Learning algorithms versus automatability of Frege systems

Ján Pich  Rahul Santhanam
University of Oxford  University of Oxford

October 2021

Abstract

We connect learning algorithms and algorithms automating proof search in propositional proof systems: for every sufficiently strong, well-behaved propositional proof system $P$, we prove that the following statements are equivalent,

1. **Provable learning.** $P$ proves efficiently that p-size circuits are learnable by subexponential-size circuits over the uniform distribution with membership queries.

2. **Provable automatability.** $P$ proves efficiently that $P$ is automatable by non-uniform circuits on propositional formulas expressing p-size circuit lower bounds.

Here, $P$ is sufficiently strong and well-behaved if I.-III. holds: I. $P$ p-simulates Jeřábek’s system $WF$ (which strengthens the Extended Frege system $EF$ by a surjective weak pigeonhole principle); II. $P$ satisfies some basic properties of standard proof systems which p-simulate $WF$; III. $P$ proves efficiently for some Boolean function $h$ that $h$ is hard on average for circuits of subexponential size. For example, if III. holds for $P = WF$, then Items 1 and 2 are equivalent for $P = WF$.

If there is a function $h \in \text{NE} \cap \text{coNE}$ which is hard on average for circuits of size $2^{n/4}$, for each sufficiently big $n$, then there is an explicit propositional proof system $P$ satisfying properties I.-III., i.e. the equivalence of Items 1 and 2 holds for $P$.

1 Introduction

Learning algorithms and automatability algorithms searching for proofs in propositional proof systems are central concepts in complexity theory, but a priori they appear rather unrelated.
Learning algorithms. In the PAC model of learning introduced by Valiant [36], a circuit class $\mathcal{C}$ is learnable by a randomized algorithm $L$ over the uniform distribution, up to error $\epsilon$, with confidence $\delta$ and membership queries, if for every Boolean function $f$ computable by a circuit from $\mathcal{C}$, when given oracle access to $f$, $L$ outputs with probability $\geq \delta$ over the uniform distribution a circuit computing $f$ on $\geq (1 - \epsilon)$ inputs. An important task of learning theory is to find out if standard circuit classes such as $\mathbb{P}/\mathsf{poly}$ are learnable by efficient circuits. A way to approach the question is to connect the existence of efficient learning algorithms to other standard conjectures in complexity theory. For example, we can try to prove that efficient learning of $\mathbb{P}/\mathsf{poly}$ is equivalent to $\mathbb{P} = \mathbb{NP}$ or to the non-existence of strong pseudorandom generators. In both cases one implication is known: $\mathbb{P} = \mathbb{NP}$ implies efficient learning of $\mathbb{P}/\mathsf{poly}$ (with small error and high confidence) which in turn breaks pseudorandom generators. However, while some progress on the opposite implications has been made, they remain open, cf. [2, 35].

Automatability. The notion of automatability was introduced in the work of Bonet, Pitassi and Raz [5]. A propositional proof system $P$ is automatable if there is an algorithm $A$ such that for every tautology $\phi$, $A$ finds a $P$-proof of $\phi$ in $p$-time in the size of the shortest $P$-proof of $\phi$. That is, even if $P$ does not prove all tautologies efficiently, it can still be automatable. Establishing (non-)automatability results for concrete proof systems is one of the main tasks of proof complexity. This led to many attempts to link the notion of automatability to other standard complexity-theoretic conjectures. For example, recently Atserias and Müller [3] proved that automating Resolution is $\mathbb{NP}$-hard and their work has been extended to other weak proof systems, e.g. [11, 12, 13]. For stronger systems, it is known that automating Extended Frege system $\mathsf{EF}$, Frege or even constant-depth Frege would break specific cryptographic assumptions such as the security of RSA or Diffie-Hellman scheme, cf. [22, 5, 4]. It remains, however, open to obtain non-automatability of strong systems like Frege under a generic assumption such as the existence of strong pseudorandom generators, let alone to prove the equivalence between such notions.

In the present paper we derive a conditional equivalence between learning algorithms for $p$-size circuits and automatability of proof systems on tautologies encoding circuit lower bounds.

1.1 Our result

An ideal connection between learning and automatability would say that for standard proof systems $P$,

"$P$ is automatable if and only if $\mathbb{P}/\mathsf{poly}$ is learnable efficiently".

We establish this modulo some provability conditions and a change of parameters. Additionally, we need to consider automatability only w.r.t. formulas encoding circuit
lower bounds. More precisely, denote by \( \text{tt}(f, s) \) a propositional formula which expresses that boolean function \( f \) represented by its truth-table is not computable by a boolean circuit of size \( s \) represented by free variables, see Section 3. So \( \text{tt}(f, s) \) is a tautology if and only if \( f \) is hard for circuits of size \( s \). Similarly, let \( \text{tt}(f, s, t) \) be a formula expressing that circuits of size \( s \) fail to compute \( f \) on \( \geq t \)-fraction of inputs. In our main result (Theorem 1) we use a slightly modified notion of automatability where the automating algorithm for a proof system \( P \) is non-uniform and outputs a \( P \)-proof of a given formula \( \text{tt}(f, n^{O(1)}) \) in \( p \)-time in the size of the shortest \( P \)-proof of \( \text{tt}(f, 2^{n^{o(1)}}; 1/2 - 1/2^{n^{o(1)}}) \), see Section 3.

**Theorem 1** (Informal, cf. Theorem 10). Let \( P \) be a propositional proof system which \( \text{APC}_1 \)-provably \( p \)-simulates \( \text{WF} \) and satisfies some basic properties. Moreover, assume that \( P \) proves efficiently \( \text{tt}(h, 2^{n/4}; 1/2 - 1/2^{n/4}) \) for some boolean function \( h \). Then, the following statements are equivalent:

1. **Provable learning.** \( P \) proves efficiently that \( p \)-size circuits are learnable by \( 2^{n^{o(1)}} \)-size circuits, over the uniform distribution, up to error \( 1/2 - 1/2^{n^{o(1)}} \), with membership queries and confidence \( 1/2^{n^{o(1)}} \).

2. **Provable automatability.** \( P \) proves efficiently that \( P \) is automatable by non-uniform circuits on formulas \( \text{tt}(f, n^{O(1)}) \).

\( \text{WF} \) is an elegant strengthening of \( \text{EF} \) introduced by Jeřábek [14], which corresponds to the theory of approximate counting \( \text{APC}_1 \), a theory formalizing probabilistic \( p \)-time reasoning, see Section 2.2. Concrete proof systems which \( \text{APC}_1 \)-provably \( p \)-simulate \( \text{WF} \) and satisfy the basic properties from Theorem 1 include \( \text{WF} \) itself or even much stronger systems such as set theory \( \text{ZFC} \) (if we interpret \( \text{ZFC} \) as a suitable system for proving tautologies, see Section 5). The error and confidence of learning algorithms can be amplified ‘for free’, see Section 2.1, but we did not make the attempts to prove that the amplification is efficiently provable already in \( \text{WF} \).

**Plausibility of the assumption.** The main assumption in Theorem 1 is the provability of a circuit lower bound \( \text{tt}(h, 2^{n/4}; 1/2 - 1/2^{n/4}) \). This assumption has an interesting status. Razborov’s conjecture about hardness of Nisan-Wigderson generators implies a conditional hardness of formulas \( \text{tt}(h, n^{O(1)}) \) for Frege (for every \( h \), cf. [33], and it is possible to consider extensions of the conjecture to all standard proof systems, even set theory \( \text{ZFC} \). On the other hand, all major circuit lower bounds for weak circuit classes and explicit boolean functions are known to be efficiently provable in \( \text{EF} \), cf. [31, 24]. If we believe that explicit circuit lower bounds such as \( \text{tt}(h, 2^{n/4}; 1/2 - 1/2^{n/4}) \), for some \( h \in \text{EXP} \), are true, it is also perfectly plausible that they are efficiently provable in a

\[ ^{1}\text{This has not been verified for lower bounds obtained via the algorithmic method of Williams [37].} \]
standard proof system such as ZFC \(^2\) or EF. Notably, if EF proves efficiently \(\text{tt}(h, 2^{n/4})\) for some boolean function \(h\), then EF simulates WF, cf. [21, Lemma 19.5.4]. If there is a p-time algorithm which given a string of length \(2^n\) (specifying the size of \(\text{tt}(h, 2^{n/4})\)) generates an EF-proof of \(\text{tt}(h, 2^{n/4})\), then EF is p-equivalent to WF. To see that, combine Lemma 1 with the fact (proved in [14]) that APC\(_1\) proves the reflection principle for WF.

As a corollary of Theorem 1 we show that, under a standard hardness assumption, there is an explicit proof system \(P\) for which the equivalence holds. This, follows, essentially, by ‘hard-wiring’ tautologies \(\text{tt}(h, 2^{n/4}, 1/2 - 1/2^{n/4})\) to WF.

**Corollary 1** (cf. Corollary 3). Assume there is a NE\(∩\)coNE-function \(h_n : \{0, 1\}^n \mapsto \{0, 1\}\) such that for each sufficiently big \(n\), \(h_n\) is not \((1/2 + 1/2^{n/4})\)-approximable by \(2^{n/4}\)-size circuits.\(^3\) Then there is a proof system \(P\) (which can be described explicitly given the definition of \(h_n\)) such that Items 1 and 2 from Theorem 1 are equivalent.

The proof of Theorem 1 reveals also a conditional proof complexity collapse, which we discuss in Section 5.

**Corollary 2** (cf. Corollary 4). Let \(P, P_0\) be propositional proof systems which APC\(_1\)-provably p-simulate WF and satisfy some basic properties. Moreover, assume that systems \(P, P_0\) prove efficiently \(\text{tt}(h_n, 2^{n/4}, 1/2 - 1/2^{n/4})\) for some boolean function \(h_n\). Then, Item 1 implies Item 2:

1. **\(P\)-provable automatability.** \(P\) proves efficiently that \(P\) is automatable by non-uniform circuits on formulas \(\text{tt}(f, n^{O(1)})\).

2. **\(P_0\)-provable proof search.** There are \(p\)-size circuits \(B\) such that \(P_0\) proves efficiently that circuits \(B\) (given just \(\text{tt}(f, n^{O(1)})\)) generate \(P\)-proofs of \(\text{tt}(f, n^{O(1)})\) or \(2^{n^{o(1)}}\)-size circuits \((1/2 + 1/2^{n^{o(1)}})\)-approximating \(f\).

### 1.2 Outline of the proof

Our starting point for the derivation of Theorem 1 is a relation between natural proofs and automatability which goes back to a work of Razborov and Krajíček. Razborov [32, 30] proved that certain theories of bounded arithmetic cannot prove explicit circuit lower bounds assuming strong pseudorandom generators exist. Krajíček [18, 20] developed the concept of feasible interpolation (a weaker version of automatability, cf. [21]) and reformulated Razborov’s unprovability result in this language, see [21, Section 17.9] for more historical remarks.

\(^2\)Efficient provability of \(\text{tt}(h, 2^{n/4}, 1/2 - 1/2^{n/4})\) in ZFC, for some \(h \in \text{EXP}\), would follow from the standard provability of this lower bound in ZFC.

\(^3\)A circuit \(C\) with \(n\) inputs \(\gamma\)-approximates function \(f : \{0, 1\}^n \mapsto \{0, 1\}\) if \(\Pr_{x \in \{0, 1\}^n}[C(x) = f(x)] \geq \gamma\).
Theorem 2 (Razborov-Krajíček [32, 30, 18] - informal version). Let $P$ be a proof system which simulates $EF$. If $P$ is automatable and $P$ proves efficiently $tt(h, n^{O(1)})$ for some function $h$, then there are $P/poly$-natural proofs useful against $P/poly$.

The second crucial ingredient we will use is a result of Carmosino, Impagliazzo, Kabanets and Kolokolova, who showed that natural proofs can be turned into learning algorithms [7]. This allows us to conclude the following.

Theorem 3 (Informal, cf. Theorem 6). Let $P$ be a proof system simulating $EF$. If $P$ proves efficiently $tt(h, n^{O(1)})$ for some function $h$, then automatability of $P$ implies the existence of subexponential-size circuits learning $p$-size circuits over the uniform distribution, with membership queries.

Theorem 3 directly implies that if strong pseudorandom generators exist and $EF$ proves efficiently $tt(h, n^{O(1)})$ for some $h$, then $EF$ is automatable if and only if there are subexponential-size circuits learning $p$-size circuits over the uniform distribution, with membership queries. The disadvantage of this observation is that, unlike in Theorem 1, its assumptions are known to imply that both sides of the desired equivalence are false.

We note that the proof of Theorem 3 can be used to show also that optimal and automatable proof systems imply learning algorithms. Here, a propositional proof system $P$ is optimal, if for each propositional proof system $R$, an $R$-proof $\pi$ of $\phi$, implies the existence of a $poly(|\pi|)$-size $P$-proof of $\phi$.

Theorem 4 (Optimality and automatability implies learning, cf. Theorem 7). If there is an optimal proof system which is automatable, then there are subexponential-size circuits infinitely often learning $p$-size circuits over the uniform distribution.

In fact, it is possible to prove, unconditionally, that there is some propositional proof system $P$ such that automatability of $P$ is equivalent to the existence of subexponential-size circuits infinitely often learning $P/poly$ over the uniform distribution, cf. Theorem 8. The proof is, however, non-constructive so (unlike in Corollary 1) we do not know which system $P$ satisfies the equivalence.

The entrance of metamathematics. Unfortunately, it is unclear how to derive the opposite implication in Theorem 3. We do not know how to automate, say, $EF$ assuming just the existence of efficient learning algorithms. In order to get the reverse, we need to assume that an efficient learning algorithm is provably correct in a proof system $P$, which $p$-simulates $WF$. For simplicity, let $P = WF$. If we assume that $WF$ proves efficiently for some small circuits that they can learn $p$-size circuits, we can show that there are small circuits such that $WF$ proves efficiently that these circuits automate $WF$ on formulas $tt(f, n^{O(1)})$. In more detail, we first formalize in $APC_1$ the implication that $WF$-provable learning yields automatability of $WF$ on $tt(f, n^{O(1)})$ - if a learning circuit $A$ does not find a small circuit for a given function $f$, the automating circuit uses $WF$-proof of the correctness
of $A$ to produce a short $WF$-proof of $\text{tt}(f, n^{O(1)})$. Then, we translate the $\text{APC}_1$-proof to $WF$ and conclude that $WF$ proves that $WF$-provable learning implies automatability of $WF$. This allows us to show that if we have $WF$-provable learning, then $WF$ is $WF$-provably automatable on $\text{tt}(f, n^{O(1)})$.

It is important that assuming $WF$-provable learning, we are able to derive $WF$-provable automatability of $WF$, and not just automatability of $WF$. This makes it possible to obtain the opposite direction and establish the desired equivalence: If we know that $WF$ proves that $WF$ is automatable, we can formalize the proof of Theorem 3 in $WF$ and conclude the existence of $WF$-provable learning algorithms.

One could expect that $WF$-provable learning would yield just automatability of $WF$ and $WF$-provability of $WF$-provable learning would be needed to get $WF$-provably automatability of $WF$. It might be thus surprising that already the first metamathematical level achieves the desired ‘fixpoint’.

**Benefits of bounded arithmetic.** The proof of Theorem 1 relies heavily on formalizations. Among other things we need to formalize the result of Carmosino, Impagliazzo, Kabanets and Kolokolova in $\text{APC}_4^4$, and use an elaborated way of expressing complex statements about metacomplexity by propositional formulas: existential quantifiers often need to be witnessed before translating them to propositional setting. The framework of bounded arithmetic allows us to deal with these complications in an elegant way: we often reason in bounded arithmetic, possibly using statements of higher quantifier complexity, and only subsequently translate the outcomes to propositional logic, if the resulting (proved) statement has $\text{coNP}$ form. Notably, already propositional formulas expressing probabilities in the definition of learning algorithms require more advanced tools - the probabilities are encoded using suitable Nisan-Wigderson generators which come out of the notion of approximate counting in $\text{APC}_1$, cf. Section 3.2.

### 1.3 Related results

Learning algorithms and automatability have been linked already in the work of Alekhnovich, Braverman, Feldman, Klivans and Pitassi [1], who showed an informal connection between learning of weak circuit classes and automatability of some weak systems such as tree-like Resolution. As already mentioned, Atserias and Müller [3] proved that automating Resolution is $\text{NP}$-hard and their work has been extended to other weak proof systems, see e.g. [11, 12, 13]. A direct consequence of these results is that efficient algorithms automating the respective proof systems can be used to learn efficiently classes like $\mathbb{P}/\text{poly}$. A major difference between these results and ours is that for our results to apply, the proof system

---

\(^4\)We will actually formalize ‘CIKK’ just conditionally, in order to avoid the formalization of Bertrand’s postulate.
needs to be sufficiently strong, while for the other results, the proof system needs to be weak (in the sense that lower bounds for the system are already known).

1.4 Open problems

**Unconditional equivalence between learning and automatability.** Is it possible to avoid the assumption on the provability of a circuit lower bound in Theorem 1 and establish an unconditional equivalence between learning and automatability?

**Complexity theory from the perspective of metamathematics.** Our results demonstrate that in the context of metamathematics it is possible to establish some complexity-theoretic connections which we are not able to establish otherwise. We exploit the metamathematical nature of the notion of automatability: efficient $P$-provability of the correctness of an algorithm implies efficient $P$-provability of automatability of $P$. Is it possible to take advantage of metamathematics in other contexts and resolve other important open problems in this setting? For example, could we get a version of the desired equivalence between the existence of efficient learning algorithms and the non-existence of cryptographic pseudorandom generators, cf. [26, 35, 28]? The question of basing cryptography on a worst-case assumption such as $P \neq NP$ could be addressed in this setting by showing that if a sufficiently strong proof system $P$ proves efficiently that there is no strong pseudorandom generator, then $P$ is p-bounded.

**Circuit lower bound tautologies.** How essential are circuit lower bound tautologies in our results? Consider fundamental questions of proof complexity (p-boundness, optimality, automatability) w.r.t. formulas $\text{tt}(f, s)$. Do they coincide with the original ones? Are formulas $\text{tt}(f, s)$ the hardest ones, do they admit optimal proof systems, or can we turn automatability on formulas $\text{tt}(f, s)$ into automatability on all formulas?

**Proof complexity magnification.** Is it possible to obtain the collapse from Corollary 2 for formulas expressing standard conjectures? For example, is it possible to show that the hardness of $\text{tt}(\text{SAT}, n^{O(1)})$ for some proof system $P_0$, implies hardness of $\text{tt}(\text{SAT}, n^{O(1)})$ for stronger proof systems? An instance of such a magnification phenomenon appeared in [24] (with $P_0 = \text{constant-depth Frege}$, $P = \text{Frege}$ and a different formalization of $\text{tt}(\text{SAT}, n^{O(1)})$ in $P$).

---

5The formalization of this statement would assume the existence of a p-size circuit which for any p-size circuit defining a potential pseudorandom generator outputs its distinguisher.
2 Preliminaries

2.1 Natural proofs and learning algorithms

[n] denotes \{1, \ldots, n\}. Circuit[s] denotes fan-in two Boolean circuits of size at most s. The size of a circuit is the number of gates.

**Definition 1** (Natural property [34]). Let \( m = 2^n \) and \( s, d : \mathbb{N} \rightarrow \mathbb{N} \). A sequence of circuits \( \{C_m\}_{m=1}^{\infty} \) is a Circuit[s(m)]-natural property useful against Circuit[d(n)] if

1. Constructivity. \( C_m \) has \( m \) inputs and size \( s(m) \),
2. Largeness. \( \Pr_x[C_m(x) = 1] \geq 1/m^{O(1)} \),
3. Usefulness. For each sufficiently big \( m \), \( C_m(x) = 1 \) implies that \( x \) is a truth-table of a function on \( n \) inputs which is not computable by circuits of size \( d(n) \).

**Definition 2** (PAC learning). A circuit class \( \mathcal{C} \) is learnable over the uniform distribution by a circuit class \( \mathcal{D} \) up to error \( \epsilon \) with confidence \( \delta \), if there are randomized oracle circuits \( L^f \) from \( \mathcal{D} \) such that for every Boolean function \( f : \{0,1\}^n \rightarrow \{0,1\} \) computable by a circuit from \( \mathcal{C} \), when given oracle access to \( f \), input \( 1^n \) and the internal randomness \( w \in \{0,1\}^* \), \( L^f \) outputs the description of a circuit satisfying

\[
\Pr_w[L^f(1^n, w) \ (1 - \epsilon)-approximates 
\]

\( f \] \( \geq \delta \).

\( L^f \) uses non-adaptive membership queries if the set of queries which \( L^f \) makes to the oracle does not depend on the answers to previous queries. \( L^f \) uses random examples if the set of queries which \( L^f \) makes to the oracle is chosen uniformly at random.

In this paper, PAC learning always refers to learning over the uniform distribution. While, a priori, learning over the uniform distribution might not reflect real-world scenarios very well (and on the opposite end, learning over all distributions is perhaps overly restrictive), as far as we can tell it is possible that PAC learning of p-size circuits over the uniform distribution implies PAC learning of p-size circuits over all p-samplable distributions.

**Boosting confidence and reducing error.** The confidence of the learner can be efficiently boosted in a standard way. Suppose an \( s \)-size circuit \( L^f \) learns \( f \) up to error \( \epsilon \) with confidence \( \delta \). We can then run \( L^f \) \( k \) times, test the output of \( L^f \) from every run with \( m \) new random queries and output the most accurate one. By Hoeffding’s inequality, \( m \) random queries fail to estimate the error \( \epsilon \) of an output of \( L^f \) up to \( \gamma \) with probability at most \( 2/e^{2\gamma^2 m} \). Therefore the resulting circuit of size \( \text{poly}(s, m, k) \) learns \( f \) up to error \( \epsilon + \gamma \) with confidence at least \( 1 - 2k/e^{2\gamma^2 m} - (1 - \delta)^k \geq 1 - 2k/e^{2\gamma^2 m} - e^{-k\delta} \). If we are trying to
learn small circuits we can get even confidence 1 by fixing internal randomness of learner nonuniformly without losing much on the running time or the error of the output. It is also possible to reduce the error up to which \( L^f \) learns \( f \) without a significant blowup in the running time and confidence. If we want to learn \( f \) with a better error, we first learn an amplified version of \( f \), \( \text{Amp}(f) \). Employing direct product theorems and Goldreich-Levin reconstruction algorithm, Carmosino et al. [7, Lemma 3.5] showed that for each \( 0 < \epsilon, \gamma < 1 \) it is possible to map a Boolean function \( f \) with \( n \) inputs to a Boolean function \( \text{Amp}(f) \) with \( \text{poly}(n, 1/\epsilon, \log(1/\gamma)) \) inputs so that \( \text{Amp}(f) \in \mathbb{P}/\text{poly}^f \) and there is a probabilistic \( \text{poly}(|C|, n, 1/\epsilon, 1/\gamma) \)-time machine which given a circuit \( C \) \( (1/2 + \gamma) \)-approximating \( \text{Amp}(f) \) and an oracle access to \( f \) outputs with high probability a circuit \( (1 - \epsilon) \)-approximating \( f \). We can thus often ignore the optimisation of the confidence and error parameter. Note, however, that the error reduction of Carmosino et al. requires membership queries.

**Natural proofs vs learning algorithms.** Natural proofs are actually equivalent to efficient learning algorithms with suitable parameters. In this paper we need just one implication.

**Theorem 5** (Carmosino-Impagliazzo-Kabanets-Kolokolova [7]). Let \( R \) be a \( \mathbb{P}/\text{poly} \)-natural property useful against \( \text{Circuit}[n^k] \) for \( k \geq 1 \). Then, for each \( \gamma \in (0, 1) \), \( \text{Circuit}[n^{k\gamma/a}] \) is learnable by \( \text{Circuit}[2^{O(n^\gamma)}] \) over the uniform distribution with non-adaptive membership queries, confidence 1, up to error \( 1/n^{k\gamma/a} \), where \( a \) is an absolute constant.

### 2.2 Bounded arithmetic and propositional logic

Theories of bounded arithmetic capture various levels of feasible reasoning and present a uniform counterpart to propositional proof systems.

The first theory of bounded arithmetic formalizing p-time reasoning was introduced by Cook [9] as an equational theory \( \text{PV} \). We work with its first-order conservative extension \( \text{PV}_1 \) from [23]. The language of \( \text{PV}_1 \), denoted \( \text{PV} \) as well, consists of symbols for all p-time algorithms given by Cobham’s characterization of p-time functions, cf. [8]. A \( \text{PV} \)-formula is a first-order formula in the language \( \text{PV} \). \( \Sigma^b_0 \) \((=\Pi^b_0)\) denotes \( \text{PV} \)-formulas with only sharply bounded quantifiers \( \exists x, x \leq |t|, \forall x, x \leq |t| \), where \( |t| \) is “the length of the binary representation of \( t \)”. Inductively, \( \Sigma^b_{i+1} \) resp. \( \Pi^b_{i+1} \) is the closure of \( \Pi^b_i \) resp. \( \Sigma^b_i \) under positive Boolean combinations, sharply bounded quantifiers, and bounded quantifiers \( \exists x, x \leq t \) resp. \( \forall x, x \leq t \). Predicates definable by \( \Sigma^b_i \) resp. \( \Pi^b_i \) formulas are in the \( \Sigma^p_i \) resp. \( \Pi^p_i \) level of the polynomial hierarchy, and vice versa. \( \text{PV}_1 \) is known to prove \( \Sigma^b_0(\text{PV}) \)-induction:

\[
A(0) \land \forall x \ A(x) \rightarrow A(x + 1) \rightarrow \forall x A(x),
\]

for \( \Sigma^b_0 \)-formulas \( A \), cf. Krajíček [17].
Buss [6] introduced the theory $S_2^1$ extending $\text{PV}_1$ with the $\Sigma^b_1$-length induction:

$$A(0) \land \forall x < |a|, (A(x) \rightarrow A(x+1)) \rightarrow A(|a|),$$

for $A \in \Sigma^b_1$. $S_2^1$ proves the sharply bounded collection scheme $BB(\Sigma^b_1)$:

$$\forall i < |a| \exists x < a, A(i, x) \rightarrow \exists w \forall i < |a|, A(i, |w|_i),$$

for $A \in \Sigma^b_1$ ($|w|_i$ is the $i$th element of the sequence coded by $w$), which is unprovable in $\text{PV}_1$ under a cryptographic assumption, cf. [10]. On the other hand, $S_2^1$ is $\forall \Sigma^b_1$-conservative over $\text{PV}_1$. This is a consequence of Buss’s witnessing theorem stating that $S_2^1 \vdash \exists y, A(x, y)$ for $A \in \Sigma^b_1$ implies $\text{PV}_1 \vdash A(x, f(x))$ for some $\text{PV}$-function $f$.

Following a work by Krajíček [19], Jeřábek [14, 15, 16] systematically developed a theory $\text{APC}_1$ capturing probabilistic p-time reasoning by means of approximate counting. The theory $\text{APC}_1$ is defined as $\text{PV}_1 + d\text{WPHP}(\text{PV})$ where $d\text{WPHP}(\text{PV})$ stands for the dual (surjective) pigeonhole principle for $\text{PV}$-functions, i.e. for the set of all formulas

$$x > 0 \rightarrow \exists v < x(|y| + 1) \forall u < x|y|, f(u) \neq v,$$

where $f$ is a $\text{PV}$-function which might involve other parameters not explicitly shown. We devote Section 2.3 to a more detailed description of the machinery of approximate counting in $\text{APC}_1$.

Any $\Pi^b_1$-formula $\phi$ provable in $\text{PV}_1$ can be expressed as a sequence of tautologies $||\phi||_n$ with proofs in the Extended Frege system $\text{EF}$ which are constructible in p-time (given a string of the length $n$), cf. [9]. Similarly, $\Pi^b_1$-formulas provable in $\text{APC}_1$ translate to tautologies with p-time constructible proofs in $\text{WF}$, an extension of $\text{EF}$ introduced by Jeřábek [14]. We describe the translation and system $\text{WF}$ in more detail below.

As it is often easier to present a proof in a theory of bounded arithmetic than in the corresponding propositional system, bounded arithmetic functions, so to speak, as a uniform language for propositional logic.

We refer to Krajíček [21] for basic notions in proof complexity.

**Definition 3 (WF (WPHP Frege), cf. Jeřábek [14]).** Let $L$ be a finite and complete language for propositional logic, i.e. $L$ consists of finitely many boolean connectives of constant arity such that each boolean function of every arity can be expressed by an $L$-formula, and let $\mathcal{R}$ be a finite, sound and implicationally complete set of Frege rules (in the language $L$). A WF-proof of a ($L$-)circuit $A$ is a sequence of circuits $A_0, \ldots, A_k$ such

---

5Krajíček [19] introduced a theory $BT$ defined as $S_2^1 + d\text{WPHP}(\text{PV})$ and proposed it as a theory for probabilistic p-time reasoning.
that \( A_k = A \), and each \( A_i \) is derived from some \( A_{j_1}, \ldots, A_{j_\ell} \), \( j_1, \ldots, j_\ell < i \) by a Frege rule from \( \mathcal{R} \), or it is similar to some \( A_j \), \( j < i \), or it is the dWP PH axiom,

\[
\bigvee_{\ell=1}^{m} \left( r_\ell \neq C_{i,\ell}(D_{i,1}, \ldots, D_{i,n}) \right),
\]

where \( n < m \) and \( r_\ell \) are pairwise distinct variables which do not occur in circuits \( A, C_{i,p} \), or \( A_j \) for \( j < i \), but may occur in circuits \( D_{i,1}, \ldots, D_{i,n} \).

The similarity rule in Definition 3 is verified by a specific p-time algorithm which checks that circuits \( A_i \) and \( A_j \) can be ‘unfolded’ to the same (possible huge) formula, cf. [14, Lemma 2.2.]. Intuitively, the NLOG (⊆ P) algorithm recognizes if two circuits are not similar by guessing a partial path through them, going from the output to the inputs, where on at least one instruction the circuits disagree. As defined WF depends on the choice of Frege rules and language \( L \), but for each choice the resulting systems are p-equivalent, so we can identify them. The dWP PH axiom refers to ‘dual weak pigeonhole principle’ postulating the existence of an element \( r_1, \ldots, r_m \) outside the range of a p-size map \( C_{i,1}, \ldots, C_{i,n} : \{0,1\}^n \mapsto \{0,1\}^m \). The dWP PH axiom comes with a specification of circuits \( C_{i,1}, \ldots, C_{i,m}, D_{i,1}, \ldots, D_{i,n} \) so that we can recognize the axiom efficiently.

The translation of a \( \Pi_1^b \) formula \( \phi \) into a sequence of propositional formulas \( |\phi|_\pi \) works as follows. For each PV-function \( f(x_1, \ldots, x_k) \) and numbers \( n_1, \ldots, n_k \) we have a p-size circuit \( C_f \) computing the restriction \( f : 2^{n_1} \times \cdots \times 2^{n_k} \mapsto 2^{t(n_1, \ldots, n_k)} \), where \( b \) is a suitable ‘bounding’ polynomial for \( f \). The formula \( |f|_\pi(p, q, r) \) expresses that \( C_f \) outputs \( r \) on input \( p \), with \( q \) being the auxiliary variables corresponding to the nodes of \( C_f \). The formula \( |\phi(x)|_\pi(p, q) \) is defined as \( |\phi'(x)|_\pi(p, q) \), where \( \phi' \) is the negation normal form of \( \phi \), i.e. negations in \( \phi' \) are only in front of atomic formulas. The formula \( |\phi'(x)|_\pi(p, q) \) is defined inductively in a straightforward way so that \( |\ldots| \) commutes with \( \lor, \land \). The atoms \( p \) correspond to variables \( x \), atoms \( q \) correspond to the universally quantified variables of \( \phi \) and to the outputs and auxiliary variables of circuits \( C_f \) for functions \( f \) appearing in \( \phi \). Sharply bounded quantifiers are replaced by polynomially big conjunctions resp. disjunctions. For the atomic formulas we have,

\[
|f(x)|_\pi := |f(x)|_\pi(p, q, r) \land |g(x)|_\pi(p, q', r') \to \bigwedge_i r_i = r'_i,
\]

\[
|\neg f(x)|_\pi := |f(x)|_\pi(p, q, r) \land |g(x)|_\pi(p, q', r') \to \neg \bigwedge_i r_i = r'_i,
\]

\[
|f(x) \leq g(x)|_\pi := |f(x)|_\pi(p, q, r) \land |g(x)|_\pi(p, q', r') \to \bigwedge_i (r_i \land \bigwedge_{j > i} (r_j = r'_j) \to r'_i),
\]

\[
|\neg f(x) \leq g(x)|_\pi := |f(x)|_\pi(p, q, r) \land |g(x)|_\pi(p, q', r') \to \neg \bigwedge_i (r_i \land \bigwedge_{j > i} (r_j = r'_j) \to r'_i).
\]
2.3 Approximate counting

In order to prove our results we will need to use Jeřábek’s theory of approximate counting. This section recalls the properties of APC₁ we will need.

By a definable set we mean a collection of numbers satisfying some formula, possibly with parameters. When a number \( a \) is used in a context which asks for a set it is assumed to represent the integer interval \([0, a)\), e.g. \( X \subseteq a \) means that all elements of set \( X \) are less than \( a \). If \( X \subseteq a \), \( Y \subseteq b \), then \( X \times Y := \{bx + y \mid x \in X, y \in Y\} \subseteq ab \) and \( X \cup Y := X \cup \{y + a \mid y \in Y\} \subseteq a + b \). Rational numbers are assumed to be represented by pairs of integers in the natural way. We use the notation \( x \in \text{Log} \iff \exists y, x = \|y\| \) and \( x \in \text{LogLog} \iff \exists y, x = \|\|y\|\| \).

Let \( C : 2^n \to 2^m \) be a circuit and \( X \subseteq 2^n, Y \subseteq 2^m \) definable sets. We write \( C : X \to Y \) if \( \forall y \in Y \exists x \in X, C(x) = y \). Jeřábek [16] gives the following definitions in APC₁ (but they can be considered in weaker theories as well).

**Definition 4.** Let \( X, Y \subseteq 2^n \) be definable sets, and \( \epsilon \leq 1 \). The size of \( X \) is approximately less than the size of \( Y \) with error \( \epsilon \), written as \( X \preceq_\epsilon Y \), if there exists a circuit \( C \), and \( v \neq 0 \) such that

\[
C : v \times (Y \cup \epsilon 2^n) \to v \times X.
\]

\( X \approx_\epsilon Y \) stands for \( X \preceq_\epsilon Y \) and \( Y \preceq_\epsilon X \).

Since a number \( s \) is identified with the interval \([0, s)\), \( X \preceq_\epsilon s \) means that the size of \( X \) is at most \( s \) with error \( \epsilon \).

The definition of \( X \preceq_\epsilon Y \) is an unbounded \( \exists \Pi^2_1 \)-formula even if \( X, Y \) are defined by circuits so it cannot be used freely in bounded induction. Jeřábek [16] solved this problem by working in HARD⁴, a conservative extension of APC₁, defined as a relativized theory \( \text{PV}_1(\alpha) + \text{dWPHP}(\text{PV}(\alpha)) \) extended with axioms postulating that \( \alpha(x) \) is a truth-table of a function on \( \|x\| \) variables hard on average for circuits of size \( 2\|x\|/4 \), see Section 3.2. In HARD⁴ there is a \( \text{PV}_1(\alpha) \) function \( \text{Size} \) approximating the size of any set \( X \subseteq 2^n \) defined by a circuit \( C \) so that \( X \approx_\epsilon \text{Size}(C, 2^n, 2^{\epsilon^{-1}}) \) for \( \epsilon^{-1} \in \text{Log} \), cf. [16, Lemma 2.14]. If \( X \cap t \subseteq 2^{|t|} \) is defined by a circuit \( C \) and \( \epsilon^{-1} \in \text{Log} \), we can define

\[
\Pr_{x < t}[x \in X]_\epsilon := \frac{1}{t} \text{Size}(C, 2^{|t|}, 2^{\epsilon^{-1}}).
\]

The presented definitions of approximate counting are well-behaved:

**Proposition 1** (Jeřábek [16]). \((\text{in PV}_1)\) Let \( X, X', Y, Y', Z \subseteq 2^n \) and \( W, W' \subseteq 2^m \) be definable sets, and \( \epsilon, \delta < 1 \). Then

i) \( X \subseteq Y \Rightarrow X \preceq_0 Y \),

ii) \( X \preceq_\epsilon Y \land Y \preceq_\delta Z \Rightarrow X \preceq_{\epsilon + \delta} Z \),

iii) \( X \preceq_\epsilon X' \land W \preceq_\delta W' \Rightarrow X \times W \preceq_{\epsilon + \delta + \epsilon \delta} X' \times W' \).

iv) \( X \preceq_\epsilon X' \land Y \preceq_\delta Y' \) and \( X', Y' \) are separable by a circuit, then \( X \cup Y \preceq_{\epsilon + \delta} X' \cup Y' \).
Proposition 2 (Jeřábek [16]). (in APC)

1. Let $X, Y \subseteq 2^n$ be definable by circuits, $s, t, u \leq 2^n$, $\epsilon, \delta, \theta, \gamma < 1, \gamma^{-1} \in \text{Log}$. Then
   
   i) $X \preceq_{\gamma} Y$ or $Y \preceq_{\gamma} X$,
   
   ii) $s \preceq_{\epsilon} X \preceq_{\delta} t \Rightarrow s < t + (\epsilon + \delta + \gamma)2^n$,
   
   iii) $X \preceq_{\epsilon} Y \Rightarrow 2^n \setminus Y \preceq_{\epsilon + \gamma} 2^n \setminus X$,
   
   iv) $X \approx_{\epsilon} s \land Y \approx_{\delta} t \land X \cap Y \approx_{\theta} u \Rightarrow X \cup Y \approx_{\epsilon + \delta + \theta + \gamma} s + t - u$.

2. (Disjoint union) Let $X_i \subseteq 2^n$, $i < m$ be defined by a sequence of circuits and $\epsilon, \delta \leq 1$, $\delta^{-1} \in \text{Log}$. If $X_i \preceq_{\epsilon} s_i$ for every $i < m$, then $\bigcup_{i<m}(X_i \times \{i\}) \preceq_{\epsilon + \delta} \sum_{i<m} s_i$.

3. (Averaging) Let $X \subseteq 2^n \times 2^m$ and $Y \subseteq 2^m$ be definable by circuits, $Y \preceq_{\epsilon} t$ and $X_y \preceq_{\delta} s$ for every $y \in Y$, where $X_y := \{x \mid \langle x, y \rangle \in X\}$. Then for any $\gamma^{-1} \in \text{Log}$,

   $X \cap (2^n \times Y) \preceq_{\epsilon + \delta + \epsilon \delta + \gamma} s t$.

When proving $\Sigma_1^b$ statements in APC we can afford to work in $S_2^1 + dWPHP(PV) + BB(\Sigma_2^b)$ and, in fact, assuming the existence of a single hard function in PV gives us the full power of APC. Here, $BB(\Sigma_2^b)$ is defined as $BB(\Sigma_1^b)$ but with $A \in \Sigma_2^b$.

Lemma 1 ([24]). Suppose $S_2^1 + dWPHP(PV) + BB(\Sigma_2^b) \vdash \exists y A(x, y)$ for $A \in \Sigma_1^b$. Then, for every $\epsilon < 1$, there is $k$ and PV-functions $g, h$ such that PV proves

\[ |f| \geq |x|^k \land \exists y, |y| = ||f||, C_h (y) \neq f(y) \rightarrow A(x, g(x, f)) \]

where $f(y)$ is the $y$th bit of $f$, $f(y) = 0$ for $y > |f|$, and $C_h$ is a circuit of size $\leq 2^{|x||f||}$ generated by $h$ on $f, x$. Moreover, APC $\vdash \exists y A(x, y)$.

3 Formalizing complexity-theoretic statements

3.1 Circuit lower bounds

An ‘almost everywhere’ formulation of a circuit lower bound for circuits of size $s$ and a function $f$ says that for every sufficiently big $n$, for each circuit $C$ with $n$ inputs and size $s$, there exists an input $y$ on which the circuit $C$ fails to compute $f(y)$.

If $f : \{0, 1\}^n \rightarrow \{0, 1\}$ is an NP function and $s = n^k$ for a constant $k$, this can be written down as a $\forall \Sigma_2^b$ formula $\text{LB}(f, n^k)$,

\[ \forall n, n > n_0 \forall \text{ circuit } C \text{ of size } \leq n^k \exists y, |y| = n, C(y) \neq f(y), \]

where $n_0$ is a constant and $C(y) \neq f(y)$ is a $\Sigma_2^b$ formula stating that a circuit $C$ on input $y$ outputs the opposite value of $f(y)$.

If we want to express $s(n)$-size lower bounds for $s(n)$ as big as $2^O(n)$, we add an extra assumption on $n$ stating that $\exists x, n = ||x||$. That is, the resulting formula $\text{LB}_{tt}(f, s(n))$
has form ‘∀x, n; n = ||x|| → .’. Treating x, n as free variables, LB_{tt}(f, s(n)) is Π^b_1 if f is, for instance, SAT because n = ||x|| implies that the quantifiers bounded by 2^{O(n)} are sharply bounded. Moreover, allowing f ∈ NE lifts the complexity of LB_{tt}(f, s(n)) just to ∀Σ^b_1. The function s(n) in LB_{tt}(f, s(n)) is assumed to be a PV-function with input x (satisfying ||x|| = n).

In terms of the Log-notations, LB(f, n^k) implicitly assumes n ∈ Log while LB_{tt}(f, n^k) assumes n ∈ LogLog. By chosing the scale of n we are determining how big objects are going to be ‘feasible’ for theories reasoning about the statement. In the case n ∈ LogLog, the truth-table of f (and everything polynomial in it) is feasible. Assuming just n ∈ Log means that only the objects of polynomial-size in the size of the circuit are feasible. Likewise, the theory reasoning about the circuit lower bound is less powerful when working with LB(f, n^k) than with LB_{tt}(f, n^k). (The scaling in LB_{tt}(f, s) corresponds to the choice of parameters in natural proofs and in the formalizations by Razborov [31].)

We can analogously define formulas LB_{tt}(f, s(n), t(n)) expressing an average-case lower bound for f, where f is a free variable (with f(y) being the yth bit of f and f(y) = 0 for y > |f|). More precisely, LB_{tt}(f, s(n), t(n)) generalizes LB_{tt}(f, s(n)) by saying that each circuit of size s(n) fails to compute f on at least t(n) inputs, for PV-functions s(n), t(n). Since n ∈ LogLog, LB_{tt}(f, s(n), t(n)) is Π^b_1.

**Propositional version.** An s(n)-size circuit lower bound for a function f : \{0, 1\}^n → \{0, 1\} can be expressed by a 2^{O(n)}-size propositional formula tt(f, s),

\[
\bigvee_{y \in \{0, 1\}^n} f(y) \neq C(y)
\]

where the formula f(y) \neq C(y) says that an s(n)-size circuit C represented by poly(s) variables does not output f(y) on input y. The values f(y) are fixed bits. That is, the whole truth-table of f is hard-wired in tt(f, s).

The details of the encoding of the formula tt(f, s) are not important for us as long as the encoding is natural because systems like EF considered in this paper can reason efficiently about them. We will assume that tt(f, s) is the formula resulting from the translation of Π^b_1 formula LB_{tt}(h, s), where n_0 = 0, n, x are substituted after the translation by fixed constants so that x = 2^{2n}, and h is a free variable (with h(y) being the yth bit of h and h(y) = 0 for y > |h|) which is substituted after the translation by constants defining f.

Analogously, we can express average-case lower bounds by propositional formulas tt(f, s(n), t(n)) obtained by translating LB_{tt}(h, s(n), t(n)2^n), with n_0 = 0, fixed x = 2^{2n} and h substituted after the translation by f.

### 3.2 Learning algorithms

A circuit class C is defined by a PV-formula if there is a PV-formula defining the predicate C ∈ C. Definition 2 can be formulated in the language of HARD^A: A circuit class C
(defined by a PV-formula) is learnable over the uniform distribution by a circuit class $\mathcal{D}$ (defined by a PV-formula) up to error $\epsilon$ with confidence $\delta$, if there are randomized oracle circuits $L^f$ from $\mathcal{D}$ such that for every Boolean function $f : \{0,1\}^n \mapsto \{0,1\}$ (represented by its truth-table) computable by a circuit from $\mathcal{C}$, for each $\gamma^{-1} \in \text{Log}$, when given oracle access to $f$, input $1^n$ and the internal randomness $w \in \{0,1\}^*$, $L^f$ outputs the description of a circuit satisfying

$$
\Pr_w[L^f(1^n, w) \ (1 - \epsilon)\text{-approximates } f]_\gamma \geq \delta.
$$

The inner probability of approximability of $f$ by $L^f(1^n, w)$ is counted exactly. This is possible because $f$ is represented by its truth-table, which implies that $2^n \in \text{Log}$.\footnote{It could be interesting to develop systematically a standard theory of learning algorithms in \textit{APC} and \textit{WF}, but it is not our goal here. Note, for example, that when we are learning small circuits it is not clear how to boost the confidence to 1 in \textit{APC}, because we don’t have counting with exponential precision.}

**Propositional version.** In order, to translate the definition of learning algorithms to propositional formulas we need to look more closely at the definition of $\text{HARD}^A$.

$\textit{PV}_1$ can be relativized to $\textit{PV}_1(\alpha)$. The new function symbol $\alpha$ is then allowed in the inductive clauses for introduction of new function symbols. This means that the language of $\textit{PV}_1(\alpha)$, denoted also $\textit{PV}(\alpha)$, contains symbols for all p-time oracle algorithms.

**Proposition 3** (Jeřábek [14]). For every constant $\epsilon < 1/3$ there exists a constant $n_0$ such that $\textit{APC}_1$ proves: for every $n \in \text{LogLog}$ such that $n > n_0$, there exist a function $f : 2^n \rightarrow 2$ such that no circuit of size $2^n$ computes $f$ on $\geq (1/2 + 1/2^n)2^n$ inputs.

**Definition 5** (Jeřábek [14]). The theory $\text{HARD}^A$ is an extension of the theory $\textit{PV}_1(\alpha) + \text{dWPHP}(\textit{PV}(\alpha))$ by the axioms

1. $\alpha(x)$ is a truth-table of a Boolean function in $||x||$ variables,
2. $\text{LB}_{tt}(\alpha(x), 2||x||/4, 2||x||/4(1/2 - 1/2||x||/4))$, for constant $n_0$ from Proposition 3,
3. $||x|| = ||y|| \rightarrow \alpha(x) = \alpha(y)$.

By inspecting the proof of Lemma 2.14 in [16], we can observe that on each input $C, 2^n, 2^{-1}$ the $\textit{PV}_1(\alpha)$-function \textit{Size} calls $\alpha$ just once (to get the truth-table of a hard function which is then used as the base function of the Nisan-Widgerson generator). In fact, \textit{Size} calls $\alpha$ on input $x$ which depends only on $|C|$, the number of inputs of $C$ and w.l.o.g. also just on $|\epsilon^{-1}|$ (since decreasing $\epsilon$ leads only to a better approximation). In combination with the fact that the approximation $\text{Size}(C, 2^n, 2^{-1}) \approx \epsilon X$, for $X \subseteq 2^n$ defined by $C$, is not affected by a particular choice of the hard boolean function generated by $\alpha$, we get that $\textit{APC}_1$ proves

$$
\text{LB}_{tt}(y, 2||y||/4, 2||y||(1/2 - 1/2||y||/4)) \wedge ||y|| = S(C, 2^n, 2^{-1}) \rightarrow S_{zz}(C, 2^n, 2^{-1}, y) \approx \epsilon X,
$$
where \( Sz \) is defined as \( \text{Size} \) with the only difference that the call to \( \alpha(x) \) on \( C, 2^n, 2^{\epsilon^{-1}} \) is replaced by \( y \) and \( S(C, 2^n, 2^{\epsilon^{-1}}) = ||x|| \) for a \( \text{PV} \)-function \( S \).

This allows us to express \( \Pr_{x < t}[x \in X]_e = a \), where \( \epsilon^{-1} \in \text{Log} \) and \( X \cap t \subseteq 2^{|t|} \) is defined by a circuit \( C \), without a \( \text{PV}_1(\alpha) \) function, by formula

\[
\forall y \ (\text{LB}_{tt}(y, 2||y||/4, 2||y|| (1/2 − 1/2||y||/4)) ∧ ||y|| = S(C, 2^{|t|}, 2^{\epsilon^{-1}}) \rightarrow Sz(C, 2^{|t|}, 2^{\epsilon^{-1}}, y)/t = a).
\]

We denote the resulting formula by \( \Pr^y_{x < t}[x \in X]_e = a \). We will use the notation \( \Pr^y_{x < t}[x \in X]_e \) in equations with the intended meaning that the equation holds for the value \( Sz(\cdot, \cdot, \cdot)/t \) under corresponding assumptions. For example, \( t \cdot \Pr^y_{x < t}[x \in X]_e \preceq_\delta a \) stands for \( \forall y, \exists v, \exists \text{ circuit } \hat{C} \) (defining a surjection) which witnesses that \( \text{LB}_{tt}(y, 2||v||/4, 2||v|| (1/2 − 1/2||v||/4)) ∧ ||v|| = S(C, 2^{|t|}, 2^{\epsilon^{-1}}) \) implies \( Sz(C, 2^{|t|}, 2^{\epsilon^{-1}}, y) \preceq_\delta a' \).

The definition of learning can be now expressed without a \( \text{PV}_1(\alpha) \) function: If circuit class \( C \) is defined by a \( \text{PV} \)-function, the statement that a given oracle algorithm \( L \) (given by a \( \text{PV} \)-function with oracle queries) learns a circuit class \( C \) over the uniform distribution up to error \( \epsilon \) with confidence \( \delta \) can be expressed as before with the only difference that we replace \( \Pr_w[L^f(1^n, w) (1 − \epsilon)-\text{approximates } f]_\gamma \geq \delta \) by

\[
\Pr^y_w[L^f(1^n, w) (1 − \epsilon)-\text{approximates } f]_\gamma \geq \delta.
\]

Since the resulting formula \( A \) defining learning is not \( \Pi^b_1 \) (because of the assumption \( \text{LB}_{tt} \)) we cannot translate it to propositional logic. We will sidestep the issue by translating only the formula \( B \) obtained from \( A \) by deleting subformula \( \text{LB}_{tt} \) (but leaving \( ||y|| = S(\cdot, \cdot, \cdot) \) intact) and replacing the variables \( y \) by fixed bits representing a hard boolean function. In more detail, \( \Pi^b_1 \) formula \( B \) can be translated into a sequence of propositional formulas \( \text{lear}^y_\gamma(L, C, \epsilon, \delta) \) expressing that “if \( C \in C \) is a circuit computing \( f \), then \( L \) querying \( f \) generates a circuit \( D \) such that \( \Pr[D(x) = f(x)] \geq 1 − \epsilon \) with probability \( \geq \delta \), which is counted approximately with precision \( \gamma \)”.

Note that \( C, f \) are represented by free variables and that there are also free variables for error \( \gamma \) from approximate counting and for boolean functions \( y \). As in the case of \( \text{tt} \)-formulas, we fix \( |f| = 2^n \), so \( n \) is not a free variable. Importantly, \( \text{lear}^y_\gamma(L, C, \epsilon, \delta) \) does not postulate that \( y \) is a truth-table of a hard boolean function. Nevertheless, for any fixed (possibly non-uniform) bits representing a sequence of boolean functions \( h = \{h_m\}_{m > m_0} \) such that \( h_m \) is not \((1/2 + 1/2^{m/4})\)-approximable by any circuit of size \( 2^{m/4} \), we can obtain formulas \( \text{lear}^y_\gamma(L, C, \epsilon, \delta) \) by substituting bits \( h \) for \( y \).

Using a single function \( h \) in \( \text{lear}^y_\gamma(L, C, \epsilon, \delta) \) does not ruin the fact that (the translation of function) \( Sz \) approximates the respective probability with accuracy \( \gamma \) because \( Sz \) queries a boolean function \( y \) which depends just on the number of atoms representing \( \gamma^{-1} \) and on the size of the circuit \( D \) defining the predicate we count together with the number of inputs of \( D \). The size of \( D \) and the number of its inputs are w.l.o.g. determined by the number of inputs of \( f \).
If we are working with formulas $\text{lear}^h_{\alpha}(L, C, \epsilon, \delta)$, where $h$ is a sequence of bits representing a hard boolean function, in a proof system which cannot prove efficiently that $h$ is hard, our proof system might not be able to show that the definition is well-behaved - it might not be able to derive some standard properties of the function $Sz$ used inside the formula. Nevertheless, in our theorems this will never be the case: our proof systems will always know that $h$ is hard.

In formulas $\text{lear}^h_{\alpha}(L, C, \epsilon, \delta)$ we can allow $L$ to be a sequence of nonuniform circuits, with a different advice string for each input length. One way to see that is to use additional input to $L$ in $\Pi_1^b$ formula $B$, then translate the formula to propositional logic and substitute the right bits of advice for the additional input. Again, the precise encoding of the formula $\text{lear}^h_{\alpha}(L, C, \epsilon, \delta)$ does not matter very much to us but in order to simplify proofs we will assume that $\text{lear}^h_{\alpha}(L, \text{Circuit}[n^k], \epsilon, \delta)$ has the form $\lnot tt(f, n^k) \rightarrow R$, where $n, k$ are fixed, $f$ is represented by free variables and $R$ is the remaining part of the formula expressing that $L$ generates a suitable circuit with high probability.

3.3 Automatability

Let $\Phi$ be a class of propositional formulas. We say that a proof system $P$ is automatable w.r.t. $\Phi$ up to proofs of size $s$ if there is a PV-function $A$ such that for each $\phi \in \Phi$ and each $t$-size $P$-proof of $\phi$ with $t \leq s$, $A(\phi, 1^t)$ is a $P$-proof of $\phi$.

In our main theorem we will need a slightly modified notion of automatability where the automating algorithm outputs a proof of a given tautology $\phi$ which is not much longer than a proof of an associated tautology $\psi$. (Formula $\psi$ will be closely related to $\phi$: while $\phi$ will express a worst-case lower bound, $\psi$ will express an average-case lower bound for the same function.)

Let $\Phi$ be a class of pairs of propositional formulas. We say that a proof system $P$ is automatable w.r.t. $\Phi$ up to proofs of size $s$ if there is a PV-function $A$ such that for each pair $\langle \psi, \phi \rangle \in \Phi$ and each $t$-size $P$-proof of $\psi$ with $t \leq s$, $A(\phi, 1^t)$ is a $P$-proof of $\phi$.

**Propositional version.** If $\Phi$ is defined by a PV-function and $s \in Log$ is a PV-function, the statement that an algorithm $A$ (given by a PV-function) automates system $P$ w.r.t. $\Phi$ up to proofs of size $s$ is $\Pi_1^b$. Therefore, it can be translated into a sequence of propositional formulas $\text{aut}_P(A, \Phi, s)$. Again, in formulas $\text{aut}_P(A, \Phi, s)$ we can allow $A$ to be a sequence of nonuniform circuits and $s$ to be arbitrary possibly nonuniform parameter.

4 Lower bounds versus learning in proof complexity

In this section we show how automatability together with efficient provability of a lower bound, or together with optimality, implies efficient learning. This shows some of the methods used in the proof of Theorem 10, but the proof of Theorem 10 can be read
Theorem 6. Let $k \geq 1$ be a constant and $P$ be a proof system which simulates $\mathbf{EF}$ such that: I. for each $P$-proof $\pi_0$ of $\phi$ and each $P$-proof $\pi_1$ of $\phi \rightarrow \psi$, there is a poly$(|\pi_0|, |\pi_1|)$-size $P$-proof of $\psi$; II. for each $P$-proof $\pi$ of $\phi$ and a possibly partial substitution $\rho$ of atoms of $\phi$ by arbitrary formulas, there is a poly$(|\pi|, |\phi|_\rho)$-size $P$-proof of $\phi|_\rho$, where $\phi|_\rho$ is the formula $\phi$ after applying substitution $\rho$.

Assume that $P$ proves efficiently $\text{tt}(h, 3n^k)$, for some boolean function $h = \{h_n\}_{n>n_0}$ and some $n_0$. Then, automatability of $P$ implies that for each $\gamma \in (0, 1)$, $\text{Circuit}[n^{k\gamma/a}]$ is learnable by $\text{Circuit}[2^{O(n^\gamma)}]$ over the uniform distribution, with non-adaptive membership queries, confidence 1, up to error $1/n^{k\gamma/a}$, where $a$ is an absolute constant.

Proof sketch. By Theorem 5, it suffices to construct a $P/\text{poly}$-natural property useful against $\text{Circuit}[n^k]$. This is achieved by the proof of Theorem 2, which we sketch - more details can be found in the proof of Theorem 10 (direction 2. $\rightarrow$ 1.). Assume $P$ proves efficiently $\text{tt}(h_n, 3n^k)$ for some boolean function $h = \{h_n\}_{n>n_0}$. If $P$ simulates $\mathbf{EF}$ and satisfies properties I.-II., then $P$ proves efficiently also

$$\text{tt}(g, n^k) \lor \text{tt}(h_n \oplus g, n^k),$$

where $g$ is represented by free variables and $h_n \oplus g$ is a bitwise XOR of $h_n$ and $g$. This is because an $n^k$-size circuit $C_1$ computing $g$ and an $n^k$-size circuit $C_2$ computing $h_n \oplus g$ can be combined into a $3n^k$-size circuit $C_1 \oplus C_2$ computing $h_n$. (Properties I.-II. can be used to simulate Frege rules, see Lemma 3.) Automatability of $P$ now implies the existence of a $P/\text{poly}$-natural property useful against $\text{Circuit}[n^k]$: For each boolean functions $g$, we can either find efficiently a $P$-proof of $\text{tt}(g, n^k)$ or we recognize that $\text{tt}(h_n \oplus g, n^k)$ holds (if $\text{tt}(h_n \oplus g, n^k)$ did not hold, we could use properties I.-II. to substitute its falsifying assignment to the proof of (4.1) and obtain a short $P$-proof of $\text{tt}(g, n^k)$ - formally, we use here also the fact that $P$-proves efficiently $\phi(b)$, whenever $b$ is a satisfying assignment of $\phi$, see Lemma 3, Item 2.), and one of these options happens for at least 1/2 of all functions $g$.

Theorem 7 (Optimality and automatability implies learning). If there is an optimal proof system which is automatable, then for each $\gamma \in (0, 1)$, each $k \geq 1/\gamma$, for infinitely many $n$, $\text{Circuit}[n^{k\gamma}]$ is learnable by $\text{Circuit}[2^{O(n^\gamma)}]$ over the uniform distribution, with non-adaptive membership queries, confidence $1/2 - 1/2^{4n^\gamma}$, up to error $1/2 - 1/2^{3n^\gamma}$.

Proof sketch. If $\text{SAT} \in \text{Circuit}[3n^k]$ for infinitely many $n$, then there is a $P/\text{poly}$-natural property useful against $\text{Circuit}[n^{\log n}]$, for infinitely many $n$, and the conclusion of the theorem follows from the proof of Theorem 5, see Theorem 9. Assume $\text{SAT} \notin \text{Circuit}[3n^k]$ holds for all sufficiently big $n$. Then there is a proof system $P$ which proves efficiently
tt(SAT, 3n^k) for all sufficiently big n: P is by definition allowed to derive every substitutional instance of tt(SAT, 3n^k) (which is a formula recognizable in 2^{O(n)}-time) and, otherwise, it proceeds as EF. Therefore, we can follow the proof of Theorem 6, noting that the automatability of P can be replaced by the automatability of the optimal system, and obtain the desired conclusion.

\[ \text{Theorem 8. For each } \gamma \in (0, 1), \text{ each } k \geq 1/\gamma, \text{ there is a proof system } P \text{ such that} \]
\[ \text{1.) } P \text{ is automatable if and only if } 2.) \text{ for infinitely many } n, \text{ Circuit}[n^{k\gamma}] \text{ is learnable by Circuit}[2^{O(n^\gamma)}] \text{ over the uniform distribution, with non-adaptive membership queries, confidence } 1/2^{4n^\gamma}, \text{ up to error } 1/2 - 1/2^{3n^\gamma}. \]

Proof. Let \( \gamma \in (0, 1) \) and \( k \geq 1 \). If Item 2.) from the statement of the theorem holds, then let \( P \) be a proof system with exponentially long proofs of all tautologies, so \( P \) is trivially automatable and the equivalence holds. Suppose Item 2.) does not hold. Then, similarly as in the proof of Theorem 7, we conclude that SAT \( \notin \) Circuit[3n^ka] for all sufficiently big \( n \) and constant \( a \) from Theorem 6. It remains to observe that the proof system \( P \) from the proof of Theorem 7, with tt(SAT, 3n^ka) instead of tt(SAT, 3n^k), is not automatable. This follows by Theorem 6. Alternatively, in the case that Item 2.) fails, we can use the fact that \( P \neq \text{NP} \) implies the existence of a non-automatable proof system, cf. [29, Lemma 5.3].

The disadvantage of Theorem 8 in comparison to Corollary 3 is that it does not provide an explicit definition of \( P \). It is possible to get an explicit construction of \( P \) based on a hardness assumption, but the hardness assumption would then itself falsify both sides of the desired equivalence. Another advantage of Corollary 3 is that it applies to all well-behaved proof systems simulating a ‘ground’ system \( P \).

### 4.1 Learning algorithms from natural proofs in APC_1

An essential component of the transformation of natural proofs into learning algorithms is the Nisan-Wigderson generator and specific combinatorial designs on which it is based, cf. [25]. In order to formalize the transformation in APC_1 we would need to construct combinatorial designs in APC_1. A construction of combinatorial designs has been formalized already in [14], but our transformation requires algebraic construction of designs obtained by evaluating polynomials on a finite field. A complication is that the algebraic construction uses Bertrand’s postulate of the existence of a prime between \( x \) and 2x. We will bypass the problem of proving Bertrand’s postulate in APC_1 simply by assuming the existence of such a prime.\(^8\) That is, we will not prove the existence of combinatorial designs unconditionally, but only under the assumption of the primality of a number in the

---

\(^8\)It is possible that the desired formalization of Bertrand’s postulate can be obtained from the work of Paris, Wilkie and Woods [27].
interval \([x, 2x]\). Fortunately, in our setting, we will have \(2^x \in \text{Log}\), so once we translate the resulting statements to propositional logic, we will use Bertrand’s postulate (even though we have not formalized it in \(\text{APC}_1\)) to conclude that the assumption will have a trivial WF-proof. The construction of Nisan and Wigderson otherwise does not require developing new methods, but we need to verify that each step is doable in \(\text{APC}_1\). In fact, \(\text{PV}_1\) will suffice.

For \(x \in \{0, 1\}^n\) and \(S \subseteq [n]\), denote by \(x|S\) an \(|S|\)-bit string consisting of \(x_i\)'s such that \(i \in S\) (in the natural order).

**Lemma 2** (in \(\text{PV}_1\)). Let \(d \geq 2\). If \(2^n \in \text{Log}\) and \(n^d \leq p \leq 2n^d \in \text{Log}\) is a prime, there is a \(2^n \times m\) 0-1 matrix \(A\) with \(n^d\) ones per row and \(m = pn^d\) which is also an \((n, n^d)\)-design meaning that for \(J_i(A) := \{j \in [m]; a_{i,j} = 1\}\), for each \(i \neq j\), \(|J_i(A) \cap J_j(A)| \leq n\) and \(|J_i(A)| = n^d\). Moreover, for sufficiently big \(n\), there are \(n^{9d}\)-size circuits which given \(i \in \{0, 1\}^n\) and \(w \in \{0, 1\}^m\) output \(w|J_i(A)\).

**Proof.** Let \(n^d \leq p \leq 2n^d\) be a prime and \(F\) a field of size \(p\). \(F\) can be constructed in \(\text{PV}_1\), cf. [15, Section 4.3]. We construct the matrix \(A\) so that the \(i\)-th row consists of positions \((u, v)\), for \(u \in \{0, \ldots, n^d - 1\}\), \(v \in F\), with \(1\)'s exactly on positions \((u, q(u))\), for \(u \in \{0, \ldots, n^d - 1\}\), where \(q\) is a polynomial of degree \(n\) with binary coefficients corresponding to the binary representation of \(i\). Formally, an \(n\)-degree polynomial is represented by a sequence of its coefficients. The evaluation of an \(n\)-degree polynomial on an element from \(F\) can be done in \(\text{poly}(n)\)-time and is well-defined in \(\text{PV}_1\), cf. [15, Section 4.3]. By definition, \(A\) is a \(2^n \times m\) 0-1 matrix with \(n^d\) ones per row. \(|J_i(A) \cap J_j(A)| \leq n\), for \(i \neq j\), follows from the fact that a non-zero \(n\)-degree polynomial over \(F\) has \(\leq n\) roots, which is provable in \(\text{PV}_1\), cf. [15, Lemma 4.3.6].

It remains to prove the ‘moreover’ part. The \(n^{9d}\)-size circuit first evaluates the \(i\)-th polynomial on inputs \(0, \ldots, n^d - 1\). This way it obtains \(J_i(A)\) and \(w|J_i(A)\). Evaluating an \(n\)-degree polynomial on an input from \(F\) can be done \(\text{PV}_1\)-provably by a circuit of size \(n \cdot \text{poly}(d \log n)\). (In more detail, we compute \(x, x^2, \ldots, x^n\) over \(F\) in a standard way by an \(n \cdot \text{poly}(d \log n)\)-size circuit, then multiply \(x^i\)'s with the corresponding coefficients and sum the results over \(F\), which takes another \(n \cdot \text{poly}(d \log n)\)-size circuit.) Thus, given \(i\), the \(n^d\) indices from \(J_i(A)\) can be generated by a circuit of size \(n^{d+1} \cdot \text{poly}(d \log n)\). Given \(w\) and indices from \(J_i(A)\), the bits \(w|J_i(A)\) can be generated by a circuit of size \(O(n^{d+1}d \log n)\) so the total size of the circuit generating \(w|J_i(A)\) on \(i\) and \(w\) is \(\leq n^{d+2} + O(dn^{2d+1}) \leq n^{9d}\). \(\square\)

The formalization of the transformation of natural proofs into learning algorithms follows from a straightforward inspection of the original proof as well.

**Theorem 9.** There is a \(\text{PV}\)-function \(L\) such that \(\text{APC}_1\) proves: For \(k \geq 1\), \(d \geq 2\), \(2^{n^d}, n^{dk}, \delta^{-1} \in \text{Log}\), \(\delta < 1/N^3\) and a prime \(n^d \leq p \leq 2n^d\), let \(R_N\) be a circuit with \(N = 2^n\) inputs such that for sufficiently big \(N\),
1. $R_N(x) = 1$ implies that $x$ is a truth-table of a boolean function with $n$ inputs hard for Circuit[10dk],

2. $\{x \mid R_N(x) = 1\} \geq \delta \ 2^N/N$.

Then, circuits with $n^d$ inputs and size $n^{dk}$ are learnable by circuit $L(R_N, p)$ over the uniform distribution with membership queries, confidence $1/N^4$, up to error $1/2 - 1/N^3$. Here, the confidence is counted approximately with error $\delta$ using PV-function $Sz$ and the corresponding assumptions $LB_{\text{tt}}$ expressing hardness of a boolean function $y$, i.e. using formulas $\Pr[y[\cdot]]_{\delta}$.

Proof. We reason in APC1. Consider a Nisan-Wigderson generator based on a circuit $C$ which we aim to learn. Specifically, for $d \geq 2$ and $n^{2d} \leq m = pn^d \leq 2n^{2d}$, let $A = \{a_{ij}\}_{i,j} \in [N]$ be an $N \times m$ 0-1 matrix with $n^d$ ones per row. Then define an NW-generator $NW_C : \{0,1\}^m \mapsto \{0,1\}^N$ as

$$(NW_C(w))_i = C(w|J_i(A)).$$

We can assume that $A$ is, in addition, a combinatorial design from Lemma 2. Therefore, if $C$ has $n^d$ inputs and size $n^{dk}$, then for each $w \in \{0,1\}^m$, $(NW_C(w))_i$ is a function on $n$ inputs $x$ computable by circuits of size $n^{10dk}$, for sufficiently big $n$. We want to learn $C$ by a circuit $L(R_N, p)$ of size $2^{O(n)}$.

We will use circuits $R_N$ which function as distinguishers for $NW_C$: By the assumption of the theorem, a trivial surjection witnesses that $\{w \mid R_N(NW_C(w)) = 1\} \subseteq 0$. Hence, by Proposition 1 ii), $2^m \cdot \Pr_w[R_N(NW_C(w)) = 1]_{\delta} \leq 0$, and by Proposition 2 1.ii), $\Pr_w[R_N(NW_C(w)) = 1]_{\delta} < 2\delta$, for universally quantified $y$. Similarly, by the assumption of the theorem $\{u \mid R_N(u) = 1\} \geq \delta 2^N/N$, and $\Pr_u[R_N(u) = 1]_{\delta} > 1/N - 3\delta$. Therefore,

$$\Pr_u[R_N(u) = 1]_{\delta} - \Pr_w[R_N(NW_C(w)) = 1]_{\delta} > 1/N - 5\delta.$$

$L(R_N, p)$ chooses a random $i \in [N]$, random bits $r_1, \ldots, r_N$, random $w' \in \{0,1\}^{m-n^d}$ and queries the bits $C(w|J_i(A)), \ldots, C(w|J_{i-1}(A))$, for all $w \in \{0,1\}^m$ such that $w|J_i(A) = w'$. Since $A$ is an $(n,n^d)$-design, there are just $2^{O(n)}$ such queries. For $w \in \{0,1\}^m$, let $p_i := R_N(C(w|J_i(A)), \ldots, C(w|J_{i-1}(A)), r_i, \ldots, r_N)$. Then $L(R_N, p)$ outputs a circuit $L'$ which on $x \in \{0,1\}^{n^d}$ constructs $w \in \{0,1\}^m$ such that $w|J_i(A) = x$ while $w|m\setminus J_i(A) = w'$ and predicts the value $C(x)$ by outputting $\neg r_i$ iff $p_i = 1$.

We want to show that $L'$ approximates $C$ with high probability.

First, note that by Proposition 1 iii), inequality $\{u \mid R_N(u) = 1\} \geq \delta 2^N/N$ implies $\{w, r_1, \ldots, r_N \mid p_1 = 1\} \geq \delta 2^{m-N}/N$, so we have $\Pr_{w,r_1,\ldots,r_N}[p_1 = 1]_{\delta} > 1/N - 3\delta$ and $\Pr_{w,r_1,\ldots,r_N}[p_{N+1} = 1]_{\delta} < 2\delta$. Consequently, there exists $y_1, \ldots, y_{N+1}$ and $i \in [N]$ such that

$$\Pr_{w,r_1,\ldots,r_N}[p_i = 1]_{\delta} - \Pr_{w,r_1,\ldots,r_N}[p_{i+1} = 1]_{\delta} > 1/N^2 - 5\delta/N. \quad (4.2)$$
Otherwise, for all \(y_1, \ldots, y_{N+1}\), for all \(i\), \(\Pr^y[p_i = 1]_\delta - \Pr^{y_i+1}[p_{i+1} = 1]_\delta \leq 1/N^2 - 5\delta/N\), and by \(\Sigma_0^p(\text{PV})\)-induction \(\Pr^y[p_i = 1]_\delta - \Pr^{y_{N+1}}[p_{i+1} = 1]_\delta \leq 1/N - 5\delta\). As \(\text{APC}_1\) proves that some \(y_1, \ldots, y_{N+1}\) satisfy the assumptions of \(\Pr^y[.]_\delta\), for all \(j = 1, \ldots, N + 1\), this would be a contradiction. (The existence of \(y_1, \ldots, y_{N+1}\) is proved analogously as Proposition 3 and the proof does not require sharply bounded collection scheme - we can construct \(y_1, \ldots, y_{N+1}\) from the string \(w\) in [15, Lemma 4.1.8].) That is, (4.2) means that \(y_i\)'s satisfy formulas \(\text{LB}_{\mathbb{R}}\) (with suitable parameters), have the right length \(||y_i||\) and the corresponding functions \(Sz\) witness that the difference of the respective probabilities is big.

Since trivial surjections witness that, for \(i \in [N], z = w', x, r_1, \ldots, r_N < 2^{m+N}\),

\[
\{z \mid L'(x) = C(x)\} \geq_0 \{z \mid p_i = 1 \land r_i \neq C(x)\} \cup \{z \mid p_i \neq 1 \land r_i = C(x)\},
\]

and \(\{z \mid p_i = 1 \land r_i \neq C(x)\} \cap \{z \mid p_i \neq 1 \land r_i = C(x)\} \approx_0 0\), by Proposition 2 1.iv) and Proposition 1 ii),

\[
2^{m+N} \Pr_z^y[L'(x) = C(x)]_\delta \geq_{4\delta} 2^{m+N} \Pr_z^y[p_i = 1 \land r_i \neq C(x)]_\delta + 2^{m+N} \Pr_z^y[p_i \neq 1 \land r_i = C(x)]_\delta.
\]

Notably, when we applied Proposition 2 1.iv), we switched the domain of surjections from \(2^{m+N}\) to \(2^{m+N+1}\). This does not affect our inequalities because surjections witnessing \(X \preceq_\delta Y\), for \(X, Y \subseteq 2^{m+N}\) can be used to witness \(X \preceq_\delta Y\), where we see \(X, Y\) as subsets of \(2^{m+N+1}\) (and consider the error \(\delta\) w.r.t. \(2^{m+N+1}\), not w.r.t. \(2^{m+N}\)), but we need to take it into account when we now apply Proposition 2 1.ii) to conclude that\(^9\)

\[
\Pr_z^y[L'(x) = C(x)]_\delta > \Pr_z^y[p_i = 1 \land r_i \neq C(x)]_\delta + \Pr_z^y[p_i \neq 1 \land r_i = C(x)]_\delta - 10\delta.
\]

Further, we have \(\{z \mid p_i = 1 \land r_i = C(x)\} \cup \{z \mid p_i \neq 1 \land r_i = C(x)\} \approx_0 2^{m+N}/2\), so \(\Pr_z^y[p_i \neq 1 \land r_i = C(x)]_\delta > 1/2 - \Pr_z^y[p_i = 1 \land r_i = C(x)]_\delta - 8\delta\) and

\[
\Pr_z^y[L'(x) = C(x)]_\delta > \Pr_z^y[p_i = 1 \land r_i \neq C(x)]_\delta + \frac{1}{2} - \Pr_z^y[p_i = 1 \land r_i = C(x)]_\delta - 18\delta. \quad (4.3)
\]

Next, we similarly derive \(\{z \mid p_i = 1\} \approx_0 \{z \mid p_i = 1 \land r_i = C(x)\} \cup \{z \mid p_i = 1 \land r_i \neq C(x)\}\) and

\[
\Pr_z^y[p_i = 1]_\delta - 10\delta < \Pr_z^y[p_i = 1 \land r_i = C(x)]_\delta + \Pr_z^y[p_i = 1 \land r_i \neq C(x)]_\delta. \quad (4.4)
\]

Now, observe that \(\{z \mid p_i = 1 \land r_i = C(x)\} \approx_0 \{w, r_1, \ldots, r_N \mid p_{i+1} = 1 \land r_i = 1\}\). This yields \(2^{m+N} \Pr_z^{y''}[p_i = 1 \land r_i = C(x)]_\delta \geq_{2\delta} 2^{m+N} \Pr_z^{w, r_1, \ldots, r_N}[p_{i+1} = 1 \land r_i = 1]_\delta\).

Analogously, \(2^{m+N} \Pr_z^{y''}[p_i = 1 \land r_i = C(x)]_\delta \geq_{2\delta} 2^{m+N} \Pr_z^{w, r_1, \ldots, r_N}[p_{i+1} = 1 \land r_i = 0]_\delta\). As

---

\(^9\)If we worked on the domain \(2^{m+N}\) the resulting error would not be \(10\delta\) but \(5\delta\).
\{w, r_1, \ldots, r_N \mid p_{i+1} = 1 \land r_i = 1\} \cup \{w, r_1, \ldots, r_N \mid p_{i+1} = 1 \land r_i = 0\} \approx 0 \{w, r_1, \ldots, r_N \mid p_{i+1} = 1 \land r_i = 0\}, we have also \(2^{m+N} \Pr_{w, r_1, \ldots, r_N} y^{m+1} [p_{i+1} = 1 \land r_i = 1]_\delta + 2^{m+N} \Pr_{w, r_1, \ldots, r_N} y^{m+1} [p_{i+1} = 1 \land r_i = 0]_\delta \approx_{\delta} 2^{m+N} \Pr_{w, r_1, \ldots, r_N} y^{m+1} [p_{i+1} = 1]_\delta.\) It follows that \(2 \Pr_{\delta} [p_i = 1 \land r_i = C(x)]_\delta - 18 \delta < \Pr_{\delta} y_{i+1} [p_{i+1} = 1]_\delta.\) (4.5)

Combining (4.2) - (4.5) shows that for some \(i \in [N],\)

\[\Pr_{\delta} y_{w, r_1, \ldots, r_N} [L'(x) = C(x)]_\delta > 1/2 + 1/N^2 - 5\delta/N - 46\delta,\] (4.6)

where \(L'\) is generated by \(L(R_N, p)\) on \(w', x, r_1, \ldots, r_N\) and \(i.\) As in the case of (4.2), \(y\) is quantified existentially in (4.5) and satisfies the assumptions of \(\Pr[y][]\).

It remains to observe that (for universally quantified \(y\))

\[\Pr_{\delta} y_{w', x, \ldots, r_N} [L(R_N, p) \ (1/2 + 1/N^3)\text{-approximates } C]_\delta > 1/N^4.\]

For the sake of contradiction, assume this is not the case. Then \(\{w', i, r_1, \ldots, r_N \mid L' \ (1/2 + 1/N^3)\text{-approximates } C\} \lesssim_{\delta} 2^{m-n^d+n^N}/N^4\) and by averaging (Proposition 2, Item 3),

\[\{w', x, i, r_1, \ldots, r_N \mid L' \ (1/2 + 1/N^3)\text{-approximates } C\} \lesssim_{2\delta} 2^{m+n^N}/N^4.\]

Therefore, for each \(i \in [N],\)

\[\{w', x, i, r_1, \ldots, r_N \mid L' \ (1/2 + 1/N^3)\text{-approximates } C\} \lesssim_{2\delta} 2^{m+N}/4N^2.\] (4.7)

(Otherwise, by Proposition 2 1.i), there is a surjection witnessing the opposite inequality, which can be used to witness also \(\{w', x, i, r_1, \ldots, r_N \mid L' \ (1/2+1/N^3)\text{-approximates } C\} \gtrsim_{2\delta} 2^{m+N}/4N^2 \) and \(2^{m+N}/4N^2 \lesssim_{4\delta} 2^{m+n^N}/N^4 = 2^{m+N}/N^3;\) contradicting Proposition 2 1.ii)

for \(\delta < 1/N^3.\)

On the other hand, for each \(w', i, r_1, \ldots, r_N,\)

\[\{x \mid L'(x) = C(x) \land L' < (1/2 + 1/N^3)\text{-approximates } C\} \lesssim_{0} (1/2 + 1/N^3)^{2n^d},\]

which can be counted exactly because \(2^{n^d} \in Log.\) Hence, by averaging, for each \(i \in [N],\)

\[\{w', x, i, r_1, \ldots, r_N \mid L'(x) = C(x) \land L' < (1/2 + 1/N^3)\text{-approximates } C\} \lesssim_\delta (1/2+1/N^3)^{2m+N}.\]

The last approximation together with (4.7) imply that for each \(i \in [N],\)

\[\{w', x, i, r_1, \ldots, r_N \mid L'(x) = C(x)\} \lesssim_{3\delta} (1/2 + 1/2N^2)^{2m+N},\]

which in turn implies that (for universally quantified \(y\)) \(\Pr_{w', x, r_1, \ldots, r_N} y_{w', x, r_1, \ldots, r_N} [L'(x) = C(x)]_\delta < 1/2 + 1/2N^2 + 5\delta,\) contradicting (4.6) if \(\delta < 1/N^3\) and \(N\) is sufficiently big. \(\square\)
5 Main theorem

Our main theorem holds for any ‘decent’ proof system p-simulating WF, which is well-behaved in the sense that it APC1-provably satisfies some basic properties.

**Definition 6 (APC1-decent proof system).** A propositional proof system P is APC1-decent if the language L of P is finite and complete, i.e. L consists of connectives of constant arity such that each boolean function of every arity can be expressed by an L-formula, P proves efficiently its own reflection principle, i.e. formulas stating that if π is a P-proof of φ then φ holds, cf. [21], and there is a PV-function F such that APC1 proves:

1. P p-simulates WF, i.e. F maps each WF-proof of φ to a P-proof of φ.
2. P admits substitution property: F maps each triple ⟨φ, ρ, π⟩ to a P-proof of φ|ρ, where π is a P-proof of φ and φ|ρ is formula φ after applying substitution ρ which replaces atoms of φ by formulas.
3. F maps each pair ⟨π, π’⟩, where π is a P-proof of φ and π’ is a P-proof of φ → ψ, to a P-proof of ψ.

In Definition 6, WF refers to some fixed system from the set of all WF systems. It follows from the proof of Lemma 3 that if APC1 proves that P p-simulates a WF-system Q, then for every WF-system R, APC1 proves that P p-simulates R, so the particular choice of the WF-system does not matter. When we use connectives ∧, ∨, ¬, → in an APC1-decent system P, we assume that these are expressed in the language of P.

**Lemma 3.** Each WF system is APC1-decent. Moreover, for each APC1-decent proof system P the following holds.

1. For every Frege rule which derives φ from φ1, . . . , φk, there is a PV-function F such that APC1 proves that F maps each (k + 1)-tuple ⟨π1, . . . , πk, ρ⟩ to a P-proof of φ|ρ, where πi is a P-proof of φi|ρ for a substitution ρ replacing each atom of φ, φ1, . . . , φk by a formula.
2. There is a PV-function F such that APC1 proves that F maps each pair ⟨φ, b⟩, for assignment b satisfying formula φ, to a P-proof of φ(b).
3. Let π be a P-proof of E → φ, where E defines a computation of a circuit which is allowed to use atoms from φ as inputs but other atoms of E do not appear in φ, i.e. E is the conjunction of extension axioms of EF built on atoms from φ. Then, there is a poly(|π|)-size P-proof of φ.

**Proof.** WF is known to prove efficiently its own reflection principle, cf. [14]. In order to show that it is APC1-decent, it thus suffices to prove that it satisfies Items 1-3 from Definition 6.
Item 2 is established already in PV₁ by Σ₁ᵇ-induction on the length of the proof π (which can be used because of ∀Σ₁ᵇ-conservativity of S₂¹ over PV₁): F replaces each circuit C from π by C|₀ and preserves all WF-derivation rules.

Item 1 holds trivially if the given WF-system P is the WF-system P’ from Definition 6. Otherwise, we use implicational completeness of P and the completeness of the language of P to simulate all O(1) Frege rules of P’ by O(1) steps in P. (This does not require that the implicational completeness of P is provable in APC₁ because we need to simulate only O(1) Frege rules of finite size). Similarly, by Σ₁ᵇ-induction and the completeness of the language of P, we simulate each circuit in the language of P’ by a circuit in the language of P and show that this simulation preserves the similarity rule. Then, given an s-size P’-proof of φ, we obtain a poly(s)-size P-proof of φ using the simulation of Frege rules of P’, the similarity rule and dWPHP axiom, together with substituting the right circuits in Frege rules. This is done again in PV₁ by Σ₁ᵇ-induction on the length of the P’-proof.

Item 3 follows by simulating modus ponens as in the proof of Item 1.

For the ‘moreover’ part, we consider three cases:

Item 1: As in the case of WF, observe that by completeness, P proves φ₁ → ... → φ_k → φ and that this fact is provable in PV₁. By Definition 6, Item 2, APC₁ can construct a P-proof of φ₁|₀ → ... → φ_k|₀ → φ|₀. The claim then follows from k applications of Definition 6, Item 3.

Item 2: By Definition 6, Item 1, it suffices to prove the claim for WF. This follows from a Σ₁ᵇ-induction on the complexity of φ, where we strengthen the claim to: “For each multi-output circuit C and complete assignment b, F outputs a k|C|^2-size WF-proof which contains every single-output circuit C’(b) such that C’ is a subcircuit of C satisfied by b or C’ is ¬C’” for a subcircuit C” of C falsified by b. Here, k is an absolute constant”. The strengthened claim holds for literals by simulating O(1) Frege rules, which we have by Item 1. Further, O(1) Frege rules (and their substitutional instances) suffice to prove that if the claim holds for a multi-output circuit B and we extend B by one gate to a multi-output circuit B’, then the claim holds for B’ as well. (The length of the proof corresponding to B’ is ≤ k(|B|^2 + O(|B’|) < k|B’|^2, for sufficiently big k, where we use the choice of WF which guarantees linear increase of proof-size when applying substitutions and modus ponens.)

Item3: This is easy to see for P = EF, since EF can introduce extension axioms. For APC₁-decent system P, observe that a P-proof π of E → φ implies the existence of a poly(|π|)-size EF-proof of RefᵟP ∨ E → φ, where RefᵟP postulates the reflection principle for P instantiated by π, which further implies the existence of a poly(|π|)-size EF-proof of RefᵟP → φ, and finally a poly(|π|)-size P-proof of φ.

APC₁-decent proof systems can be much stronger than WF. For example, consider ZFC as a propositional proof system: a ZFC-proof of propositional formula φ is a ZFC-proof of the statement encoding that φ is a tautology. We can add the reflection of ZFC to WF,
i.e. we will allow WF to derive (substitutional instances of) formulas stating that “If $\pi$ is a ZFC-proof of $\phi$, then $\phi$ holds.” The new system is as strong as ZFC w.r.t. tautologies and it is easy to see that it is APC$_1$-decent. (The reflection of the system can be proved in APC$_1$ extended with an axiom postulating the reflection for ZFC.)

**Theorem 10** (Learning versus automatability). Let $P$ be an APC$_1$-decent proof system and assume there is a sequence of boolean functions $h = \{h_n\}_{n>n_1}$, for a constant $n_1$, such that $P$ proves efficiently $tt(h_n, 2^{n/4}, 1/2 - 1/2^{n/4})$. Then, for each constant $K$ and constant $\gamma < 1$, the following statements are equivalent.

1. **Provable learning.** For each $k \geq 1$ and $\ell \geq K + 1$, there are $2^{Kn^\gamma}$-size circuits $A$ such that for each sufficiently big $n$, $P$ proves efficiently

$$\text{lear}_{1/2^{n^\gamma}}^h(A, \text{Circuit}[n^k], 1/2 - 1/2^{Kn^\gamma}, 1/2^{Kn^\gamma}).$$

2. **Provable automatability.** For each $k \geq 1$, for each function $s(n) \geq 2^n$, there is a constant $K'$ and $s^{Kn^n}$-size circuits $B$ such that $P$ proves efficiently

$$\text{aut}_p(B, \Phi, s),$$

where $\Phi$ is the set of pairs $(tt(f, 2^{Kn^n}, 1/2 - 1/2^{Kn^\gamma}), tt(f, n^k))$ for all boolean functions $f$ with $n$ inputs.

**Proof.** (1. $\rightarrow$ 2.) We first prove the following statement in APC$_1$.

**Claim 5.1** (in APC$_1$). Assume that $\pi$ is a $P$-proof of $\text{lear}_{1/2^{n^\gamma}}^y(A, \text{Circuit}[n^k], 1/2 - 1/2^{Kn^\gamma}, \delta)$ for a circuit $A$ and a boolean function $y$ represented by fixed bits in formula $\text{lear}_{1/2^{n^\gamma}}^y(\cdot, \cdot, \cdot, \cdot)$. Further, assume that the probability that $A$ on queries to $f$ outputs a circuit $D$ such that $\Pr[D(x) = f(x)] \geq 1/2 + 1/2^{Kn^\gamma}$ is $\delta$, where the outermost probability is counted approximately with error $1/2^{Kn^\gamma}$ using PV-function $Sz$ and the corresponding assumptions $\text{LB}_n$ expressing hardness of $y$, i.e. using formulas $\text{Pr}^y[\cdot, 1/2^{n^\gamma}]$ for the same $y$ as above - we treat $y$ as a free variable here. Then there is a $\text{poly}(|\pi|)$-size $P$-proof of $tt(f, n^k)$ or $y$ does not satisfy the assumptions of $\text{Pr}^y[\cdot, 1/2^{n^\gamma}]$.

To see that the claim holds, we reason in APC$_1$ as follows. Assume $\pi$ is a $P$-proof of $\text{lear}_{1/2^{n^\gamma}}^y(A, \text{Circuit}[n^k], 1/2 - 1/2^{Kn^\gamma}, \delta)$ but $A$ on queries to $f$ outputs a circuit $(1/2 + 1/2^{Kn^\gamma})$-approximating $f$ with probability $< \delta$. Then, either $y$ does not satisfy the assumptions of $\text{Pr}^y[\cdot, 1/2^{n^\gamma}]$ or there is a trivial $2^{O(n)}$-size $P$-proof of $\neg tt(f, n^k) \rightarrow \neg R(b)$, for predicate $R$ from the definition of $\text{lear}_{1/2^{n^\gamma}}^y(A, \text{Circuit}[n^k], 1/2 - 1/2^{Kn^\gamma}, \delta)$ and a complete assignment $b$. The $P$-proof is obtained by evaluating function $Sz$ which counts the confidence of $A$ - note that functions $f, y$ and algorithm $A$ are represented inside $P$ by fixed bits so the $P$-proof just evaluates a $2^{O(n)}$-size circuit on some input, which is possible by Lemma 3, Item 2. (We use here also the fact that APC$_1$ knows that the probability
statement expressed by function \( Sz \) translates to \( \neg R \) in the negation normal form.) The formula \( \neg \tt(f, n^k) \rightarrow \neg R(b) \) is obtained from \( \neg R(b) \) by an instantiation of a single Frege rule, which is available by Lemma 3, Item 1. Applying again Lemma 3, Item 1, from a \( P \)-proof of \( \leq_{1/2^{2\ell n^\gamma}}(A, \text{Circuit}[n^k], 1/2 - 1/2^{K_n^\gamma}, 0) \) and a \( P \)-proof of \( \neg \tt(f, n^k) \rightarrow \neg R(b) \), we construct a \( \text{poly}(|\pi|) \)-size \( P \)-proof of \( \tt(f, n^k) \). This proves the claim.

Next, observe that \( \text{APC}_1 \) proves that “If for a sufficiently big \( n \) and \( \ell \geq K + 1 \) the probability that a circuit \( A \) on queries to \( f \) outputs a circuit \( (1/2 + 1/2^{K_n^\gamma}) \)-approximating \( f \) is \( \geq 1/2^{K_n^\gamma} \), where the probability is counted approximately with error \( 1/2^{\ell n^\gamma} \) using \( \text{PV} \)-function \( Sz \) and the corresponding assumptions \( \text{LB}_tt \), then there is a circuit of size \( |A| \cdot (1/2 + 1/2^{K_n^\gamma}) \)-approximating \( f \) or \( y \) does not satisfy the assumptions of \( \text{Pr}^y[\cdot]_{1/2^{2\ell n^\gamma}} \).” This is because, if such a circuit did not exist, a trivial surjection would witness that \( 2^m \) times the probability that \( A \) outputs a circuit \( (1/2 + 1/2^{K_n^\gamma}) \)-approximating \( f \), counted approximately with error \( 1/2^{\ell n^\gamma} \) using function \( Sz \), is \( \leq_{1/2^{2\ell n^\gamma}} 0 \). Here, \( 2^m \) is the domain of the surjection. By Proposition 2 1.ii), this would imply \( 2^m/2^{Kn^\gamma} < 2^{m+1}/2^{\ell n^\gamma} \), which is a contradiction for \( \ell \geq K + 1 \) and sufficiently big \( n \).

Therefore, Claim 5.1 implies that \( \text{APC}_1 \) proves that “For sufficiently big \( n \) and \( \ell \geq K + 1 \), if \( \pi \) is a \( \text{P} \)-proof of \( \leq_{1/2^{2\ell n^\gamma}}(A, \text{Circuit}[n^k], 1/2 - 1/2^{K_n^\gamma}, 1/2^{K_n^\gamma}) \) for circuits \( A \) of size \( 2^{K_n^\gamma} \), then there is a \( P \)-proof of \( \tt(f, n^k) \) or there is a \( 2^{K_n^\gamma} \)-size circuit \( (1/2 + 1/2^{K_n^\gamma}) \)-approximating \( f \) or there is a \( 2^{|y|/4} \)-size circuit \( (1/2 + 1/2^{|y|/4}) \)-approximating \( y \) or \( |y| \leq n_0 \) or \( |y| \neq S(\cdot, 2^m, 2^{2\ell n^\gamma}) \);” for \( n_0 \) from Definition 5. Since this is a \( \Sigma^b_2 \)-statement, by Lemma 1, \( \text{PV}_1 \) proves the same statement with the existential quantifiers witnessed by \( \text{PV} \)-functions assuming they are given a boolean function \( h' \) which is hard for circuits of size \( 2^{|h'|/4} \), for sufficiently big \( |h'| \).

The last statement provable in \( \text{PV}_1 \) is \( \Pi^b_2 \) so we can translate it to \( \text{EF} \). This gives us \( \text{poly}(|\pi|, 2^n) \)-size circuits \( B_0 \) such that for sufficiently big \( n \), \( \text{EF} \) proves efficiently

“If \( \ell \geq K + 1 \),

\( h' \) is not computable by a particular circuit of size \( 2^{|h'|/4} \), \( |h'| \) is sufficiently big,

\( y \) is not \( (1/2 + 1/2^{|y|/4}) \)-approximable by a particular circuit of size \( 2^{|y|/4} \), \( |y| > n_0 \),

\( |y| = S(\cdot, 2^m, 2^{2\ell n^\gamma}) \)

and \( \pi \) is a \( P \)-proof of \( \leq_{1/2^{2\ell n^\gamma}}(A, \text{Circuit}[n^k], 1/2 - 1/2^{K_n^\gamma}, 1/2^{K_n^\gamma}) \) for \( 2^{K_n^\gamma} \)-size \( A \),

then \( B_0 \) (given \( \pi, h' \) and formula \( \tt(f, n^k) \)) outputs a \( P \)-proof of \( \tt(f, n^k) \)

or \( B_0 \) outputs a \( 2^{K_n^\gamma} \)-size circuit \( (1/2 + 1/2^{K_n^\gamma}) \)-approximating \( f \).”\(^{10}\)

\(^{10}\)Formally, the statement ‘If a particular assignment \( \alpha \) satisfies formula \( \phi \), then formula \( \psi \) holds’ means that ‘If \( \alpha \) is the output of a computation of a specific circuit \( W \) (where \( W \) is allowed to use as inputs atoms from \( \psi \), but other atoms of \( W \) do not appear in \( \psi \)), and \( \alpha \) satisfies \( \phi \), then \( \psi \).’ By Lemma 3, Item 3, if we assume that the statement is efficiently provable in \( P \) and that \( P \) proves efficiently \( \phi \), then \( P \) proves efficiently \( \psi \). Note also that for \( A, B \in \Sigma^b_0 \), the translation \( |A \rightarrow B| \) is \( |\neg A| \rightarrow |B| \), which might not be the same formula as \( |A| \rightarrow |B| \). Nevertheless, \( \text{EF} \) proves efficiently that \( E \rightarrow (|A| \leftrightarrow |\neg A|) \), where \( E \) postulates that auxiliary variables of \( |A| \) encode the computation of
If we now assume that $P$ proves efficiently $\operatorname{tt}(h_n, 2^{n/4}, 1/2 - 1/2^{n/4})$ and that Item 1 holds, then by Definition 6, Items 1-3, for each $k$, there are p-size circuits $B_1$ such that for each sufficiently big $n$, $P$ proves efficiently “$B_1$ (given just formula $\operatorname{tt}(f, n^k)$) outputs a $P$-proof of $\operatorname{tt}(f, n^k)$ or $B_1$ outputs a $2^{K_{n^k}}$-size circuit $(1/2 + 1/2^{K_{n^k}})$-approximating $f$.” (We use here also the fact that $\operatorname{PV}_1$ knows that $S(\cdot, 2^m, 2^{2^{K_{n^k}}})$ depends just on $n$.) Consequently, since $P$ proves efficiently its own reflection, for each sufficiently big $n$, $P$ proves efficiently that “if $\pi$ is a $P$-proof of $\operatorname{tt}(f, 2^{K_{n^k}}, 1/2 - 1/2^{K_{n^k}})$ then $B_1$ outputs a $P$-proof of $\operatorname{tt}(f, n^k)$”.\textsuperscript{11} Finally, we make the $P$-proofs work for all $n$ by increasing the size of $B_1$ by a constant. This finishes the proof of case (1. $\rightarrow$ 2.).

(2. $\rightarrow$ 1.) The opposite implication can be obtained from Lemma 4 and 5 which formalize Theorem 3.

**Lemma 4.** For each $d \geq 2$, each $k \geq 10d$ and each sufficiently big $c$, there is a $\operatorname{PV}$-function $L$ such that for each $\operatorname{PV}$-function $B$ the theory $\operatorname{APC}_1$ proves: Assume the reflection principle for $P$ holds, $\pi$ is a $P$-proof of

$$\operatorname{tt}(h_n \oplus g, 2^{K_{n^k}}, 1/2 - 1/2^{K_{n^k}}) \lor \operatorname{tt}(g, 2^{K_{n^k}}),$$

(5.1)

where $g$ is represented by free variables, and that $B$ automates $P$ on $\Phi$ up to size $|\pi|^c$. Then, for prime $n^d \leq p \leq 2n^d$, where $2n^d \in \log$, for $\delta^{-1} \in \log$ such that $\delta < 1/N^3 = 2^{3n}$, $L(B, n, p)$ is a poly($2^n, |\pi|$)-size circuit learning circuits with $m = n^d$ inputs and size $m^k/10d$, with confidence $1/N^4$, up to error $1/2 - 1/N^3$, where the confidence is counted approximately with error $\delta$ using $\operatorname{PV}$-function $S_z$ and the corresponding assumptions $\operatorname{LB}_{tt}$ expressing hardness of a boolean function $y$, i.e. using formulas $\Pr^y[\cdot]_{\delta}$.

**Lemma 5** (‘XOR trick’). $\operatorname{PV}_1$ proves that for all boolean functions $g, h''$ with $n$ inputs, for sufficiently big $n$, $\operatorname{LB}_{tt}'(h'', 3 \cdot 2^{K_{n^k}}, 2^n(1/2 - 1/2^{K_{n^k}}))$ implies $\operatorname{LB}_{tt}'(h'' \oplus g, 2^{K_{n^k}}, 2^n(1/2 - 1/2^{K_{n^k}})) \lor \operatorname{LB}_{tt}'(g, 2^{K_{n^k}})$, where $\operatorname{LB}_{tt}'$ is obtained from $\operatorname{LB}_{tt}$ by setting $n_0 = 0$ and skipping the universal quantifier on $n$, i.e. all formulas $\operatorname{LB}_{tt}'$ refer to the same $n$.

The proof of Lemma 5 is almost immediate: By $\Sigma_1^b$-induction, a $2^{K_{n^k}}$-size circuit $C_1$ computing $g$ and a $2^{K_{n^k}}$-size circuit $C_2$ $(1/2 + 1/2^{K_{n^k}})$-approximating $h'' \oplus g$ can be combined into a circuit $C_1 \oplus C_2$ of size $3 \cdot 2^{K_{n^k}}$ which $(1/2 + 1/2^{K_{n^k}})$-approximates $h''$.

The implication (2. $\rightarrow$ 1.) can be derived from Lemma 4 and 5 as follows. Since the $\operatorname{APC}_1$-provable statement from Lemma 4 is $\Sigma_1^b$, similarly as above, we can witness it and translate to $\operatorname{EF}$ at the expense of introducing an additional assumption about the hardness

\textsuperscript{11}It is assumed that the encoding of the statement coincides with the encoding of $\operatorname{aut}_P$.
of a boolean function $h'$. That is, for each $p$-size circuit $B$ there are $\text{poly}(|\pi|, 2^{n^d})$-size circuits $A$ and $\text{poly}(|\pi|, 2^{n^d})$-size $\text{EF}$-proofs of

“If the reflection principle for $P$ is satisfied by a particular assignment, $\pi$ is a $P$-proof of (5.1), $h'$ is not computable by a particular circuit of size $2^{|h'|/4}$, $|h'|$ is sufficiently big, $y$ is not $(1/2 + 1/2^{[y]/4})$-approximable by a particular circuit of size $2^{[y]/4}$, $\|y\| > n_0$, $\|y\| = S(\cdot, \cdot, 2^{d-1})$ and $n^d \leq p \leq 2n^d$ is a prime,

then, for $\delta < 1/\gamma^3$, $\text{lear}_p^\delta(L(B, \pi, p), \text{Circuit}(m^{k/10d}), 1/2 - 1/\gamma^3, 1/\gamma^4)$ or $A(B, \pi, h')$ outputs a falsifying assignment of $\text{aut}_P(B, \Phi, |\pi|^e)$.”

Analogously, $\text{PV}_1$-proof from Lemma 5 yields $p$-size $\text{EF}$-proofs of the implication “$\text{tt}(h_n, 3 \cdot 2^{K_n^\gamma}, 1/2 - 1/2^{K_n^\gamma})$ is falsified by a particular assignment or (5.1) holds”. By the assumption of the theorem, there are $p$-size $P$-proofs of $\text{tt}(h_n, 3 \cdot 2^{K_n^\gamma}, 1/2 - 1/2^{K_n^\gamma})$ for sufficiently big $n$. Hence, by Definition 6, Items 1-3, there are $p$-size $P$-proofs of (5.1) for sufficiently big $n$. As $P$ proves efficiently also its own reflection, this yields $\text{poly}(2^{n^d})$-size $P$-proofs of

“If $h'$ is not computable by a particular circuit of size $2^{|h'|/4}$, $|h'|$ is sufficiently big, $y$ is not $(1/2 + 1/2^{[y]/4})$-approximable by a particular circuit of size $2^{[y]/4}$, $\|y\| > n_0$, $\|y\| = S(\cdot, \cdot, 2^{d-1})$ and $n^d \leq p \leq 2n^d$ is a prime,

then, for $\delta < 1/\gamma^3$, $\text{lear}_p^\delta(L(B, \pi, p), \text{Circuit}(m^{k/10d}), 1/2 - 1/\gamma^3, 1/\gamma^4)$ or $A(B, \pi, h')$ outputs a falsifying assignment of $\text{aut}_P(B, \Phi, |\pi|^e)$.”

By Bertrand’s postulate there is a prime $n^d \leq p \leq 2n^d$, so $\text{EF}$ proves that $p$ is a prime by a trivial $2^{O(n^d)}$-size proof which verifies all possible divisors. Therefore, choosing $d > 1/\gamma$, Item 2 and $p$-size $P$-proofs of $\text{tt}(h_n, 2^{n^d}, 1/2 - 1/2^{n^d})$ imply Item 1.

It remains to prove Lemma 4.

Suppose $\pi$ is a $P$-proof of (5.1). Assuming that $B$ automates $P$ on $\Phi$, we want to obtain a $P/\text{poly}$-natural property useful against $\text{Circuit}[n^k]$. To do so, observe (first, without formalizing it in $\text{APC}_1$) that for each $g$, $B$ can be used to find a proof of $\text{tt}(h_n \oplus g, n^k)$ or to recognize that $\text{tt}(g, 2^{K_n^\gamma})$ holds - if $\text{tt}(g, 2^{K_n^\gamma})$ was falsifiable, there would exist a $\text{poly}(|\pi|)$-size $P$-proof of $\text{tt}(h_n \oplus g, 2^{K_n^\gamma}, 1/2 - 1/2^{K_n^\gamma})$ obtained by substituting the falsifying assignment to the proof of (5.1) and thus $B$ would find a short proof of $\text{tt}(h_n \oplus g, n^k)$, for sufficiently big $c$. Since for random $g$, both $h_n \oplus g$ and $g$ are random functions, we know that with probability $\geq 1/2 B$ finds a proof of $\text{tt}(h_n \oplus g, n^k)$ or with probability $\geq 1/2$ it recognizes that $\text{tt}(g, 2^{K_n^\gamma})$ holds. In both cases, $B$ yields a $P/\text{poly}$-natural property useful against $\text{Circuit}[n^k]$.

Let us formalize reasoning from the previous paragraph in $\text{APC}_1$. Let $N = 2^n$ and $B'$ be the algorithm which uses $B$ to search for $P$-proofs of $\text{tt}(h_n \oplus g, n^k)$ or to recognize that
It is easy to construct a surjection $S$ witnessing that $2^N \preceq_0 G_0 \cup G_1$: $S$ maps $g \in G_1$ to $g$ and $g \in G_0$ to $g \oplus h_n$. Following the argument above we conclude that $S$ is a surjection: for each $g$, either $g \in G_1$ (and $S(g) = g$) or $g \oplus h_n \in G_0$ (and $S(g \oplus h_n) = g$). Here, we use the assumption that $\text{APC}_1$ knows that $P$ admits the substitution property and simulates Frege rules. Thus, by Proposition 1 iv), $2^N \preceq_0 2 \cdot 2^N / 3$, which yields a contradiction by Proposition 2 1.ii). Consequently, by Proposition 2 1.i), $G_0 \preceq_\delta 2^N / 3$ or $G_1 \preceq_\delta 2^N / 3$ for $\delta^{-1} \in \text{Log}$. Since $g \in G_0$ and $g \in G_1$ are decidable by p-size circuits and we assume the reflection principle for $P$ (which implies that $G_0$ is useful), this means that either $G_0$ or $G_1$ defines a $P/\text{poly}$-natural property useful against $\text{Circuit}[n^k]$.

Finally, by the $\text{APC}_1$-formalization of [7], Theorem 9, we obtain $\text{poly}(2^n, |\pi|)$-size circuit $L(B, \pi, p)$ learning circuits with $m = n^d$ inputs and size $n^{k/10}$, over the uniform distribution, with membership queries, confidence $1/N^4$, up to error $1/2 - 1/N^3$. \qed

**Corollary 3.** Assume there is a $\text{NE} \cap \text{coNE}$-function $h_n : \{0, 1\}^n \mapsto \{0, 1\}$ such that for each sufficiently big $n$, $h_n$ is not $(1/2 + 1/2^{n/4})$-approximable by $2^n/4$-size circuits. Then there is a proof system $P$ (which can be described explicitly\textsuperscript{12} given the definition of $h_n$) such that for each constant $K$ and $\gamma < 1$, Items 1 and 2 from Theorem 10 are equivalent. Moreover, the equivalence holds for each $\text{APC}_1$-decent system simulating $P$.

**Proof.** We want to construct an $\text{APC}_1$-decent proof system $P$ which proves efficiently $\text{tt}(h_n, 2^{n/4}, 1/2 - 1/2^{n/4})$, for some $h_n$. By the assumption, there is $h_n \in \text{NE} \cap \text{coNE}$, which is hard to approximate. Let $H_\epsilon$, for $\epsilon \in \{0, 1\}$, be the $P$-time predicates defining $h_n$, i.e. $h_n(x) = \epsilon \iff \exists y, |y| \leq 2^{O(|x|)}, H_\epsilon(x, y)$. Define $P$ as WF extended by a rule, which allows to derive every substitutional instance of $||L||$, where $|| \cdot ||$ is the propositional translation from Section 2.2 and $L$ is the formula

\[
[\forall x < 2^{|z||}, \left( (H_0(x, y^0_x) \lor H_1(x, y^1_x)) \land \bigwedge_{\epsilon = 0, 1} (H_\epsilon(x, y^\epsilon_x) \rightarrow z_x = \epsilon) \right) ]
\]

\[
\rightarrow \text{LB}_{\text{tt}}(z, 2^{|z||/4}, 2^{|z||}(1/2 - 1/2^{|z||/4})),
\]

for sufficiently big $n_0$ in $\text{LB}_{\text{tt}}$. By definition, $P$ proves efficiently $\text{tt}(h_n, 2^{n/4}, 1/2 - 1/2^{n/4})$ for each sufficiently big $n$ (we can hardwire the hardness of a boolean function for remaining

\textsuperscript{12}More formally, there is a $p$-time algorithm $R$ such that given predicates $H_0, H_1$ defining $h_n$ (see the proof of Corollary 3), $R$ outputs a $p$-time algorithm defining system $P$.  

30
n, if needed) and satisfies Items 1-3 from the definition of APC₁-decent systems. To see that P proves its own reflection principle we reason in APC₁: given a P-proof π, each circuit in π is derived either by a WF-rule or it is a substitutional instance of ||L||, so by Σ₁b-induction on the length of π and the APC₁-provability of the reflection of WF, cf. [14], L implies that each circuit from π holds. Similarly as in the proof of Theorem 10, we can now translate the resulting Σ₁ theorem of APC₁ to EF and remove the assumptions after moving to P. This shows that P proves efficiently its own reflection and is APC₁-decent.

Corollary 4. Let P, P₀ be APC₁-decent proof systems and assume there is a sequence of boolean functions \( h = \{ h_n \}_{n > n_1} \), for a constant \( n_1 \), such that systems P, P₀ prove efficiently \( \text{tt}(h_n, 2^{n/4}, 1/2 - 1/2^{n/4}) \). Then, for each constant K and constant \( \gamma < 1 \), Item 1 implies Item 2:

1. **P-provable automatibility.** For each \( k \geq 1 \), for each function \( s(n) \geq 2^n \), there is a constant \( K' \) and \( s^{K'} \)-size circuits \( B \) such that P proves efficiently \( \text{aut}_P(B, \Phi, s) \), where \( \Phi \) is the set of pairs \( \langle \text{tt}(f, 2^{Kn'}, 1/2 - 1/2^{Kn'}), \text{tt}(f, n^K) \rangle \) for all boolean functions \( f \) with \( n \) inputs.

2. **\( P_0 \)-provable proof search.** For each \( k \geq 1 \), there is a constant \( K' \) and \( 2^{Kn'} \)-size circuits \( B \) such that \( P_0 \) proves efficiently “\( B \) (given just \( \text{tt}(f, n^K) \)) outputs a P-proof of \( \text{tt}(f, n^K) \) or \( B \) outputs a \( 2^{Kn'} \)-size circuit (\( 1/2 + 1/2^{Kn'} \))-approximating \( f \).”

Proof. Suppose Item 1 holds. By Theorem 10, Item 1 of Theorem 10 holds. Then, following the proof of (1. \( \rightarrow \) 2.) of Theorem 10 with \( P_0 \) instead of P in the last paragraph, we obtain Item 2.

Remark on the collapse. Denote by \( P \vdash \phi_n \) the existence of a p-time algorithm which given \( \phi_n \) generates a P-proof of \( \phi_n \). Corollary 4 exploits the fact (captured by Lemma 3, Item 2) that for PV₁-decent proof systems P (defined analogously as APC₁-decent systems, with APC₁ replaced by PV₁ and WF replaced by EF) there is a p-time algorithm \( B \) such that

\[
\text{EF} \vdash \text{SAT}(x, y) \rightarrow \text{Prf}_P(B(x, y), \text{[SAT}(x, y)\]),
\]

where formula \( \text{SAT}(x, y) \) says that propositional formula (encoded by) \( x \) is satisfied by assignment \( y \), \( \text{Prf}_P(z, x) \) says that \( z \) is a P-proof of \( x \), and \( [\phi] \) is a code of formula \( \phi \). Importantly, while \( y \) stands for free atoms in the assumption \( \text{SAT}(x, y) \), it represents fixed bits (determined by \( y \)) w.r.t. \( P \) in \([\text{SAT}(x, y)\]). That is, there is a p-time algorithm which given 1[\( x \)] generates an EF-proof of \( \text{SAT}(x, y) \rightarrow \text{Prf}_P(B(x, y), \text{[SAT}(x, y)\]).

Using (5.2), it is possible to obtain a collapse similar to Corollary 4 which is essentially unconditional: Assume that PV₁ proves that a p-time algorithm efficiently generates P-proofs of the reflection principle for \( P \). If there is a p-time algorithm \( A \) such that \( P \vdash \neg \text{SAT}(x, A(x)) \lor \text{Prf}_P(A(x), x) \) (in other words, we can efficiently generate P-proofs of

31
being p-bounded and automatable), then there is a p-time algorithm $A'$ such that $\text{EF} \vdash \neg SAT(x, A'(x)) \lor Prf_p(A'(x), x)$. For example, if we can efficiently generate ZFC-proofs of ZFC being p-bounded and automatable, then we can efficiently generate EF-proofs of ZFC being p-bounded and automatable.

Intuitively, the proof proceeds as follows. Assume that, for some p-time algorithm $A$,

$$
\text{ZFC} \vdash \neg SAT(x, A(x)) \lor Prf_{\text{ZFC}}(A(x), x).
$$

Then,

$$
\text{EF} \vdash Prf_{\text{ZFC}}(\pi, [\neg SAT(x, A(x)) \lor Prf_{\text{ZFC}}(A(x), x)]),
$$

for an assignment $\pi$ which is efficiently generable from $1^{|x|}$. Similarly, since $\text{PV}_1$ proves that a p-time algorithm $C$ efficiently generates ZFC-proofs of the reflection principle for ZFC, i.e. $\text{PV}_1$ proves that $C$ given $x$ outputs a ZFC-proof of $Prf_{\text{ZFC}}(z, x) \rightarrow \phi$, where $[\phi] = x$, we have

$$
\text{EF} \vdash Prf_{\text{ZFC}}(C(x), [Prf_{\text{ZFC}}(A(x), x) \rightarrow \phi]),
$$

By (5.2),

$$
\text{EF} \vdash \neg SAT(x, A(x)) \lor Prf_{\text{ZFC}}(B(x, A(x)), [SAT(x, A(x))]).
$$

Therefore, by (5.3)-(5.5), there is a p-time algorithm $B'$ such that

$$
\text{EF} \vdash \neg SAT(x, A(x)) \lor Prf_{\text{ZFC}}(B'(x), x).
$$

Acknowledgements

We would like to thank Jan Krajíček and Iddo Tzameret for comments on a draft of the paper. This project has received funding from the European Union’s Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No 890220.

![European Flag]

References

[1] Alekhnovich M., Braverman M., Feldman V., Klivans A. R., Pitassi T.; Learnability and automatizability; FOCS, 2004.

[2] Applebaum B., Barak B., Xiao D.; On basing lower bounds for learning on worst-case assumptions; FOCS, 2008.
[3] Atserias A., Müller M; *Automating Resolution is NP-hard*; FOCS, 2019.

[4] Bonet M. L., Domingo C., Gavaldá R., Maciel A., Pitassi T.; *Non-automatizability of bounded-depth Frege proofs*; Computational Complexity, 13(1-2):47-68, 2004.

[5] Bonet M. L., Pitassi T., Raz R.; *On interpolation and automatization for Frege proof systems*; SIAM Journal of Computing, 29(6):1939-1967, 2000.

[6] Buss S.; *Bounded arithmetic*; Bibliopolis, 1986.

[7] Carmosino M., Impagliazzo R., Kabanets V., Kolokolova A.; *Learning algorithms from natural proofs*; CCC, 2016.

[8] Cobham A.; *The intrinsic computational difficulty of functions*, Proceedings of the 2nd International Congress of Logic, Methodology and Philosophy of Science, North Holland, pp. 24-30, 1965.

[9] Cook S.A.; *Feasibly constructive proofs and the propositional calculus*, Proceedings of the 7th Annual ACM Symposium on Theory of Computing, ACM Press, pp. 83-97, 1975.

[10] Cook S.A., Thapen N.; *The strength of replacement in weak arithmetic*, ACM Transactions on Computational Logic, 7(4):749-764, 2006.

[11] de Rezende S., Göös M., Nördstrom J., Pitassi T., Robere R., Sokolov D.; *Automating algebraic proof systems is NP-hard*; CCC, 2020.

[12] Garlík M.; *Failure of feasible disjunction property for k-DNF Resolution and NP-hardness of automating it*; ECCC, 2020.

[13] Göös M., Koroth S., Mertz I., Pitassi T.; *Automating Cutting Planes is NP-hard*; STOC, 2020.

[14] Jeřábek E.; *Dual weak pigeonhole principle, Boolean complexity and derandomization*; Annals of Pure and Applied Logic, 129:1-37, 2004.

[15] Jeřábek E.; *Weak pigeonhole principle and randomized computation*; Ph.D. thesis, Charles University in Prague, 2005.

[16] Jeřábek E.; *Approximate counting in bounded arithmetic*; Journal of Symbolic Logic, 72:959-993, 2007.

[17] Krajíček J.; *Bounded arithmetic, propositional logic, and complexity theory*; Cambridge University Press, 1995.
[18] Krajíček J.; *Interpolation theorems, lower bounds for proof systems, and independence results for bounded arithmetic*; Journal of Symbolic Logic, 66(2):457-486, 1997.

[19] Krajíček J.; *On the weak pigeonhole principle*; Fundamenta Mathematicae, 170(1-3):123-140, 2001.

[20] Krajíček J.; *Forcing with random variables and proof complexity*; Cambridge University Press, 2011.

[21] Krajíček J.; *Proof complexity*; Cambridge University Press, 2019.

[22] Krajíček J., Pudlák P.; *Some consequences of cryptographical conjectures for $S^1_2$ and $EF$*; Information and Computation, 140(1):82-94, 1998.

[23] Krajíček J., Pudlák P., Takeuti G.; *Bounded arithmetic and the polynomial hierarchy*; Annals of Pure and Applied Logic, 52:143-153, 1991.

[24] Müller M., Pich J.; *Feasibly constructive proofs of succinct weak circuit lower bounds*; Annals of Pure and Applied Logic, 2019.

[25] Nisan N., Wigderson A.; *Hardness vs. randomness*; J. Comp. Systems Sci., 49:149-167, 1994.

[26] Oliveira I.C., Santhanam R.; *Conspiracies between learning algorithms, circuit lower bounds, and pseudorandomness*; CCC, 2017.

[27] Paris J., Wilkie A., Woods A.; *Provability of the pigeonhole principle and the existence of infinitely many primes*; Journal of Symbolic Logic, 53(4):1235-1244, 1988.

[28] Pich J.; *Learning algorithms from circuit lower bounds*; preprint, 2020.

[29] Pitassi T., Santhanam R.; *Effectively polynomial simulations*; ICS, 2010.

[30] Razborov A.A; *On provably disjoint $NP$ pairs*, BRICS, 1994.

[31] Razborov A.A.; *Bounded arithmetic and lower bounds in boolean complexity*; Feasible Mathematics II, 344-386, 1995.

[32] Razborov A.A; *Unprovability of lower bounds on the circuit size in certain fragments of bounded arithmetic*, Izvestiya of the Russian Academy of Science, 59:201-224, 1995.

[33] Razborov A.A.; *Pseudorandom generators hard for $k$-DNF Resolution and Polynomial Calculus*; Annals of Mathematics, 181(2):415-472, 2015.

[34] Razborov A.A, Rudich S.; *Natural Proofs*; Journal of Computer and System Sciences, 55(1):24-35, 1997.
[35] Santhanam R.; *Pseudorandomness and the Minimum Circuit Size Problem*; ITCS, 2020.

[36] Valiant L.; *A theory of the learnable*; Communications of the ACM, 27, 1984.

[37] Williams R.; *Non-uniform ACC circuit lower bounds*; CCC, 2011.