A remark on 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

January 22, 2018

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. We show that the existence of *mad* pseudo proof systems is equivalent to the existence of a randomized polynomial time procedure with a highly restrictive use of randomness which produces satisfiable formulas whose satisfying assignments are probably hard to find.

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 tautologies. 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 [11] 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 only if NP = coNP [11].

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" [23, p.175]. A general method to construct such models is developed in [23] following Scott's [35] forcing with random variables. An important instance of this method is the Boolean valued

*This paper reports results of the first author's Master's Thesis [28].

†Supported by the Austrian Science Fund (FWF) under project number P25207 and Y698.
‡Supported by the Austrian Science Fund (FWF) under project number P28699.

model $K(F^n_{\mathrm{PV}})$.[1] 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. In particular, this holds for Cook's theory PV [9] formalizing feasible reasoning. 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 16) 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" [23, 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* (Definition 17). 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 (Definition 20). 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 [23, 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. In this note we prove 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.

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]

[1] A short introduction to the model-theoretic concepts is given in section 3, no knowledge of [23] is assumed.

[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 [8].

Hard sequences have been studied from at least two perspectives. The first is *speed-up*, going back at least to [36], and the second, more relevant to this paper, is to *witness failure* of feasible algorithms, studied not only for deterministic but also for randomized [12, 37, 3] and non-uniform algorithms [27, 2]. See [31, 6] for some recent discussions.

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.

A natural (see e.g. [12]) weakening of the Hard Sequence Hypothesis is to allow randomness in the construction of hard sequences. One then asks for polynomial time *samplable* (as opposed to computable) *probably hard* sequences (Definitions 4, 3). We observe that SAT-algorithms do have such sequences under cryptographic assumptions (Proposition 6). Second, we observe that SAT-*solvers* do have polynomial time computable hard sequences under a well-established hypothesis (Proposition 9): 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 $x = x_1 \cdots x_n \in \{0,1\}^n$ 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.

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 10). Intuitively, the sampler is required to witness its outputs by publishing the random seed used.

Theorem 2. *Let M be an $\aleph_1$-saturated elementary extension of $\mathbb{N}$. Then the following are equivalent:*

(a) There is a nonstandard $n \in M$ such that $K(F^n_{\mathrm{PV}})$ contains mad pseudo proof systems.

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

(c) 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 $f(x)$.

These statements are true if $\mathrm{NE} \cap \mathrm{coNE} \not\subseteq \mathrm{E}$ *and pseudo-random generators exist.*

The property in (c) could be taken as a standard definition of a mad pseudo proof system, i.e. one not referring to nonstandard models. It is our notion of invertibility that makes the equivalence of (b) and (c) an easy consequence of Levin's optimal SAT-solver [26]. The equivalence to (a) is the remark this short paper wants to communicate, as a contribution to the model-theory of the structures $K(F^n_{\mathrm{PV}})$.

2 Hard sequences

Assuming NP $\not\subseteq$ P, Gutfreund et al. [12] showed that for every fixed polynomial p every SAT-algorithm has polynomial time computable p-hard sequence. A diagonalizing argument shows that one can compute hard sequences in slightly superpolynomial time (see [12, Theorem 1.6] for such a construction) but the construction of polynomial time computable hard sequences is open.

Does randomness help? As for a notion of feasibility for sequences of random strings[3] we borrow the following from average case complexity [4]:

Definition 3. 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 [12, 37]. Here, we restrict attention to deterministic algorithms.

Definition 4. 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 $\mathbb{A}$.

We observe that, using randomness, (superpolynomial) hardness is achievable under cryptographic assumptions:[4]

Definition 5. 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 6. *Assume cryptographic pseudo-random generators with stretch $2n$ exist. Then there is a samplable sequence which is probably hard for every* SAT*-algorithm.*

[3] 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.

[4] Such assumptions are prohibitive in the context of [12, 2, 3] who are concerned with the problem to reduce average-case hardness hypotheses to worst-case hardness hypotheses.

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 $\Pr(\mathbb{A} \text{ accepts } G(U_n)) \geqslant \epsilon$ 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. □

While we are not aware of a reference for the above result, its proof is certainly folklore. Early references for similar constructions are [10, 5] where, instead of general SAT-algorithms, SAT-solvers are considered. Cryptographic assumptions are not required in this case, e.g. [24, 3] construct p-hard sequences assuming certain proof lower bounds and NP $\not\subseteq$ P respectively. The sequences in [10, 5, 3] also produce formulas along with satisfying assignments, and it is clear that deterministic such sequences, namely *dreambreakers* [3], cannot be (superpolynomially) hard for Levin's optimal SAT-solver $\mathbb{L}$:

Theorem 7 (Levin [26]). *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 8. *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. Let $\mathbb{A}$ be a SAT-solver. Assume for the sake of a contradiction that $(X_n)_{n\in\mathbb{N}}$ is $(\delta, \epsilon - \delta)$-probably hard for $\mathbb{L}$ but not for $\mathbb{A}$. Then, let p be 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 7. Then (2) implies

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

and thus

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

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

We point out that one can construct superpolynomially hard sequences for SAT-solvers under standard worst-case assumptions. The proof is essentially known. Recall, E and NE denote deterministic and nondeterministic simply exponential time $2^{O(n)}$, respectively.

Proposition 9. *The following statements are equivalent, and implied by* $\mathrm{NE} \cap \mathrm{coNE} \not\subseteq \mathrm{E}$.

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

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

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

(d) 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. $(a \Rightarrow b)$ follows from the optimality of $\mathbb{L}$ (Lemma 8). $(b \Rightarrow c)$ and $(d \Rightarrow a)$ are trivial. To prove $(c \Rightarrow d)$ we proceed as in [8, 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 $\mathit{pad}(F, y)$ 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 $\mathit{pad}(F, y)$ to F and y, respectively.

Let $\mathbb{A}$ be a SAT-solver and assume (c). 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 $\mathit{pad}(F, 1^0), \ldots, \mathit{pad}(F, 1^t)$; 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 $t_{\mathbb{B}}(F) \leqslant p(t + t_{\mathbb{A}}(\mathit{pad}(F, 1^t)))$. By (c) there is a polynomial time computable sequence $(F_n)_n$ hard for $\mathbb{B}$. Then $(\mathit{pad}(F_n, 1^n))_n$ is polynomial time computable and hard for $\mathbb{A}$. This sequence is injective and satisfies $|\mathit{pad}(F_n, 1^n)| \geqslant n$ for all $n \in \mathbb{N}$.

We have proved that (a)-(d) are equivalent. We now derive (b) 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 [7, 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 $r_1(1^n) \vee r_0(1^n) \in \mathrm{SAT}$, 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 $t_{\mathbb{A}}(r_1(1^n) \vee r_0(1^n)) \leqslant n^{O(1)}$. □

We now consider sequences sampled with some restricted use of randomness, as announced in the Introduction.[5]

Definition 10. 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.

[5] Our notion of invertibility is more restrictive than the one considered in [38].

The sampler defined in the proof of Proposition 6 is not invertible. For invertible samplers, hardness has the following handy reformulation.

Lemma 11. *Let D be an invertible polynomial time sampler for $(X_n)_{n\in\mathbb{N}}$. Then the following are equivalent.*

(a) $(X_n)_{n\in\mathbb{N}}$ *is probably hard for* $\mathbb{L}$.

(b) For every polynomial time function g and for all $\epsilon > 0$ there are infinitely many n such that $\Pr(X_n \in \textsc{sat}) = 1$ *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. $(a \Rightarrow b)$ Assume (b) 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.

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 7). 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 (a) fails this probability is $> \epsilon$ or $\Pr(X_n \in \textsc{sat}) < 1$ for almost all n. Hence, $(X_n)_{n\in\mathbb{N}}$ is not $(0, \epsilon)$-probably $(p_{\mathbb{B}} \circ p)$-hard for $\mathbb{L}$.

$(b \Rightarrow a)$ If (a) fails, there is a polynomial p and an $\epsilon > 0$ such that for almost all n, $\Pr(X_n \in \textsc{sat}) < 1$ 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 $\Pr(X_n \in \textsc{sat}) = 1$, this event is implied by the event that $t_{\mathbb{L}}(X_n) \leqslant p(|X_n|, n)$, so has probability $> \epsilon$. Thus statement (b) fails. $\square$

An easy corollary is the equivalence of statement (b) and (c) of our main theorem.

Proof of Theorem 2 $(b \Leftrightarrow c)$. It follows from the optimality of $\mathbb{L}$ that there is D as in Lemma 11 (a) if and only if if Theorem 2 (b) holds.

We show that there is D as in Lemma 11 (b) if and only if Theorem 2 (c) holds. Given a sampler D as in Lemma 11 (b), define

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

where *neg* maps (a binary string coding) a formula to a its negation. Conversely, given a function f as in Theorem 2 (c), define the invertible sampler

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

where *pad* is the padding function from the proof of Proposition 9.

For every $n \in \mathbb{N}$ and every polynomial time computable function g we have: $D(r)$ is a satisfiable formula for all $r \in \{0,1\}^n$ if and only if $f(r)$ is a falsifiable formula for all $r \in \{0,1\}^n$; further, the probability that $g(U_n)$ satisfies $D(U_n)$ is $\leqslant \epsilon$ if and only if $g(r)$ is a falsifies $f(r)$ for only a ϵ-fraction of $r \in \{0,1\}^n$. □

Hard sequences can be transformed into invertibly samplable probably hard sequences using pseudo-random generators (of the Nisan-Wigderson type). This is a standard application of the "general framework for derandomization" of [21]. For definiteness we use the parameter setting from the standard textbook [1].

Definition 12. 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 13. *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 9 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

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

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 11 we get a polynomial time function g and $\epsilon > 0$ and $n_0 \in \mathbb{N}$ such that for all $n > n_0$:

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

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

$$F_n \in \textsc{sat} \implies \Pr(g(U_n) \text{ satisfies } F_n) > \epsilon. \tag{6}$$

Call $n \in \mathbb{N}$ *good* if $n > n_0$ and $F_n \in \textsc{sat}$. We claim there is a $\textsc{sat}$-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 8), 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 $\textsc{sat}$-solver $\mathbb{A}$ as follows. Given a formula F it runs some arbitrary $\textsc{sat}$-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

So far 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 $\mathrm{Th}_\forall(L_{\mathrm{PV}})$ 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. [20]). 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 [23, Lemma 1.2.1]:

Lemma 14. *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 [23, Lemma 1.4.2]:

Lemma 15. *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 16 (Historical notes)**.** Boolean valued models date back to the work of Rasiowa and Sikorski [34], 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 [17, Chapter 14] and the references therein. In [35] 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 Boolean powers are studied in more generality in [30, 33].

The book [23] develops Scott's [35] 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 ([17, p.208],[30, 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 [19]: 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" [19, p.128] only for very special F (see [23, Theorem 3.5.2]), and $K(F^n_{\mathrm{PV}})$ does not.

As remarked in [23, 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 [34, Lemma 9.1]). The result is a restricted ultrapower of M. These have been studied for fragments of arithmetic [25, 16, 29, 22, 32, 13] ever since Skolem's definable ultrapower (see [14, 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 15 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 [23, p.162]:

Definition 17. 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 18. Hirsch et al. [15] study *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 [18].

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

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

(b) For all $g \in L_{\mathrm{PV}}$:

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

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

(c) For all $g \in L_{\mathrm{PV}}$:

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

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

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

using Lemma 15. Equivalently, for all β:

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

This is equivalent to (b) 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 $M \models \forall x \in \{0,1\}^n \ Fml(f(x))$. Then for all $g \in L_{\mathrm{PV}}$, $Sat(g(x), neg(f(x))$ is equivalent to $\neg Sat(g(\omega), f(\omega))$ in M, so (c) implies (b). Conversely, given $g \in L_{\mathrm{PV}}$ there is $g' \in L_{\mathrm{PV}}$ such that the set in (c) for g equals the set in (b) 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. □

We are interested in pseudo proof systems that never output a tautology.

Definition 20. Let $n \in M$ be nonstandard. A pseudo proof system $f^M\restriction\{0,1\}^n$ in $K(F^n_{\mathrm{PV}})$ is *mad* if

$$M \models \forall x \in \{0,1\}^n (Fml(f(x)) \wedge \neg \mathrm{Taut}(f(x))). \tag{8}$$

3.4 Proof of Theorem 2

We already showed that $(b \Leftrightarrow c)$, and now show $(a \Leftrightarrow c)$.

(a $\Rightarrow$ c) Suppose there are $n \in M\setminus\mathbb{N}$ and $f \in L_{\mathrm{PV}}$ such that $f^M\restriction\{0,1\}^n$ is a mad pseudo proof system. We claim that $f^{\mathbb{N}}$ satisfies (i) and (ii) of (c). By (8) we have $M \models \psi(n)$ where $\psi(x)$ is the L_{PV}-formula

$$\forall y \in \{0,1\}^x \ \exists z \ Sat(z, neg(f(y))). \tag{9}$$

Now, let $A_{g,f}(x)$ be the function, definable in $\mathbb{N}$, mapping m to the code of

$$\big\{\omega \in \{0,1\}^m \mid \mathbb{N} \models |g(\omega)| \leqslant |neg(f(\omega))| \wedge Sat(g(\omega), neg(f(\omega))\big\}.$$

Since $M \models \forall x \in \{0,1\}^n Fml(f(x))$ by (8), we get $\llcorner A^M_{g,f}(n) \lrcorner \in Inf_n$ by Lemma 19 (c). This implies for all $g \in L_{\mathrm{PV}}$ and all standard $\ell > 0$ that M and hence $\mathbb{N}$ models

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

This is equivalent to (c).

(c $\Rightarrow$ a) Let f be a function as in (c), i.e. for all $g \in L_{\mathrm{PV}}$ and all $\epsilon > 0$ there are infinitely many $m \in \mathbb{N}$ such that $\Pr(neg(f(U_m)) \in \textsc{sat}) = 1$ and

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

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 $K(F^n_{\mathrm{PV}})$. We can assume that $\forall y Fml(f(y))$ holds in $\mathbb{N}$ and hence in M. So to get the madness property (8) it suffices to get $M \models \psi(n)$ for the formula $\psi(x)$ defined in (9).

Now, for $g \in L_{\mathrm{PV}}$ let $B_{g,f} : \mathbb{N} \to \mathbb{N}$ map $m \in \mathbb{N}$ to

$$B_{g,f}(m) := \ulcorner\{\omega \in \{0,1\}^m \mid \mathbb{N} \models |g(\omega)| \leqslant |f(\omega)| \wedge \neg Sat(g(\omega), f(\omega))\}\urcorner. \tag{10}$$

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

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

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

$$x \geqslant \ell \wedge \psi(x) \wedge \Pr(B_{g,f}(x)) < 1/\ell. \tag{11}$$

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 $(g_i(\omega), f(\omega)) \notin Sat$, 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 $\bigcup_{i \leqslant k} \llcorner B_{g_i,f}(m) \lrcorner \subseteq \llcorner B_{g,f}(m) \lrcorner$ and thus for all $i \leqslant k$:

$$\Pr(B_{g_i,f}(m)) \leqslant \Pr(B_{g,f}(m)). \tag{12}$$

Now, by assumption, for every ℓ, there are infinitely many $m \in \mathbb{N}$ such that $\mathbb{N} \models \psi(m)$ and $\Pr(B_{g,f}(m)) < 1/\ell$. So there is such $m \geqslant \ell$. By choice of ℓ and (12) we get $\Pr(B_{g_i,f}(m)) < 1/\ell \leqslant 1/\ell_i$ 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 $M \models \psi(n)$, hence f is mad in $K(F^n_{\mathrm{PV}})$. Furthermore, $\Pr^M(B_{g,f}(n)) < 1/\ell$ for all standard $\ell > 0$ and all $g \in L_{\mathrm{PV}}$. Hence, $\llcorner B_{g,f}(n) \lrcorner \in \mathrm{Inf}_n$ for all $g \in L_{\mathrm{PV}}$. This is Lemma 19 (b), so $f^M \upharpoonright \{0,1\}^n$ is a pseudo proof system in $K(F^n_{\mathrm{PV}})$.

Finally, we show that statement (b) holds assuming $\mathrm{NE} \cap \mathrm{coNE} \not\subseteq \mathrm{E}$ and pseudorandom generators exist. By Proposition 9, the first assumption gives a polynomial time computable hard sequences for $\mathbb{L}$. By Proposition 13, the second assumption gives an invertibly samplable probably hard sequence for $\mathbb{L}$. By optimality of $\mathbb{L}$, this sequence is probably hard for every SAT-solver.

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 [38]. Finally, we are grateful to the reviewers. Thanks to their comments we could substantially improve the presentation of our work.

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] 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.

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

[5] S. R. Buss. Bounded arithmetic, cryptography and complexity. Theoria 63: 147-167, 1997.

[6] 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.

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

[8] 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.

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

[10] S. A. Cook and D. Mitchell. Finding Hard Instances of the Satisfiability Problem: A Survey. DIMACS Series in Discrete Mathematics and Theoretical Computer Science 35: 1-17, 1997.

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

[12] 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.

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

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

[15] 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.

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

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

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

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

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

[21] 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.

[22] 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.

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

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

[25] 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.

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

[27] 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.

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

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

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

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

[32] 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.

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

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

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

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

[37] 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". Proceedings of the 8th Computer Science Symposium in Russia (CSR'13), LNCS 7913: 203-211, 2013.

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