Pseudo proof systems and hard instances of SAT[*]

Jan Maly[†]
Institute of Information Systems
Technische Universität Wien
jmaly@dbai.tuwien.ac.at

Moritz Müller[‡]
Kurt Gödel Research Center
University of Vienna
moritz.mueller@univie.ac.at

Abstract

We link two concepts from the literature, namely hard sequences for the satisfiability problem SAT and so-called pseudo proof systems proposed for study by Krajíček. Pseudo proof systems are elements of a particular nonstandard model constructed by forcing with random variables. Speaking in standard terms, pseudo proof systems are propositional proof systems that are unsound in the sense that they might prove some falsifiable propositional formulas but falsifying assignments are in a certain sense hard to find. A hard sequence for a SAT-algorithm is, roughly, a feasibly computable sequence of satisfiable propositional formulas such that the algorithm needs superpolynomial time on them. We show that the existence of pseudo proof systems with a certain property that we call madness is equivalent to the existence of so-called probably hard sequences computable by certain randomized polynomial time algorithms. We discuss variants of hard sequences and survey the relevant literature.

1 Introduction

Pseudo proof systems It is a basic question of mathematical logic, unsettled to date, whether there exists a (propositional) proof system that has *short* proofs for all (propositional) tautologies. More formally, recall that abstractly a *propositional proof system* is a polynomial time function from the set of binary strings $\{0,1\}^*$ into the set TAUT of (binary strings coding) propositional tautologies. Often [19] it is additionally required, that the function is not only into TAUT, meaning *soundness*, but also onto, meaning *completeness*. Having short proofs means that the system is *polynomially bounded*: every tautology has a proof, i.e., preimage, of length polynomial in its length. Such proof systems exist if and

[*]This paper reports results of the first author's Master's Thesis [42].

[†]Supported by the Austrian Science Fund (FWF) under project number P25207 and Y698.

[‡]Supported by the Austrian Science Fund (FWF) under project number P28699.

only if NP = coNP [19]. Requiring additionally that short proofs can be computed in polynomial time characterizes P = NP. It is not known whether the usual textbook systems, called *Frege* in this context, are polynomially bounded (see [10, 11] for discussions).

As is well known, "one can think of length-of-proofs lower bounds as about problems of how to construct suitable models of particular bounded arithmetic" [36, p.175]. A general method to construct such models is developed in [36] following Scott's [53] forcing with random variables. In this context Krajíček suggests the study of so-called pseudo proof systems. These can be characterized in standard terms, i.e., without any mention of nonstandard models, and we include such a standard formulation in the statement of our main result (Theorem 2 below). Roughly, a pseudo proof system may have proofs of falsifiable formulas but these are hard to detect in that polynomial time algorithms succeed to witness falsifiability only for a vanishing fraction of proofs of a given length. We refer to Propositions 25 and 26 for precise statements. The model-theoretic context, however, is of independent interest.[1]

Namely, an important instance of the method in [36] is the Boolean valued model $K(F^n_{\mathrm{PV}})$. Its universe is given by the set of all polynomial time functions on binary strings of some fixed nonstandard length $n \in M$, where M is some fixed large nonstandard model of true arithmetic. The Boolean valuation considers two such functions equal if they differ only on an infinitesimal fraction of input strings. The model interprets the language having symbols for all polynomial time functions and relations, and it turns out that in $K(F^n_{\mathrm{PV}})$ all true universal statements in this language are valid. Intuitively, one can say that $K(F^n_{\mathrm{PV}})$ models a significant amount of feasible reasoning. In particular and more precisely, all $\forall\Sigma^b_1$-consequences of Buss' S^1_2 are valid in $K(F^n_{\mathrm{PV}})$ (see e.g. [9]). This together with its appealing and familiar definition makes $K(F^n_{\mathrm{PV}})$ an object of interest. We shall mention some related constructions (cf. Remark 21) once we gave the precise definition in Section 3.2.

The objects of $K(F^n_{\mathrm{PV}})$ "can be viewed from two different perspectives" [36, p.160], namely, first as elements of the universe of $K(F^n_{\mathrm{PV}})$ and second as functions defined on binary strings $\{0,1\}^n$. For example, viewed as an element of $K(F^n_{\mathrm{PV}})$ a propositional proof system is a tautology in the sense of $K(F^n_{\mathrm{PV}})$. Conversely, a tautology in the sense of $K(F^n_{\mathrm{PV}})$ is a *pseudo proof system* (cf. Definition 22). Viewed as a function on binary strings a pseudo proof system may be unsound. In fact, it is conceivable that *mad* pseudo proof systems exist (cf. Definition 27). Viewed as elements of $K(F^n_{\mathrm{PV}})$ these are tautologies in the sense of $K(F^n_{\mathrm{PV}})$ but viewed as functions on binary strings they *never* output a tautology.

In [36, Section 24.4] Krajíček asks for transfer principles concerning pseudo proof systems. Loosely speaking a *transfer principle* is a statement that allows to infer properties of standard objects from properties of nonstandard objects, and vice-versa. Our main result (Theorem 2) is such a transfer principle that links the existence of mad pseudo proof systems to a hypothesis concerning the computational complexity of the satisfiability problem SAT of independent interest, explained next.

[1] All relevant model-theoretic concepts will be explained, no knowledge of [36] is assumed.

Hard sequences For an algorithm solving a hard computational task there exist instances of the problem witnessing that the algorithm is not feasible. For example, P $\neq$ NP if and only if every SAT-algorithm has a hard sequence:

Definition 1. Let $Q \subseteq \{0,1\}^*$ and $\mathbb{A}$ be a *Q-algorithm*, i.e., an algorithm deciding Q, and let p be a polynomial. A sequence $(x_n)_{n\in\mathbb{N}}$ is *p-hard for* $\mathbb{A}$ if for infinitely many $n \in \mathbb{N}$:

(H1) $x_n \in Q$,

(H2) $t_{\mathbb{A}}(x_n) > p(|x_n|, n)$.

Here, $t_{\mathbb{A}}(x)$ denotes the running time of $\mathbb{A}$ on input x. Being *hard for* $\mathbb{A}$ means being p-hard for $\mathbb{A}$ for all polynomials p.

It is a natural question to ask whether such a sequence could be computable in polynomial time. Here, we say that a sequence $(x_n)_{n\in\mathbb{N}}$ of binary strings $x_n \in \{0,1\}^*$ is polynomial time computable if so is the function that computes x_n from $1^n = 1 \cdots 1$ (n times).[2]

Hard Sequence Hypothesis *For every* SAT*-algorithm* $\mathbb{A}$ *there exists a polynomial time computable sequence which is hard for* $\mathbb{A}$.

We are not aware of a place where this hypothesis has been formulated explicitly, but it is certainly implicit in many papers. We are also not aware of any well-established computational hardness hypothesis that would imply this hypothesis.

Hard sequences have been studied from at least two perspectives. The first is *speed-up*, going back at least to [54]: if $\mathbb{A}$ has a hard sequence $(x_n)_n$ with the additional property to be *in Q* (i.e., $x_n \in Q$ for all $n \in \mathbb{N}$), then there is an algorithm $\mathbb{B}$ that runs in polynomial time on $\{x_n \mid n \in \mathbb{N}\}$ (see e.g. [17, Lemma 4.1]), so is superpolynomially faster than $\mathbb{A}$.

The second perspective, more relevant to this paper, is to *witness failure* of feasible algorithms: equip a given SAT-algorithm $\mathbb{A}$ with a polynomial "clock" p, that is, stop its computation on an input x after $p(|x|)$ steps and reject if it did not halt. This yields an algorithm $\mathbb{A}^p$ without false positive answers, and a p-hard sequence contains infinitely many false negative answers of $\mathbb{A}^p$. Witnessing failure has been studied not only for deterministic algorithms but also for randomized [20, 55, 6] and non-uniform algorithms [41, 8, 2]. More recent discussions of witnessing failure include [48, 12] in the contexts of independence of S^1_2 and natural proofs, respectively.

The Hard Sequence Hypothesis has many natural variants. One can ask to produce hard sequences for larger or smaller sets of algorithms, or allow more or less computational power for their construction. Here we focus on weaker variants. The following section is intended both to discuss some of these variants and to survey the corresponding literature.

A natural (see e.g. [20]) first weakening of the Hard Sequence Hypothesis is by allowing randomness in the construction of hard sequences. One then asks for polynomial time *samplable* (as opposed to computable) *probably hard* sequences (cf. Definitions 7, 6).

[2]Note that (H2) in the definition of a polynomial time computable hard sequence is equivalent to the statement that $t_{\mathbb{A}}(x_n)$ is not $n^{O(1)}$ (matching the definition in [17]).

We observe that SAT-algorithms do have such sequences under cryptographic assumptions (cf. Proposition 9).

Second, a natural subclass of SAT-algorithms are SAT-*solvers*: SAT-algorithms which upon accepting a satisfiable input formula F also output a binary string of length $\leqslant |F|$ that satisfies F. We say that a binary string $x = x_1 \cdots x_n \in \{0,1\}^n$ satisfies F if so does the truth assignment that maps the i-th variable of F to x_i if $i \leqslant n$ and to 0 otherwise. We observe that SAT-*solvers* do have polynomial time computable hard sequences under a well-established hypothesis (cf. Proposition 13).

Transfer principle Our transfer principle links the existence of mad pseudo proof systems with the existence of probably hard sequences that are samplable with a quite restrictive use of randomness that we call *invertibility* (cf. Definition 15). Intuitively, the sampler is required to witness its outputs by publishing the random seed used.

Theorem 2. *The following are equivalent:*

1. *There is a nonstandard $n \in M$ such that $K(F^n_{\mathrm{PV}})$ contains mad pseudo proof systems.*
2. *Every* SAT-*solver has an invertibly samplable probably hard sequence.*
3. *There is a polynomial time computable function f such that for all $\epsilon > 0$ and all polynomial time computable functions g there are infinitely many $n \in \mathbb{N}$ such that*

   (i) for all $x \in \{0,1\}^n$: $f(x)$ is a falsifiable propositional formula;

   (ii) for at most an ϵ-fraction of $x \in \{0,1\}^n$: $g(x)$ is a falsifying assignment of $f(x)$.

We bias our discussion of probably hard sequences in Section 2 and of pseudo proof systems in Section 3 in such a way that Theorem 2 will easily follow by putting together all observations collected. The property in statement (3) could be taken as a standard definition of a mad pseudo proof system. We refer to Section 4.1 for a discussion.

We further show that the statements of the above theorem hold true under well-established complexity theoretic hypotheses:[3]

Corollary 3. *Assume pseudo-random generators exist and* $\mathrm{NE} \cap \mathrm{coNE} \not\subseteq \mathrm{E}$. *Then there is a nonstandard $n \in M$ such that $K(F^n_{\mathrm{PV}})$ contains mad pseudo proof systems.*

It is well-known that pseudo-random generators exist under some well-established hypothesis (see e.g. the standard textbook [1, Chapter 20]).

2 Hard sequences

2.1 Hard sequences for TAUT-algorithms

For TAUT the question whether hard sequences can be computed in polynomial time has gained a lot of attention, especially for sequences $(F_n)_n$ that are *in* TAUT (cf. page 3):

[3]E and NE denote deterministic and nondeterministic simply exponential time $2^{O(n)}$, respectively.

Theorem 4 (Krajíček and Pudlák [38]). *The following are equivalent.*

1. TAUT *does not have an* almost optimal *algorithm, i.e., an algorithm* $\mathbb{A}$ *deciding* TAUT *such that for every algorithm* $\mathbb{B}$ *deciding* TAUT *there is a polynomial* p *such that* $t_{\mathbb{A}}(F) \leqslant p(t_{\mathbb{B}}(F) + |F|)$ *for all formulas* $F \in$ TAUT.
2. *There does not exist a* p-optimal *propositional proof system, i.e., a propositional proof system* P^* *such that for all propositional proof systems* P *there exists a polynomial time computable function* t *such that* $P = P^* \circ t$.
3. *For every* TAUT*-algorithm* $\mathbb{A}$ *there is a polynomial time computable sequence in* TAUT *which is hard for* $\mathbb{A}$.
4. *For every propositional proof system* P *there exists a polynomial time computable sequence* $(F_n)_{n\in\mathbb{N}}$ *in* TAUT *such that for every polynomial time computable sequence* $(\pi_n)_{n\in\mathbb{N}}$ *there are infinitely many* n *such that* $P(\pi_n) \neq F_n$.

We attribute this theorem to [38] where the crucial equivalence of (1) and (2) is proved. See [24] for a survey of variants for nondeterministic, nonuniform and heuristic algorithms. The equivalences to the hard sequences in (3) and (4) follow from well-known facts about propositional reflection principles (see e.g. [34, Chapter 14], or [5] for a survey). More direct and more general arguments can be found in [17, Theorems 4.3(a), 6.7(a)].

Since polynomial time algorithms are trivially almost optimal, the hypothesis that one of the statements of Theorem 4 is true implies P $\neq$ NP. In fact, it implies E $\neq$ NE [38] and EE $\neq$ NEE [32] and more [3]. We mention that the hypothesis and its variant for nondeterministic algorithms are deeply rooted in mathematical logic, linked to whether "one could realize the Hilbert program in a modified, finitistic sense" [38, p.1067], to finite model theory [14], to the complexity of Gödel's proof predicate [13], and to consistency statements [16].

2.2 Hard sequences for SAT-algorithms

One might wonder what is special about TAUT or for which other problems Q do the equivalences in Theorem 4 hold true. The notion of a propositional proof system is straightforwardly generalized: a *(complete) proof system for* Q is a polynomial time computable surjection onto Q. *Almost optimality* and *p-optimality* are explained as in Theorem 4 (1), (2).

Trivially, ($1 \Leftrightarrow 2$) holds true for any Q polynomially isomorphic to TAUT, and thus for all coNP-complete Q if one assumes the Berman-Hartmanis conjecture [4]. As shown in [17] this conjecture can be avoided. Sadowski [52] proved ($1 \Leftrightarrow 2$) for SAT and Messner [46] for all paddable Q. But, as shown in [17, Theorem 7.10], ($1 \Leftrightarrow 3$) breaks for some Q if one assumes the Measure Hypothesis (see e.g. [43]). It is unknown for SAT. More precisely, consider the somewhat stronger version of the Hard Sequence Hypothesis that asks for sequences *in* SAT (cf. page 3). This version implies but is not known to be equivalent to the hypothesis that SAT does not have almost optimal algorithms.

Indeed, SAT and TAUT behave quite differently in our context. First of all, there is an obvious polynomially bounded proof system for SAT. It is, however, unlikely to be p-optimal [33]. Further, already in 1973 Levin [40] found an optimal SAT-solver (see Theorem 10). It is, however, unlikely to be almost optimal among general SAT-algorithms. This is proved in [15, Proposition 4.5], a survey of some recent applications of Levin's result.

On the positive side, Gutfreund, Shaltiel and Ta-Shma showed:

Theorem 5 ([20]). *If* $\mathrm{NP} \nsubseteq \mathrm{P}$*, then for every polynomial* p *and every* SAT*-algorithm* $\mathbb{A}$ *there is a polynomial time computable sequence which is* p*-hard for* $\mathbb{A}$*.*

A diagonalizing argument shows that one can compute hard sequences in slightly superpolynomial time (see [20, Theorem 1.6] for such a construction) but the construction of polynomial time computable hard sequences remains open.

Does randomness help? As for a notion of feasibility for sequences of random strings[4] we borrow the following from average case complexity [7]:

Definition 6. A sequence of random strings $(X_n)_{n\in\mathbb{N}}$ is *(polynomial time) samplable* if there exists a polynomial time computable *sampler for* it, that is, a function $D : \{0,1\}^* \to \{0,1\}^*$ such that $D \circ U_n$ has the same distribution as X_n for all $n \in \mathbb{N}$. Here, U_n denotes a random variable uniformly distributed in $\{0,1\}^n$.

The following definition is convenient. With suitable adjustments, it makes sense for *randomized* SAT-algorithms, and has been implicitly studied in [20, 55]. Here, we restrict attention to deterministic algorithms.

Definition 7. Let $\mathbb{A}$ be a Q-algorithm, p a polynomial and $\delta, \epsilon \geqslant 0$. A sequence $(X_n)_{n\in\mathbb{N}}$ of random strings is (δ,ϵ)*-probably* p*-hard for* $\mathbb{A}$ if for infinitely many $n \in \mathbb{N}$:

(P1) $\Pr\big(X_n \in Q\big) \geqslant 1 - \delta$,

(P2) $\Pr\big(t_{\mathbb{A}}(X_n) > p(|X_n|, n)\big) \geqslant 1 - \epsilon$.

The sequence is (δ,ϵ)*-probably hard for* $\mathbb{A}$ if for all polynomials p it is (δ,ϵ)-probably p-hard for $\mathbb{A}$. And we call it *probably hard for* $\mathbb{A}$ if for all $\epsilon > 0$ it is $(0,\epsilon)$-probably hard for $\mathbb{A}$.

Note that a hard sequence is $(0,0)$-probably hard, and conversely, any sequence of realizations of a $(0,0)$-probably hard sequence is hard.

We now show that, using randomness, (superpolynomial) hardness is achievable under cryptographic assumptions.[5] The proof below could be carried out with weak versions of such assumptions (e.g. "i.o." versions [25]) but we prefer to stick with the standard definition (see e.g. [22]):

[4]A random string is a random variable with values in $\{0,1\}^*$. Given any random variable we always use Pr to denote the probability measure of the underlying probability space.

[5]Such assumptions are prohibitive in the context of [20, 2, 6] who are concerned with the problem to reduce average-case hardness hypotheses to worst-case hardness hypotheses.

Definition 8. A *cryptographic pseudo-random generator with stretch* $2n$ is a polynomial time computable function $G : \{0,1\}^* \to \{0,1\}^*$ such that $|G(r)| = 2|r|$ for all $r \in \{0,1\}^*$ and for all positive polynomials p and all randomized polynomial time algorithms $\mathbb{A}$ we have for all sufficiently large n:

$$\big|\Pr(\mathbb{A} \text{ accepts } G(U_n)) - \Pr(\mathbb{A} \text{ accepts } U_{2n})\big| \leqslant 1/p(n). \tag{1}$$

Proposition 9. *Assume cryptographic pseudo-random generators with stretch* $2n$ *exist. Then there is a samplable sequence which is probably hard for every* SAT*-algorithm.*

Proof. Let G be a generator as assumed to exist. Clearly, its image

$$Q := \{G(r) \mid r \in \{0,1\}^*\}$$

is in NP, so there is a polynomial time reduction f from Q to SAT. Define

$$D(r) := f(G(r)),$$

and note $\Pr(D(U_n) \in \text{SAT}) = 1$ for all $n \in \mathbb{N}$. Assume for the sake of contradiction, that $(D(U_n))_n$ is not probably hard for some SAT-algorithm $\mathbb{B}$. Then there are a polynomial p and $\epsilon > 0$ such that $\Pr(t_{\mathbb{B}}(D(U_n)) \leqslant p(n)) \geqslant \epsilon$ for infinitely many *good* n.

Let $\mathbb{A}$ accept an input r if and only if $\mathbb{B}$ accepts $f(r)$ in at most $p(|r|)$ steps. Then $\Pr(\mathbb{A}$ accepts $G(U_n)) \geqslant \epsilon$ for all good n. But the event that $\mathbb{A}$ accepts U_{2n} implies the event that $\mathbb{B}$ accepts $f(U_{2n})$, hence $f(U_{2n}) \in \text{SAT}$, hence $U_{2n} \in Q$. The latter event has probability $\leqslant 2^n/2^{2n} = 2^{-n}$. Thus, for all large enough good n the difference of the probabilities in (1) is at least $\epsilon - 2^{-n} \geqslant \epsilon/2$, a contradiction. □

In [18] a similar sampler has been studied with the aim to speed-up randomized SAT-algorithms in a certain weak sense.

2.3 Hard sequences for SAT-solvers

A natural weakening of the Hard Sequence Hypothesis is to ask for hard or probably hard sequences not for general SAT-algorithms but only for SAT-solvers. Note that it follows from the self-reducibility of SAT, that P $\neq$ NP if and only if every SAT-solver has a hard sequence. Krajíček constructs for every SAT-solver $\mathbb{A}$ and every polynomial p a polynomial time computable sequence p-hard for $\mathbb{A}$. His construction works under the assumption of lower bounds for a propositional proof system proving the correctness of $\mathbb{A}$ (see [37] for a precise statement). Following [20], Bogdanov et al. [6] construct p-hard sequences for SAT-solvers under the assumption that NP $\not\subseteq$ P. For randomized SAT-solvers they obtain $(0.1, 0.34)$-probably p-hard sequences under the assumption that NP $\not\subseteq$ BPP. More importantly, Bogdanov et al. construct formulas together with their satisfying assignments (we refer to [6] for a precise statement). It is clear that these so-called *dreambreakers* cannot be (superpolynomially) hard for Levin's optimal SAT-solver $\mathbb{L}$:

Theorem 10 (Levin [40])**.** *There exists a* SAT*-solver* $\mathbb{L}$ *such that for every* SAT*-solver* $\mathbb{A}$ *there exists a polynomial* $p_{\mathbb{A}}$ *such that* $t_{\mathbb{L}}(F) \leqslant p_{\mathbb{A}}(t_{\mathbb{A}}(F) + |F|)$ *for every* $F \in$ SAT.

We shall use the following easy consequence mainly with $\delta = 0$ by referring to "the optimality of $\mathbb{L}$". Note the lemma applies to (deterministic) hard sequences because these are $(0, 0)$-probably hard sequences.

Lemma 11. *Let* $(X_n)_{n\in\mathbb{N}}$ *be a sequence of random strings and* $\epsilon \geqslant \delta \geqslant 0$*. If* $(X_n)_{n\in\mathbb{N}}$ *is* $(\delta, \epsilon - \delta)$*-probably hard for* $\mathbb{L}$*, then it is* (δ, ϵ)*-probably hard for every* SAT*-solver.*

Proof. Assume $(X_n)_{n\in\mathbb{N}}$ is $(\delta, \epsilon - \delta)$-probably hard for $\mathbb{L}$. Let $\mathbb{A}$ be a SAT-solver and p a polynomial such that for almost all $n \in \mathbb{N}$:

$$\Pr(X_n \in \text{SAT}) \geqslant 1 - \delta \implies \Pr(t_{\mathbb{A}}(X_n) \leqslant p(|X_n|, n)) \geqslant \epsilon. \tag{2}$$

Choose a nondecreasing polynomial $p_{\mathbb{A}}$ for $\mathbb{A}$ according to Theorem 10. Then (2) implies

$$\Pr(X_n \in \text{SAT}) \geqslant 1 - \delta \quad\implies\quad \Pr(t_{\mathbb{L}}(X_n) \leqslant p_{\mathbb{A}}(p(|X_n|, n)) \text{ or } X_n \notin \text{SAT}) \geqslant \epsilon,$$

and thus

$$\Pr(X_n \in \text{SAT}) \geqslant 1 - \delta \quad\implies\quad \Pr(t_{\mathbb{L}}(X_n) \leqslant p_{\mathbb{A}}(p(|X_n|, n))) \geqslant \epsilon - \delta.$$

Hence, $(X_n)_{n\in\mathbb{N}}$ is not $(\delta, \epsilon - \delta)$-probably hard for $\mathbb{L}$ □

In particular:

Proposition 12. *The following are equivalent.*

(1) There exists a samplable sequence which is probably hard for $\mathbb{L}$*.*

(2) There exists a samplable sequence which is probably hard for all SAT*-solvers.*

(3) For every SAT*-solver* $\mathbb{A}$ *there exists a samplable sequence which is probably hard for* $\mathbb{A}$*.*

Cryptographic assumptions are not needed to infer the existence of such sequences. Indeed, the following is essentially known.

Proposition 13. *The following statements are equivalent, and implied by* $\text{NE} \cap \text{coNE} \not\subseteq \text{E}$.

(1) There exists a polynomial time computable sequence which is hard for $\mathbb{L}$*.*

(2) There exists a polynomial time computable sequence which is hard for all SAT*-solvers.*

(3) For every SAT*-solver* $\mathbb{A}$ *there exists a polynomial time computable sequence which is hard for* $\mathbb{A}$*.*

(4) For every SAT*-solver* $\mathbb{A}$ *there exists an injective polynomial time computable sequence* $(F_n)_{n\in\mathbb{N}}$ *which is hard for* $\mathbb{A}$ *and such that* $|F_n| \geqslant n$ *for all* $n \in \mathbb{N}$*.*

Proof. ($1 \Rightarrow 2$) follows from the optimality of $\mathbb{L}$ (Lemma 11). ($2 \Rightarrow 3$) and ($4 \Rightarrow 1$) are trivial. To prove ($3 \Rightarrow 4$) we proceed as in [17, Proposition 3.2] using a padding function: a polynomial time computable function *pad* that maps a formula F and a string $y \in \{0,1\}^*$ to a formula $pad(F,y)$ of length at least $|F|+|y|$ that has the same satisfying assignments as F, and such that there are two polynomial time functions mapping any input of the form $pad(F,y)$ to F and y, respectively.

Let $\mathbb{A}$ be a SAT-solver and assume (3). Define an algorithm $\mathbb{B}$ as follows: given a formula F, for $t = 0,1,\ldots$ compute t steps of $\mathbb{A}$ on each of $pad(F,1^0),\ldots,pad(F,1^t)$; as soon as one of these computations halts, return the answer obtained.

Clearly, $\mathbb{B}$ is a SAT-solver and there is a polynomial p such that for every $t \in \mathbb{N}$ and every formula F we have $t_{\mathbb{B}}(F) \leqslant p(t + t_{\mathbb{A}}(pad(F,1^t)))$. By (3) there is a polynomial time computable sequence $(F_n)_n$ hard for $\mathbb{B}$. Then $(pad(F_n,1^n))_n$ is polynomial time computable and hard for $\mathbb{A}$. This sequence is injective and satisfies $|pad(F_n,1^n)| \geqslant n$ for all $n \in \mathbb{N}$.

We have proved that (1)-(4) are equivalent. We now derive (2) assuming there exists a problem $Q \in \mathrm{NE} \cap \mathrm{coNE} \setminus \mathrm{E}$. For a binary string x let $\mathrm{num}(x)$ be the natural number with binary expansion $1x$. Then

$$Q' := \{1^{\mathrm{num}(x)} \mid x \in Q\} \in \mathrm{NP} \cap \mathrm{coNP} \setminus \mathrm{P}.$$

We now proceed as in [15, Proposition 4.5]. By the NP-completeness of SAT, there are polynomial time reductions r_1 and r_0 from Q' and $\{0,1\}^* \setminus Q'$ to SAT. We can assume that $r_1(1^n)$ and $r_0(1^n)$ are propositional formulas. Then $r_1(1^n) \vee r_0(1^n) \in$ SAT, and a satisfying assignment satisfies exactly one of $r_1(1^n)$ and $r_0(1^n)$, namely $r_1(1^n)$ if $1^n \in Q'$, and $r_0(1^n)$ if $1^n \notin Q'$. Since $Q' \notin \mathrm{P}$, there is no SAT-solver $\mathbb{A}$ such that $t_{\mathbb{A}}(r_1(1^n) \vee r_0(1^n)) \leqslant n^{O(1)}$. □

The reader might wonder whether the self-reducibility of SAT allows to transform a somehow hard sequence for SAT-solvers to a somehow hard sequence for SAT-algorithms. We do not know how to preserve superpolynomial hardness in such a transformation. To point out the difficulty we include the following argument, similar to arguments in [20].

Proposition 14. *Let $\epsilon \geqslant \delta \geqslant 0$. Assume there exists a samplable $(\delta, \epsilon-\delta)$-probably hard sequence for $\mathbb{L}$. Then for every* SAT*-algorithm $\mathbb{A}$ and every polynomial p there exists a samplable sequence which is $(1/2+\delta/2, \epsilon)$-probably p-hard for $\mathbb{A}$.*

Proof. Let D' be a polynomial time sampler for a sequence which is $(\delta, \epsilon-\delta)$-probably hard for $\mathbb{L}$. Let a SAT-algorithm $\mathbb{A}$ and a polynomial p be given.

For a formula F with at least n variables and a string $x = x_1 \cdots x_n \in \{0,1\}^n$ let F^x be obtained from F by substituting for each $1 \leqslant i \leqslant n$ the Boolean value x_i for the i-th variable in F. Define a SAT-solver $\mathbb{B}$ as follows. On an input formula F, initialize $x \leftarrow \lambda$ to the empty string λ; while F^x has a free variable, run $\mathbb{A}$ in parallel on F^{x0} and F^{x1}; let $b \in \{0,1\}$ be such that $\mathbb{A}$ on F^{xb} halted first; if $\mathbb{A}$ accepted, then update $x \leftarrow xb$; else update $x \leftarrow xb'$ for $b' := 1-b$. Upon leaving the while loop, accept with output x if F^x is true (this is a Boolean formula without variables); else reject.

Observe that there is a polynomial q such that for all formulas F and all $n \in \mathbb{N}$, $t_{\mathbb{B}}(F) \leqslant q(|F|, n)$ or $\mathbb{B}$ enters the while loop for a value of x that is *n-critical for F* in that both $t_{\mathbb{A}}(F^{x0}) > p(|F^{x0}|, n)$ and $t_{\mathbb{A}}(F^{x1}) > p(|F^{x1}|, n)$. Further note that, if $F \in \text{SAT}$, then at least one of these two formulas is satisfiable.

We define a sampler D: given λ, output some fixed satisfiable formula F_0; given rb with $r \in \{0,1\}^n$ and $b \in \{0,1\}$, run $\mathbb{B}$ on $D'(r)$ until the while loop is called for an n-critical x; then output F^{xb}; if this never happens, output F_0.

By Lemma 11, $(D' \circ U_n)_n$ is (δ, ϵ)-probably hard for $\mathbb{B}$, so $\Pr(D'(U_n) \in \text{SAT}) \geqslant 1 - \delta$ and $\Pr\big(t_{\mathbb{B}}(D'(U_n)) > q(|D'(U_n)|, n)\big) \geqslant 1 - \epsilon$ for infinitely many *good* n.

Suppose n is good. For every $r \in \{0,1\}^n$ such that $F := D'(r)$ is satisfiable, $D(rb)$ is unsatisfiable only if it equals F^{xb} for some x with $F^{x(1-b)}$ satisfiable. Hence we have $\Pr(D(U_{n+1}) \in \text{SAT} \mid D'(U_n) \in \text{SAT}) \geqslant 1/2$, and thus

$$\Pr\big(D(U_{n+1}) \in \text{SAT}\big) \geqslant 1/2 \cdot \Pr\big(D'(U_n) \in \text{SAT}\big) \geqslant 1/2 - \delta/2.$$

For every $r \in \{0,1\}^n$ with $t_{\mathbb{B}}(D'(r)) > q(|D'(r)|, n)$ we have $D(rb) = F^{xb}$ for some x which is n-critical for F, so in particular $t_{\mathbb{A}}(D(rb)) > p(|D(rb)|, n)$. Thus

$$\Pr\Big(t_{\mathbb{A}}(D(U_{n+1})) > p(|D(U_{n+1})|, n)\Big) \geqslant \Pr\Big(t_{\mathbb{B}}(D'(U_n)) > q(|D'(U_n)|, n)\Big) \geqslant 1 - \epsilon.$$

Thus, $(D \circ U_n)_n$ is $(1/2 + \delta/2, \epsilon)$-probably p-hard for $\mathbb{A}$. □

We now consider sequences sampled with some restricted use of randomness, as announced in the Introduction.[6]

Definition 15. A sequence of random strings $(X_n)_{n\in\mathbb{N}}$ is *invertibly samplable* if it has a polynomial time sampler D which is *invertible*, i.e., D is injective and the partial function D^{-1} is computable in polynomial time.

The sampler defined in the proof of Proposition 9 is not invertible. For invertible samplers, hardness has the following handy reformulation.

Lemma 16. *Let D be an invertible polynomial time sampler for $(X_n)_{n\in\mathbb{N}}$. Then the following are equivalent.*

1. *$(X_n)_{n\in\mathbb{N}}$ is probably hard for $\mathbb{L}$.*
2. *For every polynomial time function g and for all $\epsilon > 0$ there are infinitely many n such that $\Pr(X_n \in \text{SAT}) = 1$ and*

$$\Pr\Big(|g(U_n)| \leqslant |D(U_n)| \text{ and } g(U_n) \text{ satisfies } D(U_n)\Big) \leqslant \epsilon. \tag{3}$$

Proof. ($1 \Rightarrow 2$) Assume (2) fails and choose g and ϵ witnessing this. Define the following algorithm $\mathbb{A}$: given as input a formula F, compute the string $y := g(D^{-1}(F))$ and check whether it has length $\leqslant |F|$ and satisfies F; if so, then accept with output y, else reject.

[6]Our notion of invertibility is more restrictive than the one considered in [56].

Further, define the algorithm $\mathbb{B}$ to run $\mathbb{A}$ in parallel with an arbitrary SAT-solver. If one of the two procedures halts accepting, then $\mathbb{B}$ accepts with the corresponding output. If both procedures reject, so does $\mathbb{B}$.

Since $\mathbb{A}$ is polynomial time bounded, there is a polynomial p such that $t_{\mathbb{B}}(F) \leqslant p(|F|)$ for every formula F accepted by $\mathbb{A}$. Since $\mathbb{B}$ is a SAT-solver, there exists a polynomial $p_{\mathbb{B}}$ such that then $t_{\mathbb{L}}(F) \leqslant p_{\mathbb{B}}(p(|F|))$ (Theorem 10). Thus

$$\begin{aligned}\Pr\Big(t_{\mathbb{L}}(X_n) \leqslant p_{\mathbb{B}}(p(|X_n|))\Big) &\geqslant \Pr\Big(\mathbb{A} \text{ accepts } X_n\Big)\\ &= \Pr\Big(|g(D^{-1}(X_n))| \leqslant |X_n| \text{ and } g(D^{-1}(X_n)) \text{ satisfies } X_n\Big).\end{aligned}$$

Note this last probability equals the probability in (3). By our assumption that (2) fails this probability is $> \epsilon$ or $\Pr(X_n \in \text{SAT}) < 1$ for almost all n. Hence, $(X_n)_{n\in\mathbb{N}}$ is not $(0,\epsilon)$-probably $(p_{\mathbb{B}} \circ p)$-hard for $\mathbb{L}$.

($2 \Rightarrow 1$) If (1) fails, there is a polynomial p and an $\epsilon > 0$ such that for almost all n, $\Pr(X_n \in \text{SAT}) < 1$ or $\Pr\big(t_{\mathbb{L}}(X_n) \leqslant p(|X_n|, n)\big) > \epsilon$.

Define a polynomial time function g as follows. On input r run $\mathbb{L}$ on $D(r)$ for at most $p(|D(r)|, |r|)$ steps. If this computation does not halt accepting, then return the empty string; else return $\mathbb{L}$'s output. Then $|g(r)| \leqslant |D(r)|$ for all r, and the probability in (3) equals the probability of the event that $g(U_n)$ satisfies $D(U_n)$. For n with $\Pr(X_n \in \text{SAT}) = 1$, this event is implied by the event that $t_{\mathbb{L}}(X_n) \leqslant p(|X_n|, n)$, so has probability $> \epsilon$. Thus statement (2) fails. □

We show how to get invertibility using pseudo-random generators (of the Nisan-Wigderson type). This is a standard application of the "general framework for derandomization" of [31]. For definiteness we use the parameter setting from the standard textbook [1].

Definition 17. Let $S : \mathbb{N} \to \mathbb{N}$. A function $G : \{0,1\}^* \to \{0,1\}^*$ is an *$S(\ell)$-pseudo-random generator* if $G(r)$ is computable in time $2^{O(|r|)}$, has length $S(|r|)$ and for all $\ell \in \mathbb{N}$ and all Boolean circuits C with at most $S(\ell)^3$ gates and at most $S(\ell)$ inputs

$$|\Pr\big(C(G(U_\ell)) = 1\big) - \Pr\big(C(U_{S(\ell)}) = 1\big)| < 0.1. \tag{4}$$

We say *pseudo-random generators exist* if there is $\delta > 0$ such that $2^{\lfloor \delta\ell \rfloor}$-pseudo-random generators exist.

Proposition 18. *Assume pseudo-random generators exist. If there exists a polynomial time computable hard sequence for $\mathbb{L}$, then there exists an invertibly samplable probably hard sequence for $\mathbb{L}$.*

Proof. Let $(F_n)_n$ be polynomial time computable and hard for $\mathbb{L}$. By Proposition 13 we can assume that the sequence is injective and $|F_n| \geqslant n$ for all n. Using the padding function *pad* from the proof of this proposition, define a sampler

$$D(r) := pad(F_{|r|}, r),$$

Clearly, D is polynomial time computable and invertible. Assume for the sake of contradiction that $(D(U_n))_n$ is not probably hard for $\mathbb{L}$. Applying Lemma 16 we get a polynomial time function g and $\epsilon > 0$ and $n_0 \in \mathbb{N}$ such that for all $n > n_0$:

$$\Pr(D(U_n) \in \text{SAT}) = 1 \implies \Pr(|g(U_n)| \leqslant |D(U_n)| \text{ and } g(U_n) \text{ satisfies } D(U_n)) > \epsilon. \quad (5)$$

Note $\Pr(D(U_n) \in \text{SAT})$ is 1 or 0 depending on whether $F_n \in \text{SAT}$ or not. Further note that a string satisfies $D(U_n)$ if and only if it satisfies F_n. Hence (5) implies

$$F_n \in \text{SAT} \implies \Pr(g(U_n) \text{ satisfies } F_n) > \epsilon. \quad (6)$$

Call $n \in \mathbb{N}$ *good* if $n > n_0$ and $F_n \in \text{SAT}$. We claim there is a SAT-solver $\mathbb{A}$ such that $t_{\mathbb{A}}(F_n) \leqslant n^{O(1)}$ for all good n. This implies that $(F_n)_n$ is not hard for $\mathbb{A}$ and thus also not for $\mathbb{L}$ (Lemma 11), a contradiction.

Let $c \in \mathbb{N}$ be such that $(1-\epsilon)^c \leqslant 0.9$. Let C_n be a size $n^{O(1)}$ circuit with $c \cdot n$ inputs that accepts $r_1 \cdots r_c$ with $r_i \in \{0,1\}^n$ if and only if at least one of $g(r_1), \ldots, g(r_c)$ satisfies F_n. If n is good, then $\Pr(C_n(U_{cn}) = 1) > 0.1$ by choice of c and (6). For every $m \geqslant cn$ we can view C_n as a circuit C'_m on m inputs. Further, there is a polynomial time function g such that $g(n, r)$ satisfies F_n whenever $r \in \{0,1\}^m$ is such that $C'_m(r) = 1$.

If we set $m_n := 2^{\lfloor \delta \ell_n \rfloor}$ where $\ell_n := \lfloor d \log n \rfloor$ for a sufficiently large constant $d \in \mathbb{N}$, then $m_n \geqslant cn$ and C'_{m_n} has size $\leqslant m_n^3$. Here, $\delta > 0$ witnesses that there exists a pseudo-random generator G. For all good n we have $\Pr(C'_{m_n}(U_{m_n}) = 1) > 0.1$, so $\Pr(C'_{m_n}(G(U_{\ell_n})) = 1) \neq 0$ by (4). Hence, for good n, $g(n, G(r))$ satisfies F_n for at least one $r \in \{0,1\}^{\ell_n}$.

Define the SAT-solver $\mathbb{A}$ as follows. Given a formula F it runs some arbitrary SAT-solver and in parallel does the following: compute $F_0, \ldots, F_{|F|}$; unless there is $n_0 < n \leqslant |F|$ such that $F_n = F$, reject; otherwise compute the strings $g(n, G(r))$ for all $\leqslant n^d$ many $r \in \{0,1\}^{\ell_n}$; if one of them satisfies $F = F_n$, then output it and accept; else reject.

It is easy to see that $\mathbb{A}$ is polynomially time bounded on F_n for good n, as desired. $\square$

3 Mad pseudo proof systems

3.1 Preliminaries: language L_{PV} and model M

Sofar we considered polynomial time on the set of binary strings $\{0,1\}^*$. To view polynomial time on $\mathbb{N}$, we view every $n \in \mathbb{N}$ as a binary string, say, by taking the binary expansion of n and deleting the most significant bit. Then $\{0,1\}^n$ corresponds to the numbers between 2^n and $2^{n+1} - 1$, and we continue to write $\{0,1\}^n$ for this interval.

We consider every r-ary polynomial time computable function $f : \mathbb{N}^r \to \mathbb{N}$ as an r-ary function symbol and every r-ary polynomial time decidable relation $R \subseteq \mathbb{N}^r$ as an r-ary relation symbol. Constants are nullary function symbols. Let L_{PV} denote the resulting first-order language. The *standard* L_{PV}*-structure* has as universe $\mathbb{N}$ and interprets all function and relation symbols from L_{PV} by themselves. We denote this structure also by $\mathbb{N}$ and in general do not distinguish structures from their universes notationally. The theory $\text{Th}_\forall(L_{\text{PV}})$ is the set of universal sentences true in $\mathbb{N}$.

To fix some notation we list some symbols of the language L_{PV}. It contains a unary function $|n|$ denoting the length of n as a binary string; the $\{0,1\}$-valued binary function $bit(i,n)$ gives the i-th bit of this string, and 0 if $i > |n|$. We say n *codes* the set $\llcorner n \lrcorner := \{i \in \mathbb{N} \mid bit(i,n) = 1\}$. For a set $A \subseteq \mathbb{N}$ coded in $\mathbb{N}$ let $\ulcorner A \urcorner$ denote its code. A finite function α is coded by m if m codes the set of $\langle i, \alpha(i)\rangle$ for i in the domain of α; here, $\langle i, j\rangle$ is a bijection from $\mathbb{N}^2$ onto $\mathbb{N}$. For readability we write $i \in A$ for $bit(i, \ulcorner A \urcorner) = 1$, and $\alpha(i)$ for a suitable L_{PV}-term applied to i and the code of α.

Positive rationals are coded by pairs $\langle n, m\rangle$ written n/m with $m \neq 0$ and ambiguously we use the symbol $\leqslant$ also with its meaning in the rationals. There is a unary function $card(n)$ in L_{PV} giving the cardinality of $\llcorner n \lrcorner$. Further, L_{PV} contains a unary function mapping $\ulcorner \emptyset \urcorner$ to 0, and $\ulcorner A \urcorner$ to the rational $card(\ulcorner A \urcorner)/2^n = \Pr(U_n \in A)$ for every nonempty $A \subseteq \{0,1\}^n$. We also write Pr for this function.

We fix an $\aleph_1$-saturated elementary extension M of $\mathbb{N}$. This means M is an extension of $\mathbb{N}$ with the property that every countable family of definable subsets of M with the finite intersection property has non-empty intersection. By *definable* we mean definable by formulas *with parameters* from M. Elements of $M \setminus \mathbb{N}$ are *nonstandard*.

We speak of sets and functions coded in M in the same sense as explained above, in particular, we have the notations $\llcorner a \lrcorner$ and $\ulcorner A \urcorner$ for elements a of M and coded subsets A of M. The interpretation of a symbol $\sigma \in L_{\mathrm{PV}}$ in M is denoted σ^M but we shall often omit the superscript M. Pairs $\langle a, b\rangle = \langle a, b\rangle^M$ written a/b with $b \neq 0$ are *M-rationals*. E.g. the values of $\Pr^M$ are M-rationals. Note that every (code of a) rational is an M-rational.

We use the following notions from nonstandard analysis (see e.g. [30]). The *standard part* of an M-rational a/b is the real

$$(a/b)^* := \inf\{q \in \mathbb{Q} \mid a/b \leqslant q\},$$

provided the set on the r.h.s. is non-empty; it is undefined otherwise. An M-rational with standard part 0 is *infinitesimal*.

3.2 Krajíček's model $K(F^n_{\mathrm{PV}})$

The model $K(F^n_{\mathrm{PV}})$ is Boolean valued with values in the Boolean algebra $\mathcal{B}_n$, defined below.

The function $n \mapsto \ulcorner \{0,1\}^n \urcorner$ is definable in the standard model $\mathbb{N}$. Since M is an elementary extension of $\mathbb{N}$, this function extends to a function on M. Evaluating it on $n \in M$ gives the code of a subset of M that we denote by $\{0,1\}^n$. Let $\mathcal{A}_n$ be the set of subsets of $\{0,1\}^n$ that are coded in M. Then $\mathcal{A}_n$ is a Boolean algebra and $\{\ulcorner A \urcorner \mid A \in \mathcal{A}_n\}$ is coded in M. Note that for every L_{PV}-formula $\varphi(x)$ (even with parameters from M) we have $\{\omega \in \{0,1\}^n \mid M \models \varphi(\omega)\} \in \mathcal{A}_n$.

Let $n \in M$ be nonstandard. The set

$$\mathrm{Inf}_n := \big\{A \in \mathcal{A}_n \mid \Pr^M(\ulcorner A \urcorner) \text{ is infinitesimal}\big\}$$

is an ideal in $\mathcal{A}_n$ (and not coded in M). Call A, A' equivalent if their symmetric difference is in Inf_n. The equivalence class of $A \in \mathcal{A}_n$ is denoted A/Inf_n. These classes form the

Boolean algebra $\mathcal{B}_n$, defined as the factor

$$\mathcal{B}_n := \mathcal{A}_n/\mathrm{Inf}_n.$$

Using the assumption that M is $\aleph_1$-saturated one can show [36, Lemma 1.2.1]:

Lemma 19. *For every nonstandard $n \in M$, the Boolean algebra $\mathcal{B}_n$ is complete.*

We now describe the model $K(F^n_{\mathrm{PV}})$. Its universe is F^n_{PV}, the set of all restrictions $f^M{\restriction}\{0,1\}^n$ of f^M to $\{0,1\}^n$ where $f \in L_{\mathrm{PV}}$ is a unary function symbol (and f^M its interpretation in M). We use $\alpha, \beta, \ldots$ to range over F^n_{PV}. Observe that every $\alpha \in F^n_{\mathrm{PV}}$ is coded in M (but not the set F^n_{PV}, viewed as a set of codes). Further observe that for every r-ary symbol $f \in L_{\mathrm{PV}}$ and every r-tuple $(\alpha_1, \ldots, \alpha_r)$ the function

$$\omega \mapsto f^M(\alpha_1(\omega), \ldots, \alpha_r(\omega)) \tag{7}$$

defined on $\omega \in \{0,1\}^n$ is in F^n_{PV}. We interpret the function symbols of L_{PV} in this way over F^n_{PV}. Then every closed L_{PV}-term t with parameters from F^n_{PV} denotes an element $t^{K(F^n_{\mathrm{PV}})}$ of F^n_{PV}. The Boolean valuation maps every L_{PV}-sentence φ with parameters from F^n_{PV} to a Boolean value $[\![\varphi]\!] \in \mathcal{B}_n$. For atomic φ this Boolean value is defined setting:

$$\begin{aligned} [\![R(t_1, \ldots, t_r)]\!] &:= \big\{\omega \in \{0,1\}^n \mid (t_1^{K(F^n_{\mathrm{PV}})}(\omega), \ldots, t_r^{K(F^n_{\mathrm{PV}})}(\omega)) \in R^M\big\}/\mathrm{Inf}_n, \\ [\![t = s]\!] &:= \big\{\omega \in \{0,1\}^n \mid t^{K(F^n_{\mathrm{PV}})}(\omega) = s^{K(F^n_{\mathrm{PV}})}(\omega)\big\}/\mathrm{Inf}_n, \end{aligned}$$

where $t, s, t_1, \ldots t_r$ are closed L_{PV}-terms with parameters from F^n_{PV} and $R \in L_{\mathrm{PV}}$ is an r-ary relation symbol. For arbitrary sentences with parameters in F^n_{PV} the Boolean value is then determined via the usual recurrence: $[\![\neg\varphi]\!] := \sim[\![\varphi]\!]$, $[\![(\varphi \vee \psi)]\!] := [\![\varphi]\!] \cup [\![\psi]\!]$, $[\![\exists x \varphi(x)]\!] := \sup_\alpha [\![\varphi(\alpha)]\!]$ where $\sim, \cup, \sup$ denote the obvious operations of $\mathcal{B}_n$ as a complete Boolean algebra. The minimal and maximal elements of $\mathcal{B}_n$ are respectively $0_{\mathcal{B}_n} := \emptyset/\mathrm{Inf}_n$ and $1_{\mathcal{B}_n} := \{0,1\}^n/\mathrm{Inf}_n$.

A sentence φ is *valid in* $K(F^n_{\mathrm{PV}})$ if $[\![\varphi]\!] = 1_{\mathcal{B}_n}$. One straightforwardly verifies [36, Lemma 1.4.2]:

Lemma 20. *Let $n \in M$ be nonstandard. If $\varphi(x, y, \ldots)$ is a quantifier-free L_{PV}-formula and $\alpha, \beta, \ldots \in F^n_{\mathrm{PV}}$, then*

$$[\![\varphi(\alpha, \beta, \ldots)]\!] = \{\omega \in \{0,1\}^n \mid M \models \varphi(\alpha(\omega), \beta(\omega), \ldots)\}/\mathrm{Inf}_n.$$

In particular, every sentence in $\mathrm{Th}_\forall(L_{\mathrm{PV}})$ is valid in $K(F^n_{\mathrm{PV}})$.

We close this subsection with some historical notes meant to back up our claim from the Introduction that the definition of $K(F^n_{\mathrm{PV}})$ follows natural and familiar lines.

Remark 21 (Historical notes)**.** Boolean valued models date back to the work of Rasiowa and Sikorski [51], and became popular when it was realized that Cohen's method of forcing can be viewed as a method to construct Boolean valued models of set theory. We refer to

[27, Chapter 14] and the references therein. In [53] Scott explained this view by constructing a model based on random variables of a higher-order theory of the reals as an ordered field. Such so-called Boolean powers are studied in more generality in [45, 50].

The book [36] develops Scott's [53] forcing with random variables as a method to build models $K(F)$ (and two-sorted extensions thereof) of bounded arithmetics. Instead of F^n_{PV} these use suitable families $F \subseteq M^\Omega$ for Ω coded in M, together with an analogously defined complete Boolean algebra $\mathcal{B}$. The crucial move being to restrict the construction to families F of random variables samplable with limited computational complexity. Technically, fullness ([27, p.208],[45, Theorem 1.4]) of the model is lost and much of the theory develops around finding conditions ensuring *partial* fullness for certain classes of formulas.

The models $K(F)$ can be seen as *partial* randomizations of M in the sense of Keisler [29]: the triple $(F, \mathrm{Pr}^M, \mathcal{B})$ satisfies only a fragment of Keisler's randomization theory. In particular, $K(F)$ satisfies Keisler's "Fullness Axiom" [29, p.128] only for very special F (see [36, Theorem 3.5.2]), and $K(F^n_{\mathrm{PV}})$ does not.

As remarked in [36, footnote 2, p.3] one can collapse $K(F^n_{\mathrm{PV}})$ to a usual two-valued model by factoring $\mathcal{B}_n$ with a suitable ultrafilter (see [51, Lemma 9.1]). The result is a restricted ultrapower of M. These have been studied for fragments of arithmetic [39, 26, 44, 35, 49, 21] ever since Skolem's definable ultrapower (see [23, IV.1.(b)]).

3.3 Pseudo proof systems

Let *Fml* be the set of naturals which (viewed as binary strings) code propositional formulas, and *Sat* contain the pairs (ℓ, m) such that $m \in \mathit{Fml}$ and ℓ (as a binary string) satisfies the formula coded by m. Then *Fml* and *Sat* are relation symbols in L_{PV}. The formula

$$\mathrm{Taut}(x) := \forall y(|y| \leqslant |x| \to \mathit{Sat}(y, x))$$

defines TAUT, viewed as a set of naturals. It follows from Lemma 20 that, if $f \in L_{\mathrm{PV}}$ is a proof system (i.e., $\forall x\ \mathrm{Taut}(f(x)) \in \mathrm{Th}_\forall(L_{\mathrm{PV}})$), then $f^M \restriction \{0,1\}^n \in F^n_{\mathrm{PV}}$ is a pseudo proof system as defined in [36, p.162]:

Definition 22. Let $n \in M$ be nonstandard. An element $\alpha \in F^n_{\mathrm{PV}}$ is a *pseudo proof system in* $K(F^n_{\mathrm{PV}})$ if $\mathrm{Taut}(\alpha)$ is valid in $K(F^n_{\mathrm{PV}})$.

Remark 23. Hirsch et al. [25] study so-called *heuristic proof systems.* These are randomized proof systems that are allowed to prove non-tautologies (with constant probability) but only *few* of them with respect to some distribution. Pseudo proof systems are conceptually different. First, they are not randomized. More importantly, the point of a pseudo proof system is that erroneous outputs (non-tautologies) are hard to detect as such, and not that there are few of them. In fact, as we shall see in the next section, it is conceivable that there are *mad* pseudo proof systems, pseudo proof systems all of whose outputs are erroneous. The notion of a pseudo proof system is more akin to Kabanets' pseudo$_{\mathrm{P}}$-classes [28].

Let $\mathit{neg} \in L_{\mathrm{PV}}$ map every $n \in \mathit{Fml}$ to its negation (to the number coding the negation of the formula coded by n).

Lemma 24. *Let $n \in M$ be nonstandard and $f \in L_{\mathrm{PV}}$. The following are equivalent.*

1. *$f^M\restriction\{0,1\}^n$ is a pseudo proof system in $K(F^n_{\mathrm{PV}})$.*
2. *For all $g \in L_{\mathrm{PV}}$:*

$$\big\{\omega \in \{0,1\}^n \mid M \models |g(\omega)| \leqslant |f(\omega)| \wedge \neg \mathit{Sat}(g(\omega), f(\omega))\big\} \in \mathrm{Inf}_n.$$

If $M \models \forall x \in \{0,1\}^n\ \mathit{Fml}(f(x))$, then these statements are equivalent to

3. *For all $g \in L_{\mathrm{PV}}$:*

$$\big\{\omega \in \{0,1\}^n \mid M \models \mathit{Sat}(g(\omega), \mathit{neg}(f(\omega)))\big\} \in \mathrm{Inf}_n.$$

Proof. Write $\alpha := f^M\restriction\{0,1\}^n$. Let β range over F^n_{PV}. Statement (1) means

$$\begin{aligned} 0_{\mathcal{B}_n} &= \sup_\beta \sim [\![|\beta| \leqslant |\alpha| \to \mathit{Sat}(\beta, \alpha)]\!] \\ &= \sup_\beta \sim \{\omega \in \{0,1\}^n \mid M \models |\beta(\omega)| \leqslant |\alpha(\omega)| \to \mathit{Sat}(\beta(\omega), \alpha(\omega))\}/\mathrm{Inf}_n, \end{aligned}$$

using Lemma 20. Equivalently, for all β:

$$\big\{\omega \in \{0,1\}^n \mid M \models |\beta(\omega)| \leqslant |f(\omega)| \wedge \neg \mathit{Sat}(\beta(\omega), f(\omega))\big\} \in \mathrm{Inf}_n.$$

This is equivalent to (2) because the $\beta \in F^n_{\mathrm{PV}}$ are precisely the functions of the form $g\restriction\{0,1\}^n$ for $g \in L_{\mathrm{PV}}$.

Suppose $M \models \forall x \in \{0,1\}^n\ \mathit{Fml}(f(x))$. Then for all $g \in L_{\mathrm{PV}}$, $\mathit{Sat}(g(x), \mathit{neg}(f(x))$ is equivalent to $\neg\mathit{Sat}(g(\omega), f(\omega))$ in M, so (3) implies (2). Conversely, given $g \in L_{\mathrm{PV}}$ there is $g' \in L_{\mathrm{PV}}$ such that the set in (3) for g equals the set in (2) for g': if $|g(\omega)| > |f(\omega)|$, then $g'(\omega)$ deletes the last $|g(\omega)| - |f(\omega)|$ many bits; this truncation does not change how the variables of the formula $f(\omega)$ are evaluated. $\square$

Whether or not a polynomial time function f gives rise to a pseudo proof system does not so much depend on the choice of the model M but it does depend on the choice of the nonstandard length n. The notions obtained by quantifying n either universally or existentially can be characterized in purely standard terms, i.e., without any mention of nonstandard models.

Proposition 25. *Let $f \in L_{\mathrm{PV}}$. The following are equivalent.*

1. *For all nonstandard $n \in M$:*

$$f^M\restriction\{0,1\}^n \text{ is a pseudo proof system in } K(F^n_{\mathrm{PV}}).$$

2. *For all $g \in L_{\mathrm{PV}}$:*

$$\limsup_{m\in\mathbb{N}}\ \Pr\Big(|g(U_m)| \leqslant |f(U_m)| \text{ and } (g(U_m), f(U_m)) \notin \mathit{Sat}\Big) = 0.$$

Proof. For $g \in L_{\mathrm{PV}}$ let $A_{g,f} : \mathbb{N} \to \mathbb{N}$ map $m \in \mathbb{N}$ to

$$A_{g,f}(m) := \ulcorner \{\omega \in \{0,1\}^m \mid \mathbb{N} \models |g(\omega)| \leqslant |f(\omega)| \wedge \neg Sat(g(\omega), f(\omega))\} \urcorner. \tag{8}$$

This function is definable in the standard model $\mathbb{N}$, so extends to a function $A^M_{g,f}$ on M. We have for all $n \in M$:

$$\llcorner A^M_{g,f}(n) \lrcorner := \{\omega \in \{0,1\}^n \mid M \models |g(\omega)| \leqslant |f(\omega)| \wedge \neg Sat(g(\omega), f(\omega))\} \in \mathcal{A}_n.$$

$(1 \Rightarrow 2)$. If (2) fails, there are $g \in L_{\mathrm{PV}}$ and $\ell \in \mathbb{N}$ such that

$$\mathbb{N} \models \forall x \exists y > x \ \ \Pr(A_{g,f}(y)) \geqslant 1/\ell.$$

By elementary equivalence, this sentence holds in M. Plug some nonstandard $a \in M$ for x and choose $n \in M$ witnessing y. Then n is nonstandard, and $M \models \Pr(A_{g,f}(n)) \geqslant 1/\ell$. Thus $(\Pr^M(A^M_g(n))^* \geqslant 1/\ell > 0$, so $\llcorner A^M_{g,f}(n) \lrcorner \notin \mathrm{Inf}_n$. But this is the set in Lemma 24 (2). Hence $f^M \restriction \{0,1\}^n$ is not a pseudo proof system in $K(F^n_{\mathrm{PV}})$.

$(2 \Rightarrow 1)$. Let $n \in M$ be nonstandard. We verify Lemma 24 (2). So let $g \in L_{\mathrm{PV}}$. We show that $M \models \Pr(A_{g,f}(n)) < 1/\ell$ for all $\ell \in \mathbb{N}$. Given $\ell \in \mathbb{N}$, (2) gives $t \in \mathbb{N}$ such that

$$\mathbb{N} \models \forall y > t \ \ \Pr(A_{g,f}(y)) < 1/\ell.$$

By elementary equivalence, this sentence holds in M. Since n is nonstandard, we have $t <^M n$. Plugging n for y gives $M \models \Pr(A_g(n)) < 1/\ell$, as desired. □

Our proof of the dual statement is less straightforward. For later use, we give a slightly more general statement involving an arbitrary L_{PV}-formula $\psi(x)$:

Proposition 26. *Let $f \in L_{\mathrm{PV}}$ and $\psi(x)$ be an L_{PV}-formula. The following are equivalent.*

1. *For some nonstandard $n \in M$ with $M \models \psi(n)$:*

$$f^M \restriction \{0,1\}^n \text{ is a pseudo proof system in } K(F^n_{\mathrm{PV}}).$$

2. *For all $g \in L_{\mathrm{PV}}$:*

$$\liminf_{m \in \mathbb{N}} \ \Pr\Big(\text{if } \mathbb{N} \models \psi(m), \text{ then } |g(U_m)| \leqslant |f(U_m)| \text{ and } (g(U_m), f(U_m)) \notin Sat\Big) = 0.$$

Note the probability in statement (2) equals 1 if $\mathbb{N} \not\models \psi(m)$, and otherwise it equals the probability in the previous Proposition 25 (2).

Proof. We use the notation $A_{g,f}(y)$ from the proof of the previous proposition (cf. (8)).

$(1 \Rightarrow 2)$. If (2) fails, there are $g \in L_{\mathrm{PV}}$ and $\ell, t \in \mathbb{N}$, such that

$$\mathbb{N} \models \forall y > t \ (\psi(y) \to \Pr(A_{g,f}(y)) \geqslant 1/\ell).$$

Then, being an elementary extension, M satisfies this sentence. Thus, for every nonstandard $n \in M$ with $M \models \psi(n)$ we have $(\Pr(A^M_{g,f}(n)))^* \geqslant 1/\ell$, so $\llcorner A^M_{g,f}(n) \lrcorner \notin \mathrm{Inf}_n$, i.e., Lemma 24 (2) fails.

($2 \Rightarrow 1$). For $g \in L_{\mathrm{PV}}$ and standard $\ell > 0$ let $\varphi_{g,f,\ell}(x)$ be the formula

$$x \geqslant \ell \wedge \psi(x) \wedge \Pr(A_{g,f}(x)) < 1/\ell. \tag{9}$$

Given finitely many such formulas $\varphi_{g_0,f,\ell_0}, \ldots, \varphi_{g_k,f,\ell_k}$ set $\ell := \max_{i \leqslant k} \ell_i$ and let $g \in L_{\mathrm{PV}}$ be computed by the following polynomial time algorithm: on input ω, compute $f(\omega), g_0(\omega), \ldots, g_k(\omega)$; if there is $i \leqslant k$ such that $|g_i(\omega)| \leqslant |f(\omega)|$ and $(g_i(\omega), f(\omega)) \notin \mathit{Sat}$, then output such $g_i(\omega)$ (say for the least such $i \leqslant k$); otherwise output 0.

Then we have for all $m \in \mathbb{N}$ that $\bigcup_{i \leqslant k} \llcorner A_{g_i,f}(m) \lrcorner \subseteq \llcorner A_{g,f}(m) \lrcorner$ and thus for all $i \leqslant k$:

$$\Pr(A_{g_i,f}(m)) \leqslant \Pr(A_{g,f}(m)). \tag{10}$$

Now, (2) gives for g and ℓ infinitely many $m \in \mathbb{N}$ such that the probability in (2) is smaller $1/\ell$. Choose such $m \geqslant \ell$. Then $\mathbb{N} \models \psi(m)$ and $\Pr(A_{g,f}(m)) < 1/\ell$. By choice of ℓ and (10) we get $\Pr(A_{g_i,f}(m)) < 1/\ell \leqslant 1/\ell_i$ for all $i \leqslant k$. That is, $m \geqslant \ell \geqslant \ell_i$ satisfies $\varphi_{g_i,\ell_i}(x)$ for all $i \leqslant k$.

Hence, any finitely many of the formulas $\varphi_{g,f,\ell}(x)$ for $g \in L_{\mathrm{PV}}$ and standard $\ell > 0$ are jointly satisfiable in $\mathbb{N}$ and hence in M. In other words, the family of subsets of M defined by these formulas has the finite intersection property. Since M is $\aleph_1$-saturated, there is $n \in M$ satisfying all these formulas. This n is nonstandard and satisfies $M \models \psi(n)$ and $\Pr^M(A_{g,f}(n)) < 1/\ell$ for all standard $\ell > 0$ and all $g \in L_{\mathrm{PV}}$. Hence, $\llcorner A_{g,f}(n) \lrcorner \in \mathrm{Inf}_n$ for all $g \in L_{\mathrm{PV}}$. This is Lemma 24 (2), so $f^M \upharpoonright \{0,1\}^n$ is a pseudo proof system in $K(F^n_{\mathrm{PV}})$. □

3.4 Madness

Definition 27. Let $n \in M$ be nonstandard. A pseudo proof system $f^M \upharpoonright \{0,1\}^n$ in $K(F^n_{\mathrm{PV}})$ is *mad* if

$$M \models \forall x \in \{0,1\}^n (\mathit{Fml}(f(x)) \wedge \neg \mathrm{Taut}(f(x))). \tag{11}$$

Putting everything together yields the statements announced in the Introduction.

Proof of Theorem 2. Statements (1) and (3) of the theorem are equivalent to statements (1) and (2) of Proposition 26 respectively, taking the formula in (11) as $\psi(n)$.

We prove the equivalence of (1) and (2).

($1 \Rightarrow 2$) Suppose there are $n \in M \setminus \mathbb{N}$ and $f \in L_{\mathrm{PV}}$ such that $f^M \upharpoonright \{0,1\}^n$ is a mad pseudo proof system. Define

$$D(r) := \mathit{pad}(\mathit{neg}(f(r)), r)$$

where *pad* is the padding function from the proof of Proposition 13. Then D is in L_{PV} and invertible. By Lemma 11 it suffices to show that D samples a sequence that is probably hard for Levin's $\mathbb{L}$.

By (11) we have $M \models \psi(n)$ where $\psi(x)$ is the L_{PV}-formula

$$\forall y \in \{0,1\}^x \; \exists z \; \mathit{Sat}(z, \mathit{neg}(f(y))). \tag{12}$$

Since $M \models \forall x \in \{0,1\}^n \mathit{Fml}(f(x))$ by (11), we get Lemma 24 (3). Note $\mathit{Sat}(y, \mathit{neg}(f(x)))$ is equivalent to $\mathit{Sat}(y, D(x))$ in M, so for all $g \in L_{\mathrm{PV}}$:

$$\big\{\omega \in \{0,1\}^n \mid M \models |g(\omega)| \leqslant |D(\omega)| \wedge \mathit{Sat}(g(\omega), D(\omega))\big\} \in \mathrm{Inf}_n.$$

This set equals $\llcorner B^M_{g,D}(n) \lrcorner$, where $B_{g,D}(x)$ is the function, definable in $\mathbb{N}$, mapping m to the code of $\big\{\omega \in \{0,1\}^m \mid \mathbb{N} \models |g(\omega)| \leqslant |D(\omega)| \wedge \mathit{Sat}(g(\omega), D(\omega))\big\}$. This implies for all $g \in L_{\mathrm{PV}}$ and all standard $\ell > 0$ that M and hence $\mathbb{N}$ models

$$\exists x \geqslant \ell \; \big(\psi(x) \wedge \Pr(B_{g,D}(x)) < 1/\ell\big).$$

This implies Lemma 16 (2), so $(D(U_n))_n$ is probably hard for $\mathbb{L}$.

($2 \Rightarrow 1$) Suppose $D \in L_{\mathrm{PV}}$ is an invertible sampler of a sequence probably hard for $\mathbb{L}$, or equivalently, suppose Lemma 16 (2) holds. We can assume that D only outputs formulas, i.e., its range is included in *Fml*. Define the function

$$f(r) := \mathit{neg}(D(r)).$$

Then $f \in L_{\mathrm{PV}}$ and we claim that there is a nonstandard $n \in M$ such that $f^M \upharpoonright \{0,1\}^n$ is a mad pseudo proof system in $K(F^n_{\mathrm{PV}})$.

Note $\forall y \mathit{Fml}(f(y))$ holds in $\mathbb{N}$ and hence in M. So to get (11) it suffices to get $M \models \psi(n)$ for the formula $\psi(x)$ defined in (12). Hence, we are left to verify Proposition 26 (2):

$$\liminf_m \; \Pr\Big(\text{if } \mathbb{N} \models \psi(m), \text{ then } |g(U_m)| \leqslant |\mathit{neg}(D(U_m))| \\ \text{and } (g(U_m), \mathit{neg}(D(U_m))) \notin \mathit{Sat}\Big) = 0,$$

for all $g \in L_{\mathrm{PV}}$. Observe $(x, \mathit{neg}(D(y))) \notin \mathit{Sat}$ and $(x, D(y)) \in \mathit{Sat}$ are equivalent. Further, $\mathbb{N} \models \psi(m)$ means $\Pr(D(U_m) \in \textsc{sat}) = 1$. Thus we have to show that for all $g \in L_{\mathrm{PV}}$ and all $\epsilon > 0$ there are infinitely many $m \in \mathbb{N}$ such that $\Pr(D(U_m) \in \textsc{sat}) = 1$ and

$$\Pr\Big(|g(U_m)| \leqslant |\mathit{neg}(D(U_m))| \text{ and } (g(U_m), D(U_m)) \in \mathit{Sat}\Big) \leqslant \epsilon.$$

This follows from Lemma 16 (2). □

Proof of Corollary 3. By assumption $\mathrm{NE} \cap \mathrm{coNE} \not\subseteq \mathrm{E}$ and Proposition 13, there exist polynomial time computable hard sequences for $\mathbb{L}$. By the assumption that pseudo-random generators exist and Proposition 18, there exists an invertibly samplable probably hard sequence for $\mathbb{L}$. By optimality of $\mathbb{L}$, this sequence is probably hard for every SAT-solver. Now apply Theorem 2. □

4 Discussion

4.1 Standard definitions of pseudo proof systems

It is not entirely straightforward to give a definition of pseudo proof systems in standard terms. Informally, one would define a polynomial time computable function f mapping binary strings to propositional formulas to be a *pseudo proof system* if for all polynomial time functions g the probability

$$\Pr\big(g(U_n) \text{ is a falsifying assignment of } f(U_n)\big)$$

is "small" (as a function of n); see statement (3) in Theorem 2. We avoided such a definition because the right notion of "small" likely depends on context. Propositions 25 and 26 respectively motivate to take "small" to mean "$o(1)$" and "not $\Omega(1)$". Further, from a purely[7] complexity theoretic perspective, "negligible" suggests itself.

4.2 Hypotheses concerning hard sequences

Consider the following three statements:

(i) Every SAT-solver has a polynomial time computable hard sequence.

(ii) Every SAT-solver has an invertibly samplable probably hard sequence.

(iii) Every SAT-solver has a samplable probably hard sequence.

Each of these statements is equivalent to the assertion that there exists a sequence of the respective type for Levin's optimal SAT-solver $\mathbb{L}$ (Lemma 11). Trivially, (i) implies (iii), and (ii) implies (iii). By Proposition 18, (i) implies (ii) if pseudo-random generators exist. By Proposition 13, (i) holds if $\mathrm{NE} \cap \mathrm{coNE} \not\subseteq \mathrm{E}$. By Theorem 2, (ii) is equivalent to the existence of mad pseudo proof systems.

We do not know a well-established hypothesis implying (i) for general SAT-algorithms instead SAT-solvers, that is, the Hard Sequence Hypothesis. In particular, we do not know whether this hypothesis is implied by the hypothesis that SAT does not have optimal algorithms, or equivalently [52], p-optimal proof systems. By Proposition 9, (iii) for SAT-algorithms holds under cryptographic assumptions.

Acknowledgments

We thank Yijia Chen, Jan Krajíček and Ján Pich for valuable comments during early stages of this work. We thank Edward Hirsch for pointing out reference [56].

[7]See, however, the remark in [36, p.15] on probability amplification in the model-theoretic setting.

References

[1] S. Arora and B. Barak. Computational Complexity, A Modern Approach. Cambridge University Press, 2009.

[2] A. Atserias. Distinguishing SAT from polynomial-size circuits, through black-box queries. Proceedings of 21st Annual IEEE Conference on Computational Complexity, IEEE Press, pp. 88-95, 2006.

[3] S. Ben-David and A. Gringauze. On the existence of optimal propositional proof systems and oracle-relativized propositional logic. Electronic Colloquium on Computational Complexity, Report TR98-021, 1998.

[4] L. Berman and J. Hartmanis. On isomorphisms and density of NP and other complete sets SIAM Journal on Computing 6:2: 305-322, 1977.

[5] O. Beyersdorff. On the correspondence between arithmetic theories and propositional proof systems - a survey. Mathematical Logic Quartely 55(2):116-137, 2009.

[6] A. Bogdanov, K. Talwar and A. Wan. Hard instances for satisfiability and quasi-one-way functions. Proceedings Innovations in Computer Science, pp. 290-300, 2010.

[7] A. Bogdanov and L. Trevisan. Average-Case Complexity. Foundations and Trends in Theoretical Computer Science 1(2), 2006.

[8] N. H. Bshouty, R. Cleve, R. Gavaldà, S. Kannan and C. Tamon. Oracles and queries that are sufficient for exact learning. Journal of Computer and System Sciences 52 (3): 421-433, 1996.

[9] S. R. Buss. First-order proof theory of arithmetic. In S. R. Buss (ed.), Handbook of Proof Theory, Studies in Logic 137, Elsevier, Ch. II, pp. 79-148, 1998.

[10] S. R. Buss. Some remarks on the lengths of propositional proofs. Archive for Mathematical Logic 34: 377-394, 1995.

[11] S. R. Buss. Propositional proofs in Frege and Extended Frege systems (Abstract). Proceedings of the 10th International Computer Science Symposium in Russia (CSR'15). LNCS 9139. Springer, pp. 1-6, 2015.

[12] B. Chapman and R. Williams. The circuit-input game, natural proofs, and testing circuits with data. Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science (ITCS'15), pp.263-270, 2015.

[13] Y. Chen and J. Flum. On the complexity of Gödel's proof predicate. The Journal of Symbolic Logic 75 (1): 239-254, 2010.

[14] Y. Chen and J. Flum. From almost optimal algorithms to logics for complexity classes via listings and a halting problem. Journal of the ACM 59(4): article 17, 2012.

[15] Y. Chen and J. Flum. On optimal inverters. The Bulletin of Symbolic Logic 20 (01): 1-23, 2014.

[16] Y. Chen, J. Flum and M. Müller. Consistency, optimality, and incompleteness. Annals of Pure and Applied Logic 164 (12) 1224-1235, 2013.

[17] Y. Chen, J. Flum and M. Müller. Hard instances of algorithms and proof systems. ACM Transactions on Computation Theory 6 (2): Article No. 7, 2014.

[18] Y. Chen, J. Flum and M. Müller. On optimal probabilistic algorithms for SAT. In: S.-D. Friedman, M. Koerwien, M. Müller (eds.), The Infinity Project, A 2009-2011 Research Programme, pp. 225-231, 2012.

[19] S. A. Cook and R. A. Reckhow. The relative efficiency of propositional proof systems. Journal of Symbolic Logic 44 (1): 36-50, 1979.

[20] D. Gutfreund, R. Shaltiel and A. Ta-Shma. If NP languages are hard on the worst-case, then it is easy to find their hard instances. Computational Complexity 16 (4): 412-441, 2007.

[21] Michal Garlík. Construction of models of bounded arithmetic by restricted reduced powers. Archive for Mathematical Logic 55 (5): 625-648, 2016.

[22] O. Goldreich. A Primer on Pseudorandom Generators. AMS University Lecture Series 55, 2010.

[23] P. Hájek and P. Pudlák. Metamathematics of First-Order Arithmetic. 2nd ed., Perspectives in Mathematical Logic, Springer, 1998.

[24] E. A. Hirsch. Optimal acceptors and optimal proof systems. Proceedings of 7th Annual Conference Theory and Applications of Models of Computation (TAMC'10), LNCS 6108, pp. 28-39, 2010.

[25] E. A. Hirsch, D. Itsykson, I. Monakhov and A. Smal. On optimal heuristic randomized semidecision procedures, with application to proof complexity and cryptography. Theory of Computing Systems 51: 179-195, 2012.

[26] J. Hirschfeld. Models of arithmetic and recursive functions. Israel Journal of Mathematics 20: 111-126, 1975.

[27] T. Jech. Set Theory, The Third Millenium Edition, Revised and Expanded. Springer Monographs in Mathematics, 2002.

[28] V. Kabanets. Easiness assumptions and hardness tests: Trading time for zero error. Journal of Computer and System Sciences 63 (2): 236-252, 2001.

[29] J. Keisler. Randomizing a model. Advances in Mathematics 143: 124-158, 1999.

[30] J. Keisler. Foundations of Infinitesimal Calculus. Prindle Weber & Schmidt, 1976.

[31] A. R. Klivans and D. van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. SIAM Journal on Computing 31 (5): 1501-1526, 2002.

[32] J. Köbler and J. Messner. Complete problems for promise classes by optimal proof systems for test sets. In Proceedings of the 13th IEEE Conference on Computational Complexity (CCC'98), pp. 132-140, 1998.

[33] J. Köbler and J. Messner. Is the standard proof system for SAT p-optimal? Proceedings of Foundations of Software Technology and Theoretical Computer Science (FSTTCS'00), LNCS 1974, pp. 361-372, 2000.

[34] J. Krajíček. Bounded Arithmetic, Propositional Logic, and Complexity Theory. Encyclopedia of Mathematics and its Applications, Cambridge University Press, 1995.

[35] J. Krajíček. Extensions of models of PV. In: J.A.Makowsky and E.V.Ravve (eds.), Logic Colloquium'95, ASL/Springer Series, Lecture Notes in Logic 11, pp.104-114, 1998.

[36] J. Krajíček. Forcing with random variables and proof complexity, London Mathematical Society Lecture Note Series 382, Cambridge University Press, 2011.

[37] J. Krajíček. A note on SAT algorithms and proof complexity, Information Processing Letters 112: 490-493, 2012.

[38] J. Krajíček and P. Pudlák. Propositional proof systems, the consistency of first order theories and the complexity of computations. Journal of Symbolic Logic 54 (3): 1063-1079, 1989.

[39] S. Kripke and S. Kochen. Nonstandard models of Peano arithmetic. In: H. Lauchli (ed.), Logic and Algorithmics: International Symposium Held in Honor of Ernst Specker, Monograph No. 30 of L'Enseignement Mathematique, University of Geneva, pp. 277-295, 1982.

[40] L. Levin. Universal sequential search problems. Problems of Information Transmission 9 (3): 265-266, 1973.

[41] R. J. Lipton and N. E. Young. Simple strategies for large zero-sum games with applications to complexity theory. Proceedings of the 26th annual ACM Symposium on Theory of Computing (STOC'94), pp. 734-740, 1994.

[42] J. Maly. Jan Krajíček's Forcing Construction and Pseudo Proof Systems. Master's Thesis, University of Vienna, 2016.

[43] E. Mayordomo. Almost every set in exponential time is P-bi-immune. Theoretical Computer Science 136 (2): 487-506, 1994.

[44] T. G. McLaughlin. Sub-arithmetical ultrapowers: a survey. Annals of Pure and Applied Logic 49 (2): 143-191, 1990.

[45] R. Mansfield. The theory of Boolean ultrapowers. Annals of Mathematical Logic 2 (3): 297-323, 1971.

[46] J. Messner. On optimal algorithms and optimal proof systems. Proceedings of the 16th Annual Symposium on Theoretical Aspects of Computer Science (STACS'99), LNCS 1563, pp. 541-550, 1999.

[47] J. Messner. On the Simulation Order of Proof Systems. Ph.D. Dissertation. University of Ulm, 2000.

[48] J. Pich. Circuit lower bounds in bounded arithmetics. Annals of Pure and Applied Logic 166 (1): 29-45, 2015

[49] P. Pudlák. Randomness, pseudorandomness and models of arithmetic. New Studies in Weak Arithmetics, P. Cgielski and Ch. Cornaros (eds.), CSLI Publications, Stanford, pp.199-216, 2013.

[50] K. Potthoff. Boolean ultrapowers. Archiv für mathematische Logik und Grundlagenforschung 16: 37-48, 1974.

[51] H. Rasiowa and R. Sikorski. Algebraic treatment of the notion of satisfiability. Fundamenta Mathematicae 40 (1): 62-95, 1953.

[52] Z. Sadowski. On an optimal deterministic algorithm for SAT. Proceedings of the 12th International Workshop on Computer Science Logic (CSL'98), LNCS 1584, pp.179-187, 1999.

[53] D. Scott. A proof of the independence of the continuum hypothesis. Mathematical Systems Theory 1 (2): 89-111, 1967.

[54] L. Stockmeyer. The Complexity of Decision Problems in Automata Theory. Ph.D. Dissertation, MIT, 1974.

[55] N. Vereshchagin, An improving on Gutfreund, Shaltiel, and Ta-Shma's paper "If NP Languages are Hard on the Worst-Case, Then it is Easy to Find Their Hard Instances". 8th Computer Science Symposium in Russia (CSR'13), LNCS 7913: 203-211, 2013.

[56] N. Vereshchagin. Encoding invariance in average case complexity. Theory of Computing Systems 54 (2): 305-317, 2014.