Electronic Colloquium on Computational Complexity, Report No. 38 (2023)

Indistinguishability Obfuscation, Range Avoidance, and Bounded Arithmetic

Rahul Ilango[*]
MIT
rilango@mit.edu

Jiatu Li
IIIS, Tsinghua University
ljt714285@gmail.com

Ryan Williams[†]
MIT
rrw@mit.edu

Abstract

The *range avoidance problem* (denoted by AVOID) asks to find a string outside of the range of a given circuit $C : \{0,1\}^n \to \{0,1\}^m$, where $m > n$. Although at least half of the strings of length m are correct answers, it is not clear how to *deterministically* find one. Recent results of Korten (FOCS'21) and Ren, Wang, and Santhanam (FOCS' 22) show that efficient deterministic algorithms for AVOID would have far-reaching consequences, including strong circuit lower bounds and explicit constructions of combinatorial objects (e.g., Ramsey graphs, extractors, rigid matrices). This strongly motivates the question: does an efficient deterministic algorithm for AVOID actually exist?

In this work, we prove under the existence of subexponentially secure *indistinguishability obfuscation* ($i\mathcal{O}$) that deterministic polynomial-time algorithms for AVOID imply $\mathsf{NP} = \mathsf{coNP}$. Combining this with Jain, Lin, and Sahai's recent breakthrough construction of $i\mathcal{O}$ from well-founded assumptions (STOC'21, EUROCRYPT'22), we provide the first plausible evidence that AVOID has no efficient deterministic algorithm. Moreover, we also prove the hardness of AVOID based on polynomially-secure $i\mathcal{O}$ and a weaker variant of the Nondeterministic Exponential Time Hypothesis (NETH).

Extending our techniques, we prove a surprising separation in *bounded arithmetic*, conditioned on similar assumptions. Assuming subexponentially secure $i\mathcal{O}$ and coNP is not infinitely often in AM, we show that AVOID has no deterministic polynomial-time algorithm even when we are allowed $O(1)$ queries to an oracle that can invert the given input circuit on an arbitrarily chosen m-bit string. It follows that the *dual Weak Pigeonhole Principle*, the combinatorial principle underlying AVOID, is not provable in Cook's theory PV_1. This gives (under plausible assumptions) the first separation of Cook's theory PV_1 for polynomial-time reasoning and Jeřábek's theory APC_1 for probabilistic polynomial-time reasoning.

[*] Supported by an NSF Graduate Research Fellowship and NSF CCF-1909429.
[†] Supported by NSF CCF-1909429.

ISSN 1433-8092

Contents

1 Introduction

Given a circuit C mapping n-bit inputs to m-bit outputs, where $m > n$, at least half of the possible m-bit strings are never output by C. How efficiently can we find such a string? This meta-computational problem is known as Range Avoidance:

Search Problem: Range Avoidance (a.k.a. Avoid)
Input: A Boolean circuit C with n inputs, and $m > n$ outputs.
Output: A string $y \in \{0,1\}^m$ such that for all $x \in \{0,1\}^n$, $C(x) \neq y$.

There is a simple randomized algorithm for Avoid: a uniformly random $y \in \{0,1\}^m$ will be outside the range of C with probability at least $1 - 2^n/2^m \geq 1/2$. Is there an efficient *deterministic* algorithm for Avoid?

This question is especially intriguing because it does not seem clear what the answer should be. Indeed, Ren-Wang-Santhanam [RSW22] remark

> "It is unknown whether Avoid $\in$ FNP, Avoid $\in$ FP, or their negations are implied by any plausible assumptions. As far as we know, we do not even have a good idea of what the 'ground truth' should be."

(FNP and FP refer to the function versions of NP and P respectively.)

Under a plausible derandomization assumption, there is a deterministic polynomial-time algorithm for Avoid *given access to an* NP*-oracle* [Kor21]. In randomized polynomial time with an NP oracle, one can repeatedly sample a uniformly random $y \in \{0,1\}^m$ and verify it is not in the range with the NP oracle. This process can be derandomized, assuming E^{NP} requires $2^{\Omega(n)}$-sized circuits [KvM02]. In fact, Korten [Kor21] shows that Avoid is in $\mathsf{FP}^{\mathsf{NP}}$ *if and only if* E^{NP} does not have $2^{o(n)}$-sized circuits infinitely often, so finding a deterministic algorithm for Avoid with a SAT oracle is equivalent to proving circuit lower bounds.

1.1 Background

Implications of Deterministic Algorithms for Range Avoidance

Kleinberg, Korten, Mitropolsky, and Papadimitriou [KKMP21] initiated the study of Avoid (in their notation, Avoid is the problem α-Empty for $\alpha \geq 1$). They showed that various explicit construction problems can be reduced to Avoid, including the problem Complexity, of outputting a function with high A-oracle circuit complexity, given the truth table of the oracle A.[1]

In follow-up work, Korten [Kor21] convincingly demonstrated that deterministic algorithms for Avoid would have significant consequences for circuit complexity and combinatorics. For example, letting C be a $\mathsf{poly}(2^\ell)$-size circuit which takes as input descriptions of $2^{\ell/10}$-size circuits and outputs their 2^ℓ-bit truth tables, solving Range Avoidance on such C amounts to finding truth tables of *high* circuit complexity, a task that is widely believed to be solvable in deterministic polynomial time.[2] Korten extended this observation considerably, showing a deterministic Avoid algorithm would imply deterministic constructions of a variety of other objects (e.g., Ramsey graphs, extractors, rigid matrices) where a random choice suffices, but explicit constructions are longstanding open problems.

Ren-Santhanam-Wang [RSW22] found further striking consequences of deterministic efficient Avoid algorithms. Among many other results, they show that a polynomial-time algorithm for NC^0_4 circuits (NC^0_k denotes circuits in which each output only depends on k inputs) with stretch $m = n + n^{o(1)}$ would already yield functions in E that require circuits of depth $n^{1-o(1)}$, a major open problem in circuit complexity. Guruswami-Lyu-Wang [GLW22] improve upon several reductions of Ren-Santhanam-Wang, and also show that Range Avoidance is in fact solvable in

[1] Kleinberg, Korten, Mitropolsky, and Papadimitriou [KKMP21] also showed that an extremely low-stretch variant of Avoid, where one is given a circuit mapping from $[2^n - 1]$ to $[2^n]$, is NP-hard (their Theorem 1). However, because the stretch of this variant is exponentially small, the complexity of the problem is quite different: for example, this version can't be easily solved with randomness, as the total number of inputs and outputs only differ by one.

[2] In more detail, constructing a 2^n-bit truth table with circuit complexity $2^{\Omega(n)}$ in $\mathsf{poly}(2^n)$ time is equivalent to showing that E requires $2^{\Omega(n)}$ circuit complexity, which is the main hypothesis powering the famous pseudorandom generators for $\mathsf{BPP} = \mathsf{P}$ [IW97, STV01].

deterministic polynomial time for NC_2^0 circuits. Gajulapalli-Golovnev-Nagargoje-Saraogi [GGNS23] show that a deterministic polynomial-time algorithm for AVOID on NC_3^0 circuits with $m = n + n^{2/3}$ implies breakthrough explicit constructions of rigid matrices. They also give deterministic polynomial-time algorithms for AVOID on NC_k^0 circuits with stretch $m \geq n^{k-1}/\log n$.

What to Believe? Arguments and Counterarguments.

All the above results underscore the significance of finding nontrivial algorithms for RANGE AVOIDANCE and the importance of understanding how difficult RANGE AVOIDANCE really is. **Should we believe RANGE AVOIDANCE is in FP (the class of polynomial-time computable functions), or not?** To illustrate the depth of this question, we briefly consider some arguments and counterarguments.

From one point of view, it is natural to imagine a world in which AVOID $\in \mathsf{FP}$. From Korten [Kor21], we already know that if E^{NP} doesn't have subexponential-size circuits, then AVOID is in $\mathsf{FP}^{\mathsf{NP}}$. In light of this, it seems natural to believe that under the stronger assumption that E (without an NP oracle) doesn't have subexponential-size circuits (that is, the widely-believed conjecture $\mathsf{E} \not\subseteq \text{io-}\mathsf{SIZE}(2^{o(n)})$ [IW97]), one could show AVOID is in FP (without an NP oracle). Furthermore, standard methods from pseudorandomness [KvM02] imply that there is a polynomial-time constructible *hitting set* for AVOID, assuming (for example) that E does not have subexponential-size SAT oracle circuits. That is, under plausible hypotheses, one *can* generate in FP a polynomial-size set $S_{s,m} \subseteq \{0,1\}^m$ such that for *every* circuit C of size s with m outputs, at least one $y \in S_{s,m}$ is not an output of C (see Appendix A for details). However, *checking* which y is a non-output apparently requires an NP oracle.

To add to these points, the existence of a randomized algorithm for AVOID seems to preclude a range of approaches to ruling out a deterministic (FP) algorithm for AVOID. For example, in Appendix B we present a barrier result *against* proving AVOID $\notin \mathsf{FP}$ using standard *black-box* randomized Turing reductions, which exploits the fact that a random string is a correct answer with high probability. Still, the fact that AVOID has a fast randomized algorithm does not necessarily mean that we should believe it has a fast deterministic one. For an extreme example, one can easily sample strings of high Kolmogorov complexity *with randomness*, but one provably cannot do this deterministically at all: sufficiently long strings generated by fixed deterministic algorithms always have low Kolmogorov complexity [Sip97, Chapter 6.4].

Indeed, one might believe AVOID $\notin \mathsf{FP}$ because the opposite may seem "too good" to be true. The prior work mentioned above shows that, if AVOID $\in \mathsf{FP}$, there are many interesting consequences for lower bounds and explicit constructions. However, as we expect all of those consequences to actually be *true*, these results alone don't give a strong argument that AVOID $\notin \mathsf{FP}$. Rather, they indicate that the opposite may be hard to prove, as it would have significant consequences.

Another intuition for the difficulty of AVOID is that the generality of the problem allows for more power than merely generating varieties of hard functions and special combinatorial objects, each of which correspond to specific *structured* instances of AVOID. Solving AVOID for *all* circuits, even arbitrary ones whose descriptions may be very "scrambled" and complex, could be far more powerful than the generation of interesting mathematical objects.

In summary, it was entirely unclear whether RANGE AVOIDANCE should be solvable in deterministic polynomial time, or not. In this paper, we shall give evidence that is not, starting from the intuition of the previous paragraph.

Indistinguishability Obfuscation

Before stating our results, we take a quick detour to discuss one of our main tools: *Indistinguishability Obfuscation* ($i\mathcal{O}$), a notion first defined by Barak *et al.* [BGI+01, BGI+12]. Roughly speaking, an $i\mathcal{O}$ is a polynomial-time probabilistic algorithm that given a circuit C, outputs an "obfuscated" circuit $i\mathcal{O}(C)$ computing the same function. The security guarantee of $i\mathcal{O}$ is that, for any two circuits C and C' of the same size that compute the same function, $i\mathcal{O}(C)$ and $i\mathcal{O}(C')$ are computationally indistinguishable to a class of "adversaries" (for example, polynomial-sized circuits). See Section 2.1 for a formal definition.

For many years, $i\mathcal{O}$ had the dual deficiency of neither having any candidate constructions, nor having particularly interesting applications. However, in the two decades since its definition, both of these statements have seen dramatic reversals. While $i\mathcal{O}$'s security definition initially seems weak (as it only gives a guarantee about circuits computing the same function), it turns out to be extremely powerful. We now know that nearly every cryptographic primitive (e.g., one-way functions [KMN+22], public-key encryption [SW21], multi-party non-interactive key exchange [BZ17], etc.) can be constructed assuming $i\mathcal{O}$ exists and NP is not in BPP infinitely often (see Section 1.3 of Jain-Lin-Sahai [JLS21] for a more comprehensive list).

Similarly, while many initial candidate constructions required new assumptions that were later broken, groundbreaking work of Jain, Lin, and Sahai [JLS21] showed that $i\mathcal{O}$ exists assuming four "well-founded" assumptions (this has been improved to three assumptions, in [JLS22b]). Moreover, the $i\mathcal{O}$ they construct has strong security properties: no polynomial-sized circuit adversary can distinguish the $i\mathcal{O}$ of equivalent circuits except with subexponentially small probability. We refer to $i\mathcal{O}$ with these security properties as *JLS-security* (a formal definition is in Section 2.1). Following the work of Jain, Lin, and Sahai, $i\mathcal{O}$ has now become a widely-believed assumption in cryptography (see for example the recent Quanta article of Klarreich [Kla20]).

1.2 Our Results

In this paper, we give the first concrete evidence that RANGE AVOIDANCE is hard to solve deterministically when the number of outputs $m(n) = \mathsf{poly}(n)$. In particular, our conditional lower bound for RANGE AVOIDANCE follows from indistinguishability obfuscation and various forms of NP $\neq$ coNP. Our argument is quite general in that it holds for a variety of parameters with trade-offs on the assumptions, but we state a simple version first.

Theorem 1. *Assume that* NP $\neq$ coNP *and $i\mathcal{O}$ with JLS-security exists. Then for all $c \geq 1$, there is a $k \geq c$ such that there is no deterministic polynomial-time algorithm for* AVOID *on n^k-size circuits with n inputs and $m(n) = n^c$ outputs.*

That is, assuming NP $\neq$ coNP and JLS-secure $i\mathcal{O}$ exists, there are *no* efficient deterministic algorithms for AVOID, even if the number m of output bits is allowed to be an arbitrarily large polynomial in n. Note that when m is an arbitrarily large polynomial in n, all but an exponentially small fraction (2^{-m+n}) of length-m strings are outside of the range of C. Interestingly, the hard instances in our proof are circuits C with at most *two elements* in their range![3] In fact, one of those two elements is always the string 0^m.

Before we discuss extensions of Theorem 1, let us briefly motivate how the assumptions in Theorem 1 arise. Suppose a deterministic algorithm for AVOID exists, and one is aiming to show a contradiction. What can you do with a deterministic algorithm, that you could not do with a randomized algorithm (which we know exists)? With a deterministic efficient algorithm for AVOID, one can guarantee that for every circuit C, there is a short "proof" that a *specific* string y_C is outside the range of C. In particular, the computation history of the deterministic AVOID algorithm running on C and outputting y_C, constitutes such a "proof." In contrast, it is unclear how to get such a guarantee from the simple randomized algorithm for AVOID that picks a string uniformly at random. A priori, it seems powerful that every circuit C has a short proof that a specific y_C is outside its range. If the description C was complex enough to function like a "black box," then the shortest proof that y_C is not in the range may simply evaluate C on all inputs and observe that C never outputs y_C. Thus, at a very high level, a deterministic AVOID algorithm may provide short proofs for statements that might not have short proofs (motivating an assumption like NP $\neq$ coNP) for circuits that behave like black boxes (motivating an assumption like $i\mathcal{O}$). Of course, this is a very rough intuition; for more details we point the reader to the (relatively short) proof of Theorem 1 in Section 3. We remark that our work is certainly not the first time that $i\mathcal{O}$ is being applied in complexity theory (see Section 2.1 for references).

Extensions. We now discuss extensions of Theorem 1, which illustrate various tradeoffs between the time complexity of AVOID, simulations of coNP with nondeterminism, and the allowed stretch m. A more general version of our result rules out subexponential-time ($2^{n^{o(1)}}$-time) deterministic algorithms for AVOID, as well as algorithms for AVOID where the number of outputs m can be subexponential in n. To rule out subexponential-time AVOID algorithms, we apparently require a stronger (but still standard) notion of security for $i\mathcal{O}$ than that of JLS-security. In particular, we need that no subexponential-size circuit adversary can distinguish the $i\mathcal{O}$ of equivalent circuits, except with subexponentially small probability. We refer to $i\mathcal{O}$ with this security as *subexponentially-secure $i\mathcal{O}$*. We point the reader to Section 2.1 for formal definitions, but we stress that subexponentially-secure $i\mathcal{O}$ is very plausible

[3] Having a range of only two elements is best-possible, in a sense. For circuits C with only one string in their range, there is a simple AVOID algorithm: output any m-bit string different from $C(0^n)$.

and holds if, for example, corresponding security guarantees hold for the three well-founded assumptions used by Jain-Lin-Sahai [JLS22a].

In the statement below (and throughout this paper) we always assume the number of outputs $m(n)$ is a time-constructible function.

Theorem 2. *Assume subexponentially-secure $i\mathcal{O}$ exists. For every $m(n) > n$ there is an $s(n) = \mathsf{poly}(m)$ such that, if there is a deterministic t-time algorithm for* Avoid *circuits with $m(n)$ outputs and size $s(n)$, then*

$$\mathsf{coNP} \subseteq \bigcup_{k\in\mathbb{N}} \mathsf{NTIME}[t(m^k(n))]$$

Here, $\mathsf{NTIME}[t]$ refers to the set of languages computable by a nondeterministic Turing machine in time $O(t)$. Setting $t = 2^{n^{o(1)}}$, we conclude that Range Avoidance cannot be solved in deterministic $2^{s^{o(1)}}$ time on circuits of size s, assuming subexponentially-secure $i\mathcal{O}$ and a rather weak exponential-time hypothesis. Let $\mathsf{NSUBEXP} := \bigcap_{k\in\mathbb{N}} \mathsf{NTIME}[2^{n^{1/k}}]$.

Corollary 3. *Assuming* $\mathsf{coNP} \not\subseteq \mathsf{NSUBEXP}$ *and subexponentially-secure $i\mathcal{O}$,* Range Avoidance *cannot be solved in deterministic $2^{s^{o(1)}}$ time on circuits of size s.*

The assumption $\mathsf{coNP} \not\subseteq \mathsf{NSUBEXP}$ is significantly weaker than what is often assumed in fine-grained complexity. For example, the Nondeterministic Exponential Time Hypothesis (NETH) [CGI+16, CRTY20] states that there is an $\varepsilon > 0$ such that unsatisfiable 3SAT instances with n variables cannot be refuted in nondeterministic $2^{\varepsilon n}$ time. Note NETH is a much stronger hypothesis than $\mathsf{coNP} \not\subseteq \mathsf{NSUBEXP}$.

Setting $m(n)$ to be any constructible function that is $2^{n^{o(1)}}$ in Theorem 2, we conclude that even when the number of outputs m is close to exponential in n, Avoid still does not have a polynomial-time deterministic algorithm, assuming $\mathsf{coNP} \not\subseteq \mathsf{NSUBEXP}$ and subexponentially secure $i\mathcal{O}$.

Corollary 4. *Assuming* $\mathsf{coNP} \not\subseteq \mathsf{NSUBEXP}$ *and subexponentially-secure $i\mathcal{O}$,* Range Avoidance *cannot be solved in deterministic polynomial-time for every constructible $m(n) = 2^{n^{o(1)}}$.*

We also prove that $i\mathcal{O}$ with security weaker than JLS-security or subexponential security can still be used to show the non-existence of deterministic algorithms for Avoid. We show Avoid $\notin \mathsf{FP}$ assuming only polynomially-secure $i\mathcal{O}$ and a still weaker statement than the typical NETH.

Hypothesis 5 (NETH for Circuits)**.** *There is an $\varepsilon > 0$ such that Circuit Unsatisfiability on n-input circuits of size $2^{o(n)}$ cannot be solved nondeterministically in $2^{\varepsilon n}$ time.*

Theorem 6. *Assuming NETH for Circuits and polynomially-secure $i\mathcal{O}$,* Avoid *is not in* FP*. Moreover, under the assumptions, it follows that for all $b, c \geq 1$ there is an $\varepsilon > 0$ such that* Avoid *cannot be solved in $O(2^{c\varepsilon n})$ time on circuits of size $2^{\varepsilon n}$ with n inputs and bn outputs.*

Note that, in the case of circuits with $2^{o(n)}$ size and bn outputs, exhaustive search solves Range Avoidance in time about $2^{bn+o(n)}$. Under our assumptions, we rule out $2^{o(n)}$ time for $2^{o(n)}$-size circuits, when the number of outputs m is linear in n. Therefore, even subexponential-time improvements over exhaustive search for Range Avoidance should already be considered unlikely for these parameters, under our assumptions.

Witness Encryption Suffices. A careful inspection of our proofs reveals that our use of indistinguishability obfuscation can be replaced with a seemingly weaker cryptographic primitive called *witness encryption*. At a high level, witness encryption [GGSW13] allows one to, given a SAT formula φ, encrypt a message m such that only recipients who know a satisfying assignment to φ can decrypt m (the actual definition is more involved, but this is the basic idea). It is known that $i\mathcal{O}$ implies witness encryption as a special case [GGH+16]. We state our results in terms of $i\mathcal{O}$ instead of witness encryption for two reasons. First, currently the only known way to construct witness encryption with our desired parameters under well-founded assumptions is via $i\mathcal{O}$. Second, this paper is aimed primarily at a complexity-theoretic audience, who is likely more familiar with the notion of $i\mathcal{O}$ than witness encryption.

Nevertheless, witness encryption is believed to be a weaker assumption compared to $i\mathcal{O}$, and admits several plausible constructions. Chen, Vaikuntanathan, and Wee [CVW18] proposed a simple construction from LWE-like problems, whose security was proved later by Vaikuntanathan, Wee, and Wichs [VWW22] based on LWE-like assumptions. An alternative LWE-based construction was proposed by Tsabary [Tsa22] under similar assumptions. Barta, Ishai, Ostrovsky, and Wu [BIOW20] also gave a construction in "generic group model" based on an (unproven) hardness of approximation hypothesis of certain coding problems.

Application: Separations in Bounded Arithmetic

Bounded arithmetic refers to fragments of Peano arithmetic that aim to formalize the *(computational) complexity of reasoning*. For instance, Cook's theory PV_1 [Coo75] corresponds to "reasoning in polynomial time", Jeřábek's theory APC_1 [Jeř04, Jeř05, Jeř07a, Jeř09][4] corresponds to randomized polynomial-time computation, and Buss's theories $S^1_2, S^2_2, \ldots$ correspond to the polynomial-time hierarchy [Bus85]. Indeed, theories corresponding to other complexity classes such as TC^0, NC^1, and PSPACE have been studied (see, e.g., [Kra95, CN10]). From a proof complexity point of view, bounded theories can also be regarded as uniform versions of propositional proof systems (see, e.g., [Kra19]).

One motivation to study bounded arithmetic is that it may explain why longstanding complexity-theoretic conjectures such as $\mathsf{P} \neq \mathsf{NP}$ and $\mathsf{NEXP} \not\subseteq \mathsf{P}_{/\mathsf{poly}}$ are hard to prove. In contrast to barriers such as relativization [BGS75], natural proofs [RR97], and algebrization [AW09] that capture the limitation of specific techniques, it is more desirable to demonstrate the unprovability of these conjectures in *strong mathematical theories* with a solid logical foundation. Bounded arithmetic provides an ideal testing ground for this program. On the one hand, a rather large fragment of known algorithms and complexity theory results can be formalized in bounded theories such as PV_1 and APC_1 [Oja04, LC11, Pic15b, MP20]. On the other hand, connections between bounded arithmetic and complexity theory make it possible to employ complexity-theoretic techniques to obtain unprovability results, leading to exciting developments on the unprovability of complexity upper bounds [KO17, BM20, BKO20, CKKO21] and lower bounds [Kra11, Pic15a, PS21] in PV.

To better understand the power of feasible reasoning, it is important to prove relations (separations or equivalences) among bounded theories. In particular, it has been open for about twenty years whether the *dual Weak Pigeonhole Principle* for PV functions[5] (denoted by $\mathsf{dWPHP}(\mathsf{PV})$) is provable in PV_1. In other words, the question is whether Jeřábek's APC_1, defined as $\mathsf{PV}_1 + \mathsf{dWPHP}(\mathsf{PV})$, is the same as PV_1. Here, $\mathsf{dWPHP}(\mathsf{PV})$ is the "logic version" of AVOID[6] that says *for every* PV *function f and every n and z, there is a $y \in \{0,1\}^{n+1}$ such that for every $x \in \{0,1\}^n$, $f(z,x) \neq y$.* Similar to the complexity of AVOID, there has been no strong evidence for either $\mathsf{APC}_1 = \mathsf{PV}_1$ and $\mathsf{APC}_1 \neq \mathsf{PV}_1$. Known results on this problem include the separation of the *relativized* versions of APC_1 and PV_1 [Jer07b] and a conditional separation based on the assumption that $\mathsf{P} \subset \mathsf{SIZE}[n^k]$ for some k [Kra21]. (However, this assumption contradicts the widely-believed derandomization assumption that E requires circuits of size $2^{\Omega(n)}$.) Krajíček [Kra22] recently proposed an open problem of showing a conditional separation of PV_1 and APC_1 under a "mainstream hypothesis" as the first step to understand the logical power of $\mathsf{dWPHP}(\mathsf{PV})$. Moreover, it might seem reasonable to believe that APC_1 *is* the same as PV_1, since its complexity-theoretic counterpart $\mathsf{BPP} = \mathsf{P}$ follows from plausible circuit lower bounds [IW97, STV01].

In this work, we provide the first plausible evidence that Jeřábek's theory APC_1 is a *strict* extension of PV_1.

Theorem 7 (Corollary 23, Informal)**.** *Assuming $i\mathcal{O}$ with JLS-security and* coNP *is not infinitely often in* AM*, the dual Weak Pigeonhole Principle is not provable in* PV *(in particular,* APC_1 *is a strict extension of* PV_1*).*

Our proof utilizes the standard KPT Witnessing Theorem (see Theorem 18) which extracts an algorithm for AVOID that is allowed to *invert* the input circuit on a *constant* number of m-bit strings, assuming APC_1 is the same as PV_1. Corollary 23 then follows from an extension of the conditional lower bound for AVOID which holds against polynomial-time algorithms with such circuit-inversion oracles.[7]

[4] Note the terminology APC_1 was first used in [BKT14].

[5] By Cobham's characterization of the polynomial-time functions, PV functions (when interpreted in the standard model) are exactly polynomial-time computable functions (see Section 2.3 for details).

[6] Indeed, Korten's investigation of the complexity of explicit constructions was inspired by the early developments of Jeřábek's theory APC_1, as noted in [Kor21, Kor22].

[7] The drawback of the lower bound result comparing with Theorem 2 is that the assumption $\mathsf{NP} \neq \mathsf{coNP}$ is strengthened to $\mathsf{coNP} \not\subseteq$ i.o. AM

Theorem 8 (Theorem 21, Informal)**.** *Assuming the existence of $i\mathcal{O}$ with JLS-security and* NP *is not infinitely often in* AM*, there is no polynomial-time algorithm for* Avoid *with $O(1)$ queries to a circuit-inversion oracle.*

This conditional lower bound stands in interesting contrast to the fact that, under standard derandomization hypotheses (E does not have $2^{\Omega(n)}$-size circuits), there is a deterministic polynomial-time algorithm for Avoid that makes *polynomially-many* circuit inversion queries (follows from [Kor21]; see Appendix C for details). Alternatively, under a stronger assumption, namely E does not have $2^{\Omega(n)}$-size SAT-oracle circuits, we can construct a hitting set of size $\mathsf{poly}(n)$ (see Appendix A) and then find a non-output of the given circuit with $\mathsf{poly}(n)$ circuit-inversion queries.

Under similar assumptions, we also demonstrate a separation of APC_1 and its fragment UAPC_1 that is strong enough to prove interesting results in complexity theory and formalize approximate counting in Jeřábek's framework [Jeř07a], see Section 4.3 for more details.

Application: the Oracle Derandomization Hypothesis for Time-Bounded Kolmogorov Complexity

Our results also have bearing on other hypotheses regarding derandomization. Motivated by applications in parameterized complexity and questions related to "instance compressibility," Fortnow and Santhanam [FS11] introduced the Oracle Derandomization Hypothesis (ODH). Roughly speaking, ODH says that, given a length-n truth table z of an arbitrary Boolean function, one can efficiently deterministically generate a truth table y of length at least $n^{.01}$ such that the function represented by y has circuit complexity $n^{\Omega(1)}$ even when the circuits are given oracle access to the function represented by z. (See Hypothesis 29 for a formal definition, and Section 5 for a comparison to the related problem Complexity.)

This hypothesis is especially intriguing because it is unclear whether it should be true or false. Indeed, Fortnow and Santhanam remark:

> "In our opinion, quite apart from its relevance to compressibility, the Oracle Derandomization Hypothesis is interesting in its own right because it tests our intuitions of which kinds of derandomization are plausible and which are not... We do not have a strong belief about the truth of our derandomization assumption, but we do believe it is hard to refute."

Using essentially the same proof as in Theorem 1, we rule out a related *time-bounded Kolmogorov complexity version* of ODH (formally, Hypothesis 30) under plausible assumptions. Roughly speaking, Hypothesis 30 says that given a string z of length n, one can efficiently deterministically generate a string y of length $n^{.01}$ such that the conditional polynomial-time bounded Kolmogorov complexity of y given z is at least $n^{\Omega(1)}$. Here, the conditional t-time bounded Kolmogorov complexity of y given z refers to the length of the shortest program that outputs x on input y in time $t(|x|)$ (see Section 2.4 for a formal definition) [Kol65, Sip83, Ko86].

Theorem 9 (Informal version of Theorem 31)**.** *The time-bounded Kolmogorov complexity ODH is false assuming* $\mathsf{NP} \neq \mathsf{coNP}$ *and subexponentially secure $i\mathcal{O}$ exists.*

It is a tantalizing open question as to whether one can extend this result to rule out ODH itself. To prove Theorem 9 we crucially make use of the fact that, in this version of ODH, the computational model is able to read the entire string z. In contrast, in the (original) ODH setting, the computational model is only allowed to make a limited number of queries to the string z.

2 Preliminaries

We assume basic familiarity with notions in computational complexity [AB09]. We first review two extensively used tools: Indistinguishability Obfuscation and interactive (Arthur-Merlin) protocols. We also provide a brief introduction on bounded theories PV_1 and APC_1 (see [Kra95, CN10, Kra19] for more detailed expositions), as well as the time-bounded Kolmogorov complexity.

due to technical reasons. Intuitively, we need the Goldwasser-Sipser protocol in AM (see Lemma 13) for approximate counting, and we only know how to "eliminate" oracle queries in the range avoidance algorithm assuming infinitely often lower bounds for coNP.

2.1 Indistinguishability Obfuscation

Definition 10 (Indistinguishability Obfuscation)**.** A polynomial-time randomized algorithm $i\mathcal{O}$ that takes as input a security parameter λ and a circuit C, and randomness r is an indistinguishability obfuscator with security (S, ϵ) if both of the following hold:

- **Perfect Functionality:** For all C and λ, we have that $i\mathcal{O}(1^\lambda, C)$ outputs a circuit computing the same function as C with probability one over its randomness.
- **Indistinguishability:** For all λ, any two circuits C and C' of size at most λ computing the same function, and any $S(\lambda)$-sized adversary circuit A, we have that

$$|\Pr[A(i\mathcal{O}(1^\lambda, C) = 1] - \Pr[A(i\mathcal{O}(1^\lambda, C') = 1]| \leq \epsilon(\lambda)$$

When λ is clear from the context, we write $i\mathcal{O}(C)$ instead of $i\mathcal{O}(1^\lambda, C)$. When we want to specify the randomness r used by the $i\mathcal{O}$ algorithm, we write $i\mathcal{O}(1^\lambda, C; r)$.

We say an $i\mathcal{O}$ is *polynomially-secure* if it is secure for some $S(n) = n^{\omega(1)}$ and $\epsilon(n) < 1/n^{\omega(1)}$. We say it is *subexponentially-secure* if it is secure with $S(n) = 2^{n^\delta}$ and $\epsilon(n) = 2^{-n^\delta}$ for some $\delta > 0$. We say it is *JLS-secure* if it is secure with $S(n) = n^{\omega(1)}$ and and $\epsilon(n) = 2^{-n^\delta}$ for some $\delta > 0$. For our results, it is important that our adversaries are non-uniform circuits.

The breakthrough results of Jain-Lin-Sahai [JLS21, JLS22b] give constructions of both JLS-secure and subexponentially secure $i\mathcal{O}$.

Theorem 11 (Informal version of Jain-Lin-Sahai [JLS22b])**.** *If three "well-founded" cryptographic assumptions hold, then JLS-secure $i\mathcal{O}$ exists.*

Theorem 12 (Informal version of Jain-Lin-Sahai [JLS22a])**.** *If three "well-founded" cryptographic assumptions are secure against subexponential-sized adversaries with subexponential advantage, then subexponentially secure $i\mathcal{O}$ exists.*

Other Complexity Theory Work Using iO. Building on the work of Bitansky, Paneth, and Rosen [BPR15], Garg, Pandey, and Srinivasan [GPS16] show that computing Nash Equilibria (and thus the TFNP class PPAD) is intractable assuming one-way permutations and $i\mathcal{O}$ exist. Impagliazzo, Kabanets, and Volkovich [IKV18] show that if $i\mathcal{O}$ exists then the Minimum Circuit Size Problem [KC00] is in ZPP if and only if NP $\subseteq$ ZPP.

2.2 Arthur-Merlin Protocols

An Arthur-Merlin protocol [Bab85] for a language $L \subseteq \{0, 1\}^*$ is defined as a constant-round interactive protocol between a computationally unbounded prover (Prover) and a polynomial-time probabilistic verifier (Verifier). For a given input $x \in \{0, 1\}^n$ that is accessible by both Prover and Verifier, Prover wants to convince Verifier that $x \in L$ (even if $x \notin L$) with poly(n) bits of communication, whereas the Verifier needs to decide whether $x \in L$ based on the information provided by Prover. Formally, the protocol needs to satisfy the following properties.

- **Completeness:** If $x \in L$, it is possible for Prover to send poly(n)-bits of messages in constant rounds such that Verifier accepts with probability at least $2/3$.
- **Soundness:** If $x \notin L$, given any messages from Prover, Verifier accepts with probability at most $1/3$.

The complexity class AM is defined as the languages that have a sound and complete Arthur-Merlin protocol. As coNP $\subseteq$ AM implies the collapse of PH [BHZ87], it is widely believed that coNP $\nsubseteq$ AM and therefore UNSAT $\notin$ AM. Indeed, we know that NP $=$ AM assuming a standard derandomization hypothesis: namely, there exists a language $L \in$ NE$\cap$coNE requiring nondeterministic circuits of size $2^{\Omega(n)}$ [KvM02, MV05]. This implies that coNP is unlikely to even be infinitely often in AM.

Goldwasser-Sipser Set Lowerbound Protocol. We will need a well-known Arthur-Merlin protocol for approximately counting the size of a set that has efficiently computable membership queries.

Lemma 13 ([GS89], also see [AB09, Section 8.4])**.** *There is an Arthur-Merlin protocol such that the following holds. Suppose that both* Prover *and* $\mathsf{Verifier}$ *receive a circuit* $C : \{0,1\}^n \to \{0,1\}$ *and a number* $s \le 2^n$*. Let* $S = \{x \in \{0,1\}^n \mid C(x) = 1\}$*. Then*

- ***Completeness:*** *If* $|S| \ge s$*, then there exist messages* Prover *can send such that* $\mathsf{Verifier}$ *accepts with probability at least* $2/3$*.*
- ***Soundness:*** *If* $|S| \le s/2$*, then regardless of what* Prover *sends,* $\mathsf{Verifier}$ *accepts with probability at most* $1/3$*.*

Moreover, the protocol is a two-round public-key protocol: $\mathsf{Verifier}$ *first sends a random seed* r *and receives a message* m*; then it deterministically decides whether to accept based on* r *and* m*.*

2.3 Bounded Theories PV_1 and APC_1

Cook [Coo75] defined the theory PV related to polynomial-time complexity as an *equational theory* (i.e. sentences are of the form $t = u$ for terms t and u). Based on a machine-independent characterization of FP due to Cobham [Cob65], it can be shown that the set of function symbols introduced in PV (when interpreted in the standard model) is exactly FP. PV_1 is defined as a first-order theory that is a conservative extension of PV axiomatized by universal sentences. The formal definitions of PV and PV_1 are tedious and we refer interested readers to [Kra19, Chapter 12]. We define T_{PV} to be the universal true theory over the language $\mathcal{L}(\mathsf{PV})$ of PV over the standard model $\mathbb{N}$.[8]

To formalize the probabilistic methods that are widely used in complexity theory and combinatorics, Jeřábek [Jeř04, Jeř05, Jeř07a, Jeř09] introduced an extension of Cook's PV by including the *dual Weak Pigeonhole Principle* for PV function symbols as axioms, which is now known as APC_1 (stands for approximate counting). Let $f(\vec{w}, x)$ be function symbols[9], and let $m(n) > n$ be a function. We define the *dual weak Pigeonhole Principle* for $f(\vec{w}, \cdot)$ with stretch function m,[10] denoted by $\mathsf{dWPHP}_m(f)$, as

$$\mathsf{dWPHP}_m(f) := \forall n \in \mathsf{Log}\ \forall \vec{w}\ \exists y \in \{0,1\}^{m(n)}\ \forall x \in \{0,1\}^n\ f(\vec{w}, x) \ne y, \tag{1}$$

which claims that $f(\vec{w}, \cdot) : \{0,1\}^n \to \{0,1\}^{m(n)}$ cannot be surjective. Here, $\forall n \in \mathsf{Log}$ is short for $\forall N\ \forall n = |N|$, which means that the feasible reasoning is with respect to strings of length n instead of $\log n$. We use $\mathsf{dWPHP}(f)$ (where m is omitted in the subscript) to mean $\mathsf{dWPHP}_m(f)$ for $m(n) = n + 1$. We define

$$\mathsf{dWPHP}(\mathsf{PV}) := \{\mathsf{dWPHP}(f) \mid f \in \mathcal{L}(\mathsf{PV})\},$$

and the theory APC_1 is defined as $\mathsf{PV}_1 + \mathsf{dWPHP}(\mathsf{PV})$. Since any polynomial-time function can be computed with a multi-output polynomial-size circuit, APC_1 can also be defined equivalently as $\mathsf{PV}_1 + \mathsf{dWPHP}(\mathsf{Eval})$, where $\mathsf{Eval}(C, x)$ evaluates the circuit C on the input x.

Remark 14. The stretch function m for $\mathsf{dWPHP}(\mathsf{PV})$ used to define APC_1 can be a subtle issue, as we cannot prove an equivalence between $\mathsf{dWPHP}(\mathsf{PV})$ with different stretch functions within PV_1 (such equivalence can be proved within Buss's theory S^1_2 for polynomial computation, see, e.g., [Jer07b, Theorem 3.1]). Jeřábek [Jer07b] also proved that $\mathsf{PV}_1(\alpha)$ (a relativised version of PV_1) cannot prove the equivalence of $\mathsf{dWPHP}(\alpha)$ between different parameters. This will not be a problem for us, since our unprovability result works for the weakest version of dWPHP (i.e. with stretch function $m(n)$ allowed to be an arbitrarily large polynomial). △

We remark that, besides the application of bounded arithmetic to understanding complexity barriers, there are also recent interesting applications in propositional proof complexity [Kha22] and cryptography [JJ22].

[8] That is, T_{PV} contains all sentences of the form $\forall \vec{x}\ \beta$ for some quantifier-free formula β that are true in the standard model $\mathbb{N}$.

[9] Note that the inclusion of $\vec{w}$ is crucial; if we remove $\vec{w}$ in the definition of APC_1, we will obtain a (possibly) weaker fragment of APC_1 (see, e.g., [PS21, Section 2]).

[10] In the rest of the paper, we assume that the stretch function $m(n)$ is a PV-function. This is without loss of generality, as the set of PV functions (when interpreted in the standard model) is exactly FP, the class of polynomial-time computable functions.

2.4 Time-Bounded Kolmogorov Complexity

There are multiple notions of time-bounded Kolmogorov complexity in the literature; in this paper, we consider the following. Let $\mathcal{U}$ be a fixed universal Turing machine such that $\mathcal{U}(M, x, 1^t)$ runs the Turing machine encoded by M for t steps on the input x and outputs the string on the tape.

Let $t : \mathbb{N} \to \mathbb{N}$ be a function.

Definition 15 (Time-Bounded Kolmogorov Complexity [Kol65, Sip83, Ko86])**.** The K^t-complexity of a string $x \in \{0,1\}^n$, denoted by $\mathsf{K}^t(x)$, is the minimum ℓ such that for some $M \in \{0,1\}^\ell$, $\mathcal{U}(M, \varepsilon, 1^{t(n)}) = x$.

Definition 16 (Conditional K^t-Complexity)**.** The K^t-complexity of a string $x \in \{0,1\}^n$ conditioned on a string y, denoted by $\mathsf{K}^t(x|y)$, is the minimum ℓ such that for some $M \in \{0,1\}^\ell$, $\mathcal{U}(M, y, 1^{t(n)}) = x$.

Although the definitions of K^t-complexity and conditional K^t-complexity depend on the universal Turing machine $\mathcal{U}$, the results in this paper (as well as most results on these notions) are not sensitive to the choice of $\mathcal{U}$. Therefore we will omit $\mathcal{U}$ and simply use "the encoding of M" to denote the encoding of M with respect to $\mathcal{U}$.

3 No Efficient Deterministic Algorithms for Range Avoidance

In this section, we prove Theorem 1 and Theorem 2. We restate both of these theorems below.

Reminder of Theorem 1. *Assume that* $\mathsf{NP} \neq \mathsf{coNP}$ *and* $i\mathcal{O}$ *with JLS-security exists. Then for all* $c \geq 1$*, there is a* $k \geq c$ *such that there is no deterministic polynomial-time algorithm for* Avoid *on* n^k*-size circuits with* n *inputs and* n^c *outputs.*

Reminder of Theorem 2. *Assume subexponentially-secure* $i\mathcal{O}$ *exists. For every* $m(n) > n$ *there exists an* $s(n) = \mathsf{poly}(m)$ *such that if there is a deterministic* t*-time algorithm for* Avoid *circuits with* $m(n)$ *outputs and size* $s(n)$*, then*

$$\mathsf{coNP} \subseteq \bigcup_{k \in \mathbb{N}} \mathsf{NTIME}[t(m^k(n))]$$

We prove Theorem 2 and remark in the proof how to modify it prove Theorem 1.

Proof of Theorem 2. We will show that, under the assumptions, there is a $t(\mathsf{poly}(s))$-time nondeterministic algorithm $\mathcal{A}$ for the coNP-complete problem of checking whether a propositional formula with n variables and $O(n)$ size[11] is unsatisfiable. Our algorithm $\mathcal{A}$ takes as input an $O(n)$-size formula φ with n variables, and $\mathcal{A}$ accepts φ if and only if φ is unsatisfiable. $\mathcal{A}$ works as follows:

1. Nondeterministically guess a $y \in \{0,1\}^{m(n)}$ and an $r \in \{0,1\}^{\mathsf{poly}(n+\lambda)}$ where $\lambda = \mathsf{poly}(m(n))$. (The degree of the polynomial for λ will be chosen later.)
2. Let $C[\varphi, y]$ denote a circuit[12] that takes n input bits, outputs $m(n) > n$ bits, and satisfies
$$C[\varphi, y](x) = \begin{cases} 0^{m(n)}, & \text{if } \varphi(x) = 0 \\ y, & \text{if } \varphi(x) = 1 \ . \end{cases}$$
3. Accept if and only if $y = \textsc{Avoid}(i\mathcal{O}(C[\varphi, y]; r))$. (Here, we abuse notation and let Avoid and $i\mathcal{O}$ denote their corresponding algorithms.)

[11] To see this problem is coNP-complete, note that an arbitrary formula can be made linear-sized with polynomial blowup while preserving unsatisfiability: simply add extra variables that do nothing.

[12] It does not matter precisely how we implement $C[\varphi, y]$; the argument will work as long as our $C[\varphi, y]$ satisfies the specification and has size $\mathsf{poly}(m(n))$.

This completes the description of $\mathcal{A}$.

We now argue the correctness of the reduction $\mathcal{A}$. Observe that, by construction, $i\mathcal{O}(C[\varphi, y]; r)$ is a circuit with n-inputs, $m(n)$-outputs, and size $\mathsf{poly}(m(n))$. (To see this size bound, note that φ has size $O(n)$, so $C[\varphi, y]$ has size $O(m(n)^c)$ for some constant $c \geq 1$, and $i\mathcal{O}$ blows up this size by at most a fixed polynomial in $\lambda = \mathsf{poly}(m(n))$.) Thus, by setting s to be a sufficiently large polynomial in m, the input circuit $i\mathcal{O}(C[\varphi, y]; r)$ to the Avoid algorithm in item 3 is indeed an instance where the algorithm is assumed to work. Furthermore, it is easy to see that $\mathcal{A}$ runs in nondeterministic time $t(\mathsf{poly}(m(n))) + \mathsf{poly}(m(n)) = t(\mathsf{poly}(m(n)))$. Hence, to prove the theorem we just need to show soundness (if $\mathcal{A}$ accepts, then φ is unsatisfiable) and completeness (if φ is unsatisfiable, then $\mathcal{A}$ accepts).

First we show soundness. Suppose $\mathcal{A}$ accepts φ. Then there exists y and r such that $y = \textsc{Avoid}(i\mathcal{O}(C[\varphi, y]; r))$. By the correctness of Avoid and the perfect functionality of $i\mathcal{O}$, we know that y is not in the range of $C[\varphi, y]$. By construction of $C[\varphi, y]$, this means that φ is not satisfiable.

Now we show completeness. Suppose the formula φ is unsatisfiable. For simplicity of notation, we let $m = m(n)$ in the following. We begin by considering the output distribution of the Avoid algorithm on $i\mathcal{O}(C[\varphi, 0^m]; r)$ for uniformly random r (notice we have set y here to be 0^m). Since Avoid always outputs a string in $\{0,1\}^m$, there exists a "frequent" string $y^\star$ such that

$$\Pr_r[y^\star = \textsc{Avoid}(i\mathcal{O}(C[\varphi, 0^m]; r))] \geq 2^{-m}.$$

We now consider what happens when we set $y = y^\star$. The crucial point is this: because φ is unsatisfiable, observe that $C[\varphi, 0^m]$ and $C[\varphi, y^\star]$ compute the same function! Therefore, by the subexponential security of $i\mathcal{O}$, there is an $\epsilon > 0$ such that for every adversary B of size 2^{λ^ϵ} taking input of length $\mathsf{poly}(m(n))$,

$$\left|\Pr_r[B(i\mathcal{O}(C[\varphi, 0^m]) = 1] - \Pr_r[B(i\mathcal{O}(C[\varphi, y^\star]) = 1]\right| < 2^{-\lambda^\epsilon}.$$

In particular, we can consider the *non-uniform*[13] circuit adversary $B(X)$ that outputs 1 if and only if $\textsc{Avoid}(X) = y^\star$. Without loss of generality, we can assume the size of B is at most $2^{\mathsf{poly}(m(n))}$, as the exhaustive search algorithm for Avoid would provide such a size bound. Thus the size of B is at most

$$2^{\mathsf{poly}(m(n))} \leq 2^{\lambda^\epsilon}$$

when λ is a sufficiently large polynomial in m. (To modify this proof to prove Theorem 1 instead, the only difference is to observe that, if $t(q) = \mathsf{poly}(q)$, then it suffices to have $i\mathcal{O}$ with JLS-security in this step, since the circuit checking if the output of Avoid equals $y^\star$ has polynomial size.) Consequently, applying $i\mathcal{O}$ security to B, we derive

$$\begin{aligned}\Pr_r[y^\star = \textsc{Avoid}(i\mathcal{O}(C[\varphi, y^\star]; r))] &\geq \Pr_r[y^\star = \textsc{Avoid}(i\mathcal{O}(C[\varphi, 0^m]; r))] - 2^{-\lambda^\epsilon} \\ &\geq 2^{-m} - 2^{-\lambda^\epsilon} > 0,\end{aligned}$$

by setting λ to be a sufficiently large polynomial in m. Hence, we can conclude there is an r such that

$$y^\star = \textsc{Avoid}(i\mathcal{O}(C[\varphi, y^\star]; r)),$$

so $\mathcal{A}$ will accept φ. □

Remark 17. For readers familiar with the cryptographic notion of *witness encryption* [GGSW13], we note that in Theorem 2 (and in fact all theorems in this paper that assume $i\mathcal{O}$), we can relax the assumption that $i\mathcal{O}$ exists to the potentially weaker assumption that witness encryption with similar security and a deterministic decryption algorithm exists. Informally, witness encryption allows one to encrypt a string y with a SAT formula φ such that

- **(Correctness).** If φ is satisfiable, then one can efficiently decrypt y given a satisfying assignment.
- **(Security).** If φ is unsatisfiable, then the encryption of y and $0^{|y|}$ are computationally indistinguishable.

[13] The non-uniformity comes from $y^\star$.

Garg et al. [GGH+16] observed that $i\mathcal{O}$ implies witness encryption (with a deterministic decryption algorithm) as a special case. Indeed, in Garg et al.'s construction, one can witness-encrypt a message y with a formula φ by outputting $i\mathcal{O}(C[\varphi, y])$. We are implicitly using this construction in our proofs.

To modify the proof of Theorem 2 to use witness encryption instead, one modifies the algorithm $\mathcal{A}$ to the nondeterministic algorithm below:

1. Nondeterministically guess a $y \in \{0,1\}^{m(n)}$ and an $r \in \{0,1\}^{\mathsf{poly}(n+\lambda)}$
2. Let e be the witness encryption of the string y according to φ using randomness r and security parameter λ.
3. Let C be the circuit that takes as input a string $x \in \{0,1\}^n$ and attempts to output the decryption of e using the purported witness x (if decryption fails, output 0^m).
4. Accept if and only if $y = \textsc{Avoid}(C)$.

The analysis of the new algorithm is essentially the same as the analysis of the original $\mathcal{A}$, where the *perfect functionality* and *indistinguishability* properties of $i\mathcal{O}$ are now replaced by the *correctness* and *security* properties of the witness encryption scheme respectively. Our other proofs using $i\mathcal{O}$ can be similarly modified. $\triangle$

The proof of Theorem 2 can be generalized in several other ways. For one, the same proof also works to rule out zero-error randomized algorithms (although zero-error randomized algorithms also imply deterministic algorithms under a derandomization assumption).

The proof can also be generalized to work with just polynomially-secure $i\mathcal{O}$. To do this, instead of $\mathsf{NP} \neq \mathsf{coNP}$, we consider a nondeterministic version of the exponential time hypothesis:

Reminder of Hypothesis 5. *(NETH for Circuits) There is an $\varepsilon > 0$ such that Circuit Unsatisfiability problem on n-input circuits of size $2^{o(n)}$ cannot be solved nondeterministically in $2^{\varepsilon n}$ time.*

Hypothesis 5 is in fact a much weaker statement than the usual NETH, which posits that nondeterministically refuting unsatisfiable 3-CNFs requires $2^{\varepsilon n}$-size proofs verifiable in $2^{\varepsilon n}$ time. We only require an exponential lower bound in the case of refuting subexponential-size circuits.

Reminder of Theorem 6. *Assuming NETH for Circuits and polynomially-secure $i\mathcal{O}$,* $\textsc{Avoid}$ *is not in* FP. *Moreover, under the assumptions, it follows that for all $b, c \geq 1$ there is an $\varepsilon > 0$ such that* $\textsc{Avoid}$ *cannot be solved in $O(2^{c\varepsilon n})$ time on circuits of size $2^{\varepsilon n}$ with n inputs and bn outputs.*

Proof. The algorithm for UNSAT in our proof is essentially the same as that of Theorem 2, except we analyze the case of a very large security parameter λ.

Assume there is a universal constant $d \geq 1$ and an algorithm B for polynomially-secure $i\mathcal{O}$ which runs in time $O((s \cdot \lambda)^d)$ on circuits of size s with security parameter λ. Furthermore, assume that there are universal constants $b, c \geq 1$ such that for all sufficiently small $\alpha \in (0,1)$, $\textsc{Range Avoidance}$ can be solved in $O(2^{c\alpha n})$ time on circuits of size $2^{\alpha n}$ with n inputs and bn outputs. Given the assumptions, we show how to construct a nondeterministic algorithm for proving the unsatisfiability of arbitrary subexponential-size circuits in subexponential time.

Fix $b, c \geq 1$. Let $m = bn$, $\alpha \in (0,1)$ be a constant to be chosen later, and A be an algorithm for $\textsc{Range Avoidance}$ as described above. Let $\varepsilon > 0$ be an arbitrarily small constant. Given a circuit φ with n inputs and size $2^{o(n)}$, we run precisely the same reduction as Theorem 2, except with an exponentially large value of λ.

1. Nondeterministically guess $y \in \{0,1\}^m$ and $r \in \{0,1\}^\ell$, where $\ell = (\mathsf{poly}(|\varphi|)\lambda)^d$ and $\lambda = 2^{\epsilon \cdot n}$. (The parameter ℓ is upper-bounded by the running time of B.)
2. Let $C[\varphi, y]$ be a circuit taking n input bits and outputting m bits with the specification:

$$C[\varphi, y](x) = \begin{cases} 0^m, & \text{if } \varphi(x) = 0 \\ y, & \text{if } \varphi(x) = 1. \end{cases}$$

3. Accept if and only if $y = A(B(C[\varphi, y]; r))$.

It takes $\mathsf{poly}(|\varphi|)$ time to construct $C[\varphi, y]$. Given our assumption on algorithm B, the output of $B(C[\varphi, y])$ is a circuit of size $(\mathsf{poly}(|\varphi|)\lambda)^d = 2^{\varepsilon dn+o(n)}$. Thus our algorithm A for RANGE AVOIDANCE applies to the circuit $B(C[\varphi, y])$, by setting $\alpha = 2\varepsilon d$ so that $(\mathsf{poly}(|\varphi|)\lambda)^d = 2^{\varepsilon dn+o(n)} \leq 2^{\alpha n}$. Therefore the above algorithm runs in nondeterministic time $2^{c\alpha n} = 2^{2\varepsilon cdn}$. As the input circuit φ has size $2^{o(n)}$, c and d are fixed, and $\varepsilon > 0$ can be made arbitrarily small, this would refute NETH for Circuits.

As in the proof of Theorem 2, it remains to show that φ is unsatisfiable if and only if there is a y and r such that y is the output of A on $B(C[\varphi, y]; r)$. First, if such y and r exist, then analogously to the proof of Theorem 2, we conclude that φ is unsatisfiable by the construction of $C[\varphi, y]$.

For the other direction, suppose that φ is unsatisfiable. As in the proof of Theorem 2, we know that there is some $y^\star \in \{0,1\}^m$ such that

$$\Pr_r[y^\star = A(B(C[\varphi, 0^m]; r))] \geq 2^{-m}.$$

Given $y^\star$, we may define an "adversary" circuit D which has $y^\star \in \{0,1\}^m$ hard-coded, takes in an input circuit C' (ostensibly simulating $B(C[\varphi, y]; r)$ on some y and r), and outputs 1 if and only if $y^\star = A(C')$. Using a standard translation of t-time algorithms into $O(t^3)$-size circuits, the input circuit C' only has to have size at most $2^{3\varepsilon dn+o(n)}$. Translating the composition of A, B, and $C[\varphi, y]$ into circuits, the size of the adversary circuit D is at most $2^{6\varepsilon cdn}$, which is only polynomially larger than its input circuit C'.

Assuming polynomially-secure $i\mathcal{O}$, since $C[\varphi, 0^m]$ and $C[\varphi, y^\star]$ compute the same function, we have for *every* constant $K \geq 1$ that

$$\left|\Pr_r[D(B(C[\varphi, 0^m]; r)) = 1] - \Pr_r[D(B(C[\varphi, 0^m]; r)) = 1]\right| < \frac{1}{\lambda^K},$$

which implies

$$\Pr_r[y^\star = A(B(C[\varphi, y^\star]; r))] \geq \Pr_r[y^\star = A(B(C[\varphi, y^\star]; r))] - \frac{1}{\lambda^K} \geq \frac{1}{2^m} - \frac{1}{\lambda^K}. \tag{2}$$

The above probability is greater than 0, provided that $2^m < \lambda^K$, i.e.,

$$\lambda > 2^{m/K}. \tag{3}$$

Recall that we set $\lambda = 2^{\varepsilon n}$ for an arbitrarily small fixed $\varepsilon > 0$, and $m = bn$ for a fixed $b \geq 1$, while the constant $K \geq 1$ can be as arbitrarily large as needed (independently of all other constants). Therefore (3) holds.

By (2), there is an r such that $y^\star = A(B(C[\varphi, y^\star]; r))$, as desired. □

4 Application in Bounded Arithmetic: Separating APC_1 and PV_1

In this section, we prove a conditional separation of the bounded theories APC_1 and PV_1, assuming that $i\mathcal{O}$ with JLS-security exists and coNP is not infinitely often in AM.

The only result from logic that we need is the standard KPT Witnessing Theorem for $\forall\exists\forall$ formulas, which connects the provability of any $(\forall\exists\forall)$-sentence with a Student-Teacher game for interactively computing a witness to the existential quantifier.

Theorem 18 (KPT Witnessing Theorem for T_{PV} [KPT91]). *For every quantifier-free formula $\varphi(\vec{x}, y, z)$ in the language $\mathcal{L}(\mathsf{PV})$, if $\mathsf{T}_{\mathsf{PV}} \vdash \forall\vec{x}\,\exists y\,\forall z\,\varphi(\vec{x}, y, z)$, then there is a $k \in \mathbb{N}$ and $\mathcal{L}(\mathsf{PV})$-terms $t_1, t_2, \ldots, t_k$ such that*

$$\mathsf{T}_{\mathsf{PV}} \vdash \forall\vec{x}\,\forall z_1\,\forall z_2\,\ldots\,\forall z_k \bigvee_{i=1}^{k} \varphi(\vec{x}, t_i(\vec{x}, z_1, \ldots, z_{i-1}), z_i). \tag{4}$$

It is well-known that the terms $t_1, t_2, \ldots, t_k$ extracted from the proof in the KPT Witnessing Theorem can be interpreted as an k-round interactive computation of a witness y such that $\forall z\ \varphi(\vec{x}, y, z)$ given the input $\vec{x}$. Consider the following game between a *Student* who wants to find a correct witness y and a *Teacher* who will provide help.

In the first round, the Student proposes $y_1 := t_1(\vec{x})$ as a candidate, and, in the case, y_1 is not a correct witness of $\exists y\, \forall z\, \varphi(\vec{x}, y, z)$, the Teacher provides a *counterexample* z_1 such that $\varphi(\vec{x}, y_1, z_1)$ is false. The Student then proposes a new candidate $y_2 := t_2(\vec{x}, z_1)$ based on the counterexample given in the first round, and the Teacher, again, provides a counterexample z_2 if y_2 is not a correct witness, etc. Equation (4) means that after k rounds of interaction between the Student and the Teacher, at least one of the $y_1, y_2, \ldots, y_k$ proposed by the Student has to be a correct witness of $\exists y\, \forall z\, \varphi(\vec{x}, y, z)$. We refer the readers to [Kra19] for more discussion about the KPT Witnessing Theorem and the Student-Teacher game.

4.1 Provability of dWPHP and the Tractability of Avoid

The KPT Witnessing Theorem (Theorem 18) provides a connection between the provability of dWPHP and the tractability of Avoid, in the sense of the Student-Teacher game. Let $\mathsf{Eval}(C, x) := C(x)$ be the circuit evaluation function. Recall that $\mathsf{dWPHP}_\ell(\mathsf{Eval})$ refers to the following sentences:

$$\mathsf{dWPHP}_\ell(\mathsf{Eval}) := \forall n \in \mathsf{Log}\ \forall \text{circuits } C : \{0,1\}^n \to \{0,1\}^\ell\ \exists y \in \{0,1\}^\ell\ \forall x \in \{0,1\}^n\ [\mathsf{Eval}(C, x) \neq y].$$

Suppose that $\mathsf{dWPHP}_\ell(\mathsf{Eval})$ is provable in $\mathsf{T_{PV}}$. Then there is a constant-round Student-Teacher game that finds a $y \in \{0,1\}^\ell$ witnessing the existential quantifier, where the Teacher provides counterexamples for the universal quantifier over $x \in \{0,1\}^n$. Taking a closer look at this game, we can see that this corresponds to an algorithm for Avoid with circuit-inversion oracle queries, which is formally defined as follows.

Definition 19 (Solving Avoid with a Circuit-Inversion Oracle)**.** Let $m = m(n)$ and $k = k(n)$. A *polynomial-time algorithm with k circuit-inversion oracle queries for* Avoid *with m outputs* is a polynomial-time oracle algorithm A such that given a circuit $C : \{0,1\}^n \to \{0,1\}^m$ and access to an oracle $\mathcal{O}(\cdot) : \{0,1\}^m \to \{0,1\}^n$ with at most k queries, $A^{\mathcal{O}(\cdot)}(C)$ outputs a $y \in \{0,1\}^m$ such that $C(\mathcal{O}(y)) \neq y$. Furthermore, $\mathcal{O}(y)$ always returns an x such that $C(x) = y$, when such an x exists.

Theorem 20. *For every constructive function $m(n) \leq \mathsf{poly}(n)$ such that $m(n) > n$, if $\mathsf{T_{PV}} \vdash \mathsf{dWPHP}_\ell(\mathsf{Eval})$, then* Avoid *with m outputs has a polynomial-time algorithm with $O(1)$ circuit-inversion oracle queries.*

Proof. Let $m(n) = \mathsf{poly}(n)$ be some constructive function. By the assumptions, we know that

$$\mathsf{T_{PV}} \vdash \forall n \in \mathsf{Log}\ \forall C\ \exists y \in \{0,1\}^{m(n)}\ \forall x \in \{0,1\}^n\ [\mathsf{Eval}(C, x) \neq y].$$

By the KPT Witnessing Theorem (Theorem 18), there is a $k = O(1)$ and $\mathcal{L}(\mathsf{PV})$-terms $t_1, t_2, \ldots, t_k$ such that

$$\mathsf{T_{PV}} \vdash \forall n\, \forall C\, \forall z_1\ \ldots\ \forall z_n\ \Big([\mathsf{Eval}(C, z_1) \neq t_1(n, C)] \vee [\mathsf{Eval}(C, z_2) \neq t_2(n, C, z_1)] \vee \ldots \\ \vee [\mathsf{Eval}(C, z_k) \neq t_k(n, C, z_1, \ldots, z_{k-1})]\Big).$$

Now we show that given any circuit $C : \{0,1\}^n \to \{0,1\}^{m(n)}$ and access to an oracle $\mathcal{O} : \{0,1\}^{m(n)} \to \{0,1\}^n$, there is a polynomial-time algorithm that makes k queries to $\mathcal{O}(\cdot)$ and finds a $y \in \{0,1\}^{m(n)}$ such that $C(\mathcal{O}(y)) \neq y$. Let $f_1, f_2, \ldots, f_k \in \mathsf{FP}$ be the functions that are the interpretations of $t_1, t_2, \ldots, t_k$, respectively, in the standard model. We play the aforementioned Student-Teacher game, where in the i-th round, the Student proposes $y_i := f_i(n, C, z_1, \ldots, z_{i-1})$, and the Teacher responds with $z_i := \mathcal{O}(y_i)$. By Equation (4) and the soundness of $\mathsf{T_{PV}}$ in the standard model, we know that for some $i \in [k]$, the Teacher fails to provide a correct counterexample z_i in the i-th round, i.e., $\mathsf{Eval}(C, z_i) \neq y_i$. The algorithm can then output y_i. □

4.2 Impossibility of Solving Avoid with a Circuit-Inversion Oracle

Now we show that Avoid has no polynomial-time algorithm with $O(1)$ circuit-inversion oracle queries under plausible assumptions by generalizing the proof of Theorem 2.

Theorem 21. *Let $m = m(n) = \mathsf{poly}(n)$ and $k = O(1)$ such that $m \geq n + 1$. Assume that coNP is not infinitely often in AM and $i\mathcal{O}$ with JLS-security exists. Then there is no polynomial-time deterministic algorithm for* Avoid *on circuits with m outputs using k circuit-inversion oracle queries.*

Proof. Let m and k be defined as above and let $i\mathcal{O}$ be a JLS-secure indistinguishability obfuscator. To prove that Avoid with the given parameters cannot be solved in deterministic polynomial time, it suffices to show that for every k-query oracle algorithm A, there exists a circuit C mapping n-bits to m-bits and a (consistent) inversion oracle $\mathcal{O}(\cdot) : \{0,1\}^m \to \{0,1\}^n$ such that $A^{\mathcal{O}(\cdot)}(C)$ outputs a y where $C(\mathcal{O}(y)) = y$ (in which case, A fails to solve Avoid on C). For any polynomial-time algorithm A that makes k queries to $\mathcal{O}(\cdot) : \{0,1\}^m \to \{0,1\}^n$, we can decompose A into $k+1$ polynomial-time algorithms $A_1, A_2, \ldots, A_{k+1}$ (without oracle queries) that work as follows:

- A_1: Given the input circuit C, it computes $y_1 = A_1(C)$ and queries $\mathcal{O}(y_1)$.
- A_2: Letting x_1 be the answer to the last query, it computes $y_2 = A_2(C, x_1)$ and queries $\mathcal{O}(y_2)$.
- A_3: Letting x_2 be the answer to the last query, it computes $y_3 = A_3(C, x_1, x_2)$ and queries $\mathcal{O}(y_3)$.
- $\ldots$
- A_{k+1}: Letting x_k be the answer to the last query, it computes $y_{k+1} = A_{k+1}(C, x_1, \ldots, x_k)$ and outputs y_{k+1}.

Therefore to rule out the existence of the oracle algorithm A as described above, it suffices to show that for all deterministic polynomial-time algorithms $A_1, \ldots, A_{k+1}$, there is a circuit $C : \{0,1\}^n \to \{0,1\}^m$, strings $y_1, \ldots, y_{k+1} \in \{0,1\}^m$, and strings $x_1, \ldots, x_{k+1} \in \{0,1\}^n$ such that:

- **(Oracle Consistency).** For all $i, j \in [k+1]$ such that $y_i = y_j$, we have $x_i = x_j$.
  (That is, the oracle gives consistent answers x_i to input strings y_i.)
- **(Oracle Inverting).** For every $i \in [k+1]$, we have $C(x_i) = y_i = A_i(C, x_1, \ldots, x_{i-1})$.
  (That is, given y_i, the oracle indeed provides an x_i such that $C(x_i) = y_i$.)

To add more detail, the aforementioned circuit C and any oracle satisfying $\mathcal{O}(y_i) = x_i$ for every $i \in [k+1]$ can force each A_i (equivalently, the i-th oracle query of A) to output y_i. In such a case, A_{k+1} (equivalently, the oracle algorithm A) outputs a string $y_{k+1} = C(x_{i+1})$ that is in the output range of the input circuit C, and thus **does not** solve Avoid.

We first introduce some notation. For n-variable 3-CNF formulas $\varphi_1, \ldots, \varphi_k$ of size n and strings $y_1, \ldots, y_k \in \{0,1\}^m$, we let $C[\varphi_1, \ldots, \varphi_k; y_1, \ldots, y_k]$ denote a polynomial-sized circuit that takes an input $(x, i) \in \{0,1\}^n \times [k]$ and outputs

$$C[\varphi_1, \ldots, \varphi_k; y_1, \ldots, y_k](x, i) := \begin{cases} 0^m, & \text{if } \varphi_i(x) = 0 \\ y_i, & \text{if } \varphi_i(x) = 1 \,. \end{cases}$$

In the case that we do not specify a φ_i and its corresponding y_i, such as in $C[\varphi_1, \varphi_2; y_1, y_2]$ for $k = 3$, we adopt the convention that any missing φ_i is the trivial unsatisfiable formula $\perp$ and $y_i = 0^m$. For instance when $k = 3$, $C[\varphi_1, \varphi_2; y_1, y_2] := C[\varphi_1, \varphi_2, \perp; y_1, y_2, 0^m]$.

Fix any polynomial-time algorithm A with k oracle queries and its decomposition as polynomial-time algorithms $A_1, A_2, \ldots, A_{k+1}$. We begin by making the following claim.

Claim 22. *For all $j \in \{0, \ldots, k+1\}$ and for all sufficiently large n, there exist n-variable satisfiable 3-CNF formulas $\varphi_1, \ldots, \varphi_j$ of size n, strings $x_1, \ldots, x_j \in \{0,1\}^n$, and strings $y_1, \ldots, y_j \in \{0,1\}^m$ such that the following holds.*

- *For all distinct $i_1, i_2 \in [j]$, if $\varphi_{i_1} = \varphi_{i_2}$, then $x_{i_1} = x_{i_2}$.*
- *For every $i \in [j]$, $\varphi_i(x_i) = 1$.*
- *Let $\hat{C}_j = i\mathcal{O}(C[\varphi_1, \ldots, \varphi_j; y_1, \ldots, y_j]; r)$. Over the random seed r of $i\mathcal{O}$, it holds with probability at least $2^{-\Omega((2k-j)m)}$ that for every $i \in [j]$, $y_i = A_i(\hat{C}_j, x_1, \ldots, x_{i-1})$.*

Observe that when $j = k+1$, Claim 22 implies the existence of $\varphi_1, \ldots, \varphi_{k+1}, x_1, \ldots, x_{k+1}$, and $y_1, \ldots, y_{k+1}$ that satisfy the aforementioned *Oracle Consistency* property (which follows from the first bullet of Claim 22) and *Oracle Inverting* property (which follows from the second and the third bullet of Claim 22 and the perfect functionality of $i\mathcal{O}$). Indeed, this claim shows that the polynomial-time algorithm A with k circuit-inversion oracle queries will **fail** on $\hat{C}_{k+1}$ (A will output a string in the range of $\hat{C}_{k+1}$) with probability at least $2^{-\Omega((k-1)m)}$. Therefore, to prove the theorem, it remains to prove Claim 22.

Before we start, one crucial definition is in order. Define a circuit D to be *j-good* if for every $i \in [j]$, $y_i = A_i(D, x_1, \ldots, x_{i-1})$. In other words, D is *j-good* if it satisfies the property in the third bullet of Claim 22.

We prove Claim 22 by induction on j. The base case $j = 0$ is trivially true. Now we assume the claim is true for $j-1$, which gives 3-CNF formulas $\varphi_1, \ldots, \varphi_{j-1}$ of size n, strings $x_1, \ldots, x_{j-1}$, and strings $y_1, \ldots, y_{j-1}$ that make the claim true for $j-1$. Let $\lambda = \mathsf{poly}(m)$ be the security parameter to be determined later, and let $\ell = \mathsf{poly}(n, \lambda)$ be the randomness complexity of $i\mathcal{O}$ with security parameter λ. We define an AM protocol $\mathcal{P}$, detailed in Algorithm 1, that attempts to solve UNSAT (i.e., the Prover aims to convince the Verifier that a given formula is unsatisfiable). For simplicity, we assume without loss of generality that our formulas are 3-CNFs with n clauses and n variables.

```
  Input: A 3-CNF formula $\varphi(x)$ on $n$ variables and clauses.
1 Prover sends 3-CNF formulas $\varphi_1, \ldots, \varphi_{j-1}$, strings $x_1, \ldots, x_{j-1} \in \{0,1\}^n$, and $y_1, \ldots, y_j \in \{0,1\}^m$;
2 Verifier rejects if $\varphi_i(x_i) \neq 1$ for some $i \in [j]$, or $\varphi_{i_1} = \varphi_{i_2}$ and $x_{i_1} \neq x_{i_2}$ for distinct $i_1, i_2 \in [j]$;
  // For $r \in \{0,1\}^\ell$, let $C_j^r := i\mathcal{O}(C[\varphi_1, \ldots, \varphi_{j-1}, \varphi; y_1, \ldots, y_j]; r)$.
  // Let $E : \{0,1\}^\ell \to \{0,1\}$ be a circuit such that $E(r) = 1$ if and only if $C_j^r$ is $j$-good.
3 Prover and Verifier run the Goldwasser-Sipser protocol (Lemma 13) on the instance $(E, \delta)$;
  // The parameter $\delta = 2^{\ell - \Omega((2k-j)m)}$.
  // Prover aims to convince Verifier that $|\{r \in \{0,1\}^\ell \mid E(r) = 1\}| \geq \delta$.
```

Algorithm 1: The AM protocol $\mathcal{P}$ that aims to solve UNSAT.

Completeness of $\mathcal{P}$. We now show that the AM protocol $\mathcal{P}$ is complete; the Verifier accepts every unsatisfiable 3-CNF formula φ on n variables with probability at least $2/3$. Let φ be an arbitrary unsatisfiable formula, and let $\hat{C}_{j-1}$ be the random variable $\hat{C}_{j-1} := i\mathcal{O}(C[\varphi_1, \ldots, \varphi_{j-1}; y_1, \ldots, y_{j-1}]; r)$ defined over $r \in \{0,1\}^\ell$. By the induction hypothesis, we know that

$$\Pr_{r \in \{0,1\}^\ell}\left[\hat{C}_{j-1} \text{ is } (j-1)\text{-good}\right] \geq 2^{-\Omega((2k-j+1)m)}.$$

By an averaging argument, there is a $y \in \{0,1\}^m$ such that

$$\Pr_{r \in \{0,1\}^\ell}\left[[\hat{C}_{j-1} \text{ is } (j-1)\text{-good}] \wedge [y = A_j(\hat{C}_{j-1}, x_1, \ldots, x_{j-1})]\right] \geq 2^{-\Omega((2k-j+1)m)} \cdot 2^{-m} = 2^{-\Omega((2k-j)m)}. \quad (5)$$

Let $y_j \in \{0,1\}^m$ be one such y, and let $\hat{C}_j$ be the random variable $\hat{C}_j := i\mathcal{O}(C[\varphi_1, \ldots, \varphi_{j-1}, \varphi; y_1, \ldots, y_j]; r)$ defined over $r \in \{0,1\}^\ell$. Since φ is unsatisfiable, it follows that

$$C[\varphi_1, \ldots, \varphi_{j-1}, \varphi; y_1, \ldots, y_j] \quad \text{and} \quad C[\varphi_1, \ldots, \varphi_{j-1}; y_1, \ldots, y_{j-1}]$$

compute the same function. By the JLS-security of $i\mathcal{O}$, we know that $\hat{C}_j$ and $\hat{C}_{j-1}$ are $2^{-\lambda^\varepsilon}$-indistinguishable against any polynomial-sized adversary. To verify that a circuit $\hat{C}$ of $\mathsf{poly}(n)$ size is j-good, we need to check $y_i = A_i(\hat{C}, x_1, \ldots, x_{i-1})$ for every $i \in [j]$, which can be done by a circuit of $\mathsf{poly}(n)$ size. This means by Equation (5) that

$$\begin{aligned}\Pr_{r \in \{0,1\}^\ell}\left[\hat{C}_j \text{ is } j\text{-good}\right] &= \Pr_{r \in \{0,1\}^\ell}\left[[\hat{C}_j \text{ is } (j-1)\text{-good}] \wedge [y_j = A_j(\hat{C}_j, x_1, \ldots, x_{j-1})]\right] \\ &\geq 2^{-\Omega((2k-j)m)} - 2^{-\lambda^\epsilon}. \end{aligned} \quad (6)$$

Let $\lambda := m^{2/\epsilon} = \mathsf{poly}(n)$ and $\delta := 2^\ell \cdot (6) = 2^{\ell - \Omega(2k-j)m}$. The Prover will work as follows. In the first step, the Prover sends 3-CNF formulas $\varphi_1, \ldots, \varphi_{j-1}$, strings $x_1, \ldots, x_{j-1} \in \{0,1\}^n$, and $y_1, \ldots, y_j \in \{0,1\}^m$. By Equation (6), we know that $\hat{C}_j$ is j-good with probability at least $\delta/2^\ell$, which means by the definition of $E : \{0,1\}^\ell \to \{0,1\}$ (which can be implemented in $\mathsf{poly}(n)$ size) that there is a Prover for the Goldwasser-Sipser protocol in Line 3 of Algorithm 1 such that the Verifier accepts with probability at least $2/3$. This concludes the completeness of the protocol $\mathcal{P}$.

Employ the Lack of Soundness. At this point, we have shown that $\mathcal{P}$ is a polynomial-time AM protocol attempting to check unsatisfiability, and that $\mathcal{P}$ has the completeness property. By the assumption that coNP is not infinitely often in AM, $\mathcal{P}$ cannot solve unsatisfiability even infinitely often, which means that this $\mathcal{P}$ does not have soundness for *all* sufficiently large n. In other words, there is a Prover such that for sufficiently large n, the Verifier accepts some satisfiable 3-CNF formula φ on n variables with probability $> 1/3$. Let φ_j be this formula φ and let

- 3-CNF formulas $\varphi_1, \varphi_2, \ldots, \varphi_{j-1}$,
- strings $x_1, x_2, \ldots, x_{j-1} \in \{0,1\}^n$, and
- strings $y_1, y_2, \ldots, y_j \in \{0,1\}^m$

be the message sent in Line 1 (of Algorithm 1) by this Prover on the input φ_j. Since the Verifier does not reject in Line 2, we have

- $\varphi_i(x_i) = 1$ for every $i \in [j-1]$, and
- for all distinct $i_1, i_2 \in [j]$, if $\varphi_{i_1} = \varphi_{i_2}$, then $x_{i_1} = x_{i_2}$.

We define the string $x_j \in \{0,1\}^n$ to be x_i if there is some $i \in [j-1]$ such that $\varphi_i = \varphi_j$; otherwise, we set x_j to be an arbitrary n-bit string such that $\varphi_j(x_j) = 1$. Now we show that the formulas $\varphi_1, \ldots, \varphi_j$, strings $x_1, \ldots, x_j \in \{0,1\}^n$, and strings $y_1, \ldots, y_j \in \{0,1\}^m$ satisfy the conditions of Claim 22, which will conclude the proof.

Let $\hat{C}_j$ be the random variable defined as $\hat{C}_{j-1} := i\mathcal{O}(C[\varphi_1, \ldots, \varphi_j; y_1, \ldots, y_j]; r)$. Since the Verifier accepts with probability $> 1/3$, by the soundness of the Goldwasser-Sipser protocol (Lemma 13), we know that

$$\Pr_{r \in \{0,1\}^\ell}\left[\hat{C}_j \text{ is } j\text{-good}\right] \geq \frac{1}{3} \cdot 2^{-\ell} \cdot \delta = 2^{-\Omega((2k-j)m)}.$$

This implies the third bullet of Claim 22. The first two bullets hold by the definition of φ_i, x_i, and y_i. □

By combining Theorem 20 and Theorem 21, we know that $\mathsf{dWPHP}_\ell(\mathsf{Eval})$ is not provable in $\mathsf{T_{PV}}$ based on the assumptions of Theorem 21. Since $\mathsf{T_{PV}}$ is an extension of PV_1, this further means that $\mathsf{dWPHP(PV)}$ is not provable in PV, which separates PV_1 and APC_1. We summarize the results as follows.

Corollary 23. *Assume the existence of JLS-secure $i\mathcal{O}$ and coNP is not infinitely often in AM. For every constructive function $\ell(n) \leq \mathsf{poly}(n)$ such that $\ell(n) > n$, $\mathsf{T_{PV}} \nvdash \mathsf{dWPHP}_\ell(\mathsf{Eval})$. In particular, APC_1 is a strict extension of PV_1.*

4.3 An Extension: Separating UAPC_1 with APC_1

Given the conditional separation of PV_1 and APC_1, it is natural to further investigate the bounded theories in between. For instance, we may ask whether there are non-trivial fragments of APC_1 (i.e. equal to neither PV_1 nor APC_1) that are strong enough to formalize circuit lower bounds or sustain Jeřábek's framework for approximate counting [Jeř07a]. We initiate the study of this question by proving that under plausible assumptions, an important fragment of APC_1, which we call UAPC_1 (stands for Uniform Approximate Counting), is a strict sub-theory of APC_1.

Let f be a function. We define the *uniform dual Weak Pigeonhole Principle* for f, denoted by $\mathsf{dWPHP}'(f)$, be the sentence[14]

$$\mathsf{dWPHP}'(f) := \forall n > 0\, \forall m \in \mathsf{Log}\, \exists y < n(m+1)\, \forall x < nm\; f(x) \neq y, \tag{7}$$

which says that $f : [nm] \to [n(m+1)]$ cannot be surjective. We define

$$\mathsf{dWPHP}'(\mathsf{PV}) := \{\mathsf{dWPHP}'(f) \mid f \in \mathcal{L}(\mathsf{PV})\}$$

and $\mathsf{UAPC}_1 := \mathsf{PV}_1 + \mathsf{dWPHP}'(\mathsf{PV})$.

The main difference between $\mathsf{dWPHP}'(f)$ and $\mathsf{dWPHP}(f)$ is that $\mathsf{dWPHP}'(f)$ does not allow the function f to take extra parameters $\vec{w}$ in Equation (1). Intuitively speaking, it only states that any *uniform* PV function f whose codomain is sufficiently larger than its domain cannot be surjective, and says nothing about the non-uniform functions.

[14] We can also define dWPHP'_ℓ with stretch function ℓ as $\mathsf{dWPHP}'_\ell(f) := \forall n \in \mathsf{Log}\, \exists y \in \{0,1\}^{\ell(n)}\, \forall x \in \{0,1\}^n\, f(x) \neq y$. Since $\mathsf{dWPHP}'(f)$ defined by Equation (7) denotes the dual Weak Pigeonhole Principle for $f : [nm] \to [n(m+1)]$, where the size of the codomain is even smaller than twice of the size of the domain, $\mathsf{dWPHP}'(f)$ implies $\mathsf{dWPHP}_\ell(f)$ for every $\ell(n) \geq n+1$ in any reasonable base theory.

Despite being (seemingly) weaker, UAPC_1 is known to formalize many results based on approximate counting and probabilistic methods, e.g., the existence of hard Boolean functions and rigid matrices. In particular, as observed by Pich and Santhanam [PS21], UAPC_1 proves the main theorem in [Jeř07a] for approximate counting via Nisan-Wigderson generators (see Appendix D for details).

Let $\mathsf{T}^0_{\mathsf{APC}} := \mathsf{T}_{\mathsf{PV}} + \mathsf{dWPHP}'(\mathsf{PV})$ be an extension of UAPC_1. We will prove the following result.

Theorem 24 (Separating UAPC_1 and APC_1)**.** *Assume the existence of JLS-secure $i\mathcal{O}$ and* coNP *is not infinitely often in* $\mathsf{NP}_{/\mathsf{poly}}$. *For every constructive function $\ell(n)$ such that $n < \ell(n) \le \mathsf{poly}(n)$,* $\mathsf{T}^0_{\mathsf{APC}} \nvdash \mathsf{dWPHP}_\ell(\mathsf{Eval})$. *In particular,* APC_1 *is a strict extension of* UAPC_1.

The Consequence of the Provability of dWPHP **in** UAPC_1**.** We will use the following KPT-style witnessing theorem for $\mathsf{T}^0_{\mathsf{APC}}$ due to Pich and Santhanam [PS21], where the witnessing functions are computable by polynomial-sized circuits instead of uniform algorithms. (Note that this is the reason that we need to assume non-uniform coNP lower bounds in Theorem 24.)

Theorem 25 (KPT Witnessing for $\mathsf{T}^0_{\mathsf{APC}}$ [PS21, Theorem 4])**.** *For every quantifier-free formula $\varphi(\vec{x}, y, z)$ in the language $\mathcal{L}(\mathsf{PV})$, if $\mathsf{T}^0_{\mathsf{APC}} \vdash \forall \vec{x}\ \exists y\ \forall z\ \varphi(\vec{x}, y, z)$, then there are a constant $k \in \mathbb{N}$ and k functions (in the standard model) $f_1(\vec{x}), f_2(\vec{x}, z_1), \dots, f_k(\vec{x}, z_1, \dots, z_{k-1})$ such that the following holds.*

For every $\vec{n} = (n_1, \dots, n_t) \in \vec{\mathbb{N}}$ and for every $m_1, m_2, \dots, m_k \in \mathbb{N}$, it holds in the standard model that:

- *either $\varphi(\vec{n}, f_1(\vec{n}), m_1)$ is true;*
- *or $\varphi(\vec{n}, f_2(\vec{n}, m_1), m_2)$ is true;*
- *or $\varphi(\vec{n}, f_3(\vec{n}, m_1, m_2), m_3)$ is true;*
- *$\dots$;*
- *or $\varphi(\vec{n}, f_k(\vec{n}, m_1, \dots, m_{k-1}), m_k)$ is true.*

Moreover, each f_i is computable by a family of polynomial-sized deterministic circuits.

Similar to algorithms for Avoid with circuit-inversion oracles (see Definition 19), we can define its non-uniform counterpart, where the oracle queries are implemented as a special gate of the circuits.

Definition 26 (Circuits for Avoid with Circuit-Inversion Oracle Gates)**.** Let $m = m(n)$ and $k = k(n)$. A *polynomial-sized circuit family with k circuit-inversion oracle gates* for Avoid with stretch m is a polynomial-sized circuit family $\{F_s\}$ where each F_s contains at most k oracle gates of fan-in s and fan-out s.

Moreover, $\{F_s\}$ is said to solve Avoid if for every circuit $C : \{0,1\}^n \to \{0,1\}^m$ of size s and any $\mathcal{O} : \{0,1\}^m \to \{0,1\}^n$ such that $\mathcal{O}(y)$ outputs an $x \in \{0,1\}^n$ satisfying $C(x) = y$ provided that such an x exists, $F_s(C)$ outputs a $y \in \{0,1\}^m$ satisfying $C(\mathcal{O}(y)) \neq y$, when we interpret the fan-in-s oracle gates in F_s as computing $\mathcal{O}$, where the $s - m$ unused input bits are ignored and the $s - n$ unused output bits always output 0.

Theorem 27. *For every constructive function $\ell = \mathsf{poly}(n)$, if $\mathsf{T}^0_{\mathsf{APC}} \vdash \mathsf{dWPHP}_\ell(\mathsf{Eval})$, then* Avoid *with stretch ℓ is computable by a family of polynomial-sized circuits with $O(1)$ circuit-inversion oracle gates.*

Proof. Let $\ell(n) = \mathsf{poly}(n)$ be some constructive function. By the assumptions, we know that

$$\mathsf{T}^0_{\mathsf{APC}} \vdash \forall n\ \forall C\ \exists y \in \{0,1\}^{\ell(n)}\ \forall x \in \{0,1\}^n\ \mathsf{Eval}(C, x) \neq y.$$

By Theorem 25, there is a $k = O(1)$ and k functions $f_1, \dots, f_k$ computable by families of polynomial-sized circuits such that for every n, every circuit $C : \{0,1\}^n \to \{0,1\}$, and every $x_1, x_2, \dots, x_n$:

- either $C(x_1) \neq f_1(n, C)$;
- or $C(x_2) \neq f_2(n, C, x_1)$;
- or $C(x_3) \neq f_3(n, C, x_1, x_2)$;
- $\dots$
- or $C(x_k) \neq f_k(n, C, x_1, \dots, x_k)$.

Now we show that AVOID is also computable by families of polynomial-sized circuits with at most k circuit-inversion oracle gates. Let s be the input length (i.e. we are dealing with circuits encoded by an s-bit string) and consider the following circuit:

1. Given the input circuit C, we compute its input length n and $y_1 = f_1(n, C) \in \{0,1\}^{\ell(n)}$.
2. We feed y_1 to the circuit-inversion oracle gate and let x_1 be the output of it. If $C(x_1) \neq y_1$, we output y_1; otherwise, we compute $y_2 = f_2(n, C, x_1) \in \{0,1\}^{\ell(n)}$.
3. We feed y_2 to the circuit-inversion oracle gate and let x_2 be the output of it. If $C(x_2) \neq y_2$, we output y_2; otherwise, we compute $y_3 = f_3(n, C, x_1, x_2) \in \{0,1\}^{\ell(n)}$.
4. $\dots$

k. We feed y_{k-1} to the circuit-inversion oracle gate and let x_k be the output of it. By the discussion above, $C(x_k) \neq y_k$, where $y_k := f_k(n, C, x_1, \dots, x_{k-1}) \in \{0,1\}^{\ell(n)}$. We output y_k.

The correctness of this circuit is obvious. Since each f_i is computable by a family of polynomial-sized circuits, we know that this circuit is of polynomial size and contains $k = O(1)$ circuit-inversion oracle gates. □

Impossibility of Solving AVOID by Circuits with Circuit-Inversion Oracle Gates. To prove Theorem 24, it now suffices to rule out small circuits solving AVOID with $O(1)$ circuit-inversion oracle gates.

Theorem 28. *Let $m = m(n) = \mathsf{poly}(n)$ and $k = O(1)$ such that $m \geq n + 1$. Assume that coNP is not infinitely often in $\mathsf{NP}_{/\mathsf{poly}}$ and $i\mathcal{O}$ with JLS-security exists. Then AVOID with m outputs is not computable by a family of polynomial-sized circuits with at most k circuit-inversion oracle gates.*

Proof Sketch. We follow the proof of Theorem 21, where the witnessing algorithms $A_1, A_2, \dots, A_{k+1}$ are now (families of) polynomial-sized circuits. The only problem is that the AM protocol $\mathcal{P}$ for UNSAT (see Algorithm 1) needs to be replaced by a polynomial-sized nondeterministic circuit. Here we only show how to deal with this issue.

Towards a contradiction, we assume that AVOID with m outputs is computable by an algorithm A that is computable by a family of polynomial-sized circuits with at most k circuit-inversion oracle gates. As in Theorem 21, we can decompose A into $k+1$ algorithms $A_1, A_2, \dots, A_{k+1}$ such that each A_i is computable by a family of polynomial-sized circuits *without* circuit-inversion oracle gates. Let $\mathcal{P}$ be the AM-protocol in Algorithm 1 constructed from the algorithms $A_1, A_2, \dots, A_{k+1}$ here. Since each A_i is computable by a family of polynomial-sized circuits, the protocol $\mathcal{P}$ will be a three-round public-key protocol with a verifier V computable by a family of polynomial-sized circuits. Let n be the length (and the number of variables) of the input formula φ. The protocol will have the following structure:

1. Prover sends a string s_1 of length $\ell_1 = \mathsf{poly}(n)$.
2. Verifier generates a random string r of length $\ell_r = \mathsf{poly}(n)$.
3. Prover sends a message s_2 of length $\ell_2 = \mathsf{poly}(n)$.
4. Verifier accepts φ if and only if $V(\varphi, s_1, s_2, r) = 1$.

Moreover, following the proof of Theorem 21, we can show that this protocol is *complete*, that is for every unsatisfiable formula $\varphi \in \{0,1\}^n$, there is an $s_1 \in \{0,1\}^{\ell_1}$, such that with probability at least $2/3$ over the choice of $r \in \{0,1\}^{\ell_r}$, there is an $s_2 \in \{0,1\}^{\ell_2}$ such that $V(\varphi, s_1, s_2, r) = 1$. To complete the proof, it suffices to prove that the protocol cannot be sound even infinitely often. (The rest of the proof follows from the proof of Theorem 21, see the paragraph *Employ the Lack of Soundness*.)

Towards a contradiction, we assume that for infinitely many n, every satisfiable formula $\varphi \in \{0,1\}^n$, and every $s_1 \in \{0,1\}^{\ell_1}$, it holds that with probability at most $1/3$ over the choice of $r \in \{0,1\}^{\ell_r}$, there is an $s_2 \in \{0,1\}^{\ell_2}$ such that $V(\varphi, s_1, s_2, r) = 1$. For every such n, $\mathcal{P}$ is a *sound and complete* AM-protocol for UNSAT on input length n. Therefore, it suffices to translate $\mathcal{P}$ on every such input length n into an equivalent $\mathsf{poly}(n)$-sized nondeterministic circuit, since this will contradict the assumption that NP is not infinitely often in $\mathsf{NP}_{/\mathsf{poly}}$. This translation follows from the fact that $\mathsf{promiseAM}_{/\mathsf{poly}} \subseteq \mathsf{promiseNP}_{/\mathsf{poly}}$. For completeness, we provide a self-contained proof of the result.

Let $k_{\mathsf{rep}} = \mathsf{poly}(n)$ be a parameter to be determined later and $V_{\mathsf{AM}} : \{0,1\}^n \times (\{0,1\}^{\ell_r})^{k_{\mathsf{rep}}} \to \{0,1\}$ be the following polynomial-sized nondeterministic circuit:

- Given the input $\varphi \in \{0,1\}^n$ and $r_1, r_2, \ldots, r_{k_{\mathsf{rep}}} \in \{0,1\}^{\ell_r}$, *accept* if and only if there exist $s_1 \in \{0,1\}^{\ell_1}$ and $s_2 \in \{0,1\}^{\ell_2}$ such that for at least $k_{\mathsf{rep}}/2$ of $i \in [k_{\mathsf{rep}}]$, $V(\varphi, s_1, s_2, r_i) = 1$.

We treat V_{AM} be a randomized nondeterministic circuit for UNSAT, where the second input will be uniformly chosen. That is, we repeat the protocol $\mathcal{P}$ for k_{rep} times with independent random strings and output the majority output of the protocol. We will set k_{rep} to be a sufficiently large polynomial so that the error probability of the randomized algorithm is at most 2^{-2n} and a good random string can be fixed as advice.

- **(Completeness).** Suppose $\varphi \in \{0,1\}^n$ is an unsatisfiable formula. We know by the completeness of $\mathcal{P}$ (Theorem 21) that there is a string s_1 such that
$$\Pr_{r \in \{0,1\}^{\ell_r}} \left[\exists s_2 \in \{0,1\}^{\ell_2}\ V(\varphi, s_1, s_2, r) = 1\right] \geq 2/3. \tag{8}$$
Fix this string s_1. For every $i \in [k_{\mathsf{rep}}]$, let X_i be a random variable defined over the i-th random string $r_i \in \{0,1\}^{\ell_r}$ of V_{AM}, where $X_i = 1$ if and only if there is an $s_2 \in \{0,1\}^{\ell_2}$ such that $V(\varphi, s_1, s_2, r) = 1$. It follows from eq. (8) that $\mathbb{E}[X_i] \geq 2/3$. By the Chernoff bound,
$$\Pr\left[X_1 + \cdots + X_{k_{\mathsf{rep}}} < k_{\mathsf{rep}}/2\right] \leq 2^{-\Omega(k_{\mathsf{rep}})}.$$
This implies that, for a uniformly chosen random string $(r_1, \ldots, r_{k_{\mathsf{rep}}}) \in \{0,1\}^{\ell_r k_{\mathsf{rep}}}$, V_{AM} accepts φ with probability at least $1 - 2^{-\Omega(k_{\mathsf{rep}})}$.
- **(Soundness).** Suppose $\varphi \in \{0,1\}^n$ is a satisfiable formula, we know by the soundness of $\mathcal{P}$ on input length n that for every s_1,
$$\Pr_{r \in \{0,1\}^{\ell_r}} \left[\exists s_2 \in \{0,1\}^{\ell_2}\ V(\varphi, s_1, s_2, r) = 1\right] \leq 1/3. \tag{9}$$
Fix an arbitrary $s_1 \in \{0,1\}^{\ell_1}$. For every $i \in [k_{\mathsf{rep}}]$, let $Y_i^{s_1}$ be a random variable defined over the i-th random string $r_i \in \{0,1\}^{\ell_r}$ of V_{AM}, where $Y_i^{s_1} = 1$ if and only if there is an $s_2 \in \{0,1\}^{\ell_2}$ such that $V(\varphi, s_1, s_2, r) = 1$. It follows from eq. (9) that $\mathbb{E}[Y_i^{s_1}] \leq 1/3$. By the Chernoff bound,
$$\Pr\left[Y_1^{s_1} + \cdots + Y_{k_{\mathsf{rep}}}^{s_1} \geq k_{\mathsf{rep}}/2\right] \leq 2^{-\Omega(k_{\mathsf{rep}})}.$$
Therefore we can see that
$$\begin{aligned} \Pr\left[V_{\mathsf{AM}} \text{ accepts } \varphi\right] &= \Pr\left[\exists s_1 \in \{0,1\}^{\ell_1}\ Y_1^{s_1} + \cdots + Y_{k_{\mathsf{rep}}}^{s_1} \geq k_{\mathsf{rep}}/2\right] \\ &\leq \sum_{s_1 \in \{0,1\}^{\ell_1}} \Pr[Y_1^{s_i} + \cdots + Y_{k_{\mathsf{rep}}}^{s_i} \geq k_{\mathsf{rep}}/2] && \text{(Union Bound)} \\ &\leq 2^{\ell_1 - \Omega(k_{\mathsf{rep}})}. && \text{(Equation (9))} \end{aligned}$$

By setting $k_{\mathsf{rep}} = \ell_1 + O(n) \leq \mathsf{poly}(n)$, we can set the error probability of V_{AM} to be less than 2^{-2n}. Therefore by a union bound, we can fix a "good" random string $\hat{r}$ such that $V_{\mathsf{AM}}(\cdot, \hat{r})$ solves UNSAT on input length n. This completes the proof. □

5 The Oracle Derandomization Hypothesis for Time-Bounded Kolmogorov Complexity

In this section, we investigate the Oracle Derandomization Hypothesis (ODH) and its variants.

Hypothesis 29 (Oracle Derandomization Hypothesis [FS11]). *For any m with $n \leq m \leq \mathsf{poly}(n)$, there is a deterministic algorithm A mapping m bits to n bits such that for all $z \in \{0,1\}^m$ we have that $y = A(z)$ satisfies that the circuit complexity*[15] *of y given oracle access to z is at least $n^{\Omega(1)}$ (when y and z are viewed as truth tables in the natural way).*

[15] Actually, [FS11] considers nondeterministic circuit complexity instead of the usual circuit complexity.

We note that ODH is closely related to the COMPLEXITY problem studied by Kleinberg, Korten, Mitropolsky, and Papadimitriou [KKMP21], which is defined as follows:

Search Problem: COMPLEXITY
Input: A length-n truth table z of a Boolean function.
Output: A length-n truth table y such that the circuit complexity of the function represented by y given oracle access to the function represented by z is at least $\Omega(\frac{n}{\log^2 n})$.

There are two main differences between ODH, and having a deterministic polynomial-time algorithm for COMPLEXITY. First, to solve COMPLEXITY, one is interested in truth tables with near-maximal conditional circuit complexity, while in ODH it suffices to output truth tables that have conditional circuit complexity $n^{\Omega(1)}$. Second, in COMPLEXITY, $|z| = |y|$, while in ODH one needs to handle cases where $|y|$ is polynomially smaller than $|z|$.

We consider a time-bounded Kolmogorov complexity version of ODH. Roughly speaking, this version says that given a string z of length n, one can efficiently deterministically generate a string y of length $n^{.01}$ such that the time-bounded Kolmogorov complexity of y given z is large. We give a formal definition below.

Hypothesis 30 (Time-Bounded Kolmogorov Complexity Oracle Derandomization Hypothesis)**.** *For every m with $n \leq m \leq \mathsf{poly}(n)$ and for any $t = \mathsf{poly}(n)$, there is a deterministic algorithm A mapping m bits to n bits such that for all $z \in \{0,1\}^m$ we have that $y = A(z)$ is a string such that $\mathsf{K}^t(y|z) = n^{\Omega(1)}$.*

Theorem 31. *Hypothesis 30 is false assuming $\mathsf{NP} \neq \mathsf{coNP}$ and JLS-secure $i\mathcal{O}$ exists.*

Proof. The proof is very similar to the proof of Theorem 2.

Let $m = \mathsf{poly}(n)$ and $t = \mathsf{poly}(n)$ be parameters we set later. For contradiction, let A be an algorithm mapping m bits to n bits such that for all $z \in \{0,1\}^m$ we have $\mathsf{K}^t(y|z) = \Omega(n^\epsilon)$ for some $\epsilon > 0$, where $y = A(z)$.

We give a polynomial-time nondeterministic algorithm for checking if an $n^{\epsilon/2}$-variable formula φ is unsatisfiable.

1. Nondeterministically guess a $y \in \{0,1\}^n$ and an $r \in \{0,1\}^{\mathsf{poly}(n+\lambda)}$ where $\lambda = \mathsf{poly}(n)$ is the security parameter of the JLS-secure $i\mathcal{O}$ to be determined later.
2. Let $C[\varphi, y]$ denote a $\mathsf{poly}(n)$-size circuit that takes n input bits and outputs m bits and satisfies
$$C[\varphi, y](x) = \begin{cases} 0^m, & \text{if } \varphi(x) = 0 \\ y, & \text{if } \varphi(x) = 1\ . \end{cases}$$
3. Set z to be the m-bit string given by the description of the circuit $i\mathcal{O}(C[\varphi, y]; r)$. (We set $m > n$ so that this is possible. We can also pad the description with zeroes if necessary.)
4. Accept if and only if $y = A(z)$.

It is easy to see that this is a polynomial-time nondeterministic algorithm. We now will show soundness and completeness for sufficiently large n.

First, we show soundness. If φ is satisfiable, then we claim that for all choices of y and r, we have $\mathsf{K}^t(y|z) \leq O(n^{\epsilon/2})$. This is because if φ is satisfiable, then there is an input $x^\star \in \{0,1\}^{n^{\epsilon/2}}$ such that $\varphi(x^\star) = 1$. Consequently, $C[\varphi, y](x^\star) = y$. Thus, when z is interpreted as the *description* of the circuit $i\mathcal{O}(C[\varphi, y]; r)$, we have $z(x^\star) = y$. This shows that $\mathsf{K}^t(y|z) \leq O(n^{\epsilon/2})$ (setting t to be a sufficiently large polynomial). This completes our proof of soundness.

Now we show completeness. Suppose φ is unsatisfiable. As in the proof of Theorem 2, we consider A being run on $i\mathcal{O}(C[\varphi, 0^n]; r)$ for uniformly random r. As A must output an element of $\{0,1\}^n$, we know there's a $y^\star$ satisfying
$$\Pr_r[y^\star = A(i\mathcal{O}(C[\varphi, 0^m]; r))] \geq 2^{-n}.$$

To this end, we (just as in Theorem 2) consider an adversary circuit B which takes in a circuit C' (ostensibly of the form $C' = i\mathcal{O}(C[\varphi, 0^m])$ and outputs 1 if and only if $y^\star = A(C')$. As in Theorem 2, we can argue that the size of the adversary B is small enough that our $i\mathcal{O}$ assumption applies.

As in Theorem 2, $C[\varphi, 0^n]$ and $C[\varphi, y^\star]$ compute the same function. Thus, by the JLS-security of $i\mathcal{O}$, applied to the adversary circuit B on the input circuits $i\mathcal{O}(C[\varphi, 0^n])$ and $i\mathcal{O}(C[\varphi, y^\star)$, we have

$$\begin{aligned}\Pr_r[y^\star = A(i\mathcal{O}(C[\varphi, y^\star]; r))] &\geq \Pr_r[y^\star = A(i\mathcal{O}(C[\varphi, 0^m]; r))] - 2^{-\lambda^\epsilon} \\ &\geq 2^{-n} - 2^{-\lambda^\epsilon} > 0\end{aligned}$$

by setting λ to be a sufficiently large polynomial in n. Thus, there is an r such that

$$y^\star = A(i\mathcal{O}(C[\varphi, y^\star]; r)),$$

so $\mathcal{A}$ accepts φ. □

6 Conclusion

We conclude with several open questions and directions of particular interest.

Intractability of $\mathcal{C}$-Avoid? For a given circuit class $\mathcal{C}$ (e.g., AC^0, TC^0, NC^1), Ren, Santhanam, and Wang [RSW22] introduce the $\mathcal{C}$-AVOID problem, which considers AVOID over circuits drawn from $\mathcal{C}$. They showed many interesting lower bound consequences from showing $\mathcal{C}$-AVOID is in FP (or even $\mathsf{FP}^{\mathsf{NP}}$).

Our work suggests the following natural question:

*What is the "weakest" circuit class $\mathcal{C}$ such that, under plausible assumptions, $\mathcal{C}$-*AVOID *is* not *in* FP*?*

It seems reasonable that NC-AVOID (i.e., RANGE AVOIDANCE over $\mathsf{poly}(\log n)$-depth circuits of $\mathsf{poly}(n)$ size) is not in FP, under similar assumptions to ours.

Intractability of Range Avoidance on Uniform Circuits? The specific instances used to show lower bound and combinatorial consequences of AVOID $\in$ FP [Kor22, RSW22, GLW22] arise from *uniform* circuits. Formally, for these instances, there is a deterministic machine M with an $O(1)$-bit description such that, given 1^n, M runs in $\mathsf{poly}(n)$ time and prints the description of a C_n on n inputs and $m > n$ outputs. Could AVOID on such uniformly-generated circuits be in FP? On the one hand, our arguments concluding AVOID $\notin$ FP evidently rely on the non-uniformity of input circuits in a crucial way. On the other hand, since (for example) NP-complete problems on uniformly-generated instances typically remain hard in a different way (e.g., NEXP-complete), some of the authors are skeptical that this special case of AVOID is easy.

The Landscape Around Range Avoidance. Prior to our work, there were no examples of a relational problem that has both an efficient randomized algorithm (i.e. in the class FBPP) and an inefficient deterministic algorithm (for example, in FPH), but was unlikely to have an efficient deterministic algorithm. (For comparison, Aaronson, Buhrman, and Kretschmer [ABK23] give an example of problem that is in $\mathsf{FEXP} \cap \mathsf{FBPP}$ but unconditionally not in FP.) AVOID has these properties, and (for large stretch) apparently lies in $(\mathsf{FBPP} \cap \mathsf{FPH}) \setminus \mathsf{FP}$, under the assumptions of this paper.[16] What are other examples of such problems? The space of such problems seems interesting to study, in itself.

The Structure of Probabilistic Feasible Reasoning. We have demonstrated that under plausible assumptions, PV_1 is a strict sub-theory of APC_1. Furthermore, the fragment UAPC_1 of APC_1 that sustains the basic mechanism of Jeřábek's approximate counting framework [Jeř07a] is strictly weaker than APC_1 under similar assumptions. This motivates revisiting the question of what is the "right" theory to capture probabilistic feasible reasoning. For instance, we may ask the following question: Is UAPC_1 conservative over PV_1? Is there a strict fragment of UAPC_1 that (in some sense) captures probabilistic feasible reasoning? Are there interesting mathematical theorems that are provable in APC_1 but not provable in UAPC_1 (or its weaker fragments)?

[16] Here, our notion of FBPP is that there is a randomized polynomial-time algorithm that generates a solution with probability at least $3/4$ (say). In our case, the error probability for solving AVOID directly depends on the stretch of the instance. See [ABK23] for further discussion on FBPP.

Acknowledgements.

We thank Lijie Chen, Yilei Chen, Aayush Jain, Antonina Kolokolova, Oliver Korten, Rachel Huijia Lin, Zhenjian Lu, Pavel Pudlák, Igor Carboni Oliveira, Hanlin Ren, Amit Sahai, Rahul Santhanam, Neekon Vafa, Vinod Vaikuntanathan, and Tianqi Yang for discussions on this work. Hanlin Ren pointed us to the Oracle Derandomization Hypothesis. Part of the work was done when Jiatu Li was visiting the University of Warwick as an intern of Igor Carboni Oliveira.

References

[AB09] Sanjeev Arora and Boaz Barak. *Computational Complexity - A Modern Approach*. Cambridge University Press, 2009.

[ABK23] Scott Aaronson, Harry Buhrman, and William Kretschmer. A qubit, a coin, and an advice string walk into a relational problem. *Electron. Colloquium Comput. Complex.*, TR23-015, 2023.

[AW09] Scott Aaronson and Avi Wigderson. Algebrization: A new barrier in complexity theory. *ACM Trans. Comput. Theory*, 1(1):2:1–2:54, 2009.

[Bab85] László Babai. Trading group theory for randomness. In *STOC*, pages 421–429. ACM, 1985.

[BGI$^+$01] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In *CRYPTO*, volume 2139 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2001.

[BGI$^+$12] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. *J. ACM*, 59(2):6:1–6:48, 2012.

[BGS75] Theodore P. Baker, John Gill, and Robert Solovay. Relativizations of the P =? NP question. *SIAM J. Comput.*, 4(4):431–442, 1975.

[BHZ87] Ravi B. Boppana, Johan Håstad, and Stathis Zachos. Does co-NP have short interactive proofs? *Inf. Process. Lett.*, 25(2):127–132, 1987.

[BIOW20] Ohad Barta, Yuval Ishai, Rafail Ostrovsky, and David J. Wu. On succinct arguments and witness encryption from groups. In *CRYPTO (1)*, volume 12170 of *Lecture Notes in Computer Science*, pages 776–806. Springer, 2020.

[BKO20] Jan Bydzovsky, Jan Krajícek, and Igor Carboni Oliveira. Consistency of circuit lower bounds with bounded theories. *Log. Methods Comput. Sci.*, 16(2), 2020.

[BKT14] Samuel R. Buss, Leszek Aleksander Kolodziejczyk, and Neil Thapen. Fragments of approximate counting. *J. Symb. Log.*, 79(2):496–525, 2014.

[BM20] Jan Bydzovsky and Moritz Müller. Polynomial time ultrapowers and the consistency of circuit lower bounds. *Arch. Math. Log.*, 59(1-2):127–147, 2020.

[BPR15] Nir Bitansky, Omer Paneth, and Alon Rosen. On the cryptographic hardness of finding a nash equilibrium. In *FOCS*, pages 1480–1498. IEEE Computer Society, 2015.

[Bus85] Samuel R Buss. *Bounded arithmetic*. Princeton University, 1985.

[BZ17] Dan Boneh and Mark Zhandry. Multiparty key exchange, efficient traitor tracing, and more from indistinguishability obfuscation. *Algorithmica*, 79(4):1233–1285, 2017.

[CGI$^+$16] Marco L. Carmosino, Jiawei Gao, Russell Impagliazzo, Ivan Mihajlin, Ramamohan Paturi, and Stefan Schneider. Nondeterministic extensions of the strong exponential time hypothesis and consequences for non-reducibility. In *ITCS*, pages 261–270. ACM, 2016.

[CKKO21] Marco Carmosino, Valentine Kabanets, Antonina Kolokolova, and Igor Carboni Oliveira. Learn-uniform circuit lower bounds and provability in bounded arithmetic. In *FOCS*, pages 770–780. IEEE, 2021.

[CN10] Stephen Cook and Phuong Nguyen. *Logical foundations of proof complexity*, volume 11. Cambridge University Press Cambridge, 2010.

[Cob65] Alan Cobham. The intrinsic computational difficulty of functions. *Proc. Logic, Methodology and Philosophy of Science*, pages 24–30, 1965.

[Coo75] Stephen A. Cook. Feasibly constructive proofs and the propositional calculus (preliminary version). In *STOC*, pages 83–97. ACM, 1975.

[CRTY20] Lijie Chen, Ron D. Rothblum, Roei Tell, and Eylon Yogev. On exponential-time hypotheses, derandomization, and circuit lower bounds: Extended abstract. In *FOCS*, pages 13–23. IEEE, 2020.

[CVW18] Yilei Chen, Vinod Vaikuntanathan, and Hoeteck Wee. GGH15 beyond permutation branching programs: Proofs, attacks, and candidates. In *CRYPTO (2)*, volume 10992 of *Lecture Notes in Computer Science*, pages 577–607. Springer, 2018.

[FS11] Lance Fortnow and Rahul Santhanam. Infeasibility of instance compression and succinct pcps for NP. *J. Comput. Syst. Sci.*, 77(1):91–106, 2011.

[GGH$^+$16] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. *SIAM J. Comput.*, 45(3):882–929, 2016.

[GGM84] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. On the cryptographic applications of random functions. In *CRYPTO*, volume 196 of *Lecture Notes in Computer Science*, pages 276–288. Springer, 1984.

[GGNS23] Karthik Gajulapalli, Alexander Golovnev, Satyajeet Nagargoje, and Sidhant Saraogi. Range avoidance for constant-depth circuits: Hardness and algorithms. *CoRR*, abs/2303.05044, 2023.

[GGSW13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In *STOC*, pages 467–476. ACM, 2013.

[GLW22] Venkatesan Guruswami, Xin Lyu, and Xiuhan Wang. Range avoidance for low-depth circuits and connections to pseudorandomness. In *APPROX/RANDOM*, volume 245 of *LIPIcs*, pages 20:1–20:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[GPS16] Sanjam Garg, Omkant Pandey, and Akshayaram Srinivasan. Revisiting the cryptographic hardness of finding a nash equilibrium. In *CRYPTO (2)*, volume 9815 of *Lecture Notes in Computer Science*, pages 579–604. Springer, 2016.

[GS89] Shafi Goldwasser and Michael Sipser. Private coins versus public coins in interactive proof systems. *Adv. Comput. Res.*, 5:73–90, 1989.

[IKV18] Russell Impagliazzo, Valentine Kabanets, and Ilya Volkovich. The power of natural properties as oracles. In *Computational Complexity Conference*, volume 102 of *LIPIcs*, pages 7:1–7:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

[IW97] Russell Impagliazzo and Avi Wigderson. $P = BPP$ if E requires exponential circuits: Derandomizing the XOR lemma. In *STOC*, pages 220–229. ACM, 1997.

[Jeř04] Emil Jeřábek. Dual weak pigeonhole principle, boolean complexity, and derandomization. *Ann. Pure Appl. Log.*, 129(1-3):1–37, 2004.

[Jeř05] Emil Jeřábek. *Weak pigeonhole principle, and randomized computation*. PhD thesis, Ph. D. thesis, Faculty of Mathematics and Physics, Charles University, Prague, 2005.

[Jeř07a] Emil Jeřábek. Approximate counting in bounded arithmetic. *J. Symb. Log.*, 72(3):959–993, 2007.

[Jer07b] Emil Jerábek. On independence of variants of the weak pigeonhole principle. *J. Log. Comput.*, 17(3):587–604, 2007.

[Jeř09] Emil Jeřábek. Approximate counting by hashing in bounded arithmetic. *J. Symb. Log.*, 74(3):829–860, 2009.

[JJ22] Abhishek Jain and Zhengzhong Jin. Indistinguishability obfuscation via mathematical proofs of equivalence. In *FOCS*, pages 1023–1034. IEEE, 2022.

[JLS21] Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from well-founded assumptions. In *STOC*, pages 60–73. ACM, 2021.

[JLS22a] Aayush Jain, Huijia Lin, and Amit Sahai. Personal Communication, 2022.

[JLS22b] Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from LPN over $\mathbb{F}_p$, DLIN, and PRGs in NC^0. In *EUROCRYPT (1)*, volume 13275 of *Lecture Notes in Computer Science*, pages 670–699. Springer, 2022.

[KC00] Valentine Kabanets and Jin-yi Cai. Circuit minimization problem. In *STOC*, pages 73–79. ACM, 2000.

[Kha22] Erfan Khaniki. Nisan-wigderson generators in proof complexity: New lower bounds. In *CCC*, volume 234 of *LIPIcs*, pages 17:1–17:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[KKMP21] Robert Kleinberg, Oliver Korten, Daniel Mitropolsky, and Christos H. Papadimitriou. Total functions in the polynomial hierarchy. In *ITCS*, volume 185 of *LIPIcs*, pages 44:1–44:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[Kla20] Erica Klarreich. Computer scientists achieve 'crown jewel' of cryptography. *Quanta Magazine*, Nov 2020.

[KMN$^+$22] Ilan Komargodski, Tal Moran, Moni Naor, Rafael Pass, Alon Rosen, and Eylon Yogev. One-way functions and (im)perfect obfuscation. *SIAM J. Comput.*, 51(6):1769–1795, 2022.

[Ko86] Ker-I Ko. On the notion of infinite pseudorandom sequences. *Theor. Comput. Sci.*, 48(3):9–33, 1986.

[KO17] Jan Krajícek and Igor Carboni Oliveira. Unprovability of circuit upper bounds in cook's theory PV. *Log. Methods Comput. Sci.*, 13(1), 2017.

[Kol65] Andrei N Kolmogorov. Three approaches to the quantitative definition of information. *Problems of information transmission*, 1965.

[Kor21] Oliver Korten. The hardest explicit construction. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 433–444. IEEE, 2021.

[Kor22] Oliver Korten. Derandomization from time-space tradeoffs. In *CCC*, volume 234 of *LIPIcs*, pages 37:1–37:26. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[KPT91] Jan Krajíček, Pavel Pudlák, and Gaisi Takeuti. Bounded arithmetic and the polynomial hierarchy. *Ann. Pure Appl. Log.*, 52(1-2):143–153, 1991.

[Kra95] Jan Krajíček. *Bounded arithmetic, propositional logic, and complexity theory*, volume 60 of *Encyclopedia of mathematics and its applications*. Cambridge University Press, 1995.

[Kra11] Jan Krajíček. On the proof complexity of the nisan-wigderson generator based on a hard NP ∩ conp function. *J. Math. Log.*, 11(1), 2011.

[Kra19] Jan Krajíček. *Proof complexity*, volume 170. Cambridge University Press, 2019.

[Kra21] Jan Krajíček. Small circuits and dual weak PHP in the universal theory of p-time algorithms. *ACM Trans. Comput. Log.*, 22(2):11:1–11:4, 2021.

[Kra22] Jan Krajícek. On the existence of strong proof complexity generators. *Electron. Colloquium Comput. Complex.*, TR22-120, 2022.

[KvM02] Adam R. Klivans and Dieter van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM J. Comput.*, 31(5):1501–1526, 2002.

[LC11] Dai Tri Man Le and Stephen A. Cook. Formalizing randomized matching algorithms. *Log. Methods Comput. Sci.*, 8(3), 2011.

[MP20] Moritz Müller and Ján Pich. Feasibly constructive proofs of succinct weak circuit lower bounds. *Ann. Pure Appl. Log.*, 171(2), 2020.

[MV05] Peter Bro Miltersen and N. V. Vinodchandran. Derandomizing arthur-merlin games using hitting sets. *Comput. Complex.*, 14(3):256–279, 2005.

[NW94] Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994.

[Oja04] Kerry Ojakian. *Combinatorics in bounded arithmetic*. PhD thesis, Carnegie Mellon University, 2004.

[Pic15a] Ján Pich. Circuit lower bounds in bounded arithmetics. *Ann. Pure Appl. Log.*, 166(1):29–45, 2015.

[Pic15b] Ján Pich. Logical strength of complexity theory and a formalization of the PCP theorem in bounded arithmetic. *Log. Methods Comput. Sci.*, 11(2), 2015.

[PS21] Ján Pich and Rahul Santhanam. Strong co-nondeterministic lower bounds for NP cannot be proved feasibly. In *STOC*, pages 223–233. ACM, 2021.

[RR97] Alexander A. Razborov and Steven Rudich. Natural proofs. *J. Comput. Syst. Sci.*, 55(1):24–35, 1997.

[RSW22] Hanlin Ren, Rahul Santhanam, and Zhikun Wang. On the range avoidance problem for circuits. In *FOCS*, pages 640–650. IEEE, 2022.

[Sip83] Michael Sipser. A complexity theoretic approach to randomness. In *STOC*, pages 330–335. ACM, 1983.

[Sip97] Michael Sipser. *Introduction to the theory of computation*. PWS Publishing Company, 1997.

[STV01] Madhu Sudan, Luca Trevisan, and Salil P. Vadhan. Pseudorandom generators without the XOR lemma. *J. Comput. Syst. Sci.*, 62(2):236–266, 2001.

[SW21] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: Deniable encryption, and more. *SIAM J. Comput.*, 50(3):857–908, 2021.

[Tsa22] Rotem Tsabary. Candidate witness encryption from lattice techniques. In *CRYPTO (1)*, volume 13507 of *Lecture Notes in Computer Science*, pages 535–559. Springer, 2022.

[VWW22] Vinod Vaikuntanathan, Hoeteck Wee, and Daniel Wichs. Witness encryption and null-io from evasive LWE. In *ASIACRYPT (1)*, volume 13791 of *Lecture Notes in Computer Science*, pages 195–221. Springer, 2022.

A Hitting Set for Range Avoidance

We say that a Hitting Set for AVOID on circuits of size s with m outputs and n inputs ($m > n$) is a set $S_{m,s}$ of m-bit strings such that every circuit of size s with m outputs fails to output at least one string in $S_{m,s}$.

Here, we note how (under standard derandomization assumptions) one can construct a Hitting Set for AVOID on circuits of size s with $m > n$ outputs, in deterministic $\mathsf{poly}(s, m)$ time. We stress that our methods here are standard, and we are simply recording this fact for convenience of the reader.

Theorem 32. *Assume that there is an $\varepsilon > 0$ such that some function in $\mathsf{E} = \mathsf{TIME}[2^{O(n)}]$ does not have SAT-oracle circuits of size $2^{\varepsilon n}$, for almost every input length n. Then there is an algorithm A that takes 1^s and 1^m as input, runs in $\mathsf{poly}(s, m)$ time, and outputs a set $S_{m,s}$ of m-bit strings, such that every circuit of size s fails to output at least one string in $S_{m,s}$.*

Proof. Recall that the well-known connections from circuit complexity lower bounds to pseudorandom generators relativize [KvM02]. Thus our hypothesis implies that there is a polynomial-time algorithm generating a *discrepancy set* for SAT-oracle circuits. Namely, there is a polynomial-time algorithm B such that, on inputs $1^{n'}$ and $1^{s'}$, for sufficiently large n' and s', B outputs a set of $\mathsf{poly}(n', s')$ strings $T_{n',s'}$ of length n' such that, for every SAT-oracle circuit C with n' inputs and s' size,

$$\left| \Pr_{x \sim U_{n'}} [C(x) = 1] - \Pr_{x \sim T_{n',s'}} [C(x) = 1] \right| < 1/10,$$

where "$x \sim T_{n',s'}$" indicates that x is chosen uniformly at random from $T_{n',s'}$. In other words, $T_{n',s'}$ "fools" every such SAT-oracle circuit C.

We now construct a specific SAT-oracle circuit C, and argue that fooling C allows us to construct our desired hitting set from $T_{n',s'}$.

First of all, we observe that for $m > n$, a standard union bound argument implies that a uniform random subset of m-bit strings of cardinality $2s^2$ is a hitting set for size-s circuits, in that every circuit of size s fails to output at least one string in $S_{m,s}$ with probability $1 - o(1)$. (There are only $s^{O(s)}$ such circuits, and for each such circuit, a random m-bit string is in its range with probability at most $1/2$. Thus, choosing $2s^2$ such strings will "hit" all s-size circuits with probability $1 - o(1)$.)

Let m be the desired output-length parameter and s be the desired size parameter for our hitting set $S_{m,s}$. Our SAT-oracle circuit C will be designed to output 1 if and only if its input encodes a hitting set. In particular, C takes $2s^2 \cdot m$ inputs, treats the inputs as a sequence of strings $y_1, \ldots, y_{2s^2} \in \{0,1\}^m$, and uses its SAT oracle to check if there exists a circuit C' of size s, an input x' of length less than m, and an $i = 1, \ldots, 2s^2$ such that $C'(x') = y_i$. If so, our circuit C outputs 0, otherwise C outputs 1. Moreover, observe that C can itself be implemented in at most $k(sm)^k$ size for some constant k, using SAT oracle gates.

Since a random choice of m-bit strings is a hitting set with high probability, we have

$$\Pr_{x \sim U_n} [C(x) = 1] = 1 - o(1).$$

Therefore, if we run our polynomial-time algorithm B on input $1^{n'}$ and $1^{s'}$, with $n' = 2s^2 m$ and $s' = k(sm)^k$, B outputs a set of $\mathsf{poly}(s, m)$ strings $T_{n',s'}$ of length $n' = 2s^2 m$ such that

$$\Pr_{x \sim T_{n',s'}} [C(x) = 1] \geq 9/10 - o(1).$$

Therefore, breaking every n'-bit string in $T_{n',s'}$ into $2s^2$ strings of length m appropriately, the union of all such m-bit strings is a hitting set, as desired. □

B A Barrier Against Black-Box Reductions

In this section, we show a barrier result to proving that AVOID is intractable via standard black-box reductions.

We say there is a *black-box randomized polynomial-time reduction* from a problem A to a problem B if there is a probabilistic polynomial-time oracle Turing machine R such that for any oracle $\mathcal{O}$ that solves B we have that $R^{\mathcal{O}}$ solves A.

We show that any black-box randomized polynomial-time reduction from a problem A to AVOID with sufficiently large stretch in the number of outputs actually implies a randomized polynomial-time algorithm for A. This rules out basing the intractability of deterministic algorithms for AVOID on, say, lower bounds against SAT (or any problem outside of BPP) via black-box randomized reductions.

Theorem 33. *If there is a black-box randomized reduction from a problem A to* AVOID *on circuits with n-inputs and n^2 outputs, then A can be solved in randomized polynomial-time.*

Proof (Sketch). Let R be the probabilistic polynomial-time oracle Turing machine R guaranteed by the black-box reduction. Set $k > 1$ to be a constant such that R runs in $o(n^k)$ time and thus makes at most $o(n^k)$ oracle queries.

We claim the that the following is a randomized polynomial-time algorithm for A. Given an instance x, simulate running $R^{\mathcal{O}}(x)$ and output the answer. Whenever a query (i.e. a circuit C mapping t bits to t^2-bits) is asked to the oracle $\mathcal{O}$, respond as follows:

1. If this query has been asked before, answer with the previous answer.
2. If $t \leq k \log n$, then brute force through all the strings in the range of C, and output the lexicographically first string not in the range of C.
3. If $t > k \log n$, pick a uniformly random string of length t^2 and respond to the query with that.

It is easy to see that this is a polynomial-time randomized algorithm. We now show that it computes the answer to A with high probability. Because R is a black box reduction, observe that as long as we always answer queries to $\mathcal{O}$ with valid answers to AVOID, that $R^{\mathcal{O}}(x)$ must correctly solve A on x. Queries answered when $t \leq k \log n$ will be correct by construction. Each query answered when $t > k \log n$ will be incorrect with probability at most

$$2^{-t^2+t} < 2^{-k^2 \log^2 n + k \log n} \ll \frac{1}{\Omega(n^{-k})}.$$

Union bounding over $o(n^k)$ queries made by R, the probability that all oracle queries receive correct answers is at least $1 - o(1)$, as desired. □

C An Algorithm for AVOID with a Circuit-Inversion Oracle

In this section, we give a polynomial-time reduction from AVOID to the problem of constructing a truth table of high circuit complexity. (For the latter problem, having a polynomial-time algorithm is equivalent to establishing circuit lower bounds for E.) The reduction is essentially from [Kor21] (although Korten's formulation of the result is slightly weaker), which is inspired by earlier provability results in bounded arithmetic [Jeř04] and the Goldreich-Goldwasser-Micali construction of pseudorandom functions [GGM84].

Let $\varepsilon \in (0, 1)$ be a constant.

Search Problem: HARD$_\varepsilon$
Input: 1^N such that $N = 2^n$ is a power of two.
Output: The truth table of a function $f : \{0,1\}^n \to \{0,1\}$ that requires Boolean circuits of size $2^{\varepsilon n}$.

Lemma 34. *There is a polynomial-time algorithm for* AVOID *on circuits $C : \{0,1\}^n \to \{0,1\}^{n+1}$ that has access to an oracle that inverts the input circuit, and makes one oracle call to* AVOID *on a circuit $D : \{0,1\}^n \to \{0,1\}^{2n}$.*

Proof. Let $C_1 := C$ be the input circuit. For $i \in \{2, 3, \ldots, n\}$, we construct the circuit $C_i : \{0,1\}^n \to \{0,1\}^{n+i}$:

$$C_i(x) := (C(C_{i-1}(x)|_{[n]}), C_{i-1}(x)|_{\setminus[n]}),$$

where $C_{i-1}(x)|_{[n]}$ denotes the first n bits of $C_{i-1}(x)$ and $C_{i-1}(x)|_{\setminus[n]}$ denotes the last $i-1$ bits. We call our AVOID oracle once on $C_n : \{0,1\}^n \to \{0,1\}^{2n}$ to obtain a string $y_n \in \{0,1\}^{2n}$ outside of the range of C_n, and then make at most n oracle queries to the circuit-inversion oracle, as follows.

For every $i = n-1, n-2, \ldots, 1$, we will find a $y_i \in \{0,1\}^{n+i}$ that is outside of the range of C_i. Assume that we have found one such $y_{i+1} = (y'_{i+1}, y''_{i+1})$, where $y'_{i+1} \in \{0,1\}^{n+1}$. We call the inversion oracle on y'_{i+1} to obtain an $x \in \{0,1\}^n$ such that $C(x) = y'_{i+1}$. If the oracle fails to invert y'_{i+1}, we obtain a non-output $y'_{i+1} \in \{0,1\}^{n+1}$ of the input circuit C; otherwise, we define $y_i := (x, y''_{i+1})$. In the latter case, we know by the definition of C_{i+1} that y_i is outside of the range of C_i. At the end, we obtain a string $y_1 \in \{0,1\}^{n+1}$ outside of the range of $C_1 = C$. □

Theorem 35. *There is a polynomial-time algorithm for* AVOID *on circuits with n inputs and $m > n$ outputs that has access to an oracle that inverts the input circuit, and makes one call to an oracle for* HARD$_\varepsilon$.

Proof. Given a circuit C with n inputs and $m > n$ outputs, we can start by padding C with dummy inputs if necessary so that $m = n + 1$. Furthermore, by applying Lemma 34, we may assume that C has n inputs and $2n$ outputs (assume it is the oracle call in Lemma 34).

Let k be a parameter to be determined later, and let $C_0 : \{0,1\}^n \to \{0,1\}$ be the first output bit of $C(x)$. For simplicity, define C_{lft} and C_{rgt} to be the first n bits and the last n bits of the $2n$-bit output of C, respectively (i.e., $C(x) = (C_{\mathsf{lft}}(x), C_{\mathsf{rgt}}(x))$). For $i \in \{1, 2, \ldots, k\}$, we construct a circuit $C_k : \{0,1\}^n \to \{0,1\}^{2^k}$ defined as

$$C_k(x) := (C_{k-1}(C_{\mathsf{lft}}(x)), C_{k-1}(C_{\mathsf{rgt}}(x))).$$

Claim 36. *For every $x \in \{0,1\}^n$, there is a circuit $D : \{0,1\}^k \to \{0,1\}$ of size $O(k(n + |C|))$ such that the truth table of D is $C_k(x) \in \{0,1\}^{2^k}$.*

Proof. The circuit D is essentially the evaluation function of the GGM construction of PRF [GGM84]. (Also see Figure 2 in the proof of Theorem 7 in [Kor21].) □

We choose $k = 10 \cdot \varepsilon^{-1} \cdot \log(n + |C|) = O(\log |C|)$, so that the circuit size of D in Claim 36 is smaller than $2^{\varepsilon k}$ for sufficiently large n. By calling the HARD$_\varepsilon$ oracle on the input 1^{2^k}, we will obtain a $y_k \in \{0,1\}^{2^k}$ that requires $2^{\varepsilon k}$ size circuits, which, by Claim 36, is outside of the range of C_k.

We define the following recursive algorithm with inputs $(i, y) \in \{0, 1, \ldots, k\} \times \{0,1\}^i$ that attempts to invert the circuit C_i on the string y with the help of the circuit-inversion oracle:

- If $i = 0$, we know that C_i outputs the first bit of C and $y \in \{0,1\}$. We call the inversion oracle on $(y, 0^{2n-1}) \in \{0,1\}^{2n}$.
- If $i \geq 1$, denote $y = (y', y'')$, where $y', y'' \in \{0,1\}^{2^{i-1}}$. We recursively try to invert the circuit C_{i-1} on the inputs y' and y''. If we successfully obtain $x', x'' \in \{0,1\}^n$ such that $C_{i-1}(x') = y'$ and $C_{i-1}(x'') = y''$, we call the circuit-inversion oracle to invert C on the input $(x', x'') \in \{0,1\}^{2n}$ and output the answer $\hat{x}$, which, by the definition of C_k, satisfies that $C_k(\hat{x}) = y$.

Our recursive algorithm runs in time $2^{O(k)} \cdot \mathsf{poly}(|C|) = \mathsf{poly}(|C|)$ and makes at most $2^{O(k)} = \mathsf{poly}(|C|)$ queries to the circuit-inversion oracle. We run this recursive algorithm on the input (k, y_k). Since y_k is outside of the range of C_k, this recursive algorithm will necessarily fail on an oracle query; in other words, it asks the oracle to invert a string that is outside of the range of the input circuit C. We can then output this string. □

In particular, assuming that E is not computable infinitely often by circuits of size $2^{\varepsilon n}$, there is a polynomial-time algorithm that on 1^{2^n} outputs (for almost all n) a solution to HARD$_\varepsilon$. Therefore, by Theorem 35, assuming the circuit lower bound for E, we solve AVOID in polynomial-time with (polynomially many) queries to an oracle that inverts the input circuit.

D The Power of UAPC_1

In this section, we demonstrate the power of the theory UAPC_1 by showing that Jeřábek's main theorem of approximate counting for APC_1 [Jeř07a] can be proved for UAPC_1. This was observed by Pich and Santhanam [PS21, Section 2]. For simplicity, we only present Jeřábek's theorem, outline the proof in [Jeř07a], and verify that all applications of the dual weak pigeonhole principle can be replaced by its uniform variant.

We adopt set-theoretic notation, where a natural number a is identified with the set $\{0, 1, \ldots, a-1\}$. Recall that $n \in \mathsf{Log}$ is the shorthand of $\exists V\ n = |V|$. We say $\varepsilon^{-1} \in \mathsf{Log}$ if there is an $n \in \mathsf{Log}$ such that $\varepsilon^{-1} \le n$.

A *bounded definable set* is a set of numbers of the form $X = \{x < a \mid \varphi(x)\}$, where a is a number and φ is a formula. In particular, given n and a Boolean circuit $C : \{0,1\}^n \to \{0,1\}$ (where strings are encoded as numbers, say $C : 2^n \to 2$), the bounded set defined by C is $X_C := \{x < 2^n \mid C(x) = 1\}$. As bounded theories (e.g. PV_1) cannot represent sets, the notion of bounded definable sets $X = \{x < a \mid \varphi(x)\}$ is only defined in the meta-theory, and $x \in X$ is a shorthand of the first-order formula $x < a \wedge \varphi(x)$.

For bounded definable sets $X \subseteq a$ and $Y \subseteq b$, we define $X \times Y := \{bx + y \mid x \in X, y \in Y\} \subseteq ab$ and $X \dot\cup Y := X \cup \{y + a \mid y \in Y\} \subseteq a + b$. In particular, $x \times y = xy$ and $x \dot\cup y = x + y$, if we consider $x \subseteq a, y \subseteq b$ as bounded sets.

For a function $f : a \to b$ and a bounded definable set $X \subseteq a$, we use $f[X]$ to denote the range of f over the domain X, i.e., the bounded set $\{y < b \mid \exists x \in X, y = f(x)\}$. We use id_X to denote the identity function over X.

Main theorem for approximate counting. Let a be a number and $X \subseteq a$ be a bounded definable set. The goal of Jeřábek's approximate counting is to estimate $|X|/a$ up to a small additive error. Since the brute-force definition of the size of a set $X \subseteq a$ requires $\mathsf{poly}(a) = \exp(O(|a|))$ time to enumerate $\{0, 1, \ldots, a-1\}$, Jeřábek [Jeř07a] defines the approximate size of bounded definable sets as follows.

Let $C : 2^n \to 2^m$ be a Boolean circuit and $X \subseteq 2^n$, $Y \subseteq 2^m$ be definable sets. We say C computes a function from X to Y, denoted by $C : X \to Y$, if $C[X] \subseteq Y$. In addition, we use $C : X \hookrightarrow Y$ to denote that C is injective. We introduce the notation $C : X \twoheadrightarrow Y$ to denote that $Y \subseteq C[X]$. Note that $C : X \twoheadrightarrow Y$ does not necessarily imply $C : X \to Y$. As above, the notation is defined in meta-theory as the shorthands of corresponding first-order sentences in PV_1. For instance, let $X = \{x < a \mid \varphi(x)\}$ and $Y = \{y < b \mid \psi(y)\}$, then $C : X \to Y$ is shorthand for $\forall x, x < a \wedge \varphi(x) \to C(x) < b \wedge \psi(C(x))$.

Now we define (approximate) size comparison of definable sets. Let $X, Y \subseteq 2^n$ be definable sets and $\varepsilon \le 1$. We say that X is approximately smaller than Y with error ε, denoted by $X \preceq_\varepsilon Y$, if for some Boolean circuit G and $v \neq 0$, $G : v \times (Y \dot\cup \varepsilon 2^n) \twoheadrightarrow v \times X$. The number v is introduced due to technical reasons. Intuitively, $X \preceq_\varepsilon Y$ means that there is an (efficiently computable) surjection from $Y \dot\cup \varepsilon 2^n$ to X, which is the natural way to define $|X| \le |Y| + \varepsilon 2^n$ in bounded theories. We say that X and Y are of the same size up to an error ε, denoted by $X \approx_\varepsilon Y$, if $X \preceq_\varepsilon Y$ and $Y \preceq_\varepsilon X$. In particular, we say that X is of size s up to an error ε if $X \approx_\varepsilon s$, which intuitively stands for $s - \varepsilon 2^n \le |X| \le s + \varepsilon 2^n$.

The following theorem suggests that for every bounded set $X \subseteq 2^n$ defined by circuits, there is an $s \le 2^n$ such that X is of size s up to an error ε provable in UAPC_1. Intuitively, the theorem shows that under UAPC_1, for every circuit $C : \{0,1\}^n \to \{0,1\}$, let $X = \{x \in \{0,1\}^n \mid C(x) = 1\}$ be the set of inputs accepted by C, there is an efficiently computable surjection from $X \dot\cup \varepsilon 2^n$ to s and an efficiently computable surjection from $s + \varepsilon 2^n$ to X, which formalizes $|X| \in [s - \varepsilon 2^n, s + \varepsilon 2^n]$ (i.e. $s/2^n$ approximates the acceptance probability of C up to an additive error ε).

Theorem 37 (Main theorem for approximate counting)**.** *The following is provable in* UAPC_1*. For every bounded set* $X = \{x \le 2^n \mid C(x) = 1\}$ *defined by a Boolean circuit* C *and* $\varepsilon^{-1} \in \mathsf{Log}$*, there exist an* $s \le 2^n$ *such that* $X \approx_\varepsilon s$*. Moreover, for some* $v \le \mathsf{poly}(n\varepsilon^{-1}|C|)$ *and circuits* $G_\xi, H_\xi, \xi \in \{0,1\}$ *of size* $\mathsf{poly}(n\varepsilon^{-1}|C|)$*,*

$$\begin{aligned} &G_0 : v(s + \varepsilon 2^n) \twoheadrightarrow v \times X \quad && H_0 : v \times X \hookrightarrow v(s + \varepsilon 2^n) \quad && G_0 \circ H_0 = \mathsf{id}_{v \times X}; \\ &G_1 : v \times (X \dot\cup \varepsilon 2^n) \twoheadrightarrow vs \quad && H_1 : vs \hookrightarrow v \times (X \dot\cup \varepsilon 2^n) \quad && G_1 \circ H_1 = \mathsf{id}_{vs}. \end{aligned}$$

We first outline the proof of this theorem for APC_1 in [Jeř07a] and then check that the applications of $\mathsf{dWPHP}(\mathsf{PV})$ in both steps can be replaced by $\mathsf{dWPHP}'(\mathsf{PV})$. The proof consists of two steps.

1. From $\mathsf{dWPHP}(\mathsf{PV})$, we can show that there is an average-case hard truth table. That is for $k \in \mathsf{LogLog}$, there is a function $f : 2^k \to 2$ (represented by its truth table) such that for some small $\varepsilon \in (0,1)$ and every circuit $D : 2^k \to 2$ of size $2^{\varepsilon k}$, $|\{x < 2^n \mid f(x) = D(x)\}| \le (1/2 + 2^{-\varepsilon k})2^k$. Note that the size of the set is defined by a brute-force counting algorithm, as $k \in \mathsf{LogLog}$ implies that it is feasible to enumerate all strings of length k and count the number of strings x such that $D(x) = f(x)$.
2. We adopt the correctness proof of the Nisan-Wigderson pseudorandom generator [NW94] to approximate the size of X, which is the acceptance probability of the circuit C that defines it, by the acceptance probability of C on the pseudorandom generator. More precisely, let $\mathsf{NW}_f : 2^t \to 2^n$ be the PRG for some $t \in \mathsf{LogLog}$ and $Y := \{y < 2^t \mid C(\mathsf{NW}_f(y)) = 1\}$, we will show that $X \times 2^t \approx_\varepsilon 2^n \times Y$. Since the seed length is $t \in \mathsf{LogLog}$, the size s' of Y can be feasibly computed, we can show that $X \approx_\varepsilon s$, where $s := 2^{n-t}s'$.

We will show that the first step is provable in $\mathsf{UAPC}_1 = \mathsf{PV}_1 + \mathsf{dWPHP}'(\mathsf{PV})$, and the second step is provable in PV_1 given the result from the first step.

Step 1: Hard truth table. The proof of the existence of a hard truth table is similar to the reduction from HARD to AVOID. Let TT be the following algorithm: given a circuit $D : 2^k \to 2$ of size $2^{\varepsilon k}$ and a (succinctly encoded) string $w \in 2^m$ such that $|w| \le (1/2 + m^{-\varepsilon})m$, TT outputs $\mathsf{tt}(D) \oplus w$, where $m = 2^k$ and $\mathsf{tt}(D)$ is the truth table of D. It can be easily seen that the output length of TT is sufficiently larger than its input length for a small constant $\varepsilon > 0$. It is provable in PV_1 by the definition of the hard truth tables that any string outside of the range of TT is a desired hard truth table. Moreover, TT is a uniform PV function without any additional parameters. Therefore the existence of a hard truth table can be proved by $\mathsf{PV}_1 + \mathsf{dWPHP}'(\mathsf{TT}) \subseteq \mathsf{UAPC}_1$.

Step 2: Nisan-Wigderson PRG. With a closer inspection of the proof in [Jeř07a], we can see that it does not use the power of $\mathsf{dWPHP}(\mathsf{PV})$, i.e., the proof can be formalized in PV_1 given the hard truth table constructed in Step 1. For simplicity, we only sketch the proof and refer the readers to the original paper [Jeř07a] for more details.

The main idea is to formalize the standard correctness proof of Nisan-Wigderson PRG via the hybrid argument (see, e.g., [AB09]) in PV_1. Let $f : 2^k \to 2$ be an (average-case) hard function and $\mathsf{NW}_f : 2^t \to 2^n$ be the Nisan-Wigderson PRG. That is, for an (explicit) combinatorial design $S = (S_1, \dots, S_n)$ satisfying

1. $S_i \subseteq [t]$ for every $i \in [n]$,
2. $|S_i| = k$ for every $i \in [n]$, and
3. $|S_i \cap S_j|$ is "small" for every $i \ne j$,

the i-th bit of $\mathsf{NW}_f(x)$ is $f(x|_{S_i})$, where $x|_{S_i} := (x_{j_1}, x_{j_2}, \dots, x_{j_k}) \in 2^k$, $S_i = \{j_1, j_2, \dots, j_k\}$, and $j_1 < j_2 < \dots < j_k$. Let $X := \{x < 2^n \mid C(x) = 1\}$ and $Y := \{y < 2^t \mid C(\mathsf{NW}_f(y)) = 1\}$. Recall that we want to prove that $X \times 2^t \approx_\varepsilon 2^n \times Y$.

We define hybrids $M_0, M_1, \dots, M_n$ as follows. For every $i \in \{0, 1, \dots, n\}$, we define

$$M_i := \{(x,y) \in 2^n \times 2^t \mid C(\mathsf{NW}_f(y)_1, \mathsf{NW}_f(y)_2, \dots, \mathsf{NW}_f(y)_i, x_{i+1}, x_{i+2}, \dots, x_n) = 1\} \subseteq 2^n \times 2^t.$$

Notice that $M_0 = X \times 2^t$ and $M_n = 2^n \times Y$. To prove that $M_0 \approx_\varepsilon M_n$, it is sufficient to show that for every $i \in \{0, 1, \dots, n-1\}$, $M_i \approx_{\varepsilon'} M_{i+1}$ for $\varepsilon' := \varepsilon/n$. This can be shown by proving $M_0 \approx_{\varepsilon' \cdot i} M_i$ with an induction on i. However, since $M_0 \approx_{\varepsilon' \cdot i} M_i$ is not defined by a quantifier-free formula, one needs to slightly strengthen the induction hypothesis to have explicit $G_i^1, G_i^2, H_i^1, H_i^2, v_i^1, v_i^2$ that witness $M_0 \approx_{\varepsilon' \cdot i} M_i$. That is,

$$G_i^1 : v_i^1 \times (M_0 \mathbin{\dot\cup} \varepsilon' i 2^n) \twoheadrightarrow v_i^1 \times M_i \qquad H_i^1 : v_i^1 \times M_i \hookrightarrow v_i^1 \times (M_0 \mathbin{\dot\cup} \varepsilon' i 2^n) \qquad G_i^1 \circ H_i^1 = \mathsf{id}_{v_i^1 \times M_i}, \tag{10}$$

$$G_i^2 : v_i^2 \times (M_i \mathbin{\dot\cup} \varepsilon' i 2^n) \twoheadrightarrow v_i^2 \times M_0 \qquad H_i^2 : v_i^2 \times M_0 \hookrightarrow v_i^2 \times (M_i \mathbin{\dot\cup} \varepsilon' i 2^n) \qquad G_i^2 \circ H_i^2 = \mathsf{id}_{v_i^2 \times M_0}. \tag{11}$$

In such case, we only need the induction principle for quantifier-free formulas and thus it is provable in PV_1.

It remains to show that $M_i \approx_{\varepsilon'} M_{i+1}$ (with explicit witnesses in the sense of Equations (10) and (11)). Fix this i. Let $h : 2^t \to 2^{t-k} \times 2^k$ be the bijection that maps $y \in 2^t$ to $(y|_{S_{i+1}}, y|_{[t] \setminus S_{i+1}})$, where both h and h^{-1} are computable by PV-functions. Let $f_j^w(u) := f(h^{-1}(u,w)|_{S_j})$. By the definition, we know that $f_i^w(u) = f(u)$, and $f_j^w(u)$ for

$j \neq i$ depends on a small fraction of bits of u, since $|S_i \cap S_j|$ is small. We define

$$\begin{aligned} M'_i &:= \{(x_{i+2},\dots,x_n,r,w,u) \mid C(f_1^w(u),\dots,f_i^w(u),r,x_{i+2},\dots,x_n)=1\} \subseteq 2^{n-i-1}\times 2\times 2^{t-k}\times 2^k, \\ M'_{i+1} &:= \{(x_{i+2},\dots,x_n,r,w,u) \mid C(f_1^w(u),\dots,f_i^w(u),f(u),x_{i+2},\dots,x_n)=1\} \subseteq 2^{n-i-1}\times 2\times 2^{t-k}\times 2^k. \end{aligned}$$

One can notice that there are PV-functions $g_1 : M_i \to M'_i \times 2^i$ and $g_2 : M_{i+1} \to M'_{i+1} \times 2^i$ that are bijections, therefore we only need to show that $M'_i \approx_{\varepsilon'} M'_{i+1}$.

Fix $w < 2^{t-k}$ and $x_{i+2},\dots,x_n < 2$. We define the sets

$$\begin{aligned} U^{w,\vec{x}} &:= \{(r,u) \in 2\times 2^k \mid C(f_1^w(u),\dots,f_i^w(u),r,x_{i+2},\dots,x_n)=1\} \subseteq 2\times 2^k, \\ V^{w,\vec{x}} &:= \{(r,u) \in 2\times 2^k \mid C(f_1^w(u),\dots,f_i^w(u),f(u),x_{i+2},\dots,x_n)=1\} \subseteq 2\times 2^k. \end{aligned}$$

We will choose $k \in \mathsf{LogLog}$ so that it is possible to precisely count the size of $U^{w,\vec{x}}$ and $V^{w,\vec{x}}$ in brute-force. Since f is hard on average against circuits, we can prove that $|U^{w,\vec{x}}| - |V^{w,\vec{x}}|$ is small. Since the functions that witness $U^{w,\vec{x}} \approx_{\varepsilon'} V^{w,\vec{x}}$ is uniform in w and $\vec{x}$, we can further extend it to witness that $M'_i \approx_{\varepsilon'} M'_{i+1}$, which completes the proof.

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il