A Dichotomy Theorem for Learning Quantified Boolean Formulas

Víctor Dalmau*

Departament LSI, Universitat Politècnica de Catalunya,
Campus Nord, Mòdul C5,
Jordi Girona Salgado, 1-3, Barcelona 08034, Spain,
dalmau@lsi.upc.es

26 March 1997

Abstract

We consider the following classes of quantified boolean formulas. Fix a finite set of basic boolean functions. Take conjunctions of these basic functions applied to variables and constants in arbitrary way. Finally quantify existentially or universally some of the variables. We prove the following *dichotomy theorem*: For any set of basic boolean functions, the resulting set of formulas is either polynomially learnable from equivalence queries alone or else it is not PAC-predictable even with membership queries under cryptographic assumptions. Furthermore we identify precisely which sets of basic functions are in which of the two cases.

*Work supported in part by the Spanish Government through grant AP94 43716784 and the DGICYT (project BP92-0709) and the EC through the Esprit BRA Program (Working Group 8556, NeuroColt and project 20244, ALCOM IT).

1 Introduction

The problem of learning an unknown boolean formula under some determined protocol has been widely studied. It is well known that, even restricted to propositional formulas, the problem is hard [3, 13] in the usual learning models. Therefore people have attempted to learn subclasses of propositional boolean formulas, specially inside CNF and DNF. For example, k-DNF formulas, k-term DNF formulas, monotone-DNF formulas, Horn formulas, and their dual counterparts [1, 4, 2] have all been shown exactly learnable using membership and equivalence queries in Angluin's model [1] while the question of whether DNF formulas are learnable is still open. The more powerful formalism of predicate logics is used in several applications of learning in artificial intelligence and knowledge representation but its study, from the computational learning theory point of view, is recent. See [16] and the further references in this paper.

In this paper we study the complexity of learning some subclasses of quantified boolean formulas called quantified boolean formulas over a basis S. These formulas are still propositional formulas but augmented with the additional capability of quantification.

Let $S = \{R_1, \ldots, R_m\}$ be a finite set of logical relations. Define an $\exists\forall$-Formula(S) to be any boolean formula formed by quantified conjunctions of any number of clauses of the form $R_i(\xi_1, \ldots, \xi_k)$, where $\xi_1, \ldots, \xi_k$ are variables or constants and k is the rank of R_i.

An example: Consider the problem of learning a boolean formula formed by a quantified conjunction of clauses with three literals per clause. Every such formula can be expressed as a formula in $\exists\forall$-Formula(S) with the set of logical relations $S = \{R_0, R_1, R_2, R_3\}$, defined by:

$$\begin{aligned} R_0(x,y,z) &\equiv x \vee y \vee z, \\ R_1(x,y,z) &\equiv \overline{x} \vee y \vee z, \\ R_2(x,y,z) &\equiv \overline{x} \vee \overline{y} \vee z, \\ R_3(x,y,z) &\equiv \overline{x} \vee \overline{y} \vee \overline{z}. \end{aligned}$$

The main result of this paper characterizes the complexity of learning $\exists\forall$-Formula(S) for every finite set S of logical relations. The most striking feature of this characterization is that for any S, $\exists\forall$-Formula(S) is either polynomially learnable with equivalence queries alone or, under some cryptographic assumptions, not polynomially predictable even with membership queries. In fact, for the hardness result, it is enough to consider formulas with

existential quantifiers only or without constants. This dichotomy is somewhat surprising since one might expect that any such large and diverse class of concepts would include some representatives of the many intermediate learning models.

Furthermore, we give an interesting classification of the polynomially learnable classes. We show that $\exists\forall$-Formula(S) is polynomially learnable if and only if at least one of the following conditions holds:

(a) Every relation in S is definable by a CNF formula in which each clause has at most 2 literals.

(b) Every relation in S is definable by a CNF formula in which each clause has no negated variables or has at most one negated variable and at most one affirmed variable.

(c) Every relation in S is definable by a CNF formula in which each clause has no affirmed variables or has at most one affirmed variable and at most one negated variable.

(d) Every relation in S is the set of solutions of a system of linear equations over the two-element field $\{0, 1\}$.

A few but not many Dichotomy results in complexity theory are already known. The first one is a dichotomy result for the generalized satisfiability decision problem by Schaefer [19]. The others concern the H-coloring of graphs [11], the subgraph homeomorphism [9], the inverse generalized satisfiability problem [12], the generalized satisfiability counting problem [6], and the aproximability of minimization and maximization problems [7, 14, 15]. It is remarkable that most of the dichotomy results shown before are in the framework of generalized satisfiability problems proposed by Schaefer. Our result is inspired as well by the framework and techniques of Schaefer and this fact allow us to compare the complexity of different problems on generalized quantified boolean formulas. For example, from Schaefer's Dichotomy Theorem [19] and the Dichotomy Theorem of this paper can be inferred that, in this framework, learnability is slightly harder than satisfiability.

The aim of this paper is to study the complexity of learning generalized quantified boolean formulas, but some intermediate results are interesting in themselves. In particular, the technique used in the semantic characterization of weakly antimonotone logical relations can be useful to characterize other logical relations defined as conjunction of some restricted kinds of clauses.

2 Definitions and Notation

2.1 Learning Models

Most of the terminology used in this section comes from [3]. Let X denote $\{0,1\}^*$; binary strings will represent both examples and concept names. If x is a string and $b \in \{0,1\}$ is a constant, $|x|$ denotes its length and $|x|_b$ denotes the number of occurrences of b in x. For any natural number n, $X^{[n]} = \{x \in X : |x| \leq n\}$. A *representation of concepts* $\mathcal{C}$ is any subset of $X \times X$. We interpret an element $\langle u, x \rangle$ of $X \times X$ as consisting of a *concept name* u and an *example* x. The example x is a member of the concept u if and only if $\langle u, x \rangle \in \mathcal{C}$. Define the *concept represented by* u as

$$K_{\mathcal{C}}(u) = \{x : \langle u, x \rangle \in \mathcal{C}\}$$

The *set of concepts represented by* $\mathcal{C}$ is

$$\{K_{\mathcal{C}}(u) : u \in X\}$$

In this paper we use two models of learning, both of which are fairly standard: Angluin's model of exact learning from equivalences queries [1] and the model of PAC-prediction with membership queries as defined by Angluin and Kharitonov in [3].

Let $\mathcal{H}$ be a representation class. A learning algorithm with queries is an algorithm A that takes as input a bound s on the size of the target concept representation and a bound n on the length of the examples. It may make any number of queries or requests, the responses to which are determined by the unknown target concept c. A must eventually halt with an output concept name v. The concept $K_{\mathcal{H}}(v)$ is interpreted as A's guess of the target concept. The most common kinds of queries are the *membership* and the *equivalence* queries. A membership query takes a string $x \in X$ as input and returns 1 if $x \in c$ and 0 otherwise. An equivalence query takes a concept name h as input and returns *yes* if $c = K_{\mathcal{H}}(h)$ and a counterexample $x \in c \triangle K_{\mathcal{H}}(h)$ otherwise. *A runs in polynomial time* if its running time (counting one step for oracle call) is bounded by a polynomial in s and n.

We say that A successfully *exactly learns* a representation of concepts $\mathcal{C}$, if and only if for all positive integers s, n, for all concept names $u \in X^{[s]}$, when A is run with inputs s and n, and oracles determined by $c = K_{\mathcal{C}}(u)$, A outputs a concept name v such that $c = K_{\mathcal{H}}(v)$. If $\mathcal{C} = \mathcal{H}$ we say that A learns

properly $\mathcal{C}$, otherwise we say that A learns *improperly* $\mathcal{C}$. A representation of concepts $\mathcal{C}$ is *polynomially exactly learnable* if and only if there is a learning with queries algorithm A that runs in polynomial time and successfully learns exactly $\mathcal{C}$.

A *prediction with membership algorithm*, or *pwm-algorithm*, is a possibly randomized algorithm A that takes as input a bound s on the size of the target concept representation, a bound n on the length of examples, and an accuracy bound ϵ. It may make three different kinds of oracle calls, the responses to which are determined by the unknown target concept c and the unknown distribution D on $X^{[n]}$, as follows:

- A membership query takes a string $x \in X$ as input and returns 1 if $x \in c$ and 0 otherwise.
- A request for a random classified example takes no input and returns a pair $\langle x, b \rangle$, where x is a string chosen independently according to D and $b = 1$ if $x \in c$ and $b = 0$ otherwise.
- A request for an element to predict takes no input and returns a string x chosen independently according to D.

A may make any number of membership queries or requests for random classified examples. However, A must eventually make one and only one request for an element to predict and eventually halt with an output of 1 or 0 without making any further oracle calls. The output is interpreted as A's guess of how the target concept classifies the element returned by the request for an element to predict. *A runs in polynomial time* if its running time (counting one step per oracle call) is bounded by a polynomial in s, n, and $1/\epsilon$.

We say that A successfully *predicts* a representation of concepts $\mathcal{C}$ if and only if for all positive integers s and n, for all positive rationals ϵ, for all concept names $u \in X^{[s]}$, when A is run with inputs s, n, and ϵ, and oracles determined by $c = K_{\mathcal{C}}(u)$ and D, A asks membership queries that are in X and the probability is at most ϵ that the output of A is not equal to the correct classification of x by $K_{\mathcal{C}}(u)$, where x is the string returned by the (unique) request for an element to predict. We can say that A predicts $\mathcal{C}$ in PAC sense, with the additional help of membership queries. See [20] for a formal definition of the PAC model.

A representation of concepts $\mathcal{C}$ is *polynomially predictable with membership queries* if and only if there is a *pwm*-algorithm A that runs in polynomial

time and successfully predicts $\mathcal{C}$. If a representation of concepts is learnable in polynomial time with membership and equivalence queries then it is polynomially predictable with membership queries.

2.2 Reducibility among prediction problems

To compare the difficulty of learning problems in the prediction model we use the *prediction-preserving reducibility with membership queries* as defined by Angluin and Kharitonov [3]. It is denoted by $\leq_{pwm}$ and it extends Pitt and Warmuth's *prediction-preserving reducibility* [18] to the presence of membership queries.

Definition 1 *Let $\mathcal{C}$ and $\mathcal{C}'$ be representations of concepts. Let $\perp$ and $\top$ be elements not in X. Then $\mathcal{C}$ is* pwm-reducible *to $\mathcal{C}'$, denoted $\mathcal{C} \leq_{pwm} \mathcal{C}'$, if and only if there exist three mappings g,f, and h with the following properties:*

1. *There is a nondecreasing polynomial q such that for all natural numbers s and n and for $u \in X^{[s]}$, $g(s,n,u)$ is a string u' of length at most $q(s,n,|u|)$.*

2. *For all natural numbers s and n, for every string $u \in X^{[s]}$, and for every $x \in X^{[n]}$, $f(s,n,x)$ is a string x' and $x \notin K_{\mathcal{C}}(u)$ if and only if $x' \in K_{\mathcal{C}'}(g(s,n,u))$. Moreover, f is computable in time bounded by a polynomial in s, n, and $|x|$, hence there exists a nondecreasing polynomial t such that $|x'| \leq t(s,n,|x|)$.*

3. *For all natural numbers s and n, for every string $u \in X^{[s]}$, and for every $x' \in X$, $h(s,n,x')$ is either $\perp$, $\top$, or a string $x \in X$. If $h(s,n,x') = \top$ then $x' \in K_{\mathcal{C}'}(g(s,n,u))$, if $h(s,n,x') = \perp$ then $x' \notin K_{\mathcal{C}'}(g(s,n,u))$, and otherwise $x' \in K_{\mathcal{C}'}(g(s,n,u))$ if and only if $x \notin K_{\mathcal{C}}(u)$. Moreover, h is computable in time bounded by a polynomial in s, n, and $|x'|$.*

In (2), and independently in (3), the expression "$x \notin K_{\mathcal{C}}(u)$" can be replaced with "$x \in K_{\mathcal{C}}(u)$", as discussed in [3].[1]

The only properties of this reducibility that are needed in this paper were shown in [3]:

[1] Perhaps the use of $x \in K_{\mathcal{C}}(u)$ instead of its negated form seems more natural and, in fact, in [3] properties (2) and (3) are defined using $x \in K_{\mathcal{C}}(u)$. We use the negated form because this is the one we use in the only explicit pwm-reduction of this paper.

Lemma 2 *The pwm-reduction is transitive, i.e., let $\mathcal{C}, \mathcal{C}'$ and $\mathcal{C}''$ be representations of concepts, if $\mathcal{C} \leq_{pwm} \mathcal{C}' \leq_{pwm} \mathcal{C}''$ then $\mathcal{C} \leq_{pwm} \mathcal{C}''$.*

Lemma 3 *Let $\mathcal{C}$ and $\mathcal{C}'$ be representations of concepts. If $\mathcal{C} \leq_{pwm} \mathcal{C}'$ and $\mathcal{C}'$ is polynomially predictable with membership queries, then $\mathcal{C}$ is also polynomially predictable with membership queries.*

2.3 Logical Preliminaries

Let $V = \{x_1, x_2, \ldots\}$ be an infinite set of boolean variables. A literal is a variable or its negation. An assignment is a vector in X. For any assignment $t \in X$ and for any integer j, $t[j] \in \{0,1\}$ denotes the jth component of t. A logic relation of rank k (k integer) is a subset of $\{0,1\}^k$. There exists an unique assignment of length 0, we call it λ.

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

Let $S = \{R_1, \ldots, R_m\}$ be any finite set where each R_i is a logical relation of rank k_i. R_i denotes both the logical relation and its symbol. The set of formulas formed by conjunctions of relations in S with constants is denoted Formula(S). Specifically, Formula(S) is the smallest set of formulas such that:

- For all $R \in S$ of rank k, $R(y_1, \ldots, y_k) \in$ Formula(S) where $y_i \in V \cup \{0,1\}$ for $1 \leq i \leq k$.
- For all $F, G \in$ Formula(S), $F \wedge G \in$ Formula(S).

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

- For all $F \in$ Formula(S), $F \in \exists\forall$-Formula(S).
- For all $F \in \exists\forall$-Formula(S) and for all $\xi \in V$, $\exists\xi F$ and $\forall\xi F$ are in $\exists\forall$-Formula(S).

We call $\exists$-Formula(S) the subset of $\exists\forall$-Formula(S) that we obtain if we allow only existential quantifiers.

Each formula F defines a logical relation $[F]$ if we apply the usual semantics of first-order logic and the variables are taken in lexicographical order.

For every set of logical functions S we define Relation$(S) = \{[F] : F \in$ Formula$(S)\}$. Analogously, we define $\exists\forall$-Relation(S) and $\exists$-Relation(S).

For any set of formulas $\mathcal{F}$, we define $\mathcal{C}_{\mathcal{F}}$ as the representation class formed from formulas in $\mathcal{F}$.

A clause is *bijunctive* if it has at most 2 literals. A clause is *horn* (resp. *antihorn*) if it has at most one unnegated (resp. negated) variable. A clause is *weakly monotone* (resp. *weakly antimonotone*) if it is (i) the disjunction of unnegated variables (resp. negated variables) or (ii) the disjunction of at most two literals with at most one negated (resp. unnegated) variable. That is, a clause weakly antimonotone is a horn clause where we only allow rules like $(y_1 \ldots y_n \rightarrow \mathit{false})$ and $(y_i \rightarrow y_j)$ where y_i is variable or a constant. The logical relation R of rank k is bijunctive (resp. horn, antihorn, weakly monotone, weakly antimonotone) if $R(x_1, \ldots, x_k)$ is logically equivalent to some CNF formula where each clause is bijunctive (resp. horn, antihorn, weakly monotone, weakly antimonotone). The logical relation R of rank k is *affine* if $R(x_1, \ldots, x_k)$ is logically equivalent to some system of linear equations over the two-element field $\{0, 1\}$.

We can extend the definitions above to formulas and sets of relations: The formula F is bijunctive (resp. horn, antihorn, weakly monotone, weakly antimonotone, affine) if $[F]$ is bijunctive (resp. horn, antihorn, weakly monotone, weakly antimonotone, affine). The set S of logical relations is bijunctive (resp. horn, antihorn, weakly monotone, weakly antimonotone, affine) if every $R \in S$ is bijunctive (resp. horn, antihorn, weakly monotone, weakly antimonotone, affine).

The *degree* of a logical relation R or rank k, is the minimal value $d \leq k$ such that R can be expressed as a d-CNF formula. Analogously, the degree of a formula F is the degree of the logical relation $[F]$. The *degree* of a finite set of logical relations S is the maximum of the degrees of all relations in S.

3 The Dichotomy Theorem

This section states and proves the main result of this paper:

Theorem 4 *(Dichotomy Theorem for Learnability) Let S a finite set of logical relations. If S satisfies one of the conditions (a)-(d) below, then $\mathcal{C}_{\exists\forall\text{-}Formula(S)}$ is polynomially exactly learnable with improper equivalence queries. Otherwise, $\mathcal{C}_{\exists\text{-}Formula(S)}$ is not polynomially predictable with mem-*

bership queries under the assumption that public key encryption systems secure against chosen ciphertext attack exist.

a.- S *is bijunctive.*

b.- S *is affine.*

c.- S *is weakly monotone.*

d.- S *is weakly antimonotone.*

We refer the reader to [3] for definitions of the cryptographic concepts.

Schaefer [19] proves a similar dichotomy theorem for the satisfiability of Formula(S). He shows that this problem is polynomial-time solvable if and only if S is bijunctive, horn, antihorn of affine. Otherwise the problem is NP-complete. We can note here that if a basis S does not fall in the classes in Schaefer's theorem then the representation class $\exists\forall$-Formula(S) is not *honest*, i.e., we cannot decide in polynomial time if an example belongs to a concept.

From the comparison of both theorems it follows that, in this framework, learning is slightly harder than deciding satisfiability. More precisely, satisfiability when S is horn is polynomial-time decidable, but for polynomial time learnability we must guarantee that S falls in a more restricted class, namely the weakly antimonotone sets.

Observe that for the negative results only existencial quantifiers are needed. In [8] can be found a proof of the negative result in the case that constants are not allowed. Therefore the main theorem does not change at all if we are restricted to formulas without universal quantifiers or without constants.

Here is a bird's eye view of the proof. First we prove that the expressive power of formulas over sets in the theorem is essentially the same with and without using quantifiers. This allows us to reduce learning these formulas to learning easy classes of quantifier-free formulas. On the other hand, we show that the quantified formulas over any non-basic set can express a double implication $xy \to z$ or its dual. Then we show that this implication is enough to simulate monotone boolean circuits which are hard to learn under cryptographic assumptions.

3.1 Results in Logic

3.1.1 Closure by quantifiers

We begin this section showing some sets of logic relations "closed" by quantifiers. That is, sets of logic relations where the quantifiers do not help to obtain a more reduced representation. Specifically we show that the bijunctive, weakly monotone, weakly antimonotone, and affine sets S of logical relations satisfy this property. This property will be very important in the study of the learnability of $\exists\forall$-Formula(S).

We begin with some claims of easy proof:

Let F be a formula over V, let $\xi \in V$ be a variable and w a literal or a constant. We define $F\left[\begin{smallmatrix}\xi\\ w\end{smallmatrix}\right]$ as the formula formed from F by replacing each occurrence of ξ by w. If $W \subset V$ is a set of variables then $F\left[\begin{smallmatrix}W\\ w\end{smallmatrix}\right]$ denotes the result of substituting w for every occurrence of every variable in W. Multiple substitutions are denoted by expressions such as $F\left[\begin{smallmatrix}V, & V', & V''\\ w, & w', & w''\end{smallmatrix}\right]$ with obvious meaning.

Claim 5 *Let $x \in V$ be a variable. If F is a bijunctive (resp. weakly monotone of degree d, weakly antimonotone of degree d, affine) formula then $F\left[\begin{smallmatrix}x\\ 0\end{smallmatrix}\right]$ and $F\left[\begin{smallmatrix}x\\ 1\end{smallmatrix}\right]$ are bijunctive (resp. weakly monotone of degree d, weakly antimonotone of degree d, affine).*

Let $C = x \vee C'$ and $D = \overline{x} \vee D'$ be two clauses, where C' is the rest of clause C, and similarly for D and D'. That is, the two clauses contain two opposite literals. Then the clause $C' \vee D'$, containing all literals of the two clauses except for the two opposite literals is called the *resolvent* of C and D respect the variable x, denoted by $\mathcal{R}(C, D, x)$.

Claim 6 *Let $x \in V$ be a variable. Let C_x and $C_{\overline{x}}$ be a couple of clauses that contain the literal x and $\overline{x}$ respectively. If C_x and $C_{\overline{x}}$ are bijunctive (resp. weakly monotone of degree d, weakly antimonotone of degree d), then $\mathcal{R}(C_x, C_{\overline{x}}, x)$ is bijunctive (resp. weakly monotone of degree d, weakly antimonotone of degree d).*

We note here that in the case of general horn (resp. antihorn) clauses although the resolvent of two horn (resp. antihorn) clauses is a horn (resp. antihorn) clause, we cannot guarantee that degree does not increase.

Claim 7 *For every variable $x \in V$ and for every formula F, $\forall x F \equiv F\left[\begin{smallmatrix} x \\ 0 \end{smallmatrix}\right] \wedge F\left[\begin{smallmatrix} x \\ 1 \end{smallmatrix}\right]$.*

Claim 8 *For every variable $x \in V$ and for every CNF formula F such that $F \equiv \bigwedge_{C \in G_x \cup G_{\overline{x}} \cup G} C$ where G_x, $G_{\overline{x}}$, and G are sets of clauses that contain the literal x, the literal $\overline{x}$, and none of them respectively, the following equivalence holds: $\exists x F \equiv \bigwedge_{C_x \in G_x, C_{\overline{x}} \in G_{\overline{x}}} \mathcal{R}(C_x, C_{\overline{x}}, x) \wedge \bigwedge_{C \in G} C$.*

From Claims 5, 6, 7, and 8 we can derive the following lemma:

Lemma 9 *Let F a bijunctive (resp. weakly monotone of degree d, weakly antimonotone of degree d) formula. Then for every variable $x \in V$, $\forall x F$ and $\exists x F$ are both bijunctive (resp. weakly monotone of degree d, weakly antimonotone of degree d) formulas.*

The analogous for affine functions is proved differently.

Lemma 10 *Let F be an affine formula. Then for every variable $x \in V$, $\forall x F$ and $\exists x F$ are both affine.*

Proof.
Let F be equivalent to a system of linear equations over the two element field $\{0, 1\}$. To prove the case $\forall x F$ we only need apply Claims 5 and 7 or, even easier, note that $\forall x F \equiv$ *false* if x appears in F, and $\forall x F \equiv F$ otherwise. So we only need to consider the case $\exists x F$. We can take some equation that contains the variable x, separate x in one side of the equation and substitute the other side in the rest of equations. This process gives us a system of equations equivalent to $\exists x F$. ■

We can eliminate systematically the quantifiers from inside to outside preserving the bijunctivity (resp. weakly monotonicity of degree d, weakly antimonotonicity of degree d, affinity) obtaining the following result:

Corollary 11 *If S is a bijunctive (resp. weakly monotone of degree d, weakly antimonotone of degree d, affine) set of logical relations then $\exists\forall$-Relation(S) is bijunctive (resp. weakly monotone of degree d, weakly antimonotone of degree d, affine).*

From now, a set of logical relations S will be called *basic set* if S is bijunctive, weakly monotone, weakly antimonotone, or affine and the representation class $\exists\forall$-Formula(S) formed from a basic set S will be called *basic representation class.*

3.1.2 Semantic characterizations

In this section we give semantic characterizations of some sets of logical relations. Other semantic characterizations can be found in [19].

The following result is from [17]. In [19] we can find a more complicated characterization of Horn logical relations.

Lemma 12 *([17], Problem 4.4.7.) Let R be a logical relation of rank k. R is horn (resp. antihorn) if and only if for all $t, t' \in R$, $t \wedge t' \in R$ (resp. $t \vee t' \in R$).*

At this point we show a characterization of the weakly antimonotone relations. We need the following notation:

For every logical relation R of rank k and for every set $T = \{i_1, \ldots, i_j\}$ of j positions $0 \leq i_1 < \ldots < i_j \leq k$ we define $R|T$ as the subset of $\{0,1\}^j$ such that $t \in R|T$ if and only if there exists some assignment $t' \in R$ such that t agrees with t' in the positions indexed by T. That is, $R|T = [\exists x_{l_1}, \ldots, \exists x_{l_{k-j}} R(x_1, \ldots, x_k)]$ where $\{l_1, \ldots, l_{k-j}\} = \{1, \ldots, k\} \perp T$, and therefore $R|T \in \exists\text{-Relation}(\{R\})$.

For all $t \in \{0,1\}^k$ and $T \subset \{1, \ldots, k\}$ we define $t|T$ as the assignment $t' \in \{0,1\}^{|T|}$ that agrees with t in the positions indexed by T. We have $t|\emptyset = \lambda$ and $R|\emptyset$ is the relation *true* if $R \neq \emptyset$ and *false* otherwise.

Let R be a logical relation of rank k and $t \in \{0,1\}^k$ an assignment. We say that t is *j-compatible*[2] if for every subset $T \subset \{1, \ldots, k\}$ of size $|T| < j$ we can find some assignment $t' \in R$ such that t and t' agree in the positions indexed by T. Assignment λ is always 0-compatible.

A more intuitive view of the meaning of compatibility is the following claim.

Claim 13 *A logical relation R is expressable in j-CNF if and only if there is no $(j+1)$-compatible assignment not in R.*

The notion of j-compatibility is the key to characterize weakly antimonotone logical relations:

Lemma 14 *Let R be a logical relation of rank k. The following conditions are equivalent:*

[2] The notion of compatibility used in this paper differs only slightly from the compatibility defined by Kavvadias and Sidderi [12]. Precisely, the j-compatibility corresponds exactly to the $(j-1)$-compatibility in the sense of [12].

a.- *R is weakly antimonotone.*

b.- *For every $T \subset \{1, \ldots, k\}$ and every $|T|$-compatible assignment $t \notin R|T$:*

i.- $|t|_0 = 0$, *or*

ii.- $|t|_0 = 1$ *and* $|t|_1 \leq 1$.

c.- *For every $T \subset \{1, \ldots, k\}$ and every $|T|$-compatible assignment $t \notin R|T$:*

i.- $|t|_0 \leq 1$, *and*

ii.- *If* $|t|_0 = 1$ *then* $|t|_1 \leq 1$.

Proof.
The equivalence between conditions (b) and (c) is immediate. We only need to show the equivalence between conditions (a) and (b).

- **[b⇒a]**. For every T, set of positions $0 \leq i_1 < \ldots < i_j \leq k$ and for every $|T|$-compatible assignment $t \notin R|T$ we define

$$C_t^T = \bigvee_{l=1}^{j} x_{i_l}^{t[l]}$$

where

$$x_i^j = \begin{cases} x_i & \text{if } j = 0 \\ \overline{x}_i & \text{otherwise} \end{cases}$$

If t satisfies (bi) then C_t^T is antimonotone and falls in type (i) of weakly antimonotone clauses, otherwise t satisfies (bii), C_t^T has at most two literals with at most one unnegated literal and falls in type (ii) of weakly antimonotone clauses. Therefore C_t^T is a weakly antimonotone clause. From the construction of C_t^T and the assumption $t \notin R|T$ it is clear that for all assignments $t' \in \{0,1\}^k$ that falsify C_t^T we have $t' \notin R$.

For every logical relation R of rank k we define F as the conjunction of all clauses C_t^T where $T \subset \{1, \ldots, k\}$ and $t \notin R|T$ is a $|T|$-compatible assignment. We show that $[F] = R$:

  - It is clear that if $F(t) = 0$ then t falsifies some clause $C_{t'}^T$ and therefore $t \notin R$.

– Fix an assignment $t \notin R$. We apply the following algorithm:

Step 1 Assign $i := 1$, $t_1 := t$, $T_1 = \{1, \ldots, k\}$;

Step 2 If t_i is $|T_i|$-compatible with respect to the relation $R|T_i$ then **stop**, otherwise there exists some subset $T \subset T_i$ such that $t_i|T \notin R|T$. Assign $t_{i+1} := t_i|T$, $T_{i+1} := T$, $i := i + 1$; go to step 2.

Note that $|T_i|$ decreases in each step of the algorithm. When $T_i = \emptyset$, $t_i = \lambda$ is 0-compatible and therefore condition in step (2) is always reached and the algorithm always finds a t_i which is $|T_i|$-compatible with respect the relation $R|T_i$. So, F contains $C_{t_i}^{T_i}$ that is falsified by t.

- **[a⇒b]** If R is a weakly antimonotone logical relation then for every $T \subset \{1, \ldots, k\}$, by Corollary 11, $R|T$ is a weakly antimonotone logical relation. Let $F = \bigwedge_{j=1}^{s} C_j$ be a CNF formula where each clause is weakly antimonotone such that F denotes the logical relation $R|T$. Let $t \notin R|T$ be a $|T|$-compatible assignment. By the $|T|$-compatibility, t cannot falsify any clause of less than $|T|$ literals, and therefore must falsify a clause of exactly $|T|$ literals C_l. If C_l is of type (i) or (ii) in the definition of weakly antimonotone clauses then satisfies the conditions (bi) or (bii) respectively.

■

There exists a obvious dual characterization of the weakly monotone logical relations.

3.1.3 Main Result in Logics

We are now ready to state the main result in logics that we need. It says that all non-basic set of logical relations can express an implication $[\overline{x} \vee \overline{y} \vee z]$ or its dual logical relation $[x \vee y \vee \overline{z}]$.

Theorem 15 *Let S be a finite non-basic set of logical relations then $\{[x \vee y \vee \overline{z}], [\overline{x} \vee \overline{y} \vee z]\} \cap \exists$-Relation$(S) \neq \emptyset$.*

This property will be very important to show the non-learnability of non-basic representation classes. The remainder of this section is devoted to the proof of Theorem 15. The following results are from [19].

Lemma 16 *[19] Let R be a logical relation which is not horn, then $\{[x \not\equiv y], [x \vee y]\} \cap \exists\text{-}Relation(\{R\}) \neq \emptyset$.*

Lemma 17 *[19] Let R be a logical relation which is not antihorn, then $\{[x \not\equiv y], [\overline{x} \vee \overline{y}]\} \cap \exists\text{-}Relation(\{R\}) \neq \emptyset$.*

Corollary 18 *[19] Let S be a finite set of relations which is not horn and not antihorn, then $[x \not\equiv y] \in \exists\text{-}Relation(S)$.*

Lemma 19 *[19] Let S be a finite set of logical relations which is not affine and not bijunctive, then $\exists\text{-}Relation(S \cup \{[x \not\equiv y]\})$ is the set of all logical relations.*

The following lemmas will apply when conditions (ci) or (cii) in Lemma 14 fail.

Lemma 20 *Let R be a logical function of rank $k \geq 2$. Suppose that there is a k-compatible assignment $t \notin R$ that contains at least two 0. Then $\{[x \not\equiv y], [x \vee y]\} \cap Relation(\{R\}) \neq \emptyset$.*

Proof.
Let $V' = \{x_1, \ldots, x_k\}$ be a set of k variables. Let $1 \leq i < j \leq k$ such that $t[i] = t[j] = 0$. The assignment obtained by flipping one of bits i, j in t belongs to R by the k-compatibility of t. Then $[R(x_1, \ldots, x_k) \left[\begin{smallmatrix} x_l \in V' - \{x_i, x_j\} \\ t[x_l] \end{smallmatrix} \right]] \in \{[x \not\equiv y], [x \vee y]\}$. ■

Lemma 21 *Let R be a logical relation of rank $k \geq 3$. Suppose that there is a k-compatible assignment $t \notin R$ that contains exactly one 0. Then $\{[\overline{x} \vee \overline{y} \vee z], [x \not\equiv y], [x \vee y]\} \cap \exists\text{-}Relation(\{R\}) \neq \emptyset$.*

Proof.
For all integers $0 < i \leq k$, for all assignments $t \in \{0,1\}^k$, and for every constant $b \in \{0,1\}$ we define $t_{b \leftarrow j}$ as the assignment obtained from t substituting the jth element by b.

Let $V' = \{x_1, \ldots, x_k\}$ be a set of k variables. Let $x_i \in V'$ be the variable such that $t[i] = 0$. By the k-compatibility of t we have $1^k \in R$ and $t_{0 \leftarrow j} \in R$ for all $j \neq i$.

For all $t' \in \{0,1\}^k$ such that $t' \neq t$ and $t'[i] = 0$, t' can be expressed as $t' = \bigwedge_{j \neq i, t'[j]=0} t_{0 \leftarrow j}$, then we can assume $t' \in R$, otherwise by Lemma 12 R is not horn and by Lemma 16 $\{[x \not\equiv y], [x \vee y]\} \cap \exists\text{-Relation}(\{R\}) \neq \emptyset$.

At this point, we can show that a simple implication (i.e. $[x \to y]$ can be generated from R. We study two cases: If there is no $t' \in R$ such that $t' \neq 1^k$ and $t'[i] = 1$ then let $1 \leq j \neq i \leq k$ be any integer. Let x_l be any variable in $V \perp \{x_i, x_j\}$, then $(\exists x_l R(x_1, \ldots, x_k) \begin{bmatrix} V' - \{x_i, x_j\} \\ x_l \end{bmatrix}) \equiv (\overline{x_i} \vee x_j)$. Otherwise, let $t' \in R$ be an assignment such that $t' \neq 1^k$ and $t'[i] = 1$, let $W \subset V'$ be the set of variables $x_j \neq x_i$ such that $t'[j] = 1$, and let $x_l \in V \perp \{x_i\}$ be any variable different from x_i. Then $(R \begin{bmatrix} W, & V' - W - \{x_i\} \\ 1, & x_l \end{bmatrix}) \equiv (\overline{x_l} \vee x_i)$.

The double implication (i.e. $[x \wedge y \to z]$) follows immediately: Let $x_j \in V'$ be a variable not equal to x_i and let $x_l, x_m \in V \perp \{x_i, x_j\}$. Then we have $(\exists x_i R(x_1, \ldots, x_k) \wedge (\overline{x_i} \vee x_m) \begin{bmatrix} V' - \{x_i, x_j\} \\ x_l \end{bmatrix}) \equiv (\overline{x_j} \vee \overline{x_l} \vee x_m)$. ■

The following result follows from Lemmas 20 and 21.

Lemma 22 *Let R be a logical relation of rank k that is not weakly antimonotone. Then $\{[x \not\equiv y], [\overline{x} \vee \overline{y} \vee z], [x \vee y]\} \cap \exists\text{-}Relation(\{R\}) \neq \emptyset$.*

Proof.
Let $T \subset \{1, \ldots, k\}$ and $t \notin R$ an assignment such that condition (c) in Lemma 14 is falsified. There are two cases according what condition is falsified. If condition (ci) is falsified then t contains at least two zeros and by Lemma 20 $\{[x \not\equiv y], [x \vee y]\} \cap \text{Relation}(\{R|T\}) \neq \emptyset$. If condition (cii) is falsified then by Lemma 21 $\{[x \not\equiv y], [\overline{x} \vee \overline{y} \vee z], [x \vee y]\} \cap \exists\text{-Relation}(\{R|T\}) \neq \emptyset$. ■

By duality, we have:

Lemma 23 *Let R be a logical relation of rank k that is not weakly monotone. Then $\{[x \not\equiv y], [x \vee y \vee \overline{z}], [\overline{x} \vee \overline{y}]\} \cap \exists\text{-}Relation(\{R\}) \neq \emptyset$.*

Directly from Lemmas 22 and 23 we have:

Corollary 24 *Let S be a finite set of logical relations that is not weakly monotone and not weakly antimonotone, then $\{[x \not\equiv y], [\overline{x} \vee \overline{y} \vee z], [x \vee y \vee \overline{z}]\} \cap \exists\text{-}Relation(S) \neq \emptyset$.*

And now Theorem 15 follows from Corollary 24 and Lemma 19. ■

3.2 Basic Classes are Polynomially Exactly Learnable

In this section we show positive learnability results for the basic classes. To show the positive results we prove that in a certain sense each of these classes is embedded in a known learnable class and therefore learnable using the algorithm for the more general class. We start by a formal definition of embedding between classes and a simple observation:

Definition 25 *Let $\mathcal{C}, \mathcal{C}'$ be representations of concepts. We say that $\mathcal{C}$ is* embedded *in $\mathcal{C}'$ if there exists a polynomial p such that for every concept name $u \in X$ there exists some concept name $u' \in X$ such that $|u'| \leq p(|u|)$ and $K_{\mathcal{C}}(u) = K_{\mathcal{C}'}(u')$.*

Observation 26 *Let $\mathcal{C}$ be a representation class which is polynomially exactly learnable with equivalence queries in $\mathcal{H}$. For every representation class $\mathcal{C}'$ embedded in C, $\mathcal{C}'$ is polynomially exactly learnable with equivalence queries in $\mathcal{H}$.*

We only need a list of learnable classes and to prove that every basic class is embedded in any of them. The only learnable classes that we need are:

Theorem 27 *[1, 20] For all integers $k \geq 0$ the class $\mathcal{C}_{k-CNF}$ is polynomially exactly learnable with $O(n^k)$ proper equivalence queries.*

Theorem 28 *[5] Let be $\mathcal{C}_{Af}$ the set of formulas formed by conjunctions of equations over the two-element field $\{0,1\}$. The class $\mathcal{C}_{Af}$ is polynomially exactly learnable with $n+1$ proper equivalence queries.*

In the previous theorems and through the rest of this paper, n denotes the number of variables.

These two classes satisfy another nice property: all elements in these classes have polynomial size in n. Specifically:

- For every u concept in $\mathcal{C}_{k-CNF}$, $|u| \in O(n^k \log n)$. (The $\log n$ factor appears because writing down a variable name requires $\log n$ bits).
- For every u concept in $\mathcal{C}_{Af}$, $|u| \in O(n^2 \log n)$.

If we want to prove that some representation class is embedded in another class that satisfies the previous property then we do not need to worry about the length of the representation because this condition is satisfied automatically. From this observation and from Corollary 11 we can derive the following results:

Claim 29 *Let S be a finite set of logical relations. The following conditions hold:*

- *If S is bijunctive then $\mathcal{C}_{\exists\forall\text{-}Formula(S)}$ is embedded in $\mathcal{C}_{2-CNF}$.*
- *If S is weakly monotone of degree k then $\mathcal{C}_{\exists\forall\text{-}Formula(S)}$ is embedded in $\mathcal{C}_{k-CNF}$.*
- *If S is weakly antimonotone of degree k then $\mathcal{C}_{\exists\forall\text{-}Formula(S)}$ is embedded in $\mathcal{C}_{k-CNF}$.*
- *If S is affine then $\mathcal{C}_{\exists\forall\text{-}Formula(S)}$ is embedded in $\mathcal{C}_{Af}$.*

We are now ready for the positive learnability results. From the previous claim and Observation 26 we can derive the following theorem:

Theorem 30 *Let S be a finite set of logical relations. The following conditions hold:*

- *If S is bijunctive then $\mathcal{C}_{\exists\forall\text{-}Formula(S)}$ is polynomially exactly learnable with $O(n^2)$ equivalence queries in $\mathcal{C}_{2-CNF}$.*
- *If S is weakly monotone of degree k then $\mathcal{C}_{\exists\forall\text{-}Formula(S)}$ is polynomially exactly learnable with $O(n^k)$ equivalence queries in $\mathcal{C}_{k-CNF}$.*
- *If S is weakly antimonotone of degree k then $\mathcal{C}_{\exists\forall\text{-}Formula(S)}$ is polynomially exactly learnable with $O(n^k)$ equivalence queries in $\mathcal{C}_{k-CNF}$.*
- *If S is affine then $\mathcal{C}_{\exists\forall\text{-}Formula(S)}$ is polynomially exactly learnable with $n+1$ equivalence queries in $\mathcal{C}_{Af}$.*

3.3 Boolean Circuits are pwm-reducible to Non-basic Classes

Let $\mathcal{C}_{BC}$ be the class of boolean circuits with $\{\vee, \wedge, \neg\}$ gates and $\mathcal{C}_{MBC}$ the class of monotone boolean circuits, i.e., with no '$\neg$' gates. We consider the input variables as gates of fan-in 0. In [3] it is showed that under some cryptographic assumptions boolean circuits are not polynomially predictable with membership queries:

Theorem 31 *[3] If there exist public key encryption systems secure against CC-attack then $\mathcal{C}_{BC}$ is not polynomially predictable with membership queries.*

To show the pwm-reduction from boolean circuits to non-basic classes we need the following lemmas:

Lemma 32 *(Standard technique) $\mathcal{C}_{BC} \leq_{pwm} \mathcal{C}_{MBC}$.*

Lemma 33 $\mathcal{C}_{MBC} \leq_{pwm} \exists$*-Formula*$(\{\overline{x} \vee \overline{y} \vee z\})$

Proof.
Let C be a monotone boolean circuit where m is its number of gates. We can represent each gate j in C by a variable x_j and construct the following formula:

$$F = \exists x_{i_1} \ldots \exists x_{i_r} \varphi \begin{bmatrix} x_o \\ 0 \end{bmatrix}$$

where o is the output gate, $\{i_j : 1 \leq j \leq r\}$ is the set that contains exactly all the $\wedge$-gates and $\vee$-gates except the output gate o, and φ is a conjunction that contains exactly the following clauses:

- For each $\wedge$-gate i, φ contains the clause $\overline{x_j} \vee \overline{x_k} \vee x_i$ where j and k are the ancestors of i.
- For each $\vee$-gate i, φ contains the clauses $\overline{x_j} \vee \overline{x_j} \vee x_i$ and $\overline{x_k} \vee \overline{x_k} \vee x_i$ where j and k are the ancestors of i.

It is easy to show that C and F denote complementary boolean functions.

For all naturals n and s and for every concept representation $u \in \mathcal{C}_{MBC}$ such that $|u| \leq s$, let C be the monotone boolean circuit represented by u and let u' be the representation of the boolean formula F formed from C

as shown before. We define $g(s,n,u) = u'$ if the number of input variables in C is n and the representation of the constant formule *True* otherwise. For all $x \in X^{[n]}$ we define $f(s,n,x) = h(s,n,x) = x$. Clearly, f, g and h satisfy the conditions (1), (2) and (3) in Definition 1 and therefore define a pwm-reduction. ■

This reduction is very similar to the reduction from the evaluation of monotone boolean circuit problem to the horn satisfiability problem, used to show P-completeness of the latter problem ([10], pages 167-168).

By duality we have:

Lemma 34 $\mathcal{C}_{MBC} \leq_{pwm} \mathcal{C}_{\exists\text{-}Formula(\{x \vee y \vee \overline{z}\})}$

We put the previous lemmas together with Theorems 15 and 31 and we obtain the following result:

Corollary 35 *Let S a finite non-basic set of logical relations, then:*

a.- The set $\exists$-Formula(S) contains $[x \vee y \vee \overline{z}]$ or $[\overline{x} \vee \overline{y} \vee z]$.

b.- The class $\mathcal{C}_{MBC}$ is pwm-reducible to $\mathcal{C}_{\exists\text{-}Formula(S)}$.

c.- The class $\mathcal{C}_{BC}$ is pwm-reducible to $\mathcal{C}_{\exists\text{-}Formula(S)}$.

d.- The class $\mathcal{C}_{\exists\text{-}Formula(S)}$ is not polynomially predictable with membership queries under the assumption that public key encryption systems secure against CC-attack exist.

Finally, Theorem 4 follows from Theorem 30 and Corollary 35. ■

4 Acknowledgments

First, I am very grateful to Ricard Gavaldà for very helpful discussions and extensive comments on this work and its presentation and for pointing out the relationship given in Lemma 33. I would also like to thank my officemates Jorge Castro, Carlos Domingo and David Guijarro for their helpful comments.

References

[1] D. Angluin. *Queries and concept learning*, Machine Learning 2 (1988), 319-342.

[2] D. Angluin, M. Frazier, and L. Pitt, *Learning conjunctions of Horn clauses*, Machine Learning 9 (1992), 147-164.

[3] D. Angluin, and M. Kharitonov. *When won't membership queries help?*, J. Comput. System Sci. 50 (1995) 336-355.

[4] U. Berggren. *Linear time deterministic learning of k-term DNF*, in "6th Annual Workshop on Computational Learning Theory" (1993), 37-40.

[5] Z. Chen, and S. Homer. *On Learning counting functions with queries*, in "7th Annual ACM Conference on Computational Learning Theory, COLT'94" (1994), 218-227.

[6] N. Creignou, and M. Hermann. *Complexity of Generalized Satisfiability Counting Problems*, Information and Computation 125, (1996), 1-12.

[7] N. Creignou. *A Dichotomy Theorem for Maximum Generalized Satisfiability Problems*, Journal of Computer and System Sciences 51(3), (1995) 511-522.

[8] V. Dalmau. In preparation.

[9] S. Fortune, J. Hopcroft, and J. Wyllie. *The directed subgraph homeomorphism problem*, Theor. Comp. Sci. 81(2) (1980), 111-121.

[10] R. Greenlaw, H. J. Hoover, and W. L. Ruzzo, *Limits to parallel computation: P-completeness theory*, Oxford University Press (1995).

[11] P. Hell, and N. Nešetřil. *On the complexity of H-coloring*, J. Combin. Theory Ser. B, 48 (1990), 92-110.

[12] D. Kavvadias, and M. Sidderi. *The Inverse Satisfiability Problem*, in "2nd Annual International Conference on Computing and Combinatorics, COCOON'96" (1996), 250-259.

[13] M. Kearns, and L. G. Valiant, *Cryptographic limitations on learning boolean formulae and finite automata*, J. ACM 41(1) (1994), 67-95.

[14] S. Khanna, M. Sudan, and L. Trevisan. *Constraint Satisfaction: The Approximability of Minimization Problems.* To appear in "Proceedings of the 12th IEEE Conference on Computational Complexity." (1997).

[15] S. Khanna, M. Sudan, and P. Williamson. *A complete classification of the approximability of maximization problems derived from boolean constraint satisfaction.* To appear in "Proceedings of the 29th Annual ACM Symposium on the Theory of Computing". (1997).

[16] W. Mass, and G. Turán. *On learnability and predicate logic.* In "Bar-Ilan Symposium on the Foundations of Artificial Intelligence BISFAI '95" (1995), 75-85.

[17] C. H. Papadimitriou. *Computational complexity*, Addison-Wesley (1994).

[18] L. Pitt, and M. Warmuth. *Prediction preserving reducibility*, J. Comput. System Sci. 41 (1990), 430-467.

[19] T. J. Schaefer. *The complexity of satisfiability problems*, in "10th Annual ACM Symposium on Theory of Computing" (1978), 216-226.

[20] L. G. Valiant. *A theory of the learnable*, Comm. ACM 27 (1984), 1134-1142.