Electronic Colloquium on Computational Complexity, Report No. 94 (2013)

On Non-automatizability in PAC-Semantics

Brendan Juba*
Harvard University
bjuba@alum.mit.edu

June 12, 2013

Abstract

We consider the proof search (*"automatizability"*) problem for integrated learning and reasoning, a problem modeling certain kinds of data mining and common sense reasoning [14]. In such a problem, the approximate validity (i.e., under Valiant's PAC-Semantics [24]) of an input query formula over a background probability distribution is verified using incomplete examples from the distribution; queries featuring a proof from some learnable premises are distinguished from queries that are falsified with moderately high probability under the distribution. The introduction of tolerance for some approximation and examples from the target distribution raise the possibility that this problem may be easier than classical automatizability in many circumstances. In particular, for certain restricted distributions and information-masking processes, the automatizability problem for resolution can be solved in quasipolynomial time [15]. Nevertheless, we argue here that the known cryptographic non-automatizability results [18, 8, 7] carry over to even the highly restricted kinds of distributions considered in that work.

1 Introduction

Learning is generally not an end unto itself—rather, it usually comprises one piece of a larger analysis or application. When one considers the learning task in the larger context of performing an analysis in data mining, for example, the corresponding combined algorithmic problem may be much easier than designing separate algorithms for the learning and for performing the analysis. An early form of this observation is due to Khardon and Roth [16], who showed how to answer $O(\log n)$-CNF queries against an unknown DNF using complete examples. Interestingly, Khardon and Roth's approach does not rely on bounding the complexity of proving the query. It succeeds whenever the query is entailed by the unknown DNF. Unfortunately, this feature limits the extent to which their approach can be applied in partial information settings: most such settings of any generality feature instances of theorem-proving as special cases, and indeed, they could only handle rather limited fragments [17]. Since background knowledge concerning attributes beyond the data are most naturally modeled as problems with incomplete information, this limits the applicability of such methods. An approach to this problem [14] that handles such partial information more generally (drawing on a rather general partial information learning model due to Michael [21]) proceeds as follows: one fixes (a tractable fragment of) a proof system and attempts to answer queries by testing for proofs of the query that may use as premises formulas that are learnable

*Supported by ONR grant number N000141210358.

ISSN 1433-8092

from the partial information. This approach is feasible for all "natural" tractable fragments (in the sense of Beame, Kautz, and Sabharwal [2]), and moreover, much like in Khardon and Roth's work, circumvents various learning-theoretic barriers. Moreover, in follow-up work [15], we observed that under a simpler partial information model due to Decatur and Gennaro [10], for a specific learning problem (that may still be hard under such partial information), resolution is quasi-automatizable. This is of interest as the best algorithms for resolution proof search [9, 5] are only guaranteed to run in time $2^{\tilde{O}(\sqrt{n})}$, and results of Alekhnovich and Razborov [1] suggest that resolution may not be (polynomial-time) automatizable. (Alekhnovich and Razborov further conjecture that resolution may not even be quasi-automatizable, in contrast to the special case of tree-like resolution.) In light of this it is natural to ask whether (and to what extent) any of the other negative results for proof search still apply—or, whether the setting more like Khardon and Roth's complete information setting, in which provability poses no constraint on our ability to answer queries. In this work, we show that the kind of cryptographic non-automatizability results established via the hardness of interpolation, first explored in the work of Krajíček and Pudlák [18], largely remain in effect. In fact, we will observe that these results already establish non-automatizabilty of the same proof systems in the more general "distribution-free" setting we first considered [14]. But moreover, we show here that they can also be easily adapted to establish non-automatizability of various Frege proof systems in the same restricted model under which resolution (*"RES"*) is quasi-automatizable.

In particular, Bonet, Pitassi, and Raz [8] essentially showed that answering queries using TC_0-Frege proofs is as hard as breaking the Diffie-Hellman key exchange protocol (i.e., as hard as factoring). Here, we find that even given incomplete examples under Decatur and Gennaro's model for the specific problem of unsupervised learning of parities under the uniform distribution, the same conclusion holds. Similarly, Bonet et al. [7] showed how to adapt the above work of Bonet, Pitassi, and Raz to establish the non-automatizability of depth-d Frege proofs, under an assumption that factoring cannot be performed in time 2^{n^ϵ} (i.e., for ϵ sufficiently small). This work follows by translating the TC_0-Frege proofs for small (polylogarithmic-size) keys to bounded-depth Frege proofs. We show that we can apply the same transformation to likewise obtain an analogous result for depth-d Frege. We obtain our results by noting that in the context of reasoning about a cryptosystem, examples from Decatur and Gennaro's partial information model correspond to a relatively benign form of leakage. Thus, some of the oldest ideas in leakage-resilient cryptography – namely, the use of a parity encoding of secret values [12] – suffice to reduce to a setting in which the examples are of no value. Thus, so long as a large parity formula is available in the formulas of the proof system (which is the case for TC_0-Frege) we can obtain the same results as when no examples are provided. Conveniently, the distribution produced by this transformation (for a fixed secret) is precisely a uniform distribution satisfying an unknown system of parity constraints, i.e., an instance of the precise learning problem for which RES was shown to be quasi-automatizable .

2 Preliminaries

2.1 PAC-Semantics

PAC-Semantics was originally introduced by Valiant [24] in order to capture logical reasoning from premises obtained by machine learning. The difficulty is that when an algorithm produces a formula ψ for predicting a label bit ℓ, the formula $[\psi(x) = \ell]$ is in general *not* a tautology. (Indeed, it may even be falsified with some small probability under new data drawn from the same source used for

training.) Nevertheless, if ψ is the rule produced by a PAC-learning algorithm [23], the formula $\varphi(x, \ell) = [\psi(x) = \ell]$ (probably) does possess the following weaker kind of "approximate" validity:

Definition 1 ($(1-\epsilon)$-valid) *We say that a formula φ is* $(1-\epsilon)$-valid *w.r.t. a background distribution D over Boolean assignments to free variables if* $\Pr_{x \in D}[\varphi(x) = \top] \geq 1 - \epsilon$.

Broadly speaking, the main algorithmic problem in PAC-Semantics is to certify formulas as $(1-\epsilon)$-valid on the basis of *examples* drawn from the background distribution. (Specifically, these will be *incomplete* examples, as we discuss in the next section.) In this work, we will distinguish between two families of such problems, paralleling a distinction in learning theory: when the background distribution D is completely arbitrary, we will refer to this as the *"distribution-free"* version of the problem, in contrast to when D is a member of some pre-specified family, which we will refer to as a *"restricted distribution"* problem. The family of distributions we will consider in this work is the following family of *"affine distributions":*

Definition 2 (Affine distribution) *A distribution D over Boolean assignments is an* affine distribution *if, when the Boolean values are interpreted as elements of $\mathbb{F}_2$ in the natural way, there exists a linear system $Ax = b$ such that D is the uniform distribution over solutions to the system.*

Affine distributions are an "unsupervised" generalization of the problem of learning parity formulas under the uniform distribution—one way of viewing such problems is that there is an unknown parity constraint over the example and label bits, and the examples are generated by choosing solutions to this equation uniformly at random. The difference is that in an affine distribution, there may be many constraints, and there is no distinguished label bit (this is the sense in which it is unsupervised).

We are particularly interested in restricted distribution problems concerning affine distributions because prior work [15] suggests that certifying $(1-\epsilon)$-validity using (general, DAG-like) resolution proofs for these problems may be easier than the classical proof search problem for resolution. In this work, we will show by contrast that cryptographic assumptions imply that certifying $(1-\epsilon)$-validity under the same restricted family of distributions remains intractable for some more powerful (Frege) proof systems.

2.2 Model of partial information

Reasoning in PAC-Semantics is only interesting in a partial information model: it is trivial to answer queries using complete examples, and such settings do not capture most interesting examples of applications in, e.g., data mining. The underlying partial information model we consider was proposed by Michael [21].

Definition 3 (Masking process) *A* mask *is a function $m : \{0,1\}^n \to \{0,1,*\}^n$ such that for every $x \in \{0,1\}^n$, whenever $m(x)_i \neq *$, $m(x)_i = x_i$. A* masking process *is a mask-valued random variable (i.e., a random function). We denote the distribution over* partial examples $\rho \in \{0,1,*\}^n$ *obtained by the result of applying a masking process M to an example drawn from a distribution D over $\{0,1\}^n$ by $M(D)$.*

Notice, this general definition permits arbitrary correlations between the hiding of indices by M in the partial example and the underlying example actually drawn from D. One cannot hope to

beat worst-case performance at reasoning when given only examples where M and D are arbitrary: Any arbitrary setting of the masked indices may have appeared with (conditional) probability 1, so one must rule out the existence of any falsifying assignments before it is safe to report that a query is (highly) valid, and such a case is easily seen to be essentially equivalent to classical reasoning. The problem becomes interesting when the masking process is sufficiently restricted to permit generalization beyond the revealed partial information; one natural example of such a process that appears in many other works (starting with a work of Decatur and Gennaro [10] in learning) is the following:

Definition 4 (μ-independent masking process) *For any $\mu \in (0,1)$, the* μ-independent masking process *(denoted M_μ) produces a mask m by tossing an independent μ-biased coin for each $i \in [n]$, putting $m(x)_i = x_i$ for all x if it comes up heads (with probability μ), and otherwise (with probability $1-\mu$) putting $m(x)_i \equiv *$.*

An important notion in proof complexity is that of a *restriction* of a formula; we can naturally interpret our partial examples as restrictions as follows:

Definition 5 (Restriction) *Given a formula $\varphi(x_1, \ldots, x_n)$ defined over linear threshold and parity connectives and a partial example $\rho \in \{0,1,*\}^n$ we define* the restriction of φ under ρ, *denoted $\varphi|_\rho$, as follows by induction on the construction of φ:*

- *For any Boolean constant b, $b|_\rho = b$.*
- *For any variable x_i, if $\rho_i = *$, then $x_i|_\rho = x_i$, and otherwise (for $\rho_i \in \{0,1\}$), $x_i|_\rho = \rho_i$.*
- *For a parity connective over $\psi_1, \ldots, \psi_k$, if $\ell \geq 1$ of the ψ_i (indexed by $i_1, \ldots i_\ell$) do not simplify to Boolean values under ρ, then (indexing the rest by $j_1, \ldots, j_{k-\ell}$)*

$$\oplus(\psi_1, \ldots, \psi_k)|_\rho = \oplus(\psi_{i_1}|_\rho, \ldots, \psi_{i_\ell}|_\rho, (\psi_{j_1}|_\rho \oplus \cdots \oplus \psi_{j_{k-\ell}}|_\rho))$$

  and otherwise it simplifies to a Boolean constant, $\oplus(\psi_1, \ldots, \psi_k)|_\rho = \psi_1|_\rho \oplus \cdots \oplus \psi_k|_\rho$.
- *A linear threshold connective $[\sum_{i=1}^k c_i \psi_i \geq b]$, $(c_1, \ldots, c_k, b \in \mathbb{Q})$ simplifies to 1 if*

$$\sum_{i:\psi_i|_\rho = 1} c_i + \sum_{i:\psi_i|_\rho \notin \{0,1\}} \min\{0, c_i\} \geq b$$

  simplifies to 0 if

$$\sum_{i:\psi_i|_\rho = 1} c_i + \sum_{i:\psi_i|_\rho \notin \{0,1\}} \max\{0, c_i\} < b$$

  and otherwise is given by

$$\left[\sum_{i:\psi_i|_\rho \notin \{0,1\}} c_i(\psi_i|_\rho) \geq \left(b - \sum_{i:\psi_i|_\rho = 1} c_i \right) \right].$$

That is, $\varphi|_\rho$ is a formula over the variables x_i such that $\rho_i = *$. We define AND and OR using the threshold connective and NOT using the parity connective in the natural way. We briefly remark that for the uniform distribution U_n, $M_\mu(U_n)$ is essentially the usual definition of a random restriction with parameter μ. We will be interested in the regime where μ is a constant (e.g., 1%).

Finally, we can define our central notion of interest:

Definition 6 (PAC-automatizability) *When we say that a proof system is* PAC-automatizable in time $T(N, 1/\delta, 1/\gamma)$*, we mean that for any fixed constant $\epsilon > 0$, there is an algorithm that is given φ, $\gamma, \delta > 0$, and N as input and obtains samples from $M(D)$ for a given distribution D and masking process M. This algorithm runs in time $T(N, 1/\gamma, 1/\delta)$ and with probability $1 - \delta$ distinguishes φ that are $(\epsilon+\gamma)$-valid from φ that have a refutation of size N in the system from additional premises $\psi_1, \ldots, \psi_k$ such that $\psi_1 \wedge \cdots \wedge \psi_k$ simplifies to true under partial examples drawn from $M(D)$ with probability at least $1 - \epsilon + \gamma$.*[1]

2.3 Bounded-depth Frege systems

We will use the standard bounded-depth Frege sequent systems of propositional logic defined by Maciel and Pitassi [19]. In these systems, each line is of the form $A_1, \ldots, A_s \to B_1, \ldots, B_t$ (that is, the conjunction of the A_i's implies the disjunction of the B_j's) where each A_i and B_j is a bounded-depth formula from the appropriate class; we will consider two such classes in this work, AC_0 and TC_0. These classes both use the connectives $\vee$, $\wedge$, and $\neg$, and TC_0 additionally features the $\oplus_b$ connectives that are true iff the number of inputs that are true modulo 2 is $b \in \{0, 1\}$, and the Th_k connective, a threshold connective that is true iff at least k of the inputs are true. All of these connectives (except $\neg$) have unbounded fan-in, and we define the depth of a formula to be the maximum depth of nesting of these connectives; the depth of a proof is then the maximum depth of any formula appearing in the proof. The size of the proof is the sum of the sizes of all of the formulas appearing in the proof. In the systems we consider, the depths will be bounded by some absolute constant (independent of the number of variables n) and the size of the proofs (and hence also their lengths) will be bounded by some polynomial in the number of variables.

The initial sequents (axioms) are the following

- $0 \to$ and $\to 1$
- The empty connective sequents $\to \wedge()$, $\vee() \to$, $\oplus_1() \to$, $\to \oplus_0()$, $\mathrm{Th}_k() \to$ for $k \geq 1$
- $A \to A$ for any formula A
- $\to \mathrm{Th}_0(A_1, \ldots, A_k)$ for any $A_1, \ldots, A_k$ and $k \geq 0$.

The rules of inference are the following

- (Weakening) From $\Gamma \to \Delta$, infer $\Gamma, A \to \Delta$ or $\Gamma \to \Delta, A$ for any formula A.
- (Contraction) From $\Gamma, A, A \to \Delta$ infer $\Gamma, A \to \Delta$; from $\Gamma \to \Delta, A, A$ infer $\Gamma \to \Delta, A$.
- (Permutation) From $A_1, \ldots, A_s \to B_1, \ldots, B_t$, infer $A_{\pi(1)}, \ldots, A_{\pi(s)} \to B_{\pi'(1)}, \ldots, B_{\pi'(t)}$ for any permutations π on $[s]$ and π' on $[t]$.
- (Cut) From $\Gamma, A \to \Delta$ and $\Gamma' \to A, \Delta'$ infer $\Gamma, \Gamma' \to \Delta, \Delta'$.
- (Negation-left) From $\Gamma \to A, \Delta$ infer $\neg A, \Gamma \to \Delta$.
- (Negation-right) From $A, \Gamma \to \Delta$ infer $\Gamma \to \neg A, \Delta$.
- (And-left) From $A_1, \wedge(A_2, \ldots, A_r), \Gamma \to \Delta$, infer $\wedge(A_1, \ldots, A_r), \Gamma \to \Delta$.
- (And-right) From $\Gamma \to A_1, \Delta$ and $\Gamma \to \wedge(A_2, \ldots, A_r), \Delta$, infer $\Gamma \to \wedge(A_1, \ldots, A_r), \Delta$.
- (Or-left) From $A_1, \Gamma \to \Delta$ and $\vee(A_2, \ldots, A_r), \Gamma \to \Delta$, infer $\vee(A_1, \ldots, A_r), \Gamma \to \Delta$.
- (Or-right) From $\Gamma \to A_1, \vee(A_2, \ldots, A_r), \Delta$, infer $\Gamma \to \vee(A_1, \ldots, A_r), \Delta$.
- (Mod-left) From $A_1, \oplus_{1-b}(A_2, \ldots, A_r), \Gamma \to \Delta$ and $\oplus_b(A_2, \ldots, A_r), \Gamma \to A_1, \Delta$ infer $\oplus_b(A_1, \ldots, A_r), \Gamma \to \Delta$.

[1] It turns out to be more natural and convenient to state this definition in terms of refutations of φ as we have done here, since this is how we will use the algorithms in most circumstances. We could equivalently (but less conveniently) have said that we were distinguishing when $\neg\varphi$ has a proof from when $\neg\varphi$ is *not* $(1 - \epsilon - \gamma)$-valid (where φ is the query actually provided to the algorithm).

- (Mod-right) From $A_1, \Gamma \to \oplus_{1-b}(A_2, \ldots, A_r), \Delta$ and $\Gamma \to A_1, \oplus_b(A_2, \ldots, A_r), \Delta$, infer $\Gamma \to \oplus_b(A_1, \ldots, A_r), \Delta$.
- (Threshold-left) From $\mathrm{Th}_k(A_2, \ldots, A_r), \Gamma \to \Delta$ and $A_1, \mathrm{Th}_{k-1}(A_2, \ldots, A_r), \Gamma \to \Delta$, infer $\mathrm{Th}_k(A_1, \ldots, A_r), \Gamma \to \Delta$.
- (Threshold-right) From $\Gamma \to A_1, \mathrm{Th}_k(A_2, \ldots, A_r), \Delta$ and $\Gamma \to \mathrm{Th}_{k-1}(A_2, \ldots, A_r), \Delta$ infer $\Gamma \to \mathrm{Th}_k(A_1, \ldots, A_r), \Delta$.

The main result of Bonet et al. [7] is essentially a translation from TC_0-Frege proofs to AC_0-Frege proofs:

Theorem 7 (Theorem 6.1 of Bonet et al. [7]) *Suppose that $\Gamma \to \Delta$ has a TC_0-Frege proof of size polynomial in n in which the threshold and parity connectives all have fan-in bounded by $O(\log^k n)$. Then there is an AC_0 formula equivalent to $\Gamma \to \Delta$ that is polynomial-time computable from $\Gamma \to \Delta$ and has an AC_0-Frege proof of size greater by a factor of at most $O(n^K)$ where K depends only on k.*

Actually, Bonet et al. give specific definitions of such threshold and parity connectives. They do not explicitly state or argue for the efficient computation of the transformation (of the conclusion $\Gamma \to \Delta$) but this is essentially immediate. This translation enables non-automatizability for (sufficiently simple) specific TC_0-Frege formulas to be carried over to non-automatizability for AC_0-Frege formulas.

2.3.1 Substitutions

A *substitution* is a mapping from formulas to formulas defined by its action on free variables, taking them to arbitrary propositional formulas. For a substitution θ and propositional formula φ, we typically denote the result of applying θ to φ by $\theta\varphi$. Since the rules of inference for Frege systems remain instances of the same rules under any substitution, the following (essentially standard) fact is easily established:

Proposition 8 *Let θ be any substitution taking variables to depth-d_1 formulas, and suppose that there is a depth-d_2 Frege proof of φ from $\{\psi_1, \ldots, \psi_k\}$. Then there is a depth-$(d_1 + d_2)$ Frege proof of $\theta\varphi$ from $\{\theta\psi_1, \ldots, \theta\psi_k\}$.*

In particular, we will be substituting formulas consisting of a parity connective over variables for the variables of the original formula. This increases the depth of a formula by one and increases the size by at most a factor of n' (where there are n' variables in the substitutions). It thus takes TC_0-Frege proofs to TC_0-Frege proofs.

2.4 The generalized Diffie-Hellman assumption and propositional encoding

The Diffie-Hellman key exchange scheme was one of the contributions of the seminal work by Diffie and Hellman [11]. Roughly speaking, over the group $\mathbb{Z}_p^*$ (for a possibly composite p) and quadratic residue g, the parties come to share a secret key $\sigma = g^{ab} \bmod p$ by publicly exchanging $g^a \bmod p$ and $g^b \bmod p$ for privately chosen secrets a and b. It is known that computing $g^{ab} \bmod p$ from the public information (g, g^a, g^b, and p) when p is the product of integers equal to 3 modulo 4 ("Blum integers") is as hard as factoring such integers [6, 20, 22], which is commonly presumed to be hard.

Bonet, Pitassi, and Raz [8] exhibited TC_0 formulas essentially asserting that either any desired ith bit of the secret g^{ab} is equal to 0 or is equal to 1, such that moreover the (contradictory) conjunction of these formulas for any i has a polynomial-size TC_0-Frege refutation. We will revisit their argument in more detail in the next section, but first we will give an overview of the formulas they use as these are somewhat nontrivial. We will review the variables used in particular, as the role played by these variables is crucial to our reductions.

Precisely, fixing an ith bit (we will follow them and only consider the least significant bit here) they give two formulas A_0 and A_1 respectively asserting that the given bit of g^{ab} is 0 or 1. A_0 and A_1 will actually use different sets of variables for the secret values a and b; in A_1 we will denote these values by variables c and d, respectively. The rest of the variables are shared between the two formulas.

These common variables include binary encodings of the n-bit integers p and g, as well as encodings of $g^{2^i} \bmod p$ for each $i \leq 2n$, and encodings of $p_i = i \cdot p$ for $i \leq n$. They also include binary encodings of $x = g^a \bmod p$, $y = g^b \bmod p$, and likewise $x^{2^i} \bmod p$ and $y^{2^i} \bmod p$ for $i \leq 2n$. In order to compute quotients and remainders, they further include values k_i and r_i for $i \leq n$ such that $0 \leq r_i \leq p$ and $2^i = p \cdot k_i + r_i$. We remark that the values of all of these variables can be computed from the public values in polynomial time.

The formulas perform the exponentiation of g by the secret values a and b (resp., c and d) by using an iterated product that is performed by an iterated sum in the exponent in the chinese remainder representation; conversion to and from the chinese remainder representation is performed in essence by iterated sums in modular arithmetic, with the isomorphisms between an additive representation (as $\mathbb{Z}_{q_i-1}$) and multiplicative representation of $\mathbb{Z}^*_{q_i}$ for the primes q_i used in the chinese remainder representation performed by table lookups. Naturally, these primes have size $O(\log n)$ so that these tables have polynomial size. Moreover, the primes (and generators) used for the representation only depend on the length of the integers n (not p or any of the other public or private values for the DH scheme). Thus, even a brute-force search for appropriate values can be performed in time polynomial in n, and these will be "hard-wired" into the formulas.

Given the iterated product, the formula A_0 is now given by the conjunction of formulas asserting for $i \leq n$, $2^i = p \cdot k_i + r_i$, $1 \leq r_i < p$, and $p_i = i \cdot p$ (to facilitate modular arithmetic); $\prod_i g^{2^i a_i} \bmod p = x$ and $\prod_i g^{2^i b_i} \bmod p = y$ (so a and b correspond to the private values reflected in the public values x and y); for every $j \leq n$, $\prod_i g^{2^{i+j} a_i} \bmod p = x^{2^j} \bmod p$ and $\prod_i g^{2^{i+j} b_i} \bmod p = y^{2^j} \bmod p$ (in support of the next step, where x^{2^j} and y^{2^j} are similarly publicly computable from x and y); and finally, $\prod_{i,j} g^{2^{i+j} a_i b_j} \bmod p$ ($= g^{ab} \bmod p$, the shared secret) is even. A_1 is similar, replacing a and b by c and d, and finally concluding that $\prod_{i,j} g^{2^{i+j} c_i d_i} \bmod p$ ($= g^{cd} \bmod p$) is odd.

3 Cryptographic non-automatizability in PAC-Semantics

We begin by observing that essentially *all* of the existing cryptographic non-automatizability results actually establish non-automatizability under (general, "distribution-free") PAC-Semantics. Concretely, let us recall the argument used by Bonet, Pitassi, and Raz [8] to establish that TC_0-Frege is not automatizable given the security of the Diffie-Hellman protocol (and hence, given that factoring is intractable). The main step in their argument can be summarized as follows:

Theorem 9 (Bonet, Pitassi, Raz [8]) *The TC_0 formula $A^i_0 \wedge A^i_1$ described in Section 2.4 (where A^1_0 defines "$g^a \bmod p = x$, $g^b \bmod p = y$, and the ith bit of $g^{ab} \bmod p$ is 0" and A^i_1 defines*

"$g^c \bmod p = x$, $g^d \bmod p = y$ and the ith bit of $g^{cd} \bmod p$ is 1") has a polynomial-size TC_0-Frege refutation. There is also an algorithm that given n and $i \leq n$, runs in polynomial-time and produces A_0^i and A_1^i as output.

In particular, by plugging in the known public values for the variables in these TC_0-formulas, the formula asserting e.g. that the ith bit is 0 when it is actually 1 has a polynomial-size TC_0-Frege refutation. Thus, the ability to decide whether it is A_0^i or A_1^i that has a polynomial-size TC_0-Frege refutation allows us to recover the ith bit of $g^{ab} \bmod p$, completely breaking the security of DH. Of course, a and b are (essentially) determined by g, p, $g^a \bmod p$ and $g^b \bmod p$. This trivially establishes that PAC-automatizability of TC_0-Frege for arbitrary distributions and masking processes is as hard as breaking DH:

Theorem 10 *Suppose TC_0-Frege is PAC-automatizable in time $T(N, 1/\gamma, 1/\delta)$ for all distributions and masking processes on formulas of size N for $T(N, 3, 1/\delta) \geq \Omega(N^k)$ for some sufficiently large constant k. Then for some polynomial P, there is an algorithm running in time $O(n \cdot T(P(n), 3, n/\delta))$ that recovers $g^{ab} \bmod p$ with probability $1 - \delta$ from any n-bit p, generator g of $\mathbb{Z}_p^*$, $g^a \bmod p$, and $g^b \bmod p$ where a and b are arbitrary.*

Proof: Let such p, g, $g^a \bmod p$, and $g^b \bmod p$ be given, and let a and b be their natural representatives (i.e., in the range $(0, \phi(p))$ where ϕ is Euler's ϕ-function). Let D be a distribution that puts *all* of its mass on this single assignment, and let M be the masking process that hides only the values of the (vectors of) variables encoding the binary representations of a, b, and their copies c and d. Note that samples from $M(D)$ correspond to a vector encoding the public values and masking the secret values. Now, for any ith bit of the secret $g^{ab} \bmod p$, Theorem 9 establishes that the formulas A_0^i and A_1^i can be generated in time $O(n^k)$ for some k and have a TC_0-Frege refutation (without any additional premises) of size $N = P(n)$ for some polynomial P. In particular, plugging in the public values, it follows that the formula A_b^i (where the ith bit is actually b) is satisfied by the secret values—i.e., is 1-valid where since $1 - \gamma > 1/2$, the algorithm with $\epsilon = 1/2$ must accept such a query. Moreover, the other formula must have a refutation of size $P(n)$; that is, this other formula is actually 0-valid, and so, since $\gamma < 1/2$, the algorithm must reject it when $\epsilon = 1/2$. Thus, by running our algorithm on A_0^i for $i = 1, \ldots, n$, we recover $g^{ab} \bmod p$ in time $O(n \cdot T(P(n), 3, n/\delta))$ as claimed. ■

We will strengthen this basic result in two stages. First, we will show that even if the distribution is restricted to an affine distribution and the masking process is M_μ for some constant μ, TC_0-Frege still cannot be PAC-automatized. We will then show that this result can be carried over to AC_0-Frege given a stronger assumption about the hardness of factoring, as in the work of Bonet et al. [7].

For the first stage, we will simply substitute a parity of a vector of $k(n)$ new variables for each secret Boolean variable, and let $D_n^{\oplus k(n)}$ be the distribution in which these parities are constrained to take the same values as the underlying secrets; then for any constant μ, we can simulate access to the result of $M_\mu(D_n^{\oplus k(n)})$ with negligibly small failure probability. This then provides a "leakage resilient" encoding of these secret values that is strong enough to withstand the relatively benign leakage provided by M_μ. (This use of the parity encoding to obtain leakage-resilience goes back to work by Ishai, Sahai, and Wagner [12].)

Lemma 11 *Let D_n and M be point distributions over n variables and masks on n variables, respectively. Let θ be a substitution that takes each variable x such that $M(x) = *$ to a parity of*

$k(n)$ new variables, $x_1, \ldots, x_{k(n)}$, and leaves all other variables fixed. Let $D^{\oplus k(n)}$ be the distribution over this new set of variables such that the variables left fixed by θ take the same value as in D_n and the new variables are uniformly distributed over values satisfying $x_1 \oplus \cdots \oplus x_{k(n)} = b$ where x took value b in D_n. Then for any p-valid formula φ under D_n, $\theta\varphi$ is also p-valid under $D^{\oplus k(n)}$. Moreover, there is a distribution that can be sampled in linear time given an example from $M(D_n)$ that is $1 - n\mu^{k(n)}$-statistically close to $M_\mu(D^{\oplus k(n)})$.

Proof: Since every assignment in the support of $D^{\oplus k(n)}$ satisfies $x_1 \oplus \cdots \oplus x_{k(n)} = b$ where the original variable x takes value b with probability 1 in D_n, it is immediate that $\theta\varphi$ takes the same (fixed) value under every assignment drawn from $D^{\oplus k(n)}$ as φ took under the sole assignment in the support of D_n. We can sample from $M_\mu(D^{\oplus k(n)})$ as follows: we construct an assignment to the new set of variables by first taking the known values from our partial assignment from $M(D_n)$, and then filling in the unknown (new) variables by tossing an unbiased coin for each variable. We denote this distribution by $\tilde{D}$. We then sample M_μ in the natural way, and output the result.

To see that $M_\mu(\tilde{D})$ is statistically close to $M_\mu(D^{\oplus k(n)})$, we merely note that for each masked x under M, whenever at least one of the new variables x_i is masked by M_μ, then the distribution induced by $M_\mu(D^{\oplus k(n)})$ is uniform over the unmasked x_i, i.e., identical to $M_\mu(\tilde{D})$. Therefore, as long as every block of masked variables has at least one masked variable, which happens with probability $1 - \mu^{k(n)}$, the distributions are identical. A union bound over the (at most n) blocks gives the desired bound. ■

Repeating the argument of Theorem 10, the desired strengthening is now almost immediate:

Theorem 12 *Suppose TC_0-Frege is PAC-automatizable for affine distributions under M_μ for constant μ in time $T(N, 1/\gamma, 1/\delta)$ for $T(N, 3, 1/\delta) \geq \Omega(N^c)$ and $T(N, 3, 1/\delta) < 2^{o(N \log \frac{1}{\delta})^{1/c})}$ for sufficiently large c. Then for some polynomial P, there is an algorithm running in time $\tilde{O}(nT(P(n) \cdot \log \frac{1}{\delta}, 3, \frac{n}{\delta}))$ that recovers $g^{ab} \bmod p$ with probability $1 - \delta$ from any n-bit p, generator g of $\mathbb{Z}_p^*$, $g^a \bmod p$, and $g^b \bmod p$, where a and b are arbitrary.*

Proof: We first note that by Proposition 8, if φ has a TC_0-Frege refutation of size $P(n)$, then for any $k(n)$, the substitution instance $\theta\varphi$ taking some variables to parities of $k(n)$ new variables has a TC_0-Frege refutation of the same length in which each formula has size greater by at most a factor of $k(n)$. If we take $k(n) = n \log \frac{n}{\delta}$ and c such that $P(n) < O(n^{c-2})$, then $k(n) \geq \log \frac{nT(k(n)P(n),3,n/\delta)}{\delta} \log \frac{1}{\mu}$ for sufficiently large n since $\log T(N, 3, n/\delta) < o((N \log \frac{n}{\delta})^{1/c}) = o(n \log \frac{n}{\delta})$ by assumption. We may then apply Lemma 11 to simulate samples from $M_\mu(D^{\oplus k(n)})$. Since the algorithm takes at most $T(N, 3, n/\delta)$ examples, and each example is good with probability at least $1 - \frac{\delta}{2nT(N,3,n/\delta)}$, the overall probability that the algorithm fails is greater by at most $\delta/2n$, for an overall probability of δ/n. We use this algorithm to recover each bit of $g^{ab} \bmod p$ as before with a total failure probability of at most $1 - \delta$ as needed. ■

Theorem 12 gives a range of non-automatizability bounds for a range of assumptions on the hardness of factoring: for example, we can conclude that TC_0-Frege cannot be PAC-automatized in (quasi-)polynomial time unless integer factoring has a (quasi-)polynomial-time algorithm, even for the restricted case of affine distributions under the independent masking process M_μ. This stands in contrast to RES, which is quasipolynomial time PAC-automatizable in this restricted case [15]. We can also obtain 2^{n^ϵ}-hardness of TC_0-Frege given 2^{n^ϵ}-hardness for integer factoring. Under this latter (strong) assumption, we can obtain a weaker conclusion for AC_0-Frege, following Bonet et al. [7, Theorem 7.1]:

Theorem 13 *Suppose* AC_0*-Frege is PAC-automatizable for affine distributions under* M_μ *for constant* μ *in time* $2^{\log^c(N/\delta)}$ *for some constant* c*. Then for all* $\eta > 0$*, there is an algorithm for integer factoring running in time* 2^{n^η}*.*

Proof: Theorem 9 establishes that the DH formula on m bits generally has a refutation of size polynomial in m; suppose we take $m = \log^{(c+1)/\eta} n$. Then $A_0^i \wedge A_1^i$ has a refutation of size polynomial in m. We let θ be the substitution taking each of the $O(m)$ (secret) variables to a parity of size $k(n) = 2C \log \frac{1}{\mu} \log^c(n/\delta)$ for some $\delta = m^{-s}/2$ (C and s given below). Then Proposition 8 implies that $\theta(A_0^i \wedge A_1^i)$ (still) has a TC_0-Frege refutation of size polynomial in m; Theorem 7 now implies that this TC_0-Frege refutation has an efficiently computable AC_0-Frege translation of size $O(n^K)$ where K depends only on c and η.

Noting that the asymptotic size of these proofs is independent of C, we can fix $C = (K^c + 1)$ so that the assumed algorithm for PAC-automatizing AC_0-Frege under M_μ and affine distributions decides such instances in time $2^{C \log^c n}$. Lemma 11 enables us to simulate its examples from $M_\mu(D^{\oplus k(n)})$ with failure probability $O(m\mu^{k(n)}) = O(\frac{1}{2^{2C \log^c(n/\delta))}}) < \delta \frac{1}{m 2^{C \log^c n}}$ (for sufficiently large n), and hence we can simulate $m2^{C \log^c n}$ samples from $M_\mu(D^{\oplus k(n)})$ with overall failure probability δ. Thus, following the approach of Theorem 10, we can recover $g^{ab} \bmod p$ from g, p, $g^a \bmod p$, and $g^b \bmod p$ for m-bit values by making m queries. Following the reduction of Biham, Boneh, and Reingold [6], this enables us to factor (Blum) integers with constant success probability overall in polynomial in m repetitions. Suppose this reduction makes $O(m^r)$ queries; then we take $s = r$, and find overall that we solve instances of size $m = (\log^{c+1} n)^{1/\eta}$ in time $O(m^r 2^{C \log^c n}) < 2^{\log^{c+1} n} = 2^{m^\eta}$ (for m sufficiently large). ■

4 Discussion and directions for future work

We remark that an analogue of our substitution of parities for variables appears elsewhere in proof complexity for a different purpose, increasing the (space) complexity of refutations (that is, "hardness amplification") in weaker proof systems [3, 4, 13].[2] In a sense, this "hardness amplifying" feature of the construction suggests why this same construction does not apply to RES, and therefore is consistent with the PAC quasi-automatizability of RES [15]: a formula hit with the parity substitution cannot have a "simple" refutation, where "simple" means low-space. (We warn that "simple" cannot be interpreted as "short" here, as Ben-Sasson and Nordström [4] in particular showed that formulas with proofs of linear length may still require quasi-linear space.)

The central question for future work is (still) whether or not proof systems such as RES are (quasi-)automatizable in the *distribution-free* setting. Indeed, if RES is not PAC-automatizable for general distributions, we require techniques that are rather different than the known techniques, which all trace back to the work of Alekhnovich and Razborov [1], and fundamentally rely on *estimating the length* of the shortest proof, a feature that is not provided in the natural promise formulation under PAC-Semantics—that is, in PAC-Semantics we only distinguish statements with short proofs from false statements, as opposed to distinguishing statements with short proofs from statements with only long proofs.

[2] Actually, the constructions there are slightly different: since these weaker proof systems do not actually feature the parity connective, one actually obtains a conjunction of formulas that is equivalent to the parity formula (featuring exponentially many such formulas in the size of the parity).

In any case, to the extent that we see proof systems such as RES featuring feasible interpolation turn out to be automatizable under PAC-Semantics and proof systems such as AC_0-Frege that do not have feasible interpolation remain hard, we are motivated to ask whether or not it is possible to characterize automatizability in PAC-Semantics (at least for affine distributions and the μ-independent masking process) in terms of interpolation. Of course, we likely need to strengthen the definition of interpolation for such a thing to be plausible: for example, rather than the usual "non-uniform" notion of *feasible* interpolation, we might demand *uniform* interpolation in which a uniform algorithm (given access to partial examples from the background distribution) decides which member of an input conjunction is false, promised that the conjunction has a small refutation in a given proof system. Note that PAC-automatizability still enables us to support such "uniform interpolation," following the existing reduction (originally due to Impagliazzo) reported in the work of Bonet, Pitassi, and Raz [8]. The question is whether there is a converse to this reduction.

Acknowledgements

The author would like to thank Les Valiant and Paul Beame for discussions that motivated this work.

References

[1] Michael Alekhnovich and Alexander A. Razborov. Resolution is not automatizable unless W[P] is tractable. *SIAM J. Comput.*, 38(4):1347–1363, 2008.

[2] Paul Beame, Henry Kautz, and Ashish Sabharwal. Towards understanding and harnessing the potential of clause learning. *JAIR*, 22:319–351, 2004.

[3] Eli Ben-Sasson. Size-space tradeoffs for resolution. *SIAM J. Comput.*, 38(6):2511–2525, 2009.

[4] Eli Ben-Sasson and Jakob Nordström. Understanding space in proof complexity: Separations and trade-offs via substitutions. In *Proc. 2nd ICS*, pages 16–29, 2011.

[5] Eli Ben-Sasson and Avi Wigderson. Short proofs are narrow – resolution made simple. *J. ACM*, 48(2):149–169, 2001.

[6] Eli Biham, Dan Boneh, and Omer Reingold. Breaking generalized Diffie-Hellman modulo a composite is no easier than factoring. *Inform. Process. Lett.*, 70:83–87, 1999.

[7] Maria Luisa Bonet, Carlos Domingo, Ricard Gavaldá, Alexis Maciel, and Toniann Pitassi. Non-automatizability of bounded-depth Frege proofs. *Comput. Complex.*, 13:47–68, 2004.

[8] Maria Luisa Bonet, Toniann Pitassi, and Ran Raz. On interpolation and automization for Frege proof systems. *SIAM J. Comput.*, 29(6):1939–1967, 2000.

[9] Matthew Clegg, Jeff Edmonds, and Russell Impagliazzo. Using the Gröbner basis algorithm to find proofs of unsatisfiability. In *Proc. 28th STOC*, pages 174–183, 1996.

[10] Scott E. Decatur and Rosario Gennaro. On learning from noisy and incomplete examples. In *Proc. 8th COLT*, pages 353–360, 1995.

[11] Whitfield Diffie and Martin Hellman. New directions in cryptography. *IEEE Trans. Inform. Theory*, 22:423–439, 1976.

[12] Yuval Ishai, Amit Sahai, and David Wagner. Private circuits: Securing hardware against probing attacks. In D. Boneh, editor, *Proc. CRYPTO 2003*, volume 2729 of *LNCS*, pages 463–481. Springer, Berlin, 2003.

[13] Jan Johannsen. Exponential separations in a hierarchy of clause learning proof systems. Technical Report TR13-072, ECCC, 2013.

[14] Brendan Juba. Implicit learning of common sense for reasoning. To appear in IJCAI'13, 2013. Preliminary version: *Learning implicitly in reasoning in PAC-Semantics*, arXiv:1209.0056v1 [cs.AI].

[15] Brendan Juba. PAC quasi-automatizability of resolution over restricted distributions. Technical Report 1304.4633 [cs.DS], arXiv, 2013.

[16] Roni Khardon and Dan Roth. Learning to reason. *J. ACM*, 44(5):697–725, 1997.

[17] Roni Khardon and Dan Roth. Learning to reason with a restricted view. *Machine Learning*, 35:95–116, 1999.

[18] Jan Krajíček and Pavel Pudlák. Some consequences of cryptographical conjectures for S_2^1 and EF. In D. Leivant, editor, *Logic and Computational Complexity*, volume 960 of *LNCS*, pages 210–220. Springer, Berlin, 1995.

[19] Alexis Maciel and Toniann Pitassi. Towards lower bounds for bounded-depth Frege proofs with modular connectives. In P. Beame and S. Buss, editors, *Proof Complexity and Feasible Arithmetics*, number 39 in DIMACS Ser. Discrete Math. Theoret. Comput. Sci., pages 195–227. AMS, 1998.

[20] Kevin McCurley. A key distribution system equivalent to factoring. *J. Cryptology*, 1:95–105, 1988.

[21] Loizos Michael. Partial observability and learnability. *Artificial Intelligence*, 174(11):639–669, 2010.

[22] Z. Shmuely. Composite Diffie-Hellman public-key generating systems are hard to break. Technical Report 356, Technion, Haifa, 1985.

[23] Leslie G. Valiant. A theory of the learnable. *Communications of the ACM*, 18(11):1134–1142, 1984.

[24] Leslie G. Valiant. Robust logics. *Artificial Intelligence*, 117:231–253, 2000.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de