Electronic Colloquium on Computational Complexity, Report No. 45 (2025)

Student-Teacher Constructive Separations and (Un)Provability in Bounded Arithmetic: Witnessing the Gap

Marco Carmosino, Stefan Grosser

April 11, 2025

Abstract

Let $\mathcal{C}$ be a complexity class and A be a language. The statement "$A \notin \mathcal{C}$" is a *separation* of A from $\mathcal{C}$. A separation is *constructive* if there is an efficient algorithm called a *refuter* that prints counterexamples to the statement "M decides A" for every $\mathcal{C}$-algorithm M. Concretely, refuters witness errors of M on A by printing, on input 1^n, an n-bit string x such that $M(x) \neq A(x)$. Many recent breakthroughs in lower bounds and derandomization, like the algorithmic method [12], rely on constructive separations as a core component. Chen, Jin, Santhanam, and Williams [14] studied the consequences of constructivizing classical non-constructive lower bounds in complexity theory. They showed that (1) constructivizing many known separations would imply breakthrough lower bounds, and (2) some separations are impossible to constructivize.

We study a more general notion of "efficient refutation" in terms of $\mathcal{C}$-*Student-Teacher Games*, where the $\mathcal{C}$-refuter (Student) is allowed to adaptively propose candidate counterexamples x_i to an omniscient Teacher. If x_i fails to witness an error, Teacher reveals a counterexample y_i to the statement "x_i is a counterexample to the statement 'M decides A' " — the nature of y_i depending on how the separated language A and complexity class $\mathcal{C}$ are defined. We show:

- If there is a P-Student-Teacher constructive separation of Palindromes from one-tape nondeterministic $O(n^{1+\varepsilon})$ time [39], then $\mathsf{NP} \not\subset \mathsf{SIZE}[n^k]$ for every k.
- If there is a uniform $\mathsf{AC}^0[\mathsf{qpoly}]$-Student-Teacher protocol generating truth tables of super fixed polynomial circuit complexity, then $\mathsf{P} \neq \mathsf{NP}$.
- There is no P-Student-Teacher protocol which for infinitely many $c > 0$, generates high-K^{n^c} strings.

Our results imply a conditional separation of Jeřábek's theory VAPC from V^1, a theory equivalent to Buss's theory S^1_2. This improves and significantly simplifies the work of Ilango, Li, and Williams [25], who separate VAPC from the weaker theory VPV under the existence of indistinguishability obfuscation. We do not use cryptographic assumptions in our separation. Instead we introduce a natural and plausible conjecture on the uniformity of proofs in bounded arithmetic, inspired by Kreisel's Conjecture in logic. We believe this conjecture to be of independent interest.

ISSN 1433-8092

1 Introduction

Constructive lower bounds are a key concern of complexity theory. We know that hard functions *exist*, but not how to *exhibit* them efficiently. There are two ways to formalize the notion of constructivity. The *algorithmic* perspective asks for the computational complexity of searching for witnesses to a complexity lower bound, like hard truth tables. The *proof-theoretic* perspective asks for the weakest logical theory that proves complexity lower bounds. This paper obtains new results about both formulations of constructivity and the relationship between them.

In particular, we study the computational model of *Student-Teacher Games*, which links the proof-theoretic and algorithmic perspectives on constructivity. Roughly, if a complexity lower bound LB is provable in a "bounded" logical theory, then "efficient" Student-Teacher games witnessing LB follow. We obtain new results about Student-Teacher games witnessing: lower bounds for deciding Palindromes, existence of time-bounded Kolmogorov-random strings, and existence of Boolean functions of high circuit complexity. From these results, we conditionally derive (1) consequences about the provability of these statements and (2) separations of expressive and well-studied logical theories. We structure our introduction as follows:

(i) Building on prior work studying the consequences of algorithmic constructivity in complexity theory [14, 28, 50], we show that efficient Student-Teacher games witnessing known complexity lower bounds imply breakthrough lower bounds.

(ii) We scrutinize the relationship between algorithmic constructivity and proof-theoretic constructivity. The natural translation into bounded arithmetic of complexity-theoretic statements that mention "polynomially bounded resources" results in a *schema* of logical sentences, one for each *fixed* polynomial. This disrupts the well-known connection between provability of lower bounds and witnessing Student-Teacher games.

(iii) We identify a new family of conjectures called *Witnessing Hypotheses for Uniform Proofs* which give Student-Teacher witnessing from a *schema* of lower bounds. We demonstrate that these conjectures are well-founded, and connect them to the famous Kreisel Conjecture in mathematical logic.

(iv) As a consequence of these conjectures, we give the first known conditional separation between the well-studied bounded arithmetic theories VAPC and V^1 (equivalently APC_1 and S^1_2). Moreover, we do so *without* cryptographic assumptions. This constitutes substantial progress towards understanding the necessary tools needed to show unprovability in bounded arithmetic.

In the remainder of this introduction we give context, motivation, a more detailed description of our results, and a list of open problems about constructive complexity theory.

1.1 Algorithmic Constructivity in Complexity Theory

Underpinning many recent developments in complexity theory is the notion of *constructive* lower bounds. Namely, algorithms for solving refutation problems and avoidance search problems.

1. *Refutation* — If a lower bound holds for a problem Π against a model of computation M, then the Π-*Refutation for* M problem is: given an algorithm A from M and a number n, print a string of length n for which A fails to solve Π correctly; i.e. a counterexample to the claim that "A solves Π."

2. *Avoidance* — Fix $\Lambda = \bigcup_{n\in\mathbb{N}} \Lambda_n$ an infinite set of compressible strings, where each Λ_n denotes the n-bit strings described by a particular set of bounded-complexity devices, such as Boolean circuits of size at most $\log(n)^2$. The Λ-*Avoid problem* is then: given a number n, print an n-bit string outside Λ_n. That is, print a counterexample to the claim "every n-bit string is compressed by a Λ-device."

Refutation has been an explicit object of study since Kabanets [27] introduced refuters to give an unconditional weak derandomization of RP. Since then, upper bounds on refuters have been a driving force behind derandomization and lower bounds. A seminal example is the algorithmic method of Williams to give lower bounds against ACC^0. These lower bounds [52, 12] crucially use a refuter for the NTIME hierarchy theorem, with [12] in particular using an almost-everywhere refuter of Fortnow and Santhanam [20] against

$\mathsf{NTIMEGUESS}[T(n), O(n)]$, for time-constructive $T(n)$. Refutation has also been recently shown by Chen, Tell, and Williams [13] to unify many previous techniques that give conditional derandomization results.

Avoidance search problems are also intimately tied with circuit lower bounds and derandomization. Korten [28] showed that many important explicit construction problems (e.g. computing hard truth tables, rigid matrices, and high time-bounded Kolmogorov complexity strings) are all reducable to the avoidance problem Empty.

Empty: Given circuit $C\colon \{0,1\}^m \to \{0,1\}^n$, with $n > m$ output a string $x \in \{0,1\}^n$ outside the range of C.

This shows that circuit lower bounds and derandomization are implied by fast algorithms for Empty. Ren, Santhanam, and Wang [50] deepened this connection by studying algorithms for $\mathcal{C}$-Avoid, parametrized by a circuit class $\mathcal{C}$. Finally, Chen, Hirahara, and Ren [10], and a follow-up by Li [36], used Korten's reduction from finding hard truth tables to Empty in order to give truly exponential circuit lower bounds for $\mathsf{S_2E}$.

Constructive Separations. To formally study the power and limitations of constructive lower bounds, Chen, Jin, Santhanam, and Williams [14] asked what happens if you can convert several classical *non-constructive* lower bounds into constructive ones? Their definition of constructivity goes through efficient Refutation algorithms, and implicitly Avoidance algorithms.[1]

Definition 1.1 ($\mathcal{C}$-Refuter)**.** Let $f\colon \{0,1\}^* \to \{0,1\}$ be a function and let A be an algorithm. The refutation search problem $\mathrm{Ref}_{f,A} \coloneqq \{(n,x) \mid x \in \{0,1\}^n \text{ and } f(x) \neq A(x)\}$ asks to find an input x where A disagrees with function f. An algorithm $R(1^n)$ is a $\mathcal{C}$-refuter against A if $R \in \mathcal{C}$ and for infinitely many n, $(n, R(1^n)) \in \mathrm{Ref}_{R,A}$.

Definition 1.2 ($\mathcal{C}$-Constructive Separation)**.** For complexity classes $\mathcal{A}, \mathcal{B}, \mathcal{C}$, a separation $\mathcal{B} \not\subset \mathcal{A}$ is called $\mathcal{C}$-constructive if for some language $\mathcal{L}_B$ decidable in $\mathcal{B}$ and any proposed algorithm $A \in \mathcal{A}$ that decides $\mathcal{L}_B$, there is a $\mathcal{C}$-refuter R_A.

Chen et al. [14] gave several insights on constructive separations. First, they showed that a P-constructive separation for the classic Palindromes lower bound of Maass [39] implies a major complexity separation.

Theorem 1.3 (Theorem 3.4, [14])**.** If Maass's lower bound against deciding Palindromes with one-tape nondeterministic Turing machines of subquadratic time can be made P-constructive, then $\mathsf{E} \not\subset \mathsf{SIZE}[2^{\delta n}]$, for some $\delta > 0$.

They also showed that efficient Avoid algorithms imply complexity separations.

Theorem 1.4 (Implicit to Theorem 1.7(i), [14])**.** If there is a uniform $\mathsf{AC}^0[\mathsf{qpoly}]$ algorithm solving Avoid for circuits of size $s(n) = n^{(\log n)^{\omega(1)}}$, then $\mathsf{P} \neq \mathsf{NP}$.

However, not all known lower bounds can be made constructive. Chen et al. [14] observed that there can be no polynomial time algorithm which on input 1^n, outputs an n-bit string of high-$\mathsf{K}^{\mathsf{poly}}$ complexity (see Proposition 4.4). This contrasts in a peculiar way with the lower bounds of Theorem 1.3 and Theorem 1.4. Chen et al. [14] argue that understanding better which lower bounds are likely to be constructive or non-constructive will be key to progress in complexity theory. See their paper for more details.

1.2 Our Results: Student-Teacher Constructive Separations

In this paper, we generalize the results of Chen et al. [14] to the setting of *Student-Teacher* refuters.

Definition 1.5 (Student-Teacher Game (*Informal*))**.** Let $\varphi(\overline{X}) = \forall Y\, \theta(\overline{X}, Y)$, for θ a quantifier-free formula, and let $p(n), q(n)$ be polynomials. We say $S(1^n)$ is a *$\mathcal{C}$-Student-Teacher game* if S is an algorithm in $\mathcal{C}$ with access to a *counterexample oracle* $CX[\varphi]$ which given an $\overline{X} \in \{0,1\}^{p(n)}$, either responds "YES" or returns a $Y \in \{0,1\}^{q(n)}$ such that $\theta(\overline{X}, Y)$ is false. We further write $CX[\varphi, r(n)]$ for a function $r(n)$ to indicate S gets access to $r(n)$ calls to the oracle CX.

[1] An avoidance algorithm can be thought of as a refuter for the "always-YES" algorithm against some hard language. For example, a polynomial time algorithm solving Avoid for the function that maps descriptions of $s(n)$-size circuits to their truth-tables gives a polynomial time algorithm refuting the "always-YES" algorithm for $\mathsf{MCSP}[s(n)]$.

Student-Teacher games are a natural model of computation which appear in learning algorithms and bounded arithmetic. For many complexity lower bounds, provability in a weak logical theory implies an efficient Student-Teacher refuter, rather than a standard refuter as seen in [14]. This means that to study the (un)provability of complexity lower bounds, it is necessary to study Student-Teacher games. We make this connection more explicit in the following section. See a more detailed definition of Student-Teacher games in Section 2.6.

As an example of a Student-Teacher game, consider a P-Student-Teacher refuter S_M solving the refutation search problem $\mathrm{Ref}_{\mathsf{Pal},M}$, with M a one-tape subquadratic time nondeterministic Turing machine. Here, $\varphi_{\mathsf{Pal}}(X, W^*)$ expresses the following formula:

$$\varphi_{\mathsf{Pal}}(X, W^*) \triangleq \text{“For every witness } W \text{ of length } |X|^{1.1}, [M(X, W) = 0 \text{ and } X \text{ is a palindrome] or } [M(X, W^*) = 1 \text{ and } X \text{ is not a palindrome.]”}$$

Each round, S would propose X, W^* to the counterexample oracle CX, where either CX says "YES" if X is an input that M fails to decide whether X is a palindrome, or CX responds to S with a witness W such that M does correctly decide X.

Palindromes. We generalize Theorem 1.3 to P-Student-Teacher refuters.

Theorem 1.6. If for any nondeterministic one-tape subquadratic time Turing machine M there is a P-Student-Teacher game $S_M(1^n)$ with counterexample oracle $CX[\varphi_{\mathsf{Pal}}, O(1)]$ solving $\mathrm{Ref}_{\mathsf{Pal},M}$ for n-bit inputs, then $\mathsf{NP} \not\subset \mathsf{SIZE}[n^k]$ for any $k \geq 0$.

A P-Student-Teacher refuter is considerably stronger than a P-refuter in the context of Palindromes lower bounds. The oracle $CX[\varphi_{\mathsf{Pal}}]$ acts as a restricted NP-oracle, as finding a witness W where $M(X, W)$ is correct is an NP-language. Nevertheless, we show that P-Student-Teacher refuters for one-tape subquadratic NTMs deciding Pal still imply breakthrough circuit lower bounds.

Weak Shannon Counting. We give a slightly orthogonal result to Theorem 1.4. Here, we consider avoidance algorithms for weak Shannon counting. Namely, for a fixed $b \in \mathbb{N}$ and given an input 1^N, output a truth-table of length N which is not computed by a size $(\log N)^b$ Boolean circuit. Let $\varphi_{WSC}(X, b)$ express the following formula:

$$\varphi_{WSC}(X, b) \triangleq \text{“For every circuit } C \text{ of size } (\log |X|)^b\text{, the truth table generated by } C \text{ disagrees with } X \text{ on some bit.”}$$

We introduce the notion of an *absolute* Student-Teacher game, which solves $\mathsf{SIZE}[n^b]$-Avoid for *infinitely many* b. Meaning, the student S takes two inputs: 1^N and b (represented in binary) and solves $\mathsf{SIZE}[n^b]$-Avoid.

Theorem 1.7. If there is an absolute poly log-uniform AC^0[qpoly]-Student-Teacher protocol $S(1^N, b)$ with oracle $CX[\varphi_{WSC}, O(1)]$ solving $\mathsf{SIZE}[n^b]$-Avoid for infinitely many $b \in \mathbb{N}$, then $\mathsf{P} \neq \mathsf{NP}$.

If you simply fix a $b \in \mathbb{N}$ instead of having an absolute Student-Teacher protocol, then such a Student-Teacher protocol *does* exist (see Section 5). However, we only get our consequence $\mathsf{P} \neq \mathsf{NP}$ for an absolute Student-Teacher protocol.

Our notion of an absolute Student-Teacher protocol appears naturally in the context of bounded arithmetic and witnessing theorems. See Section 1.4 and Section 4.3 for a discussion.

High-$\mathsf{K}^{\mathsf{poly}}$ Strings. We now give an avoidance problem which provably has no Student-Teacher game. Let $\varphi_{K^t}(X, b)$ express the following formula:

$$\varphi_{\mathsf{K}^t}(X, b) \triangleq \text{“For every Turing machine and advice pair } (M, \alpha) \text{ of description length } |X|/4\text{, running } M \text{ for } n^b \text{ steps with advice } \alpha \text{ has output disagreeing with } X.\text{”}$$

Let $\mathsf{BHaltDesc}[n^c, p(n)]$ be the class of Turing machine and advice pairs (M, α) of total description length $p(n)$ which, starting with α on the tape of M, runs for n^c time. We then have the following.

Theorem 1.8. There is no absolute P-Student-Teacher protocol $S(1^n, b)$ (b given in unary) with oracle $CX[\varphi_{\mathsf{K}^t}, \mathsf{poly}(n)]$ solving $\mathsf{BHaltDesc}[n^b, n/4]$-Avoid.

1.3 Proof Theoretic Constructivity in Complexity Theory

Our results on Student-Teacher constructive separations have several consequences on the provability of lower bounds in bounded arithmetic. Bounded arithmetic is a related and more fine-grained notion of constructivity that comes from not just the algorithms for refutation and avoidance, but also from the logical expressivity needed to prove the correctness of these algorithms.

Bounded Arithmetic. Bounded arithmetic studies fragments of Peano Arithmetic (PA) which use reasoning inherent to computational complexity classes. The earliest example is the theory $\mathsf{I}\Delta_0$, introduced by Parikh [45]. He showed that reasoning in $\mathsf{I}\Delta_0$ corresponds to the Linear Time Hierarchy (LTH), and that certain operations like exponentiation are infeasible in this theory. One of the most important and well-studied bounded arithmetic theories is Cook's theory VPV. It is an essentially equivalent version of Cook's original theory, PV_1, defined in his seminal 1975 paper [18]. This theory was the first proposed to exactly characterize polynomial-time computation and reasoning, and more generally was the first theory introduced to explicitly connect standard complexity classes and bounded arithmetic. PV stands for *polynomially verifiable*, and one of Cook's original motivations for defining this theory was"that the verification method must be uniform, in the sense that one can see (by the [PV_1-]proof Π) that the verification will always succeed" [19, pg. 83]. For example, if VPV proves a statement like $\forall X\, \varphi(X)$, then there is a polynomial time algorithm $\textsc{Verify}_\varphi$ which on input Y verifies that $\varphi(Y)$ holds.

This constructive property is called *witnessing.* If a theory T proves the existence of some object, then this implies there is an efficient algorithm that generates this object. As an example, suppose VPV were to prove the following Π_2 statement describing a circuit lower bound for language $\mathcal{L}$, which is decided by machine M:

"For every input length n and circuit $C \in \mathcal{C}$, there exists an input x of length n such that $C(x) \neq M(x)$"

Then, the witnessing property for VPV says there is a polynomial time algorithm which finds an incorrect x when given n, C as input. Notice that this a P-refuter!

While a provable Π_2 statement directly translates into a refuter, the situation is more complicated for Π_i statements with $i \geq 3$. Focusing on $i = 3$, witnessing properties give Student-Teacher protocols. For a Π_3 statement $\forall n \exists X \forall Y \theta(n, X, Y)$, a witnessing Student-Teacher protocol S would take as input 1^n, and query the counterexample oracle $CX[\varphi]$ on guesses for a satisfying X, for $\varphi = \forall Y\, \theta(n, X, Y)$. See Sections 2.4 and 2.5 for more details on witnessing theorems.

Π_3 formulas naturally encode many refutation and avoidance type statements. Maass's Palindromes lower bound, Shannon counting, and the existence of High-$\mathsf{K}^{\mathsf{poly}}$ strings are all examples. Hence, the (un)provability of these statements in bounded arithmetic is closely tied to Student-Teacher constructive separations.

A Gap Between Constructive Separations and Provability. In Chen et al. [14], they noted a gap between constructive separations and provability. A P-constructive separation of $\mathcal{B} \not\subset \mathcal{A}$ does not at all guarantee that $\mathsf{VPV} \vdash$ "$\mathcal{B} \not\subset \mathcal{A}$". This can be for several reasons: the refuter might not be *provably correct* inside VPV, or $\mathcal{B} \not\subset \mathcal{A}$ might only be formalizable as a Π_3 formula, which by witnessing gives a Student-Teacher game with an polynomial time student, rather than just a P-refuter. For these reasons, the consequences of a (non)constructive separation of $\mathcal{B} \not\subset \mathcal{A}$ may not have any bearing on the (un)provability of the same lower bound $\mathcal{B} \not\subset \mathcal{A}$.

An explicit example of this gap, given by [17, 11], is proving the correctness of the AKS primality testing algorithm [1]. We can formalize the correctness of it as the following formula:

$$\forall n \left[\mathsf{AKS}(n) = 1 \longleftrightarrow \forall 1 < d < n,\, d \nmid n\right],$$

where AKS is a function symbol for the AKS primality testing algorithm. If this statement were provable in VPV, then there would be a polynomial time algorithm which on input 1^n, n a composite number, would be able to determine a factor of n. Hence, VPV proving the correctness of AKS would imply that factoring has a polynomial time algorithm. This shows that the proof of correctness in [1] of their polynomial time algorithm AKS uses functions which are themselves not polynomial time computable, unless factoring is easy.

Formalizing Lower Bounds as Schemas. Straightforward translations of many complexity lower bounds into the language of bounded arithmetic require *schemas* of formulas: a sequence of formulas indexed by substitution of fixed polynomials $\{n^c\}_{c\in\mathbb{N}}$. Theories of bounded arithmetic cannot quantify over arbitrary polynomials in a natural sense.[2] A simple example of this would be the Deterministic Time Hierarchy theorem.

$$\mathsf{DTIMEH}(c) \triangleq \text{“For every sufficiently large n and Turing machine } M\text{, there exists an input } X \text{ of length } n \text{ where simulating } M \text{ on } X \text{ for } n^c \text{ time incorrectly decides hard language } \mathcal{L}_{c+1}\text{”}$$

For each $c \in \mathbb{N}$, you get a new formula $\mathsf{DTIMEH}(c)$ describing that $\mathsf{DTIME}[n^{c+1}] \not\subset \mathsf{DTIME}[n^c]$. This is necessary in bounded arithmetic as exponentiation is not a feasible operation, so a single formula DTIMEH quantifying over all c is not possible in a theory like VPV. The same issue also occurs when writing down, say, $\mathsf{P} \neq \mathsf{NP}$ in the language of VPV.

This poses a problem when trying to study the provability of lower bounds. Applying witnessing to a schema of Π_3 formulas $\Psi[c]$ would result in a schema of Student-Teacher games, each solving a different search problem parametrized by c. Further, a Student-Teacher game for $\Psi[c_0]$ is not required to share any structure or runtime with a Student-Teacher game for $\Psi[c_1]$, with $c_0 \neq c_1$. This means that our results on *absolute* Student-Teacher games do not automatically imply provability consequences.

1.4 Our Results: Consequences in Bounded Arithmetic

We initiate the study of the following natural question about lower bound schemas.

Question 1.9. Let $\mathsf{LB}(c)$ be the logical translation of some complexity theoretic lower bound, parametrized by $c \in \mathbb{N}$. If $\mathsf{VPV} \vdash \mathsf{LB}(c)$, for every c, then does VPV use the same “proof” for every c?

While it is unclear at all if $\mathsf{VPV} \vdash$ “$\mathsf{P} \neq \mathsf{NP}$”, it is known that $\mathsf{VPV} \vdash \mathsf{DTIMEH}(c)$, for every $c \in \mathbb{N}$. Amazingly, VPV could use “the same” proof for every c! This follows as the *refuter* for DTIMEH is completely agnostic to c, and hence is the same regardless of the value of c. Specifically, there is a hard language $\mathcal{L} \in \mathsf{DTIME}[n^{c+1}] \backslash \mathsf{DTIME}[n^c]$ where a refuter runs linearly in n to construct a counterexample of a proposed machine $M \in \mathsf{DTIME}[n^c]$ deciding $\mathcal{L}$. Does this property of the Deterministic Time Hierarchy theorem hold more generally for other complexity lower bounds?

We denote by ‘Witnessing Hypothesis for Uniform Proofs’ (WHUP) that such a phenomenon in fact holds, and that for certain classes of lower bound schemas, if a theory $\mathcal{T}$ proves the schema, then it does so with the same proof.

Hypothesis 1.10 (WHUP for theory VPV (Informal))**.** Let $\mathsf{VPV} \vdash \forall n\, \exists X\, \forall Y\, \theta(n, X, Y, n^c)$, for a quantifier-free θ and for infinitely many $c \in \mathbb{N}$, and let $\varphi(n, X, c) = \forall Y\, \theta(n, X, Y, n^c)$.[3] Then there is a witnessing *absolute* Student-Teacher game $S(1^n, c)$ which, for infinitely many c, finds a satisfying X of length n using $O(1)$ oracle calls to $CX[\varphi]$.

WHUPs can be used to connect the (un)provability of *schemas* of formulas in bounded arithmetic with *absolute* Student-Teacher constructive separations. Our Witnessing Hypotheses are similar to and inspired by a conjecture of Kreisel for Peano Arithmetic. See Section 4 for a detailed discussion where we carefully define Witnessing Hypotheses and provide many supporting examples for the validity of WHUPs. We conclude this section with a sketch of each of our results on provability.

Palindromes. First, we extend Theorem 1.6 to provability in VPV. Let Pal be a Π_3 formula expressing Maass's lower bound.

Theorem 1.11. If $\mathsf{VPV} \vdash \mathsf{Pal}$, then $\mathsf{NP} \not\subset \mathsf{SIZE}[n^k]$.

This complements recent work of Chen, Li, and Oliveira [11], showing that if Maass's lower bound is provable in VPV, then collision resistant hash functions do not exist.

[2] Bounded theories can include auxiliary quantified variables in a statement to “pad up” and reason about super-polynomial functions. However, this transformation *greatly expands the set of feasible objects.* For example, the padded formalizations of circuit complexity can discuss the entire 2^n-bit truth table of an n-input n^k-gate circuit C — an object inaccessible to poly-time algorithms given only C. Müller and Pich [42] explain the trade-offs in detail.

[3] In the full Witnessing Hypothesis presented in Hypothesis 4.14, we further restrict the structure of the schemas and how c may be used. See Section 4 and Definition 4.13 for more details.

Weak Shannon Counting. Building on the work of Thapen [51], Jeřábek [26], studied the theory $\mathsf{VAPC} :=$ $\mathsf{VPV} + \mathsf{dWPHP}(\mathsf{VPV})$ which adds the *dual weak pigeonhole principle*, the combinatorial principle behind Empty and $\mathcal{C}$-Avoid. He showed VAPC has an intimate relationship with randomized complexity (ZPP) and approximate counting. Namely, all provably total functions in VAPC are contained in ZPP, and conversely, a large natural subclass of ZPP is definable in VAPC. It is possible that VAPC may completely characterize ZPP, but this would require showing that ZPP has a complete problem [51].

VAPC is interesting because of its ability to formalize most known complexity lower bounds. Jeřábek showed that it can formalize Shannon counting arguments, and Müller and Pich [42] further illustrated its power by formalizing Parity lower bounds and the method of approximations. Recent results in *unprovability* have also been shown. Chen, Li, and Oliveira [11] showed that if collision resistant hash functions exist, then Maass's palindromes lower bound is not formalizable in VAPC.

We give a result orthogonal to Jeřábek's provability of Shannon counting in VAPC. We introduce a theory $\mathsf{V}^0_{\#}$ to characterize quasipolynomial AC^0 reasoning. This theory is incomparable to VAPC, but we show it is also capable of proving weak Shannon counting.

Lemma 1.12. Let $b \in \mathbb{N}$. $\mathsf{V}^0_{\#}$ proves the existence of truth tables not computable by Boolean circuits of size n^b.

Further, under a WHUP for $\mathsf{V}^0_{\#}$, we have consequences for the provability of hard truth tables.

Theorem 1.13. Assuming a WHUP for the theory $\mathsf{V}^0_{\#}$, if $\mathsf{V}^0_{\#}$ proves for every $b \in \mathbb{N}$ that there are truth tables of hard for size n^b circuits, then $\mathsf{P} \neq \mathsf{NP}$.

We then get as a clear corollary,

Corollary 1.14. A WHUP for $\mathsf{V}^0_{\#}$ implies that $\mathsf{P} \neq \mathsf{NP}$.

See Theorem 5.10 for full details.

Conditional Separation of V^1 and VAPC We show the following surprising unprovability result.

Theorem 1.15. Under a Witnessing Hypothesis, VPV (or even V^1) cannot show the existence of High-K^{n^b} strings, for almost every $b \in \mathbb{N}$.

As a corollary, we conditionally separate theories V^1 and VAPC (equivalently S^1_2 and APC_1).

Theorem 1.16. Under a Witnessing Hypothesis, VAPC is not equivalent to V^1.

Proof. In the work of Korten [28], it was shown that APC_1 proves the existence of high-$\mathsf{K}^{\mathsf{poly}}$ strings. Furthermore, the APC_1-proofs that "there is a High-K^{n^b} string" for each b seem very "uniform" — the only substantial difference is which dual weak pigeonhole principle is invoked. Each proof uses $\mathsf{dWPHP}(U_d)$ where U_d is the n^d-step universal Turing machine function symbol.

However, under a Witnessing Hypothesis (Hypothesis 4.16), V^1 does not show these strings exist. □

This improves and greatly simplifies the result of Ilango, Li, and Williams [25] separating VAPC and VPV under the existence of indistinguishability obfuscation and $\mathsf{NP} \not\subset$ i.o. coAM. Our result is also the first such conditional separation between any bounded arithmetic theory $\mathcal{T}$ and VAPC which uses a plausible[4] non-cryptographic assumption.

Separating theories of bounded arithmetic should be far easier to prove than demonstrating the existence of cryptographic objects like collision resistant hash functions or indistinguishability obfuscation. It is then desirable to give conditional separations of theories using assumptions much weaker than cryptography. We believe Witnessing Hypotheses are such an assumption. However, Corollary 1.14 indicates that WHUPs are still quite strong, as some will imply major complexity separations. Is this the case with a WHUP for VPV? Are WHUPs in fact EQUAL to the existence of some cryptographic object?

[4] In [33], Krajíček showed that assuming Kolmogorov's Conjecture, $\mathsf{P} \subset \mathsf{SIZE}[n^k]$ for some fixed k, then VAPC is strictly stronger than VPV. However, Kolmogorov's Conjecture is widely believed to be false.

1.5 Our Techniques

Constructive Separations. We employ the general strategy of Chen et al. [14] to show that efficient refuters imply circuit lower bounds.

(i) Assume, for sake of contradiction, a complexity collapse (eg. $\mathsf{P} = \mathsf{NP}$ or $\mathsf{P} \subset \mathsf{SIZE}[n^k]$). Show that a $\mathcal{C}$-refuter from a $\mathcal{C}$-constructive separation of $\mathcal{B} \not\subset \mathcal{A}$ produces outputs of very small circuit complexity.

(ii) Show that there exists a too efficient algorithm $M \in \mathcal{A}$ for a hard language $\mathcal{L}_{\mathcal{B}}$ which is correct on all inputs of low circuit complexity. This forms a contradiction.

As mentioned in Section 1.4, this argument is not sufficient on its own to discuss the consequences of provability of lower bounds, as provability and witnessing implies Student-Teacher refuters instead of standard refuters. To handle this, we introduce several novel *round collapse* techniques to remove the Teacher from Student-Teacher protocols. This gives a reduction to the proof strategy of [14].

Round Collapses. Round collapse techniques have seen widespread recent study to show the unprovability of Π_3 sentences in theories of bounded arithmetic [8, 25, 33, 9, 16, 47]. We continue this line of work by introducing three novel round collapse arguments.

A common issue with round collapse techniques is that they are very ad hoc and strongly depend on the discussed lower bound. In the case of Maass's Palindromes lower bound, we introduce in Section 3 a very general technique to deal with collapsing a P-Student-Teacher protocol whose counterexample oracle $CX[\varphi, O(1)]$ solves an NP-language. Recall for a one-tape subquadratic time NTM M, $\varphi_{\mathsf{Pal}}(X)$ certifies that for every witness W, $M(X, W) = 0$ when X is a palindrome. Hence the counterexample oracle $CX[\varphi_{\mathsf{Pal}}]$ solves the NP language of determining a witness W' where $M(X, W') = 1$. Assuming $\mathsf{NP} \subset \mathsf{SIZE}[n^k]$, we may use the Easy Witness Lemma of Murray and Williams [43] to give a compressed description of W'. By providing this compression of W' as advice to the Student, we can replace a single query to $CX[\varphi_{\mathsf{Pal}}]$. Repeating this argument allows the conversion of a P-Student-Teacher refuter into a $\mathsf{P}/o(n)$-refuter.

Our other round collapses are much more ad hoc. For weak Shannon counting and Theorem 1.7, we generalize the technique of Chen et al. [14] to efficiently simulate a $\mathsf{polylog}$-uniform $\mathsf{AC}^0_d[\mathsf{qpoly}]$ refuter $C(1^n)$ with a general sublinear size Boolean circuit, assuming $\mathsf{P} = \mathsf{NP}$. Their idea is to show that computing the i-th bit of $C(1^n)$ is a $\Sigma^p_d[\mathsf{polylog}(n)]$ problem, which collapses to $\mathsf{DTIME}[\mathsf{polylog}(n)]$ under $\mathsf{P} = \mathsf{NP}$. We show this argument completely in Lemma 5.5. Where we must generalize this argument is to further allow a $\mathsf{polylog}$-uniform $\mathsf{AC}^0[\mathsf{qpoly}]$ Student-Teacher refuter, and provide a method to remove the CX oracle gates. We do so in Section 5.

The round collapse for high-$\mathsf{K}^{\mathsf{poly}}$ strings and Theorem 1.8 is conceptually the most natural. We take direct inspiration from the $\mathsf{DTIME}[n]$-constructive separation of $\mathsf{DTIME}[n^{c+1}] \not\subset \mathsf{DTIME}[n^c]$. The linear time refuter $R_M(1^n)$ simply outputs the padded source code of M, $\langle M \rangle \circ 0^{n-|\langle M \rangle|}$. Our observation in Section 4 is that for an absolute P-Student-Teacher protocol solving $\mathsf{BHaltDesc}[n^c, n/4]$-$\mathsf{Avoid}$ for all $c \in \mathbb{N}$, the source code of the Student is a valid response for the counterexample oracle $CX[\varphi_{\mathsf{K}^t}]$. We give a fine-grained reflection argument to generally transform a P-Student-Teacher protocol for $\mathsf{BHaltDesc}[n^c, n/4]$-$\mathsf{Avoid}$ into a polynomial time algorithm, even when polynomially many Teacher queries are made by the Student-Teacher protocol.

1.6 Comparison to Other Work

AC^0 reasoning and Provable Circuit Lower Bounds. Several previous works have studied the provability of circuit lower bounds in bounded arithmetic via round collapses. Pich [47] showed unconditionally that the theory V^0, corresponding to log-uniform $\mathsf{AC}^0[\mathsf{poly}]$ reasoning, cannot prove superpolynomial size circuit lower bounds. This contrasts with our Lemma 1.12, where we show that the theory $\mathsf{V}^0_{\#}$ proves fixed polynomial size circuit lower bounds. This suggests an intriguing question of finding the exact logical strength necessary for proving fixed polynomial size circuit lower bounds.

Separating VAPC and VPV. Krajíček [33] gave the first conditional separation of VPV and VAPC via round collapse techniques. Unfortunately, his round collapse required the unlikely assumption that $\mathsf{P} \subset \mathsf{SIZE}[n^k]$. In [32], Krajíček called for reasonable assumptions under which VAPC is strictly stronger than VPV. This was achieved by Ilango, Li, and Williams [25], who showed that under indistinguishability obfuscation and $\mathsf{NP} \not\subset$ i.o. coAM, these theories are indeed separated. These conditional separations of [33, 25] were accomplished by studying the dual weak pigeonhole principle dWPHP(VPV) and Student-Teacher protocols for solving Empty.

Using a Witnessing Hypothesis, we conditionally separate VAPC from an even stronger theory V^1, which contains VPV. Further, we make use of a weaker, *uniform* version of the dual weak pigeonhole principle.

1.7 Open Problems

This work suggests several continuations and open problems. We provide two directions pertaining to (un)provability, and several towards understanding and proving WHUPs.

Improving the Palindromes round collapse. While we show that a constant round Student-Teacher refuter for Palindromes would imply $\mathsf{NP} \not\subset \mathsf{SIZE}[n^k]$ for any $k > 0$, we fall short of proving this for $\omega(1)$ rounds. Is there a polynomial round Student-Teacher refuter for Palindromes? This could be used to extend our provability consequence to theories V^1/S^1_2.

Unprovability for APC^1. In Section 4, we show that under a Witnessing Hypothesis for VPV, generating high $\mathsf{K}^{\mathsf{poly}}$ strings is not feasible in VPV. Can this be generalized to unprovability in APC^1. This would likely have to be for a notion of *zero-error* time-bounded Kolmogorov complexity, which the authors are unaware of appearing in the present meta complexity literature.

What do proofs look like? Amongst our examples of the "absolute" witness phenomenon, like the refuter for DTIMEH, what do the VPV proofs *actually* look like? This would be a basic building block to understand before attempting to prove a WHUP. We emphasize that we know the proofs, but not a structural measure or property that makes it clear they are "the same" across different values of the parameter. Surprisingly simple polynomial schemas have proofs in VPV where we do not have a solid understanding of their structure. One example is,

$$\varphi(b, c) \triangleq \forall n \; c > b \rightarrow n^c > n^b.$$

As VPV is defined for the purpose of encapsulating polynomial time computation (rather than performing arithmetic), even simple arithmetic identities can have "complicated" proofs. Showing that the sequent calculus proofs of $\varphi(b, c)$ over VPV are the same for all $b, c \in \mathbb{N}$ would be of interest.

Correct Formulation of "Same" Proofs. We phrase our WHUPs based on the notion of Herbrand proofs from the famous Herbrand's Theorem in first order logic. This allows us to interplay with witnessing theorems nicely. However, it is possible that our notion of "same proof" is still too coarse, and that WHUPs would be more appropriately phrased in another way. One potential example would be the notion of *uniform* proofs, proposed by Buss [4], where proofs are given an efficiently decidable direct connection language as you would a circuit.

Proving a WHUP. Perhaps the most obvious would be actually showing a WHUP to be true for VPV, $\mathsf{V}^0_{\#}$, or any other bounded arithmetic theory. We believe that past work on Kreisel's Conjecture [24, 23, 30] serves as an excellent starting point. For example, Krajícek and Pudlák [30] show that Kreisel's Conjecture is true over any theory which is finitely axiomatizable, of which many theories of bounded arithmetic are (including V^1 and $\mathsf{V}^0_{\#}$).

Consequences of WHUPs. Corollary 1.14 shows that WHUPs can have immediate consequences if true. Are there more examples of WHUP consequences, but for standard theories like VPV and V^1? Further, are there consequences if WHUPs are *false*?

1.8 Paper Organization

In Section 2, we give the requisite preliminaries in bounded arithmetic and complexity theory. In Section 3, we give our results on Student-Teacher constructive separations for Palindromes, and its applications to provability in VPV. In Section 4, we introduce Witnessing Hypotheses for Uniform Proofs and apply them to get the unprovability of finding high-$\mathsf{K}^{\mathsf{poly}}$ strings in VPV. We further give a detailed discussion of the viability of WHUPs and their inspiration from the famous Kreisel Conjecture in logic. Finally, in Section 5, we introduce the theory $\mathsf{V}^0_{\#}$ and show that a WHUP for this theory would imply $\mathsf{P} \neq \mathsf{NP}$.

2 Preliminaries

Basic knowledge of complexity classes is assumed. See [3] or any text on complexity theory for a reference of the standard definitions. We attempt to keep this paper as self-contained as possible for mathematical logic and bounded arithmetic; however, we recommend seeing the SIGACT column of Oliveira [44] which surveys much of the recent work on the provability of complexity theory. This survey provides invaluable context to the motivations of this paper.

2.1 Circuit Uniformity

A family of circuits $C = \{C_n\}_{n\geq 1}$ is called uniform if some uniform algorithm is able to, on input n, compute a fixed binary encoding of $\langle C_n \rangle$. We will use the *direct connection* encoding of circuits, where $\langle C_n \rangle_i = 1$ if and only if i encodes a triple (g, h, r) with g and h being gate indices, and r indicating the type of g (namely, one of NOT/AND/OR/INPUT/OUTPUT). In the case that r is an INPUT type, it must also indicate which input bit out of n. Topologically, h feeds into g as an input, unless r indicates that g is an INPUT type. We will also need an *oracle direct connection* encoding. This is a slight modification where we add two types of gates: ORACLE, and Oracle OUTPUT, where ORACLE represents a black-box oracle that takes in $p(n)$ bits and outputs $q(n)$ bits. For both of these gate types, r must also indicate which of the $p(n)$ input bits a gate h is feeding into ORACLE, or which of the $q(n)$ output bits an OUTPUT gate g is. Note that given a circuit with $s(n)$ gates, its (oracle) direct connection encoding will be of length at most $s(n)^3$.

Definition 2.1 (LOGTIME-uniformity)**.** We say that a circuit family $C = \{C_b\}_n$, where C_n is of size $s(n)$, is *logtime uniform* if there is a linear time algorithm U which on input n and an index $i < |\langle C_n \rangle|$, both represented in binary, outputs the i-th bit of $\langle C_n \rangle$. Similarly, such a circuit family is *polylogtime uniform* if the uniform algorithm U runs in time polynomial in the input size.

2.2 Basic Logic and Terminology

We will assume basic knowledge of propositional and first-order logic, as well as Gentzen's sequent calculus. We remind the reader of some of the standard syntax below. For a concise and complete introduction to the necessary logic and proof theory, see Chapters I-III of [17] or Chapters I and II of [5].

Definition 2.2 (Syntax)**.**
Symbols and Terms: The symbols appearing in first-order logic are the usual logical connectives ($\neg$, $\wedge$, $\vee$, $\rightarrow$), quantifiers ($\forall$, $\exists$), specified function and predicate symbols, and constants (0-arity functions). As well, arbitrary names for variables are allowed. A *term* is inductively defined: any variable x is a term, and for any function symbol f of arity k and terms $t_1, \ldots, t_k$, $f(t_1, \ldots, t_k)$ is a term.
Formulas: A *formula* is also inductively defined. Atomic formulas are of the form $P(t_1, \ldots, t_k)$ for a predicate P of arity k, and general formulas are built up from atomic ones by applying logical connectives and quantifiers. We say a variable x in a formula is bound if it is in the scope of a quantifier $Q\,x$. Otherwise, it is free. A formula with no free variables is called a *sentence.*
Substitution: Let $A(x)$ be a formula with x a free variable. For a term t, we denote $A(t/x)$ to be the *substitution* of t for x in A, where we replace every occurence of the free variable x in A with t.

Definition 2.3. A *first-order theory* T is a set of sentences which is closed under logical implication. Specifically, if T derives via a sequent calculus proof the sentence φ, then $\varphi \in T$. A set of sentences Γ are an *axiomatization* of T if $\Gamma \subset T$ and all of T is derivable from Γ via sequent calculus proofs. The *language* of a theory T, $\mathcal{L}(T)$, is the set of symbols for functions, predicates, and constants (0-ary functions) used in the logical sentences contained in T. A theory is said to be *universal* if it has an axiomatization with only universally quantified sentences in prenex normal form.

We can compare theories by considering the set of theorems that they prove. The appropriate notion is

Definition 2.4 (Conservative Extension)**.** Suppose that $\mathcal{T}_1$ and $\mathcal{T}_2$ are two theories where $\mathcal{T}_1 \subseteq \mathcal{T}_2$ and the vocabulary of $\mathcal{T}_2$ may contain function or predicate symbols not in $\mathcal{T}_1$. We say $\mathcal{T}_2$ is a *conservative extension* of $\mathcal{T}_1$ if for every formula φ in the vocabulary of $\mathcal{T}_1$, if $\mathcal{T}_2 \vdash \varphi$ then $\mathcal{T}_1 \vdash \varphi$.

In other words, the second theory proves nothing new over the original vocabulary.

In this paper, we study the first-order theory of arithmetic, Peano Arithmetic (PA), as well as its subtheories. We denote by $\mathbb{N}$ the standard model of PA, which should be interpreted as the 'real world'. The defining feature of Peano Arithmetic (and its intended model $\mathbb{N}$) is induction: for a formula $\varphi(x, \overline{y})$, the *axiom of induction*, $I_x\varphi$, is the sentence:

$$\forall \overline{y} \left(\varphi(0, \overline{y}) \wedge \forall x \left(\varphi(x, \overline{y}) \rightarrow \varphi(x+1, \overline{y}) \right) \rightarrow \forall x \varphi(x, \overline{y}) \right).$$

Peano Arithmetic is defined by basic arithmetic axioms and the axiom of induction for every formula φ. For a restricted class of formulas Φ, we define $\mathsf{I}\Phi$ as the subtheory of PA with induction restricted to formulas $\varphi \in \Phi$.

2.3 Peano Arithmetic

We recall the characterization of provably recursive functions of $\mathsf{I}\Sigma_n$ [7].

Definition 2.5. Let T be a subtheory of PA and $f : \mathbb{N}^k \rightarrow \mathbb{N}$. The function f is Σ_i-definable in T iff there is a formula $\varphi(x_1, \ldots, x_k, y) \in \Sigma_i$ such that:

1. $T \vdash (\forall \vec{x})(\exists! y)\varphi(\vec{x}, y)$

2. $\{(\vec{a}, b) \ : \ \mathbb{N} \models \varphi(\vec{a}, b)\}$ is the graph of f, i.e. $\varphi(\vec{a}, b)$ holds iff $f(\vec{a}) = b$ for all naturals $\vec{a}, b$.

Σ_1-definable functions in a theory $\mathcal{T}$ are also commonly called the *provably recursive* functions of $\mathcal{T}$.

Lemma 2.6 (Informal)**.** Let f be a function that is provably recursive in PA. Then we can freely add the function symbol f to $\mathcal{L}(\mathsf{PA})$ and the defining axioms of f to PA without modifying the strength of PA.

Definition 2.7. Let $n \geq 1$. The set of functions which are *primitive recursive in* Σ_n is defined inductively by:

1. Constant function 0, succcessor function, and all projection functions are primitive recursive in Σ_n.

2. Closure under composition.

3. If $g : \mathbb{N}^k \rightarrow \mathbb{N}$ and $h : \mathbb{N}^{k+2} \rightarrow \mathbb{N}$ are primitive recursive in Σ_n, then so is the function f defined by

$$f(0, \vec{z}) = g(\vec{z})$$
$$f(m+1, \vec{z}) = h(m, \vec{z}, f(m, \vec{z}))$$

4. If $\varphi(\vec{z})$ is a formula $(\exists x)\psi(x, \vec{z})$ where $B \in \Pi_{n-1}$ then U_A is primitive recursive in Σ_n.

Theorem 2.8 (Theorem 12, [7])**.** The Σ_n-definable functions of $\mathsf{I}\Sigma_n$ are the functions which are primitive recursive in Σ_n.

2.4 Theories of Bounded Arithmetic

We will be working with two-sorted theories, which deal with both a number-type (think in $\mathbb{N}$) and a finite binary string type. The binary string type has an equivalent interpretation as a set type, where the i-th index of a string X being 1 indicates that i is in the set X. We follow the convention of denoting numbers in lower case (x, y, $z, \dots$) and strings in upper case (X, Y, $Z, \dots$). All theories in this paper are theories of arithmetic, and all share the language of arithmetic ($\mathcal{L}_A^2$), which contains the set of first-sort functions and predicates, $\{0, 1, +, \cdot, S, |\cdot|; =, \leq\}$ and the set of second-sort functions and predicates, $\{X(\cdot), |\cdot|; =_2\}$. Here, S refers to the successor function of a number, $X(i)$ outputs in number type the i-th bit of string X, and $|\cdot|$ on a string-type variable outputs a number-type which is the length of the string.

In two-sorted bounded arithmetic theories, function symbols can be thought of as the run of some resource-bounded computational model (eg. Turing machines or uniform circuits). As such, the representation of its inputs becomes important. We will take the standard convention that the string-type is presented "as itself" in binary and a number-type is represented in unary when supplied as input to a function symbol. A feature of Peano Arithmetic and its subtheories is that any function f which is "easily" definable and provably total may be freely added to the language without changing the provability of any sentences. Below, we will specify exactly what these functions are for each theory we use.

Definition 2.9. We denote a number quantifier as *bounded* by writing $\forall x < t\, \theta(x)$ or $\exists x < t\, \theta(x)$, for a term t not using x. This is syntactic shorthand for $\forall x\, [x < t \implies \theta(x)]$ and $\exists x\, [x < t \wedge \theta(x)]$ respectively. Similarly for quantifiers over strings, we say write $\forall X < t\, \theta(X)$, and $\exists X \leq t\, \theta(X)$ to indicate $\forall X\, (|X| < t \implies \theta(X))$ and $\exists X\, (|X| < t \wedge \theta(X))$. We say that a formula φ is $\Sigma_0^B = \Pi_0^B$ if the only quantifiers are bounded quantifiers over the number type (though there may be free string variables). A formula φ is $\Sigma_{i+1}^B/\Pi_{i+1}^B$, for $i \geq 0$, if φ is of the form, $\exists X < t\, \theta(X)$, for $\theta(X)$ a Π_i^B formula, or respectively, $\forall X < t\, \theta(X)$, for $\theta(X)$ a Σ_i^B formula.

Σ_i^B formulas can be thought of as an effective version of the arithmetic hierarchy, and bears many similarities and connections to the polynomial hierarchy.

Definition 2.10 (Provably Total Functions)**.** Let T be a two-sorted subtheory of PA and $f : \mathbb{N}^k \to \mathbb{N}$. The function f is Σ_i^B-definable in T iff there is a Σ_i^B-formula $\varphi(x_1, \dots, x_k, y)$ such that:

1. $T \vdash (\forall \vec{x})(\exists! y)\varphi(\vec{x}, y)$
2. $\{(\vec{a}, b) \;:\; \mathbb{N} \models \varphi(\vec{a}, b)\}$ is the graph of f, i.e. $\varphi(\vec{a}, b)$ holds iff $f(\vec{a}) = b$ for all naturals $\vec{a}, b$.

Σ_1^B-definable functions in a theory $\mathcal{T}$ are also commonly called the *provably total* functions of $\mathcal{T}$.

We may give a lemma similar to Lemma 2.6 for provably total functions.

Lemma 2.11 ((Informal))**.** Let f be a function that is provably total in a two-sorted theory T. Then we can freely add the function symbol f to $\mathcal{L}(T)$ and the defining axioms of f to T without modifying the strength of T.

Theory V^0. One of the weakest and most basic of theories in bounded arithmetic that is studied is Cook and Nguyen's theory V^0, which captures uniform-AC^0 reasoning. It is a uniform version of the propositional proof system AC^0-Frege, and superpolynomial lower bounds for AC^0-Frege imply unprovability in V^0.

At the base of V^0 are the so-called 2-BASIC axioms, which define the basics of how each function and predicate in $\mathcal{L}_A^2$ behaves. This includes statements like $x \cdot 0 = 0$, distributivity of addition over multiplication, and many others. See [17] for the full list of axioms. In addition to 2-BASIC are the comprehension axioms Σ_0^B-COMP, where for any Σ_0^B-formula φ, we get the axiom,

$$\exists X \leq y\, \forall z < y\, X(z) \longleftrightarrow \varphi(z).$$

Σ_0^B-COMP axioms should be thought of as giving V^0 the power to generate truth tables of AC^0-computable functions. V^0 will, in addition to $\mathcal{L}_A^2$, have a function symbol f in its language for every $\mathsf{LOGTIME}$-uniform AC^0 function f, and the Σ_1^B-defining axiom of f added to V^0. Note it is well-known that $\mathsf{LOGTIME}$-uniform AC^0 is equivalent to the $\mathsf{LOGTIME}$ Hierarchy, so we may include functions symbols for either.

V^0 is surprisingly powerful and expressive. It is capable of proving many elementary theorems about number theory and combinatorics and can perform Gödel numbering and coding of sequences. It is known that V^0 cannot reason about the Parity function ($\oplus$) or other functions which have AC^0 lower bounds.

Theory VPV. The full definition of VPV is involved, and the details do not matter here outside of its correspondence with polynomial time functions. To see a detailed definition of VPV, see [17]. The language of VPV is $\mathcal{L}_A^2$ along with a symbol f for any polynomial-time computable function f. The theory is defined by initially adding the defining axioms of five uniform-AC^0 functions, and then using Cobham's recursive definition of polynomial time functions [15] within the theory to build out the rest of FP.

Theory V^1. Adding the comprehension axioms Σ_1^B-COMP to 2-BASIC, we go from V^0 to V^1. As every polynomial time function is Σ_1^B-definable in V^1, we may freely add their definining axioms to the theory and add a function symbol for every $f \in \mathsf{FP}$. This theory characterizes polynomial time computation and reasoning, similarly to VPV. It has the benefit of being much easier to define, and is more easily generalizable to reflect reasoning in the i-th level of the polynomial hierarchy (theory V^i). It is known that $\mathsf{VPV} \subseteq \mathsf{V}^1$, but it is open if VPV and V^1 are in fact equal; under cryptographic assumptions like the hardness of factoring, Thapen showed that V^1 is strictly stronger [51]. As we shall also see, there is an important difference in the witnessing theorems for VPV compared to the witnessing theorems for V^1.

V^1 (and more generally V^i, for $i > 0$), are equivalent to the single-sorted theories S_2^i introduced by Buss in his seminal PhD Thesis [6].

VPV **Function Symbols** We will be translating several lower bounds against Turing machines of different resource bounds. In order to give VPV-translations of these statements, we must introduce some preliminary function symbols.

Let $\mathsf{Run}_M(X, n)$ be the VPV function symbol that on input X and clock bound n, runs M for n steps on input X and outputs the contents of its tape. Similarly for a nondeterministic machine M, an input X, clock n, and witness W supplied on a separate read-only witness tape, we have a VPV function symbol $\mathsf{Run}_M(X, n, W)$ which run M for n steps on input X and nondeterminism W and outputs the contents of its tape input/work tape.

Lemma 2.12 (Implicit in [18, 6])**.** There is a paddable encoding of one-tape deterministic Turing machines $\mathcal{L}_{TM} \subset \{0,1\}^*$ which is *decodable* in VPV. Specifically, there is a VPV function symbol $\mathsf{Run}(M, X, n)$ where for every Turing machine M and its binary encoding $E_M \in \mathcal{L}_{TM}$, $\mathsf{VPV} \vdash \forall X\, \forall n\ \mathsf{Run}_M(X, n) = \mathsf{Run}(E_M, X, n)$.

Similarly for one-tape *nondeterministic* Turing machines, we can give an encoding language $\mathcal{L}_{NTM} \subset \{0,1\}^*$ which is decodable. Specifically, there is a VPV function symbol $\mathsf{Run}(M, X, n, W)$ where for every nondeterministic Turing machine M and its binary encoding $E_M \in \mathcal{L}_{NTM}$, $\mathsf{VPV} \vdash \forall X\, \forall W\, \forall n\ \mathsf{Run}_M(X, n, W) = \mathsf{Run}(E_M, X, n, W)$.

The above lemma can be reformulated for k-tape Turing machines for any number k, but we will only be concerned with one-tap machines in this paper. We will always assume Turing machines are encoded as $\mathcal{L}_{TM}$ from Lemma 2.12.

2.5 Witnessing Theorems in Bounded Arithmetic

Witnessing theorems broadly show that if a theory $\mathcal{T}$ proves a $\forall\Sigma_i^B$ formula φ, then there is a function f_φ computable in a complexity class $\mathcal{C}_\mathcal{T}$ which finds a witness to the existential quantifiers in φ. We will largely work only with $\forall\Sigma_1^B$ and $\forall\Sigma_2^B$ formulas, which make witnessing conceptually simpler due to there being a single existential quantifier.

The most classical example of witnessing in Bounded Arithmetic is Buss Witnessing [6], which is written in the language of two-sorted theories in [17].

Theorem 2.13 (Buss Witnessing, [6, 17])**.** Let $\mathcal{T}$ be either V^1 or VPV, and let φ be a Σ_1^B formula. Suppose that

$$\mathcal{T} \vdash \forall X \exists Y \varphi(X, Y).$$

Then there exists a function $F \in \mathsf{FP}$ such that $\mathbb{N} \models \forall X \varphi(X, F(X))$.

Krajíček, Pudlák, and Takeuti generalized Buss Witnessing to $\forall\Sigma_2^B$ formulas as follows.

Theorem 2.14 (KPT Witnessing Theorem, [34])**.** Let T be a universal theory with language $\mathcal{L}$. Suppose that for a Σ_0^B formula φ,

$$T \vdash \forall X \, \exists Y \, \forall Z \;\; \varphi(X, Y, Z).$$

Then for a constant $k \geq 1$ and a sequence $C_1, \ldots, C_k$ of $\mathcal{L}$-string terms,

$$T \vdash \forall X \, \forall \overline{Z} \; [\, \varphi(X, C_1(X), Z_1) \,\vee\, \varphi(X, C_2(X, Z_1), Z_2) \,\vee \cdots \vee\, \varphi(X, C_k(X, Z_1, \ldots, Z_{k-1}), Z_k) \,] \,.$$

This theorem applies to VPV, as VPV is a universal theory. For V^0 and V^1, KPT Witnessing as above cannot be immediately applied as neither theory is universal. There are several ways around this. One way is to universalize the axioms of V^0 and V^1 to give conservative extensions $\overline{\mathsf{V}}^0$ and $\overline{\mathsf{V}}^1$, where KPT Witnessing can be applied. The other way is to prove KPT Witnessing directly using proof theoretic arguments and Buss Witnessing. For V^0, we will use the former method and apply the above KPT Witnessing Theorem to the universal $\overline{\mathsf{V}}^0$. For V^1, we present its own KPT Witnessing Theorem below.

Theorem 2.15 (KPT Witnessing Theorem for V^1, [31])**.** Suppose that for a Σ_0^B formula φ,

$$\mathsf{V}^1 \vdash \forall X \, \exists Y \, \forall Z . (|Z| < |X|) \, \varphi(X, Y, Z).$$

Then there is an FP function F such that,

$$\mathbb{N} \models \forall X \, \forall Z . (|Z| < |X|) \;\; \varphi(X, F(X), Z),$$

where F has access to the counterexample oracle $CX[\varphi]$ which on query (X, Y) outputs a string Z of length at most $|X|$ such that $\mathbb{N} \models \neg\varphi(X, Y, Z)$ or "yes" otherwise.

The Student-Teacher game interpretation of KPT Witnessing is very useful. A Student F, which is a search algorithm of some complexity class $\mathcal{C}$, will take in X as input and want to find a Y such that $\forall Z \; \varphi(X, Y, Z)$. They start by proposing $F_1(X)$ to the Teacher, the counterexample oracle, who either says $F_1(X)$ is correct or gives a Z_1 back to the Student as a counterexample. This repeats for r rounds until the Student proposes a correct Y.

A difference between VPV and V^1 is revealed here: the Student-Teacher game from the KPT Witnessing for VPV ends in constantly many rounds, while the Student-Teacher game for V^1 ends in polynomially many rounds. This makes unprovability of $\forall\Sigma_2^B$ formulas in V^1 potentially much harder than in VPV. Unprovability of $\forall\Sigma_2^B$ formulas usually goes by applying KPT Witnessing and showing the resulting Student-Teacher game can collapse into an impossibly fast/small algorithm *without* the counterexample oracle. The more rounds of a Student-Teacher game, the harder it is to prove that the oracle may be removed.

2.6 Student Teacher Games and Refuters

We formally introduce the Student-Teacher game framework which witnesses the KPT Witnessing Theorem.

Definition 2.16 ($\mathcal{C}$-$\mathrm{ST}^{CX[\varphi,r]}$ uniformity)**.** Let $\mathcal{C}$ be a complexity class, and for a term t, let

$$\psi \coloneqq \forall n \, \exists Y \, (|Y| < t(n)) \, \forall Z \, (|Z| = n) \;\; \varphi(n, Y, Z)$$

be a formula with $\varphi \in \Sigma_0^B$ and $\mathbb{N} \models \psi$. As well, let $r(n)$ be a time-constructible function. Define Search_φ to be the total search problem $\mathrm{Search}_\varphi \coloneqq \{(n, Y) \mid Y \text{ a binary string such that } \mathbb{N} \models \forall Z \, (|Z| = n) \; \varphi(n, Y, Z)\}$.

We say that $\mathcal{A}$ is a $\mathcal{C}$-$\mathrm{ST}^{CX[\varphi,r]}$ search algorithm for Search_φ if $\mathcal{A} \in \mathcal{C}$ and on input 1^n, $\mathcal{A}$ outputs a Y satisfying $\forall Z \, (|Z| = n) \, \varphi(n, Y, Z)$ using at most $r(n)$ many oracle queries to the counterexample oracle $CX[\varphi]$.

Many complexity lower bounds are easily formalizable as either $\forall\Sigma_1^B$ or $\forall\Sigma_2^B$ formulas in $\mathcal{L}(\mathsf{VPV})$ and $\mathcal{L}(\mathsf{V}^0)$, where the existential quantifier witnesses a mistake that some Turing machine or algorithm has made when attempting to decide a hard language. Applying witnessing theorems to these lower bounds when they are provable in bounded arithmetic gives us *refuters*.

Suppose, say, VPV were to prove a complexity lower bound formalizable as $\forall\Sigma_2^B$ formula ψ. Applying KPT Witnessing, we would then get an P-$\mathrm{ST}^{CX[\varphi,r]}$ constructive separation. For a $\forall\Sigma_1^B$ formalizable lower bound, Buss Witnessing then directly gives a P-refuter and a P-constructive separation.

2.7 Time-Bounded Kolmogorov Complexity

There are many ways to define time-bounded Kolmogorov complexity [2, 35]. Some choices made in these definitions are essentialy arbitrary, like which efficient universal Turing Machine to use. We will specify these choices carefully enough to give a particular translation of time-bounded Kolmogorov complexity into theories of (bounded) arithmetic, but our results will not depend on the precise formalization. We follow Section 2.2 of [38], elaborating on some details.

Fix a *string pair encoding function* $\langle \cdot, \cdot \rangle : \{0,1\}^+ \times \{0,1\}^+ \to \{0,1\}^+$ defined by the map $\langle u, v \rangle \mapsto \mathrm{dbl}(u) \circ 01 \circ v$, where $\mathrm{dbl}(u) = u_1 u_1 \circ u_2 u_2 \circ \cdots \circ u_{|u|} u_{|u|}$ simply double-prints each bit of u. Denote by π_1 and π_2 the left and right extraction functions, so $\pi_1(\langle u, v \rangle) = u$ and $\pi_2(\langle u, v \rangle) = v$. These pair encoding and element extraction function are linear-time computable and well-defined for all non-empty binary strings. Furthermore, delimiter overhead is only incurred for the length of the first string, plus an additive constant: $\forall u, v \ |\langle u, v \rangle| = 2|u| + 2 + |v|$.

Fix U a Universal Turing machine that can emulate any single-tape Turing Machine M with at most polynomial-time overhead. Let $\mathsf{run}_U(M, x, 1^t)$ denote the function that outputs the entire non-blank contents of the tape of M simulated on input x for t steps of U. By the assumption that U is efficient, run_U can be computed in time $\mathsf{poly}(|M|, |x|, t)$.

Finally, the *t-time bounded Kolmogorov Complexity $K^t(x)$ of a string x* is the length of the shortest two-part description d of x such that U decodes d into x:

$$K^t(x) = \min_{d \in \{0,1\}^*} \{|d| \ : \ U(\pi_1(d),\ \pi_2(d),\ 1^{t(|x|)}) = x\}$$

The K^t complexity of any string x is at most $|x|$, because the two-part description can simply "memorize" x. Consider the description $d = \langle H, x \rangle$ where H is the constant-length description of a Turing Machine that immediately halts. Because run outputs the contents of the tape of H, this is simply x. Thus we have the following

Fact 2.17. There is an absolute constant c such that for every function $t(n) > 0$ and every $x \in \{0,1\}^+$ it holds that $K^t(x) \le |x| + c$.

Observe that it is important to pay delimiter overhead for the constant-length machine H instead of the variable-length string x to obtain the basic fact above. This is implicit in every reasonable definition of time-bounded Kolmogorov complexity.

3 Provability of Palindromes Lower Bounds

In this section, we generalize the work of Chen et. al. [14] and show that provability of the palindromes lower bound in VPV implies circuit lower bounds.

To do this, we formalize Maass's lower bound as a $\forall\Sigma_2^B$ $\mathcal{L}(\mathsf{VPV})$-sentence and, assuming $\mathsf{VPV} \vdash$ "Maass", apply the KPT Witnessing theorem. We then assume a complexity upper bound that *both* collapses the Student-Teacher refuter into a P-refuter and causes a contradiction via the argument of [14].

In Section 3.1, we give a formalization of palindrome lower bounds and discuss its witnessed Student-Teacher refuter under VPV-provability. We then give a slightly generalized version of the constructive separations argument of [14]. Finally, in Section 3.3, we identify a complexity assumption that both collapses the Student-Teacher refuter and allows a standard constructive separations argument to go through.

3.1 Formalization of One-Tape Nondeterministic Turing Machine Lower Bounds

First, we state Maass's theorem in plain English.

Theorem 3.1 ([39])**.** The language PAL $:= \{p \in \{0,1\}^* \mid p \text{ a palindrome}\}$ is not computable by any one-tape nondeterministic Turing machine in $n^{1.1}$ steps.

To formalize Theorem 3.1, we will need to introduce several functions, all of which are clearly VPV function symbols. The symbol $\mathsf{ValNTM}(\cdot)$ takes in a string M and outputs 1 if and only if M is a valid encoding ($M \in \mathcal{L}_{NTM}$) of a one-tape nondeterministic Turing machine. We define $\mathsf{IsPal}(X)$ to output 1 if the string X is a palindrome, and 0 otherwise. Recall that $\mathsf{Run}(M, X, t, W)$ outputs 1 if nondeterministic machine M on input X with guess bits W ACCEPTS within t steps. Finally,

$$\mathsf{Err}^i_{PAL}(M, X, t, W) \triangleq (\mathsf{IsPal}(X) = i) \wedge (\mathsf{Run}(M, X, t, W) = 1 - i).$$

Let $\mathsf{Pal}(n_0)$ denote the following sentence.

$$\begin{aligned}\mathsf{Pal}(n_0) \triangleq \forall n\,(n > n_0)\,\forall M\,(|M| \leq n/2)\,\exists X\,(|X| = n)\,\exists W_X\,(|W_X| \leq n^{1.1})\,\forall W\,(|W| \leq n^{1.1})\\ \mathsf{ValNTM}(M) \wedge \left(\mathsf{Err}^1_{PAL}(M, X, n^{1.1}, W) \vee \mathsf{Err}^0_{PAL}(M, X, n^{1.1}, W_X)\right)\end{aligned}$$

The formalization covers two cases: either the machine M claims an input X is a palindrome when it is not (captured by Err^0), or it claims X is not a palindrome when it in fact is (captured by Err^1).

The Student-Teacher Refuter. Assuming $\mathsf{VPV} \vdash \mathsf{Pal}(n_0)$, for some n_0, we have the following Student-Teacher game interpretation via the KPT Witnessing theorem.

Let φ be the innermost Σ^B_0 formula of $\mathsf{Pal}(n_0)$, and r be the fixed constant many rounds of the Student-Teacher game. A P-Student will take as input 1^n and a machine M. In round one, they will query the Teacher on a string X and witness W_X where it thinks M incorrectly decides X is or isn't a palindrome. The Teacher will respond with a witness W that either shows the machine M correctly accepts the palindrome X on $M(X, W)$, or that the proposed witness W_X actually rejects a non-palindrome X.[5] This is an P-$ST^{CX[\varphi,r]}$ constructive separation for Maass's lower bound.

3.2 Constructive Separations for Palindromes

In order to collapse Student-Teacher games, we will need small amounts of nonuniformity to replace the Teacher's responses. This generalizes the argument of [14] that P-constructive proofs of Maass's lower bound imply breakthrough circuit lower bounds. Here, we will need $\mathsf{P}/o(n^\varepsilon)$-constructivity.

Lemma 3.2 (Lemma 3.3, [14]). There exists a one-tape nondeterministic Turing Machine M running in subquadratic time that acts correctly on all inputs x with circuit complexity $|x|^\delta$, for a fixed $0 < \delta < 1$.

Proof sketch. First, on input x, M will guess a $\log n$-input circuit C_x of size n^δ and evaluate it on all n possible inputs to verify that C_x succinctly represents x. Next, for each $0 \leq i \leq n/2$, M will evaluate C_x on i and $n - i$ and ensure that $C(i) = C(n - i)$. In total, M will run in time $n \cdot n^{O(\delta)} = o(n^2)$ for a sufficiently small constant δ. ☐

Lemma 3.3 (Generalization of Lemma 2.3, [14]). Assume that $\mathsf{P} \subset \mathsf{SIZE}[n^k]$ for some $k \geq 1$. Let $\varepsilon > 0$. Then for any $\mathsf{P}/o(n^\varepsilon)$-algorithm $R(1^n)$ with n output bits, we have that the string $R(1^n)$ has circuit complexity $o(n^\varepsilon)$.

Proof. Assume that $\mathsf{P} \subset \mathsf{SIZE}[n^k]$ for some $k \geq 1$, and let R be a P-algorithm with advice α of length $|\alpha| = o(n^\varepsilon)$ which takes in a unary input 1^n and outputs an n-bit string. For any $\varepsilon' > 0$, we can construct a new $\mathsf{P}/(|\alpha| + O(\log n))$-algorithm R' where R' takes as input $1^{n^{\varepsilon'}}$ and $i \in [n]$ in binary, is given n in binary as advice, and outputs the i-th bit of $R(1^n)$. This is clearly still a polynomial time algorithm, and by the above assumption, has a circuit of size $O(n^{\varepsilon' k} + o(n^\varepsilon))$. Set $\varepsilon' = \varepsilon/2k$ to achieve the desired circuit complexity. ☐

The following is a straightforward generalization of the second item of Theorem 3.4 in [14].

Theorem 3.4. Let $0 < \varepsilon < 1$. A P/n^ε-constructive proof of Maass' bound implies that $\mathsf{P} \not\subset \mathsf{SIZE}[n^k]$.

Proof. Suppose that $\mathsf{P} \subset \mathsf{SIZE}[n^k]$. Then by combining the above two lemmas, there is a one-tape NTM M running in subquadratic time that is correct on all strings which could be output by refuters. This contradicts Maass' lower bound being P/n^ε-constructive. ☐

[5]Note that in the second case, no response from the Teacher is actually needed as a polynomial time Student can check this condition for themselves.

3.3 Round Elimination for the Student-Teacher Refuter

Similar to the round elimination of [8], we show that every query to the counterexample oracle CX can be replaced by a sublinear advice string. There are two new ideas compared to previous work.

(i) Recognize that Teacher in the Student-Teacher refuter is just an NP predicate.

(ii) By assuming (towards a contradiction) that $\mathsf{NP} \subset \mathsf{SIZE}[n^k]$, we can use the Easy Witness Lemma for NP to "compress away" Teacher into sub-linear advive, round-by-round.

Theorem 3.5 (Easy Witness Lemma, [43])**.** Let $k > 0$. Suppose that $\mathsf{NP} \subset \mathsf{SIZE}[n^k]$. Then there is a constant $d > 0$ where for any $\mathcal{L} \in \mathsf{NP}$ and Yes-input X, there is a witness W succinctly represented by a circuit of size n^{dk^3}.

Theorem 3.6. Let r, k be positive integers. Assume that $\mathsf{NP} \subseteq \mathsf{SIZE}[n^k]$. Then an P-$ST^{CX[\varphi,r]}/a(n)$ refuter for Maass implies an P-$ST^{CX[\varphi,r-1]}/a'(n)$ refuter for $a(n) = O(n^\delta)$ with $\delta < 1$ and $a'(n) = C \cdot a(n)^{O(k^3)}$, with $C > 0$ a constant.

Proof. Let M be a nondeterministic Turing machine clocked to run in time $n^{1.1}$, and let d be the constant appearing in Theorem 3.5. First, we note that without loss of generality, the Student will only propose a palindrome to the counterexample oracle. This is because if the Student proposes a non-palindrome, then the oracle response can be compressed to 0 bits and completely removed; the Student can check for itself in linear time[6] that its proposed string X is not a palindrome, and in $n^{1.1}$ time to simulate M on X and the proposed witness W_X.

Let $p \in \{0,1\}^n$ be the first palindrome that the student queries the teacher. As no teacher queries are made yet, p is computable in $\mathsf{P}/a(n)$. Consider the following NP-language $\mathcal{L}_{wit}$:

$$\mathcal{L}^n_{wit} \coloneqq \{x \colon x \in \{0,1\}^{n^{1.1}} \text{ and } M(p,x) = 1\}.$$

Note that $\mathcal{L}^n_{wit}$ is the set of witnesses to the nondeterministic machine W that takes in 1^n as input and a string of length $a(n)$ as advice and decides if p is a 1-input to M. Further, we can pad down the input to 1^{n^ε}, for any constant $\varepsilon > 0$, and add n in binary as advice. Pick $\varepsilon < \delta$. Hence by Theorem 3.5, there exists an $x \in \mathcal{L}^n_{wit}$ that has circuit complexity $(n^\varepsilon + \log n + a(n))^{dk^3} \leq (2a(n))^{dk^3}$. We replace the teacher by instead giving the student this witness circuit at the beginning of the Student-Teacher game. As a result, we change the protocol to have $r-1$ rounds of interaction and $a(n) + (a(n) + n^\delta + \log n)^{dk^3} < (4a(n))^{dk^3}$ bits of advice. ☐

We then have the following corollaries.

Theorem 3.7 (Theorem 1.6)**.** If for any nondeterministic one-tape subquadratic time Turing machine M there is a P-Student-Teacher game $S_M(1^n)$ with counterexample oracle $CX[\varphi_{\mathsf{Pal}}, O(1)]$ solving $\mathrm{Ref}_{\mathsf{Pal},M}$ for n-bit inputs, then $\mathsf{NP} \not\subset \mathsf{SIZE}[n^k]$ for any $k \geq 0$.

Proof. Suppose there is an P-$\mathrm{ST}^{CX[\varphi,r]}$ refuter of constantly many rounds r for Palindromes. Apply Theorem 3.6 to remove the first teacher query, adding n^ε bits of advice, for any $\varepsilon > 0$ we desire. Pick $\varepsilon < 1/\left(100dk^3\right)^{2r}$. Repeatedly apply Theorem 3.6 another $r-1$ times to have a $\mathsf{P}/o(n^{1/100})$ refuter, we contradict Theorem 3.4. ☐

Theorem 3.8 (Theorem 1.11)**.** If $\mathsf{VPV} \vdash \mathsf{Pal}(n_0)$, for any $n_0 > 0$, then $\mathsf{NP} \not\subset \mathsf{SIZE}[n^k]$.

Proof. Suppose $\mathsf{VPV} \vdash \mathsf{Pal}(n_0)$ and that $\mathsf{NP} \subset \mathsf{SIZE}[n^k]$ for some $k > 0$. Then by the KPT-witnessing theorem, we get an P-$\mathrm{ST}^{CX[\varphi,r]}$ refuter of constantly many rounds r. Applying Theorem 3.7, we have a contradiction. ☐

[6]Student need not be a one-tape TM, so checking PALINDROME can indeed be linear time.

4 Existence of K^t-Random Strings

Hirahara's lower bound $\mathsf{R}_{\mathsf{K}^t} \notin \mathsf{P}$ for $t = \mathsf{qpoly}$ is **unconditionally** non-constructive [22, 14]. Could we extract a related unprovability result for VPV? Non-constructivity was established by using assumed P-refuters to print high-K^t strings for $t = \mathsf{qpoly}$ in only poly-time — a contradiction [14]. This suggests to begin studying VPV-provability of the lower bound "$\mathsf{R}_{\mathsf{K}^t} \notin \mathsf{P}$" by considering first the simpler statement "there exist K^t-random strings," abbreviated informally as $\exists \mathsf{R}_{\mathsf{K}^t}$ below.

Even $\exists \mathsf{R}_{\mathsf{K}^t}$ requires some care to express in VPV. Straightforward (i.e., without padding) translation of $\exists \mathsf{R}_{\mathsf{K}^t}$ into VPV with $t = \mathsf{qpoly}$ is impossible, because VPV-number-terms must have fixed polynomial growth. So we study instead provability of a sequence of statements asserting that high-K^{n^c} strings exist: "for sufficiently large n, there is an n-bit string X with $\mathsf{K}^t(X) > n/2$" where $t = n^c$ for each c.

Formalization 4.1 (HiK^t for VPV). Fixing n_0, define the following sequence of VPV sentences for each $c \in \mathbb{N}$.

$$\mathsf{HiK}^t[c] := \forall n.(n > n_0)\ \exists X.(|X| = n)\ \forall D.(|D| < n/2)\ \mathsf{run}(\pi_1(D),\ \pi_2(D),\ n^c) \neq X$$

Remark 4.2. The symbol c is *not* a free variable in a VPV-formula called HiK^t. It is rather the parameter of a sequence of formulas where " n^c " abbreviates the constant-length term $\underbrace{n \cdot n \cdot n \cdot \cdots \cdot n}_{c \text{ occurences of } n}$.

Fixing sufficiently large n_0, each statement $\mathsf{HiK}^t[c]$ is true in the standard model by simple counting.[7] Furthermore, the argument is essentially identical for each c, differing only by a substitution of numeric terms. Can VPV carry it out? Can VPV prove HiK^t via a "uniform" argument, such that the proofs for $\mathsf{HiK}^t[c]$ and $\mathsf{HiK}^t[c']$ with $c \neq c'$ have a clean quantitative relationship as syntactic objects?

We make some progress towards answering these questions about provability of the schema $\mathsf{HiK}^t[c]$ by giving lower bounds on Student-Teacher search for K^t-random strings for each fixed $t \in \mathsf{poly}$ (Section 4.1) and proof-theoretic hypotheses under which these lower bounds imply unprovability (Section 4.3).

4.1 Student-Teacher-Search Lower Bounds for K^t-Random Strings

First we derive a sequence of search problems from the schema HiK^t as described in Section 2.6. Extract the quantifier-free part of $\mathsf{HiK}^t[c]$ for each c as:

$$\psi_c(n, X, D) := (|D| \leq n/2 \wedge n > n_0) \to (\mathsf{run}(\pi_1(D),\ \pi_2(D),\ n^c) \neq X \wedge |X| = n)$$

Because $\mathsf{HiK}^t[c]$ is true in the standard model for every c, the problem Search_{ψ_c} is total and well-defined for every c. To ease notation, we spell out and abbreviate these search problems below.

Definition 4.3 (Search for K^t-Random Strings)**.** For each $c \in \mathbb{N}$, abbreviate the problem Search_{ψ_c} by

$$\exists \mathsf{HiK}^t[c] := \{(1^n, X)\ \mid\ \mathsf{K}^{n^c}(X) > n/2 \wedge |X| = n\}$$

An answer to the counterexample query X for $\exists \mathsf{HiK}^t[c]$ is binary string D that is

1. *short,* so $|D| < n/2$ and
2. *describes* X*,* so $D = \langle M, A \rangle$ with M run on input A for at most n^c steps halts with X on the tape.

Any such D is a valid counterexample to the claim "$\mathsf{K}^{n^c}(X) \geq n/2$." Having fixed terminology, we are ready to state and prove our lower bounds aginst Student-Teacher search for K^t-random strings.

The base case — Students that make no queries — is implicit in Proposition 1.8 of [14]. Generalizing the "indexing template" embedded in that proof yields our construction. Their argument is paraphrased below.

Proposition 4.4. For $c \geq 1$, no student running in time $\tilde{O}(n^c)$ and making zero queries solves $\exists \mathsf{HiK}^t[c+1]$.

[7] The constant n_0 need only be large enough to ensure that $\mathsf{run}(\pi_1(D), \pi_2(D), n_0^c)$ is well-defined for $|D| \geq n_0/2$. Thus n_0 can be fixed to an absolute constant depending only on the machine and pair encoding implicit in the run and π functions.

Proof. Suppose S is a student that runs in time $\tilde{O}(n^c)$ and solves $\exists\mathsf{HiK}^t[c+1]$ without making any queries. Denote by ℓ the description length of S, fix arbitrary $n \in \mathbb{N}$, and let $x_n = S(1^n)$ be the n-bit $K^{n^{c+1}}$-random string found by S. Define the *indexing* of S to be the standard, one-tape Turing Machine $\mathsf{ix}(S)$ that results from substituting S into the Indexing Template (Algorithm 1). Because S makes no queries, it can indeed be simulated by a standard one-tape Turing Machine.

By construction, $\mathsf{ix}(S)$, given input n encoded in binary, prints x_n. This takes $\tilde{O}(n^c)$ steps for a larger $\mathsf{polylog}$ factor than in the original runtime of S, accounting for time to print 1^n onto the worktape and to run $S(1^n)$. The description length of $\mathsf{ix}(S)$ is just $\ell + a$ for an absolute constant a depending on the universal machine and book-keeping code to expand the binary representation of n into 1^n. Therefore, the pair $\langle \mathsf{ix}(S),\ \mathsf{bin}(n)\rangle$ witnesses $\mathsf{K}^{n^{c+1}}(x_n) \leq 2(\ell + a) + \log n + 2$ — a contradiction for sufficiently large n. ☐

Algorithm 1 Indexing Template $\mathsf{ix}(S)$

Parameters S the description of a Turing machine

```
On input bin(n)
output S(1^n)
```

Observe that $a_n = \langle \mathsf{ix}(S),\ \mathsf{bin}(n)\rangle$ is a uniform counterexample to the claim "x_n is a K^t random string" for any zero-query student and sufficently large n. This suggests that even if a student S for $\exists\mathsf{HiK}^t$ does make queries, the description of S could be used to answer and eliminate them. Two-parameter indexing — tracking both n and number of queries made by $S(1^n)$ — suffices to realize this intuition (Algorithm 2).

Theorem 4.5. For $c \geq 1$, no student running in time n^c solves $\exists\mathsf{HiK}^t[2c+1]$.

Proof. Suppose S is a student of description length ℓ running in time n^c that solves $\exists\mathsf{HiK}^t[2c+1]$ using at most $r(n) < n^c$ queries. By Proposition 4.4, it is immediate that $r(n) \geq 1$. We will eliminate these queries by constructing a uniform sequence of valid answers — derived from S itself — that are easy to produce without a teacher. Before arguing for validity, we show that such a "reflection exchange" of answers and queries is well-defined and establish some basic properties (Claim 4.6).

More precisely, to generate counter-examples for S from the description of S, we must convert S into a standard, one-tape Turing machine (TM) — because $\exists\mathsf{HiK}^t$ is defined with respect to this *particular* model of computation. The *Reflection Template* transforms any student S into a standard Turing machine $\mathsf{rf}(S)$ by substituting the description of S into Algorithm 2 below. We must additionally handle the change in computational model from the Student, as an oracle Turing machine, to a standard one-tape Turing machine. For each standard one-tape Turing machine M, write $\ulcorner M \urcorner$ for the binary encoding of M induced by the *particular* universal machine used to define $\exists\mathsf{HiK}^t$. We can now state

Claim 4.6. There is a standard one-tape Turing machine $\mathsf{rf}(S)$ such that, fixing the sequence of answers $a_{n,j} = \langle \ulcorner \mathsf{rf}(S) \urcorner,\ \langle \ulcorner \mathsf{rf}(S) \urcorner,\ \langle \mathsf{bin}(n),\ \mathsf{bin}(j)\rangle\rangle\rangle$ and denoting by $q_{n,i}$ the induced sequence of queries $q_{n,i} =$ "the i-th query made by $S(1^n)$ after getting $a_{n,j}$ in response to the j-th query for $j \in \{1, \ldots, (i-1)\}$," the following properties hold:

1. $\mathsf{rf}(S)$ on input $\langle \ulcorner \mathsf{rf}(S) \urcorner,\ \langle \mathsf{bin}(n),\ 1\rangle\rangle$ prints the first query made by $S(1^n)$.
2. $\mathsf{rf}(S)$ on input $\langle \ulcorner \mathsf{rf}(S) \urcorner,\ \langle \mathsf{bin}(n),\ \mathsf{bin}(i)\rangle\rangle$ prints $q_{n,i}$.
3. $\mathsf{rf}(S)$ runs in time $O(\ell + n^{2c}\log n)$ on all inputs of the form $\langle \ulcorner \mathsf{rf}(S) \urcorner,\ \langle \mathsf{bin}(n),\ \mathsf{bin}(j)\rangle\rangle$.
4. The description length of $\mathsf{rf}(S)$ is $\ell + a_{\mathsf{rf}}$ for some absolute constant a_{rf}.

Proof. Observe that "running $\mathsf{rf}(S)$ on appropriate inputs" is exactly a constructive definition of the queries $q_{n,i}$ for each n and $i < r(n)$. All claimed properties follow by inspection and simulation of $\mathsf{rf}(S)$ because S is deterministic and time-bounded. The runtime blow up from $O(n^c)$ to $O(n^{2c}\log n)$ occurs due to the treatment of the oracle tape as a 2nd tape, and simulating it on the first via a standard two-to-one tape simulation [3]. None of these assertions are about the *validity* of answers $a_{n,j}$ as responses to queries $q_{n,i}$ — they assert only that both sequences are well-defined and can be obtained in bounded time by running and manipulating the description of $\mathsf{rf}(S)$. ☐

Algorithm 2 Reflection Template $\mathsf{rf}(S)$

Parameters S a student

1: On input $\langle D, \langle \mathsf{bin}(n), \mathsf{bin}(q)\rangle\rangle$
2: $i \leftarrow 1$ ▷ *assumption: S makes at least one query*
3: **loop**
4: $q_{n,i} \leftarrow$ Simulate $S(1^n)$ until it queries teacher
5: **if** $i < q$ **then**
6: Answer the simulated query $q_{n,i}$ with $\langle D,\ \langle D,\ \langle \mathsf{bin}(n),\ \mathsf{bin}(i)\rangle\rangle\rangle$ ▷ *exactly $a_{n,i}$ when $D = \ulcorner \mathsf{rf}(S) \urcorner$*
7: $i \leftarrow i+1$
8: **else**
9: **break** the loop
10: **output** $q_{n,i}$ ▷ *the last query from simulated $S(1^n)$*

To eliminate $q \geq 1$ queries from S, we answer them with a description of the reflection template applied to S — the standard machine $\mathsf{rf}(S)$. The *Autodidact Template* transforms any student S making at most $r(n)$ queries into a student $\mathsf{ad}(S,q)$ making at most $r(n)-q$ queries by substituting the description of S and $\mathsf{bin}(q)$ into Algorithm 3 below. Preservation of correctness and runtime guarantees is

Claim 4.7. Student $\mathsf{ad}(S,q)$ runs in time $O(n^{2c}\log n)$ and solves $\exists\mathsf{HiK}^t[2c+1]$ using at most $r(n)-q$ queries.

Proof. We argue by induction, showing first that student $\mathsf{ad}(S,1)$ solves $\exists\mathsf{HiK}^t[2c+1]$ within the claimed runtime and makes at most $r(n)-1$ queries. Consider the set of first queries $q_{n,1}$ asked by $S(1^n)$ for each n. These strings depend only on S and n — so intuitively, their $\mathsf{K}^{n^{2c+1}}$-complexity is bounded. Formally, the machine $\mathsf{rf}(S)$ on input $\langle\ulcorner\mathsf{rf}(S)\urcorner, \langle\mathsf{bin}(n), 1\rangle\rangle$ prints $q_{n,1}$ for each n in at most $O(n^c\log n)$ steps (items 1 and 3 of Claim 4.6). Therefore, the machine-input pair

$$\langle\ulcorner\mathsf{rf}(S)\urcorner,\ \langle\ulcorner\mathsf{rf}(S)\urcorner,\ \langle\mathsf{bin}(n),\ 1\rangle\rangle\rangle = a_{n,1}$$

of length $O(\ell)+O(\log n)$ witnesses $\mathsf{K}^{n^{c+1}}(q_{n,1}) < n/2$ for all sufficiently large n. Thus, for sufficiently large n, the string $a_{n,1}$ supplied to $S(1^n)$ by line 6 of $\mathsf{ad}(S,1)$ is a valid answer to query $q_{n,1}$. By the assumption that S solves $\exists\mathsf{HiK}^t[2c+1]$, it must produce an element of $R_{\mathsf{K}^{n^{c+1}}}$ given *any* sequence of valid answers from teacher of length at most $r(n)$. Therefore, the simulation of $S(1^n)$ executed by $\mathsf{ad}(S,1)$ will solve $\exists\mathsf{HiK}^t[2c+1]$ using at most $r(n)-1$ queries to a real teacher, because $a_{n,1}$ is a valid answer to $q_{n,1}$. Accounting for the time complexity of simulation and string manipulation, $\mathsf{rf}(S,1)$ takes at most $O(n^{2c}\log n)$ steps on inputs 1^n. This concludes the base case.

For the inductive step, suppose that student $\mathsf{ad}(S,i)$ solves $\exists\mathsf{HiK}^t[2c+1]$ using at most $r(n)-i$ queries. Inspecting the autodidact template we have that, when running $\mathsf{ad}(S,i)$: (1) all queries made by S until the $(i+1)$-th query are answered by $a_{n,j}$ for $j \in \{1,\ldots,i\}$ and (2) query $q_{n,(i+1)}$ is the *first* query answered by teacher. Because $\mathsf{ad}(S,i)$ is a student solving $\exists\mathsf{HiK}^t[2c+1]$, it must produce an element of $R_{\mathsf{K}^{n^{2c+1}}}$ given *any* sequence of valid answers from teacher of length at most $r(n)-i$. We argue that $a_{n,(i+1)}$ is a valid answer to query $q_{n,(i+1)}$.

The standard, one-tape machine $\mathsf{rf}(S)$ on input $\langle\ulcorner\mathsf{rf}(S)\urcorner,\ \langle\mathsf{bin}(n),\ \mathsf{bin}(i+1)\rangle\rangle$ prints $q_{n,(i+1)}$ in at most $O(\ell + n^c\log n)$ steps (items 2 and 3 of Claim 4.6). Therefore, the machine-input pair

$$\langle\ulcorner\mathsf{rf}(S)\urcorner,\ \langle\ulcorner\mathsf{rf}(S)\urcorner,\ \langle\mathsf{bin}(n),\ \mathsf{bin}(i+1)\rangle\rangle\rangle = a_{n,(i+1)}$$

of length at most $O(\ell)+O(\log n)+O(\log r(n))$ (by item 4 of Claim 4.6) witnesses $\mathsf{K}^{n^{2c+1}}(q_{n,(i+1)}) < n/2$ for all sufficiently large n, because we know $r(n) < n^c$ from the runtime bound of S. Therefore, student $\mathsf{ad}(S,i+1)$ correctly simulates one additional teacher response for S compared to $\mathsf{ad}(S,i)$ and so solves $\exists\mathsf{HiK}^t[2c+1]$ using at most $r(n)-(i+1)$ queries. Induction on i now proves Claim 4.7. ☐

Now conclude the proof of Theorem 4.5 by substituting $q = r(n)$ into Claim 4.7 to get that $\mathsf{ad}(S,q)$ solves $\exists\mathsf{HiK}^t[2c+1]$ using zero queries in $\tilde{O}(n^{2c})$ time, contradicting Proposition 4.4. ☐

Algorithm 3 Autodidact Template $\mathsf{ad}(S, q)$

Parameters $q \in \mathbb{N}$ and S a student

On input 1^n
$i \leftarrow 1$ ▷ *assumption: reflect at least one query*
loop
  $q_{n,i} \leftarrow$ Simulate $S(1^n)$ until it queries teacher
  if $i \leq q$ **then**
    Answer the simulated query $q_{n,i}$ with $a_{n,i} = \langle \ulcorner \mathsf{rf}(S) \urcorner,\ \langle \ulcorner \mathsf{rf}(S) \urcorner,\ \langle \mathsf{bin}(n),\ \mathsf{bin}(i) \rangle \rangle \rangle$
    $i \leftarrow i + 1$ ▷ *increment #queries reflected*
  else
    break the loop
Continue simulating $S(1^n)$ but answer all subsequent queries by asking teacher
output the output of simulated $S(1^n)$

4.2 Gap Between Student-Teacher Search Lower Bounds & VPV-Unprovability

The Student-Teacher search lower bounds above do not suffice to obtain VPV-unprovability. Suppose VPV proves $\mathsf{HiK}^t[c]$ for every c. Applying KPT-witnessing, we would obtain for every c a $\mathsf{DTIME}[q_c]$ Student-Teacher search solving $\exists\mathsf{HiK}^t[c]$, for some arbitrary polynomial q_c. There is no contradiction to Theorem 4.5, because it does not control the relationship between n^c and q_c. However, if $q_c = o(n^{c/2})$ could be guaranteed for even a single c, then unprovability of the HiK^t schema in VPV would follow.

One way forward is to make a stronger assumption about the supposed VPV-proofs of $\mathsf{HiK}^t[c]$. In larger theories than VPV that are known to prove $\mathsf{HiK}^t[c]$ for each c, the proofs are *uniform* — essentially the same for each c. An assumption like "VPV proves $\mathsf{HiK}^t[c]$ for each c and furthermore the proofs are structurally uniform" could enable control over Student runtime, such that a *single* polynomial-time algorithm witnesses K^t-random strings for *every* $t \in \mathsf{poly}$.

Such dramatic consequences of uniform proofs might seem unrealistic; the term n^c appears in the quantifier-free part of $\mathsf{HiK}^t[c]$, so shouldn't any student witnessing $\mathsf{HiK}^t[c]$ take time at least n^c? This appealing but flawed intuition presumes that witnessing requires simulation of an n^c-time machine. In reality, Teacher may be the only party responsible for an n^c-time computation — it depends on the scheme. In Section 4.5 we give several examples of VPV schemata $\Phi[c]$ parameterized by arbitrary polynomial time bounds n^c — with quantifier prefix identical or similar to $\mathsf{HiK}^t[c]$ — where both (1) each statement is provable in VPV for every c by the "same" proof and (2) witnessing the statement takes **absolute** polynomial time — **not** n^c for each c.

Summarizing the above, it is both plausible and well-motivated to ask for better control over the complexity of witnessing terms when VPV proves a parameterized sequence of theorems by "essentially the same" proof. This requires a definition of uniform proofs. Towards this end, we discuss next a similar question about Peano Arithmetic (PA) and extract a witnessing hypothesis for uniform VPV-proofs by analogy.

4.3 Kriesel's Conjecture & Witnessing Hypotheses for Uniform Proofs

A fundamental question about "merging" a sequence of theorems into a single theorem appeared in 1975 as Problem 34 on Friedman's list of One Hundred and Two Problems in Logic, attributed to Kriesel [21]. For some theories, a positive answer to this question would imply uniform witnessing.

Conjecture 4.8 (Kriesel's Conjecture, §4.4 of [49])**.** Suppose for a formula $\varphi(x)$ and a number k, one can prove $\varphi(S^c(0))$ in Peano Arithmetic using $\leq k$ steps for every c. Then $\forall c \varphi(c)$ is provable in Peano Arithmetic.

Efforts to resolve Kriesel's Conjecture (KC) uncovered a peculiar situation: KC is very sensitive to how PA is axiomatized! For example, KC is true when PA is axiomatized with a ternary relation for multiplication [46, 41] or with minimality instead of induction [23]. But KC is false when PA has a function symbol for subtraction [24], and remains open for the "textbook" presentation of PA using function symbols $\{S, +, \times\}$. Hrubeš discusses these issues in detail [23].

Let PA_L denote the theory of Peano Arithmetic with symbols for every primitive recursive function, axiomatized by a list L of formulas. If KC is true for PA_L, then it is straightforward to extract parameter-independent witnessing terms from a sequence of proofs: just apply KC followed by KPT witnessing. This interchanges the order of quantifiers as desired: a **single** sequence of witnessing terms that works **for all** sentences in the schema. One intermediate step is required — PA is not a universal theory, and so KPT does not apply directly. We work out the details below for $\exists\mathsf{K}^t\mathsf{R}$, towards developing a uniform witnessing hypothesis for the weaker theory VPV by analogy. First recall the KPT theorem, stated below for single-sorted theories.

Theorem 4.9 (Single-Sorted KPT)**.** Let T be a universal theory with vocabulary L. Let φ be an open L-formula, and suppose that $T \vdash \forall\vec{x}\ \exists y\ \forall z\ \varphi(\vec{x}, y, z)$. Then there is a finite sequence $t_1, \ldots, t_r$ of L-terms such that

$$T \vdash \forall\vec{x}\ \forall z_1, \ldots z_r[\varphi(\vec{x}, t_1(\vec{x}), z_1) \vee \varphi(\vec{x}, t_2(\vec{x}, z_1), z_2) \vee \cdots \vee \varphi(\vec{x}, t_r(\vec{x}, z_1, \ldots, z_{r-1}), z_r)]$$

Now translate "for every c and almost every n, there exists a K^{n^c}-random string of length n" into a PA-formula. Because PA-terms are not bounded by polynomials, here we *can* admit the runtime exponent c as a *free variable.* Fixing sufficiently large n_0, define

$$\mathsf{HiK}^t(c) := \forall n.(n > n_0)\ \exists x.(x < 2^n)\ \forall d.(d < 2^{n/2})\ \mathsf{run}(\pi_1(d),\ \pi_2(d),\ \mathsf{unary}(\mathsf{exp}(n, c))) \neq x$$

For every reasonable list of axioms L, if PA_L includes all primitive recursive functions, it includes the necessary function symbols and proves their relevant properties.

- $\mathsf{unary}(w)$ is the PA_L symbol for the function that outputs z such that $\mathsf{bin}(z) = 1^w$, and
- $\mathsf{exp}(n, c)$ is the PA_L symbol for the exponentiation function n^c.
- run is the PA_L function symbol for run_U from the definition of time-bounded Kolmogorov complexity,
- $\pi_1(z)$ and $\pi_2(z)$ are the PA_L symbols for the pair decoding functions (see Section 2.7).

Again, this translation of $\exists\mathsf{K}^t\mathsf{R}$ exploits the power of PA to admit c as a variable of the object language.

Proposition 4.10. Suppose KC is true for PA_L and there exist absolute constants n_0 and k such that one can prove $\mathsf{HiK}^t(S^c(0))$ in PA_L using $\leq k$ steps for every c. Then, letting PA'_L be any universal conservative extension of PA_L, there is a finite sequence of PA'_L-terms $q_1, \ldots q_r$ such that

$$\begin{aligned}\mathsf{PA}'_L \vdash \forall c\ \forall n.(n > n_0)\ \forall d_1, \ldots, d_r\ \Big[&(\mathsf{run}(\pi_1(d_1),\ \pi_2(d_1),\ \mathsf{unary}(\mathsf{exp}(n, c))) \neq q_1(n, c)) \vee \\ &(\mathsf{run}(\pi_1(d_2),\ \pi_2(d_2),\ \mathsf{unary}(\mathsf{exp}(n, c))) \neq q_2(n, c, d_1)) \vee \\ &\ldots \vee \\ &(\mathsf{run}(\pi_1(d_r),\ \pi_2(d_r),\ \mathsf{unary}(\mathsf{exp}(n, c))) \neq q_r(n, c, d_1, \ldots d_{r-1}))\Big]\end{aligned}$$

Proof. Assume that PA_L proves HiK^t as in the statement of the lemma, and KC is true of PA_L. Applying KC, we have $\mathsf{PA}_L \vdash \forall c\ \mathsf{HiK}^t(c)$ for some absolute constant n_0. Now let PA'_L be any universal conservative extension of PA_L. Because PA'_L extends PA_L, we also have $\mathsf{PA}'_L \vdash \forall c\ \mathsf{HiK}^t(c)$. Because PA'_L is universal, appeal to KPT witnessing (Theorem 4.9) concludes this proof. ☐

4.4 Conditional Unprovability of $\mathsf{HiK}^t[c]$ in VPV and V^1

By analogy to the outcome of assuming KC and applying KPT to a conservative universal extension of PA, introduce the following

Hypothesis 4.11 (Witnessing for Linecount-Uniform VPV-Proofs)**.** Let $\varphi(n, p, X, Y)$ be a $\Sigma_0^B(\mathsf{VPV})$ formula with all free variables displayed. Suppose there is an absolute constant n_0 , number k, and VPV-term t such that one can prove $\forall n.(n > n_0)\ \exists X.(|X| < t(n))\ \forall Y \varphi(n, n^c, X, Y)$ in VPV using $\le k$ steps for every c. Then there is a finite sequence $F_1, \ldots F_r$ of VPV-function symbols that are *absolutely witnessing*:

$$\text{for every } c,\ \mathsf{VPV} \vdash \forall n.(n > n_0)\ \forall Y_1, \ldots, Y_r \Big[\varphi(n, n^c, F_1(n, c), Y_1) \vee \varphi(n, n^c, F_2(n, c, Y_1), Y_2) \vee \cdots \vee \varphi(n, n^c, F_r(n, c, Y_1, \ldots Y_{r-1}), Y_r)\Big]$$

The asymmetry in how c is given to φ compared to how c is given to each F_i — n^c vs. c — is crucial for our applications. If Hypothesis 4.11 holds, then any student derived from the hypothesis takes arguments 1^n and 1^c because numeric terms are supplied in unary for two-sorted complexity classes (see Section 2.4). If c were instead given to F_i as n^c, the implicit student would take $\mathsf{poly}(n^c)$ time to print K^{n^c}-random strings of length n — and no contradiction would arise. However, combining Hypothesis 4.11 with the Student-Teacher lower bounds for $\exists \mathsf{HiK}^t$ from the last section (Theorem 4.5), we have

Corollary 4.12. Under the Witnessing Hypothesis for Linecount-Uniform VPV-Proofs, there is no fixed k such that one can prove $\mathsf{HiK}^t[c]$ in VPV using $\le k$ steps for each c.

This would rule out *linecount uniform* proofs of $\mathsf{HiK}^t[c]$. However, linecount uniformity — though well-motivated by Kriesel's Conjecture — is certainly not the only reasonable notion of uniformity in proofs. We hope that a deeper understanding of uniform VPV-proofs will emerge by studying witnessing hypotheses that emphasize different aspects of common structure in theorems and proofs. To begin the investigation, we introduce a strong witnessing hypothesis that emphasizes the common element in statements like $\mathsf{HiK}^t[c]$ — substitution of polynomial time-bounds into the execution of Turing machines, formalized as

Definition 4.13 (poly-Runtime Schema)**.** Fix a universal function symbol $\mathsf{run}(M, A, s)$ to output the tape of machine M run on input A for s steps. An infinite sequence of formulas Φ is a *poly-runtime schema* if Φ is obtained by taking an infinite union over substitution of polynomial runtimes. Formally, let φ be a formula with a free variable p occuring only in terms of the form $\mathsf{run}(M, A, p)$ — as the time bound. Then,

$$\Phi = \bigcup_{c \in \mathbb{N}} \varphi(p/n^c)$$

We refer to the c-th sentence in such a schema by Φ_c.

Hypothesis 4.14 (Witnessing for poly-Runtime Schema in VPV)**.** Suppose Φ is a poly-runtime schema with $\varphi = \forall n.(n > n_0)\ \exists X.(|X| < t(n))\ \forall Y \psi(n, p, X, Y)$ for ψ a $\Sigma_0^B(\mathsf{VPV})$ formula and t a VPV-term, and there is an absolute constant n_0 such that $\mathsf{VPV} \vdash \Phi$. Then there is a finite sequence $F_1, \ldots F_r$ of VPV-function symbols that are *absolutely witnessing*:

$$\text{for infinitely many } c,\ \mathsf{VPV} \vdash \forall n.(n > n_0)\ \forall Y_1, \ldots, Y_r \Big[\psi(n, n^c, F_1(n, c), Y_1) \vee \psi(n, n^c, F_2(n, c, Y_1), Y_2) \vee \cdots \vee \psi(n, n^c, F_r(n, c, Y_1, \ldots Y_{r-1}), Y_r)\Big]$$

The conclusion is essentially identical to that of the linecount WHUP. However Hypothesis 4.14 is much stronger: it asserts that VPV cannot help but give absolute witnessing if it proves a poly-runtime schema. Therefore, combining Hypothesis 4.14 with the Student-Teacher lower bounds for $\exists \mathsf{HiK}^t$ from the last section (Theorem 4.5), we have

Corollary 4.15. Under the Witnessing Hypothesis for poly-Runtime Schemas in VPV, there are infinitely many c such that VPV does not prove $\mathsf{HiK}^t[c]$.

Under Hypothesis 4.14, we get VPV-unprovability of $\mathsf{HiK}^t[c]$, but not V^1 unprovability. This under-exploits our Student-Teacher lower bounds for $\exists\mathsf{HiK}^t$, which can eliminate poly-many rounds from Student. So, we introduce an appropriate WHUP for V^1 — derived from the KPT Theorem for V^1 (Theorem 2.15).

Hypothesis 4.16 (Witnessing for poly-Runtime Schema in V^1)**.** Suppose Φ is a poly-runtime schema with $\varphi = \forall n.(n > n_0)\ \exists X.(|X| \leq t(n))\ \forall Y \psi(n, p, X, Y)$ for ψ a $\Sigma_0^B(\mathsf{V}^1)$ formula and t a V^1-term, and there is an absolute constant n_0 such that $\mathsf{V}^1 \vdash \Phi$. Then there is an *absolutely witnessing* FP function F such that for infinitely many c,

$$\mathbb{N}_2 \models \forall n.(n > n_0) \forall Y \psi(n, n^c, F^{CX[\Phi_c]}, Y)$$

Corollary 4.17. Under the Witnessing Hypothesis for poly-Runtime Schemas in V^1, there are infinitely many c such that V^1 does not prove $\mathsf{HiK}^t[c]$.

These two corollaries imply separations with Jeřábek's theory VAPC.

Theorem 4.18. $\mathsf{VAPC} \vdash \mathsf{HiK}^t[c]$, for all $c \in \mathbb{N}$. Further, under Witnessing Hypotheses, $\mathsf{VPV} \nvdash \mathsf{HiK}^t[c]$ and $\mathsf{V}^1 \nvdash \mathsf{HiK}^t[c]$.

Proof. It was shown by Korten [28] that $\mathsf{VAPC} \vdash \mathsf{HiK}^t[c]$. □

We spend the remainder of this section addressing the plausibility of these hypotheses, by giving examples of VPV-theorems that do enjoy absolute witnessing despite varying polynomial bounds.

Remark 4.19. It is interesting to note that our arguments do not distinguish between poly-Runtime Schemas and runtime schemas of higher time complexity (such as quasipolynomial time). Let $p_b(n) = n^{(\log n)^b}$. Then we can consider the two-sorted variant of S_3^1 and study the provability of the schema $\mathsf{HiK}^{p_b(n)}$. Our WHUPs and round collapse techniques would extend to this setting, separating S_3^1 from the theory corresponding to the quasipolynomial form of VAPC.

4.5 Examples of Schemata With "Uniform" Proofs & Absolute Witnessing

The WHUPs discussed in this section apply to VPV-schemata of the form

$$\Phi[c] := \forall n.(n > n_0)\ \exists X\ \forall Y \varphi(n^c, X, Y)$$

where $\varphi(p, X, Y)$ is Σ_0^B for each $c \in \mathbb{N}$. Here we give examples of simple VPV-theorems to illustrate that this class of schemata is non-trivial. All these examples have both proofs that are identical up to numeric substitutions and witnessing algorithms that run in some **absolute** polynomial time — **not** n^c for each c. Therefore, no contradiction can arise from assuming a WHUP (and constant-line proofs) for any of these theorems. The WHUP would just "automatically" transform proofs into witnessing algorithms that meet known complexity upper bounds. The common element in all these examples is efficient transformation of encoded Turing Machines. For each example we describe the VPV-translation and carefully discuss the complexity of witnessing. We do not argue for VPV-provability, because all these theorems follow from properties of universal machines and lemmas about efficient string manipulation that are readily available in VPV — see the discussion in Sections 2.1 and 4 of [48].

4.5.1 Machine Templates

The first three examples give basic properties of machine-only Kolmogorov complexity. Fix a universal Turing machine U and define the *machine-only t-time bounded Kolmogorov Complexity* $\mathrm{mo}\mathsf{K}_U^t(x)$ *of a string* x as the length of the shortest encoded machine that prints x when simulated by U:

$$\mathrm{mo}\mathsf{K}_U^t(x) = \min_{d \in \{0,1\}^*} \{|d| \ :\ U(d,\ \varepsilon,\ 1^{t(|x|)}) = x\}$$

This definition is brittle compared to standard time-bounded Kolmogorov complexity. The UTM never provides any input to the encoded machine d, forcing d to "hardcode" useful strings instead of reading them from an input tape. Therefore the basic fact about K^t — $\forall x \mathsf{K}^t(x) < |x| + a$ for an absolute constant a — fails. However, we can recover something similar for moK^t, even in VPV: an uniform upper bound on mo$\mathsf{K}^t(x)$ for every x.

Memorization Templates. For every polynomial time bound t, for every string length n, there is a hardcoded-string "template" machine M of length n, such that any string X of "sufficiently smaller" length can be pasted into the template to produce a new machine M'. The machine M' prints X in less than t time. Pasting is a polynomial-time string function that copies the bits of Y into a sequence of states of M. We formalize this as a VPV-schema below, varying the polynomial time bound.

$$\mathsf{MEMT}[n_0, c] := \forall n.(n > n_0)\ \exists M.(|M| = n)\ \forall X.(|X| \leq n/16)\ \mathsf{run}_U(\mathsf{paste}_U(M, X), n^c) = X$$

VPV cannot quantify over arbitrary polynomial time bounds, but it can prove the MEMT schema via an essentially-identical proof for each c. However, no contradiction can arise from a WHUP because it is easy to witness M: print the U-encoding of a Turing Machine that prints an *explicit* all-zero string instead of an *implicit* all-zero string. That is, the ith state of M is "write 0 to the tape, move the head right, transition to state i+1." The paste function replaces the "write 0" element of state i of M with bit $X(i)$. The content of this simple theorem is the gap between n and $|X|$ — it asserts an upper bound on the cost of memorizing a string relative to some fixed model of computation and encoding of machines shared by run_U and paste_U.

Notice that witnessing M in this example takes linear time completely independant of c. This is more restrictive than the consequences of a WHUP, which allows witnessing algorithms to take 1^c as an argument. Our next example actually exploits this dependence.

Clocking Templates. For every polynomial time bound p, for each sufficiently large n, a "template" machine M of length n enforces a p-step timeout on shorter machines, making sure they halt in time p and signalling a fault if they run too long. We'll formalize this in VPV using a pair encoding function: the clocking template applied to machine description D outputs $\langle h, \mathsf{run}(D, \varepsilon, n^c)\rangle$ where h is 1 if D halted within n^c steps and zero otherwise. Consider the following collection of VPV-theorems $\mathsf{CLOCKT}[n_0, c] :=$

$$\forall n.(n > n_0)\ \exists M.(|M| = n)\ \forall D.(|X| \leq n/16)(\mathsf{halt}(D, \varepsilon, n^c) \to \mathsf{run}(\mathsf{paste}(M, D), \varepsilon, n^{2c}) = \langle 1, \mathsf{run}(D, \varepsilon, n^c)\rangle) \\ \wedge (\neg\mathsf{halt}(D, \varepsilon, n^c) \to \mathsf{run}(\mathsf{paste}(M, D), \varepsilon, n^{2c}) = \langle 0, \mathsf{run}(D, \varepsilon, n^c)\rangle)$$

Once again, there is a straightforward witnessing for M: print the U-encoding of a machine that *explicitly* prints the all-zero string Z of length $n/16$ to the worktape (as in the memorization template), and then runs a n^c-clocked U to simulate Z. Pair the worktape contents of the results with 0 or 1 depending on if Z halted. The paste function then replaces the explicitly-coded Z with the encoding of D, resulting in a template with the desired behaviour.

Witnessing this template actually depends on c: the clock requires $c\log(n)$ hardcoded bits in the description of M. However, this dependence is *not* polynomial: for sufficiently large n, $c\log(n) < n$. Inspecting the WHUPs for VPV (Hypotheses 4.11, 4.14) we see that the witnessing function symbols occur as $F(n, c, \dots)$, meaning that n and c are given in unary to the witnessing algorithm. Therefore, in fixed $\mathsf{poly}(n, c)$ time we can hardcode the binary representation of n^c into a clock. This is an example where the straightforward witnessing has *exactly* the complexity implied by a WHUP.

Clocked Unrolling Templates. VPV can also discuss a local formulation[8] of machine-only $\mathsf{K}^t(x)$, which bounds the time complexity of producing each individual bit x_i of x given i in binary. Consider the following polynomial-time function, which "unrolls" a given machine into an n-bit string — essentially a machine analog of the truth-table generator for circuits [32].

For every polynomial time bound p, for each sufficiently large n, a "template" machine M of length n can extract an n-bit vector of p-step decisions from sufficiently shorter machines. That is, pasting a shorter

[8]A local formulation of standard K^t complexity appears, for example, as Definition 3 of [37] where a hardness assumption about deciding local K^t is used in a direct and elegant construction of pseudo-random functions.

Algorithm 4 Unrolling a Machine, $\mathsf{Unroll}(D, n, n^c)$

Parameters n in unary, n^c in unary

```
for all i ∈ {0, ..., n} do
    if D run on input bin(i) accepts within n^c steps then
        Print 1
    else
        Print 0
```

machine D into M and running the result agrees with $\mathsf{Unroll}(D, n, p)$. Translating into VPV define the schemea $\mathsf{UNROLLT}[n_0, c] :=$

$$\forall n.(n > n_0)\ \exists M.(|M| = n)\ \forall D.(|D| \le n/16)\ \mathsf{run}_U(\mathsf{paste}_U(M, D), \varepsilon, n^{2c+1}) = \mathsf{Unroll}(D, n, n^c)$$

Witness M in $\mathsf{poly}(n, c)$ time by printing an appropriate U-encoding of Algorithm 5 below.

Algorithm 5 Unrolling Template

Parameters D the description of a machine, n in unary, n^c in unary

```
1: Write 0^{n/16} to the worktape
2: Move the head two cells right — leaving a blank
3: Write 1^n to the worktape
4: Move the head two cells right — leaving a blank
5: Write 1^{n^c} to the worktape
6: Run Unroll on the contents of the worktape, with arguments separated by blanks
```

To accomodate paste, implement line 1 of M by *explicitly* printing 0 symbols — one state per symbol, exactly as in the previous two templates. Implement lines 3 and 4 by maintaining binary counters on the worktape. This requires $O(\log n)$ and $O(c \log n)$ bits to be hardcoded in M, respectively. Finally, the code of Unroll takes some absolute constant number of bits in the encoding of M. Just as above, printing M takes fixed polynomial time given $(1^n, 1^c)$ as input.

4.5.2 Deterministic Time Hierarchy Theorem

Consider the compressible-counterexample deterministic time hierarchy theorem, used to obtain Student-Teacher lower bound for constructing circuits [8].

Lemma 4.20. For every $c \in \mathbb{N}$, there is a language $H_c \in \mathsf{DTIME}[n^{c+1}]$ satisfying the following:

- Counterexamples: Every candidate n^c-time TM M that tries to compute H_b will make a mistake on an n-bit input $x_{\text{error}} = \ulcorner M \urcorner \circ \pi$ where $\circ$ denotes concatenation and $\pi \in 0^*$ is a padding string chosen to make $|x_{\text{error}}| = n$ for all sufficiently large n.
- Compressibility of Counterexamples: The counterexamples x_{error} are efficiently compressible to $O(\log(n))$ bits by recording both the constant-length description M and n in binary, by just padding M to the appropriate length.

Though the diagonalization machine H_c uses time $O(n^{c+1})$, the implicit refuter uses only time $O(n)$ — and is the same regardless of which polynomial "slice" of the hierarchy is being refuted! The deterministic time hierarchy theorem has a straightforward translation into a sequence of VPV-sentences.

Formalization 4.21.

$$\mathsf{DTIMEH}[c] := \forall n\ \forall M.(|M| < n/16)\ \exists X.(|X| = n)\ \mathsf{run}(M, X, n^c) \ne \mathsf{run}(H_c, X, n^{c+1})$$

This is a simpler formula than the VPV-schemata $\Phi[c]$ used in WHUPs, because the quantifier prefix is $\forall\exists$ instead of $\forall\exists\forall$. We know that $\mathsf{VPV} \vdash \mathsf{DTIMEH}[c]$ for each c, and each proof is "essentially the same" up to substitution of n^c (Lemma 3.1 of [29]). Therefore, Buss Witnessing (Theorem 2.13) applies and we immediately get refuters for H_c. But the uniformity is not expoited by Buss Witnessing – we get a sequence of refuters with arbitrary and unrelated polynomial runtime for each c, and indeed each runtime may be much larger than n^c. This is much worse than the absolute refuter obtained outside VPV.

In this simpler setting, is there a generic way to convert such uniform collections of proofs into absolute witnessing that we *already know exists*? To further assess the plausibility of such convenient witnessing, we let w_H be the VPV-term given by the compressible counterexamples of Lemma 4.20 and ask the following

Question 4.22. Does VPV prove that w_H witnesses the $\mathsf{DTIMEH}[c]$ errors for each c?

4.5.3 Efficient Conversion From Multi-Tape to One-Tape Turing Machines

It is a classical theorem that for $k \geq 2$ any k-tape Turing Machine can be simulated by a one-tape Turing Machine with at most quadratic overhead.

Theorem 4.23 (Claim 1.6 of [3])**.** If the language L can be decided in time n^c on a k-tape Turing Machine, then L can be decided in time $16kn^{2c}$ on a single-tape Turing Machine.

Formalize this in VPV by defining the function symbols run_k and run_1 to simulate k-tape and single-tape Turing Machines, respectively. Then consider the following collection of VPV-theorems $\mathsf{ONE.TAPE}[n_0, k, c] :=$

$$\forall M\ \exists M'\ \forall n.(n \geq n_0)\ \forall X.(|X| = n)\ \ \mathsf{run}_k(M, X, n^c) = \mathsf{run}_1(M', X, 16kn^{2c})$$

This quantifier prefix is identical to that of $\Phi[c]$, but the types are different: strings instead of numbers. Therefore, a witnessing algorithm for $\mathsf{ONE.TAPE}$ is given M encoded in binary and must print the encoding of M'. The encoding length $|M|$ under any reasonable encoding — which we fix using run_k — is determined by the number of states and alphabet size. The number of states in M' given by straightforward proofs of Theorem 4.23 is exponential in k but linear in the states and alphabet-size of M. Therefore, in a fixed polynomial time in $|M|$, a witnessing algorithm prints M'. Only the transformation of the "code" of M matters to the witnessing algorithm — **not** the runtime bound on M.

5 Consequences of Provably Hard Truth Tables

Under a natural witnessing hypothesis for a theory corresponding to uniform $\mathsf{AC}^0[\mathsf{qpoly}]$, we have $\mathsf{P} \neq \mathsf{NP}$.

5.1 A Theory for $\mathsf{AC}^0[\mathsf{qpoly}]$-Reasoning

In Section 2.4, we recalled the two-sorted theory V^0 which corresponds to log-uniform AC^0 circuits. Here, we extend the definition to a new theory denoted $\mathsf{V}^0_{\#}$, corresponding to $\mathsf{polylog}$-uniform $\mathsf{AC}^0[\mathsf{qpoly}]$. The definition of $\mathsf{V}^0_{\#}$ is very simple: starting from the axioms of V^0 and language $\mathcal{L}(\mathsf{V}^0)$, add the function symbol $\#$, commonly known as the *smash* operator, to the language and defining axioms of V^0. Smash is defined by axioms stating $x\#y = 2^{|x|\cdot|y|}$, for numbers x, y. It is used to give quasipolynomial growth rates of the number type.

The characterization of V^0 with uniform AC^0 is carried out in detail in Chapters IV and V of [17]. They treated AC^0 as the logtime-hierarchy LH, known to be equivalent to logtime-uniform AC^0.

Theorem 5.1 (Folklore)**.** log-uniform $\mathsf{AC}^0[\mathsf{poly}] = \mathsf{LH}$.

Importantly, this is generalizable to the $\mathsf{polylog}$-hierarchy polyLH.

Theorem 5.2 (Folklore)**.** $\mathsf{polyLH} = \mathsf{polylog}$-uniform $\mathsf{AC}^0[\mathsf{qpoly}]$

Cook and Nguyen showed that all logtime-uniform AC^0-functions are Σ_1^B-definable in V^0, as well as the converse witnessing theorem that any $\forall\Sigma_1^B$ sentence provable in V^0 has its existential quantifier witnessed by a logtime-unform AC^0 function.

This correspondence holds for uniform $\mathsf{AC}^0[\mathsf{poly}]$, but generalizes to any class of circuit sizes that is closed under composition, with the appropriate modification of the language and axioms. By adding the smash operator $\#$, it is standard to get an identical correspondence between $\mathsf{polylog}$-uniform $\mathsf{AC}^0[\mathsf{qpoly}]$ and $\mathsf{V}^0_{\#}$.

We believe this theory is of independent interest, and will discuss it's strength at the end of the section.

5.2 Stating Existential Circuit Lower Bounds in $\mathcal{L}(\mathsf{V}^0_\#)$

We first give a logical translation of the classical lower bound due to Shannon.

Theorem 5.3 (Shannon Counting)**.** Let $b > 0$. For every sufficiently large N, there exists a truth table X of length N which is not succinctly represented by any $|N|$-input circuit of size $|N|^b$.

Normally, it would be impossible to describe such a lower bound in $\mathcal{L}(\mathsf{V}^0)$ or $\mathcal{L}(\mathsf{V}^0_\#)$, as it involves evaluating *general* circuits, instead of AC^0 circuits. However, because our feasible objects will be truth tables of length 2^n, we can evaluate general circuits of each fixed polynomial size n^c in size $\mathsf{qpoly}(2^n)$-AC^0. While we normally reserve capital letters for string-types, we will use N to refer to 2^n in this section. More formally, we use the following folklore lemma about AC^0 evaluation of general circuits.

Lemma 5.4 (Folklore)**.** Let $k > 0$. There is a $\mathsf{polylog}$-uniform AC^0 circuit of size $N^{\log(N)^{3k}}$ which on input the DCL encoding of a general circuit C of size n^k and an input x of n bits, outputs $C(x)$.

Proof. By a standard counting argument, it is known that there are at most $2^{O(s(n)\log n)}$ circuits of size $s(n)$. As a DCL representation of a size $s(n)$ circuit is a string of length at most $s(n)^2$, we have that there are at most $2^{O(s(n)^2\log n)}$ DCL strings of size $s(n)$ circuits. Plugging in $s(n) = n^k$, we get that there are up to $2^{O(n^{2k}\log n)} = O(N^{(\log N)^3})$ DCL strings of size n^k circuits.

We construct an AC^0 circuit $\mathcal{E}$ to evaluate any size n^k general circuit as follows:

1. Circuit Lookup Layer: Have a multiplexer identify the which circuit C has been input
2. Input Lookup Layer: Have a multiplexer identify the circuit input x which has been specified.
3. Evalutation Layer: Output the memorized evaluation of indentified circuit C and input x.

Uniformity. Normally, the above circuit would be highly non-uniform. However, in the size regime of $N = 2^n$, this becomes feasible. A $\mathsf{polylog}$-uniformity algorithm of $\mathcal{E}$ would have runtime $\mathsf{polylog}(N^{\log(N)^{2k}})$, which is $\mathsf{DTIME}[\mathsf{poly}(n)]$. This means that a uniformity algorithm $A_\mathcal{E}$ running in time $\log(N)^c = n^c$ for circuit $\mathcal{E}$ has time to evaluate circuits of size $n^{c/3}$. Setting $c > 3k$ would allow for the uniformity algorithm to, after stages (1) and (2), evaluate the input (C, x) and give the output bits. ☐

By Lemma 5.4, we have in $\mathsf{V}^0_\#$ a sequence of function symbols for generating the truth tables of fixed-polynomial size *general* circuits given as input. This is feasible due to the input length N, where a fixed polynomial is only $\mathsf{polylog}$. Define the sequence of symbols $\mathsf{TT}_b(C, N)$ as functions that take a number N and circuit C of length $|C| = |N|^b$ and output the truth table of C of length N. These operations have function symbols and defining axioms in $\mathsf{V}^0_\#$ because the $\mathsf{polylog}$-uniform AC^0 complexity will be $N \cdot \mathsf{qpoly}(N)$ to evaluate the circuit C on each of the N possible inputs (by Lemma 5.4). We will also assert that $\mathsf{TT}_b(\cdot,\cdot)$ checks if the input circuit is valid and of length $|N|^b$, and treats it as the constant 0 function if it is not. This is because verifying a DCL encoding can be done efficiently in AC^0. We can now give the following translation,

$$\mathsf{Hard}(b) \triangleq \forall N\, \exists X\, (|X| = N)\, \forall D\, (|D| < N)\ \mathsf{TT}_b(D, N) \neq X$$

The above formula makes a choice to rely on the function symbol TT_b verifying that N is a power of two and the circuit D is a valid circuit of size $|N|^b$ instead of explicitly verifying this outside of the function symbol. We make this choice because we will need a WHUP for $\mathsf{V}^0_\#$, and the cleanest presentation of such a WHUP is given when TT_b absorbs the circuit verification procedure. See the discussion on WHUPs below for more detail.

Comparison to VAPC and Shannon Counting. The typical description of Shannon counting is that there exists a truth table x of length N, which has circuit complexity $N/\log N$. It is this formulation which Jeřábek showed is provable in VAPC. Our logical translation, however, is weaker: we only require proving a *schema* that asserts a truth table with *super-fixed-polynomial* circuit complexity exists, rather than exponential.

5.3 Round Elimination of the Student-Teacher Refuter

Student-Teacher Interpretation The structure of the Student-Teacher game is very similar to previous sections. In each round, a polylog-uniform $\mathsf{AC}^0[\mathsf{qpoly}(N)]$ Student constructs a truth table X and queries the Teacher. Every round that the Student is not correct, the Teacher will respond with a small circuit D that succinctly represents X. Crucially, Teacher's response (to be replaced with a SearchMCSP oracle) is of length $\mathsf{polylog}(N)$ and computable in PH.

The round elimination strategy will be different from previous sections. We will show that the problem of outputting the i-th bit of the Student-Teacher game is in fact in the polylog hierarchy polyLH, and use the assumption $\mathsf{P} = \mathsf{NP}$ to show that the output of the Student-Teacher game will have small circuit complexity. We begin with a warm-up lemma.

Lemma 5.5 (Lemma 2.5, [14])**.** Assume $\mathsf{P} = \mathsf{NP}$. Then for every polylogtime-uniform AC^0 algorithm A which outputs n bits on input 1^n, the output $A(1^n)$ has circuit complexity at most $\mathsf{polylog}(n)$.

Proof. Let D be the uniformity machine for AC^0 algorithm A which on input n in binary and index i in binary, reports the i-th bit of wire and gate information of A_n, the n-th AC^0 circuit of family A. Let $f(n,i)$ be the function that outputs the i-th output bit of $A_n(1^n)$. Notice that f is in PH: due to A_n being constant depth, one can existentially and universally guess gate/wire information and verify it due to D. This means the evaluation of f is in $\Sigma_d\text{-}\mathsf{TIME}[O(\log^d n)]$ for some constant d depending on the depth of circuit A and the polynomial time SAT algorithm. By assumption, $\mathsf{P} = \mathsf{PH}$, hence the evaluation of $A_n(1^n)$ may be done in deterministic $\mathsf{polylog} n$ time. It is standard to convert such a program to a circuit of polylog size. $\square$

The above lemma is a blueprint for the generalization to Student-Teacher games. Let φ_b denote the quantifier-free part of $\mathsf{Hard}[b]$, to state

Lemma 5.6. Assume $\mathsf{P} = \mathsf{NP}$. Let $r \in \mathbb{N}$ be a constant. Then any $\mathsf{polylog}$-uniform $\mathsf{AC}^0[N^{(\log N)^k}]\text{-ST}^{CX[\varphi_b, r]}$ game for $\mathsf{Hard}[b]$ has output of circuit complexity $\log(N)^m = n^m$ on inputs 1^N, for some $m = m(k) > 0$. Furthermore, if $k = O(1)$, then $m = O(1)$.

Proof. We exactly follow the proof of Lemma 5.5 with one major modification: we must replace the oracle gates/Teacher's responses.

Let the Student S be an $AC^0_d[N^{(\log N)^k}]$ counterexample oracle circuit with $d > r$ and a uniformity algorithm $A(i,n)$ which runs in time $(\log N)^{q_1}$. As well, let $\mathsf{P} = \mathsf{NP}$ be realized by a polynomial time SAT algorithm of time n^α. As $\mathsf{SearchMCSP} \in \mathsf{FNP}$ there is by assumption a fixed polynomial $p = p(N, |s(n)|)$, for $s(n)$ a size function $s(n) < 2^n/n$, where $\mathsf{SearchMCSP} \in \mathsf{DTIME}[p(N, s(n))]$. With $s(n) = n^b$, we have that there is a $\mathsf{LOGTIME}$-uniform circuit family $\{C_N\}_N$ of size $p(N)^k$ solving $\mathsf{SearchMCSP}$. By Lemma 5.4, we can evaluate C_N by a $\mathsf{polylog}$-uniform $\mathsf{AC}^0_3[N^{\log(N)^{3k}}]$ circuit $\mathcal{E}_N$. Let the uniformity algorithm for $\mathcal{E}$, $A_{\mathcal{E}}$, run in time $(\log N)^{q_2}$ for some constant q_2. From student S, we modify the oracle circuit, by replacing any oracle gate by the circuit solving $\mathsf{SearchMCSP}(\cdot, n^b)$, $\mathcal{E}_N$, and all oracle output bits by the output bits of $\mathcal{E}_N$. Denote this new circuit S^*.

By the same proof of Lemma 5.5, the output $S^*(1^N)$ has circuit complexity $(\log N)^m = n^m$, where $m = 100 d\alpha k \max(q_1, q_2)$. This follows by the repeated application of the polynomial time SAT algorithm, and the substitution of $\mathcal{E}_N$ into S for each Teacher oracle. $\square$

Finally, we will introduce a WHUP for $\mathsf{V}^0_{\#}$ in order to obtain an absolute Student-Teacher game for any $b \in \mathbb{N}$ and $\mathsf{Hard}[b]$. Proof-theoretic consequences will follow.

Witnessing Hypothesis. The structure of the schema $\mathsf{Hard}[b]$ is somewhat different from the schema $\mathsf{HiK}^t[c]$ for the existence of high $\mathsf{K}^{\mathsf{poly}}$ strings seen in Section 4. For $\mathsf{Hard}[b]$, we are substituting *function symbols* instead of substituting runtimes, contrasting with our WHUP for VPV which substitutes runtimes into a universal machine. Such a difference is natural when we go from reasoning with Turing machines to reasoning with circuits. Another variation of WHUPs is required to handle varying function symbols.

Definition 5.7. A *parametrized uniform circuit family* $\{C_n(b)\}_n$ is a circuit family where the uniformity algorithm $A(i,n,b)$ takes in an index to the DCL i, the input length n, and an additional parameter b, all represented in binary. Unless stated otherwise, we will only consider $\mathsf{polylog}$-uniformity.

Definition 5.8. Fix a parametrized uniform $\mathsf{AC}^0[\mathsf{qpoly}]$ family $C(b) = \{C_n(b)\}_n$ and let $\mathsf{f}_b(\overline{X})$ be the $\mathsf{V}^0_\#$ function symbol which evaluates C on input $\overline{X}$ with parameter b for the uniformity machine. We say that an infinite sequence of formulas Φ is a *parametrized uniform schema* if Φ is obtained by taking an infinite union over parameter values of a parametrized uniform circuit family. Formally, let φ be a formula which has a parametrized uniform function symbol f_p. We denote $\varphi(f_{p/b})$, $b \in \mathbb{N}$ to be "substituting" the value b for the parameter p, where we use the function f_b wherever f is named in φ. We set

$$\Phi = \bigcup_{c \in \mathbb{N}} \varphi(f_{p/c}).$$

Outside the theory, this can be thought of as a substitution; all we are doing is substituting numerals into the parameter of a uniformity algorithm. However, it is **not** a true term substitution in the object language, as a first order theory cannot treat functions as free variables.

The following WHUP for $\mathsf{V}^0_\#$ strengthens the WHUP for VPV using parametrized uniform circuits.

Hypothesis 5.9 ($\mathsf{V}^0_\#$ Witnessing Hypothesis for Uniform Proofs)**.** Suppose $\Phi = \bigcup_{c \in \mathbb{N}} \varphi(f_{p/c})$ is a parametrized uniform schema with $\varphi \in \Sigma^B_2$ and $\mathsf{V}^0_\# \vdash \Phi$. Then there is a finite sequence $F_1, \dots F_r$ of $\mathsf{V}^0_\#$-function symbols such that, for infinitely many c,

$$\mathsf{V}^0_\# \vdash \forall n\ \forall \vec{Y}_1, \dots, \forall \vec{Y}_r \Big(\varphi(n, n^c), F_1(n, c), \vec{Y}_1) \vee \varphi(n, n^c, F_2(n, c, \vec{Y}_1), \vec{Y}_2) \vee \\ \cdots \vee \varphi(n, n^c, F_r(n, c, \vec{Y}_1, \dots, \vec{Y}_{r-1}), \vec{Y}_r)\Big)$$

We are now ready to show our provability consequences for $\mathsf{V}^0_\#$.

Theorem 5.10. Assume Hypothesis 5.9. If $\mathsf{V}^0_\# \vdash \mathsf{Hard}(b)$, for every $b \in \mathbb{N}$, then $\mathsf{P} \neq \mathsf{NP}$.

Proof. Because $\mathsf{Hard}[b]$ is a parametrized-uniform schema, under Hypothesis 5.9, there is a *fixed* polylog-uniform $\mathsf{AC}^0[\mathbf{qpoly}(N)]$ student S for the Student-Teacher game of $\mathsf{Hard}(b)$, for infinitely many $b \in \mathbb{N}$. For sake of contradiction, assume $\mathsf{P} = \mathsf{NP}$. Let $m = m_{(5.6)}$ be the exponent in the circuit size of $S(1^N)$, per Lemma 5.6. Note that m is a constant. If $b > m$, then we have a contradiction, as the output $S(1^N)$ will have circuit complexity smaller than n^b. ☐

5.4 $\mathsf{VPV}_\#$ Proves $\mathsf{Hard}(b)$ for Every b

Before demonstrating that $\mathsf{V}^0_\# \vdash \mathsf{Hard}(b)$, we show as a conceptually simpler task that $\mathsf{VPV}_\# \vdash \mathsf{Hard}(b)$, for an appropriate logical translation of weak Shannon counting in $\mathcal{L}(\mathsf{VPV}) \cup \{\#\}$. The theory $\mathsf{VPV}_\#$ takes the theory VPV and adds the functions symbols and defining axioms for $|\cdot|_1$ and $\#$, where $|\cdot|_1$ gives the length of a *number*, and $\#$ is the smash operator. With these additions, we have the following $\mathcal{L}(\mathsf{VPV}_\#)$ logical translation $\mathsf{Hard}(b)$,

$$\forall n, N.(n = |N|_1)\ \exists F.(|F| = N)\ \forall D.(|D| = n^b)\ \mathsf{TT}(D, N) \neq F$$

We can prove weak Shannon counting in this theory by *iterated halving*, the classic procedure common in learning algorithms. Let $\mathcal{C}_b$ be the set of Boolean circuits of size n^b with n input bits. There are at most $2^{O\left(n^b \log n\right)}$ such circuits. If $N = 2^n$, then this number is bounded by $N \# N \# N \cdots \# N$, $b+1$ many times. Set $\mathcal{C}^0_b = \mathcal{C}_b$. We will proceed in rounds; in round i, we set

$$b_i = \arg \min_{b \in \{0,1\}} |\{C \in \mathcal{C}^{i-1}_k \mid \mathsf{TT}(C)_i = b\}|$$
$$\mathcal{C}^i_k = \{C \in \mathcal{C}^{i-1}_k \mid \mathsf{TT}(C)_i = b_i\}.$$

Clearly, $|\mathcal{C}^i_b| \leq |\mathcal{C}^{i-1}_b|/2$, hence halving our search space. After $r = |\mathcal{C}_b|$ rounds, we will have 0 remaining circuits matching the truth table prefix $b_1 b_2 \dots b_r$. If 0 circuits remain, then we have the answer

$b_1b_2\dots b_r0^{N-r}$. As N is feasible in the translation $\mathsf{Hard}(b)$, computing b_i each round is a feasible minimization in $\mathsf{VPV}_{\#}$. This step is feasible in $\mathsf{VPV}_{\#}$, but *not* in $\mathsf{V}^0_{\#}$. Finally, the rounds can be expressed as a Σ^B_0 induction over the bits of the resulting truth table, where one existentially guesses a *number* $b \in \{0,1\}^i$ in round $i < |N\#N\#N\cdots\#N|_1$ which is the least popular truth table prefix generated by size n^b circuits.

5.5 $\mathsf{V}^0_{\#} \vdash \mathsf{Hard}(b)$

We give a proof sketch that, in fact, $\mathsf{V}^0_{\#} \vdash \mathsf{Hard}(b)$, for every $b \in \mathbb{N}$. We need the following folklore theorem,

Theorem 5.11. [17] V^0 proves the soundness of bounded depth Frege

Corollary 5.12. $\mathsf{V}^0_{\#}$ proves the soundness of quasipolynomial size bounded depth Frege.

Lemma 5.13. $\mathsf{V}^0_{\#} \vdash \mathsf{Hard}(b)$, for every $b \in \mathbb{N}$.

Proof Sketch. It is known that depth-(1.5) Frege has quasipolynomial sized propositional proofs of the dWPHP [40]. Further, these proofs are highly uniform, in the sense that you can give a direct connection language for the proofs, as you would for circuits, and this language would be $\mathsf{polylog}$-uniform $\mathsf{AC}^0[\mathsf{poly}]$. A $\mathsf{V}^0_{\#}$ proof of $\mathsf{Hard}(b)$ would simply verify the soundness of the bounded depth Frege proof of dWPHP, substituting each propositional variable $x_{C,T}$ with the assertion that the truth table of the circuit C of size n^b is T. ☐

This proof method is likely not the easiest method; a more straightforward way would be directly taking the bounded arithmetic proof of $\mathsf{dWPHP}(\Sigma^b_1(\alpha))$ in Buss's theory $T^2_2(\alpha)$, and reformulating it as a proof in V^0, replacing the uninterpreted oracle symbol α with V^0 function symbols.

We then get as a corollary the surprising consequence,

Corollary 5.14. If Hypothesis 5.9 is true, then $\mathsf{P} \neq \mathsf{NP}$.

6 Provability of the Circuit Size Hierarchy Theorem

A standard nonconstructive result in complexity theory is the circuit size hierarchy theorem.

Theorem 6.1. (Circuit Size Hierarchy) For any $a > b \geq 1$, $\mathsf{SIZE}[n^b] \subsetneq \mathsf{SIZE}[n^a]$.

We denote for a fixed $a > b \geq 1$, $\mathsf{CSH}[a,b]$ to be the circuit size hierarchy theorem with parameters a and b.

This is implied by Shannon's classic counting argument for showing that most Boolean functions require maximal circuit complexity. In a recent result of Carmosino et. al. [9], they showed that if there exists an $a > b \geq 1$ such that VPV proves $\mathsf{CSH}[a,b]$, then $\mathsf{P} \not\subset \mathsf{SIZE}[n \log n]$, a breakthrough circuit lower bound.

A natural open question from this work is whether it is possible to extract *all* the hardness.

Question 6.2 ([9])**.** Does VPV-provability of the Circuit Size Hierarchy theorem imply $\mathsf{P} \not\subset \mathsf{SIZE}[n^k]$, for every $k \geq 0$?

We answer this question in the affirmative.

Theorem 6.3 (Informal)**.** Under the assumption of a Witnessing Hypothesis, if VPV proves $\mathsf{CSH}[a,b]$ for infinitely many pairs $a > b \geq 1$, then $\mathsf{P} \not\subset \mathsf{SIZE}[n^k]$ for any $k \geq 0$.

This is orthogonal to [9] as their result only requires VPV to prove $\mathsf{CSH}[a,b]$ for a *single* pair (a,b).

We now introduce the necessary WHUP to prove Theorem 6.3.

6.1 A $\forall\exists\forall\exists$-WHUP for VPV

7 Acknowledgements

Marco Carmosino was. Stefan Grosser was supported by the NSERC PGS D Scholarship. The authors thank Robert Robere, Sam Buss, Erfan Khaniki, and Jan Pich for helpful discussions. The authors also thank Robert Robere, Sam Buss, and Noel Arteche for suggesting improvements to an earlier draft of the paper. As well, the authors thank Erfan Khaniki for pointing out the necessity for bounding Skolem terms in our WHUPs.

Part of this work was completed while the authors attended the Simons Institute "Meta Complexity" program reunion in April 2024.

References

[1] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. "PRIMES is in P". In: *Annals of mathematics* (2004), pp. 781–793.

[2] Eric Allender et al. "The pervasive reach of resource-bounded Kolmogorov complexity in computational complexity theory". In: *J. Comput. Syst. Sci.* 77.1 (2011), pp. 14–40. DOI: 10.1016/J.JCSS.2010.06.004. URL: https://doi.org/10.1016/j.jcss.2010.06.004.

[3] Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach.* Cambridge University Press, 2009.

[4] Samuel R Buss. "Bounded arithmetic and constant depth Frege proofs". In: *Complexity of computations and proofs* 13 (2004), pp. 153–174.

[5] Samuel R Buss. *Handbook of proof theory.* Elsevier, 1998.

[6] Samuel R. Buss. *Bounded arithmetic.* Bibliopolis, 1986.

[7] Samuel R. Buss. "The witness function method and provably recursive functions of peano arithmetic". In: *Logic, Methodology and Philosophy of Science IX.* Ed. by Dag Prawitz, Brian Skyrms, and Dag Westerståhl. Vol. 134. Studies in Logic and the Foundations of Mathematics. Elsevier, 1995, pp. 29–68. DOI: https://doi.org/10.1016/S0049-237X(06)80038-8. URL: https://www.sciencedirect.com/science/article/pii/S0049237X06800388.

[8] Marco Carmosino et al. "LEARN-uniform circuit lower bounds and provability in bounded arithmetic". In: *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS).* IEEE. 2022, pp. 770–780.

[9] Marco Carmosino et al. "Provability of the Circuit Size Hierarchy and Its Consequences". In: (2024).

[10] Lijie Chen, Shuichi Hirahara, and Hanlin Ren. "Symmetric Exponential Time Requires Near-Maximum Circuit Size". In: *Proceedings of the 56th Annual ACM Symposium on Theory of Computing.* 2024, pp. 1990–1999.

[11] Lijie Chen, Jiatu Li, and Igor Carboni Oliveira. *Reverse mathematics of complexity lower bounds.* Apr. 2024. URL: https://eccc.weizmann.ac.il/report/2024/060/.

[12] Lijie Chen, Xin Lyu, and R Ryan Williams. "Almost-everywhere circuit lower bounds from non-trivial derandomization". In: *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS).* IEEE. 2020, pp. 1–12.

[13] Lijie Chen, Roei Tell, and Ryan Williams. "Derandomization vs Refutation: A Unified Framework for Characterizing Derandomization". In: *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS).* IEEE. 2023, pp. 1008–1047.

[14] Lijie Chen et al. "Constructive separations and their consequences". In: *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS).* IEEE. 2022, pp. 646–657.

[15] Alan Cobham. "The intrinsic computational difficulty of functions". In: (1965).

[16] Stephen Cook and Jan Krajíček. "Consequences of the provability of NP in P/poly". In: *The Journal of Symbolic Logic* 72.4 (2007), pp. 1353–1371.

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

[18] Stephen A Cook. "Feasibly constructive proofs and the propositional calculus (preliminary version)". In: *Logic, Automata, and Computational Complexity: The Works of Stephen A. Cook*. 2023, pp. 193–218.

[19] Stephen A. Cook. "Feasibly Constructive Proofs and the Propositional Calculus (Preliminary Version)". In: *Proceedings of the 7th Annual ACM Symposium on Theory of Computing, May 5-7, 1975, Albuquerque, New Mexico, USA*. Ed. by William C. Rounds et al. ACM, 1975, pp. 83–97. DOI: 10.1145/800116.803756. URL: https://doi.org/10.1145/800116.803756.

[20] Lance Fortnow and Rahul Santhanam. "New non-uniform lower bounds for uniform classes". In: *31st Conference on Computational Complexity (CCC 2016)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. 2016.

[21] Harvey Friedman. "One Hundred and Two Problems in Mathematical Logic". In: *J. Symb. Log.* 40.2 (1975), pp. 113–129. DOI: 10.2307/2271891. URL: https://doi.org/10.2307/2271891.

[22] Shuichi Hirahara. "Unexpected hardness results for Kolmogorov complexity under uniform reductions". In: *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*. Ed. by Konstantin Makarychev et al. ACM, 2020, pp. 1038–1051. ISBN: 978-1-4503-6979-4. DOI: 10.1145/3357713.3384251. URL: https://doi.org/10.1145/3357713.3384251.

[23] Pavel Hrubes. "Kreisel's Conjecture with minimality principle". In: *J. Symb. Log.* 74.3 (2009), pp. 976–988. DOI: 10.2178/JSL/1245158094. URL: https://doi.org/10.2178/jsl/1245158094.

[24] Pavel Hrubes. "Theories very close to *PA* where Kreisel's Conjecture is false". In: *J. Symb. Log.* 72.1 (2007), pp. 123–137. DOI: 10.2178/JSL/1174668388. URL: https://doi.org/10.2178/jsl/1174668388.

[25] Rahul Ilango, Jiatu Li, and R Ryan Williams. "Indistinguishability obfuscation, range avoidance, and bounded arithmetic". In: *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*. 2023, pp. 1076–1089.

[26] Emil Jeřábek. "Dual weak pigeonhole principle, Boolean complexity, and derandomization". In: *Annals of Pure and Applied Logic* 129.1-3 (2004), pp. 1–37.

[27] Valentine Kabanets. "Easiness assumptions and hardness tests: Trading time for zero error". In: *Journal of Computer and System Sciences* 63.2 (2001), pp. 236–252.

[28] Oliver Korten. "The hardest explicit construction". In: *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2022, pp. 433–444.

[29] Jan Krajícek and Igor C. Oliveira. "Unprovability of circuit upper bounds in Cook's theory PV". In: *Log. Methods Comput. Sci.* 13.1 (2017). DOI: 10.23638/LMCS-13(1:4)2017. URL: https://doi.org/10.23638/LMCS-13(1:4)2017.

[30] Jan Krajícek and Pavel Pudlák. "The number of proof lines and the size of proofs in first order logic." In: *Arch. Math. Log.* 27.1 (1988), pp. 69–84.

[31] Jan Krajíček. "No counter-example interpretation and interactive computation". In: *Logic from Computer Science: Proceedings of a Workshop held November 13–17, 1989*. Springer. 1992, pp. 287–293.

[32] Jan Krajíček. "On the existence of strong proof complexity generators". In: *Bulletin of Symbolic Logic* 30.1 (2024), pp. 20–40.

[33] Jan Krajíček. "Small circuits and dual weak PHP in the universal theory of p-time algorithms". In: *ACM Transactions on Computational Logic (TOCL)* 22.2 (2021), pp. 1–4.

[34] Jan Krajíček, Pavel Pudlák, and Gaisi Takeuti. "Bounded arithmetic and the polynomial hierarchy". In: *Annals of pure and applied logic* 52.1-2 (1991).

[35] Ming Li and Paul M. B. Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications, 4th Edition*. Texts in Computer Science. Springer, 2019. ISBN: 978-3-030-11297-4. DOI: 10.1007/978-3-030-11298-1. URL: https://doi.org/10.1007/978-3-030-11298-1.

[36] Zeyong Li. "Symmetric Exponential Time Requires Near-Maximum Circuit Size: Simplified, Truly Uniform". In: *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*. 2024, pp. 2000–2007.

[37] Yanyi Liu and Rafael Pass. "A Direct PRF Construction from Kolmogorov Complexity". In: *Advances in Cryptology - EUROCRYPT 2024 - 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26-30, 2024, Proceedings, Part IV*. Ed. by Marc Joye and Gregor Leander. Vol. 14654. Lecture Notes in Computer Science. Springer, 2024, pp. 375–406. DOI: 10.1007/978-3-031-58737-5_14. URL: https://doi.org/10.1007/978-3-031-58737-5%5C_14.

[38] Yanyi Liu and Rafael Pass. "On One-way Functions and Kolmogorov Complexity". In: *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*. Ed. by Sandy Irani. IEEE, 2020, pp. 1243–1254. DOI: 10.1109/FOCS46700.2020.00118. URL: https://doi.org/10.1109/FOCS46700.2020.00118.

[39] Wolfgang Maass. "Quadratic lower bounds for deterministic and nondeterministic one-tape Turing machines". In: *Proceedings of the sixteenth annual ACM symposium on Theory of computing*. 1984, pp. 401–408.

[40] Alexis Maciel, Toniann Pitassi, and Alan R Woods. "A new proof of the weak pigeonhole principle". In: *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing*. 2000, pp. 368–377.

[41] Tohru Miyatake. "ON THE LENGTH OF PROOFS IN FORMAL SYSTEMS". In: *Tsukuba Journal of Mathematics* 4.1 (1980), pp. 115–125. ISSN: 03874982. URL: http://www.jstor.org/stable/43685436 (visited on 10/06/2024).

[42] Moritz Müller and Ján Pich. "Feasibly constructive proofs of succinct weak circuit lower bounds". In: *Annals of Pure and Applied Logic* 171.2 (2020), p. 102735.

[43] Cody Murray and Ryan Williams. "Circuit lower bounds for nondeterministic quasi-polytime: an easy witness lemma for NP and NQP". In: *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*. 2018, pp. 890–901.

[44] Igor C Oliveira. "Meta-Mathematics of Computational Complexity Theory". In: *SIGACT News Complexity Theory Column (DRAFT)* ().

[45] Rohit Parikh. "Existence and feasibility in arithmetic". In: *The journal of symbolic logic* 36.3 (1971), pp. 494–508.

[46] Rohit Parikh. "Some Results on the Length of Proofs". In: *Transactions of The American Mathematical Society - TRANS AMER MATH SOC* 177 (Mar. 1973), pp. 29–29. DOI: 10.2307/1996581.

[47] Ján Pich. "Circuit lower bounds in bounded arithmetics". In: *Annals of Pure and Applied Logic* 166.1 (2015), pp. 29–45.

[48] Ján Pich. "Logical strength of complexity theory and a formalization of the PCP theorem in bounded arithmetic". In: *Log. Methods Comput. Sci.* 11.2 (2015). DOI: 10.2168/LMCS-11(2:8)2015. URL: https://doi.org/10.2168/LMCS-11(2:8)2015.

[49] Pavel Pudlák. "Chapter VIII - The Lengths of Proofs". In: *Handbook of Proof Theory*. Ed. by Samuel R. Buss. Vol. 137. Studies in Logic and the Foundations of Mathematics. Elsevier, 1998, pp. 547–637. DOI: https://doi.org/10.1016/S0049-237X(98)80023-2. URL: https://www.sciencedirect.com/science/article/pii/S0049237X98800232.

[50] Hanlin Ren, Rahul Santhanam, and Zhikun Wang. "On the range avoidance problem for circuits". In: *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2022, pp. 640–650.

[51] Neil Thapen. "The weak pigeonhole principle in models of bounded arithmetic". PhD thesis. University of Oxford, 2002.

[52] Ryan Williams. "Nonuniform ACC circuit lower bounds". In: *Journal of the ACM (JACM)* 61.1 (2014), pp. 1–32.

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il