

Black-Box Constructive Proofs Are Unavoidable

Lijie Chen   

Miller Institute for Basic Research in Science, UC Berkeley, CA, USA

Ryan Williams   

CSAIL, MIT, Cambridge, MA, USA

Tianqi Yang   

IIS, Tsinghua University, Beijing, China

Abstract

Following Razborov and Rudich, a “natural property” for proving a circuit lower bound satisfies three axioms: constructivity, largeness, and usefulness. In 2013, Williams proved that for any reasonable circuit class \mathcal{C} , $\text{NEXP} \not\subseteq \mathcal{C}$ is equivalent to the existence of a constructive property useful against \mathcal{C} . Here, a property is constructive if it can be decided in $\text{poly}(N)$ time, where $N = 2^n$ is the length of the truth-table of the given n -input function.

Recently, Fan, Li, and Yang initiated the study of *black-box natural properties*, which require a much stronger notion of constructivity, called *black-box constructivity*: the property should be decidable in randomized $\text{polylog}(N)$ time, given oracle access to the n -input function. They showed that most proofs based on random restrictions yield *black-box* natural properties, and demonstrated limitations on what black-box natural properties can prove.

In this paper, perhaps surprisingly, we prove that the equivalence of Williams holds even with this stronger notion of black-box constructivity: for any reasonable circuit class \mathcal{C} , $\text{NEXP} \not\subseteq \mathcal{C}$ is equivalent to the existence of a black-box constructive property useful against \mathcal{C} . The main technical ingredient in proving this equivalence is a smooth, strong, and locally-decodable probabilistically checkable proof (PCP), which we construct based on a recent work by Paradise. As a by-product, we show that average-case witness lower bounds for PCP verifiers follow from NEXP lower bounds.

We also show that randomness is essential in the definition of black-box constructivity: we unconditionally prove that there is no *deterministic* $\text{polylog}(N)$ -time constructive property that is useful against even polynomial-size AC^0 circuits.

2012 ACM Subject Classification Theory of computation \rightarrow Circuit complexity

Keywords and phrases Circuit lower bounds, natural proofs, probabilistic checkable proofs

Digital Object Identifier 10.4230/LIPIcs.ITCS.2023.35

Funding *Ryan Williams*: Supported by NSF CCF-2127597.

Acknowledgements We would also like to thank Jiatu Li for discussions during the early stage of this research project and anonymous reviewers for their comments.

1 Introduction

In a seminal paper [30], Razborov and Rudich argued the following:

1. Most known proofs of circuit lower bounds yield an efficient algorithm (called a *natural property*) which takes Boolean truth tables of functions as input, accepts a large fraction of its inputs, and rejects all functions computable in some circuit class \mathcal{C} . (These kinds of proofs are called “natural proofs”.)
2. Such an efficient algorithm would imply that \mathcal{C} cannot support exponentially secure pseudorandom functions (PRFs), as the algorithm could break any such candidate.

Since PRFs are widely believed to be implementable with even TC^0 circuits (constant-depth circuits of polynomial size, consisting of MAJORITY and NOT gates) [24, 19, 22], the Razborov-Rudich barrier strongly suggests that proofs yielding natural properties cannot prove lower bounds against weak circuit classes such as TC^0 . In this paper, we reconsider the question of how to “avoid” natural proofs.



© Lijie Chen, Ryan Williams, and Tianqi Yang;
licensed under Creative Commons License CC-BY 4.0

14th Innovations in Theoretical Computer Science Conference (ITCS 2023).

Editor: Yael Tauman Kalai; Article No. 35; pp. 35:1–35:24

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

To formally discuss the barrier, we begin with some notation. Let \mathcal{B}_n denote the set of all n -bit Boolean functions. We use N to denote 2^n (the length of a truth table of a function in \mathcal{B}_n). A *promise property* has the form $\mathcal{P} = \{(\mathcal{P}_n^{\text{yes}}, \mathcal{P}_n^{\text{no}})\}_{n \in \mathbb{N}}$, where $\mathcal{P}_n^{\text{yes}}, \mathcal{P}_n^{\text{no}} \subseteq \mathcal{B}_n$ and $\mathcal{P}_n^{\text{yes}} \cap \mathcal{P}_n^{\text{no}} = \emptyset$ for each $n \in \mathbb{N}$. When $\mathcal{P}_n^{\text{yes}} \cup \mathcal{P}_n^{\text{no}} = \mathcal{B}_n$, we simply call \mathcal{P} a *property*. (The original paper [30] only considered properties of Boolean functions; here we consider “promise” versions, as we wish to study *sub-linear* time decidable properties.)

Let Γ be a complexity class, and let \mathcal{C} be a circuit class. A Razborov-Rudich natural property satisfies three criteria, defined as follows.

► **Definition 1.1 (Natural Property).** *Let \mathcal{P} be a promise property.*

(Usefulness) *We say \mathcal{P} is useful against \mathcal{C} of size $s(n)$, if (1) $|\mathcal{P}_n^{\text{yes}}| \geq 1$ for every $n \in \mathbb{N}$ and (2) for every function family $f = \{f_n \in \mathcal{B}_n\}_{n \in \mathbb{N}}$ such that f admits $s(n)$ -size \mathcal{C} circuits, there are infinitely many $n \in \mathbb{N}$ such that $f_n \in \mathcal{P}_n^{\text{no}}$.¹*

We say \mathcal{P} is useful against \mathcal{C} if it is useful against n^k -size \mathcal{C} circuits for all $k \in \mathbb{N}$.

(Largeness) *We call \mathcal{P} large if there is a polynomial $p: \mathbb{N} \rightarrow \mathbb{N}$ such that for all but finitely many $n \in \mathbb{N}$, $\Pr_{f \in \mathcal{B}_n} [f \in \mathcal{P}_n^{\text{yes}}] \geq 1/p(n)$.*

(Constructivity) *We say that \mathcal{P} is Γ -constructive, if there is a promise- Γ algorithm \mathcal{A} such that for every $f \in \mathcal{P}_n^{\text{yes}}$, $\mathcal{A}(\text{tt}(f))$ accepts and for every $f \in \mathcal{P}_n^{\text{no}}$, $\mathcal{A}(\text{tt}(f))$ rejects.*

A Γ -natural property against \mathcal{C} [30] is a Γ -constructive large property useful against \mathcal{C} .²

Circumventing Natural Properties. Since Razborov and Rudich’s work, a considerable amount of effort (see, e.g., [1, 13, 10]) has gone into the problem of *circumventing* the natural proofs barrier, by identifying properties of functions which fail to satisfy one of the three criteria of natural properties. The usefulness criterion is necessary³, so the goal is to find properties avoiding either constructivity (the property does not admit an *efficient* algorithm) or largeness (the property does not accept a large fraction of functions). In 2013, Williams [36] proved that **constructivity is unavoidable**, in a rigorous sense: if even “weak” circuit lower bounds hold (against NEXP) then there must exist properties satisfying both constructivity and usefulness.

► **Theorem 1.2 ([36, 33]).** *Let $\mathcal{C} \in \{\text{AC}^0, \text{ACC}^0, \text{TC}^0, \text{NC}^1, \text{NC}, \text{P/poly}\}$. The following are equivalent:*

- $\text{NEXP} \not\subseteq \mathcal{C}$.
- There is a $\text{P}_{/\log N}$ -constructive property useful against \mathcal{C} .⁴

This theorem suggests that we should focus on looking for hard properties that are easily recognizable on specific functions, but generally do not hold for random functions or for “easy” functions.

Another Crack in The Natural Proofs Barrier? One potential weakness of the Natural Proofs barrier is that it does not seem to apply to proving *fixed-polynomial size* lower bounds (for a reference, see the discussions by Allender and Koucký in [1, 2]). For example,

¹ In particular, this means for every function family $f = \{f_n\}_{n \in \mathbb{N}}$ such that $f_n \in \mathcal{P}_n^{\text{yes}}$, f does not have $s(n)$ -size \mathcal{C} circuits.

² We also refer to P-natural properties as simply “natural properties”. Razborov and Rudich [30] proved that assuming there are exponentially secure PRFs computable by \mathcal{C} , there is no natural property useful against \mathcal{C} . Indeed, the same argument also shows there is no P-constructive large *promise property* useful against \mathcal{C} , under the same assumption.

³ $f \notin \mathcal{C}$ if and only if there is a property that accepts f , and rejects all functions computable by \mathcal{C} .

⁴ Recall that $\text{P}_{/\log N}$ denote the class of $\text{poly}(N)$ time algorithm with $\log N$ bits of advice on N -bit inputs.

the best known size lower bounds against DeMorgan formulas have been stuck at nearly cubic ($n^{3-o(1)}$) for more than two decades [4, 17],⁵ and the Natural Proofs barrier apparently does not say anything about the hardness of proving $n^{3+\varepsilon}$ -size DeMorgan formula lower bounds.

Partially motivated by this limitation, a line of work on *amplifying hardness* or *hardness magnification* [2, 20, 26, 21, 25, 11] suggested a way to *bypass* the Natural Proofs barrier with a two-step process:

- (1) prove a weak (fixed-polynomial, say $n^{3+\varepsilon}$ -size) lower bound for a specific function, and
- (2) prove some “bootstrapping” or “hardness magnification” result showing that the weak lower bound actually implies a desired super-polynomial lower bound.

These hardness magnification phenomena are especially interesting because of certain “threshold” phenomena, where one can prove decent weak lower bounds for a specific function, and one can prove that slight improvements over the decent lower bound would imply breakthrough lower bounds. For instance, Chen, Jin, and Williams [12] proved that (1) a sparse variant of MCSP does not have $n^{2-o(1)}$ -size probabilistic De Morgan formulas and (2) if the lower bound could be improved to $n^{2+\varepsilon}$ size⁶, then we would have major breakthroughs in complexity theory: more precisely, NP does not have n^k -size formulas for every $k \in \mathbb{N}$, and $\oplus P \not\subseteq NC^1$.

Black-Box Natural Properties: Another Barrier. To better understand these bootstrapping results and their limitations, Fan, Li, and Yang [14] defined the notion of *Black-Box Natural Properties*, and applied it to provide a new barrier to bootstrapping results. Black-box natural properties are natural (promise) properties that are decidable by *sublinear-time* randomized algorithms.

► **Definition 1.3** (Black-Box Natural Properties, [14]). *A black-box natural property against \mathcal{C} is a $BTIME[\text{polylog}(N)]$ -constructive large promise property useful against \mathcal{C} . In particular, there is a $\text{polylog}(N)$ -time⁷ randomized oracle algorithm A such that for every $f \in \mathcal{P}_n^{\text{yes}}$, A^f accepts with probability at least $2/3$, and for every $f \in \mathcal{P}_n^{\text{no}}$, A^f rejects with probability at least $2/3$.*

The authors showed negative results for hardness magnification, in that:

- (1) all known non-trivial lower bounds for the specific functions studied in hardness magnification actually yield *black-box natural properties*, and
- (2) under plausible cryptographic assumptions, **no** black-box natural property can establish the lower bounds needed in the corresponding bootstrapping results.

Taking Chen-Jin-Williams [12] as an example, their $n^{2-o(1)}$ -size probabilistic formula lower bound for sparse MCSP yields a black-box natural property, but no black-box natural property can yield a $n^{2+\varepsilon}$ -size lower bound against probabilistic formulas.⁸

⁵ There are some recent low-order improvements [31, 32].

⁶ Under plausible cryptographic assumptions, this sparse variant of MCSP is not even in P/poly, so we definitely believe the $n^{2+\varepsilon}$ -size lower bound should hold.

⁷ That is, for every $f \in \mathcal{B}_n$, A^f runs in $\text{poly}(n) = \text{polylog}(N)$ time.

⁸ We also remark that prior to Fan, Li, and Yang [14], the work [11] proposed another barrier for bootstrapping results, called the *locality barrier*. We do not discuss this barrier here as it is not relevant to the results in this paper, and refer readers to [11] for details.

1.1 Our Results

1.1.1 Equivalence of Black-Box Constructive Properties and Circuit Lower Bounds

Given the results of Fan, Li, and Yang [14], suppose we still wish to apply a bootstrapping result such as the one from Chen, Jin, and Williams [12]. We would need to find a property that either avoids black-box constructivity, or avoids largeness (again, usefulness is necessary). *A priori*, it may appear that one should try to avoid black-box constructivity, as it is an extremely strong requirement.

Strengthening the equivalence of Williams [33], we prove that even black-box constructivity is **unavoidable** for proving NEXP lower bounds. In fact, we are able to prove that $\text{NEXP} \not\subseteq \mathcal{C}$ is equivalent to the existence of a much stronger type of black-box constructive property, in which the algorithm only makes one-sided error and can distinguish hard functions from all functions which can be *approximated* by small circuits. That is, the usefulness criterion corresponds to an *average-case* lower bound.

► **Theorem 1.4** (Black-Box Constructivity is Unavoidable). *Let \mathcal{C} be a circuit class such that any s -size $\text{AC}^0[2] \circ \mathcal{C}$ circuit family can be simulated by an $\text{poly}(n, s)$ -size \mathcal{C} circuit family. The following are equivalent:*

- $\text{NEXP} \not\subseteq \mathcal{C}$.
- There is a $\text{coRTIME}[\text{polylog}(N)]_{/ \log N}$ -constructive property useful against $\text{avg}_{n^{-6}}\text{-}\mathcal{C}$.

More precisely, the second item above states there is a promise property \mathcal{P} such that:

1. (**Average-case useful**) For every function family $f = \{f_n \in \mathcal{B}_n\}_{n \in \mathbb{N}}$ such that f can be $(1 - n^{-6})$ -approximated by $s(n)$ -size \mathcal{C} circuits, there are infinitely many $n \in \mathbb{N}$ such that $f_n \notin \mathcal{P}_n^{\text{no}}$.⁹
2. (**One-sided error**) There is a $\text{polylog}(N)$ -time randomized oracle machine A which takes $\log N$ bits of advice, such that every $f \in \mathcal{P}_n^{\text{yes}}$ is accepted by A with probability 1 and every $f \in \mathcal{P}_n^{\text{no}}$ is accepted by A with probability at most $1/3$.

The following is an interesting corollary of Theorem 1.4, showing that average-case circuit lower bounds follow from (worst-case) circuit lower bounds, while only assuming the weak circuit class contains $\text{AC}^0[2]$.

► **Corollary 1.5.** *Let \mathcal{C} be a circuit class such that any s -size $\text{AC}^0[2] \circ \mathcal{C}$ circuit family can be simulated by an $\text{poly}(n, s)$ -size \mathcal{C} circuit family. Then, $\text{NEXP} \not\subseteq \mathcal{C}$ implies that EXP^{NP} cannot be $(1 - 1/n^6)$ -approximated by \mathcal{C} .*

Can These Properties be Derandomized? It is natural to ask whether we can further strengthen Theorem 1.4 so that $\text{NEXP} \not\subseteq \mathcal{C}$ is equivalent to the existence of $\text{DTIME}[\text{polylog}(n)]$ -constructive or $\text{RTIME}[\text{polylog}(n)]_{/ \log N}$ -constructive properties useful against \mathcal{C} . We show that this is impossible by proving that for essentially all circuit classes, $\text{DTIME}[\text{polylog}(n)]$ -constructive properties useful against them do not exist. In fact we prove that even $\text{NTIME}[\text{polylog}(N)]_{/ \text{polylog}(N)}$ -constructive properties do not exist for any circuit class \mathcal{C} that is expressible enough to simulate CNF formulas (see Section 5 for more details).

► **Theorem 1.6.** *For any circuit class \mathcal{C} such that all CNFs of t clauses have $\text{poly}(n, t)$ -size \mathcal{C} -circuits, there is no $\text{DTIME}[\text{polylog}(n)]$ -constructive property useful against \mathcal{C} .*

⁹ [9] studied a similar notion called *tolerant natural proofs*, which can be seen as natural proofs for proving average-case circuit lower bounds.

On the positive side, we observe two settings in which natural properties can be implemented in deterministic sublinear time, by examining some random restriction lemmas from prior work [15, 12].

► **Theorem 1.7.** *For every $\varepsilon \in (0, 1)$, there is a $\text{DTIME}[\text{polylog}(N)]$ -constructive large property useful against $n^{2-\varepsilon}$ -size formulas.*

► **Theorem 1.8.** *For every $d, k \in \mathbb{N}$, there is a $\text{DTIME}[\text{polylog}(N)]$ -constructive large property useful against depth- d AC^0 circuits of size n^k .*

It is important to note that Theorem 1.8 does not contradict Theorem 1.6, since Theorem 1.6 only says there is no *single* $\text{DTIME}[\text{polylog}(n)]$ -constructive property that is useful against *all* polynomial-size CNF.

How to Remove the $\log N$ -bit Advice. Similar to Williams [33], our equivalence of black-box constructive properties and NEXP circuit lower bounds from Theorem 1.4 requires $n = \log N$ bits of advice for constructivity. A natural question is whether this advice can be removed in some cases.

We show that removing the advice is possible, when the NEXP lower bound is proved via the algorithmic method [35, 37]. Recall that CAPP (CIRCUIT ACCEPTANCE PROBABILITY PROBLEM) with error δ (denoted CAPP_δ) is the following problem: Given a circuit C on n inputs, estimate $\Pr_{x \in \{0,1\}^n}[C(x) = 1]$ within an *additive error* of δ .

► **Theorem 1.9.** *Let $K \in \mathbb{N}$ be a sufficiently large constant. Let \mathcal{C} be a circuit class and $s(n) \geq n$ be a size parameter. If there is a $2^n/n^{10}$ time $\text{CAPP}_{0.1}$ algorithm for $s(n)^K$ -size $\text{AC}_2^0 \circ \mathcal{C}$ circuits, then there is a $\text{coRTIME}[\text{polylog}(N)]$ -constructive property useful against $s(n)$ -size \mathcal{C} circuits.*

The following corollary follows immediately from Theorem 1.9 and the CAPP algorithm for ACC^0 [37, 34]. (Indeed, the reference [34] provides an algorithm for exactly counting satisfying assignments.)

► **Corollary 1.10.** *For every constant $d, m \in \mathbb{N}$, there is a constant $\varepsilon \in (0, 1)$ and a $\text{coRTIME}[\text{polylog}(N)]$ -constructive property useful against 2^{n^ε} -size $\text{AC}_d^0[m]$ circuits.*

1.1.2 Bootstrapping Results on Black-box Natural Properties

We also obtain some new insights regarding black-box natural properties defined by Fan, Li, and Yang [14] (*i.e.*, $\text{BPTIME}[\text{polylog}(N)]$ -constructive large promise properties). We prove that black-box natural properties against rather small circuit size can be bootstrapped to black-box natural properties against n^k -size circuit classes, for every $k \in \mathbb{N}$. This is similar to (but not implied by) the bootstrapping results for pseudorandom functions of Fan, Li, and Yang.

We will state our results in the context of De Morgan formulas, but we note that it applies to all well-studied circuit classes as well; see Section 6 for details.

► **Theorem 1.11.** *If there is a black-box natural property a.e.-useful against DeMorgan formulas of size $n^{2+\varepsilon}$ for some $\varepsilon \in (0, 1)$, then for all $k \in \mathbb{N}$, there is a black-box natural property a.e.-useful against formulas of size n^k .¹⁰*

¹⁰See Definition 2.1 for a formal definition of a.e.-useful properties.

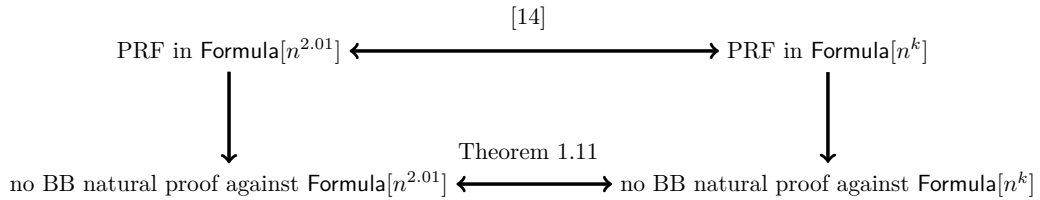
35:6 Black-Box Constructive Proofs Are Unavoidable

This theorem is striking in that it shows, regardless of whether PRFs exist or not, proving an $n^{2+\varepsilon}$ -size formula lower bound via a black-box natural property implies *superpolynomial*-size formula lower bounds (for example, $\text{NEXP} \not\subseteq \text{NC}^1$).

Here we consider a.e.-usefulness instead of the (i.o.-)usefulness defined in Definition 1.1, because all known black-box natural properties are indeed a.e.-useful. It is instructive to compare Theorem 1.11 to the following result of Fan, Li, and Yang [14].

► **Theorem 1.12** ([14]). *If there is a PRF computable by formulas of polynomial size, then there is a PRF computable by formulas of size $n^{2.01}$.*

We note that a black-box natural property against \mathcal{C} implies that there is no PRF computable in \mathcal{C} , but the reverse direction is not known to hold. Hence, an interesting open question is to prove an equivalence between black-box natural proofs and the non-existence of PRFs, or to give evidence that the two notions are not equivalent.



■ **Figure 1** Relationships between black-box natural properties and PRFs.

2 Preliminaries

We assume basic knowledge to complexity theory such as the complexity classes P , NP , $\text{P}_{/\text{poly}}$, and ACC^0 ; see the textbook by Arora and Barak [5] for an excellent reference. We define $\text{NE} = \text{NTIME}[2^{O(n)}]$. For constant depth circuit classes such as AC^0 , we use AC_d^0 to denote its subclass of depth at most d .

Let $f : \{0, 1\}^n \rightarrow \{0, 1\}$ be a Boolean function. We always use $N = 2^n$ to denote the length of its truth table. We denote the truth table of the function f by $\text{tt}(f)$, which is an N -bit string, where the i -th bit (0-index, base-2) is equivalent to $f(i)$. We define \mathcal{B}_n as the set of all n -bit Boolean functions.

Let \mathcal{C} be a circuit class. We define $\text{avg}_\delta\text{-}\mathcal{C}$ to be the class of function families that are $(1 - \delta)$ -approximable by \mathcal{C} circuits. More precisely, a function family $f = \{f_n\}_{n \in \mathbb{N}}$ belongs to $\text{avg}_\delta\text{-}\mathcal{C}$, if there is another function family $g = \{g_n\}_{n \in \mathbb{N}} \in \mathcal{C}$ such that f_n and g_n agree on a least a $(1 - \delta)$ fraction of inputs from $\{0, 1\}^n$ for every $n \in \mathbb{N}$.

We use the notation $\mathcal{C}\text{-SIZE}[s(n)]$ to denote the class of languages computable by a family of \mathcal{C} circuits of size at most $s(n)$ for all sufficiently large n . Letting \mathcal{D} be another circuit class, we let $\mathcal{D} \circ \mathcal{C}$ be the class of circuits with \mathcal{C} circuits in the bottom layer (nearest the inputs) and a \mathcal{D} circuit on the top. We say that the class \mathcal{C} is *closed under* \mathcal{D} , if $\mathcal{D} \circ \mathcal{C} = \mathcal{C}$, i.e., composing with a \mathcal{D} circuit on top does not increase the power of \mathcal{C} . We call a circuit class *typical* if it is closed under negations, projections, and conjunctions.

Let $x, y \in \{0, 1\}^n$ be two bit strings. The *statistical distance* between x and y is defined by

$$\Delta(x, y) \triangleq \frac{\|x - y\|_0}{n},$$

where $\|x - y\|_0$ is the number of indices $i \in [n]$ such that $x_i \neq y_i$. We say that x is δ -close to y if $\Delta(x, y) \leq \delta$. Conversely, we say that x is δ -far from y if it is not δ -close to y . Similarly, the statistical distance from a string x to a set S is defined by $\Delta(x, S) = \min_{y \in S} \Delta(x, y)$. If S is empty, we define $\Delta(x, S)$ to be 1 for convenience.

2.1 Sublinear Time Classes

We will extensively use complexity classes for problems solvable in sublinear time, such as $\text{DTIME}[\text{polylog}(N)]$ and $\text{coRTIME}[\text{polylog}(N)]$. Specifically, the underlying Turing machine of $\text{DTIME}[\text{polylog}(N)]$ and $\text{coRTIME}[\text{polylog}(N)]$ has two tapes: an input tape and a work tape. While the work tape is a “normal” one in the Turing machine sense, the input tape is read-only and our machine is allowed *random access* to it.

An alternative way of viewing the situation is that our algorithm has *oracle access* to the input string, and our algorithm is required to accept or reject within a sublinear (or even $\text{polylog}(N)$) number of steps. For example, let $\mathcal{A}(\text{tt}(f))$ be an algorithm that takes the truth-table of a function $f \in \mathcal{B}_n$ as input. If \mathcal{A} is in $\text{DTIME}[\text{polylog}(N)]$, then \mathcal{A} can be equivalently viewed as a $\text{poly}(n)$ -time algorithm $\hat{\mathcal{A}}^f(1^n)$ that takes 1^n as input and is given oracle access to the function $f \in \mathcal{B}_n$.

These two views are essentially equivalent, and we will use the second one (where we have oracle access to the input string) when describing our algorithms.

2.2 Properties

We are now ready to formally define what it means for a property being *constructive*, *large*, and *useful*. Recall that a *promise property* is $\mathcal{P} = \{(\mathcal{P}_n^{\text{yes}}, \mathcal{P}_n^{\text{no}})\}_{n \in \mathbb{N}}$, where $\mathcal{P}_n^{\text{yes}}, \mathcal{P}_n^{\text{no}} \subseteq \mathcal{B}_n$ and $\mathcal{P}_n^{\text{yes}} \cap \mathcal{P}_n^{\text{no}} = \emptyset$ for every $n \in \mathbb{N}$.

► **Definition 2.1** (Usefulness). *For a circuit class \mathcal{C} , we say that \mathcal{P} is useful against $s(n)$ -size \mathcal{C} circuits, if the following two conditions hold:*

- $|\mathcal{P}_n^{\text{yes}}| \geq 1$ for all input lengths $n \in \mathbb{N}$,
- For every function family $f = \{f_n \in \mathcal{B}_n\}_{n \in \mathbb{N}}$ admitting $s(n)$ -size \mathcal{C} circuits, there is an infinite increasing sequence of input lengths $\{n_i\}_{i \in \mathbb{N}}$ such that $f_{n_i} \in \mathcal{P}_{n_i}^{\text{no}}$ for every $i \in \mathbb{N}$.

Similarly, we say that \mathcal{P} is almost-everywhere useful (a.e.-useful) against $s(n)$ -size \mathcal{C} circuits, if $f_n \in \mathcal{P}_n^{\text{no}}$ for all but finitely many input lengths $n \in \mathbb{N}$.

► **Definition 2.2** (Largeness). *A property \mathcal{P} is called large if there is a constant $c \geq 1$ such that for all but finitely many $n \in \mathbb{N}$,*

$$\Pr_{f \in \mathcal{B}_n} [f \in \mathcal{P}_n^{\text{yes}}] \geq n^{-c}.$$

Note that the definition above of largeness is different from the definition in [30], where $\Pr_{f \in \mathcal{B}_n} [f \in \mathcal{P}_n^{\text{yes}}]$ is only required to be greater than $1/\text{poly}(2^n)$. These two definitions are equivalent when we are considering P-constructive properties. Our largeness definition is more suitable when studying (say) $\text{BPTIME}[\text{polylog}(N)]$ -constructive properties.

► **Definition 2.3** (Constructivity). *Let Γ be a complexity class. A property \mathcal{P} is called Γ -constructive, if there is a promise- Γ algorithm \mathcal{A} such that the following hold:*

- For every $f \in \mathcal{P}_n^{\text{yes}}$, $\mathcal{A}(\text{tt}(f))$ accepts.
- For every $f \in \mathcal{P}_n^{\text{no}}$, $\mathcal{A}(\text{tt}(f))$ rejects.

If a property \mathcal{P} is both Γ -constructive and large, we call it Γ -*natural*. We now compare these three axioms with previously studied notions in the literature.

► **Example 2.4** (Natural proof). The classical notion of a natural proof against \mathcal{C} [30] can be stated in our terminology as a P-constructive large property (*i.e.*, P-natural property) useful against \mathcal{C} .

► **Example 2.5** (Black-box natural proof). The notion of a black-box natural proof against \mathcal{C} defined by Fan, Li, and Yang [14] can be stated as a $\text{BPTIME}[\text{polylog}(N)]$ -constructive large (*i.e.*, a $\text{BPTIME}[\text{polylog}(N)]$ -natural) property useful against \mathcal{C} . In other words, there is a randomized oracle algorithm A such that:

- For every $f \in \mathcal{B}_n$, $A(\text{tt}(f))$ runs in $\text{poly}(n) = \text{polylog}(N)$ time.
- For every $f \in \mathcal{P}_n^{\text{yes}}$, $A(\text{tt}(f))$ accepts with probability at least $2/3$.
- For every $f \in \mathcal{P}_n^{\text{no}}$, $A(\text{tt}(f))$ accepts with probability at most $1/3$.

2.3 Nondeterministic Classes and Easy Witnesses

We first need a definition of what it means for a language (and a complexity class) to have small circuits encoding its witnesses. Similar to Williams [33], we restrict ourselves to “good” verifiers that only examine witnesses of length equal to a power of two, so that the witnesses can be viewed as truth-tables of Boolean functions. The following definition is from Williams [33].

► **Definition 2.6.** Let $L \in \text{NTIME}[T(n)]$, where $T(n) \geq n$ is time constructible and always a power of 2.¹¹ An algorithm $V(x, y)$ is a good predicate for L if

- V runs in time $O(T(|x|))$, and
- for all $x \in \{0, 1\}^*$, $x \in L$ if and only if there is a string $y \in \{0, 1\}^{T(|x|)}$ (a witness for x) such that $V(x, y)$ accepts.

We call $T(n)$ the witness length of V . Let $L(V)$ denote the language accepted by V .¹²

Let V be a good predicate with witness length $T(n)$ and \mathcal{C} be a circuit class. Let $\ell(n) = \log T(n)$. We say that V has \mathcal{C} witnesses of size $s(n)$, if for all input sequences $\{x_n\}_{n \in \mathbb{N}}$ such that $|x_n| = n$, there is an $s(n)$ -size \mathcal{C} circuit family $\{C_n \in \mathcal{B}_{\ell(n)}\}_{n \in \mathbb{N}}$, such that for all sufficiently large $n \in \mathbb{N}$, if $x \in L$, then $V(x, \text{tt}(C_{|x|}))$ accepts.

For simplicity, we say that V has \mathcal{C} witnesses if for some polynomial p , V has \mathcal{C} witnesses of size $p(n)$. We say that the class $\text{NTIME}[T(n)]$ has \mathcal{C} witnesses (of size $s(n)$), if for every $L \in \text{NTIME}[T(n)]$, every good predicate for L has \mathcal{C} witnesses (of size $s(n)$). And similarly, we say that NEXP has \mathcal{C} witnesses (of size $s(n)$), if for every polynomial $p(n)$, $\text{NTIME}[2^{p(n)}]$ has \mathcal{C} witnesses (of size $s(n)$).

► **Lemma 2.7** (Theorem 3.1, [33]). Let \mathcal{C} be a typical polynomial-size circuit class such that $\text{AC}^0 \circ \mathcal{C} \subseteq \mathcal{C}$. $\text{NEXP} \subset \mathcal{C}$ is equivalent to NEXP having \mathcal{C} witnesses.¹³

¹¹We require $T(n)$ to be a power of 2 for convenience. We note that this is without loss of generality since any running time bound can be rounded up to the nearest power of 2, which only incurs a constant multiplicative overhead.

¹²We note that L is indeed defined by both V and T , so strictly we should write $L(T, V)$. But T will always be clear from the context so we will slightly abuse the notation and write it as $L(V)$.

¹³Williams [33] only considered $\mathcal{C} \in \{\text{AC}^0, \text{ACC}^0, \text{TC}^0, \text{NC}^1, \text{NC}, \text{P/poly}\}$, but the proof of his equivalence only requires that $\text{AC}^0 \circ \mathcal{C} \subseteq \mathcal{C}$.

2.4 Probabilistically Checkable Proofs

We now give a brief description of probabilistically checkable proofs (PCPs); see for example Harsha’s PhD thesis [16] for a comprehensive introduction. We begin with the definition of a *PCP verifier*.

► **Definition 2.8** (PCP verifier). *Let $T(n) \geq n$ be time constructible and always a power of 2. Let $V(x, y)$ be a good predicate for some language $L \in \text{NTIME}[T(n)]$. Let $n = |x|$ be the input length. A PCP verifier PCP_V takes x as input, runs in $\text{poly}(n)$ time, and outputs a non-adaptive oracle circuit $\text{PCP}_V(x, \cdot)$ which takes an additional randomness r as input, makes q queries to the oracle, and satisfies the following two constraints:*

(Completeness) *For all $x \in L$, there is a proof π such that $\Pr_r[\text{PCP}_V(x, r)^\pi \text{ accepts}] = 1$.*

(Soundness) *For all $x \notin L$ and proofs π , we have $\Pr_r[\text{PCP}_V(x, r)^\pi \text{ accepts}] < 1/3$.*

The length of the additional randomness r is called the randomness complexity, the number of queries q made by the oracle circuit is called the query complexity, and the maximum size of all oracle circuits $D_{x,r}$ is called the decision complexity.

In general, we can assume that each oracle query is of length $\ell(n) = \lceil |r| \log q \rceil$,¹⁴ so we can identify the oracle π above with a binary string of length $2^{\ell(n)}$ (i.e., a Boolean function on $\ell(n)$ bits).

We define a PCP system for $\text{NTIME}[T(n)]$ to be a mapping that maps every good predicate V for some language $L \in \text{NTIME}[T(n)]$, into PCP_V , a PCP verifier for L .

We can similarly say that the PCP system has randomness complexity $r(n)$, query complexity $q(n)$, and decision complexity $d(n)$, if every PCP_V in its image has such randomness/query/decision complexity.

Note that here we require the oracle circuit to be *non-adaptive*, meaning that there is at most one oracle query on any path from an input to the output. We will construct non-adaptive PCPs in Section 3; restricting ourselves to the non-adaptive setting makes things cleaner. For simplicity, we define the notation

$$p_{\text{acc}}(\text{PCP}_V^\pi(x)) \triangleq \Pr_r[\text{PCP}_V(x, r)^\pi \text{ accepts}]$$

to refer to the acceptance probability of the circuit with oracle π on input x . Then, the completeness and soundness constraints can be rephrased as “ $p_{\text{acc}}(\text{PCP}_V^\pi(x)) = 1$ ” and “ $p_{\text{acc}}(\text{PCP}_V^\pi(x)) < 1/3$ ”, respectively. Also for simplicity, we define

$$W_x \triangleq \{\pi \mid p_{\text{acc}}(\text{PCP}_V^\pi(x)) = 1\}$$

to be the set of witnesses to the our PCP verifier.

The standard PCP theorem [6, 7] is equivalent to saying that there exists a PCP system for $\text{NTIME}[T(n)]$ with randomness complexity $O(\log T(n))$, query complexity $O(1)$, and decision complexity $\text{polylog}(T(n))$.

2.4.1 PCP Witnesses

Similarly to Section 2.3, we can formally define when the PCP verifiers have succinct witnesses. We first define the *witness* of a PCP verifier.

¹⁴This is because the number of positions of the oracle to be queried is bounded by $2^{\ell(n)}q$, so we can always reorganize the oracle so that every query is of this particular length.

35:10 Black-Box Constructive Proofs Are Unavoidable

► **Definition 2.9** (PCP witness). *Let $T(n) \geq n$ be time constructible and always a power of 2. Let PCP_V be a PCP verifier for some language $L \in \text{NTIME}[T(n)]$, and $x \in L$ be an YES instance. An oracle $\pi \in \mathcal{B}_{\ell(n)}$ ($\ell(n)$ is the query length of PCP_V on n -bit inputs) is called a PCP witness for x if $p_{\text{acc}}(\text{PCP}_V^\pi(x)) = 1$.*

Let PCP_V be a PCP verifier. Let $\ell(n)$ be the query length to the oracle while $|x| = n$. Similar to a good predicate having \mathcal{C} witnesses, we say that PCP_V has \mathcal{C} PCP witnesses of size $s(n)$ if for any sequence of inputs $\{x_n\}_{n \in \mathbb{N}}$ where x_n is of length n , there is a family of circuits $\{C_n \in \mathcal{B}_{\ell(n)}\}_{n \in \mathbb{N}}$ computable by \mathcal{C} circuits of size $s(n)$, such that for all sufficiently large input length $n \in \mathbb{N}$, if $x_n \in L$, then $\text{tt}(C_n)$ is a PCP witness for x_n .

Fixing a PCP system, we say that $\text{NTIME}[T(n)]$ has \mathcal{C} PCP witnesses of size $s(n)$, if for any good predicate V for some language in $\text{NTIME}[T(n)]$, its corresponding PCP verifier PCP_V has \mathcal{C} PCP witnesses of size $s(n)$.¹⁵

2.4.2 An Efficient PCP System from [8]

In Section 4.2, we will need a PCP construction by Ben-Sasson and Viola [8]. We now formally state it.

► **Theorem 2.10** ([8], Theorem 1.1). *Let $T(n) \geq n$ be time constructible and always a power of 2. Let $V(x, y)$ be a good predicate for some language in $L \in \text{NTIME}[T(n)]$. There exists a PCP verifier PCP_V for L with randomness complexity $\log T(n) + O(\log \log T(n))$, query complexity $\text{polylog}(T(n))$, and decision complexity $\text{polylog}(T(n))$, such that for any input x , the oracle circuit $\text{PCP}_V(x, \cdot)$ is an AC_2^0 oracle circuit.*

We note that the decision procedure of the PCP system above is extremely simple: an AC^0 oracle circuit of depth 2. This will be crucial in our later applications.

3 Technical Ingredient: Smooth, Strong, and Locally-Decodable PCPs

In this section, we construct a PCP system which is smooth, strong, and is local-decodable using low-depth circuits. These properties will be crucial when we prove the equivalence between NEXP lower bounds and the existence of $\text{coRTIME}[\text{polylog}(N)]$ -constructive properties in Section 4.1.

3.1 Definitions

We begin by the notions of smoothness, strongness, and locally decodability.

3.1.1 Smoothness

The notion of smoothness was formally defined by Paradise [28], although the concept appeared implicitly in much earlier works [23, 27]; see [28] for more discussions.

¹⁵Strictly speaking, we should that say $\text{NTIME}[T(n)]$ has \mathcal{C} PCP witnesses of size $s(n)$ with respect to a certain PCP system, as the definition depends on the PCP system we use. We slightly abuse notation here, since the relevant PCP system will always be clear from the context.

► **Definition 3.1** (Smoothness). *A PCP verifier is called smooth if for every input x , the probability of each bit of the proof oracle π being queried remains the same over randomness r . Formally, if we define $A(x, r, i)$ be the predicate that $\text{PCP}_V^\pi(x, r)$ queries the i -th bit of the oracle π , then for every input x and $i_1, i_2 \in \{0, 1\}^{\lceil r \log q \rceil}$, we have*

$$\Pr_r[A(x, r, i_1)] = \Pr_r[A(x, r, i_2)].$$

A PCP system is called smooth if every PCP verifier in its image is smooth.

Sometimes, smoothness is defined by the queries being “marginally uniform”. These two definitions are essentially equivalent, as long as the queries are non-adaptive (see, e.g., Remark 1.4 of [28]). Intuitively, smoothness guarantees that the bits of the proof weigh equally, so that if an oracle π is close in Hamming distance to some correct witness, then the verifier is not likely to distinguish π from a correct witness. This was made formal by Alman and Chen [3]. We include a proof since it is quite short and simple.

► **Lemma 3.2** (Claim 1 of [3]). *Suppose PCP_V is a smooth PCP verifier with query complexity q . For any input x and oracle string π of length $2^{\ell(n)}$, it holds that*

$$p_{\text{acc}}(\text{PCP}_V^\pi(x)) \geq 1 - q \cdot \Delta(\pi, W_x).$$

Proof. Suppose that $x \in L$ is an accepting input and π is an oracle string. Let $y \in W_x$ be a witness with minimum statistical distance with π , and δ be their statistical distance. Let $I = \{i \mid \pi_i = y_i\}$ be the set of indices where π and y coincides, then $|I| = (1 - \delta) \cdot 2^{\ell(n)}$ by the definition of statistical distance. By the union bound and the smoothness of PCP_V , with probability at least $1 - q\delta$ over the randomness r , the oracle circuit $\text{PCP}_V(x, r)$ only queries positions in I . Therefore, $\text{PCP}_V(x, r)$ accepts π with probability at least $1 - q\delta$, which completes the proof. ◀

3.1.2 Strongness

We now introduce the notion of strong PCPs, which complements Lemma 3.2.

► **Definition 3.3** (Strongness). *A PCP verifier PCP_V is called α -strong if, in addition to the PCP constraints, it satisfies the strong soundness defined as below:*

(Strong soundness) *For all $x \in \{0, 1\}^n$ and oracle $\pi \in \{0, 1\}^{2^{\ell(n)}}$, we have*

$$p_{\text{acc}}(\text{PCP}_V^\pi(x)) \leq 1 - \alpha \cdot \Delta(\pi, W_x).$$

We call it strong if it is α -strong for some constant $\alpha \in (0, 1)$. We say that a PCP system is (α) -strong, if every PCP verifier in its image is (α) -strong.

We remark here that strongness implies the traditional soundness, since for any $x \notin L$, the set W_x should be empty, so $\Pr[\text{PCP}_V^\pi(x)] \leq 1 - \alpha$. And indeed, α -strongness is a stronger constraint, since it not only gives an upper bound on the acceptance probability for $x \notin L$, but also for “far from correct” proofs even if $x \in L$.

Completeness and strongness together give us an upper and lower bound on the acceptance probability of an “incorrect” proof for a YES instance. The following corollary can be observed directly from Lemma 3.2 and the definition of strongness.

► **Corollary 3.4.** *Suppose PCP is a smooth and strong PCP system with constant query complexity. Then there exist constants $0 < \alpha < 1 < \beta$ such that the following holds. For any good predicate V for some language L , let PCP_V be the PCP verifier. For any $x \in \{0, 1\}^n$ and oracle $\pi \in \{0, 1\}^{2^{\ell(n)}}$, we have*

$$\alpha \cdot \Delta(\pi, W_x) \leq \Pr_r[\text{PCP}_V(x, r)^\pi \text{ rejects}] \leq \beta \cdot \Delta(\pi, W_x).$$

3.1.3 Local Decodability

We also need our PCP system to be locally decodable. To formally define locally decodable PCPs, we need to first define *canonical* PCPs.

► **Definition 3.5** (Canonical PCP). *A PCP system is called canonical, if for any good predicate $V(x, y)$, along with the PCP verifier PCP_V , there exists a canonical mapping $\Pi_V : \{0, 1\}^n \times \{0, 1\}^{|y|} \rightarrow \{0, 1\}^{2^{\ell(n)}}$, such that for any oracle $\pi \in \{0, 1\}^{2^{\ell(n)}}$, $p_{\text{acc}}(\text{PCP}_V^\pi(x)) = 1$ if and only if $\pi = \Pi_V(x, y)$ for some y with $V(x, y) = 1$. That is, there is a canonical mapping from the witnesses for V to the PCP witnesses for PCP_V .*

Now we define locally decodable PCPs.

► **Definition 3.6** (Locally decodable PCP). *Let PCP be a canonical PCP system. Let $V(x, y)$ be a good predicate where $|y| = 2^{\ell(|x|)}$, let PCP_V be its corresponding PCP verifier, and let $\Pi_V(x, y)$ be the canonical mapping of witnesses. It is called \mathcal{C} -locally decodable within distance δ , if for any input $x \in L$, there exists a collection of $\text{poly}(\ell(|x|))$ -size \mathcal{C} oracle circuits \mathcal{C}_x with input length $\ell(|x|)$, that satisfies the following. Randomly picking a circuit C from the collection \mathcal{C}_x , when C is given oracle access to π with $\Delta(\pi, W_x) < \delta$, if we let $\hat{y} = \text{tt}(C^\pi)$, then $V(x, \hat{y}) = 1$ and $\Delta(\pi, \Pi_V(x, \hat{y})) < \delta$ with constant probability. That is, $\text{tt}(C_x^\pi)$ gives a correct witness for the verifier V whose corresponding PCP witness is close to π . Formally, for any π with $\Delta(\pi, W_x) < \delta$, we have*

$$\Pr_{C \in \mathcal{C}_x} [\hat{y} \leftarrow \text{tt}(C^\pi); V(x, \hat{y}) = 1 \text{ and } \Delta(\pi, \Pi_V(x, \hat{y})) < \delta] > 2/3.$$

A locally decodable PCP provides an efficient way of reconstructing a correct witness for V from an *almost-correct* witness for the PCP verifier. Conversely, if a PCP verifier has succinct witnesses, then the original verifier V has succinct witnesses as well. This property will be crucial when we prove the equivalence between NEXP lower bounds and the existence of $\text{coRTIME}[\text{polylog}(N)]$ -constructive properties.

3.2 The Construction

In this section, we construct a PCP which is smooth, strong, and $\text{AC}^0[2]$ -locally decodable. The construction is based on the smooth and strong PCP by Paradise [28]. We observe that his PCP is locally decodable with an $\text{AC}^0[2]$ circuit. We firstly restate the theorem formally.

► **Theorem 3.7.** *Let $V(x, y)$ be an $\text{NTIME}[T(n)]$ verifier for some $T = T(n) \geq n$, where $n = |x|$ is the input length and $|y| \leq T(n)$ is the proof length. There exists a constant $d \geq 1$ and some PCP verifier PCP_V with randomness complexity $O(\log T)$, query complexity $O(1)$, and decision complexity $\text{polylog}(T)$, such that it is also smooth, strong, canonical, and $\text{AC}_d^0[2]$ -locally decodable within distance $\log^{-4}(T(n))$.*

Proof. Throughout this proof, we assume that the reader is familiar with the PCP in Paradise [28].

We only need to show that the PCP there, which is actually based on the original proof of the PCP theorem ([6, 16]), is $\text{AC}^0[2]$ -locally decodable within distance $\log^{-4}(T(n))$. To do so, we only have to focus on what a correct PCP witness looks like. Without loss of generality, we only need to prove for the particular PCP verifier deciding *circuit satisfiability* (CktSAT)¹⁶, since every good predicate can be reduced to a CktSAT instance

¹⁶The problem CktSAT is defined as, given a circuit C (under some canonical representation), decide whether there exists an input x such that $C(x) = 1$.

without modifying the witnesses. In particular, any good predicate running in time $T(n)$ can be efficiently implemented by a circuit of size $s = O(T(n) \log T(n))$.¹⁷ We will prove that it is $\text{AC}^0[2]$ -locally decodable within distance $\log^{-3}(s)$.

From now on, for simplicity we let n represent the input length to the circuit C (the input length to C is actually $O(T(n))$, but redefining to n will significantly simplify the notation below). In the PCP system of Paradise [28], on an input circuit $C \in \mathcal{B}_n$, any $x \in \{0, 1\}^n$ making $C(x) = 1$ can be transformed into a canonical PCP witness π_x , which consists of several parts $\pi_x = \pi_{x,1} \circ \pi_{x,2} \circ \dots \circ \pi_{x,t}$.¹⁸ Moreover, in the partition of π_x , the length of each part only depends on C and $|x|$, and there exists a universal constant $c \in (0, 1)$ (independent of C and x), such that $|\pi_{x,i}| \geq c|\pi_x|$ for each $i \in [t]$ (and hence $t \leq 1/c$).

We are mostly interested in one of the parts, $\pi_{x,1}$ (since as we will see shortly, from $\pi_{x,1}$ alone we can locally decode the original witness x). We now describe its construction as follows (one may refer to either Section 5.4.1 of [16] and Section 5.4 of [28]).

- (1) Let s be the number of gates (including the input gates) in the circuit C . We can assume without loss of generality that $s = \Theta(n \log n)$ since we are constructing PCPs for good predicates (so the running time of the verifier should be linear in the length of the witnesses).
- (2) Let $m = \lfloor \log s / \log \log s \rfloor$. We can assume without loss of generality that $s = h^m$ for some integer h , since we can always add dummy gates to the circuit C . Note that $h \leq O(\log s)$.
- (3) The input x induces a unique evaluation of the circuit C , which can be represented as a function $A : [s] \rightarrow \{0, 1\}$, where $A(i)$ is the output of the i -th gate in C . For notational convenience, we permute the ordering of gates in C so that for every $i \in [n]$, the i -th gate in C is the i -th input gate of C . Since $s = h^m$, we can identify $[s]$ with $[h]^m$, so that we may view A as a function $[h]^m \rightarrow \{0, 1\}$.
- (4) Let \mathbb{F} be a characteristic-two with order which is a sufficiently large polynomial in h and m (hence $|\mathbb{F}| \leq \text{polylog}(s)$), so that there is an integer ℓ making $|\mathbb{F}| = 2^{2 \cdot 3^\ell}$. We identify $[h]$ with h distinct elements in \mathbb{F} such that the 0 and 1 of $[h]$ map to the 0 and 1 in \mathbb{F} , respectively. There is a unique polynomial $\hat{A} : \mathbb{F}^m \rightarrow \mathbb{F}$, with degree at most $h - 1$ in each variable, which agrees with A on all inputs from $[h]^m$. We can represent \hat{A} by a table of its values, which can be viewed as a string σ of length $|\mathbb{F}|^m$ over the alphabet \mathbb{F} .
- (5) Let $\text{ECC} : \mathbb{F} \rightarrow \{0, 1\}^b$ be an error-correcting code of constant rate and distance, where $2^b = O(|\mathbb{F}|)$. We encode each symbol in the string σ using ECC, and obtain an encoded string $\bar{\sigma} \in (\{0, 1\}^b)^{|\mathbb{F}|^m}$. In other words, the i -th symbol of $\bar{\sigma}$ is $\bar{\sigma}_i = \text{ECC}(\sigma_i)$.
- (6) The proof $\pi_{x,1}$ is then $\kappa(s) = \text{poly}(s)$ copies of $\bar{\sigma}$.

Now for any oracle π that is $\log^{-3}(s)$ -close to a correct witness π_x , we can also partition π into corresponding $\pi_1, \pi_2, \dots, \pi_t$. Since $|\pi_1| \geq c|\pi|$, π_1 should be $1/(c \log^3(s))$ -close to $\pi_{x,1}$. Now given $i \in [n]$ as input, we wish to recover x_i using π_1 . To do so, we only need to evaluate $A(i)$. Let $d = (h - 1)m$ be the total degree of the polynomial \hat{A} . We first present the decoding algorithm in Algorithm 1, and then analyze it.

First, we will prove that Algorithm 1 is a local decoding algorithm; after that, we will argue that the algorithm can be implemented in $\text{AC}^0[2]$. To prove the local decoding property, we observe that since a is a random element in \mathbb{F}^m , the element j should also be uniformly random, so the marginal distribution of each query made by the algorithm (which is $\bar{\mu}_j$ for a uniformly selected copy $\bar{\mu}$) is uniform. By the union bound, with probability at least

¹⁷ Without loss of generality, we work over $T(n)$ -time multitape Turing machines, which can be implemented with circuits of size $O(T(n) \log T(n))$ [29].

¹⁸ For two strings x and y , $x \circ y$ means the concatenation of x and y .

■ **Algorithm 1** Locally-decoding algorithm in $\text{AC}^0[2]$.

Input : Oracle access to π_1 , which is assumed to be $\pi_{x,1}$
Input : An index $i \in [n]$

- 1 Identify i as an element in $[h]^m \subseteq \mathbb{F}^m$;
- 2 Draw $a \in \mathbb{F}^m$ uniformly at random;
- 3 Let e_1, e_2, \dots, e_{d+1} be distinct non-zero elements in \mathbb{F} ;
- 4 **for** $k \leftarrow 1$ **to** $d + 1$ **do**
- 5 Let $j \leftarrow i + a \cdot e_k \in \mathbb{F}^m$;
- 6 Select one copy out of the $\kappa(s)$ supposed copies of $\bar{\sigma}$ in π_1 uniformly at random;
 let $\bar{\mu} \in (\{0, 1\}^b)^{|\mathbb{F}^m|}$ denote the selected copy;
- 7 Query the corresponding positions in π_1 to get $\bar{\mu}_j$;
- 8 Decode μ_j from $\bar{\mu}_j$ using the decoding algorithm for the ECC;
- 9 μ_j should be an element in \mathbb{F} which is supposed to be the value $\widehat{A}(j)$;
- 10 Let $f(e_k) \leftarrow \mu_j$;
- 11 **end**
- 12 Let $\widehat{f}: \mathbb{F} \rightarrow \mathbb{F}$ be the unique polynomial of degree d such that $\widehat{f}(e_k) = f(e_k)$ for all $1 \leq k \leq d + 1$;
- 13 **return** $\widehat{f}(0)$;

$$1 - \frac{(d+1)b}{c \log^3(s)} \geq 1 - o(1)$$

every query q by the algorithm has $(\pi_1)_q = (\pi_{x,1})_q$, that is, π_1 and $\pi_{x,1}$ coincide at this position. So with probability at least $1 - o(1)$, $f(k) = \widehat{A}(i + ak)$ for all $1 \leq k \leq d + 1$. Since \widehat{A} is a polynomial of degree at most d , $\widehat{f}(0)$ must be equal to $\widehat{A}(i)$, hence is equal to $A(i)$.

To prove that the decoding algorithm is implementable in $\text{AC}^0[2]$, we only need to show that the ECC decoding and the computation of $\widehat{f}(0)$ are both implementable in $\text{AC}^0[2]$. For the ECC decoding, note that $b = O(\log \log s)$, so we can simply use a look-up table that is implementable by an AC_2^0 circuit of $2^{O(b)} = \text{polylog}(s)$ size. The computation of \widehat{f} itself does not seem to be computable in $\text{AC}^0[2]$. However, observe that we only need the value $\widehat{f}(0)$, which is equal to the constant term of the polynomial \widehat{f} . By Lagrange interpolation, defining

$$c_k = \frac{\prod_{z \in [d+1] \setminus \{k\}} (-e_z)}{\prod_{z \in [d+1] \setminus \{k\}} (e_k - e_z)},$$
 we have

$$\widehat{f}(0) = \sum_{k=1}^{d+1} f(k) \cdot c_k,$$

To compute $\widehat{f}(0)$, we can hardwire the coefficients c_k in the circuit. What remains is an addition of $d + 1$ products of $f(k)$ with some hardcoded coefficients. Since \mathbb{F} is characteristic two and has order $2^{2 \cdot 3^\ell}$, multiplication and iterated addition are implementable in $\text{AC}_{d'}^0[2]$ for some universal constant $d' \geq 1$, by work of Healy and Viola [18]. This completes the proof. ◀

4 Black-Box Constructive Properties are Unavoidable

4.1 Equivalence with NEXP Lower Bounds

In this subsection, we prove our main result (Theorem 1.4). We begin by recalling our theorem.

► **Theorem 4.1** (Formal version of Theorem 1.4). *Let \mathcal{C} be a polynomial-size circuit class such that any s -size $\text{AC}^0[\oplus] \circ \mathcal{C}$ circuit family can be simulated by an equivalent $\text{poly}(n, s)$ -size \mathcal{C} circuit family. The following are equivalent:*

- (1) $\text{NEXP} \not\subseteq \mathcal{C}$.
- (2) *There is a $\text{P}_{/\log N}$ -constructive property useful against \mathcal{C} .*
- (3) *There is a $\text{QP}_{/\log N}$ -constructive¹⁹ property useful against \mathcal{C} .*
- (4) *There is a $\text{BPTIME}[\text{polylog}(N)]_{/\log N}$ -constructive property useful against \mathcal{C} .*
- (5) *There is a $\text{coRTIME}[\text{polylog}(N)]_{/\log N}$ -constructive property useful against $\text{avg}_{n^{-6}}\text{-}\mathcal{C}$.*

In the theorem, (2) is the standard constructive property considered in Williams [33], (4) corresponds to the black-box constructive property in Fan, Li, and Yang [14], and (5) makes the black-box constructive property both one-sided error and average-case useful. Our theorem says that they are all equivalent to NEXP circuit lower bounds.

Proof. Note that (5) \Rightarrow (4) \Rightarrow (3) is trivial, (2) \Rightarrow (1) follows from Theorem 1.2, and (3) \Rightarrow (2) follows from a direct padding argument. So we only need to prove (1) \Rightarrow (5).

In the rest of the proof, we use the PCP system in Theorem 3.7 that is smooth, strong, canonical, and $\text{AC}^0[2]$ -locally decodable within distance $\log^{-4}(T(n))$. We break the theorem into two claims. The first claim transforms NE witness lower bounds into NE PCP witness lower bounds; the second claim transforms the NE PCP witness lower bound into a constructive property.

▷ **Claim 4.2.** Let $s(n) \in [n^2, 2^n/(10n)]$ be some size function. If NE does not have $\mathcal{C}\text{-SIZE}[s(n)]$ witnesses, then there is some constant $c \in (0, 1)$ such that NE does not have $\text{avg}_{n^{-5}}\text{-}\mathcal{C}\text{-SIZE}[s(cn)^c]$ PCP witnesses.

▷ **Claim 4.3.** Let $s(n) \in [n^2, 2^n/(10n)]$ be some size function. If NE does not have $\text{avg}_{n^{-5}}\text{-}\mathcal{C}\text{-SIZE}[s(n)]$ PCP witnesses, then there is a $\text{coRTIME}[\text{polylog}(N)]_{/\log N}$ -constructive property useful against $\text{avg}_{n^{-6}}\text{-}\mathcal{C}\text{-SIZE}[s(\Omega(n))]$.

Indeed, with these two claims and Lemma 2.7, the implication (1) \Rightarrow (5) directly follows.²⁰ The rest of the proof is devoted to proving the two claims.

Proof of Claim 4.2. If NE does not have $\mathcal{C}\text{-SIZE}[s(n)]$ witnesses, then there is some good predicate V for some language $L \in \text{NE}$, and there is an input sequence $\{x_n\}_{n \in \mathbb{N}}$ with $|x_n| = n$, such that the following holds:

- For every circuit family $C = \{C_n\}_{n \in \mathbb{N}}$ that admits $s(n)$ -size \mathcal{C} circuits²¹, there are infinitely many $n \in \mathbb{N}$ such that $x_n \in L$ and $V(x_n, \text{tt}(C_n)) = 0$.

¹⁹ $\text{QP} = \text{DTIME}[2^{\text{polylog}(n)}]$ is the class *deterministic quasi-polynomial time*.

²⁰ Also note that NE has \mathcal{C} witnesses of size $s(n)$ if and only if for every $L \in \text{NEXP}$ there is a k such that L has \mathcal{C} witnesses of size $s(kn^k)$.

²¹ Suppose $L \in \text{NTIME}[T(n)]$ for some $T(n) = 2^{\Theta(n)}$. Then C_n has $\log T(n) = \Theta(n)$ bits of input. Note that here the size bound on C_n is $s(n)$.

35:16 Black-Box Constructive Proofs Are Unavoidable

Now let us consider the corresponding PCP verifier PCP_V . Recall that we use the notation $\ell(n)$ to denote the oracle query length in the PCP system at inputs of length n . For the class NE here, we should keep in mind that $\ell(n) = \Theta(n)$. We will show that for the same input sequence $\{x_n\}$, for some constant $c \in (0, 1)$, for every function family $\pi = \{\pi_n \in \mathcal{B}_{\ell(n)}\}_{n \in \mathbb{N}}$ that can be $(1 - n^{-5})$ -approximated by $s(cn)^c$ -size \mathcal{C} circuits, there are infinitely many $n \in \mathbb{N}$ such that $x_n \in L$ and π_n is not a witness for x_n .

Suppose (for a contradiction) that for every $c \in (0, 1)$, there is a function family

$$\pi = \{\pi_n \in \mathcal{B}_{\ell(n)}\}_{n \in \mathbb{N}}$$

that can be $(1 - n^{-5})$ -approximated by $s(cn)^c$ -size \mathcal{C} circuits, such that for all sufficiently large $n \in \mathbb{N}$, $x_n \in L$ implies that π_n is a witness for x_n . By the assumption on π , there is another function family $\pi' = \{\pi'_n \in \mathcal{B}_{\ell(n)}\}_{n \in \mathbb{N}}$ that admits $s(cn)^c$ -size \mathcal{C} circuits, such that π'_n and π_n agree on at least a $(1 - n^{-5})$ fraction of inputs, for every $n \in \mathbb{N}$.

We now apply the local decoding algorithm of our PCP system. For all sufficiently large $n \in \mathbb{N}$, if $x_n \in L$, then there is a collection of $\text{poly}(n)$ -size $\text{AC}_d^0[2]$ oracle circuits C_{x_n} (where d is the constant in Theorem 3.7), such that for any π with $\Delta(\pi, W_{x_n}) < n^{-4}$,

$$\Pr_{C \in \mathcal{C}_{x_n}} [V(x_n, \text{tt}(C^\pi)) = 1] > 2/3.$$

So there is some circuit $C_n \in \mathcal{C}_{x_n}$ such that $V(x, \text{tt}(C_n^{\pi'_n})) = 1$. However, since C_n is an $\text{AC}_d^0[2]$ oracle circuit for every n (we can set C_n be the trivial circuit if $x_n \notin L$) and π' admits a \mathcal{C} circuit family of size $s(cn)^c$, so $\{C_n^{\pi'_n}\}_{n \in \mathbb{N}}$ admits a \mathcal{C} circuit family of size $(s(cn)^c)^K$ for some universal constant $K > 1$.²² This implies a contradiction when $c < 1/K$. \triangleleft

Proof of Claim 4.3. By the assumption, there is some constant $d > 1$ and a PCP verifier PCP_V deciding a language $L \in \text{NTIME}[2^{dn}]$, and an input sequence $\{x_n\}_{n \in \mathbb{N}}$ with $|x_n| = n$ such that the following holds:

- For every function family $\pi = \{\pi_n \in \mathcal{B}_{\ell(n)}\}_{n \in \mathbb{N}}$ that is $(1 - n^{-5})$ -approximable by $s(n)$ -size \mathcal{C} circuits, there are infinitely many $n \in \mathbb{N}$ such that $x_n \in L$ and $\Pr_r[\text{PCP}_V^{\pi_n}(x_n, r)] < 1$.

Suppose that on input x of length n , PCP_V has a proof oracle of input length $\ell(n)$. Note that since $L \in \text{NTIME}[2^{dn}]$, $\ell(n) > dn$. Our first step is to show the following:

- For every function family $\pi = \{\pi_n \in \mathcal{B}_{\ell(n)}\}_{n \in \mathbb{N}}$ that can be $(1 - n^{-5})$ -approximated by $s(n)$ -size \mathcal{C} circuits, there are infinitely many $n \in \mathbb{N}$ such that $x_n \in L$ and $\Pr_r[\text{PCP}_V^{\pi_n}(x_n, r)] < 1 - n^{-6}$.

Indeed, if the above does not hold, then there is a function family $\pi = \{\pi_n \in \mathcal{B}_{\ell(n)}\}_{n \in \mathbb{N}}$ that can be $(1 - n^{-5})$ -approximated by $s(n)$ -size \mathcal{C} circuits, such that for every sufficiently large $n \in \mathbb{N}$, $x_n \in L$ implies that $\Pr[\text{PCP}_V^{\pi_n}(x)] \geq 1 - 1/n^6$. By the smoothness and strongness of our PCP (Corollary 3.4), there is a family of correct PCP witnesses $\hat{\pi} = \{\hat{\pi}_n\}_{n \in \mathbb{N}}$ that is $\Theta(n^{-6})$ -close to π . There also exists a function family $\pi' = \{\pi'_n \in \mathcal{B}_{\ell(n)}\}_{n \in \mathbb{N}}$ that admits a $s(n)$ -size \mathcal{C} circuit family and is n^{-6} -close to π . Therefore, π' is also $\Theta(n^{-6})$ -close to $\hat{\pi}$. But this means that $\hat{\pi}$ is a family of correct PCP witness that can be $(1 - \Theta(n^{-6}))$ -approximated by $s(n)$ -size \mathcal{C} circuits, a contradiction to our assumption.

Now we can define a property \mathcal{P} that is useful against $\text{avg}_{n^{-6}\text{-}\mathcal{C}\text{-SIZE}}[s(\Omega(n))]$. The property is defined in Algorithm 2 where $N = 2^n$ and $n = \ell(m)$ for some integer m .

²²This step crucially uses our assumption on the circuit class \mathcal{C} .

■ **Algorithm 2** The $\text{coRTIME}[\text{polylog}(N)]_{/\log N}$ -constructive property \mathcal{P} .

```

Input : Oracle access to a function  $f : \{0, 1\}^{\log N} \rightarrow \{0, 1\}$ 
Advice :  $\alpha_N \in \{0, 1\}^*$  and  $\beta_N \in \{0, 1\}$ 
1 if  $N \neq 2^{\ell(\alpha_N)}$  or  $\beta_N = 0$  then
2 |   accept;
3 else
4 |   for  $i \leftarrow 1$  to  $100 \log^6 N$  do
5 |     Draw a uniformly random  $r$  as the randomness for  $\text{PCP}_V$ ;
6 |     if  $\text{PCP}_V(\alpha_N, r)^f$  rejects then
7 |       reject;
8 |     end
9 |   end
10 |  accept;
11 end

```

On input length $N = 2^{\ell(m)}$, we give x_m as the advice to Algorithm 2. Since $N > 2^{dm}$, the advice length is $m < \log N$. Moreover, it is easy to see that Algorithm 2 runs in $\text{poly}(m) = \text{polylog}(N)$ time.

It remains to verify that the usefulness criterion holds. On input length N , if $N \neq 2^{\ell(m)}$ for all $m \in \mathbb{N}$, then \mathcal{P} accepts every f . If $N = 2^{\ell(m)}$ for some $m \in \mathbb{N}$, if $x_m \notin L$, then \mathcal{P} again accepts every f . (We use an additional bit of advice, β_N , to tell \mathcal{P} whether $x_m \in L$). If $x_m \in L$, then Algorithm 2 accepts any PCP witness f for $\text{PCP}_V(x)$ with certainty (since $x_m \in L$, such f exists). In either case, $\mathcal{P}_N^{\text{yes}} \neq \emptyset$.

On the other hand, for any function family $f = \{f_{\ell(m)} \in \mathcal{B}_{\ell(m)}\}_{m \in \mathbb{N}}$ that can be $(1 - \ell(m)^{-6})$ -approximated by $s(m)$ -size \mathcal{C} circuits, there are infinitely many $m \in \mathbb{N}$ such that $\text{PCP}_V^{f_{\ell(m)}}(x_m)$ rejects with probability at least $1/m^6 > \log^{-6} N$. For those m , Algorithm 2 rejects with probability at least $1/6$. Hence it indeed defines a $\text{coRTIME}[\text{polylog}(N)]_{/\log N}$ -constructive property useful against $\text{avg}_{n^{-6-\mathcal{C}}}\text{-SIZE}[s(\Omega(n))]$. ◁

◀

Note that by combining Lemma 2.7 and Claim 4.2, we can derive average-case witness lower bounds directly from NEXP circuit lower bounds.

4.2 Eliminating the Advice with CAPP Algorithms

In this subsection, we prove Theorem 1.9, showing how a nontrivial CAPP algorithm implies a randomized sublinear-time constructive property without any advice.

► **Theorem 1.9** Let $K \in \mathbb{N}$ be a sufficiently large constant. Let \mathcal{C} be a circuit class and $s(n) \geq n$ be a size parameter. If there is a $2^n/n^{10}$ -time $\text{CAPP}_{0.1}$ algorithm for $s(n)^K$ -size $\text{AC}_2^0 \circ \mathcal{C}$ circuits, then there is a $\text{coRTIME}[\text{polylog}(N)]$ -constructive property useful against $s(n)$ -size \mathcal{C} circuits.

Proof of Theorem 1.9. Take an unary language L in $\text{NTIME}[2^m] \setminus \text{NTIME}[2^m/m]$ [38]. We use the PCP system of Ben-Sasson and Viola [8], which has a PCP verifier PCP_L satisfying the following:

35:18 Black-Box Constructive Proofs Are Unavoidable

(Efficiency) $\text{PCP}_L(x, \cdot)$ is a non-adaptive AC_2^0 oracle circuit that takes $\ell(m)$ bits of randomness, and queries an oracle $\mathcal{O}: \{0, 1\}^{\ell(m)} \rightarrow \{0, 1\}$, where $\ell(m) = m + O(\log m)$. Without loss of generality, we assume that $\ell(m)$ is an increasing function.

(Completeness) If $x \in L$, there is some $\pi: \{0, 1\}^{\ell(m)} \rightarrow \{0, 1\}$ such that $p_{\text{acc}}(\text{PCP}_L^\pi(x)) = 1$.

(Soundness) If $x \notin L$, then for all $\pi: \{0, 1\}^{\ell(m)} \rightarrow \{0, 1\}$, we have $p_{\text{acc}}(\text{PCP}_L^\pi(x)) < 1/3$.

We now claim there is a $\text{coRTIME}[\text{polylog}(N)]$ -constructive property useful against \mathcal{C} . On input length $n \in \mathbb{N}$, let $m \in \mathbb{N}$ be such that $n = \ell(m)$. If there is no such m or $1^m \notin L$, then define $\mathcal{P}_n^{\text{yes}} = \{0, 1\}^n$ and $\mathcal{P}_n^{\text{no}} = \emptyset$. Otherwise, we define the promise property as:

- $\mathcal{P}_n^{\text{yes}}$ contains all oracles $\pi: \{0, 1\}^n \rightarrow \{0, 1\}$ such that $p_{\text{acc}}(\text{PCP}_L^\pi(1^m)) = 1$,
- $\mathcal{P}_n^{\text{no}}$ contains all oracles $\pi: \{0, 1\}^n \rightarrow \{0, 1\}$ such that $p_{\text{acc}}(\text{PCP}_L^\pi(1^m)) < 2/3$.

Suppose the property defined above is not useful against $s(m)$ -size \mathcal{C} circuits. Then there exists a function family $f = \{f_m \in \mathcal{B}_{\ell(m)}\}_{m \in \mathbb{N}}$ such that for all sufficiently large input length m , either $1^m \notin L$, in which case $p_{\text{acc}}(\text{PCP}_L^\pi(1^m)) < 1/3$ for all oracles $\pi \in \mathcal{B}_{\ell(m)}$; or $1^m \in L$ and $\Pr[\text{PCP}_L^{f_m}(1^m)] > 2/3$.

We can then design another algorithm \mathcal{A}^{PCP} that attempts to solve L faster: On input x , reject immediately if $x \neq 1^{|x|}$. Otherwise, guess an $s(m)$ -size $\ell(m)$ -input \mathcal{C} circuit C , and apply the assumed $\text{CAPP}_{0.1}$ algorithm to estimate the acceptance probability $\Pr_{r \in \{0, 1\}^{\ell(m)}}[\text{PCP}_L^C(x, r)]$ within an additive error of 0.1 (note that $\text{PCP}_L^C(x, r)$ for a fixed x is an $\ell(m)$ -input $\text{AC}_2^0 \circ \mathcal{C}$ circuit of $s(m)^K$ size, since K is sufficiently large). Our algorithm accepts if and only if the estimated probability is at least 1/2. The algorithm decides L on all sufficiently large input lengths, and runs in at most $2^m/m$ nondeterministic time, contradicting the fact that $L \notin \text{NTIME}[2^m/m]$. ◀

5 On the Impossibility of Stronger Constructivity Notions

Seeing that $\text{coRTIME}[\text{polylog}(N)]$ -constructive useful properties are equivalent to NEXP lower bounds, one may naturally ask whether the same equivalence holds for even weaker constructivity notions (say, $\text{DTIME}[\text{polylog}(N)]$) instead of $\text{coRTIME}[\text{polylog}(N)]$. In this section, we show that we cannot prove the same equivalence for $\text{RTIME}[\text{polylog}(N)]$, or even $\text{NTIME}[\text{polylog}(N)]$, in place of $\text{coRTIME}[\text{polylog}(N)]$.

First, we show in Section 5.1 an impossibility result: there are no $\text{DTIME}[\text{polylog}(N)]$ -constructive properties which are useful against all polynomial-size CNFs. In fact, we prove a stronger statement that even allows both non-determinism and non-uniformity. In Section 5.2, we complement this impossibility result by showing that if we only want our property to be useful against constant-depth circuits of some *fixed* polynomial size, then super-efficient deterministic constructive (and even large) properties can be obtained from known results on pseudorandom restrictions.

5.1 Non-existence of Deterministic Constructive Properties against Polynomial-size Classes

We first show the general impossibility result.

► **Theorem 5.1.** *Let \mathcal{C} be a circuit class such that all CNFs of t clauses have $\text{poly}(n, t)$ -size \mathcal{C} circuits. There is no $\text{NTIME}[\text{polylog}(N)]_{/\text{polylog}(N)}$ -constructive property useful against $\mathcal{C}\text{-SIZE}[\text{poly}]$.*

Proof. Towards a contradiction, assume there is a constant $c > 1$ and some $O(\log^c(N))$ -size *non-deterministic* circuit C (with an oracle for the input truth table) computing a property useful against \mathcal{C} . We can express the computation of C as a collection of decision trees $\mathcal{D} = \{D_i\}$ of $O(\log^c(N))$ -depth, where for any input $x \in \{0, 1\}^N$,

$$C(x) \text{ accepts} \iff \exists D_i \in \mathcal{D} \text{ such that } D_i(x) \text{ accepts.}$$

By usefulness of the property, there exists some \hat{x} such that $C(\hat{x})$ accepts, so there is also some $\hat{D}_j \in \mathcal{D}$ such that $\hat{D}_j(\hat{x})$ accepts.

Let $I = \{i_1, i_2, \dots, i_k\}$ be the bit positions queried by the decision tree \hat{D}_j on the input \hat{x} . Since \hat{D}_j has depth $O(\log^c(N))$, we have $k \leq O(\log^c(N)) \leq O(n^c)$. We now define a function f that agrees with \hat{x} on I , and is defined to be 1 on all other positions. Formally,

$$f(j) = \begin{cases} \hat{x}_j, & j \in I \\ 1, & \text{otherwise} \end{cases}.$$

Observe that f can be implemented by a CNF with k clauses. Since we assumed that CNFs of size s can be implemented by $\text{poly}(s, n)$ \mathcal{C} circuits, we have $f \in \mathcal{C}\text{-SIZE}[\text{poly}]$. However, as the truth table of f agrees with \hat{x} on I , $\hat{D}_j(\text{tt}(f))$ should accept. This contradicts the assumption that the family of decision trees \mathcal{D} defines a property that is useful against \mathcal{C} . \blacktriangleleft

5.2 DTIME[polylog(N)]-natural Properties against Fixed-polynomial-size Classes from Pseudorandom Restrictions

We can obtain very efficient deterministic properties against classes of circuits for which there is a pseudorandom restriction of short seed length that simplifies circuits from the class. In more detail, we need the pseudorandom restriction to have only logarithmic randomness. Combining with known pseudorandom restrictions in the literature, we can prove Theorem 1.7 and 1.8 from the introduction.

First, we will first formally define what a *pseudorandom restriction lemma* is; then, we will show how to derive DTIME[polylog(N)]-natural properties from such a lemma.

► **Definition 5.2** (Pseudorandom Restriction Lemma). *A class \mathcal{C} is said to admit a pseudorandom restriction lemma with randomness $r(n)$ and freedom $d(n)$, if there exists a $\text{poly}(n)$ -time algorithm \mathcal{A} that on input 1^n takes randomness of length $r(n)$ and produces a restriction $\rho_n \in \{0, 1, *\}^n$ (a.k.a. a partial assignment) to n bits of input, such that*

- (i) *The number of undetermined (unset) variables in ρ_n is at least $d(n)$.*
- (ii) *For every family of circuits $C = \{C_n \in \mathcal{B}_n\}_{n \in \mathbb{N}} \in \mathcal{C}$ and for all sufficiently large n , the function computed by C_n restricted to ρ_n becomes a constant (all-zeros or all-ones) function with constant probability. Formally, for all sufficiently large n ,*

$$\Pr_{z \in \{0, 1\}^{r(n)}} [\rho_n \leftarrow \mathcal{A}(1^n, z) \text{ such that } C_n|_{\rho_n} \text{ is a constant}] \geq \Omega(1).$$

► **Theorem 5.3.** *Let \mathcal{C} be any circuit class. If \mathcal{C} admits a pseudorandom restriction lemma with randomness $r(n) = O(\log n)$ and freedom $d(n) = 2 \log \log n$, then there is a DTIME[polylog(N)]-natural property useful against \mathcal{C} .*

Proof. Without loss of generality, we can assume that the number of undetermined variables is always exactly $d(n)$, as we can arbitrarily restrict variables until the number of undetermined variables becomes $d(n)$. Suppose that the algorithm for constructing the pseudorandom restriction is $\mathcal{A}(1^n, z)$, where $z \in \{0, 1\}^{r(n)}$ is the randomness. Our property is simple:

given oracle access to a function f , enumerate over all $2^{r(n)} = \text{poly}(n)$ possible seeds z , and test if any of the restrictions $\rho_z \leftarrow \mathcal{A}(1^n, z)$ makes f constant by enumerating all possible assignments to the inputs left unset by ρ_z (note there are only $2^{d(n)} = \text{polylog}(n)$ such assignments). We formally describe this procedure in Algorithm 3.

■ **Algorithm 3** DTIME[polylog(N)]-natural property from pseudorandom restrictions.

Input : Oracle access to a function $f : \{0, 1\}^n \rightarrow \{0, 1\}$

- 1 **for** $z \in \{0, 1\}^{r(n)}$ **do**
- 2 Let $\rho_z \leftarrow \mathcal{A}(1^n, z)$;
- 3 **if** $f|_{\rho_z}$ *is a constant function* **then**
- 4 Reject;
- 5 **end**
- 6 **end**
- 7 Accept;

The algorithm clearly runs in $\text{poly}(n) = \text{polylog}(N)$ time. We only need to show that the largeness and usefulness criteria are satisfied by the algorithm. To prove usefulness, consider any function $f \in \mathcal{C}$. Since \mathcal{A} is a pseudorandom restriction as defined in Definition 5.2, by property (ii) there exists some randomness $z \in \{0, 1\}^{r(n)}$ making $f|_{\rho}$ constant, so our algorithm must reject. To establish largeness, we upper bound the probability that Algorithm 3 rejects a random function. This can be done by a simple union bound:

$$\Pr_{f \in \{0,1\}^n \rightarrow \{0,1\}} [\text{Algorithm 3 rejects}] \leq 2^r \cdot 2 \cdot 2^{-2^d} = 2^{O(\log n) - \log^2 n} < o(1). \quad \blacktriangleleft$$

Applying known pseudorandom restriction lemmas with the desired parameters, we can obtain DTIME[polylog(N)]-natural properties useful against certain classes. For example, Goldreich and Wigderson [15] proved that for any constant $c > 0$, the class of AC^0 circuits of size at most n^c admits a pseudorandom restriction lemma with randomness $O(\log n)$ and freedom $n^{\Omega(1)}$.²³ Therefore by Theorem 5.3, there is a DTIME[polylog(N)]-natural properties useful against AC^0 circuits of size n^c (hence proving Theorem 1.8). Applying the pseudorandom restriction of [14] to Theorem 5.3, we obtain a DTIME[polylog(N)]-natural property useful against probabilistic DeMorgan formulas of $n^{2-\varepsilon}$ size (hence proving Theorem 1.7).

6 A Bootstrapping Result For Black-Box Natural Properties

We prove general bootstrapping results on the existence of $\text{coRTIME}[\text{polylog}(N)]$ -constructive *natural properties* in this section. In particular, we show that if a circuit class is capable of efficiently implementing almost-universal hash functions, then black-box natural properties useful against fixed polynomial size imply such natural properties against all polynomial-size circuits. We follow an idea of [14] for reducing the complexity of computing PRFs: “kernelize” the oracle with an almost universal hash function. To begin, we define the notion of almost universal hash functions.

²³Technically, Theorem 3.2 of [15] only shows that with high probability the restricted function is a constant-junta (i.e., the remaining function after the restriction depends on at most $O(1)$ variables). Note that we can always randomly restrict an additional fraction of the inputs to cover all the $O(1)$ dependent inputs with constant probability.

► **Definition 6.1** (Almost universal hash function). *A family of hash function is a collection $\mathcal{H} = \{\mathcal{H}_n\}_{n \in \mathbb{N}}$, where each \mathcal{H}_n is defined as a distribution of functions from $\{0, 1\}^n$ to $\{0, 1\}^m$ with output length $m = m(n)$. \mathcal{H} is called ε -almost universal if for all sufficiently large n and any two $x, y \in \{0, 1\}^n$ with $x \neq y$,*

$$\Pr_{h \in \mathcal{H}_n} [h(x) = h(y)] \leq \varepsilon(n).$$

We say that \mathcal{H} is almost-universal, if it is $n^{-\alpha(n)}$ -almost universal for some $\alpha(n) = \omega(1)$.

We say that \mathcal{H} is computable by \mathcal{C} circuits of size $s(n)$, if every sequence of functions from the collection $h = \{h_n \in \mathcal{H}_n\}_{n \in \mathbb{N}}$ admits a \mathcal{C} circuit family of size $s(n)$.

We say that \mathcal{H} is uniform if the distribution \mathcal{H}_n is samplable in $\text{poly}(n)$ time.

We now prove that efficient \mathcal{C} -circuit constructions of almost-universal hash functions imply bootstrapping results for black-box natural properties against \mathcal{C} -circuits. Recall that a property is $\text{BPTIME}[\text{polylog}(N)]$ -natural if it is $\text{BPTIME}[\text{polylog}(N)]$ -constructive and large, and a property is a.e.-useful against a circuit class \mathcal{C} if it is useful against \mathcal{C} for all but finitely many input lengths. We say that a circuit class \mathcal{C} is *efficiently closed under composition*, if given two circuit families $f = \{f_n : \{0, 1\}^n \rightarrow \{0, 1\}^{m_1(n)}\}_{n \in \mathbb{N}}$ and $g = \{g_n : \{0, 1\}^{m_1(n)} \rightarrow \{0, 1\}^{m_2(n)}\}_{n \in \mathbb{N}}$ computable by \mathcal{C} circuits of size $s_1(n)$ and $s_2(n)$ respectively, their composition $g \circ f = \{g_{m_1(n)} \circ f_n\}_{n \in \mathbb{N}}$ can be computed by \mathcal{C} circuits of size $s_1(n) + s_2(m_1(n))$. Note that all “typical” classes, such as P/poly , NC^1 , TC^0 , and AC^0 , satisfy this property.

► **Theorem 6.2.** *Let \mathcal{C} be a circuit class efficiently closed under composition and let $s(n)$ be a polynomial in n . Suppose for every $c \in (0, 1)$, there is a family of almost-universal hash functions with output length $m = m(n) = n^c$ that is uniform and computable by \mathcal{C} circuits of size $s(n)$. Then the following holds:*

If there are $\text{BPTIME}[\text{polylog}(N)]$ -natural properties a.e.-useful against \mathcal{C} circuits of size $s(n) + \sqrt{n}$, then for all $k \geq 1$, there are $\text{BPTIME}[\text{polylog}(N)]$ -natural properties a.e.-useful against \mathcal{C} circuits of n^k size.

Proof. Assume there is a $\text{BPTIME}[\text{polylog}(N)]$ -natural property $\mathcal{P} = \{(\mathcal{P}_n^{\text{yes}}, \mathcal{P}_n^{\text{no}})\}_{n \in \mathbb{N}}$ that is a.e.-useful against \mathcal{C} circuits of size $s(n) + \sqrt{n}$. That is, there is a constant $c > 0$ and an algorithm $\mathcal{A}^f(1^n, r)$ with randomness $|r| \leq \text{poly}(n)$ and oracle access to $f \in \mathcal{B}_n$, running in n^c time, such that

- For every $f \in \mathcal{P}_n^{\text{yes}}$, $\Pr_r [\mathcal{A}^f(1^n, r) \text{ accepts}] \geq 2/3$.
- For every $f \in \mathcal{P}_n^{\text{no}}$, $\Pr_r [\mathcal{A}^f(1^n, r) \text{ accepts}] \leq 1/3$.

By the largeness criterion, there is a polynomial $p(n)$ such that $\Pr_{f \in \mathcal{B}_n} [f \in \mathcal{P}_n^{\text{yes}}] \geq 1/p(n)$.

Let $k \geq 1$ be an arbitrary constant such that $n^k > s(n)$ for all sufficiently large n . We now define a $\text{BPTIME}[\text{polylog}(N)]$ -natural property that is a.e.-useful against \mathcal{C} circuits of size n^k . The property is defined by Algorithm 4. In particular, let Algorithm 4 be denoted by $\widehat{\mathcal{A}}^f(1^n, r)$ where $r = (r_0, h)$ is its internal randomness. We define a (promise) property $\widehat{\mathcal{P}} = \{(\widehat{\mathcal{P}}_n^{\text{yes}}, \widehat{\mathcal{P}}_n^{\text{no}})\}_{n \in \mathbb{N}}$ as follows: for a function $f \in \mathcal{B}_n$,

- $f \in \widehat{\mathcal{P}}_n^{\text{yes}}$ if $\Pr_r [\widehat{\mathcal{A}}^f(1^n, r) \text{ accepts}] \geq 5/8$.
- $f \in \widehat{\mathcal{P}}_n^{\text{no}}$ if $\Pr_r [\widehat{\mathcal{A}}^f(1^n, r) \text{ accepts}] \leq 3/8$.

We now prove that the property $\widehat{\mathcal{P}}$ satisfies the desired conditions.

35:22 Black-Box Constructive Proofs Are Unavoidable

■ **Algorithm 4** BPTIME[polylog(N)]-natural property a.e.-useful against n^k size \mathcal{C} circuits.

-
- Input** : 1^n
Oracle : oracle access to a function $f : \{0, 1\}^n \rightarrow \{0, 1\}$
- 1 Let $\hat{n} = n^{2k}$, and \mathcal{H} be an almost-universal hash function with output length $m(n) = n^{1/(2k)}$;
 - 2 Randomly sample a function $h \leftarrow \mathcal{H}_{\hat{n}}$ in $\text{poly}(\hat{n}) = \text{poly}(n)$ time;
 - 3 Define $f' : \{0, 1\}^{\hat{n}} \rightarrow \{0, 1\}$ as $f'(x) = f(h(x))$;
 - 4 Simulate $\mathcal{A}^{f'}(1^{\hat{n}}, r_0)$ on random r_0 , and accept if and only if the simulation accepts;
-

(Constructivity) Showing $\widehat{\mathcal{P}}$ is constructive amounts to showing that Algorithm 4 runs in $\text{poly}(\log N)$ time. This follows because \mathcal{A} is constructive (it can be implemented in $\text{poly}(\log N)$ time).

(Largeness) To prove largeness, we need to show that $\Pr_{f \in \mathcal{B}_n}[f \in \widehat{\mathcal{P}}_n^{\text{yes}}] \geq 1/\text{poly}(n)$ for all sufficiently large n .

Fix an input length n and randomness r_0 . Assume that over oracle f' , the distinct positions of the oracle f' queried by $\mathcal{A}(1^{\hat{n}}, r_0)$ are z_1, z_2, \dots, z_t where $t \leq \hat{n}^c = n^{ck}$. Since \mathcal{H} is almost universal, by a simple union bound,

$$\Pr_{h \in \mathcal{H}_{\hat{n}}} [\forall i \neq j, h(z_i) \neq h(z_j)] \geq 1 - n^{2ck - \omega(1)} \geq 1 - n^{-\omega(1)}.$$

Let $\mathcal{E}_n(h, r_0, f')$ denote the event that for all $i \neq j$, $h(z_i) \neq h(z_j)$. Then we calculate that

$$\begin{aligned} & \mathbb{E}_{f \in \mathcal{B}_n, r_0, h \in \mathcal{H}_{\hat{n}}} [\widehat{\mathcal{A}}^f(1^n, r_0, h) \text{ accepts}] \\ & \geq \mathbb{E}_{f \in \mathcal{B}_n, r_0, h \in \mathcal{H}_{\hat{n}}} [\widehat{\mathcal{A}}^f(1^n, r_0, h) \text{ accepts} \wedge \mathcal{E}_n(h, r_0, f')] \\ & \geq \mathbb{E}_{r_0, f' \in \mathcal{B}_{\hat{n}}} [\mathcal{A}^f(1^{\hat{n}}, r_0) \text{ accepts}] - n^{-\omega(1)} \\ & \geq \frac{2}{3p(n)} - n^{-\omega(1)}. \end{aligned}$$

Applying Markov's inequality, we find that

$$\Pr_{f \in \mathcal{B}_n} \left[\Pr_{r_0, h \in \mathcal{H}_{\hat{n}}} [\widehat{\mathcal{A}}^f(1^n, r_0, h) \text{ accepts}] \geq 5/8 \right] \geq \Omega(1/p(n)),$$

hence proving the largeness criterion.

(Usefulness) To prove usefulness, we consider any sufficiently large input length n , and any function $f \in \mathcal{B}_n$ that admits a \mathcal{C} circuit of size n^k . For every choice of $h \in \mathcal{H}_{\hat{n}}$, by our assumptions on \mathcal{C} , the function $f'(x) = f(h(x))$ is computable by a \mathcal{C} circuit of size $s(\hat{n}) + n^k = s(\hat{n}) + \sqrt{\hat{n}}$. By the usefulness of \mathcal{A} , we have that $\Pr_{r_0} [\mathcal{A}^{f'}(1^{\hat{n}}, r_0) \text{ accepts}] \leq 1/3$. As this inequality holds for every choice of h , we have

$$\Pr_{r_0, h \in \mathcal{H}_{\hat{n}}} [\widehat{\mathcal{A}}^f(1^n, r_0, h) \text{ accepts}] \leq 1/3,$$

hence $f \in \mathcal{P}_n^{\text{no}}$. ◀

References

- 1 Eric Allender. Cracks in the defenses: Scouting out approaches on circuit lower bounds. In *Computer Science - Theory and Applications, Third International Computer Science Symposium in Russia, CSR 2008, Moscow, Russia, June 7-12, 2008, Proceedings*, volume 5010 of *Lecture Notes in Computer Science*, pages 3–10. Springer, 2008. doi:10.1007/978-3-540-79709-8_2.
- 2 Eric Allender and Michal Koucký. Amplifying lower bounds by means of self-reducibility. *J. ACM*, 57(3):14:1–14:36, 2010.
- 3 Josh Alman and Lijie Chen. Efficient construction of rigid matrices using an NP oracle. In *Proc. 60th FOCS*, pages 1034–1055. IEEE Comp. Soc., 2019. doi:10.1109/FOCS.2019.00067.
- 4 Alexander E Andreev. On a method for obtaining more than quadratic effective lower bounds for the complexity of π -schemes. *Moscow Univ. Math. Bull.*, 42(1):63–66, 1987.
- 5 Sanjeev Arora and Boaz Barak. *Computational Complexity - A Modern Approach*. Cambridge University Press, 2009. URL: <http://www.cambridge.org/catalogue/catalogue.asp?isbn=9780521424264>.
- 6 Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *J. ACM*, 45(3):501–555, 1998. doi:10.1145/278298.278306.
- 7 Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: A new characterization of NP. *J. ACM*, 45(1):70–122, 1998. doi:10.1145/273865.273901.
- 8 Eli Ben-Sasson and Emanuele Viola. Short PCPs with projection queries. In *Proc. 41st Internat. Colloq. on Automata, Languages and Programming (ICALP'14)*, pages 163–173. Springer, 2014. doi:10.1007/978-3-662-43948-7_14.
- 9 Marco L. Carmosino, Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova. Agnostic learning from tolerant natural proofs. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2017, August 16-18, 2017, Berkeley, CA, USA*, volume 81 of *LIPICs*, pages 35:1–35:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPICs.APPROX-RANDOM.2017.35.
- 10 Brynmor Chapman and Ryan Williams. The circuit-input game, natural proofs, and testing circuits with data. In *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science, ITCS 2015, Rehovot, Israel, January 11-13, 2015*, pages 263–270. ACM, 2015. doi:10.1145/2688073.2688115.
- 11 Lijie Chen, Shuichi Hirahara, Igor Carboni Oliveira, Ján Pich, Ninad Rajgopal, and Rahul Santhanam. Beyond natural proofs: Hardness magnification and locality. In *11th Innovations in Theoretical Computer Science Conference, ITCS, pages 70:1–70:48, 2020*. doi:10.4230/LIPICs.ITCS.2020.70.
- 12 Lijie Chen, Ce Jin, and R. Ryan Williams. Sharp threshold results for computational complexity. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 1335–1348. ACM, 2020. doi:10.1145/3357713.3384283.
- 13 Timothy Y. Chow. Almost-natural proofs. *J. Comput. Syst. Sci.*, 77(4):728–737, 2011. doi:10.1016/j.jcss.2010.06.017.
- 14 Zhiyuan Fan, Jiayu Li, and Tianqi Yang. The exact complexity of pseudorandom functions and tight barriers to lower bound proofs. *Electron. Colloquium Comput. Complex.*, page 125, 2021. URL: <https://eccc.weizmann.ac.il/report/2021/125>, arXiv:TR21-125.
- 15 Oded Goldreich and Avi Wigderson. On derandomizing algorithms that err extremely rarely. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 109–118. ACM, 2014. doi:10.1145/2591796.2591808.
- 16 Prahladh Harsha. *Robust PCPs of Proximity and Shorter PCPs*. PhD thesis, Massachusetts Institute of Technology, 2004.
- 17 Johan Håstad. The shrinkage exponent of de morgan formulas is 2. *SIAM J. Comput.*, 27(1):48–64, 1998. doi:10.1137/S0097539794261556.

- 18 Alexander Healy and Emanuele Viola. Constant-depth circuits for arithmetic in finite fields of characteristic two. In *STACS 2006, 23rd Annual Symposium on Theoretical Aspects of Computer Science, Marseille, France, February 23-25, 2006, Proceedings*, volume 3884 of *Lecture Notes in Computer Science*, pages 672–683. Springer, 2006. doi:10.1007/11672142_55.
- 19 Matthias Krause and Stefan Lucks. Pseudorandom functions in tc^0 and cryptographic limitations to proving lower bounds. *Comput. Complex.*, 10(4):297–313, 2001. doi:10.1007/s000370100002.
- 20 Richard J. Lipton and Ryan Williams. Amplifying circuit lower bounds against polynomial time, with applications. *Comput. Complex.*, 22(2):311–343, 2013.
- 21 Dylan M. McKay, Cody D. Murray, and R. Ryan Williams. Weak lower bounds on resource-bounded compression imply strong separations of complexity classes. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019*, pages 1215–1225. ACM, 2019. doi:10.1145/3313276.3316396.
- 22 Eric Miles and Emanuele Viola. Substitution-permutation networks, pseudorandom functions, and natural proofs. *J. ACM*, 62(6):46:1–46:29, 2015. doi:10.1145/2792978.
- 23 Dana Moshkovitz and Ran Raz. Two-query PCP with subconstant error. *J. ACM*, 57(5):29:1–29:29, 2010. doi:10.1145/1754399.1754402.
- 24 Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudo-random functions. *J. ACM*, 51(2):231–262, 2004. doi:10.1145/972639.972643.
- 25 Igor Carboni Oliveira, Ján Pich, and Rahul Santhanam. Hardness magnification near state-of-the-art lower bounds. In *34th Computational Complexity Conference, CCC 2019*, pages 27:1–27:29, 2019. doi:10.4230/LIPIcs.CCC.2019.27.
- 26 Igor Carboni Oliveira and Rahul Santhanam. Hardness magnification for natural problems. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018*, pages 65–76, 2018. doi:10.1109/FOCS.2018.00016.
- 27 Christos H. Papadimitriou and Mihalis Yannakakis. Optimization, approximation, and complexity classes. *J. Comput. Syst. Sci.*, 43(3):425–440, 1991. doi:10.1016/0022-0000(91)90023-X.
- 28 Orr Paradise. Smooth and strong pcps. *Comput. Complex.*, 30(1):1, 2021. doi:10.1007/s00037-020-00199-3.
- 29 Nicholas Pippenger and Michael J. Fischer. Relations among complexity measures. *J. ACM*, 26(2):361–381, 1979. doi:10.1145/322123.322138.
- 30 Alexander A. Razborov and Steven Rudich. Natural proofs. *J. Comput. Syst. Sci.*, 55(1):24–35, 1997. doi:10.1006/jcss.1997.1494.
- 31 Avishay Tal. Shrinkage of de morgan formulae by spectral techniques. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 551–560. IEEE Computer Society, 2014. doi:10.1109/FOCS.2014.65.
- 32 Avishay Tal. Formula lower bounds via the quantum method. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 1256–1268. ACM, 2017. doi:10.1145/3055399.3055472.
- 33 R. Ryan Williams. Natural proofs versus derandomization. *SIAM J. Comput.*, 45(2):497–529, 2016. doi:10.1137/130938219.
- 34 R. Ryan Williams. New algorithms and lower bounds for circuits with linear threshold gates. *Theory Comput.*, 14(1):1–25, 2018. doi:10.4086/toc.2018.v014a017.
- 35 Ryan Williams. Improving exhaustive search implies superpolynomial lower bounds. *SIAM J. Comput.*, 42(3):1218–1244, 2013. doi:10.1137/10080703X.
- 36 Ryan Williams. Natural proofs versus derandomization. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 21–30. ACM, 2013. doi:10.1145/2488608.2488612.
- 37 Ryan Williams. Nonuniform ACC circuit lower bounds. *J. ACM*, 61(1):2, 2014. doi:10.1145/2559903.
- 38 Stanislav Žák. A Turing machine time hierarchy. *Theoretical Computer Science*, 26(3):327–333, 1983.