# LEARNABILITY OF RELATIVELY QUANTIFIED GENERALIZED FORMULAS

ANDREI BULATOV, HUBIE CHEN, AND VÍCTOR DALMAU

Abstract. In this paper we study the learning complexity of a vast family of quantifed formulas called *Relatively Quantified Generalized Formulas*. This class of formulas is parameterized by a set of predicates, or basis, which contains its basic building blocks. We show a complete classification. We show that every such family of formulas is either polynomially learnable with equivalence queries or not polynomially predictable with membership queries under some cryptographic assumption. We also provide a simple criteria that allows to distinguish the learnable cases from those that are not.

## 1. Introduction

The problem of learning the models of an unknown formula has been widely studied. It is well-known that the general problem of learning an arbitrary formula is hard, even in the case of propositional formulas, under the usual learning models [3, 16]. This general non-learnability motivates the research program of distinguishing restricted classes of formulas that are learnable from those that are not learnable. Towards this end, a number of efficiently learnable classes of propositional formulas have been identified, for instance [1, 2, 6, 14].

An attractive framework of formula classes was studied by Schaefer [21]; he considered *generalized formulas*, defined as conjunctions of predicates, and considered restricted classes of generalized formulas where the predicates must come from a restricted predicate set, called a *basis*. Schaefer proved a now classic dichotomy theorem: for any basis over a two-element (boolean) domain, the problem of deciding if a generalized formula over the basis is true (that is, has a model) is either polynomial time tractable or NP-complete. Generalized formulas have, since Schaefer's theorem, been heavily studied in several different contexts [4, 7, 8, 9, 10, 17, 18] (see [11] for a monograph on boolean generalized formulas) ; they are widely acknowledged as a useful way of obtaining restricted versions of logical formulas in order to systematically study the distinction between tractability and intractability.

In the context of learning, Dalmau [12], along with Jeavons in [13], has studied the learnability of such formula classes. In particular, Dalmau [12] proved an analog of Schaefer's theorem in the setting of learnability: for every basis over a two-element domain, quantified generalized formulas over the basis are either learnable, or non-learnable. The case of bases over domains of size greater than two was considered by Dalmau and Jeavons [13]: they established general algebraic technology for studying learnability of generalized formulas, and gave some broad sufficient conditions for learnability.

In this paper, we continue this line of investigation by proving a classification theorem on the learnability of an extremely broad class of quantified generalized formulas. We consider *relatively quantified generalized formulas*, where quantification is allowed over any subset of

HUBIE CHEN IS A PHD. STUDENT AND HENCEFORTH QUALIFIES AS A CANDIDATE FOR THE E.M. GOLD AWARD the domain of a variable. We prove a full classification theorem on such relatively quantified formulas, showing that for *any* basis over a finite domain, the concept class consisting of relatively quantified formulas over the basis are either learnable in the exact model with equivalence queries, or non-learnable in the PAC-prediccion model with membership queries. Our classification theorem thus demonstrates an extremely sharp dichotomy on a diverse family of concept classes: none of the concept classes characterize any of the learning models intermediate between the two that we have just named, such as exact learning with equivalence and membership queries, or PAC learning (both with and without membership queries).

One of the primary tools we use to obtain our classification theorem is the algebraic approach to studying learnability of formulas [13]. This approach has demonstrated that the learnability of generalized formulas over a basis depends only on a set of operations called the *polymorphims* of the basis. Our main theorem states that relatively quantified formulas over a basis are learnable if and only if the basis has a particular type of polymorphism which we call a *generalized majority-minority (GMM) operation.* Our positive learnability results demonstrate that quantified formulas with a GMM polymorphism have a novel, compact representation which we call a *signature.* Our negative results use algebraic techniques to show that quantified formulas without a GMM polymorphism must have a two-element domain that is non-learnable by the dichotomy theorem of Dalmau [12]. We believe it remarkable that our negative results, which can be applied to bases over domains of arbitrary size, can ultimately be derived from the negative results for two-element bases.

## 2. Preliminaires

2.1. **Formulas.** We use $[n]$ to denote the set containing the first $n$ positive integers, that is, $\{1, \dots, n\}$. Let $V = \{x_1, x_2, \dots\}$ be an infinite contable set of variables and tet $A$ be a finite set called the *domain.* A $k$-ary *relation* ($k$ integer) on $A$ is a subset of $A^k$ where $k$ is called the *rank* or *arity* of the relation. The set $\{1, \dots, k\}$ is referred to as the set of *indices* or *coordinates* of $R$.

We use the term *formula* in a wide sense, to mean any well-formed formula, formed from variables, logical connectives, parentheses, relation symbols, and existential and universal quantifiers.

**Definition 1** ([13])**.** Let $A$ be any finite set and let $\Gamma = \{R_1, R_2, \dots\}$ be any set of relations over $A$, where each $R_i$ has rank $k_i$. ($R_i$ denotes both the relation and its symbol.)

The set of *quantified generalized formulas over the basis* $\Gamma$, denoted by $\forall\exists$-Form($\Gamma$), is the smallest set of formulas such that:

- **(a)** For all $R \in \Gamma$ of rank $k$, $R(y_1, \dots, y_k) \in \forall\exists\text{-Form}(\Gamma)$ where $y_i \in V$ for $1 \le i \le k$.
- **(b)** For all $\Phi, \Psi \in \forall\exists\text{-Form}(\Gamma)$, $\Phi \wedge \Psi \in \forall\exists\text{-Form}(\Gamma)$.
- **(c)** For all $\Phi \in \forall\exists\text{-Form}(\Gamma)$ and for all $x \in V$, $\exists x\Phi \in \forall\exists\text{-Form}(\Gamma)$.
- **(d)** For all $\Phi \in \forall\exists\text{-Form}(\Gamma)$ and for all $x \in V$, $\forall x\Phi \in \forall\exists\text{-Form}(\Gamma)$.

If we remove condition **(d)** in the previous definition we obtain a reduced class of formulas called *existentially quantified generalized formulas over the basis* $\Gamma$, denoted by $\exists$-Form($\Gamma$).

In this paper we focus our attention on a related family of formulas, called *relatively quantified generalized formulas*, and denoted $\forall\exists_{\mathrm{r}}$-Form($\Gamma$). In a $\forall\exists_{\mathrm{r}}$-Form($\Gamma$) formula, every occurrence of a quantifier $Q$, $Q \in \{\exists, \forall\}$, is followed by an expression of the form $(x \in B)$

where $B$ is a subset of $A$. The intended meaning of the expression $(x \in B)$ is to indicate that the quantification of $x$ is relative to the elements from $B$. That is, the semantics of $\exists(x \in B)\Phi$ is "there exists some $x$ in $B$ such that $\Phi$" whereas $\forall(x \in B)\Phi$ is interpreted as "for each $x \in B$, $\Phi$". The *relatively existentially quantified generalized formulas* $\exists_r$-Form($\Gamma$) are defined similarly. It is immediate to observe that the usual quantification can be simulated by just setting $B$ to $A$, and hence $\forall\exists$-Form($\Gamma$) is contained in $\forall\exists_r$-Form($\Gamma$) for each $\Gamma$. Indeed, the relationship between $\forall\exists$-Form($\Gamma$) and $\forall\exists_r$-Form($\Gamma$) is even closer, as one can easily observe the following:

**Proposition 1.** *Let $\Gamma$ be any finite set of relations containing every unary relation. Then $\forall\exists_r$-Form($\Gamma$) ($\exists_r$-Form($\Gamma$)) and $\forall\exists$-Form($\Gamma$) ($\exists$-Form($\Gamma$))) are equivalent.*

**Proof.** Clearly, $\forall\exists$-Form($\Gamma$) $\subseteq$ $\forall\exists_r$-Form($\Gamma$). We prove the converse by induction on the structure of a formula $\Phi$. The base case of induction, $\Phi \in \Gamma$, is obvious. Now, if $\Phi, \Psi \in \forall\exists$-Form($\Gamma$), then $\Phi \wedge \Psi \in \forall\exists$-Form($\Gamma$); $\exists(x \in B)\Phi$ is equivalent to $\exists x(\Phi \wedge B(x))$, where $B$ is the predicate interpreted as the unary relation $B$; and $\forall(x \in B)\Phi$, $B = \{b_1, \ldots, b_k\}$, is equivalent to

$$(\exists x\Phi \wedge b_1(x)) \wedge \ldots \wedge (\exists x\Phi \wedge b_k(x)),$$

where the $b_i(x)$ is a predicate interpreted as the unary relation $\{(b_i)\}$. ■

In our analysis the size of our formulas will be relevant. We shall denote the size $|\Phi|$ of a formula $\Phi$ as the lenght of the string that encodes it (in any reasonable encoding). Let $F, G$ be two families of formulas. We shall say that $F$ is polynomially contained in $G$ if there exists a polynomial $p$ such that for each formula $\Phi$ in $F$ there exists an equivalent formula $\Psi$ in $G$ such that $|\Psi| \leq p(|\Phi|)$. If furthermore $G$ is polynomially contained in $F$ then we say that $F$ and $G$ are polynomially equivalent. By looking at Proposition 1 with this definitions on hand we can easily infer that for each $\Gamma$, $\forall\exists$-Form($\Gamma$) is polynomially contained in $\forall\exists_r$-Form($\Gamma$). Furthermore, if $\Gamma$ contains all unary relations then $\exists$-Form($\Gamma$) and $\exists_r$-Form($\Gamma$) are polynomially equivalent.

Each formula $\Phi$ defines a relation $R_\Phi$ if we apply the usual semantics of first-order logic, and the variables are taken in lexicographical order. More formally, let $\Phi$ be a formula over the free variables $x_{i_1}, x_{i_2}, \ldots, x_{i_m}$ where $i_1 < i_2 < \cdots < i_m$, then

$$R_\Phi = \{\langle a_1, a_2, \ldots, a_m\rangle : x_{i_j} = a_j \ (1 \leq j \leq m) \text{ satisfies } \Phi\}$$

2.2. **Learning Preliminaries.** We use two models of learning, both of them fairly standard: the model of exact learning with equivalence queries defined by Angluin [1] and the model of PAC-prediction with membership queries as defined by Angluin and Kharitonov [3]. We have chosen this models because each of them represents one opposite extreme in the spectrum of usual learning models. Consequently, by proving our positive results in the model of exact learning with equivalence queries and our negative learnability results in the model of PAC-prediction with membership queries we obtain results that are also valid for the many learning models lying in between, such as the PAC model [23], exact learning with equivalence and membership queries [1], or the PAE model [5], all of them among the most studied in computational learning theory.

In our setting a concept $c$ from a concept class $C$ is merely a subset of a domain space $X$, paired with an enconding of $c$. A learning algorithm $A$ has the goal of identifying a target concept $t$. It may make any number or queries or requests to a teacher or oracle, who is commited to answer according to $t$, altough not necessarily in a collaborative way. In the Exact model with equivalence queries [1] the learner supplies the oracle with a hypothesis $h$ and the oracle either says equivalent or returns a counterexample $x \in h \triangle t$. If the hypothesis provided, $h$, does not belong to the concept class $C$ then we speak of *improper* equivalence queries. We say that $A$ learns a concept class $C$ if for every target concept $t$, $A$ halts with an output concept $v$ equivalent to $c$. $A$ runs in polynomial time if its running time is bounded by a polynomial on the size of the representation of $t$ and the size of the largest counterexample.

In the PAC-prediction with membership queries [3] the oracle also holds a probability distribution $D$ on the space of examples $X$. The learner is allowed to make two kinds of oracle calls. In a membership query the learner supplies some $x \in X$ and the oracle answers "yes" if $x \in t$ and "no" otherwise. In a request for a random classified example the oracle returns a pair $(x, b)$ where $x$ is choosen according to $D$ and $b$ is 1 if $x \in t$ and 0 otherwise. The learning algorithm $A$ must eventually make one request for an element to predict and halt with an output of 1 or 0. We say that $A$ leans a concept class $C$ if for every target concept $t$ and for every distribution probability $D$ on $S$, $A$ predicts the correct classification of the requested example with probability at least $1 - \epsilon$, for some accuracy bound $\epsilon$. We say that $A$ runs in polynomial time if its running time is bounded by a polynomial on the size of the representation of $t$, the size of the larges counterexample, and $1/\epsilon$. We shall say that a concept class $C$ is polynomially learnable in any of the models if there exists a learning algorithm $A$ that learns $C$ and runs in polynomial time.

We use families of formulas as concept clases. The concept represented by a formula $\Phi$ is the set of all assignments (over the free variables of $\Phi$) that satisfy the formula. We shall use throughout the paper the following observation: Let $F_1$ and $F_2$ be families of formulas such that $F_1$ is polynomially contained in $F_2$. Then if $F_2$ is polynomially learnable in any of the previous models then so it $F_1$.

2.3. **Polymorphisms.** Our main tool is polymorphisms. Every set of relations, $\Gamma$, on a set $A$ has a collection of associated operations on $A$. An $n$-ary operation $f$ *preserves* an $m$-ary relation $R$ (or $f$ is a *polymorphism* of $R$, or $R$ is *invariant* under $f$) if, for any $\mathbf{a_1} = (a_{11}, \ldots, a_{m1}), \ldots, \mathbf{a_n} = (a_{1n}, \ldots, a_{mn}) \in R$, the tuple $f(\mathbf{a_1}, \ldots, \mathbf{a_n})$ defined as $(f(a_{11}, \ldots, a_{1n}), \ldots, f(a_{m1}, \ldots, a_{mn}))$ belongs to $R$. If $f$ preserves every relation from the set $\Gamma$, we say that $f$ is a polymorphism of $\Gamma$. The set of all polymorphisms of $\Gamma$ is denoted by $\mathrm{Pol}(\Gamma)$.

**Example 1.** Let $R$ be the solution space of a system of linear equations over a field $F$. Then the *affine* operation $m(x, y, z) = x - y + z$ is a polymorphism of $R$. Indeed, let $A \cdot \mathbf{x} = \mathbf{b}$ be the system defining $R$, and $\mathbf{x}, \mathbf{y}, \mathbf{z} \in R$. Then

$$A \cdot m(\mathbf{x}, \mathbf{y}, \mathbf{z}) = A \cdot (\mathbf{x} - \mathbf{y} + \mathbf{z}) = A \cdot \mathbf{x} - A \cdot \mathbf{y} + A \cdot \mathbf{z} = \mathbf{b}.$$

In fact, the converse can also be shown: if $R$ is invariant under $m$ then it is the solution space of a certain system of linear equations.

The next theorem amounts to say that the learnability and non-learnability of a set of formulas can be expressed in terms of polymorphisms.

**Theorem 1** ([13]). *Let $\Gamma$, $\Gamma_0$ be sets of relations on a finite set $A$ and $\Gamma_0$ is finite. Then:*

(1) *If $\mathrm{Pol}(\Gamma) \subseteq \mathrm{Pol}(\Gamma_0)$ and $\exists$-Form$(\Gamma)$ is polynomially predictable with membership queries then so is $\exists$-Form$(\Gamma_0)$.*

(2) *If* $\mathrm{Pol}(\Gamma) \subseteq \mathrm{Pol}(\Gamma_0)$ *and* $\forall\exists$-Form$(\Gamma)$ *is polynomially predictable with membership queries then so is* $\forall\exists$-Form$(\Gamma_0)$.

In the same paper, it was shown that two types of polymorphisms guarantee the learnability of a class of formulas. An operation $f : A^k \to A$ with $k \geq 3$ is said to be a *near-unanimity* operation if

$$f(x, y, \ldots, y) = \cdots = f(y, \ldots, y, x) = y$$

for all $x, y \in A$. In this paper, it will be convenient for us to call such an operation *generalized majority*. The operation $x \cdot y^{-1} \cdot z$, where $\cdot$ and $^{-1}$ are the operations of a certain finite group, is called a *coset generating* operation. Note that every affine operation is a coset generating operation, in this case $G$ is Abelian.

**Theorem 2** ([13])**.** *Let* $\Gamma$ *be a set of relations on a finite set* $A$*. If* $\Gamma$ *has a polymorphism* $f$ *such that* $f$ *is either a near-unanimity or coset generating operation, then* $\forall\exists$-Form$(\Gamma)$ *is polynomially exactly learnable with equivalence queries.*

## 3. Main Result

The class of operations we introduce here generalizes both near-unanimity and coset generating operations.

**Definition 2.** An operation $f : A^k \to A$ with $k$ odd and $k \geq 3$ is a *generalized majority-minority (GMM) operation* if for all $a, b \in A$, either

$$f(x, y, \ldots, y, y) = f(y, x, \ldots, y, y) = \cdots = f(y, y, \ldots, y, x) = y, \quad \text{for } x, y \in \{a, b\} \tag{1}$$

or

$$f(x, y, \ldots, y, y) = f(y, x, \ldots, y, y) = \cdots = f(y, y, \ldots, y, x) = x, \quad \text{for } x, y \in \{a, b\}. \tag{2}$$

A GMM operation $f$ is called *conservative* if, for any $x_1, \ldots, x_k$, $f(x_1, \ldots, x_k) \in \{x_1, \ldots, x_k\}$.

Let us fix a GMM operation on a set $A$. A pair $a, b \in A$ is said to be a *majority* pair if $f$ on $a, b$ satisfies (1). It is said to be a *minority* pair if $f$ satisfies the (2).

**Theorem 3.** *Let* $A$ *be a finite set and let* $\Gamma$ *be a finite set of relations on* $A$*. Then* $\forall\exists_{\mathrm{r}}$-Form$(\Gamma)$ *is polynomially exactly learnable with (improper) equivalence queries if and only if* $\Gamma$ *is invariant under a conservative GMM operation. Otherwise* $\exists_{\mathrm{r}}$-Form$(\Gamma)$ *is not polynomially predictable with membership queries under the assumption that public key encryption systems secure against chosen ciphertext attack exists.*

As it has been mentioned in Section 2, for each $\Gamma$, $\forall\exists$-Form$(\Gamma)$ is polynomially contained in $\forall\exists_{\mathrm{r}}$-Form$(\Gamma)$. Consequently, the positive learnability results also hold for the class of generalized quantified formulas, i.e., $\forall\exists$-Form$(\Gamma)$ is polynomially exactly learnable if $\Gamma$ is invariant under a conservative GMM operation. However, the negative learnability results do not hold in general for the class of generalized quantified formulas. We can however extend the negative results to $\exists$-Form$(\Gamma)$ by imposing some assumptions on $\Gamma$. Since for each $\Gamma$ containing every unary relation, we have that $\exists_{\mathrm{r}}$-Form$(\Gamma)$ is polynomially contained in $\exists$-Form$(\Gamma)$ we can infer that, for every such $\Gamma$, if $\exists$-Form$(\Gamma)$ is polynomially predictable with membership queries, then $\Gamma$ must be invariant under a conservative GMM operation. In fact we are able to strenghten this result by only requiring that the basis $\Gamma$ contains all unary relations of cardinality at most 2. In this case we can remove the condition of conservativity of a GMM operation.

**Theorem 4.** *Let $A$ be a finite set and let $\Gamma$ be a set of relations all unary relations of cardinality at most 2 such that $\exists$-Form($\Gamma$) is polynomially predictable with membership queries. If there exists public key encryption systems secure against chosen ciphertext attack then $\Gamma$ must be invariant under a GMM operation.*

Thus we have

**Corollary 1.** *Let $A$ be a finite set and let $\Gamma$ be a finite set of relations on $A$ containing all unary relations of cardinality at most 2. Then $\forall\exists$-Form($\Gamma$) is polynomially exactly learnable with (improper) equivalence queries if and only if $\Gamma$ is invariant under a GMM operation. Otherwise $\exists$-Form($\Gamma$) is not polynomially predictable with membership queries under the assumption that public key encryption systems secure against chosen ciphertext attack exists.*

The positive learnability results are shown in Section 4. Due to space restriction we could not include the negative learnability results in the body of the paper. They are shown in the appendix.

## 4. Positive Learnability Results

Let $\Gamma$ be a set of relations on a finite set $A$ and $f : A^k \to A$ its GMM polymorphism, $k$ odd. In this section we shall show that for $\forall\exists_{\mathrm{r}}$-Form($\Gamma$) is polynomially learnable in the exact model with equivalence queries. Our learning algorithm will make use of an alternative and compact representation for relatively quantified generalized formulas called *signature*. The equivalence queries and the output of the learning algorithm will be signatures.

This section contains three subsections. In the first subection we introduce signatures and we show that for every relation concept expressed by a formula in $R \in \forall\exists_{\mathrm{r}}$-Form($\Gamma$) there exists a succint representation (of the size bounded by a polynomial in the arity of $R$) in the form of a signature. In the second subsection we prove that signatures are polynomially evaluable, that is, we can decide in polynomial time whether given a signature and a tuple $\mathbf{a}$, whether $\mathbf{a}$ belongs to the relation represented by the signature. Finally, in the third subsection, we present an algorithm that polynomially learns signatures with equivalence queries.

### 4.1. Signatures.

Our approach relies in the following result.

**Lemma 1.** *Let $\Gamma$ be a set of relations on a finite set $A$ and $f$ its GMM polymorphism.*

*1. Every relation represented by a formula from $\forall\exists$-Form($\Gamma$) is invariant under $f$.*

*2. If $f$ is conservative, then every relation represented by a formula from $\forall\exists_{\mathrm{r}}$-Form($\Gamma$) is invariant under $f$.*

Its proof is an easy exercice using Proposition 1 (see also [4]). Consequenlty from now on we shall view concepts as relations invariant under $f$. Signatures will be a compact representation of such relations.

Let $R$ be an $n$-ary relation invariant under $f$. Recall that, for a set $I = \{i_1, \dots, i_m\} \subseteq \{1, \dots, n\}$, $1 \le i_1 < \dots < i_m \le n$ the projection of a tuple $\mathbf{a} = (a_1, \dots, a_n) \in R$ is the tuple $(a_{i_1}, \dots, a_{i_m})$, and the projection of $R$ is the relation $\mathrm{pr}_I = \{\mathrm{pr}_I\, \mathbf{a} \mid \mathbf{a} \in R\}$.

Recall that $f$, $k$ and $a$ are fixed. A signature of arity $n$ is a triple (Pro, Rec, Wit) where

- Pro is a collection of pairs $(I, \mathbf{a})$ where $I$ is a subset of $[n]$ of cardinality at most $k-1$ and $\mathbf{a}$ is a tuple on $A$ whose arity matches the cardinality of $I$.

- Rec is a collection of triples $(i, a, b)$ where $1 \leq i \leq n$ and $a, b$ is a minority pair.
- Wit is a collection of $n$-ary tuples on $A$.

Furthermore Pro, Rec and Wit must satisfy the following conditions:

- For every $(I, \mathbf{a}) \in \mathrm{Pro}$, Wit contains some tuple $\mathbf{x}$ such that $\mathrm{pr}_I\, \mathbf{x} = \mathbf{a}$.
- For every $(i, a, b) \in \mathrm{Rec}$, Wit contains some tuples $\mathbf{a}, \mathbf{b} \in R$ are such that $\mathrm{pr}_{\{1,\ldots,i-1\}}\, \mathbf{a} = \mathrm{pr}_{\{1,\ldots,i-1\}}\, \mathbf{b}$, $\mathrm{pr}_i\, \mathbf{a} = a$, and $\mathrm{pr}_i\, \mathbf{b} = b$.

We shall denote by $\mathrm{Sig}_n$ the set of all signatures of arity $n$ and by Sig the set $\bigcup_{n\geq 0} \mathrm{Sig}_n$.

Let $R$ be an $n$-ary relation and let (Pro, Rec, Wit) be a signature. We say that (Pro, Rec, Wit) represents $R$, ((Pro, Rec, Wit) is a signature of $R$) if the three following conditions are satisfied:

- For every $1 \leq k' \leq k$, for every $I \subseteq [n]$ such that $|I| \leq k' - 1$ and for every $(k'-1)$-ary typle $\mathbf{a}$ on $A$, $(I, \mathbf{a}) \in \mathrm{Pro}$ if and only if $\mathbf{a} \in \mathrm{pr}_I\, R$.
- For every $1 \leq i \leq n$ and for every minority pair $a, b \in A$, $(i, a, b) \in \mathrm{Rec}$ if and only if there are $\mathbf{a}, \mathbf{b} \in R$ such that $\mathrm{pr}_{\{1,\ldots,i-1\}}\, \mathbf{a} = \mathrm{pr}_{\{1,\ldots,i-1\}}\, \mathbf{b}$, $\mathrm{pr}_i\, \mathbf{a} = a$, and $\mathrm{pr}_i\, \mathbf{b} = b$.
- Wit is contained in $R$.

For a set of tuples $T$, we denote by $\langle T \rangle$ the smallest relation invariant under $f$ and containing $T$. We refer to this relation as to the *relation generated by $T$*.

**Theorem 5.** *Let $R$ and $S$ be $n$-ary relations on $A$ invariant under $f$ with signatures* $(\mathrm{Pro}_R, \mathrm{Rec}_R, \mathrm{Wit}_R)$ *and* $(\mathrm{Pro}_S, \mathrm{Rec}_S, \mathrm{Wit}_S)$. *If the signatures are identical then $R = S$.*

**Proof.**

We start with an auxiliary statement.

Claim. If $S \subseteq R$ then $R = S$.

We shall show that $R \subseteq S$. In particular, we shall show by induction on $i$ that $\mathrm{pr}_{\{1,\ldots,i\}}\, R \subseteq \mathrm{pr}_{\{1,\ldots,i\}}\, S$. Take $i \leq n$. If $i \leq k-1$ then the required inclusion easily follows from $\mathrm{Pro}_R = \mathrm{Pro}_S$. So let $i \geq k$ and $R' = \mathrm{pr}_{\{1,\ldots,i\}}\, R$, $S' = \mathrm{pr}_{\{1,\ldots,i\}}\, S$ and $\mathbf{a} = (a_1, \ldots, a_i) \in R'$. By induction hypothesis, for some $b_i$, the tuple $(a_1, \ldots, a_{i-1}, b_i) = \mathbf{a}'$ belongs to $S'$.

We consider two cases.

*Case 1.* $\{a_i, b_i\}$ is majority.

In this case we show that, for every $I \subseteq \{1, \ldots, i\}$, $\mathrm{pr}_I\, \mathbf{a} \in \mathrm{pr}_I\, S'$. We show it by induction on the cardinality $m$ of $I$. The result is true for $m \leq k-1$ due to the fact that $\mathrm{Pro}_R = \mathrm{Pro}_S$. It is also true for every set $I$ that does not contain $i$, since $\mathbf{a}' \in S'$ certifies it. Thus let $I = \{j_1, \ldots, j_m\}$ be any set of indices $1 \leq j_1 < j_2 < \cdots < j_m = i$ with $m \geq k$ and also let $S'' = \mathrm{pr}_I\, S'$ and $\mathbf{a}^* = \mathrm{pr}_I\, \mathbf{a}$. To simplify the notation let us denote $\mathbf{a}^* = (c_1, \ldots, c_m)$. By induction typothesis, $S''$ contains the tuples $\mathbf{d_1} = (d_1, c_2, \ldots, c_m)$, $\mathbf{d_2} = (c_1, d_2, c_3, \ldots, c_m)$, $\ldots$, $\mathbf{d_m} = (c_1, \ldots, c_{m-1}, d_m)$ for some $d_1, \ldots, d_m \in A$. If for some $i$, $c_i = d_i$ then we are done. Otherwise, we can assume that $d_m = b_i$ and $c_m = a_i$ and henceforth $\{d_m, c_m\}$ is majority. If for some $j$, the pair $\{d_j, c_j\}$ is minority then we are done, because $f(\mathbf{d_j}, \mathbf{d_j}, \ldots, \mathbf{d_j}, \mathbf{d_m}) = \mathbf{a}^*$. Otherwise, $\{d_j, c_j\}$ is majority for any $j$. In this case we have $f(\mathbf{d_1}, \ldots, \mathbf{d_k}) = \mathbf{a}^*$.

*Case 2.* $\{a_i, b_i\}$ is minority.

Since $\mathbf{a}' \in S'$ and $S \subseteq R$, we have $\mathbf{a}' \in R'$. Furthermore, since $\mathbf{a} \in R'$, we can conclude that $(i, a_i, b_i)$ belongs to $\mathrm{Rec}_R$ and hence to $\mathrm{Rec}_S$. Consequently, there are $c_1, \ldots, c_{i-1}$, such that $\mathbf{c} = (c_1, \ldots, c_{i-1}, a_i)$ and $\mathbf{c}' = (c_1, \ldots, c_{i-1}, b_i)$ are in $S'$. The tuples $\mathbf{e} = (d_1, \ldots, d_{i-1}, a_i) =$

$f(\mathbf{a}', \mathbf{a}', \ldots, \mathbf{a}', \mathbf{c})$ and $\mathbf{d}' = (d_1, \ldots, d_{i-1}, b_i) = f(\mathbf{a}', \mathbf{a}', \ldots, \mathbf{a}', \mathbf{c}')$, belong to $S'$. If $\{c_j, a_j\}$ is majority for a certain $j$, then $d_j = a_j$. Therefore, if $d_j \neq a_j$ then $\{d_j, a_j\}$ is minority. Thus we have $f(\mathbf{d}, \mathbf{d}', \ldots, \mathbf{d}', \mathbf{a}') = \mathbf{a}$ belongs to $S'$ and we are done.

Let $T = \langle \mathrm{Wit}_R \rangle$ ( and $= \langle \mathrm{Wit}_S \rangle$). Observe that $(\mathrm{Pro}_S, \mathrm{Rec}_S, \mathrm{Wit}_S)$ is a signature of $T$. Since $T \subseteq S$ and $T$ and $S$ have the same signature, we have $S = T$. Similarly $R = T$ and we are done. ■

Consequently, a signature represents at most one relation invariant under $f$ (it is possible however that a given signature does not represent any relation). Notice that the size of Pro, Rec, Wit is allways bounded by a polynomial in $n$. (Recall that $f$ and hence $k$ and $A$ are fixed).

4.2. **Signatures are polynomially evaluable.** Our learning algorithm will use signatures as its concept class. That is, both the equivalence queries and the output of the algorithm will be signatures. Consequently, in order to make sure that we can use the output of our algorithm in order to classify examples we must show that signatures are *polynomially evaluable*, i.e, we can define in polynomial time, given a tuple $\mathbf{a}$ and a signature $s$, whether the tuple $\mathbf{a}$ belongs to the relation represented by the signature $s$.

Let $R$ be an $n$-ary relation, let $b_1, \ldots, b_m \in A$, and let $s = (\mathrm{Pro}_R, \mathrm{Rec}_R, \mathrm{Wit}_R)$ be a signature of $R$. By $R_{a_1,\ldots,a_m}$ we denote the relation

$$\{(b_1, \ldots, b_{n-m}) \mid (a_1, \ldots, a_m, b_1, \ldots, b_{n-m}) \in R\}$$

A signature $s_{a_1,\ldots,a_m}$ of $R_{a_1,\ldots,a_m}$ can be efficiently computed as the following theorem shows.

**Theorem 6.** *Let $R$ be an $n$-ary relation invariant under $f$ and $a_1, \ldots, a_m \in A$. There exists an algorithm that, given a signature $s = (\mathrm{Pro}_R, \mathrm{Rec}_R, \mathrm{Wit}_R)$ of $R$ computes a signature $s_{a_1,\ldots,a_m}$ of $R_{a_1,\ldots,a_m}$ with running time polynomial on $n$.*

**Proof.** Let $d \in A$. We shall show an algorithm that contructs a signature $(\mathrm{Pro}_{R_d}, \mathrm{Rec}_{R_d}, \mathrm{Wit}_{R_d})$ of $R_d$. Clearly, $\mathrm{Pro}_{R_d} \subseteq \mathrm{Pro}_R$.

The algorithm has two phases. In the first phase, it computes $\mathrm{Pro}_{R_d}$ and the elements of $\mathrm{Wit}_{R_d}$ witnessing the tuples from $\mathrm{Pro}_{R_d}$. Take $(I, \mathbf{a}) \in \mathrm{Pro}_R$. Let $J = \{1\} \cup \{i+1 : i \in I\}$ and $S = \langle \mathrm{pr}_J \mathrm{Wit}_R \rangle$; note that $S$ is a relation of arity at most $k$. It is not difficult to see that $S$ can be computed efficiently since the maximum possible total number of tuples in $S$ is $|A|^k$. If $(d, \mathbf{a}) \notin S$, then $(I, \mathbf{a}) \notin \mathrm{Pro}_{R_d}$. Otherwise, we can find easily a tuple $\mathbf{b}$ in the relation generated by $\mathrm{Wit}_R$ such that $\mathrm{pr}_J \mathbf{b} = (d, \mathbf{a})$. We include this tuple into $\mathrm{Wit}_{R_d}$.

In the second phase the algorithm computes $\mathrm{Rec}_{R_d}$ and the elements of $\mathrm{Wit}_{R_d}$ witnessing the tuples from $\mathrm{Rec}_{R_d}$. If $(i+1, a, b) \notin \mathrm{Rec}_R$ then $(i, a, b) \notin \mathrm{Rec}_{R_d}$. Otherwise we do the following. Let $I = \{i\}$ be the set of indices containing exactly $i$. If $(I, a) \notin \mathrm{Pro}_{R_d}$ then $(i, a, b) \notin \mathrm{Rec}_{R_d}$. Otherwise, let $\mathbf{e}' \in \mathrm{Wit}_{R_d}$ be the tuple with $a = \mathrm{pr}_i \mathbf{e}'$ that witnesses it. The tuple $\mathbf{e} = (d, \mathbf{e}')$ belongs to $R$, $\mathrm{pr}_1 \mathbf{e} = d$, and $\mathrm{pr}_{i+1} \mathbf{e} = a$. Since $(i+1, a, b) \in \mathrm{Rec}_R$, there are $\mathbf{a}', \mathbf{b}' \in \mathrm{Wit}_R \subseteq R$ such that $\mathrm{pr}_i \mathbf{a}' = \mathrm{pr}_i \mathbf{b}'$, $\mathrm{pr}_{i+1} \mathbf{a}' = a$, and $\mathrm{pr}_{i+1} \mathbf{b}' = b$. If $\mathrm{pr}_1 \mathbf{a}' = d$ then we are done. So, we assume that $\mathrm{pr}_1 \mathbf{a}' \neq d$ and consider two cases depending on whether $\{\mathrm{pr}_1 \mathbf{a}', d\}$ is majority or minority. If $\{\mathrm{pr}_1 \mathbf{a}', d\}$ is majority then we set $\mathbf{a}^* = f(\mathbf{e}, \mathbf{e}, \ldots, \mathbf{e}, \mathbf{a}')$ and we set $\mathbf{b}^* = f(\mathbf{e}, \mathbf{e}, \ldots, \mathbf{e}, \mathbf{b}')$. It is not hard to see that $\mathbf{a}^*, \mathbf{b}^*$ are in $R$, $\mathrm{pr}_1 \mathbf{a}^* = \mathrm{pr}_1 \mathbf{b}^* = d$ and $\mathrm{pr}_{\{1,\ldots,i\}} \mathbf{a}^* = \mathrm{pr}_{\{1,\ldots,i\}} \mathbf{b}^*$. Since $\{a, b\}$ is minority, we have that $\mathrm{pr}_{i+1} \mathbf{a}^* = a$ and $\mathrm{pr}_{i+1} \mathbf{b}^* = b$. If $\{\mathrm{pr}_1 \mathbf{a}', d\}$ is minority then we set $\mathbf{a}^* = f(\mathbf{e}, \mathbf{a}', \ldots, \mathbf{a}', \mathbf{a}')$ and $\mathbf{b}^* = f(\mathbf{e}, \mathbf{a}', \ldots, \mathbf{a}', \mathbf{b}')$. Finally, we include $\mathrm{pr}_{2,\ldots,n} \mathbf{a}^*, \mathrm{pr}_{2,\ldots,n} \mathbf{b}^*$ into $\mathrm{Wit}_{R_d}$.

It is easy to see that the running time of the algorithm is polynomial in $n$.

Let us proof that the algorithm contructs the right sets. It is straightforward to see that $\mathrm{Wit}_{R_d} \subseteq R_d$. Let us start with $\mathrm{Pro}_{R_d}$. It suffices to show that, for every collection of indexes $I$ of size $k'-1$, $k' \leq k$ and any $(k'-1)$-tuple $\mathbf{a}$ the pair $(I, \mathbf{a})$ belongs to $\mathrm{Pro}_{R_d}$ if and only if $\mathbf{a} \in \mathrm{pr}_I R_d$. To see this, first observe that $\mathbf{a} \in \mathrm{pr}_I R_d$ if and only if $(d, \mathbf{a}) \in \mathrm{pr}_J R$. By Theorem 5, $R = \langle \mathrm{Wit}_R \rangle$. Hence, $\mathrm{pr}_J R = \langle \mathrm{pr}_J \mathrm{Wit}_R \rangle$ and the required property follows from the construction of $\mathrm{Pro}_{R_d}$.

Then we consider $\mathrm{Rec}_{R_d}$. In this case we have to show that there exist some $d_1, \ldots, d_{i-1}$ such that $(d_1, \ldots, d_{i-1}, a), (d_1, \ldots, d_{i-1}, b) \in \mathrm{pr}_{\{1,\ldots,i\}} R_d$ if and only if $(i, a, b) \in \mathrm{Rec}_{R_d}$. If such tuples exist then $(d, d_1, \ldots, d_{i-1}, a), (d, d_1, \ldots, d_{i-1}, b) \in \mathrm{pr}_{\{1,\ldots,i+1\}} R$ and $(i+1, a, b) \in \mathrm{Rec}_R$. Then there is $\mathbf{c} \in R_d$ such that $\mathrm{pr}_i \mathbf{c} = a$. Therefore $(\{i\}, a) \in \mathrm{Pro}_R$ and consequently the algorithm includes $(i, a, b)$ into $\mathrm{Rec}_{R_d}$ in phase 2. Conversely, if $(i, a, b) \in \mathrm{Rec}_{R_d}$ then, for some $d_1, \ldots d_{i-1}$, we get $(d, d_1, \ldots, d_{i-1}, a), (d, d_1, \ldots, d_{i-1}, b) \in \mathrm{pr}_{\{1,\ldots,i+1\}} R$ and, therefore, $(d_1, \ldots, d_{i-1}, a)$, $(d_1, \ldots, d_{i-1}, b) \in \mathrm{pr}_{\{1,\ldots,i\}} R_d$.

By iterating the process we can obtain for any given $a_1, \ldots, a_l \in A$, a signature of $R_{a_1,\ldots,a_l} = (\ldots(R_{a_1})_{a_2})_{\ldots})_{a_l}$. ■

By convention the exists one tuple of arity 0, denoted $\lambda$. Consequently there are only two relations of arity 0, the empty relation $\emptyset$ and the relation $\{\lambda\}$. This enables us to deal with signatures of relations of arity 0 in a natural way. If $R$ is $\emptyset$ then $\mathrm{Pro}(R) = \mathrm{Rec}(R) = \mathrm{Wit}(R) = \emptyset$ whereas if $R$ is $\{\lambda\}$ then $\mathrm{Pro}(R) = \{(\emptyset, \lambda)\}$, $\mathrm{Rec}(R) = \emptyset$, and $\mathrm{Wit}(R) = \{\lambda\}$.

**Corollary 2.** *Let $R$ be an $n$-ary relation invariant under $f$. There exists an algorithm that, given a signature of $R$ and a tuple $\mathbf{a} = (a_1, \ldots, a_n) \in A^n$, decides whether $\mathbf{a} \in R$ with running time polynomial in $n$.*

**Proof.** To check if a given tuple $(a_1, \ldots, a_n)$ belongs to $R$, it suffice to check whether the set $\mathrm{Wit}(R_{a_1,\ldots,a_n})$ contains $\lambda$. ■

It is important to notice here that the algorithm which certifies that signatures are polynomially evaluable does not necessary succeed when the signature that receives as input does not represent a relation. However, for every $n$-ary signature $(\mathrm{Pro}, \mathrm{Rec}, \mathrm{Wit})$ and every $a_1, \ldots, a_m$ we can define the signature $(\mathrm{Pro}_{a_1,\ldots,a_m}, \mathrm{Rec}_{a_1,\ldots,a_m}, \mathrm{Wit}_{a_1,\ldots,a_m})$ as the output of the algorithm introduced in the proof of Theorem 6 regardless on whether $(\mathrm{Pro}, \mathrm{Rec}, \mathrm{Wit})$ is the signature of a relation or not. The reason for this is that we are interested in regarding the set of all signatures as our representation class. In this case, we will assume that the relation represented by a signature $s$ consists of all tuples $\mathbf{a}$ for which the algorithm with input $s$ and $\mathbf{a}$ outputs "yes". It is easy to observe that the relation represented by a signature $(\mathrm{Pro}, \mathrm{Rec}, \mathrm{Wit})$ is allways a subset of $\langle \mathrm{Wit} \rangle$.

4.3. **Signatures are polynomially learnable with membership queries.** Making use of the algorithm presented in Theorem 6, it is not difficult to find an algorithm that learns signatures with equivalence queries. The algorithm is presnted in Figure 1.

The target concept of the algorithm is an $n$-ary relation invariant under $f$. Recall that $f$, $k$ and $A$ are fixed. Notice that every round of the algorithm, either Pro or Rec increases its size. Furthermore, at any stage of the execution of the algorithm the relation represented by $(\mathrm{Pro}, \mathrm{Rec}, \mathrm{Wit})$ is a subset of $\langle \mathrm{Wit} \rangle \subseteq R$. This two combined facts ensure that the algorithm ends in a number of steps polynomial in $n$.

**Theorem 7.** *The class* Sig *is polynomially learnable with equivalence queries.*

*Step 1.* **set** Pro, Rec, Wit $:= \emptyset$
*Step 2.* **while** $EQ((\mathrm{Pro}, \mathrm{Rec}, \mathrm{Wit}))=$ 'no' **do**
let $\mathbf{a} = (a_1, \ldots, a_n)$ be the counterexample produced by $EQ$
*Step 2.1* **if** for some $I = \{i_1, \ldots, i_{k-1}\}$, $i_1 < i_2 < \cdots < i_{k-1}$ the pair $(I, \mathrm{pr}_I\, \mathbf{a}) \notin \mathrm{Pro}$ **then**
*Step 2.1.1* **set** $\mathrm{Pro} := \mathrm{Pro} \cup (I, \mathrm{pr}_I\, \mathbf{a})$ and $\mathrm{Wit} := \mathrm{Wit} \cup \{\mathbf{a}\}$
*Step 2.2* **else**
For each $0 \leq i \leq n$ compute $(\mathrm{Pro}_{a_1,\ldots,a_i}, \mathrm{Rec}_{a_1,\ldots,a_i}, \mathrm{Wit}_{a_1,\ldots,a_i})$.
Let $i$ be the smallest integer such that $\lambda \notin \mathrm{Wit}_{a_1,\ldots,a_i}$ (such $i$ must exist if $\mathbf{a}$ does not belong to the relation represented by (Pro, Rec, Wit))
*Step 2.2.1* **pick** an element $\mathbf{b} = (b_i, b_{i+1}, \ldots, b_n) \in \mathrm{Wit}_{a_1,\ldots,a_{i-1}}$ ($\{a_i, b_i\}$ must be a minority pair by Theorem 5)
*Step 2.2.* **set** $\mathrm{Rec} := \mathrm{Rec} \cup \{(i, a_i, b_i)\}$ and $\mathrm{Wit} := \mathrm{Wit} \cup \{\mathbf{a}, (a_1, \ldots, a_{i-1}, b_i, b_{i+1}, \ldots, b_n)\}$
**endwhile**
*Step 3.* **return** Pro, Rec, Wit

FIGURE 1. Algorithm learning signatures with membership queries

**Corollary 3.** *Let $A$ be a finite set and let $\Gamma$ be a finite set of relation on $A$.*

- *If $\Gamma$ is invariant under a GMM operation then* $\forall\exists$-Form($\Gamma$) *is polynomially learnable with equivalence queries in* Sig.
- *If $\Gamma$ is invariant under a conservative GMM operation then* $\forall\exists_{\mathrm{r}}$-Form($\Gamma$) *is polynomially learnable with equivalence queries in* Sig.

## References

[1] D. Angluin. Queries and Concept Learning. *Machine Learning*, 2:319–342, 1988.
[2] D. Angluin, M. Frazier, and L. Pitt. Learning Conjunctions of Horn Clauses. *Machine Learning*, 9:147–164, 1992.
[3] D. Angluin and M. Kharitonov. When won't Membership Queries help. *Journal of Computer and System Sciences*, 50:336–355, 1995.
[4] F. Börner, A.A. Bulatov, P.G. Jeavons, and A.A. Krokhin. Quantified constraints and surjective polymorphisms. In *Proceedings of 17th International Workshop Computer Science Logic, CSL'03*, volume 2803 of *Lecture Notes in Computer Science*, pages 58–70. Springer-Verlag, 2003.
[5] N. Bshouty, J. Jackson, and C. Tamon. Exploiring Learnability between Exact and PAC. In *15th Annual ACM Conference on Computational Learning Theory, COLT'02*, pages 244–254, 2002.
[6] N.H. Bshouty. Exact Learning Boolean Functions via the Monotone Theory. *Information and Computation*, pages 146–153, November 1995.
[7] A.A. Bulatov. Tractable conservative constraint satisfaction problems. In *Proceedings of the 18th Annual IEEE Simposium on Logic in Computer Science*, pages 321–330, Ottawa, Canada, June 2003. IEEE Computer Society.
[8] H. Chen. Quantified constraint satisfaction problems: Closure properties, complexity, and algorithms. Technical report, Cornell University, 2003.
[9] N. Creignou. A Dichotomy Theorem for Maximum Generalized Satisfiability Problems. *Journal of Computer and System Sciences*, 51(3):511–522, 1995.
[10] N. Creignou and M. Hermann. Complexity of Generalized Satisfiability Counting Problems. *Information and Computation*, 125:1–12, 1996.

[11] N. Creignou, S. Khanna, and M. Sudan. *Complexity Classification of Boolean Constraint Satisfaction Problems*, volume 7 of *Monographs on Discrete Mathematics and Applications*. SIAM, 2001.
[12] V. Dalmau. A Dichotomy Theorem for Learning Quantified Boolean Formulas. *Machine Learning*, 35(3):207–224, 1999.
[13] V. Dalmau and P. Jeavons. Learnability of Quantified Formulas. *Theoretical Computer Science*, (306):485–511, 2003.
[14] Jeffrey C. Jackson. An Efficient Membership-query Algorithm for Learning DNF with respect to the Uniform Distribution. *Journal of Computer and System Sciences*, 55(3):414–440, December 1997.
[15] P. Jeavons, D. Cohen, and M.C. Cooper. Constraints, Consistency and Closure. *Artificial Intelligence*, 101:251–265, 1998.
[16] Michael Kearns and Leslie Valiant. Cryptographic limitations on learning Boolean formulae and finite automata. *Journal of the ACM*, 41(1):67–95, January 1994.
[17] S. Khanna, M. Sudan, and L. Trevisan. Constraint Satisfaction: The Approximability of Minimization Problems. In *12th IEEE Conference on Computational Complexity*, 1997.
[18] S. Khanna, M. Sudan, and P. Williamson. A Complete Classification fo the Approximability of Maximation Problems Derived from Boolean Constraint Satisfaction. In *29th Annual ACM Symposium on Theory of Computing*, 1997.
[19] N. Pippenger. *Theories of Computability*. Cambridge University Press, 1997.
[20] E.L. Post. *The Two-Valued Iterative Systems of Mathematical Logic*, volume 5 of *Annals of Mathematics Studies*. Princeton, N.J, 1941.
[21] T.J. Schaefer. The Complexity of Satisfiability Problems. In *10th Annual ACM Symposium on Theory of Computing*, pages 216–226, 1978.
[22] A. Szendrei. *Clones in Universal Algebra*, volume 99 of *Seminaires de Mathématiques Supéreiores*. University of Montreal, 1986.
[23] L. Valiant. A Theory of the Learnable. *Comm. ACM*, 27(11):1134–1142, 1984.

## 5. Appendix: Negative Learnability Results (To be read at the discrection of the Program Comitee)

In this section we shall proof Theorem 4. As a rather direct consequence of Theorem 4 we can prove the non-learnability part of Theorem 3.

**Corollary 4.** *Let $A$ be a finite set and let $\Gamma$ be a finite set of relations on $A$ such that $\exists_{\mathrm{r}}$-Form($\Gamma$) is polynomially predictable with membership queries. If there exists public key encryption systems secure against chosen ciphertext attack then $\Gamma$ is invariant under a GMM operation.*

**Proof.** Let $\mathrm{eq}_A$ be the relation $\{(a,a)|a \in A\}$. By an straightforward adapation of the proof of Lemma 10 in [13] it is possible to prove that if $\exists_{\mathrm{r}}$-Form($\Gamma$) is polynomially predictable then so is $\exists_{\mathrm{r}}$-Form($\Gamma \cup \{\mathrm{eq}_A\}$). Let $\Gamma' = \Gamma \cup \{\mathrm{eq}_A\} \cup 2^A$ be the basis obtained by adding to $\Gamma$ the equality relation $\mathrm{eq}_A$ and all unary relations. It is rather direct to see that given a formula $\Phi$ in $\exists$-Form($\Gamma'$) it is possible to construct an equivalent formula $\Psi$ in $\exists_{\mathrm{r}}$-Form($\Gamma \cup \{\mathrm{eq}_A\}$): For each atomic formula "$R(x)$" in $\Phi$ where $R$ is unary we just have to construct the formula $\exists y(y \in R)\Phi'$ where $\Phi'$ is obtained from $\Phi$ by replacing $R(x)$ by $eq_A(x,y)$. Since the size of this construction is polynomial in the size of $\Phi$ we can infer that $\exists$-Form($\Gamma'$) in polynomially contained in $\exists_{\mathrm{r}}$-Form($\Gamma \cup \{\mathrm{eq}_A\}$) and hence, $\exists$-Form($\Gamma'$) is polynomially predictable with membership queries. Finally, by Theorem 4 $\Gamma'$ (and hence $\Gamma$) is invariant under a GMM operation when the cryptographic assumptions of the theorem are satistied. ∎

In order to proof Theorem 4 we need first to introduce some notation.

**Definition 3.** When $\Gamma$ is a set of relations on $A$, we say that $\Gamma$ is *colorable* if for all $a, b \in A$, there exists an operation $f$ in Pol($\Gamma$) such that the restriction of $f$ to $\{a,b\}$ is either near-unanimity (NU) or the affine operation $g(x,y,z) = x \oplus y \oplus z$ on $\{a,b\}$.

We use $x_1 \oplus \cdots \oplus x_k$ (with $k$ odd) to denote the operation on a two-element domain $\{a,b\}$ that is defined as $a$ if an odd number of the $x_i$ are equal to $a$, and defined as $b$ if an odd number of the $x_i$ are equal to $b$.

When a set of relations $\Gamma$ (on $A$) is colorable, we shall associate a color with each two-element subset $\{a,b\}$ of $A$, in the following way: $\{a,b\}$ is colored *yellow* if there exists a polymorphism of $\Gamma$ that, when restricted to $\{a,b\}$, is near-unanimity; $\{a,b\}$ is colored *blue* if there exists a polymorphism of $\Gamma$ that, when restricted to $\{a,b\}$, is equal to $g(x,y,z) = x \oplus y \oplus z$ and there does not exist a polymorphism of $\Gamma$ that, when restricted to $\{a,b\}$, is near-unanimity.

Theorem 4 is obtained as an immediate consequence of the following two results.

**Theorem 8.** *Let $A$ be a finite set, let $\Gamma$ be a finite set of relations on $A$ containing all unary relations on $A$ of size at most 2 such that $\exists$-Form($\Gamma$) is polynomially predictable with membership queries. Then under the assumption that public key encryption systems secure against chosen ciphertext attack exists, the set $\Gamma$ is colorable.*

**Theorem 9.** *Let $\Gamma$ be a set of relations on $A$ containing all unary relations on $A$ of size at most 2. If the set of relations $\Gamma$ is colorable, then the set of operations* Pol($\Gamma$) *contains a GMM operation.*

**Proof.** (Theorem 8) A complete proof of this lemma requires the introduction of a considerable amount of algebraic background. Here we will only sketch briefly the main ingredients of the proof adding pointers that will allow the interested reader to fill in the omitted details. The basic concept that we shall use in this proof is that of *clone*. A set of operations $F$ on $A$ is called a clone if it satisfies the following conditions:

$C_1$ For each $1 \leq m \leq n$, $F$ contains the *projection operation* $\mathrm{proj}_{n,m}$ defined by

$$\mathrm{proj}_{n,m}(x_1, \ldots, x_n) = x_m$$

$C_2$ For each $1 \leq n, m$, any $n$-ary operation $f$ in $F$ and any $m$-ary operations $g_1, \ldots, g_n$ in $F$, $F$ contains the composite operation $h$ defined by

$$h(x_1, \ldots, x_n) = f(g_1(x_1, \ldots, x_m), \ldots, g_n(x_1, \ldots, x_m))$$

Clones have been studied intensively in universal algebra (see [22] for a monograph on clone theory).

Let us assume that there exist some $a, b$ in $A$ such that, for any polymorphism $f$ of $\Gamma$, the restriction of $f$ on $\{a, b\}$ is not NU nor the ternary operation $x \oplus y \oplus z$. We aim to show that $\exists$-Form($\Gamma$) is not polynomially predictable with membership queries.

Let us define $F$ as the set of operations on $\{a, b\}$ that contains precisely, for any polymorphism $f$ of $\Gamma$, its restriction on $\{a, b\}$. It is an easy exercise to verify that $F$ constitutes a clone. A complete classification of all clones on a 2-element domain was obtained by Post [20]. In the following we shall make an intensive use of Post classification. We do not provide a description of Post's classification here. The interested reader can found a complete description in the excelent monograph by Szendrei. In our analysis we follow [13] which itself follows [19]. It is not difficult to see, by inspecting Post's classification, that every clone containing only conservative, and hence idempotent, operations on a 2-element set and that does not contain a NU operation nor the ternary operation $g(x, y, z)$ is of a very restricted form. In particular (see [13]) $F$ must be one of the following clones: $F_1^\infty, F_2^\infty, F_5^\infty, F_6^\infty, S_1, P_1, O_1$. It can be seen that if $\Gamma$ is *finite* and $F$ contains any of $F_1^\infty$, $F_2^\infty$, $F_5^\infty$, or $F_6^\infty$, then $F$ must contain also a NU operation, or more succintly, $F_1^\infty$, $F_2^\infty$, $F_5^\infty$, and $F_6^\infty$ cannot be the clone of polymorphisms of a finite $\Gamma$. Consequently we are left with only three cases, $F$ must be one of: $S_1$, $P_1$ or $O_1$. It can be seen (again by inspecting Post's classification) that in all three cases at least one of the relations $\neg x \vee \neg y \vee z$ or $x \vee y \vee \neg z$ (here $a$ is interpreted as false and $b$ is interpreted as true) is invariant under every operation of $F$, and hence, under every polymorphism of $\Gamma$.

At this point we have to use a previous learning result. In [12] it is shown that both $\exists$-Form($\{\neg x \vee \neg y \vee z\}$) and $\exists$-Form($\{x \vee y \vee \neg z\}$) are not polynomially predictable with membership queries under the cryptographic assumptions of the theorem. Consequently, if we set $\Gamma_0$ to $\{\neg x \vee \neg y \vee z\}$ and to $\{x \vee y \vee \neg z\}$ (depening to the case) in Theorem 1 we can conclude that, again under the cryptographic assumption of the theorem, $\exists$-Form($\Gamma$) is not polynomially predictable with membership queries. ∎

In order to establish Theorem 9, we first prove some helpful lemmas. We first show that relations of a particular form can be decomposed.

**Definition 4.** A relation $R \subseteq A^n$ is *d-decomposable* when the following holds: $\mathbf{t} \in R$ if and only if for all $I \subseteq [n]$ with $|I| \leq d$, $\mathrm{pr}_I \mathbf{t} \in \mathrm{pr}_I R$.

**Lemma 2.** *Let $\Gamma$ be a colorable set of relations on a finite set $A$ containing all unary relations on $A$ of size at most 2. There exists a constant $d$ such that every relation $R \subseteq A^n$ satisfying the following two properties is $d$-decomposable:*

- *$R$ is invariant under all polymorphisms of $\Gamma$, and*
- *for each $i \in [n]$, the projection $\mathrm{pr}_i R$ is contained in a yellow edge.*

**Proof.** For each yellow edge $\{a, b\} \subseteq A$, there exists (by definition) a polymorphism that is near-unanimity when restricted to $\{a, b\}$. Let $r$ be the maximum arity of such polymorphisms (over all yellow edges). Set $n = r|A|^2$ and $d = n - 1$. We prove that any relation satisfying the hypotheses is $d$-decomposable. It suffices to show that for any relation $R \subseteq A^m$ of arity $m \geq n$ (that satisfies the hypotheses) is $(m-1)$-decomposable. Let $R \subseteq A^m$ be such a relation and assume that $\mathbf{t}$ is a tuple (of length $m$) such that for all $I \subseteq [m]$ with $|I| = m-1$, it holds that $\mathrm{pr}_I \mathbf{t} \in \mathrm{pr}_I R$. We aim to show that $\mathbf{t} \in R$.

By assumption, there exist tuples $\mathbf{t^1}, \ldots, \mathbf{t^m} \in R$ such that for all $i \in [m]$, it holds that $\mathrm{pr}_{[m]\setminus\{i\}} \mathbf{t^i} = \mathrm{pr}_{[m]\setminus\{i\}} \mathbf{t}$, that is, $\mathbf{t^i}$ is equal to $\mathbf{t}$ at all coordinates except (possibly) the $i$th coordinate.

By the pigeonhole principle, there exists a subset $I \subseteq [m]$ of cardinality $|I| \geq m/|A|$ such that $\mathrm{pr}_I \mathbf{t} = (a, \ldots, a)$ for some $a \in A$. By another application of the pigeonhole principle, there exists a subset $J \subseteq I$ of cardinality $|J| \geq |I|/|A|$ such that for all $i, j \in J$ it holds that $\mathrm{pr}_i \mathbf{t}^i = \mathrm{pr}_j \mathbf{t}^j$. Let $b$ denote the element of $A$ such that $\{b\} = \{\mathrm{pr}_i t^i : i \in J\}$.

Let $f : A^s \to A$ be a polymorphism of $\Gamma$ (and hence $R$) of arity $s \leq r$ that acts as a near-unanimity operation when restricted to $\{a, b\}$. Since $|J| \geq |I|/|A| \geq m/|A|^2 \geq n/|A|^2 \geq r \geq s$, we can pick $s$ distinct elements $i_1, \ldots, i_s$ of $J$. It is straightforward to verify that $f(\mathbf{t^{i_1}}, \ldots, \mathbf{t^{i_s}}) = \mathbf{t}$, implying that $\mathbf{t}$ is in $R$. ∎

Next, we show that for any colorable set of relations, we can find a polymorphism that is near-unanimity on any yellow edge.

**Lemma 3.** *Let $\Gamma$ be a colorable set of relations on a finite set $A$ containing all unary relations on $A$ of size at most 2. There exists a polymorphism $f : A^k \to A$ of $\Gamma$ with $k$ odd and $k \geq 3$ such that the restriction of $f$ to any yellow edge is near-unanimity.*

**Proof.** By Lemma 2, we can select an even integer $d \geq 2$ so that for any relation $S$ satisfying the properties of Lemma 2, $S$ is $d$-decomposable. Set $k = d + 1$. Our proof follows the proof outline of [15].

Let $R$ be the relation of arity $|A|^k$ over index set $A^k$, containing all tuples $\mathbf{t}$ such that the map $f_{\mathbf{t}} : A^k \to A$ defined by $f(a_1, \ldots, a_k) = \mathbf{t}_{(a_1,\ldots,a_k)}$ is a polymorphism of $\Gamma$. It is not difficult to see that $R$ is itself invariant under any polymorphism of $\Gamma$.

Define $R'$ as $\mathrm{pr}_I R$, where $I$ contains all coordinates $(a_1, \ldots, a_k)$ where all but at most one of the $a_i$s are equal, and $\{a_1, \ldots, a_k\}$ is contained in a yellow edge. For such a coordinate $(a_1, \ldots, a_k)$, there exists an index $j$ such that $\{a_i : i \in [k] \setminus \{j\}\}$ contains only one value; we call this value the *same value* of $(a_1, \ldots, a_k)$. Since every polymorphism of $\Gamma$ preserves all one and two element subsets of $A$ (since they are unary relations in $\Gamma$), every projection $\mathrm{pr}_{(a_1,\ldots,a_k)} R'$ (with $(a_1, \ldots, a_k) \in I$)) is contained in a yellow edge, and hence satisfies the properties of Lemma 2 and is $d$-decomposable.

Let $\mathbf{t}$ be the tuple over index set $I$ where for all $(a_1, \ldots, a_k) \in I$, it holds that $\mathrm{pr}_{(a_1,\ldots,a_k)} \mathbf{t}$ is equal to the same value of $(a_1, \ldots, a_k)$. We claim that $\mathbf{t} \in R'$ (which gives the theorem). Let $J$ be an arbitrary subset of $I$ with $|J| = d$. Since $R'$ is $d$-decomposable, it suffices to show that $\mathrm{pr}_J \mathbf{t} \in \mathrm{pr}_J R'$. Since $k = d + 1$ there exists an index $j \in [k]$ such that $a_j$ is the same value of $(a_1, \ldots, a_k)$ for all $(a_1, \ldots, a_k) \in J$. Let $\mathbf{u}$ be the tuple in $R$ such that $\mathrm{pr}_{(a_1,\ldots,a_k)} \mathbf{u} = a_j$ for all $(a_1, \ldots, a_k) \in A^k$. We have $\mathrm{pr}_J \mathbf{t} = \mathrm{pr}_J \mathbf{u} \in \mathrm{pr}_J R'$. ∎

With Lemma 3 in hand, we now prove Theorem 9.

**Proof.** (Theorem 9) We have a polymorphism $f : A^k \to A$ (with $k$ odd and $k \geq 3$) that is majority on every yellow edge, by Lemma 3. It can be shown (see for example Proposition 2 of [7]) that there is a ternary polymorphism $g(x, y, z)$ that acts as $x \oplus y \oplus z$ on every blue edge. By composing $g$ with itself, we obtain a polyomrphism $h : A^k \to A$ that acts as $h(x_1, \ldots, x_k) = x_1 \oplus \cdots \oplus x_k$ on every blue edge. We define $f' : A^k \to A$ by $f'(x_1, \ldots, x_k) = h(f(x_1, \ldots, x_k), f(x_2, \ldots, x_k, x_1), \ldots,$
$f(x_k, x_1, \ldots, x_{k-1}))$. We claim that $f'$ is GMM; in particular, we claim that $f'$ satisfies (1) on every yellow edge, and (2) on every blue edge. That $f'$ is majority on every yellow edge follows from the definition of $f$ and the fact that $h$ is idempotent. By Post's theorem [20], on any blue edge $\{a, b\}$ it must be the case that $f(x_1, \ldots, x_k) = x_{i_1} \oplus \cdots \oplus x_{i_l}$ for $l$ odd; if not, $f$ can be used with an operation acting as $x \oplus y \oplus z$ on $\{a, b\}$ to derive a polymorphism that is majority on $\{a, b\}$, contradicting that we colored $\{a, b\}$ blue and not yellow. From this fact, it follows that $f'$ is in fact equal to $h$ on every blue edge, and hence $f'$ satisfies (2) on every blue edge. ■