2 - 1/\text{poly}(n)$. This problem is the "average-case" analogue of ACC^0-Avoid.

(ACC^0-Partial-AvgHard). Given $x_1, \ldots, x_\ell \in \{0,1\}^n$ for some $\ell = \text{quasi-poly}(n)$, we can compute ℓ bits $y_1, \ldots, y_\ell \in \{0,1\}$ in $\mathsf{FP}^{\mathsf{NP}}$ such that for every $2^{\log^c(n)}$-size ACC^0 circuit C, $\Pr_i[C(x_i) \neq y_i] \geq 1/2 - 1/\text{poly}(n)$, where $c = O(1)$. This problem generalises the strong average-case circuit lower bounds against ACC^0 in a different way.

Our algorithms can be seen as natural generalisations of the best known almost-everywhere average-case lower bounds against ACC^0 circuits by Chen, Lyu, and Williams (FOCS'20). Note that both problems above have been studied prior to our work, and no $\mathsf{FP}^{\mathsf{NP}}$ algorithm was known even for weak circuit classes such as GF(2)-linear circuits and DNF formulas.

Our results follow from a strengthened algorithmic method: slightly non-trivial algorithms for the *Satisfying-Pairs* problem for $\mathscr{C}$ implies $\mathsf{FP}^{\mathsf{NP}}$ algorithms for $\mathscr{C}$-Avoid (as well as $\mathscr{C}$-Remote-Point and $\mathscr{C}$-Partial-AvgHard). Here, given $\mathscr{C}$-circuits $\{C_i\}$ and inputs $\{x_j\}$, the $\mathscr{C}$-Satisfying-Pairs problem asks to (approximately) count the number of pairs (i, j) such that $C_i(x_j) = 1$.

A technical contribution of this work is a construction of a *short, smooth, and rectangular PCP of Proximity* that combines two previous PCP constructions, which may be of independent interest. It serves as a key tool that allows us to generalise the framework for Avoid to the average-case scenarios.

CCS CONCEPTS

• **Theory of computation → Circuit complexity**.

KEYWORDS

circuit complexity, explicit constructions, range avoidance, remote point problem, satisfying pairs problem

ACM Reference Format:
Yeyuan Chen, Yizhi Huang, Jiatu Li, and Hanlin Ren. 2023. Range Avoidance, Remote Point, and Hard Partial Truth Table via Satisfying-Pairs Algorithms. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing (STOC '23), June 20–23, 2023, Orlando, FL, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3564246.3585147

1 INTRODUCTION

Proving unconditional lower bounds for non-uniform circuits is one of the grand challenges in theoretical computer science, with the holy grail of proving $\mathsf{NP} \not\subseteq \mathsf{P}_{/\text{poly}}$. Unfortunately, progress in unconditional circuit lower bounds has been slow, and the best lower bound for any explicit function against general circuits is only slightly above $3n$ [28, 37]. A long-standing, and somewhat embarrassing, open problem is to find any language in $\mathsf{EXP}^{\mathsf{NP}}$ (exponential time with an NP oracle) that cannot be computed by polynomial-size circuits. It seems unlikely that $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathsf{P}_{/\text{poly}}$, but we appear to be very far from ruling out this possibility.

To add more embarrassment, it has been known since 1949 [46] that *most* Boolean functions over n inputs require circuits of size $\Omega(2^n/n)$. 70 years later, we still struggle to spell out even a single such function from a plethora of them! It turns out that circuit lower bounds are not alone, and the difficulty of "finding hay in a haystack" ([10, Chapter 21]) is a general phenomenon in theoretical

STOC '23, June 20–23, 2023, Orlando, FL, USA

ACM ISBN 978-1-4503-9913-5/23/06…$15.00
https://doi.org/10.1145/3564246.3585147

computer science. For example, most graphs are Ramsey graphs [27] and most matrices are rigid matrices [49], but it remains major open problems to explicitly construct Ramsey graphs and rigid matrices with good parameters [6, 14, 16, 43].

Our lack of progress in such explicit construction problems suggests the necessity of a systematic study of their difficulty. As a first step towards building a complexity theory for explicit construction problems, Korten [36] studied the complexity class APEPP defined in [35], and argued that this is the complexity class that corresponds to explicit construction problems. APEPP is the class of total search problems that are polynomial-time reducible to the following problem:

Problem 1.1 (Range Avoidance Problem, denoted as Avoid). Given the description of a circuit $C : \{0,1\}^n \to \{0,1\}^\ell$, where $\ell > n$, output any string $y \in \{0,1\}^\ell$ that is not in the range of C. That is, for every $x \in \{0,1\}^n$, $C(x) \neq y$.

The existence of such y follows from the *dual weak pigeonhole principle*: if we throw 2^n pigeons into 2^ℓ holes, where $\ell \geq n+1$, then there is an empty hole. Thus Avoid is a *total* search problem. Moreover, a random string $y \in \{0,1\}^\ell$ is a valid solution w.p. $1 - 2^{n-\ell} \geq 1/2$, thus there is a trivial randomised algorithm for Avoid. Therefore, the focus is to design *deterministic* algorithms for Avoid.

The following is a good example of how Avoid captures the complexity of explicit constructions:

Example 1.2 ([36, Section 3.1]). Proving circuit lower bounds can be rephrased as solving the following total search problem, denoted as Hard: On input 1^N where $N = 2^n$, output the truth table of a function $f : \{0,1\}^n \to \{0,1\}$ that cannot be computed by circuits of size s (say $s = 2^{n/2}$).

Let $\mathsf{TT} : \{0,1\}^{O(s \log s)} \to \{0,1\}^{2^n}$ be the circuit that takes as input the description of a size-s circuit and outputs the truth table of this circuit. (The circuit TT is sometimes called the *truth table* generator, hence the name TT.) If we could solve Avoid on the particular instance TT, we would find a truth table $tt \in \{0,1\}^{2^n}$ without size-s circuits, thereby proving a circuit lower bound. It follows that Hard polynomial-time reduces to Avoid, and thus Hard $\in$ APEPP.

More precisely, solving Avoid for TT in polynomial time is equivalent to proving a circuit lower bound for E, and solving Avoid for TT in $\mathsf{FP}^{\mathsf{NP}}$ is equivalent to proving a circuit lower bound for E^{NP}.

1.1 Range Avoidance for Restricted Circuit Classes

In a recent paper, Ren, Santhanam, and Wang [45] suggested studying the range avoidance problem for restricted circuit classes. Let $\mathscr{C}$ be a circuit class and $\ell := \ell(n) > n$ be a stretch function. Consider the following problem:

Problem 1.3 ($\mathscr{C}$-Avoid). Given the description of a circuit $C : \{0,1\}^n \to \{0,1\}^{\ell(n)}$, where each output bit of C is a $\mathscr{C}$ circuit, output any string $y \in \{0,1\}^{\ell(n)}$ that is not in the range of C. That is, for every $x \in \{0,1\}^n$, $C(x) \neq y$.

There are lots of reasons for studying the problem $\mathscr{C}$-Avoid, but we only mention one of them here. Many interesting explicit construction problems reduce to $\mathscr{C}$-Avoid for restricted circuit classes $\mathscr{C}$ and (sometimes) large stretch functions ℓ. For example:

- For any "nice" circuit class $\mathscr{C}$, the problem of proving circuit lower bounds against $\mathscr{C}$ can be reduced to $\mathscr{C}$-Avoid via the truth table generator in Example 1.2, where the input of the truth table generator is replaced by a $\mathscr{C}$ circuit (instead of a general circuit).
- Guruswami, Lyu, and Wang [32] showed that the problem of finding rigid matrices and optimal binary linear codes can be reduced to NC^1-Avoid. By a further result in [45], these problems also reduce to NC^0_4-Avoid (i.e., each output bit depends on at most 4 input bits). A recent work [30] showed that the problem of finding rigid matrices can even be reduced to NC^0_3-Avoid.

In general, for any explicit construction problem Π, we can identify a circuit class $\mathscr{C}$ that is as "simple" as possible, as well as a stretch function $\ell(n)$ that is as large as possible, such that Π reduces to $\mathscr{C}$-Avoid with stretch $\ell(n)$. The hope is that by making progress on the range avoidance problem for restricted circuits and by optimising the reduction (i.e., optimising $\mathscr{C}$ and $\ell(n)$), we could solve many explicit construction problems systematically.

An "Algorithmic Method" for range avoidance. Inspired by the Algorithmic Method for proving circuit lower bounds (e.g. [17, 21, 24, 39, 50, 51]), Ren, Santhanam, and Wang [45] proposed a framework to solve $\mathscr{C}$-Avoid in $\mathsf{FP}^{\mathsf{NP}}$ using the following data structure problem:

Problem 1.4 (Hamming Weight Estimation). Let $\mathscr{C}$ be a circuit class and $\ell := \ell(n)$ be a stretch function. The data structure problem has two phases:

(Preprocessing) Given description of a circuit $C : \{0,1\}^n \to \{0,1\}^\ell$, where each output bit of C is a $\mathscr{C}$ circuit, we need to preprocess the circuit in P^{NP} (i.e., in polynomial time with an NP oracle) and output a data structure $\mathsf{DS} \in \{0,1\}^{\mathrm{poly}(\ell)}$.

(Query) Given an input x and oracle access (i.e., random access) to DS, we need to estimate the Hamming weight of $C(x)$ in "non-trivial" time, i.e., deterministic $\ell/\log^{\omega(1)} \ell$ time.

It was shown in [45] that for "typical" circuit classes[1] $\mathscr{C}$, a non-trivial data structure for the Hamming Weight Estimation problem for $\mathscr{C}$ implies an $\mathsf{FP}^{\mathsf{NP}}$ algorithm for $\mathscr{C}$-Avoid.

One drawback of [45] is that their framework does not imply new unconditional algorithms for range avoidance.[2] For comparison, the original Algorithmic Method has made significant progress on proving *unconditional* circuit lower bounds that we do not know how to prove otherwise. One motivation for the current paper is to address this drawback by designing new and unconditional range avoidance algorithms via the Algorithmic Method. In particular,

[1] In the literature, a circuit class is said to be typical if it satisfies certain natural closure properties. In this paper, a *typical* circuit class $\mathscr{C}$ should contain the identity circuit and be closed under negations and projections. More precisely, (1) $\mathscr{C}$ contains every circuit that always outputs its input; (2) for any $\mathscr{C}$ circuit C of size s and projection proj, both $\neg C$ and $C \circ \mathrm{proj}$ have $\mathscr{C}$ circuits of size $\mathrm{poly}(s)$, and the descriptions of these circuits can be computed in $\mathrm{poly}(s)$ time.

[2] Actually, [45] provided an unconditional range avoidance algorithm for de Morgan formulas with non-trivial parameters. Subsequently, [32] improved this result by using simpler techniques and achieving better parameters; in particular, the algorithm in [32] does not require the Algorithmic Method.

can we solve ACC^0-Avoid with parameters that match the circuit lower bounds in [21]?

1.2 The Remote Point Problem

The Algorithmic Method is extremely good at proving average-case circuit lower bounds [20–22]. Therefore, it is natural to wonder if there is an "average-case analogue" of [45].

For two strings $x, y \in \{0,1\}^n$, their *relative Hamming distance* is defined as the fraction of indices where x and y differ, formally $\delta(x,y) := \frac{1}{n}|\{i \in [n] : x_i \neq y_i\}|$. The "average-case analogue" of the range avoidance problem is the following problem:

Problem 1.5 ($\mathscr{C}$-Remote-Point)**.** Given the description of a circuit $C : \{0,1\}^n \to \{0,1\}^\ell$ and a parameter $\delta > 0$, where each output bit of C is a $\mathscr{C}$ circuit, output any string $y \in \{0,1\}^\ell$ that is δ-far from the range of C. That is, for every $x \in \{0,1\}^n$, $\delta(C(x), y) \geq \delta$.

By Chernoff bound, if $\delta < 1/2 - c\sqrt{n/\ell}$ for some absolute constant $c > 0$, then a random length-ℓ string is a valid solution for Remote-Point w.h.p. Therefore, the challenge is to find deterministic algorithms for Remote-Point.

It is not hard to see that $\mathscr{C}$-Remote-Point for the truth table generator TT corresponds to average-case circuit lower bounds. In particular, the regime where δ is a small constant corresponds to proving "weak" average-case lower bounds (e.g. [17, 25]), and the regime where δ is close to 1/2 (say, $\delta = 1/2 - 1/n$) corresponds to proving "strong" average-case lower bounds (e.g. [21, 22]).[3]

The remote point problem was discussed in [35]. Indeed, an important special case of the problem has been studied by Alon, Panigrahy, and Yekhanin [9], namely the case that C is a linear transformation over GF(2). In other words, we are given a linear code $C : \{0,1\}^n \to \{0,1\}^\ell$ and we want to find a string far from every codeword. They introduced this problem as an intermediate step towards constructing rigid matrices. In this paper, we call this special case XOR-Remote-Point.

It is already quite hard to solve this special case deterministically. Alon, Panigrahy, and Yekhanin [9] designed a polynomial-time algorithm for XOR-Remote-Point when $\ell > 2n$ and $\delta = O(\log n/n)$. For slightly larger δ, say $\delta = 0.1$, no deterministic algorithm is known even with an NP oracle. Arvind and Srinivasan [11] showed that for certain parameters, a polynomial-time algorithm for XOR-Remote-Point implies a polynomial-time algorithm for AC^0-Partial-Hard (defined later in Section 1.3).

1.3 Hard Partial Truth Tables

We also consider the following problem that generalises the task of proving circuit lower bounds (in a different way from Avoid and Remote-Point):

Problem 1.6 (Hard Partial Truth Tables against $\mathscr{C}$, denoted as $\mathscr{C}$-Partial-Hard)**.** Given a list of input strings $z_1, z_2, \ldots, z_\ell \in \{0,1\}^n$ and a parameter s, find a list of output bits $b_1, b_2, \ldots, b_\ell \in \{0,1\}$ such that the partial function defined by $\{(z_i, b_i)\}_{i \in [\ell]}$ cannot be computed by $\mathscr{C}$ circuits of size s. In other words, for every size-s $\mathscr{C}$ circuit C, there exists an index $i \in [\ell]$ such that $C(z_i) \neq b_i$.

It is easy to see that $\mathscr{C}$-Partial-Hard generalises the problem of proving circuit lower bounds against $\mathscr{C}$. Indeed, if we take $\ell := 2^n$ and $z_1, z_2, \ldots, z_\ell$ be an enumeration of length-n strings, then $\mathscr{C}$-Partial-Hard becomes exactly the problem of proving circuit lower bounds against $\mathscr{C}$. It is also easy to see that when $\ell > O(s \log s)$, this problem is in APEPP: given the input $(z_1, z_2, \ldots, z_\ell)$, we can construct a circuit $\mathsf{TT}' : \{0,1\}^{O(s\log s)} \to \{0,1\}^\ell$ which takes the description of a $\mathscr{C}$ circuit C as input, and outputs the concatenation of $C(z_1), C(z_2), \ldots, C(z_\ell)$. Finding a non-output of TT' is equivalent to finding a solution of $\mathscr{C}$-Partial-Hard, thus this problem reduces to Avoid.

This problem was introduced by Arvind and Srinivasan [11] under the name "circuit lower bounds with help functions." Let $h_1, h_2, \ldots, h_n : \{0,1\}^m \to \{0,1\}$ denote a sequence of *help functions*, $\mathscr{C}$ be a circuit class, and $s \in \mathbb{N}$ be a size parameter. The goal is to construct the truth table of a function $f : \{0,1\}^m \to \{0,1\}$ that is hard to compute for size-s $\mathscr{C}$ circuits, even when the circuit has access to these help functions. Formally, for any size-s circuit $C : \{0,1\}^n \to \{0,1\}$, there exists an input $x \in \{0,1\}^m$ such that

$$C(h_1(x), h_2(x), \ldots, h_n(x)) \neq f(x).$$

This problem is equivalent to Partial-Hard with $\ell = 2^m$ inputs of length n, namely for every $x \in \{0,1\}^m$, there is an input $h_1(x) \circ h_2(x) \circ \cdots \circ h_n(x) \in \{0,1\}^n$ in the Partial-Hard instance.

This problem appears to be very hard. Neither [11] nor we are aware of an efficient deterministic solution for $\mathscr{C} = AC^0$ with (say) $\ell, s \in$ quasi-poly(n). That is, although exponential-size lower bounds against AC^0 are known [2, 29, 34, 55], we do not have any idea about how to prove such a lower bound for partial functions. Even when $\mathscr{C}$ is the class of *polynomial-size DNF*, to the best of our knowledge, there is no known deterministic algorithm for $\mathscr{C}$-Partial-Hard.

Besides being a natural problem itself, $\mathscr{C}$-Partial-Hard also arises when we study the closure of non-uniform complexity classes (under reductions). Recall that AC^0 denotes the class of languages computable by a *non-uniform* family of polynomial-size constant-depth circuits; in particular, AC^0 contains undecidable languages such as unary versions of the halting problem. A language L Turing-reduces to some language in AC^0 if and only if $L \in \mathsf{P}_{/\mathrm{poly}}$ [42], thus proving $\mathsf{EXP} \not\leq_T^p AC^0$ is likely beyond current techniques. But what about *mapping reducibility*? Can we show that $\mathsf{EXP} \not\leq_m^p AC^0$? It turns out that a deterministic algorithm for AC^0-Partial-Hard implies that $\mathsf{EXP} \not\leq_m^p AC^0$ [11, Theorem 5]. Of course, there is nothing special with AC^0, and it can be replaced by other non-uniform classes. Therefore, $\mathscr{C}$-Partial-Hard sheds light on ruling out many-one reducibility of EXP (and other complexity classes) to non-uniform classes.

We also define an average-case version of $\mathscr{C}$-Partial-Hard, which is equivalent to proving average-case lower bounds with help functions.

Problem 1.7 (Average-Case Hard Partial Truth Tables against $\mathscr{C}$, denoted as $\mathscr{C}$-Partial-AvgHard)**.** Given a list of input strings $z_1, z_2, \ldots, z_\ell \in \{0,1\}^n$ and parameters s, δ, find a list of output bits $b_1, b_2, \ldots, b_\ell \in \{0,1\}$ such that the partial function defined by

[3]Typically, a strong average-case lower bound states that certain problems cannot be $(1/2 + 1/s)$-approximated by size-s circuits. Suppose $\mathsf{TT} : \{0,1\}^n \to \{0,1\}^\ell$ is the truth table generator, then n is roughly the size of the circuit (i.e., $n \approx s$). In this regard, strong average-case circuit lower bounds correspond to Remote-Point where $\delta = 1/2 - 1/n$.

$\{(z_i, b_i)\}_{i\in[\ell]}$ is δ-far from being computable by $\mathscr{C}$ circuits of size s. In other words, for every size-s $\mathscr{C}$ circuit C, there are at least $\delta\ell$ indices $i \in [\ell]$ such that $C(z_i) \neq b_i$.

2 OUR RESULTS

We now briefly describe our main results. Interested readers are referred to the full version of the paper for more details.

2.1 Explicit Constructions from Satisfying-Pairs Algorithms

We start with the following observation: In the framework of solving Avoid via the Algorithmic Method [45], the data structure for Problem 1.4 does not need to be *online*. Instead, it suffices to design a data structure that preprocesses a circuit $C : \{0,1\}^n \to \{0,1\}^\ell$, receives a *batch* of inputs $x_1, x_2, \ldots, x_M$, and estimates the Hamming weight of each $C(x_i)$ in *non-trivial total time*, i.e., $\ell M/\log^{\omega(1)}(\ell M)$ time. Moreover, we observe that it is not even necessary to estimate the individual Hamming weights $C(x_i)$; it suffices to estimate the *average* Hamming weight of $C(x_i)$ for $i \in [M]$. Indeed, we arrive at the following problem called *Satisfying Pairs.*

Problem 2.1 ($\mathscr{C}$-Satisfying-Pairs)**.** Let N, M, s, n be parameters. Given (single-output) $\mathscr{C}$ circuits $C_1, \ldots, C_N : \{0,1\}^n \to \{0,1\}$ of size s and input strings $x_1, x_2, \ldots, x_M \in \{0,1\}^n$, compute or estimate

$$\Pr_{i\leftarrow[M],j\leftarrow[N]}[C_j(x_i) = 1]. \tag{1}$$

We define the decisional and counting versions of the satisfying pairs problem as follows.

- Gap$_\delta$-$\mathscr{C}$-Satisfying-Pairs is the problem of distinguishing between (1) = 1 and (1) < $1 - \delta$;
- Approx$_\varepsilon$-$\mathscr{C}$-Satisfying-Pairs is the problem of estimating Eq. (1) within additive error ε;
- $\mathscr{C}$-Satisfying-Pairs is the problem of deciding whether Eq. (1) > 0;
- #$\mathscr{C}$-Satisfying-Pairs is the problem of exactly computing Eq. (1).

We consider the regime where the input length n and the circuit size s are much smaller than N and M. In such case, a deterministic algorithm for $\mathscr{C}$-Satisfying-Pairs is said to be *non-trivial* if it runs in time $NM/\log^{\omega(1)}(NM)$.[4]

Remark 2.2. The circuit-analysis problems that arise in the Algorithmic Method[5] are special cases of Satisfying Pairs problems. For instance, we can reduce #SAT of the circuit C to #Satisfying-Pairs with $N = 2^{n/2}$ and $M = 2^{n/2}$, where the inputs $(x_1, x_2, \ldots, x_M)$ consists of all strings of length $n/2$, and the circuits are $\{C_y : y \in \{0,1\}^{n/2}\}$, where $C_y(x) := C(x \circ y)$. Similarly, $\mathscr{C}$-Satisfying-Pairs corresponds to $\mathscr{C}$-SAT, Gap-$\mathscr{C}$-Satisfying-Pairs corresponds to $\mathscr{C}$-GapUNSAT, and Approx-$\mathscr{C}$-Satisfying-Pairs corresponds to $\mathscr{C}$-CAPP.

2.1.1 Range Avoidance from Satisfying-Pairs.

By plugging the observation above in [45], we prove that non-trivial algorithms for Satisfying-Pairs imply $\mathsf{FP}^{\mathsf{NP}}$ algorithms for Avoid.

Theorem 2.3 (Informal). *Let $\mathscr{C}$ be a typical circuit class and $\mathscr{C}' := \mathsf{OR}_2 \circ \mathscr{C}$.[6] Suppose that there is a non-trivial algorithm for* Approx$_\varepsilon$*-$\mathscr{C}'$-Satisfying-Pairs for every constant $\varepsilon > 0$, then $\mathscr{C}$-Avoid with certain parameters can be solved in* $\mathsf{FP}^{\mathsf{NP}}$.

This informal theorem hides the trade-off between the parameters of $\mathscr{C}$-Avoid and $\mathscr{C}'$-Satisfying-Pairs. In general, to solve $\mathscr{C}$-Avoid with smaller stretch ℓ (with respect to the input length n), we need to have non-trivial algorithms for $\mathscr{C}'$-Satisfying-Pairs where the number of inputs N and the number of circuits M are smaller with respect to the circuit size s and the input length n. We highlight two typical choices of parameters of Theorem 2.3 as follows.

Corollary 2.4. *There is a constant $\varepsilon > 0$ such that the following holds. Let $\mathscr{C}$ be a typical circuit class, $\mathscr{C}' := \mathsf{OR}_2 \circ \mathscr{C}$, and $s = s(n)$ be a non-decreasing size parameter.*

- *Suppose that there is a non-trivial algorithm for* Approx$_\varepsilon$*-$\mathscr{C}'$-Satisfying-Pairs for $N = n^{1+\Omega(1)}$ $\mathscr{C}'$-circuits of size $2s(n)$ and $M = n^{1+\Omega(1)}$ inputs of length n. Then there is an* $\mathsf{FP}^{\mathsf{NP}}$ *algorithm for $\mathscr{C}$-Avoid with stretch ℓ and circuit size s,[7] for some $\ell = n^{1+\Omega(1)}$.*
- *Suppose that there is a non-trivial algorithm for* Approx$_\varepsilon$*-$\mathscr{C}'$-Satisfying-Pairs for $N =$ quasi-poly(n) $\mathscr{C}'$-circuits of size $2s(n)$ and $M =$ quasi-poly(n) inputs of length n. Then there is an* $\mathsf{FP}^{\mathsf{NP}}$ *algorithm for $\mathscr{C}$-Avoid with stretch ℓ and circuit size s, for some $\ell =$ quasi-poly(n).*

2.1.2 Remote Point from Satisfying-Pairs.

With the help of smooth and rectangular PCPPs (see Section 2.3) and a list-decodable code with linear-sum decoder from [21], we show that non-trivial algorithms for Satisfying-Pairs imply Remote-Point algorithms in $\mathsf{FP}^{\mathsf{NP}}$.

Theorem 2.5 (Informal). *Let $\mathscr{C}$ be a typical circuit class and $\mathscr{C}' := \mathsf{AND}_{O(1)} \circ \mathscr{C}$. Suppose that there is a non-trivial algorithm for* Approx$_\varepsilon$*-$\mathscr{C}'$-Satisfying-Pairs for every constant $\varepsilon > 0$, then $\mathscr{C}$-Remote-Point with certain parameters can be solved in* $\mathsf{FP}^{\mathsf{NP}}$.

In particular: suppose for every constant $\varepsilon > 0$, there is a non-trivial algorithm for Approx$_\varepsilon$*-$\mathscr{C}'$-Satisfying-Pairs for $N =$ quasi-poly(n) $\mathscr{C}'$-circuits of size $O(s)$ and $M =$ quasi-poly(n) inputs of length n; then for some stretch function $\ell =$ quasi-poly(n), there is an* $\mathsf{FP}^{\mathsf{NP}}$ *algorithm for $\mathscr{C}$-Remote-Point that takes as input a circuit $C : \{0,1\}^n \to \{0,1\}^\ell$ where each output bit of C is a $\mathscr{C}$-circuit of size s, and outputs a y that is 0.49-far from* Range(C).

Our framework provides a Remote-Point algorithm for the regime corresponding to "strong average-case lower bounds", i.e., the distance between the output y and Range(C) is close to 1/2.

[4]Analogous to the preprocessing phase in Problem 1.4, one could also add a P^{NP}-preprocessing phase that sees the circuits but not the inputs. Algorithms with such preprocessing phase would still imply our results, but the Satisfying-Pairs algorithms in this paper do not need this preprocessing phase.

[5]The definitions of circuit-analysis problems such as SAT or CAPP can be found in Lijie Chen's PhD thesis [18].

[6]Here, $\mathsf{OR}_d \circ \mathscr{C}$ refers to the composition of a single fan-in-d OR gate being the output gate of the circuit and (at most) d $\mathscr{C}$ circuits feeding the top OR gate.

[7]Note that the circuit size parameter of $\mathscr{C}$-Avoid refers to the maximum circuit size of each output bit of $C : \{0,1\}^n \to \{0,1\}^\ell$, instead of the total circuit size of C.

In fact, the distance can be as large as $1/2 - 1/\mathrm{poly}(n)$ given an Approx-$\mathscr{C}$-Satisfying-Pairs algorithm with small enough error.

Note that the stretch for $\mathscr{C}$-Remote-Point that we can solve in $\mathsf{FP}^{\mathsf{NP}}$ depends on both the parameters of the satisfying pairs algorithms and the rate of the linear-sum list-decodable code. Since the code from [21] has a quasi-polynomial rate, our framework cannot solve Remote-Point with a stretch smaller than quasi-polynomial. It is an interesting open problem to improve the stretch of Remote-Point that can be solved by our framework, possibly by designing new linear-sum decodable codes with a better rate; see, e.g., [20].

2.1.3 Hard Partial Truth Table from Satisfying-Pairs. Similar to the frameworks for Avoid and Remote-Point, we can solve the problems Partial-Hard and Partial-AvgHard via non-trivial algorithms for Satisfying-Pairs.

Theorem 2.6 (Informal). *Let $\mathscr{C}$ be a typical circuit class.*

- *Suppose that there is a non-trivial algorithm for $\mathsf{Approx}_\varepsilon$-$\mathscr{C}'$-Satisfying-Pairs for every $\varepsilon > 0$ and $\mathscr{C}' := \mathsf{OR}_2 \circ \mathscr{C}$, then $\mathscr{C}$-Partial-Hard with certain parameters can be solved in $\mathsf{FP}^{\mathsf{NP}}$.*
- *Suppose that there is a non-trivial algorithm for $\mathsf{Approx}_\varepsilon$-$\mathscr{C}''$-Satisfying-Pairs for every $\varepsilon > 0$ and $\mathscr{C}'' := \mathsf{AND}_{O(1)} \circ \mathscr{C}$, then $\mathscr{C}$-Partial-AvgHard with certain parameters can be solved in $\mathsf{FP}^{\mathsf{NP}}$.*

These results are proved using essentially the same approach as the framework for Avoid and Remote-Point; therefore, the trade-off between parameters for Satisfying-Pairs and Partial-Hard (resp. Partial-AvgHard) is similar to that for Satisfying-Pairs and Avoid (resp. Remote-Point). We omit the details and refer the readers to the full version of the paper.

Remark 2.7. It is not surprising to have a unified framework for Avoid and Partial-Hard (as well as their average-case analogues Remote-Point and Partial-AvgHard), because they can be considered as the *dual* problem of each other. Let $\mathsf{Eval} : \{0,1\}^{O(s\log s)} \times \{0,1\}^n \to \{0,1\}$ be the circuit-evaluation function that takes a circuit C of size s and an input of length n, and outputs $C(x)$. We can interpret Avoid and Partial-Hard as follows:

- **(Avoid).** Given size-s circuits $C_1, C_2, \dots, C_\ell$, find $y_1, y_2, \dots, y_\ell \in \{0,1\}$ such that for every $x \in \{0,1\}^n$, there is an $i \in [\ell]$ such that $\mathsf{Eval}(C_i, x) \neq y_i$.
- **(Partial-Hard).** Given inputs $x_1, x_2, \dots, x_\ell \in \{0,1\}^n$, find $y_1, y_2, \dots, y_\ell \in \{0,1\}$ such that for every size-s circuit C, there is an $i \in [\ell]$ such that $\mathsf{Eval}(C, x_i) \neq y_i$.

Clearly, Avoid and Partial-Hard are essentially the same problem on the table $\mathsf{Eval}(\cdot,\cdot)$ with the rows and columns being exchanged.

2.2 Unconditional Results for Explicit Constructions

The seemingly marginal improvement of using non-trivial algorithms for Satisfying-Pairs instead of its online version Hamming Weight Estimation (see Problem 1.4) plays an important role in the design of unconditional $\mathsf{FP}^{\mathsf{NP}}$ algorithms for ACC^0-Remote-Point and ACC^0-Partial-Hard. This is because we can indeed design non-trivial algorithms for ACC^0-Satisfying-Pairs.

2.2.1 XOR-Remote-Point from XOR-Satisfying-Pairs. We start from a simpler case where the circuit class $\mathscr{C} = \mathsf{XOR}$, i.e., the circuit is an XOR of some of its input bits. Since an XOR circuit C can be represented by a vector $\vec{v} \in \{0,1\}^n$ such that $C(x) = \langle v, x\rangle \bmod 2$, #XOR-Satisfying-Pairs is nothing but the counting version of the *Orthogonal Vector* problem over $\mathbb{F}_2$, which admits a non-trivial algorithm [6, 15]. By combining this with Theorem 2.3, we obtain an unconditional $\mathsf{FP}^{\mathsf{NP}}$ algorithm for XOR-Remote-Point.[8]

Theorem 2.8 (XOR-Remote-Point $\in \mathsf{FP}^{\mathsf{NP}}$). *There is a constant $c_u \ge 1$ such that the following holds. Let $\varepsilon := \varepsilon(n) \ge 2n^{-c_u}$ be error parameter and $\ell := \ell(n) \ge 2^{\log^{c_u+5} n}$ be stretch, then there is an $\mathsf{FP}^{\mathsf{NP}}$ algorithm that takes as input a circuit $C : \{0,1\}^n \to \{0,1\}^\ell$, where each output bit of C is computed by an XOR gate, and outputs a string y that is $(1/2 - \varepsilon)$-far from $\mathsf{Range}(C)$.*

2.2.2 A Non-trivial Algorithm for ACC^0-Satisfying-Pairs. By adapting the technique introduced by Williams [54] to design non-trivial #SAT algorithms for ACC^0 circuits with an earlier quasi-polynomial size simulation of $\mathsf{SYM} \circ \mathsf{ACC}^0$ circuits by $\mathsf{SYM} \circ \mathsf{AND}$ circuits [3, 12], we can obtain a non-trivial algorithm for #ACC^0-Satisfying-Pairs, formally stated as follows.

Theorem 2.9. *For every constants m, ℓ, c, there is a constant $\varepsilon \in (0,1)$ such that the following holds. Let $n := 2^{\log^\varepsilon N}$ and $s := 2^{\log^c n}$. There is a deterministic algorithm running in $\tilde{O}((N/n)^2)$ time that given N strings $x_1, x_2, \dots, x_N \in \{0,1\}^n$ and N $\mathsf{AC}^0_\ell[m]$ circuits $C_1, C_2, \dots, C_N : \{0,1\}^n \to \{0,1\}$ of size s, outputs the number of pairs $(i,j) \in [N] \times [N]$ such that $C_i(x_j) = 1$.*

2.2.3 Explicit Constructions for ACC^0. The $\mathsf{FP}^{\mathsf{NP}}$ algorithm for ACC^0-Remote-Point and ACC^0-Partial-AvgHard follows from this algorithm together with Theorem 2.5 and Theorem 2.6.

Theorem 2.10 (ACC^0-Remote-Point $\in \mathsf{FP}^{\mathsf{NP}}$). *There is a constant $c_u \ge 1$ such that for every constant $d, m \ge 1$, there is a constant $c_{\mathsf{str}} := c_{\mathsf{str}}(d,m) \ge 1$, such that the following holds.*

Let $n < s(n) \le 2^{n^{o(1)}}$ be a size parameter, $\varepsilon := \varepsilon(n) \ge 2n^{-c_u}$ be an error parameter and $\ell := \ell(n) \ge 2^{\log^{c_{\mathsf{str}}} s}$ be a stretch function, then there is an $\mathsf{FP}^{\mathsf{NP}}$ algorithm that takes as input a circuit $C : \{0,1\}^n \to \{0,1\}^\ell$, where each output bit of C is computed by an $\mathsf{AC}^0_d[m]$ circuit of size s, and outputs a string y that is $(1/2-\varepsilon)$-far from $\mathsf{Range}(C)$.

Theorem 2.11 (ACC^0-Partial-AvgHard $\in \mathsf{FP}^{\mathsf{NP}}$). *There is a constant $c_u \ge 1$ such that for every constants $d, m \ge 1$, there is a constant $c_{\mathsf{str}} := c_{\mathsf{str}}(d,m) \ge 1$, such that the following holds.*

Let $n < s(n) \le 2^{n^{o(1)}}$ be a size parameter, $\varepsilon := \varepsilon(n) \ge 2n^{-c_u}$ be an error parameter and $\ell := \ell(n) \ge 2^{\log^{c_{\mathsf{str}}} s}$ be a stretch function, then there is an $\mathsf{FP}^{\mathsf{NP}}$ algorithm that given inputs $x_1, \dots, x_\ell \in \{0,1\}^n$, it outputs a string $y \in \{0,1\}^\ell$ such that for any $s(n)$-size $\mathsf{AC}_d[m]$ circuit C, y is $(1/2-\varepsilon)$-far from $C(x_1) \circ \cdots \circ C(x_\ell)$.

[8] The reduction from Remote-Point to Satisfying-Pairs has a small overhead on the circuit class (i.e. the upper $\mathsf{AND}_{O(1)}$ in Theorem 2.5). By a standard trick using Fourier analysis (see the full version of the paper, also see [24]), we can change the upper circuit class to be $\mathsf{XOR}_{O(1)}$ so that we only need to design Satisfying-Pairs algorithms for $\mathsf{XOR}_d \circ \mathsf{XOR} = \mathsf{XOR}$.

It is worth noting that the ACC^0-Remote-Point algorithm here recovers the best known almost-everywhere average-case circuit lower bounds against ACC^0 [21]. This is done by considering the special case where the input circuit is the truth table generator $\mathsf{TT} : \{0,1\}^{O(s \log s)} \to \{0,1\}^{2^n}$ that prints the truth table of a given ACC^0 circuit.

Corollary 2.12. *For every constant $d, m \geq 1$, there is an $\varepsilon > 0$ and a language $L \in \mathsf{E}^{\mathsf{NP}}$ such that L_n cannot be $(1/2 + 2^{-n^{\varepsilon}})$-approximated by $\mathsf{AC}^0_d[m]$ circuits of size $2^{n^{\varepsilon}}$, for all sufficiently large n.*

2.2.4 Lower Bounds on the Many-One Closure of ACC^0. Following the observation of Arvind and Srinivasan [11], the $\mathsf{FP}^{\mathsf{NP}}$ algorithm for ACC^0-Partial-AvgHard can be used to prove *unconditionally* that E^{NP} cannot be mapping reduced to languages decidable by small-size *non-uniform* families of ACC^0 circuits.[9] To the best of our knowledge, this is the first unconditional result on ruling out the mapping reducibility from uniform classes to non-trivial non-uniform classes.

Corollary 2.13. *Let $d, m \in \mathbb{N}$ be constants, $\mathsf{AC}^0_d[m]$ denote the class of languages computable by a* non-uniform *family of polynomial-size $\mathsf{AC}^0_d[m]$ circuits. Then, there is a language $L^{\mathrm{hard}} \in \mathsf{E}^{\mathsf{NP}}$ that does not have polynomial-time mapping reductions to any language in $\mathsf{AC}^0_d[m]$.*

2.3 A Smooth and Rectangular PCPs of Proximity

One of the main technical ingredients in our framework for the average-case construction problems (i.e. Remote-Point and Partial-AvgHard) is a PCP of Proximity (PCPP) that is *short*, *smooth*, and (almost) *rectangular*.

A PCPP verifier V for a language L provides a super-efficient probabilistic proof system for checking whether $x \in L$ or x is far from being in L. Given an input x and a proof π, the verifier with access to some random bits only probes constantly many bits of x and π. If $x \in L$, then it accepts with an appropriate proof π; if the relative Hamming distance between x and any $x' \in L$ is at least δ, then it rejects with constant probability regardless of the proof π. (The distance δ is called the *proximity parameter* of the PCPP.) In addition, our PCPP verifier is equipped with the following properties:

- **(Shortness).** For any language $L \in \mathsf{NTIME}[T(n)]$ such that $n \leq T(n) \leq 2^{\mathrm{poly}(n)}$, the PCPP proof for L has length $T(n) \cdot \mathrm{polylog}(T(n))$.
- **(Rectangularity).** The input and the proof are treated as matrices. Moreover, the queries of the verifier to the input and proof matrices can be done *rectangularly*, in the sense that there are a row verifier V_{row} and a column verifier V_{col} with (almost) independent random seeds that generate the row and column indices of the queries, respectively.
- **(Smoothness).** The queries of the verifier to the proof matrix are uniformly random. As a consequence, it means that the PCPP proof can tolerate errors in a correct proof.

[9] In fact, it suffices to have an $\mathsf{FP}^{\mathsf{NP}}$ algorithm for ACC^0-Partial-Hard (which is a trivial consequence of an $\mathsf{FP}^{\mathsf{NP}}$ algorithm for ACC^0-Partial-AvgHard) for this application.

We refer the readers to the full version of the paper for formal definitions of these properties.

Before our work, Bhangale, Harsha, Paradise, and Tal [14] constructed a short, smooth, and rectangular PCP (instead of PCPP) built upon [13] with an application of constructing *rigid matrices* (also see [6, 49]). Ren, Santhanam, and Wang [45] constructed a short and rectangular PCPP based on [13, 14] for the Algorithmic Method for Avoid. It turns out that to generalise [45] to the "average-case" explicit construction problems Remote-Point and Partial-AvgHard, we need both *smoothness* (as in [14]) and PCPs of *proximity* (as in [45]). A technical contribution of this work is to combine [14] and [45] to obtain a *smooth PCPP*.

Theorem 2.14 (Informal). *Let $T(n)$ be a good function. For every language $L \in \mathsf{NTIME}[T(n)]$, there is a short, smooth, and (almost) rectangular PCP of proximity verifier V for L, with perfect completeness, constant soundness error, and constant query complexity.*

Following standard techniques in the algorithmic approach to lower bounds (see, e.g., [24]), we also construct a short and rectangular (non-smooth) PCPP that makes at most two queries to the input and the proof matrices to minimise the overhead on the circuit class when we reduce Avoid and Partial-Hard to Satisfying-Pairs (i.e. the upper OR_2 in Theorem 2.3 and Theorem 2.6). The constructions and the analysis are omitted in this extended abstract.

2.4 Further Related Work

In this section, we discuss several related works that share similar techniques or consider similar concepts.

2.4.1 Satisfying-Pairs and the Polynomial Method. We note that the Satisfying-Pairs problems for restricted circuit classes nicely capture a wide range of algorithmic problems that have been extensively studied. For instance, the Orthogonal Vector Problem over $\mathbb{F}_2$ corresponds to XOR-Satisfying-Pairs, and the (decision version of) Nearest Neighbor Problem corresponds to the Satisfying-Pairs of polynomial threshold functions (see, e.g., [4, 52]).

There is a successful line of research on non-trivial algorithms for this kind of problems via the *polynomial method* [44, 47] in circuit complexity. Williams [53] developed an $n^3/2^{(\log n)^{\Omega(1)}}$-time algorithm for the All-Pairs Shortest Path problem using the Razborov-Smolensky polynomial representation of $\mathsf{AC}^0[p]$ circuits [44, 47, 48] and a fast batch evaluation of polynomials via fast rectangular matrix multiplication [26]. Similar techniques were used to design non-trivial algorithms for the Orthogonal Vector Problem over $\mathbb{F}_2$ [1, 15] and the (approximate) Nearest Neighbor Problems (with respect to Hamming distance, ℓ_1-distance, and ℓ_2-distance) [4, 5, 7]. Chen and Wang [23] (following [8]) generalised the polynomial method in algorithm design by showing a connection between Satisfying-Pairs problems and quantum communication protocols, with an application in $\mathrm{Approx}_{\varepsilon}$-XOR-Satisfying-Pairs (which is called Approximate #OV in [23]).

2.4.2 Explicit Obstructions. Related to the Partial-Hard problem is the notion of explicit obstructions [19, 38]: on input 1^n, one wants to output a list of (x_i, y_i) deterministically, such that $x_i \neq x_j$ for $i \neq j$, and for all n-input circuit C from a certain circuit class $\mathscr{C}$, there is some i such that $C(x_i) \neq y_i$. This notion is weaker than deterministic algorithms for Partial-Hard, as one has the freedom of choosing the inputs $\{x_i\}$. Chen, Jin, and Williams [19] exhibited a "sharp threshold" phenomenon for explicit obstructions against de Morgan formulas: an explicit obstruction for $\mathsf{Formula}[n^{1.99}]$ provably exists, while an explicit obstruction for $\mathsf{Formula}[n^{2.01}]$ would imply very strong circuit lower bounds.

3 TECHNICAL OVERVIEW

As mentioned in Section 2.1, the range avoidance algorithm follows from slightly modifying the framework in [45] and using an algorithm for Satisfying-Pairs. In what follows, we briefly illustrate our techniques for the remote point problem and for constructing hard partial truth tables. The high-level idea is to reduce these problems to Avoid and invoke our framework for Avoid to solve them.

3.1 Remote Point

The start point of our $\mathsf{FP}^{\mathsf{NP}}$ algorithm for Remote-Point via non-trivial algorithms for Satisfying-Pairs is the following reduction from Remote-Point to Avoid. Suppose that $C : \{0,1\}^n \to \{0,1\}^\ell$ is the input circuit. Let $\mathsf{Enc} : \{0,1\}^{\ell'} \to \{0,1\}^\ell$ be the encoding procedure of an error correcting code, and $\mathsf{Dec} : \{0,1\}^\ell \to \{0,1\}^{\ell'}$ be the corresponding decoding procedure, where Dec can correct a δ fraction of errors. Define the circuit $C'(x) := \mathsf{Dec}(C(x))$, and let z be any string not in the range of C', then $\mathsf{Enc}(z)$ is $(1-\delta)$-far from $\mathsf{Range}(C)$. To see this, assume for contradiction that $\mathsf{Enc}(z)$ is $(1-\delta)$-close to some $C(x)$, then $\mathsf{Dec}(C(x))$ should return exactly z, contradicting that z is a non-output of C'.

Suppose that Dec can be implemented in the circuit class $\mathscr{C}_{\mathsf{Dec}}$, then this is a reduction from $\mathscr{C}$-Remote-Point to $(\mathscr{C}_{\mathsf{Dec}} \circ \mathscr{C})$-Avoid. Therefore, we would like the complexity of $\mathscr{C}_{\mathsf{Dec}}$ to be as small as possible. There are decoders that tolerate a small constant fraction of errors in AC^0 [31], so it might be possible to implement $\mathscr{C}_{\mathsf{Dec}}$ in AC^0. However, when δ is very close to $1/2$ (say $\delta = 1/2 - \varepsilon$), we enter the list-decoding regime where $\mathscr{C}_{\mathsf{Dec}}$ seems to need the power of majority [33]. Can we solve $\mathscr{C}$-Remote-Point without invoking any circuit-analysis algorithms for $\mathsf{MAJ} \circ \mathscr{C}$?

Fortunately, the required techniques already appeared in previous works on the Algorithmic Method for proving strong average-case circuit lower bounds. In [21], they provided an error-correcting code that corrects a $1/2 - \varepsilon$ fraction of errors, where the decoder $\mathsf{Dec}_{\mathsf{CLW}}$ can be implemented as a *linear sum*, i.e., each output is a linear combination of the input bits.[10] Intuitively, this means that we can reduce $\mathscr{C}$-Remote-Point to $(\mathsf{Sum} \circ \mathscr{C})$-Avoid, where Sum denotes the layer of $\mathsf{Dec}_{\mathsf{CLW}}$. Using the framework for range avoidance established above, it suffices to solve Satisfying-Pairs for $\mathsf{Sum} \circ \mathscr{C}$ circuits.[11] But it is easy to see that Satisfying-Pairs for $\mathsf{Sum}\circ\mathscr{C}$ circuits directly reduces to Satisfying-Pairs for $\mathscr{C}$ circuits! Therefore, the error-correcting code in [21] allows us to use an algorithm for $\mathscr{C}$-Satisfying-Pairs to directly solve $\mathscr{C}$-Remote-Point, with little or no circuit complexity overhead.

The above discussion omitted several important technical details:

- It turns out that $\mathsf{Dec}_{\mathsf{CLW}}$ is only an *approximate* list-decoding algorithm: given a corrupted codeword that is $(1/2-\varepsilon)$-close to the correct codeword, we can only recover a message that is δ-close to the correct message (instead of perfectly recovering the correct message).
  This drawback is handled by *smooth PCPPs* [40], which has the property that any slightly corrupted version of a correct proof is still accepted with good probability. As we need a rectangular PCPP in [45], what we actually need is a *smooth and rectangular PCPP* (see Theorem 2.14). We remark that [21] also encountered this difficulty; they got around it by combining a PCP and a PCPP for Circuit-Eval. It is not clear how to generalise this strategy to our case.
- Another technical complication is that $\mathsf{Dec}_{\mathsf{CLW}}$ outputs *real values* instead of Boolean values. It is only guaranteed that the decoded message is close to the original message *in ℓ_1-norm*. Consequently, after guessing the PCPP proof, we also need to verify that it is "close to Boolean", This difficulty also appears in [21]; however, we need to carefully define what it means by "close to Boolean" in our case.
- Since $\mathsf{Dec}_{\mathsf{CLW}}$ works in the list-decoding regime, it also receives an advice string (specifying the index of the codeword in the list). In the above discussion, we omitted the advice string to highlight the main ideas. It turns out that the dependency of the decoder on the advice string *cannot* be captured by linear sums. Therefore, we need to define an ad hoc "linear sum" circuit class (in Section 2.4 of the full version) that receives both an input and an advice string and computes a linear combination over the input, where the "linear combination" depends on the advice. It turns out that we need the dependency on the advice to be *local*, which is fortunately satisfied by the code in [21].

Another reduction via succinct dictionaries. We mention that there is another reduction from Remote-Point to Avoid which appears in [32, 36]. Let $C : \{0,1\}^n \to \{0,1\}^\ell$ be a circuit, $y \in \{0,1\}^\ell$ be a string that is not δ-far from $\mathsf{Range}(C)$. Then we can find a string $x \in \{0,1\}^n$ and a "noise" string $e \in \{0,1\}^m$ of relative Hamming weight at most δ such that $y = C(x) \oplus e$, where $\oplus$ refers to bit-wise XOR. Consider the circuit $C'(x,e) := C(x) \oplus e$. To solve the remote point problem for C, it suffices to solve the range avoidance problem for C'. Using a "succincter" dictionary to represent e [41], [32] managed to show that this reduction also preserves circuit complexity, and in particular reduces NC^1-Remote-Point to NC^1-Avoid.

A drawback of this approach is that it reduces Remote-Point to range avoidance instances with a small stretch. Indeed, suppose C' is a circuit from n' inputs to ℓ outputs, and $\delta = \Omega(1)$, then

$$n' \geq |\Pi(e)| \geq \log \binom{\ell}{\delta\ell} = \Omega(\ell).$$

In contrast, the algorithmic method in both [45] and this paper could not solve range avoidance instances with such a small stretch

[10] Chen et al. [21] stated this result as a non-standard XOR lemma in their Appendix A. We re-prove it in the form of error-correcting codes in the full version of this paper.

[11] We made a simplification here. Actually, we need to solve Satisfying-Pairs for $\mathsf{NC}^0 \circ \mathsf{Sum} \circ \mathscr{C}$ circuits. Using the distributive property, we can push the NC^0 circuits below the Sum layer, thus it suffices to solve Satisfying-Pairs for $\mathsf{Sum} \circ \mathsf{NC}^0 \circ \mathscr{C}$ circuits. In this informal exposition, we may assume that $\mathscr{C}$ is closed under top NC^0 gates, which means that a Satisfying-Pairs algorithm for $\mathsf{Sum} \circ \mathscr{C}$ now suffices.

($\ell = c \cdot n$ for some constant c), even with the best possible algorithms for Satisfying-Pairs. Therefore we do not use this approach in this paper.

3.2 Hard Partial Truth Table

There is a simple reduction from Partial-Hard to Avoid. Suppose we are given strings $x_1, x_2, \ldots, x_N$. Let TT' be the circuit that receives a size-s circuit C as input, and outputs the concatenation of $C(x_1), C(x_2), \ldots, C(x_N)$. If $N > O(s \log s)$ then the circuit TT' is stretching. It is also easy to see that solving the range avoidance of TT' is equivalent to solving the Partial-Hard problem.

In this paper, we essentially combine this reduction with the frameworks for Avoid and Remote-Point (see Theorems 2.3 and 2.5). In other words, we could have reduced Partial-Hard to Avoid in a black-box way and derived the main results. However, this reduction only reduces $\mathscr{C}$-Partial-Hard to $\mathscr{C}'$-Avoid, where $\mathscr{C}'$ is any circuit class that can solve $\mathscr{C}$-Eval in the following sense: for every fixed input x, there is a $\mathscr{C}'$ circuit C' that takes as input the description of a $\mathscr{C}$ circuit C, and outputs $C(x)$. For most circuit classes of interest (e.g., $\mathscr{C} \in \{\mathsf{AC}^0, \mathsf{ACC}^0, \mathsf{NC}^1, \mathsf{P}_{/\mathrm{poly}}\}$), we could simply let $\mathscr{C}' = \mathscr{C}$; however, this is not necessarily true for more refined circuit classes (such as $\mathscr{C} = \mathsf{ACC} \circ \mathsf{THR}$). We choose to derive the main results for hard partial truth table from scratch instead of reducing it to the framework for range avoidance and remote point problem, partly because we also want our results to hold for these more refined circuit classes.

ACKNOWLEDGMENTS

We got the initial idea of this work when Jiatu was a research intern at Igor Carboni Oliveira's group and Hanlin was visiting Igor. Hanlin wants to thank his supervisor Rahul Santhanam for introducing the problems Partial-Hard [11] and XOR-Remote-Point [9] to him and for helpful discussions. We thank Lijie Chen and an anonymous STOC reviewer for pointing out that the technique in [54] can be slightly adapted to obtain a non-trivial algorithm for #ACC-Satisfying-Pairs. We also thank Zhikun Wang and Tianqi Yang for the discussion on this work.

REFERENCES

[1] Amir Abboud, R. Ryan Williams, and Huacheng Yu. 2015. More Applications of the Polynomial Method to Algorithm Design. In *SODA*. SIAM, 218–230. https://doi.org/10.1137/1.9781611973730.17

[2] Miklós Ajtai. 1983. Σ_1^1-Formulae on finite structures. *Ann. Pure Appl. Log.* 24, 1 (1983), 1–48. https://doi.org/10.1016/0168-0072(83)90038-6

[3] Eric Allender and Vivek Gore. 1991. On Strong Separations from AC^0 (Extended Abstract). In *Fundamentals of Computation Theory, 8th International Symposium, FCT '91, Gosen, Germany, September 9-13, 1991, Proceedings (Lecture Notes in Computer Science, Vol. 529)*, Lothar Budach (Ed.). Springer, 1–15. https://doi.org/10.1007/3-540-54458-5_44

[4] Josh Alman, Timothy M. Chan, and R. Ryan Williams. 2016. Polynomial Representations of Threshold Functions and Algorithmic Applications. In *FOCS*. IEEE Computer Society, 467–476. https://doi.org/10.1109/FOCS.2016.57

[5] Josh Alman, Timothy M. Chan, and R. Ryan Williams. 2020. Faster Deterministic and Las Vegas Algorithms for Offline Approximate Nearest Neighbors in High Dimensions. In *SODA*. SIAM, 637–649. https://doi.org/10.1137/1.9781611975994.39

[6] Josh Alman and Lijie Chen. 2019. Efficient Construction of Rigid Matrices Using an NP Oracle. In *FOCS*. IEEE Computer Society, 1034–1055. https://doi.org/10.1109/FOCS.2019.00067

[7] Josh Alman and R. Ryan Williams. 2015. Probabilistic Polynomials and Hamming Nearest Neighbors. In *FOCS*. IEEE Computer Society, 136–150. https://doi.org/10.1109/FOCS.2015.18

[8] Josh Alman and R. Ryan Williams. 2017. Probabilistic rank and matrix rigidity. In *STOC*. ACM, 641–652. https://doi.org/10.1145/3055399.3055484

[9] Noga Alon, Rina Panigrahy, and Sergey Yekhanin. 2009. Deterministic Approximation Algorithms for the Nearest Codeword Problem. In *APPROX-RANDOM (Lecture Notes in Computer Science, Vol. 5687)*. Springer, 339–351. https://doi.org/10.1007/978-3-642-03685-9_26

[10] Sanjeev Arora and Boaz Barak. 2009. *Computational Complexity - A Modern Approach*. Cambridge University Press. http://www.cambridge.org/catalogue/catalogue.asp?isbn=9780521424264

[11] Vikraman Arvind and Srikanth Srinivasan. 2010. Circuit Lower Bounds, Help Functions, and the Remote Point Problem. In *ICS*. Tsinghua University Press, 383–396. http://conference.iiis.tsinghua.edu.cn/ICS2010/content/papers/30.html

[12] Richard Beigel and Jun Tarui. 1994. On ACC. *Comput. Complex.* 4 (1994), 350–366. https://doi.org/10.1007/BF01263423

[13] Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil P. Vadhan. 2006. Robust PCPs of Proximity, Shorter PCPs, and Applications to Coding. *SIAM J. Comput.* 36, 4 (2006), 889–974. https://doi.org/10.1137/S0097539705446810

[14] Amey Bhangale, Prahladh Harsha, Orr Paradise, and Avishay Tal. 2020. Rigid Matrices From Rectangular PCPs or: Hard Claims Have Complex Proofs. In *FOCS*. IEEE, 858–869. https://doi.org/10.1109/FOCS46700.2020.00084

[15] Timothy M. Chan and R. Ryan Williams. 2021. Deterministic APSP, Orthogonal Vectors, and More: Quickly Derandomizing Razborov-Smolensky. *ACM Trans. Algorithms* 17, 1 (2021), 2:1–2:14. https://doi.org/10.1145/3402926

[16] Eshan Chattopadhyay and David Zuckerman. 2019. Explicit two-source extractors and resilient functions. *Annals of Mathematics* 189, 3 (2019), 653–705. https://doi.org/10.4007/annals.2019.189.3.1

[17] Lijie Chen. 2019. Non-deterministic Quasi-Polynomial Time is Average-Case Hard for ACC Circuits. In *FOCS*. IEEE Computer Society, 1281–1304. https://doi.org/10.1109/FOCS.2019.00079

[18] Lijie Chen. 2022. *Better Hardness via Algorithms, and New Forms of Hardness versus Randomness.* Ph. D. Dissertation. Massachusetts Institute of Technology.

[19] Lijie Chen, Ce Jin, and R. Ryan Williams. 2020. Sharp threshold results for computational complexity. In *STOC*. ACM, 1335–1348. https://doi.org/10.1145/3357713.3384283

[20] Lijie Chen and Xin Lyu. 2021. Inverse-exponential correlation bounds and extremely rigid matrices from a new derandomized XOR lemma. In *STOC*. ACM, 761–771. https://doi.org/10.1145/3406325.3451132

[21] Lijie Chen, Xin Lyu, and R. Ryan Williams. 2020. Almost-Everywhere Circuit Lower Bounds from Non-Trivial Derandomization. In *FOCS*. IEEE, 1–12. https://doi.org/10.1109/FOCS46700.2020.00009

[22] Lijie Chen and Hanlin Ren. 2022. Strong Average-Case Circuit Lower Bounds from Nontrivial Derandomization. *SIAM J. Comput.* 51, 3 (2022), STOC20–115–STOC20–173. https://doi.org/10.1137/20M1364886

[23] Lijie Chen and Ruosong Wang. 2019. Classical Algorithms from Quantum and Arthur-Merlin Communication Protocols. In *ITCS (LIPIcs, Vol. 124)*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 23:1–23:20. https://doi.org/10.4230/LIPIcs.ITCS.2019.23

[24] Lijie Chen and R. Ryan Williams. 2019. Stronger Connections Between Circuit Analysis and Circuit Lower Bounds, via PCPs of Proximity. In *CCC (LIPIcs, Vol. 137)*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 19:1–19:43. https://doi.org/10.4230/LIPIcs.CCC.2019.19

[25] Ruiwen Chen, Igor Carboni Oliveira, and Rahul Santhanam. 2018. An Average-Case Lower Bound Against ACC^0. In *LATIN (Lecture Notes in Computer Science, Vol. 10807)*. Springer, 317–330. https://doi.org/10.1007/978-3-319-77404-6_24

[26] Don Coppersmith. 1982. Rapid Multiplication of Rectangular Matrices. *SIAM J. Comput.* 11, 3 (1982), 467–471. https://doi.org/10.1137/0211037

[27] Paul Erdős. 1959. Graph theory and probability. *Canadian Journal of Mathematics* 11 (1959), 34–38. https://doi.org/10.4153/CJM-1959-003-9

[28] Magnus Gausdal Find, Alexander Golovnev, Edward A. Hirsch, and Alexander S. Kulikov. 2016. A Better-Than-3n Lower Bound for the Circuit Complexity of an Explicit Function. In *FOCS*. IEEE Computer Society, 89–98. https://doi.org/10.1109/FOCS.2016.19

[29] Merrick L. Furst, James B. Saxe, and Michael Sipser. 1984. Parity, Circuits, and the Polynomial-Time Hierarchy. *Math. Syst. Theory* 17, 1 (1984), 13–27. https://doi.org/10.1007/BF01744431

[30] Karthik Gajulapalli, Alexander Golovnev, Satyajeet Nagargoje, and Sidhant Saraogi. 2023. Range Avoidance for Constant-Depth Circuits: Hardness and Algorithms. *CoRR* (2023). https://doi.org/10.48550/arXiv.2303.05044

[31] Shafi Goldwasser, Dan Gutfreund, Alexander Healy, Tali Kaufman, and Guy N. Rothblum. 2007. Verifying and decoding in constant depth. In *STOC*. ACM, 440–449. https://doi.org/10.1145/1250790.1250855

[32] Venkatesan Guruswami, Xin Lyu, and Xiuhan Wang. 2022. Range Avoidance for Low-Depth Circuits and Connections to Pseudorandomness. In *APPROX/RANDOM (LIPIcs, Vol. 245)*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 20:1–20:21. https://doi.org/10.4230/LIPIcs.APPROX/RANDOM.2022.20

[33] Dan Gutfreund and Guy N. Rothblum. 2008. The Complexity of Local List Decoding. In *APPROX-RANDOM (Lecture Notes in Computer Science, Vol. 5171)*.

Springer, 455–468. https://doi.org/10.1007/978-3-540-85363-3_36

[34] Johan Håstad. 1989. Almost Optimal Lower Bounds for Small Depth Circuits. *Adv. Comput. Res.* 5 (1989), 143–170.

[35] Robert Kleinberg, Oliver Korten, Daniel Mitropolsky, and Christos H. Papadimitriou. 2021. Total Functions in the Polynomial Hierarchy. In *ITCS (LIPIcs, Vol. 185)*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 44:1–44:18. https://doi.org/10.4230/LIPIcs.ITCS.2021.44

[36] Oliver Korten. 2021. The Hardest Explicit Construction. In *FOCS*. IEEE, 433–444. https://doi.org/10.1109/FOCS52979.2021.00051

[37] Jiatu Li and Tianqi Yang. 2022. $3.1n - o(n)$ circuit lower bounds for explicit functions. In *STOC*. ACM, 1180–1193. https://doi.org/10.1145/3519935.3519976

[38] Ketan Mulmuley. 2011. On P vs. NP and geometric complexity theory: Dedicated to Sri Ramakrishna. *J. ACM* 58, 2 (2011), 5:1–5:26. https://doi.org/10.1145/1944345.1944346

[39] Cody D. Murray and R. Ryan Williams. 2020. Circuit Lower Bounds for Nondeterministic Quasi-polytime from a New Easy Witness Lemma. *SIAM J. Comput.* 49, 5 (2020). https://doi.org/10.1137/18M1195887

[40] Orr Paradise. 2021. Smooth and Strong PCPs. *Comput. Complex.* 30, 1 (2021), 1. https://doi.org/10.1007/s00037-020-00199-3

[41] Mihai Pătraşcu. 2008. Succincter. In *FOCS*. IEEE Computer Society, 305–313. https://doi.org/10.1109/FOCS.2008.83

[42] Nicholas Pippenger. 1979. On Simultaneous Resource Bounds (Preliminary Version). In *FOCS*. IEEE Computer Society, 307–311. https://doi.org/10.1109/SFCS.1979.29

[43] C. Ramya. 2020. Recent Progress on Matrix Rigidity - A Survey. *CoRR* abs/2009.09460 (2020).

[44] Alexander A Razborov. 1987. Lower bounds on the size of bounded depth circuits over a complete basis with logical addition. *Mathematical Notes of the Academy of Sciences of the USSR* 41, 4 (1987), 333–338. https://doi.org/10.1007/BF01137685

[45] Hanlin Ren, Rahul Santhanam, and Zhikun Wang. 2022. On the Range Avoidance Problem for Circuits. In *FOCS*. IEEE, 640–650.

[46] Claude E. Shannon. 1949. The synthesis of two-terminal switching circuits. *Bell System technical journal* 28, 1 (1949), 59–98. https://doi.org/10.1002/j.1538-7305.1949.tb03624.x

[47] Roman Smolensky. 1987. Algebraic Methods in the Theory of Lower Bounds for Boolean Circuit Complexity. In *STOC*. ACM, 77–82. https://doi.org/10.1145/28395.28404

[48] Roman Smolensky. 1993. On Representations by Low-Degree Polynomials. In *FOCS*. IEEE Computer Society, 130–138. https://doi.org/10.1109/SFCS.1993.366874

[49] Leslie G. Valiant. 1977. Graph-Theoretic Arguments in Low-Level Complexity. In *MFCS (Lecture Notes in Computer Science, Vol. 53)*. Springer, 162–176. https://doi.org/10.1007/3-540-08353-7_135

[50] R. Ryan Williams. 2013. Improving Exhaustive Search Implies Superpolynomial Lower Bounds. *SIAM J. Comput.* 42, 3 (2013), 1218–1244. https://doi.org/10.1137/10080703X

[51] R. Ryan Williams. 2014. Nonuniform ACC Circuit Lower Bounds. *J. ACM* 61, 1 (2014), 2:1–2:32. https://doi.org/10.1145/2559903

[52] R. Ryan Williams. 2014. The Polynomial Method in Circuit Complexity Applied to Algorithm Design (Invited Talk). In *FSTTCS (LIPIcs, Vol. 29)*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 47–60. https://doi.org/10.4230/LIPIcs.FSTTCS.2014.47

[53] R. Ryan Williams. 2018. Faster All-Pairs Shortest Paths via Circuit Complexity. *SIAM J. Comput.* 47, 5 (2018), 1965–1985. https://doi.org/10.1137/15M1024524

[54] R. Ryan Williams. 2018. New Algorithms and Lower Bounds for Circuits With Linear Threshold Gates. *Theory Comput.* 14, 1 (2018), 1–25. https://doi.org/10.4086/toc.2018.v014a017

[55] Andrew Chi-Chih Yao. 1985. Separating the Polynomial-Time Hierarchy by Oracles (Preliminary Version). In *FOCS*. IEEE Computer Society, 1–10. https://doi.org/10.1109/SFCS.1985.49

Received 2022-11-07; accepted 2023-02-06