Consistency of circuit lower bounds with bounded theories

Jan Bydžovský[*]   Jan Krajíček[†]   Igor C. Oliveira[‡]

February 22, 2020

Abstract

Proving that there are problems in P^{NP} that require boolean circuits of super-linear size is a major frontier in complexity theory. While such lower bounds are known for larger complexity classes, existing results only show that the corresponding problems are hard on *infinitely many input lengths*. For instance, proving *almost-everywhere* circuit lower bounds is open even for problems in MAEXP. Giving the notorious difficulty of proving lower bounds that hold for all large input lengths, we ask the following question:

Can we show that a large set of techniques cannot prove that NP *is easy infinitely often?*

Motivated by this and related questions about the interaction between *mathematical proofs* and *computations*, we investigate circuit complexity from the perspective of logic.

Among other results, we prove that for any parameter $k \geq 1$ it is consistent with theory T that computational class $\mathcal{C} \nsubseteq i.o.\mathrm{SIZE}(n^k)$, where $(T, \mathcal{C})$ is one of the pairs:

$$T = \mathsf{T}_2^1 \text{ and } \mathcal{C} = \mathsf{P}^{\mathsf{NP}}, \quad T = \mathsf{S}_2^1 \text{ and } \mathcal{C} = \mathsf{NP}, \quad T = \mathsf{PV} \text{ and } \mathcal{C} = \mathsf{P}.$$

In other words, these theories cannot establish infinitely often circuit upper bounds for the corresponding problems. This is of interest because the weaker theory PV already formalizes sophisticated arguments, such as a proof of the PCP Theorem [Pic15b]. These consistency statements are unconditional and improve on earlier theorems of [KO17] and [BM18] on the consistency of lower bounds with PV.

---

[*]Institute of Discrete Mathematics and Geometry, Technische Universität Wien – `jan.bydz@gmail.com`
[†]Faculty of Mathematics and Physics, Charles University in Prague – `krajicek@karlin.mff.cuni.cz`
[‡]Department of Computer Science, University of Warwick – `igor.oliveira@warwick.ac.uk`

1 Introduction

Understanding the computational power of polynomial size boolean circuits is one of the most mysterious questions in computer science. Despite major efforts to address this problem and significant progress in several *restricted* settings (e.g. [Mul99, Ros10, MW18]), it is consistent with current knowledge that every problem in NP can be computed by circuits containing no more than $4n$ gates [FGHK16]. This bound is much weaker than the lower bound results conjectured by most (but not all) researchers in the field. For instance, it is reasonable to expect that computing k-clique on n-vertex graphs requires circuits of size $n^{\Omega(k)}$, but we appear to be very far from establishing a result of this form for *unrestricted* boolean circuits.

Fixed-polynomial size circuit lower bounds. Given the difficulty of proving stronger lower bounds for problems in NP, a natural research direction is to investigate super-linear and fixed-polynomial circuit size lower bounds for problems in larger complexity classes. This line of work was started by Kannan [Kan82], who showed that for each $k \geq 1$ there is a problem in $\Sigma_2^p \cap \Pi_2^p$ that cannot be computed by circuits of size n^k. The result was subsequently improved by Köbler and Watanabe [KW98], who obtained the same lower bound for the class $\mathsf{ZPP}^{\mathsf{NP}} \subseteq \Sigma_2^p \cap \Pi_2^p$, and by Cai [Cai07], who showed it for $\mathsf{S}_2^p \subseteq \mathsf{ZPP}^{\mathsf{NP}}$. Two incomparable results were then obtained by Vinodchandran [Vin05] and Santhanam [San09], who proved that $\mathsf{PP} \not\subseteq \mathsf{SIZE}[n^k]$ and $\mathsf{MA}/1 \not\subseteq \mathsf{SIZE}[n^k]$, respectively.[1]

Modulo the use of a single bit of advice on each input length, Santhanam's lower bound is known to imply all aforementioned results. Unfortunately, there exist barriers to adapting his techniques to prove super-linear lower bounds for smaller classes such as NP, as explained by Aaronson and Wigderson [AW09]. Establishing such lower bounds is also open for P^{NP}, and constitutes an important frontier in the area of fixed-polynomial size lower bounds.[2] Interestingly, it is known that proving that $\mathsf{P}^{\mathsf{NP}} \not\subseteq \mathsf{SIZE}[n^k]$ for all k is equivalent to showing a stronger Karp-Lipton collapse under the assumption that $\mathsf{NP} \subseteq \mathsf{SIZE}[\mathsf{poly}]$ [CMMW19].[3] We refer to [CC06, GZ11] for more information about uniform complexity classes around P^{NP}.

While existing circuit lower bounds might not be entirely satisfactory from the perspective of the uniform complexity of the problems, there is another important issue with these results: they only establish hardness on *infinitely many input lengths*. Could it be the case that some natural problems are easy on some input lengths and hard on others? Perhaps the existence of exceptional mathematical structures[4] of certain sizes might affect (non-uniform) complexity theory around some input lengths? This possibility seems unlikely, but we are far from understanding the situation. For instance, a basic question in complexity that remains open is whether the nondeterministic time-hierarchy theorem can be extended to an almost-everywhere result (see [BFS09]). On the algorithmic side, an intriguing example is that the natural problem of generating canonical prime numbers admits a faster algorithm on infinitely many input lengths [OS17], but showing that the algorithm succeeds on all input lengths is open. More recent works such as [FS17] and [MW18] show that quite often *some* control can

---

[1]We use $\mathsf{SIZE}[s]$ to denote the set of languages computable by circuits of size at most $s(n)$ on every large enough input length. We say that a language L is in $i.o.\mathsf{SIZE}[s]$ if there is a language $L' \in \mathsf{SIZE}[s]$ such that L and L' agree on infinitely many input lengths.

[2]Indeed, a proof that $\mathsf{E}^{\mathsf{NP}} \not\subseteq \mathsf{SIZE}[n^{1.01}]$ would be considered a breakthrough by some researchers in the field.

[3]Some of our consistency results can be interpreted from this perspective: it is possible to establish stronger "logical" Karp-Lipton collapses if $\mathsf{NP} \subseteq \mathsf{SIZE}[\mathsf{poly}]$ and this inclusion is *provable* in certain theories [CK07].

[4]In the sense of https://en.wikipedia.org/wiki/Exceptional_object

be obtained over the set of hard input lengths. Still, proving an *almost-everywhere* circuit size lower bound beyond $4n$ gates remains open even for problems in $\mathsf{MATIME}[2^n]$ (see [BFT98] for a related lower bound).

Addressing these questions without further assumptions (i.e. unconditionally) appears to be extremely challenging. In this work, we attempt to provide *formal evidence* that some problems in *lower uniform complexity classes* are hard on *every large enough input length*. This can be done via the investigation of circuit complexity from the perspective of mathematical logic. More precisely, we are interested in *unconditional* results showing that lower bounds such as $\mathsf{NP} \nsubseteq i.o.\mathsf{SIZE}[n^3]$ are *consistent* with certain logical theories.[5] To obtain interesting results, we consider theories that can formalize a variety of techniques from algorithms, complexity, and related areas. We focus on first-order theories in the standard sense of mathematical logic, which offers a principled way of investigating consistency statements of the form above. We describe next the theories relevant to our work.

Bounded Arithmetic. Bounded arithmetic theories are fragments of Peano Arithmetic with close connections to computational complexity and proof complexity. Such theories have been widely investigated by logicians and complexity theorists since the 1970's. Among the most influential theories we have Cook's equational theory PV [Coo75] and its corresponding first-order formalization [KPT91] (see also [Jeř06]),[6] Buss's theories S_2^1 and T_2^1 [Bus86], and extensions of these theories by variants of the pigeonhole principle developed primarily by Jeřábek [Jeř04, Jeř05, Jeř07, Jeř09] (such as theory APC_1 extending PV). The objects of study in these theories are natural numbers (representing finite binary strings), and the basic functions and relations are given by polynomial-time (p-time) algorithms in some programming scheme. For instance, Cook [Coo75] relied on Cobham's theorem [Cob65] that all p-time functions can be generated from few initial ones by composition and bounded recursion on notation. For convenience, the language $L(\mathsf{PV})$ we adopt here is the language of PV, having a function symbol for each p-time algorithm.[7] There are relation symbols $=$ and $\leq$ with their usual meaning, and all other relations we want to include are represented by their characteristic functions. The specific axiomatization of PV is not important here: everything will also work for the theory of all true universal $L(\mathsf{PV})$-sentences (to be denoted by True_0), and $\mathsf{PV} \subseteq \mathsf{True}_0$. We only note that PV proves induction for all p-time predicates by formalizing binary search, cf. [KPT91, Kra95].

The original language of theories S_2^1 and T_2^1 as defined in [Bus86] is a finite subset of $L(\mathsf{PV})$, but we consider theories $S_2^1(\mathsf{PV})$ and $T_2^1(\mathsf{PV})$ in the richer language $L(\mathsf{PV})$. (We will add to these theories even more axioms, which makes any consistency statement stronger.) The principal axioms of the two theories are length-induction (LIND) and induction (IND), respectively, accepted for $\Sigma_1^b(\mathsf{PV})$-formulas.[8] Theory $S_2^1(\mathsf{PV})$ is close to PV (it is $\forall \Sigma_1^b(\mathsf{PV})$-conservative over it), but $T_2^1(\mathsf{PV})$ appears to be significantly stronger (cf. [Kra95]). Theories

---

[5]Note also that establishing a consistency statement is a *necessary* step before the corresponding circuit lower bound can be unconditionally established, since a true statement is always consistent with a sound theory.

[6]In this paper we use PV to refer to its first-order formulation (cf. [Kra95, Section 5.3]).

[7]This does not necessarily imply that PV can prove the relevant properties of its function symbols. For instance, the AKS algorithm [AKS02] for testing primality appears as some symbol $f_{\mathsf{AKS}} \in L(\mathsf{PV})$, but PV might not be able to prove that x is prime if and only if $f_{\mathsf{AKS}}(x) = 1$.

[8]We review later in the text some definitions necessary in this work. For more information about standard concepts in bounded arithmetic, we refer to a reference such as [Kra95].

PV, $S_2^1(PV)$ and $T_2^1(PV)$ and their extensions by a form of the pigeonhole principle (often referred to as dWPHP or sWPHP) are actually quite strong for the purposes of complexity theory. They are now known to formalize many key theorems in algorithms, combinatorics, complexity, and related fields (cf. [WP87, Bus86, Kra95, Raz95, Jeř05, Jeř04, Jeř07, Jeř09, CN10, Pic14, Pic15a, Pic15b, BKZ15, LC11, Oja04, Le14, MP17] and references therein).

Recall that the class of $\Sigma_1^b(PV)$-formulas consists of formulas of the form

$$\exists y_1 \le t_1(\overline{x}) \ldots \exists y_k \le t_k(\overline{x}) \, A(\overline{x}, \overline{y}) \ ,$$

where the t_i are $L(PV)$-terms not involving y_i, and A is quantifier free. The definition of this class in the original language of S_2^1 is a bit more complicated (distinguishing two kinds of bounded quantifiers), but in our language $L(PV)$ it is equivalent to this simpler definition. The class of $\Sigma_2^b(PV)$-formulas is defined similarly, but the formula A can also be the negation of a $\Sigma_1^b(PV)$-formula (these negations are $\Pi_1^b(PV)$-formulas). The predicates definable over the natural numbers by $\Sigma_1^b(PV)$-formulas and by $\Sigma_2^b(PV)$-formulas are exactly the predicates from $\Sigma_1^p = \mathsf{NP}$ and from Σ_2^p, respectively. We shall denote the theory of all true $\forall \Sigma_1^b(PV)$-sentences by True_1.

Our results. For an $L(PV)$-formula $\varphi(x)$ and an integer $k \ge 1$, the $L(PV)$-sentence $\mathsf{UB}_k^{i.o.}(\varphi)$ is defined as follows:

$$\forall 1^{(n)} \exists 1^{(m)} (m \ge n) \exists C_m (|C_m| \le m^k) \forall x (|x| = m), \ \varphi(x) \equiv (C_m(x) = 1) \ . \tag{1}$$

The sentence $\mathsf{UB}_k^{i.o.}(\varphi)$ formalizes that the m-bit boolean functions defined by φ (over different input lengths) are computed infinitely often (*i.o.*) by circuits of size m^k.[9]

We unconditionally establish that almost-everywhere circuit lower bounds for complexity classes contained in P^{NP} are consistent with bounded arithmetic theories.

Theorem 1 (Consistency of almost-everywhere circuit lower bounds with bounded theories). *Let $k \ge 1$ be any positive integer. For any of the following pairs of an $L(PV)$-theory T and a uniform complexity class $\mathcal{C}$:*

(a) $T = \mathsf{T}_2^1(PV) \cup \mathsf{True}_1$ *and* $\mathcal{C} = \mathsf{P}^{\mathsf{NP}}$,

(b) $T = \mathsf{S}_2^1(PV) \cup \mathsf{True}_0$ *and* $\mathcal{C} = \mathsf{NP}$,

(c) $T = \mathsf{PV} \cup \mathsf{True}_0$ *and* $\mathcal{C} = \mathsf{P}$,

there is an $L(PV)$-formula $\varphi(x)$ defining a language $L \in \mathcal{C}$ such that T does not prove the sentence $\mathsf{UB}_k^{i.o.}(\varphi)$.

Our arguments are somewhat non-constructive and do not provide a single explicit formula $\varphi(x)$ in each case of the result. Informally, Theorem 1 shows (in particular) the following consistency statements:

$$T_2^1(PV) \nvdash \mathsf{P}^{\mathsf{NP}} \subseteq i.o.\mathsf{SIZE}[n^k], \quad S_2^1(PV) \nvdash \mathsf{NP} \subseteq i.o.\mathsf{SIZE}[n^k], \quad PV \nvdash \mathsf{P} \subseteq i.o.\mathsf{SIZE}[n^k].$$

---

[9]The notation $1^{(n)}$ means that n is the length of another variable. We abuse notation and use $|C_m|$ to denote the number of gates in C_m. We refer to [Pic15a] for a detailed discussion of the formalization of circuit complexity in bounded arithmetic.

In other words, there are models of these theories (satisfying a large fraction of modern complexity theory) that contain explicit problems that require circuits of size n^k on every large enough input length.[10] Another interpretation is that one can develop theories of computational complexity that postulate the existence of hard problems (as new axioms) without ever proving a contradictory statement.[11] As alluded to above, given the expressive power of these theories, we view the consistency results as evidence that such lower bounds hold in the standard mathematical universe. Nevertheless, if one strongly believes in an inclusion such as $\mathsf{NP} \subseteq \mathsf{SIZE}[n^k]$ for a large enough k, then Theorem 1 shows that even to prove this inclusion on infinitely many input lengths it will be necessary to use mathematical arguments that are beyond the reasoning capabilities of the corresponding theories.

We stress that True_0 and True_1 contain several statements of interest about algorithms, boolean circuits, extremal combinatorial objects, etc. Theorem 1 shows that even assuming such statements as axioms the corresponding theories cannot prove fixed-polynomial size circuit upper bounds.[12]

We note that the particular syntactic form of φ defining the hard language in Theorem 1 items (a) and (c) is irrelevant as long as p-time functions and predicates are defined by open formulas of the language of PV and the SAT predicate used in the argument is defined by a Σ_1^b-formula. Indeed, if two open $L(\mathsf{PV})$-formulas define the same predicate then this universal statement is included in theory True_0 and hence (c) holds identically for all open formulas defining the same language. An analogous observation applies to (a): languages in P^{NP} are definable by Δ_2^b-formulas w.r.t. the theory[13] and the universal statement stating their equivalence is thus in True_1.

Related work and techniques. Some works have investigated the *unprovability* of circuit lower bounds, or equivalently, the *consistency of upper bounds*. We refer to the introduction of [MP17] for more information about this line of work, and to Appendix A for some related remarks that might be of independent interest. Theorem 1 and our techniques are more directly connected to [CK07], [KO17], and [BM18]. We review the relevant results next.

Cook and Krajíček [CK07] (see also [Kra98]) were the first to systematically investigate the consistency of circuit lower bounds. They established several results showing that $\mathsf{NP} \not\subseteq \mathsf{SIZE}[\mathsf{poly}]$ is consistent with PV, S_2^1, and T_2^1 under appropriate assumptions regarding the collapse of PH. For instance, it was shown (in particular) that $\mathsf{T}_2^1 \nvdash \mathsf{NP} \subseteq \mathsf{SIZE}[\mathsf{poly}]$ if $\mathsf{PH} \not\subseteq \mathsf{P}^{\mathsf{NP}}$. While their results are *conditional*, [CK07] considered consistency statements for a fixed language in NP with respect to all polynomial bounds. In [KO17], two of the authors established an *unconditional* result showing that $\mathsf{PV} \nvdash \mathsf{P} \subseteq \mathsf{SIZE}[n^k]$, where k is any fixed integer. This consistency statement was subsequently improved by [BM18], who considered a more natural formalization of the statement that a language has circuits of size $O(n^k)$ and

---

[10]A bit more precisely, the lower bound holds for every input length $n \geq n_0$, where n_0 is an element of the model. Note that n_0 might be a nonstandard element of this model.

[11]One can even contemplate the possibility that more advanced consistency results might allow the development of "logic-based" cryptography: protocols that are unconditionally secure against all efficient algorithms that can be proved correct in a given theory.

[12]For instance, $\mathsf{T}_2^1(\mathsf{PV}) \cup \mathsf{True}_1$ proves the correctness of the AKS primality testing algorithm (see Footnote 7), i.e., it shows that $\forall x \, (\exists y \, (1 < y < x \wedge y \,|\, x) \leftrightarrow f_{\mathsf{AKS}}(x) = 0)$ since this sentence is in True_1. This implies that this theory proves that primality testing can be done by circuits of size n^c for a fixed c on every large enough input length n.

[13]In other words, the languages have both Σ_2^b and Π_2^b definitions that are provably equivalent in the theory.

adapted the argument [KO17] using polynomial-time ultrapowers.[14] All previous results refer to the consistency of lower bounds on infinitely many input lengths, and Theorem 1 part (c) strictly improves upon [KO17] and [BM18].[15]

In terms of techniques, the proof of Theorem 1 explores methods from complexity theory and mathematical logic to establish the unprovability of infinitely often upper bounds. We combine ideas from the conditional results of [CK07] with the unconditional approach of [KO17]. The general theme is to obtain computational information from proofs in the corresponding bounded theories. For instance, under the assumption that there is a PV-proof π that a problem in P admits *non-uniform* circuits of size n^k, we attempt to extract from π a more "uniform" construction of such circuits. The ideal plan is to contradict existing lower bounds against uniform circuits, such as those investigated in [SW14] and other works. However, as explained in [KO17], implementing this plan is not straightforward, since the "uniformity" one obtains from π does not match existing results in the area of uniform circuit lower bounds. Moreover, the proof of Theorem 1 creates additional difficulties because the uniform circuit lower bounds, already insufficient, only hold on infinitely many input lengths. In order to overcome this difficulty, we make use of further insights on the logical side of the argument. In turn, this requires appropriate extensions of the complexity-theoretic arguments.

Extensions and open problems. One can adapt the methods used in the proof of Theorem 1 to show that APC^1 and indeed theory

$$\mathsf{S}^1_2(\mathsf{PV}) \cup \mathsf{sWPHP}(\mathsf{PV}) \;\nvdash\; \mathsf{UB}^{i.o.}_k(\varphi) \;, \tag{2}$$

for some $L(\mathsf{PV})$-formula $\varphi(x)$ defining a language in $\mathsf{ZPP}^{\mathsf{NP}[O(\log n)]}$.[16] In contrast, existing (infinitely often) lower bounds for $\mathsf{ZPP}^{\mathsf{NP}}$ seem to hold only when the NP oracle is adaptively queried polynomially many times [KW98, Cai07], or with respect to non-adaptive queries but for a promise version of this class (see the discussion in [San09, Section 3.2]). There is strong evidence that asking more queries increases computational power (see [CC06, CP08] and references therein), and it is known that polynomially many non-adaptive queries to an NP oracle are equivalent in power to logarithmic many adaptive queries [Hem89, BH91]. The problem of proving super-linear circuit lower bounds for $\mathsf{ZPP}^{\mathsf{NP}}_{\mathsf{tt}}$ (i.e. $\mathsf{ZPP}^{\mathsf{NP}}$ with non-adaptive queries) was investigated recently by [DPV18], and in a sense the consistency statement in (2) addresses this question with respect to APC^1.

On the one hand, this consistency statement feels less appealing than the results in Theorem 1 due to its proximity to existing lower bounds in complexity theory. But on the other hand, it highlights the importance of APC^1 in connection to frontier questions in complexity theory

---

[14]The formalizations in [KO17] and [BM18] differ on how the $O(\cdot)$ notation is handled, and we refer to the corresponding papers for details. Here the sentences $\mathsf{UB}^{i.o.}_k(\varphi)$ refer to infinitely often upper bounds, and this issue is not relevant.

[15]In model-theoretic terms, [KO17] and [BM18] provide models where the circuit lower bound holds on some large enough input length. A slight modification of the proof in [BM18] gives a fixed model with arbitrarily large hard input lengths (Moritz Müller, private communication). On the other hand, our results provide a model where the lower bound holds on every large enough input length.

[16]This is obtained as in Theorem 1 parts (a) and (b) by proving the following "logical" Karp-Lipton collapse: If $\mathsf{S}^1_2(\mathsf{PV}) \cup \mathsf{sWPHP}(\mathsf{PV}) \vdash \mathsf{NP} \subseteq \mathsf{SIZE}[\mathsf{poly}]$ then PH collapses to $\mathsf{ZPP}^{\mathsf{NP}[O(\log n)]}$. The proof of the latter adapts the argument in [CK07, Theorem 5.1 (*ii*)], using randomization to obtain witnesses for the required dWPHP axioms and an NP oracle to check that they are correct. (A bit more formally, the idea is to first Skolemize the theory, reducing the argument to the case of $\mathsf{S}^1_2(\mathsf{PV})$, then to handle the newly introduced function symbols by witnessing them in the standard model through a probabilistic computation with an NP oracle.)

and lower bounds. As one of our main open problems, we ask for the proof of stronger consistency results for the theory APC^1. For instance, can one show that $\mathsf{APC}^1 \nvdash \mathsf{MA} \subseteq \mathsf{SIZE}[n^k]$, partially addressing the use of non-uniform advice in [San09]? In connection to this and related problems, it might be fruitful to investigate a potential extension of the equivalence in [CMMW19] to a result that relates consistency statements, witnessing theorems, and logical Karp-Lipton theorems.

It would also be interesting to improve our consistency results for S_2^1 and T_2^1 with respect to the uniformity of the hard problems, and to establish a non-trivial statement about the consistency of circuit lower bounds with T_2^2 (Theorem 1 part (a) extends to S_2^2 using a similar argument and appropriate results from [CK07]).

We include in Appendix A a discussion on the consistency of $\mathsf{P} \neq \mathsf{NP}$ and its connection to the unprovability of circuit lower bounds.

2   Background and notation

In order to emphasize the main ideas, we assume some familiarity with logic, bounded arithmetic, and complexity theory. Everything needed can be found in [Kra95]. The interested reader can consult [CN10] for a more recent reference in bounded arithmetic, [Bus97] for a concise introduction, and [Pud13] for an accessible exposition. For more background in circuit complexity, we refer to [Juk12]. For a discussion of the formalization of complexity theory and circuit complexity in bounded arithmetic, see [MP17] and references therein.

Our proofs will rely on some results and arguments from [CK07] and [KO17], and we refer to the detailed presentation in these papers instead of repeating the proofs here. In more detail, what is needed from [CK07] is that some or their theorems can be modified to include True_0 or True_1. On the other hand, the proof of Theorem 1 (c) can only be followed if the reader is familiar with the simpler argument from [KO17].

We use $\mathsf{P}^{\mathsf{NP}[\ell(n)]}$ to denote the set of languages decided by a deterministic polynomial time machine that makes at most $\ell(n)$ queries to an NP oracle. We will assume without loss of generality that the oracle is some fixed NP-complete language such as formula satisfiability.

3   Consistency of lower bounds with bounded arithmetic

This section proves Theorem 1. Let k be a positive integer. We argue in each item as follows.

(a) We consider two cases. If the polynomial hierarchy PH collapses to P^{NP}, then we can define a language $L \in \mathsf{P}^{\mathsf{NP}}$ such that $L \notin i.o.\mathsf{SIZE}[n^k]$. More precisely, L computes on input length n as the lexicographic first truth-table corresponding to a function $h \colon \{0,1\}^n \to \{0,1\}$ that cannot be computed by circuits of size n^k. This language can be easily specified using a constant number of quantifiers over strings of length $\mathsf{poly}(n)$ (cf. [Kan82]). By the equivalence between languages in Σ_i^p and predicates definable by $\Sigma_i^b(\mathsf{PV})$ formulas (see e.g. [Kra95, Theorem 3.2.12]), there is an $L(\mathsf{PV})$-formula $\varphi_L(x)$ that defines L (using the correspondence between $\{0,1\}^*$ and $\mathbb{N}$). Since $\mathsf{T}_2^1(\mathsf{PV}) \cup \mathsf{True}_1$ is sound and L is hard on every large enough input length, this theory cannot prove the sentence $\mathsf{UB}_k^{i.o.}(\varphi_L)$.

Assume now that PH does not collapse to P^{NP}. Let $\varphi_{\mathsf{SAT}}(x)$ be a $\Sigma_1^b(\mathsf{PV})$-formula that

defines the formula satisfiability problem (SAT). We take a particular formulation of $\varphi_{\mathsf{SAT}}(x)$ for which the input encoding is paddable, meaning that inputs of the satisfiability problem of length $\ell < m$ can be easily converted into equivalent inputs of length m. If $\mathsf{T}_2^1(\mathsf{PV}) \cup \mathsf{True}_1$ does not prove $\mathsf{UB}_k^{i.o.}(\varphi_{\mathsf{SAT}})$, we are done, given that this formula defines a language in $\mathsf{NP} \subseteq \mathsf{P}^{\mathsf{NP}}$. Suppose $\mathsf{T}_2^1(\mathsf{PV}) \cup \mathsf{True}_1 \vdash \mathsf{UB}_k^{i.o.}(\varphi_{\mathsf{SAT}})$. This formula has unbounded existential quantifiers, but since $\mathsf{T}_2^1(\mathsf{PV}) \cup \mathsf{True}_1$ is axiomatized by bounded formulas, Parikh's theorem (cf. [Kra95, Section 5.1]) implies that there is an $L(\mathsf{PV})$-term $t(x)$ such that $\mathsf{UB}_k^{i.o.}(\varphi_{\mathsf{SAT}})$ is provable in the theory even if the existential quantifiers are bounded by $t(1^{(n)})$. In particular, m and $|C_m|$ in Equation (1) can be bounded as

$$m, \ |C_m| \ \leq \ n^{O(1)} \ . \tag{3}$$

By our assumption on paddability, a circuit C_m deciding satisfiability on formulas encoded using m bits also works for all formulas of length $n \leq m$. But by (3), $|C_m| \leq n^{O(1)}$ and hence C_m can serve as a polynomial size circuit solving SAT on formulas of size n. Consequently, if $\mathsf{T}_2^1(\mathsf{PV}) \cup \mathsf{True}_1$ proves that SAT is infinitely often in $\mathsf{SIZE}[n^k]$ it also proves that SAT is in $\mathsf{SIZE}[\mathsf{poly}(n)]$. We now invoke the argument of [CK07, Theorem 5.1 (iii)] who showed (in particular) that if T_2^1 proves that SAT $\in \mathsf{SIZE}[\mathsf{poly}]$ then PH collapses to P^{NP}. Their proof can be adapted to $\mathsf{T}_2^1(\mathsf{PV}) \cup \mathsf{True}_1$, since all sentences in True_1 are witnessed by $\mathsf{FP}^{\mathsf{NP}}$ functions and adding these sentences as new axioms does not affect the required witnessing theorem.[17] This collapse of PH is in contradiction to our assumption in this case of the proof, which completes the argument.

(b) Consider the formula $\varphi_{\mathsf{SAT}}(x)$ defined in item (a) above. If $\mathsf{S}_2^1 \cup \mathsf{True}_0 \nvdash \mathsf{UB}_k^{i.o.}(\varphi_{\mathsf{SAT}})$ there is nothing else to prove. Otherwise, by the same argument via Parikh's Theorem it follows that $\mathsf{S}_2^1 \cup \mathsf{True}_0$ proves that SAT admits polynomial size circuits on every input length. Now the argument in [CK07, Theorem 5.1 (ii)] (easily modifiable to handle True_0 because axioms in it are universal sentences) implies that every language $L \in \mathsf{PH}$ is also in $\mathsf{P}^{\mathsf{NP}[c \cdot \log n]}$ for some $c \in \mathbb{N}$. In particular, every such language is in $\mathsf{P}^{\mathsf{NP}[n]}$. Consequently, by Kannan's construction [Kan82] there is a language $L_{\mathsf{hard}} \in \mathsf{P}^{\mathsf{NP}[n]}$ such that $L_{\mathsf{hard}} \notin i.o.\mathsf{SIZE}[n^{k+2}]$. We will need the following lemma.[18]

Lemma 1. *If* $\mathsf{NP} \subseteq i.o.\mathsf{SIZE}[n^k]$ *then* $\mathsf{P}^{\mathsf{NP}[n]} \subseteq i.o.\mathsf{SIZE}[n^{k+2}]$.

Proof. Let L be a language in $\mathsf{P}^{\mathsf{NP}[n]}$ decided by a deterministic polynomial-time oracle machine M running in time at most $q(n)$. For convenience, we assume without loss of generality that M makes *exactly* n queries before accepting or rejecting an input string, regardless of the answers provided by its NP oracle O.

We consider the language L_{aux} containing all tuples $(a, j, b_1, \ldots, b_n, 1^{(t)}, c)$, where $|a| = n$, $1 \leq j \leq n$, each $b_i \in \{0, 1\}$, $t \in \mathbb{N}$ is a padding parameter, and $c \in \{0, 1\}$ is a control bit, which satisfy the following conditions:

---

[17] In more detail, adding a function symbol for these witnessing functions turns sentences from True_1 into universal sentences, and universal sentences do not influence witnessing theorems. For example, if $\forall x \exists y \ (y \leq s(x) \wedge A(x, y))$ is in True_1 and f is the symbol for the associated witnessing function, the universal sentence will be $\forall x \ (f(x) \leq s(x) \wedge A(x, f(x)))$.

[18] The proof of the lemma uses ideas from the proof of [Kra93, Proposition 1.3] showing that S_2^1 can define all $\mathsf{FP}^{\mathsf{NP}[wit, O(\log n)]}$ functions, extending a proof from [Bus86] that T_2^1 can define all $\mathsf{FP}^{\mathsf{NP}}$ functions. A similar argument was employed in the proof of [FSW09, Theorem 10] (without the infinitely often condition).

– If $c = 0$, then when M computes on a and its first j queries are answered according to $b_1, \ldots, b_j$, for each $i \leq j$ if $y_i \in \{0, 1\}^\star$ is the i-th query and $b_i = 1$ we have $y_i \in O$.

– If $c = 1$, the machine M accepts a within $q(n)$ steps under oracle answers b_i for $1 \leq i \leq n$.

Since $O \in \mathsf{NP}$ and M is a deterministic polynomial time machine, $L_{\mathsf{aux}} \in \mathsf{NP}$. Using the hypothesis of the lemma, for infinitely many values of n there exists $t \leq 10n$ and a circuit D_n for L_{aux} of size at most $C(n + \log n + n + t + 1)^k \leq n^{k+1}$ (for large enough n) that decides L_{aux} with respect to our parameter n (the input length for an instance of L). Note that the parameter t allows us to hit the "good" input lengths without technical considerations about the input encoding employed in the definition of L_{aux}.

We will use D_n (with the correct value t non-uniformly hardcoded in the input) as a subroutine in order to solve L on inputs of length n, as described next. First, we recover the correct oracle answers $d_1, \ldots, d_n$ for a given input string a. This is done in n steps, where the i-th step recovers d_i. To recover d_1, we use D_n to compute

$$D_n(a, \overset{j}{1}, \overbrace{1, \star, \ldots, \star}^{\vec{b}}, 1^{(t)}, \overset{c}{0}),\tag{4}$$

where each $\star$ can be replaced by an arbitrary bit. If the output is 1, the first query made by M on a has a positive answer with respect to O (since positive queries must be strings in O by the definition of tuples in L_{aux} when $c = 0$). Otherwise, we must have $d_1 = 0$. Next, we invoke

$$D_n(a, \overset{j}{2}, \overbrace{d_1, 1, \star, \ldots, \star}^{\vec{b}}, 1^{(t)}, \overset{c}{0}),\tag{5}$$

knowing that the answer to the first query is correct. By the same argument, we are able to recover d_2, and proceeding similarly, we can recover all correct answers $d_1, \ldots, d_n$. Finally, by invoking $D_n(a, n, d_1, \ldots, d_n, 1^{(t)}, 1)$ with $c = 1$ and using the correct oracle answers, we can decide if $a \in L$. Clearly, this entire computation can be performed by a circuit of size at most $O(n \cdot |D_n|) = O(n^{k+2})$, which completes the proof. $\square$

It follows from Lemma 1 and the properties of L_{hard} that there is a language $L \in \mathsf{NP}$ such that $L \notin i.o.\mathsf{SIZE}[n^k]$. Consequently, if $\varphi_L(x)$ is a formula that defines L then $\mathsf{S}_2^1 \cup \mathsf{True}_0 \nvdash \mathsf{UB}_k^{i.o.}(\varphi_L)$. This completes the proof of item (b).

(c) We follow the overall strategy of the proof of [KO17, Theorem 2.1] (which combines the proof of [SW14, Theorem 1.1] with other ideas), but the infinitely often statement considered here introduces certain difficulties. In particular, it is not clear how to adapt the proof in [SW14] to show that P is not contained infinitely often in P-uniform $\mathsf{SIZE}[n^k]$. In general, combining different computations that succeed infinitely often might not produce a computation that succeeds infinitely often. We explain below how the argument from [KO17] can be modified to establish the stronger statement in part (c). (For simplicity of notation, we restrict our discussion to PV, but the argument works for $\mathsf{PV} \cup \mathsf{True}_0$ as well.)

Let $g_{k'}$ for $k' = 3k$ be the PV function symbol provided by [KO17, Lemma 3.1]. Recall that PV proves that any uniform algorithm h running in time at most $n^{k'-1}$ will fail to compute $g_{k'}$, even if h is given a certain amount of advice that can depend on the input length. If $\mathsf{PV} \nvdash \mathsf{UB}_k^{i.o.}(g_{k'})$ we are done. Otherwise, applying the KPT Theorem (see e.g. [KO17, Theorem 4.1]) to sentence $\mathsf{UB}_k^{i.o.}(g_{k'})$ (note crucially that $\mathsf{UB}_k^{i.o.}(g_{k'})$ has the right quantifier complexity),

we obtain a fixed $r \in \mathbb{N}$ (independent of n) and PV function symbols $f_1, \ldots, f_r$ such that on input $1^{(n)}$ each function f_i outputs $n \leq n_i \leq n^{a_i}$ (represented as $1^{(n_i)}$) and a circuit $C_{n_i}^i$ of size at most n_i^k that is a candidate circuit for $g_{k'}$ on inputs of length n_i (the upper bound n^{a_i} is provable in PV). As usual in applications of the KPT Theorem, each function f_i in addition to $1^{(n)}$ might also depend on potential counter-examples to the correctness of the pairs $(n_j, C_{n_j}^j)$ for $j < i$. In other words, from the provability of $\mathsf{UB}_k^{i.o.}(g_{k'})$ theory PV proves the universal closure of the following disjunction:[19]

$$[f_1(1^{(n)}) = (1^{(n_1)}, C_{n_1}^1) \wedge |C_{n_1}^1| \leq n_1^k \wedge (|x_1| = n_1 \rightarrow C_{n_1}^1(x_1) = g_{k'}(x_1))] \vee$$

$$[f_2(1^{(n)}, x_1) = (1^{(n_2)}, C_{n_2}^2) \wedge |C_{n_2}^2| \leq n_2^k \wedge (|x_2| = n_2 \rightarrow C_{n_2}^2(x_2) = g_{k'}(x_2))] \vee \qquad (6)$$

$$\ldots \vee [f_r(1^{(n)}, x_1, \ldots, x_{r-1}) = (1^{(n_r)}, C_{n_r}^r) \wedge |C_{n_r}^r| \leq n_r^k \wedge (|x_r| = n_r \rightarrow C_{n_r}^r(x_r) = g_{k'}(x_r))] .$$

Modifying the strategy of [KO17], we argue that either $\mathsf{PV} \nvdash \mathsf{UB}_k^{i.o.}(\widetilde{f}_1)$ for a certain PV function symbol $\widetilde{f}_1$ that depends on f_1 (we are done in this case), or PV proves that the circuit $C_{n_1}^1$ output by $f_1(1^{(n)})$ does not succeed in computing $g_{k'}$ on inputs of length $n_1 = n_1(1^{(n)})$ for *infinitely many values of* n. It will be important that such values of n are polynomially gapped, and that an infinite set S_1 of strings of the form $1^{(n)}$ corresponding to them can be enumerated by a PV function symbol $u_1(1^{(\ell)})$. This allows us to eliminate one disjunct in the sentence obtained from the KPT Theorem if we quantify not over $1^{(n)}$ for all n but just over strings $1^{(n)}$ in the image of $u_1(1^{(\ell)})$,[20] since on these specific $1^{(n)}$ the function f_1 never succeeds in generating a circuit that correctly computes $g_{k'}$ on input length $n_1 = n_1(1^{(n)})$, and in addition (as we explain below) there is a PV function symbol that provably produces counter-examples. The proof of our result can be completed by iterating the argument r times while focusing on the relevant input lengths. The idea is similar in spirit to [KO17], but the argument is more involved because intuitively we need to consider a chain $S_r \subseteq \ldots \subseteq S_1$ of infinite sets of input parameters: If $j \leq i$ then PV proves that function f_j from the KPT disjunction (with appropriate counter-examples) does not succeed on $1^{(n)} \in S_i$ (assuming the provability of certain auxiliary sentences $\mathsf{UB}_k^{i.o.}(\widetilde{f}_j)$). We provide the details next.

Recall that in the terminology of [KO17] the function symbol $\widetilde{f}$ decides L_{succ}, a padded version of the language L_{dc} encoding the direct connection language of the circuits generated by f. Our definition of $\widetilde{f}_1$ is analogous to the construction in [KO17], but we need to change the amount of padding in order to accommodate the new setting. Here $f_1(1^{(n)})$ might generate candidate circuits for $g_{k'}$ on larger input lengths. Moreover, the circuits for $\widetilde{f}_1$ obtained from the provability of $\mathsf{UB}_k^{i.o.}(\widetilde{f}_1)$ are only guaranteed to work infinitely often. Handling these complications in the case of f_1 (and in subsequent cases) will be possible because $g_{k'}$ is hard on *every large enough input length* and the relevant input lengths ($n_1 = n_1(1^{(n)}) \leq n^{a_1}$ in the case of f_1) are *provably computable in polynomial time*.

In more detail, let L_{dc}^1 encode the direct connection language of the sequence of circuits $C_{n_1}^1$ on $n_1 \leq n^{a_1}$ input bits produced by $f_1(1^{(n)})$. Similarly to [KO17], our language L_{succ}^1 will be a succinct version of L_{dc}^1. This time we compress the tuples encoding $C_{n_1}^1$ to:

$$\langle \mathsf{Bin}(n), 1^{(n^{1/10k})}, u, v, w, 1^t \rangle, \qquad (7)$$

---

[19] For simplicity of notation, we left out in each row of (6) the condition $n_i \geq n$, for $i = 1, 2, \ldots, r$.

[20] As opposed to [KO17], which focuses on larger input lengths after each iteration of the argument.

where crucially t is arbitrary. (The parameter t is needed in connection to an infinitely often circuit upper bound for L^1_{succ}, since it makes this language paddable. The use of t here is different than in [KO17], where it appears only for convenience and as a function of other input parameters.) Under our assumptions, a p-time algorithm $\widetilde{f}_1$ deciding L^1_{succ} can be defined in PV. Suppose that $\mathsf{PV} \vdash \mathsf{UB}^{i.o.}_k(\widetilde{f}_1)$. Recall that, assuming $C^1_{n_1}$ is a correct circuit for $g_{k'}$, a small circuit for $\widetilde{f}_1$ allows one to obtain a short advice string representing a circuit that decides the tuples of $C^1_{n_1}$, which in turn allows us to compute $g_{k'}$ in time $\ll n_1^{k'-1}$. Arguing in PV and adapting the proof of [KO17, Lemma 3.2] in the natural way (i.e. by padding t appropriately and using the almost-everywhere hardness of $g_{k'}$), it follows that for infinitely many choices of $1^{(n)}$, $C^1_{n_1}$ does not compute $g_{k'}$ on inputs of length $n_1 = n_1(1^{(n)})$. Equivalently,

$$\mathsf{PV} \vdash \forall 1^{(\ell)} \exists 1^{(n)} (n \geq \ell) \exists x (|x| = n_1(1^{(n)})),\ g_{k'}(x) \neq C^1_{|x|}(x) . \tag{8}$$

Using Herbrand's Theorem and in analogy to [KO17, Lemma 3.2], there are PV function symbols u_1 and e_1 witnessing these existential quantifiers. Furthermore, provably in PV we have $|u_1(1^{(\ell)})| \leq \ell^{c_1}$ for some constant c_1. Therefore, we can take S_1 as the infinite set of strings $1^{(n)}$ obtained from $u_1(1^{(\ell)})$ over all choices of ℓ, and $e_1(1^{(\ell)})$ witnesses that the corresponding circuits $C^1_{n_1}$ are incorrect over the associated input lengths $n_1 = n_1(1^{(n)})$.

The formula obtained from our initial application of the KPT Theorem to $\mathsf{UB}^{i.o.}_k(g_{k'})$ can now be simplified in PV to a formula equivalent to:

$$\forall 1^{(n)} \in S_1,\ \text{"KPT disjunct for } j \in [2, r] \text{ under the counter-example } x_1 \overset{\text{def}}{=} e_1(1^{(\ell)}) \text{"}, \tag{9}$$

where the quantifier $\forall 1^{(n)} \in S_1$ is expressed in PV by "$\forall 1^{(\ell)} \forall 1^{(n)}$ such that $1^{(n)} = u_1(1^{(\ell)})$". A bit more precisely, the second and later disjuncts in the KPT expression (6) contain functions f_i for $i > 1$ depending on $1^{(n)}$ and on each x_j for which $j < i$, where the x_j are the variables for counter-examples to the correctness of circuits $C^j_{|x_j|}$. Now substitute everywhere $1^{(n)} = u_1(1^{(\ell)})$ and $x_1 = e_1(1^{(\ell)})$. By the choice of u_1 and e_1, this substitution provably falsifies the first disjunct and also $n_1(1^{(n)}) \geq n \geq \ell$. Hence (6) is turned into a KPT expression with $r - 1$ disjuncts, i.e.:

$$[f_2(u_1(1^{(\ell)}), e_1(1^{(\ell)})) = (1^{(n_2)}, C^2_{n_2}) \wedge |C^2_{n_2}| \leq n_2^k \wedge (|x_2| = n_2 \to C^2_{n_2}(x_2) = g_{k'}(x_2))] \vee \ldots \tag{10}$$

$$\vee\, [f_r(u_1(1^{(\ell)}), e_1(1^{(\ell)}), x_2, \ldots, x_{r-1}) = (1^{(n_r)}, C^r_{n_r}) \wedge |C^r_{n_r}| \leq n_r^k \wedge (|x_r| = n_r \to C^r_{n_r}(x_r) = g_{k'}(x_r))],$$

where for convenience of notation we have omitted the conditions $n_i \geq \ell$, for $i = 2, \ldots, r$. One can also replace $f_2(u_1(1^{(\ell)}), e_1(1^{(\ell)}))$ by an equivalent term in the language of PV, say, $f'_1(1^{(\ell)})$, and similarly for each f_i appearing in the expression above. We have therefore eliminated one disjunct from the formula appearing in Equation (6).

The result is proved as in [KO17] by iterating this argument in the natural way until some derived sentence $\mathsf{UB}^{i.o.}_k(\widetilde{f}_i)$ is unprovable or one eliminates all disjuncts. The latter case leads to a contradiction. (Intuitively, the sets S_i contain infinitely many elements and on every string $1^{(n)}$ one of the functions obtained from the initial KPT disjunction must succeed when given appropriate counter-examples. Eliminating all disjuncts contradicts the formula obtained from KPT witnessing, or more precisely, one of the subsequent formulas derived from it in the argument presented above.)

For instance, in the case we get $r = 2$ after the application of the KPT Theorem, assuming that PV also proves $\mathsf{UB}^{i.o.}_k(\widetilde{f'_1})$, and arguing identically as before, we now get functions u'_1 and

e_1' computing witnesses (lengths and inputs) that circuits provided by f_1' fail infinitely often to compute $g_{k'}$.[21] This is contradictory, because this time the formula from Equation (10) claims that f_1' must succeed (recall that $f_1'(1^{(\ell)}) = f_2(u_1(1^{(\ell)}), e_1(1^{(\ell)}))$) as there are no more disjuncts if $r = 2$.

This completes the proof of Theorem 1 item (c).

Remark. We note that in Theorem 1 it is possible to *syntactically* enforce the language L to be in the class $\mathcal{C}$ from the formula $\varphi(x)$ and theory T. In general, this follows from the definability of these languages in the corresponding theories by formulas of appropriate complexity (see e.g. [Bus97, Section 2.6]). In more detail, for part (a), as we observed in the concluding remarks of Section 1, the consistency result extends to theory $\mathsf{S}_2^2(\mathsf{PV})$. In this case, a language in P^{NP} is definable in the theory via two provably equivalent Σ_2^b and Π_2^b formulas. (The provability of the equivalence needs to be done in $\mathsf{S}_2^2(\mathsf{PV})$.) For part (b), the corresponding language L is definable in $\mathsf{S}_2^1(\mathsf{PV})$ by a $\Sigma_1^b(\mathsf{PV})$ formula. Lastly, for part (c) the proof presented above already implies the claim, since the language is given by a PV function symbol. Note that in parts (b) and (c) provability in the theory is not necessary: the syntactic form of the formula (i.e. $\Sigma_1^b(\mathsf{PV})$ and atomic PV formula, respectively) imply that the language is in the corresponding class.

Acknowledgements

References

[AKS02] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. PRIMES is in P. *Ann. of Math.*, 2:781–793, 2002.

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

[BFS09] Harry Buhrman, Lance Fortnow, and Rahul Santhanam. Unconditional lower bounds against advice. In *International Colloquium on Automata, Languages and Programming* (ICALP), pages 195–209, 2009.

[BFT98] Harry Buhrman, Lance Fortnow, and Thomas Thierauf. Nonrelativizing separations. In *Conference on Computational Complexity* (CCC), pages 8–12, 1998.

[BH91] Samuel R. Buss and Louise Hay. On truth-table reducibility to SAT. *Inf. Comput.*, 91(1):86–102, 1991.

---

[21]Complementing our initial informal discussion, while the set S_1 is the range of u_1, the set $S_2 \subseteq S_1$ is the range of the composed map $u_1 \circ u_1'$.

[BKZ15] Samuel R. Buss, Leszek Kołodziejczyk, and Konrad Zdanowski. Collapsing modular counting in bounded arithmetic and constant depth propositional proofs. *Transactions of the American Mathematical Society*, 367(11):7517–7563, 2015.

[BM18] Jan Bydžovský and Moritz Müller. Polynomial time ultrapowers and the consistency of circuit lower bounds. *Archive for Mathematical Logic (to appear)*, 2018.

[Bus86] Samuel R. Buss. *Bounded arithmetic*, volume 86. Bibliopolis, 1986.

[Bus97] Samuel R. Buss. Bounded arithmetic and propositional proof complexity. In *Logic of computation*, pages 67–121. Springer, 1997.

[Cai07] Jin-yi Cai. S_2^p is a subset of $\mathsf{ZPP}^{\mathsf{NP}}$. *J. Comput. Syst. Sci.*, 73(1):25–35, 2007.

[CC06] Jin-yi Cai and Venkatesan T. Chakaravarthy. On zero error algorithms having oracle access to one query. *J. Comb. Optim.*, 11(2):189–202, 2006.

[CK07] Stephen A. Cook and Jan Krajícek. Consequences of the provability of $NP \subseteq P/poly$. *J. Symb. Log.*, 72(4):1353–1371, 2007.

[CMMW19] Lijie Chen, Dylan M. McKay, Cody D. Murray, and Ryan Williams. Relations and equivalences between circuit lower bounds and Karp-Lipton theorems. In *Computational Complexity Conference* (CCC), 2019.

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

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

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

[CP08] Richard Chang and Suresh Purini. Amplifying $\mathsf{ZPP}^{\mathsf{sat}[1]}$ and the two queries problem. In *Conference on Computational Complexity* (CCC), pages 41–52, 2008.

[DPV18] Peter Dixon, Aduri Pavan, and N. V. Vinodchandran. On pseudodeterministic approximation algorithms. In *Symposium on Mathematical Foundations of Computer Science* (MFCS), pages 61:1–61:11, 2018.

[FGHK16] Magnus Gausdal Find, Alexander Golovnev, Edward A. Hirsch, and Alexander S. Kulikov. A better-than-$3n$ lower bound for the circuit complexity of an explicit function. In *Symposium on Foundations of Computer Science* (FOCS), *USA*, pages 89–98, 2016.

[FS17] Lance Fortnow and Rahul Santhanam. Robust simulations and significant separations. *Inf. Comput.*, 256:149–159, 2017.

[FSW09] Lance Fortnow, Rahul Santhanam, and Ryan Williams. Fixed-polynomial size circuit bounds. In *Conference on Computational Complexity* (CCC), pages 19–26, 2009.

[GZ11] Oded Goldreich and David Zuckerman. Another proof that BPP ⊆ PH (and more). In *Studies in Complexity and Cryptography*, pages 40–53. 2011.

[Hem89] Lane A. Hemachandra. The strong exponential hierarchy collapses. *J. Comput. Syst. Sci.*, 39(3):299–322, 1989.

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

[Jeř05] Emil Jeřábek. Weak pigeonhole principle, and randomized computation. *Ph.D. Thesis, Charles University in Prague*, 2005.

[Jeř06] Emil Jeřábek. The strength of sharply bounded induction. *Mathematical Logic Quarterly*, 52(6):613–624, 2006.

[Jeř07] Emil Jeřábek. Approximate counting in bounded arithmetic. *J. Symb. Log.*, 72(3):959–993, 2007.

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

[Juk12] Stasys Jukna. *Boolean Function Complexity - Advances and Frontiers*. Springer, 2012.

[Kan82] Ravi Kannan. Circuit-size lower bounds and non-reducibility to sparse sets. *Information and Control*, 55(1-3):40–56, 1982.

[KO17] Jan Krajíček and Igor Carboni Oliveira. Unprovability of circuit upper bounds in Cook's theory PV. *Logical Methods in Computer Science*, 13(1), 2017.

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

[Kra93] Jan Krajíček. Fragments of bounded arithmetic and bounded query classes. *Transactions of the American Mathematical Society*, 338(2):587–598, 1993.

[Kra95] Jan Krajíček. *Bounded Arithmetic, Propositional Logic, and Complexity Theory*. Cambridge University Press, 1995.

[Kra98] Jan Krajíček. Extensions of models of PV. *Lecture Notes in Logic*, 11:104–114, 1998.

[KW98] Johannes Köbler and Osamu Watanabe. New collapse consequences of NP having small circuits. *SIAM J. Comput.*, 28(1):311–324, 1998.

[LC11] Dai Tri Man Le and Stephen A. Cook. Formalizing randomized matching algorithms. *Logical Methods in Computer Science*, 8(3), 2011.

[Le14]   Dai Tri Man Le. Bounded arithmetic and formalizing probabilistic proofs. *Ph.D. Thesis, University of Toronto*, 2014.

[Lip94]   Richard J. Lipton. Some consequences of our failure to prove non-linear lower bounds on explicit functions. In *Structure in Complexity Theory Conference* (CCC), pages 79–87, 1994.

[MP17]   Moritz Müller and Ján Pich. Feasibly constructive proofs of succinct weak circuit lower bounds. *Electronic Colloquium on Computational Complexity* (ECCC), 24:144, 2017.

[Mul99]   Ketan Mulmuley. Lower bounds in a parallel model without bit operations. *SIAM J. Comput.*, 28(4):1460–1509, 1999.

[MW18]   Cody Murray and R. Ryan Williams. Circuit lower bounds for nondeterministic quasi-polytime: an easy witness lemma for NP and NQP. In *Symposium on Theory of Computing* (STOC), pages 890–901, 2018.

[Oja04]   Kerry Ojakian. Combinatorics in bounded arithmetic. *Ph.D. Thesis, Carnegie Mellon University*, 2004.

[OS17]   Igor Carboni Oliveira and Rahul Santhanam. Pseudodeterministic constructions in subexponential time. In *Symposium on Theory of Computing* (STOC), pages 665–677, 2017.

[Pic14]   Ján Pich. Complexity theory in feasible mathematics. *Ph.D. Thesis, Charles University in Prague*, 2014.

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

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

[Pud13]   Pavel Pudlák. *Logical Foundations of Mathematics and Computational Complexity - A Gentle Introduction.* Springer, 2013.

[Raz95]   Alexander A. Razborov. Bounded arithmetic and lower bounds in boolean complexity. In *Feasible Mathematics II*, pages 344–386. Springer, 1995.

[Ros10]   Benjamin Rossman. Average-case complexity of detecting cliques. *Ph.D. Thesis, MIT*, 2010.

[San09]   Rahul Santhanam. Circuit lower bounds for Merlin-Arthur classes. *SIAM J. Comput.*, 39(3):1038–1061, 2009.

[SW14]   Rahul Santhanam and Ryan Williams. On uniformity and circuit lower bounds. *Computational Complexity*, 23(2):177–205, 2014.

[Vin05]   N. V. Vinodchandran. A note on the circuit complexity of PP. *Theor. Comput. Sci.*, 347(1-2):415–418, 2005.

[WP87] A. J. Wilkie and Jeff B. Paris. On the scheme of induction for bounded arithmetic formulas. *Ann. Pure Appl. Logic*, 35:261–302, 1987.

A Consistency of $\mathsf{P} \neq \mathsf{NP}$ from unprovability of lower bounds

Imagine that against most expectations P is actually equal to NP and there is a polynomial time algorithm f (i.e. a PV function symbol) that finds a satisfying assignment for all satisfiable formulas. In other words, if $\psi_{\mathsf{SAT}}(x, y)$ denotes an $L(\mathsf{PV})$-formula that checks if y satisfies the formula encoded by x, then the sentence

$$\varphi_{\mathsf{P}=\mathsf{NP}}(f) \stackrel{\text{def}}{=} \forall x \, \forall y \, [\psi_{\mathsf{SAT}}(x, y) \to \psi_{\mathsf{SAT}}(x, f(x))] \tag{11}$$

is true in the standard model. Now suppose that in order to prove the universal statement $\varphi_{\mathsf{P}=\mathsf{NP}}(f)$ in Equation (11) you have to use concepts (definitions, predicates, etc.) that cannot be defined as polynomial-time algorithms. To be more specific, assume that (11) is provable using induction for non-deterministic polynomial-time algorithms (corresponding to theory $\mathsf{T}_2^1(\mathsf{PV})$), but not using induction for polynomial-time algorithms only (corresponding to theory PV). Could we still maintain that the mere existence of f implies that the satisfiability problem is "feasible"?

This question is more philosophical than mathematical, and we are not going to offer an answer. Instead, we suggest to consider a strictly mathematical question.

Conjecture 1. *For no polynomial-time algorithm f theory PV proves the sentence $\varphi_{\mathsf{P}=\mathsf{NP}}(f)$.*

Informally, Conjecture 1 states that PV and by standard conservation results S_2^1 are both consistent with $\mathsf{P} \neq \mathsf{NP}$. That is, either $\mathsf{P} \neq \mathsf{NP}$ as often assumed, and hence the conjecture is trivially true, or $\mathsf{P} = \mathsf{NP}$ but you cannot prove it using only polynomial-time concepts and reasoning. For this reason, Conjecture 1 is a formal weakening of the conjecture that $\mathsf{P} \neq \mathsf{NP}$.

We do not claim any originality for the conjecture; not only it follows from $\mathsf{P} \neq \mathsf{NP}$ but the statement is also known to follow from the conjectures that bounded arithmetic does not collapse to PV or that the Extended Frege propositional proof system is not polynomially bounded. The conjecture must have been also one of the ideas leading Stephen Cook to his seminal paper [Coo75]. We think it is a weakening of the P vs. NP conjecture that has an intrinsic relevance to it, and that it ought to be studied more (cf. [CK07] for more discussion).

In this appendix, we observe that Conjecture 1 is related to the *unprovability of circuit lower bounds*. For a PV function symbol h and a circuit size parameter $k \in \mathbb{N}$, consider the sentence

$$\mathsf{LB}_k^{a.e.}(h) \stackrel{\text{def}}{=} \neg \mathsf{UB}_k^{i.o.}(h) \, , \tag{12}$$

where $\mathsf{UB}_k^{i.o.}(h)$ is the sentence from Equation (1). Intuitively, $\mathsf{LB}_k^{a.e.}(h)$ states that the language defined by h is hard on input length m for circuits of size m^k whenever $m \geq n$, for a fixed value n.

Theorem 2 (Consistency of lower bounds with PV from the unprovability of lower bounds)**.** *If there exists $k \in \mathbb{N}$ such that for no function symbol h theory PV proves the sentence $\mathsf{LB}_k^{a.e.}(h)$, then Conjecture 1 holds.*

Note that the hypothesis of Theorem 2 is weaker than the assumption that PV does not prove that $\mathsf{NP} \not\subseteq \mathsf{SIZE}[n^k]$ for some k. Roughly speaking, Theorem 2 shows that if PV does not prove circuit lower bounds then $\mathsf{P} \neq \mathsf{NP}$ is consistent with PV.

Sketch of the proof of Theorem 2. The argument proceeds in the contrapositive. We formalize in PV the result that if $\mathsf{P} = \mathsf{NP}$ then for each parameter k, $\mathsf{P} \not\subseteq i.o.\mathsf{SIZE}[n^k]$ (see e.g. [Lip94, Theorem 3]). Recall that this is obtained by combining the collapse of PH to P together with Kannan's argument [Kan82] showing that PH can define languages that are almost-everywhere hard against circuits of fixed-polynomial size. The usual proof of this claim shows via a counting argument the existence of a truth-table of size 2^n that is hard against circuit size n^k. A potential issue is that this result might not be available in PV.

We overcome this difficulty as follows. From the provability in PV that $\mathsf{P} = \mathsf{NP}$, it follows that the hierarchy $\mathsf{T}_2(\mathsf{PV})$ of bounded arithmetic theories $\mathsf{T}_2^i(\mathsf{PV})$ collapses to PV [KPT91]. Recall that the surjective weak pigeonhole principle sWPHP for PV function symbols is provable in $\mathsf{T}_2^2(\mathsf{PV})$ (see e.g. [Kra95]). Define a PV function symbol g that takes as input a circuit C of size n^k and outputs the first n^{k+1} bits of the truth-table computed by C. From $\mathsf{sWPHP}(g)$ we now derive in PV that the prefix of some truth-table is not computable by circuits of size n^k, if n is sufficiently large. We can (implicitly) extend the lexicographic first truth-table prefix satisfying this property with zeroes, and use the resulting truth-table to define a PV-formula $\varphi(x)$ with a constant number of bounded quantifiers that defines a language L that is hard against circuits of size n^k, where the hardness is provable in PV. Since the provability in PV that $\mathsf{P} = \mathsf{NP}$ implies the provability in PV that PH collapses to P, it follows that $\varphi(x)$ is equivalent in PV to the language defined by some PV function symbol h. In other words, $\mathsf{PV} \vdash \mathsf{LB}_k^{a.e.}(h)$, which completes the proof of Theorem 2. $\square$