

Canonical Horn Representations and Query Learning^{*}

Marta Arias¹ and José L. Balcázar²

¹ LARCA Research Group, Departament LSI
Universitat Politècnica de Catalunya, Spain
marias@lsi.upc.edu

² Departamento de Matemáticas, Estadística y Computación
Universidad de Cantabria, Spain
joseluis.balcazar@unican.es

Abstract. We describe an alternative construction of an existing canonical representation for definite Horn theories, the *Guigues-Duquenne* basis (or GD basis), which minimizes a natural notion of implicational size. We extend the canonical representation to general Horn, by providing a reduction from definite to general Horn CNF. We show how this representation relates to two topics in query learning theory: first, we show that a well-known algorithm by Angluin, Frazier and Pitt that learns Horn CNF always outputs the GD basis independently of the counterexamples it receives; second, we build strong polynomial certificates for Horn CNF directly from the GD basis.

1 Introduction

The present paper is the result of an attempt to better understand the classic algorithm by Angluin, Frazier, and Pitt [2] that learns propositional Horn formulas. A number of intriguing questions remain open regarding this algorithm; in particular, we were puzzled by the following one: we expected the outcome of the algorithm to depend on the specific counterexamples received as answers, along the run, because it is easily seen that, in general, each equivalence query depends heavily upon the counterexamples received so far, and, in fact, the outcome of the algorithm is an equivalence query: the first one with positive answer. However, if hypotheses are expressed in terms of implications (or “meta-clauses” in the terminology of [2]), attempts at providing an example of different counterexamples leading to equivalent but different outputs consistently fail.

In this paper we prove that such attempts must in fact fail: we describe a canonical representation of Horn functions in terms of implications, and show that the algorithm of [2] always outputs this particular representation. It turns out that this canonical representation is well-known in the field of Formal Concepts, and bears the name of the authors that, to the best of our knowledge,

^{*} Work partially supported by MICINN projects SESAAME-BAR (TIN2008-06582-C03-01) and FORMALISM (TIN2007-66523).

first described it: the *Guigues-Duquenne basis* or GD basis [8, 15]. In addition, the GD basis has the important quality of being of minimum size.

However, the GD basis is defined for definite Horn formulas only. Fortunately, we are able to extend the notion of GD basis to general Horn formulas by means of a reduction from general to definite Horn formulas. This reduction allows us to lift the characterization of the output of AFP as the generalized GD basis. Furthermore, the generalized GD representation provides the basis for building strong polynomial certificates with $p(m, n) = m$ and $q(m, n) = \binom{m+1}{2} + m + 1 = \binom{m+2}{2}$ for the class of general Horn formulas, extending a similar construction from [4] which applied only to definite Horn.

2 Preliminaries

We start by reviewing basic definitions and results from Horn logic. We work within the standard framework in logic, where one is given an indexable set of propositional variables of cardinality n , boolean functions are subsets of the boolean hypercube $\{0, 1\}^n$, and these functions are represented by logical formulas over the variable set in the standard way. Binary strings of length n assign a boolean value for each variable, and are therefore called assignments; given any boolean function or formula H , the fact that assignment x makes it true (or “satisfies” it) is denoted $x \models H$. Following the standard overloading of the operator, $H \models H'$ means that, for every assignment x , if $x \models H$ then $x \models H'$.

A literal is a variable or its negation. A conjunction of literals is a *term*, and if none is negated it is a *positive term*; we often identify positive terms and mere sets of variables; in fact, we switch back and forth between set-based notation and assignments. We denote terms, or equivalently subsets of variables, with greek letters and assignments with letters from the end of the alphabet. We may abuse notation at times and it should be understood that if we use a subset α when an assignment is expected, it is to be interpreted as the assignment that sets to 1 exactly those variables in α . We denote this explicitly when necessary by $x = \text{BITS}(\alpha)$. Similarly, if we use an assignment x where a subset of variables is expected, it is to be understood that we mean the set of variables that are set to 1 in x . We denote this explicitly by $\alpha = \text{ONES}(x)$. Clearly, we have a bijection between sets of propositional variables and assignments, and $x = \text{BITS}(\text{ONES}(x))$ and $\alpha = \text{ONES}(\text{BITS}(\alpha))$ for all assignments x and variable sets α . The following lemma is a direct consequence of the definition of satisfiability over conjunctions:

Lemma 1. $x \models \alpha$ iff $\alpha \subseteq \text{ONES}(x)$ iff $\text{BITS}(\alpha) \leq x$. □

2.1 Horn Logic

In this paper we are only concerned with Horn functions, and their representations using conjunctive normal form (CNF). A Horn CNF formula is a conjunction of Horn clauses. A clause is a disjunction of literals. A clause is *definite*

Horn if it contains exactly one positive literal, and it is *negative* if all its literals are negative. A clause is *Horn* if it is either definite Horn or negative. Since we are dealing with Horn functions only, we drop the “Horn” adjective frequently.

Horn clauses are generally viewed as implications where the negative literals form the antecedent of the implication (a positive term), and the singleton consisting of the positive literal, if it exists, forms the consequent of the clause. Note that both can be empty; if the consequent is empty, then we are dealing with a negative Horn clause. Furthermore, we allow our representations of Horn CNF to deviate slightly from the standard in that we represent clauses sharing the same antecedent together in one implication. Namely, an implication $\alpha \rightarrow \beta$, where both α and β are possibly empty sets of propositional variables, is to be interpreted as the conjunction of definite Horn clauses $\bigwedge_{b \in \beta} \alpha \rightarrow b$ if $\beta \neq \emptyset$, and as the negative clause $\alpha \rightarrow \square$ if $\beta = \emptyset$.³ A semantically equivalent interpretation is to see both sets of variables α and β as positive terms; the Horn formula in its standard form is obtained by distributivity on the variables of β .

Example 1. Let $(\neg a \vee b) \wedge (\neg a \vee c) \wedge (\neg a \vee \neg b)$ be a standard representation of a Horn function over the variable set $\{a, b, c\}$. In standard implicational form this expression becomes $(a \rightarrow b) \wedge (a \rightarrow c) \wedge (ab \rightarrow \square)$. In our compact representation we write $(a \rightarrow bc) \wedge (ab \rightarrow \emptyset)$. All these represent the same boolean function. When clear from context, we drop the set representation in our examples.

We refer to our generalized notion of conjunction of clauses sharing the antecedent as *implication*; the term *clause* retains its classical meaning (namely, a disjunction of literals). Notice that an implication may not be a clause, e.g. $(a \rightarrow bc)$ corresponds in classical notation to the formula $\neg a \vee (b \wedge c)$. Thus, $(a \rightarrow bc)$, $(ab \rightarrow c)$ and $(ab \rightarrow \emptyset)$ are Horn implications but only the latter two are Horn clauses. Furthermore, we often use sets to denote conjunctions, as we do with positive terms, also at other levels: a generic (implicational) CNF $\bigwedge_i (\alpha_i \rightarrow \beta_i)$ is often denoted in this text by $\{(\alpha_i \rightarrow \beta_i)\}_i$. Parentheses are mostly optional and generally used for ease of reading.

Clearly, an assignment $x \in \{0, 1\}^n$ satisfies the implication $\alpha \rightarrow \beta$, denoted $x \models \alpha \rightarrow \beta$, if it either fails the antecedent or satisfies the consequent, that is, $x \not\models \alpha$ or $x \models \beta$ respectively, where now we are interpreting both α and β as positive terms. By convention, $x \not\models \emptyset$ for any assignment x .

A Horn function admits several syntactically different Horn CNF representations; in this case, we say that these representations are equivalent. Such representations are also known as *theories* or *bases* for the boolean function they represent. The *size* of a Horn function is the minimum number of clauses that a Horn CNF representing it must have. The *implication size* of a Horn size is defined analogously, but allowing formulas to have implications instead of clauses.

³ Notice that this differs from an alternative existing interpretation [14], obsolescent by now, in which $\alpha \rightarrow \beta$ represents the clause $(\neg x_1 \vee \dots \vee \neg x_k \vee y_1 \vee \dots \vee y_{k'})$, where $\alpha = \{x_1, \dots, x_k\}$ and $\beta = \{y_1, \dots, y_{k'}\}$. Though identical in syntax, the semantics are different; in particular, ours can only represent a conjunction of definite Horn clauses whereas the other represents a general possibly non-Horn clause.

Clearly, every clause is an implication, and thus the implication size of a given Horn function is always at most that of its standard size as measured in the number of clauses. Not all boolean functions are Horn. The following semantic characterization is a well-known classic result of [13, 11], proved in the context of propositional Horn logic e.g. in [12]:

Theorem 1. *A boolean function admits a Horn CNF basis if and only if the set of assignments that satisfy it is closed under bit-wise intersection.* \square

2.2 Definite Horn Functions

Forward chaining. We describe the well-known method of *forward chaining* for definite Horn functions. Notice that it directly extends to our compressed representation where consequents of clauses can contain more than one variable. Given a definite Horn CNF $H = \{\alpha_i \rightarrow \beta_i\}_i$ and a subset of propositional variables α , we construct chains of subsets of propositional variables $\alpha = \alpha^{(0)} \subset \alpha^{(1)} \subset \dots \subset \alpha^{(k)} = \alpha^*$. Each $\alpha^{(i)}$ with $i > 0$ is obtained from its predecessor $\alpha^{(i-1)}$ in the following way: if $\text{BITS}(\alpha^{(i-1)})$ satisfies all implications in H , then the process can stop with $\alpha^{(i-1)} = \alpha^*$. If, on the other hand, $\text{BITS}(\alpha^{(i-1)})$ violates some implication $\alpha_j \rightarrow \beta_j \in H$, then $\alpha^{(i)}$ is set to $\alpha^{(i-1)} \cup \alpha_j$.

Similarly, one can construct an increasing chain of assignments (using the usual bit-wise partial order of hypercubes) $x = x^{(0)} < x^{(1)} < \dots < x^{(k)} = x^*$ using our bijection $\alpha^{(i)} = \text{ONES}(x^{(i)})$ and $x^{(i)} = \text{BITS}(\alpha^{(i)})$ for all i .

Theorem 2. *The objects x^* and α^* are well-defined and computed by the forward chaining procedure regardless of the order in which implications in H are chosen. The forward chaining process finishes in linear time in $|H|$ and n . Moreover, x^* and α^* depend only on the underlying function being represented, and not on the particular choice of representation H ; and for each $x^{(i)}$ and $\alpha^{(i)}$ along the way, we have that $(x^{(i)})^* = x^*$ and $(\alpha^{(i)})^* = \alpha^*$.* \square

It is well-known that the forward chaining procedure is a sound and complete deduction method for definite Horn CNF [7]:

Theorem 3. *Let h be a definite Horn function, and let α be an arbitrary variable subset. Then $h \models \alpha \rightarrow b$ if and only if $b \in \alpha^*$.* \square

Closure operator and equivalence classes. It is easy to see that the \star operator is extensive (that is, $x \leq x^*$ and $\alpha \subseteq \alpha^*$), monotonic (if $x \leq y$ then $x^* \leq y^*$, and if $\alpha \subseteq \beta$ then $\alpha^* \subseteq \beta^*$) and idempotent ($x^{**} = x^*$, and $\alpha^{**} = \alpha^*$) for all assignments x, y and variable sets α, β ; that is, \star is a closure operator [4]. Thus, we refer to x^* as the *closure* of x w.r.t. a definite Horn function.

It should be always clear from the text with respect to what definite Horn function we are taking the closure, hence it is omitted from the notation used. An assignment x is said to be *closed* iff $x^* = x$, and similarly for variable sets. Furthermore, it is not hard to see that closed elements are always positive (by

construction via the forward chaining procedure, they must satisfy all implications), and assignments that are not closed are always negative (similarly, they must violate some implication). This closure operator induces a partition over the set of assignments $\{0, 1\}^n$ in the following straightforward way: two assignments x and y belong to the same class if $x^* = y^*$. This notion of equivalence class carries over as expected to the power set of propositional variables: the subsets α and β belong to the same class if $\alpha^* = \beta^*$. It is worth noting that each equivalence class consists of a possibly empty set of assignments that are not closed and a single closed set, its representative.

3 The Guigues-Duquenne Basis for Definite Horn

In this section we characterize and show how to build a canonical basis for definite Horn functions that is of minimum implication size. This canonical representation has been introduced before as the Guigues-Duquenne basis (the *GD basis*) [8]. Here, we introduce it in a form that is, to our knowledge, novel, although it is relatively close to the approach of [15]. Moreover, we base the construction of the GD basis on the notion of *saturation*, a notion that has been used already in the context of Horn functions and seems very natural [5, 4].

First, we prove several interesting properties that serve as the basis for our work.

Definition 1. Let $B = \{\alpha_i \rightarrow \beta_i\}_i$ be a basis for some definite Horn function. Then, B is left-saturated if the following 2 conditions hold:

1. $\text{BITS}(\alpha_i) \not\models \alpha_i \rightarrow \beta_i$, for all i
2. $\text{BITS}(\alpha_i) \models \alpha_j \rightarrow \beta_j$, for all $i \neq j$

Alternatively, it can be more succinctly described by the following equivalence: a basis $\{\alpha_i \rightarrow \beta_i\}_i$ is left-saturated if $i = j \Leftrightarrow \text{BITS}(\alpha_i) \not\models \alpha_j \rightarrow \beta_j$.

Definition 2. Let $B = \{\alpha_i \rightarrow \beta_i\}_i$ be a basis for some definite Horn function. Then, B is right-saturated if for all i , $\beta_i = \alpha_i^*$. Accordingly, we denote right-saturated bases with $\{\alpha_i \rightarrow \alpha_i^*\}_i$.

Definition 3. We say that a basis B is saturated iff it is left- and right-saturated.

Lemma 2. If B is left-saturated then B is irredundant.

Proof. Suppose that B is left-saturated but redundant. Then it contains an implication $(a \rightarrow b)$ that is redundant, where both a and b are now variable sets. By definition of left-saturation, we have that $\text{BITS}(a) \models (\alpha \rightarrow \beta)$ for implications $(\alpha \rightarrow \beta)$ other than $(a \rightarrow b)$. Thus $\text{BITS}(a) \models B \setminus \{(a \rightarrow b)\}$, contradicting the fact that B and $B \setminus \{(a \rightarrow b)\}$ are equivalent. \square

Lemma 3. Let $B = \{\alpha_i \rightarrow \beta_i\}_i$ be an irredundant basis. Then, $\text{BITS}(\alpha_i) \not\models \alpha_i \rightarrow \beta_i$ for all i .

Proof. We are dealing with definite implications, therefore no β_i can be empty. Moreover, the fact that B is irredundant implies that some variable in β_i must be from outside α_i , otherwise the implication would be redundant. Thus we have that $\text{BITS}(\alpha_i) \models \alpha_i$ but $\text{BITS}(\alpha_i) \not\models \beta_i$ because the variable of β_i outside of α_i is set to 0 in $\text{BITS}(\alpha_i)$ by definition. \square

Lemma 4. *Let $B = \{\alpha_i \rightarrow \alpha_i^*\}_i$ be a saturated basis for some definite Horn function h . Then, $\text{BITS}(\alpha_i) \not\models h$ and $\text{BITS}(\alpha_i^*) \models h$ hold for all i .*

Proof. It must hold that $\alpha_i \subset \alpha_i^*$ for all i , since otherwise B would be redundant, and hence could not be saturated. This implies that α_i is not closed and α_i^* is, and the lemma follows. \square

Lemma 5. *Let $B = \{\alpha_i \rightarrow \alpha_i^*\}_i$ be an irredundant and right-saturated implicational basis. Then, the following hold for all $i \neq j$*

1. $\alpha_i \neq \alpha_j$
2. $\alpha_i \subseteq \alpha_j \Rightarrow \alpha_i \subset \alpha_j$

Proof. Follows simply from the fact that if $\alpha_i = \alpha_j$ with $i \neq j$, then the implications $\alpha_i \rightarrow \alpha_i^*$ and $\alpha_j \rightarrow \alpha_j^*$ would, in fact, be the same implication, and thus B is redundant. The second statement is a direct consequence of the first. \square

Lemma 6. *Let $B = \{\alpha_i \rightarrow \alpha_i^*\}_i$ be an irredundant, right-saturated basis. Then, B is left-saturated if and only if the following implication is true for all $i \neq j$:*

$$\alpha_i \subset \alpha_j \Rightarrow \alpha_i^* \subseteq \alpha_j.$$

Proof. Assume that the implication $\alpha_i \subset \alpha_j \Rightarrow \alpha_i^* \subseteq \alpha_j$ holds for all $i \neq j$. We show that B must be left-saturated, namely, that the following equivalence must hold: $i = j \Leftrightarrow \text{BITS}(\alpha_i) \not\models \alpha_j \rightarrow \alpha_j^*$. If $i = j$, Lemma 3 guarantees that $\text{BITS}(\alpha_i) \not\models \alpha_i \rightarrow \alpha_i^*$. For the other direction, assume by way of contradiction that $\text{BITS}(\alpha_i) \not\models \alpha_j \rightarrow \alpha_j^*$, for some $i \neq j$. This happens if $\text{BITS}(\alpha_i) \models \alpha_j$ (namely, $\alpha_j \subseteq \alpha_i$ by Lemma 1 and thus $\alpha_j \subset \alpha_i$ by Lemma 5) but $\text{BITS}(\alpha_i) \not\models \alpha_j^*$ (namely, $\alpha_j^* \not\subseteq \alpha_i$). But this contradicts the implication $\alpha_i \subset \alpha_j \Rightarrow \alpha_i^* \subseteq \alpha_j$.

Finally, assume that $\alpha_i \subset \alpha_j$ for some $i \neq j$, and that B is left-saturated, which implies by definition that $\text{BITS}(\alpha_j) \models \alpha_i \rightarrow \alpha_i^*$: then $\alpha_i^* \subseteq \alpha_j$ follows. \square

The following Lemma is a variant of a result of [15] translated into our notation. We include the proof that is, in fact, missing from [15].

Lemma 7. *Let $B = \{\alpha_i \rightarrow \alpha_i^*\}_i$ be a saturated basis for a definite Horn function. Then for all i and β it holds that $(\beta \subseteq \alpha_i \text{ and } \beta^* \subset \alpha_i^*) \Rightarrow \beta^* \subseteq \alpha_i$.*

Proof. Let us assume that the conditions of the implication are true, namely, that $\beta \subseteq \alpha_i$ and $\beta^* \subset \alpha_i^*$. We proceed by cases: if β is closed, then $\beta^* = \beta$ and the implication is trivially true since $\beta \subseteq \alpha_i$ clearly implies $\beta^* \subseteq \alpha_i$ when $\beta^* = \beta$. Otherwise, β is not closed. Let $\beta = \beta^{(0)} \subset \beta^{(1)} \subset \dots \subset \beta^{(k)} = \beta^*$ be the series of elements constructed by the forward chaining procedure described

in Section 2.2. We argue that if $\beta^{(l)} \subseteq \alpha_i$ and $\beta^{(l)} \subset \beta^*$, then $\beta^{(l+1)} \subseteq \alpha_i$ as well. By repeatedly applying this fact to all the elements along the chain, we arrive at the desired conclusion, namely, $\beta^* \subseteq \alpha_i$. Let $\beta^{(l)}$ be such that $\beta^{(l)} \subseteq \alpha_i$ and $\beta^{(l)} \subset \beta^*$. Thus $\beta^{(l)}$ violates some implication $(\alpha_k \rightarrow \alpha_k^*) \in B$. Our forward chaining procedure assigns $\beta^{(l+1)}$ to $\beta^{(l)} \cup \alpha_k^*$. The following inequalities hold: $\alpha_k \subseteq \beta^{(i)}$ because $\beta^{(i)} \not\models \alpha_k \rightarrow \alpha_k^*$, $\beta^{(i)} \subseteq \alpha_i$ by assumption; hence $\alpha_k \subseteq \alpha_i$. Using Lemma 6, and noticing the fact that, actually, $\alpha_k \subset \alpha_i$ since $\beta^{(i)} \subset \alpha_i$ (otherwise we could not have $\beta^* \subset \alpha_i^*$), we conclude that $\alpha_k^* \subseteq \alpha_i$. We have that $\alpha_k^* \subseteq \alpha_i$ and $\beta^{(i)} \subseteq \alpha_i$ so that $\beta^{(i+1)} = \beta^{(i)} \cup \alpha_k^* \subseteq \alpha_i$ as required. \square

We arrive at our version of the GD basis:

Theorem 4. *Definite Horn functions have a unique saturated basis.*

Proof. Let B_1 and B_2 be two equivalent and saturated bases. Let $(a \rightarrow a^*)$ be an arbitrary comparable pair in B_1 . We show that $(a \rightarrow a^*) \in B_2$ as well. By symmetry, this implies that $B_1 = B_2$.

By Lemma 3, we have that $\text{BITS}(a) \not\models B_1$ and thus $\text{BITS}(a)$ must violate some implication $(b \rightarrow b^*) \in B_2$, hence it must hold that $b \subseteq a$ but $b^* \not\subseteq a$. The rest of the proof is concerned with showing that assuming $b \subset a$ leads to a contradiction. If so, then $b = a$ and thus $(a \rightarrow a^*) \in B_2$ as well as desired.

Let us assume then that $b \subset a$ so that, by monotonicity, $b^* \subseteq a^*$. If $b^* \subset a^*$, then we can use Lemma 7 with $\alpha_i = a$ and $\beta = b$ and conclude that $b^* \subseteq a$, contradicting the fact that $\text{BITS}(a) \not\models (b \rightarrow b^*)$. Thus, it should be that $b^* = a^*$. Now, consider $(b \rightarrow a^*) \in B_2$. Clearly b is negative and thus it must violate some implication $(c \rightarrow c^*) \in B_1$, namely, $c \subseteq b$ but $c^* \not\subseteq b$. If $c = b$, then we have $(a \rightarrow a^*) \in B_1$ and $(c \rightarrow c^*)$ with $c \subset a$ and $c^* = b^* = a^*$ contradicting the fact that B_1 is irredundant. Thus, $c \subset b$ and so $c^* \subseteq b^*$. If $c^* \subset b^*$ then we use Lemma 7 as before but with $\alpha_i = b$ and $\beta = c$ and we conclude that $c^* \subseteq b$. Again, this means that $b \models (c \models c^*)$ contradicting the fact that b violates this implication. So the only remaining case is $c^* = b^*$ but this means that we have the implications $(a \rightarrow a^*) \in B_1$ and $(c \rightarrow c^*) \in B_1$ with $c \subset a$ but $a^* = c^*$ which again makes B_1 redundant. \square

3.1 Constructing the GD Basis

So far, our definition of saturation only tests whether a given basis is actually saturated; we study now a saturation process to obtain the GD basis. New definitions are needed. Let H be any Horn CNF, and α any variable subset. Let $H(\alpha)$ be those clauses of H whose antecedents fall in the same equivalence class as α , namely, $H(\alpha) = \{\alpha_i \rightarrow \beta_i \mid \alpha_i \rightarrow \beta_i \in H \text{ and } \alpha^* = \alpha_i^*\}$.

Given a Horn function H and a variable subset α , we introduce a new operator \bullet that we define as follows: α^\bullet is the closure of α with respect to the subset of clauses $H \setminus H(\alpha)$. That is, in order to compute α^\bullet one does forward chaining starting with α but one is not allowed to use the clauses in $H(\alpha)$. This operator has been used in the literature before in related contexts, for example in [15].

Example 2. Let $H = \{a \rightarrow b, a \rightarrow c, c \rightarrow d\}$. Then, $ac^* = abcd$ but $ac^\bullet = acd$ since $H(ac) = \{a \rightarrow b, a \rightarrow c\}$ and we are only allowed to use the clause $c \rightarrow d$ when computing ac^\bullet .

Computing the GD basis of a definite Horn H . The algorithm for constructing the GD basis of an arbitrary definite Horn CNF H is presented in Figure 1 and works as follows. First, saturate every clause $C = \alpha \rightarrow \beta$ in H by replacing it with the implication $\alpha^\bullet \rightarrow \alpha^*$. Then, remove possibly redundant implications, namely: (1) remove implications s.t. $\alpha^\bullet = \alpha^*$, and (2) remove duplicates, and (3) remove subsumed implications, i.e., implications $\alpha^\bullet \rightarrow \alpha^*$ for which there is another implication $\beta^\bullet \rightarrow \beta^*$ s.t. $\alpha^* = \beta^*$ but $\beta^\bullet \subset \alpha^\bullet$.

Let us denote with $GD(H)$ the implicational definite Horn CNF obtained by applying this procedure to input H . Note that this algorithm is designed to work when given a definite Horn CNF both in implicational or standard form.

The procedure can be computed in quadratic time, since finding the closures of antecedent and consequent of each clause can be done in linear time w.r.t. the size of the initial Horn CNF H .

GD(H)	
1	for every clause $C \in H$ \triangleright /* C has antecedent α and consequent β */
2	do $\beta \leftarrow \alpha^*$ \triangleright /* right-saturate C */
3	$\alpha \leftarrow \alpha^\bullet$ \triangleright /* left-saturate C */
4	remove from H clauses $\alpha \rightarrow \beta$ such that $\alpha = \beta$
5	remove from H duplicate clauses
6	remove from H subsumed clauses
	\triangleright /* i.e. remove $\alpha \rightarrow \beta$ if there is an $\alpha' \rightarrow \beta'$ s.t. $\beta^* = \beta'^*$ but $\alpha' \subset \alpha$ */
7	return H

Fig. 1. Constructing the GD basis for definite Horn CNF

Example 3. Following our previous example, let $H = \{a \rightarrow b, a \rightarrow c, c \rightarrow d\}$. Then, $a^\bullet = a$, $a^* = abcd$, $c^\bullet = c$, and $c^* = cd$. Thus our H after the for loop is $\{a \rightarrow abcd, a \rightarrow abcd, c \rightarrow cd\}$, and the final GD basis is $GD(H) = \{a \rightarrow abcd, c \rightarrow cd\}$.

Example 4. Let $H = \{a \rightarrow b, a \rightarrow c, ad \rightarrow e, ab \rightarrow e\}$. We compute the closures of the antecedents: $a^* = abce$, $ad^* = abcde$, and $ab^* = abce$. Therefore, $H(a) = \{a \rightarrow b, a \rightarrow c, ab \rightarrow e\}$, $H(ad) = \{ad \rightarrow e\}$, and $H(ab) = H(a)$. Thus, $a^\bullet = a$, $ad^\bullet = abcde$, and $ab^\bullet = abce$. After saturation of every clause in H , we obtain $H' = \{a \rightarrow abce, a \rightarrow abce, abcde \rightarrow abcde, abce \rightarrow abce\}$. It becomes clear that the third clause was, in fact, redundant. Also, the fourth implication is subsumed by the first two (after right-saturation) and we can group the first and second implications together into a single one. Hence, $GD(H) = \{a \rightarrow abce\}$.

In the remainder of this Section we show that the given algorithm computes the unique saturated representation of its input. First, we need a simple lemma:

Lemma 8. *Let H be any basis for a definite Horn CNF over variables $X = \{x_1, \dots, x_n\}$. For any $\alpha, \beta, \gamma \subseteq X$, the following statements hold:*

1. $\alpha \subseteq \alpha^\bullet \subseteq \alpha^*$
2. If $H \models \beta \rightarrow \gamma$, $\beta \subseteq \alpha^\bullet$ but $\beta^* \subset \alpha^*$, then $\gamma \subseteq \alpha^\bullet$

Proof. The first statement follows from the fact that to get from α to α^\bullet we do forward chaining with the clauses in $H \setminus H(\alpha)$, but to get to α^* we are allowed to use all clauses in H .

This second statement is in fact a variant of Lemma 4 of [15]. To see that it holds, we first prove that $H \setminus H(\alpha) \models \beta \rightarrow \gamma$. Assume w.l.o.g. that H is right-saturated⁴. By assumption, $H \models \beta \rightarrow \gamma$. Therefore, if we do forward chaining starting from β we end up with a variable subset that contains γ , namely $\gamma \subseteq \beta^*$. We need to make sure that no clause in $H(\alpha)$ is used in the forward chaining process, so that $H \setminus H(\alpha) \models \beta \rightarrow \gamma$ as well. But if this were not so, and some clause with an antecedent in the same class as α were used, the forward chaining procedure would add α^* to the closure being computed, and thus we would have that $\alpha^* \subseteq \gamma \subseteq \beta^*$, thus contradicting our assumption. So, no clauses in $H(\alpha)$ can be used and therefore $H \setminus H(\alpha) \models \beta \rightarrow \gamma$.

Finally, to compute α^\bullet we do forward chaining with clauses in $H \setminus H(\alpha)$. Since $\beta \subseteq \alpha^\bullet$ and $H \setminus H(\alpha) \models \beta \rightarrow \gamma$, it must be that $H \setminus H(\alpha) \models \alpha \rightarrow \gamma$ so that $\gamma \subseteq \alpha^\bullet$ as required. \square

Lemma 9. *The algorithm computing $GD(H)$ outputs the GD basis of H for any definite Horn formula H .*

Proof. Let H be the input to the algorithm, and let H' be its output. We show that H' must be saturated. Let $\alpha \rightarrow \beta$ be an arbitrary implication in the output H' . Because of the initial saturation process, we can refer to this implication as $\alpha^\bullet \rightarrow \alpha^*$. Clearly, $(\alpha^\bullet)^* = \alpha^*$, and H' is right-saturated. It is only left to show that H' is left-saturated. By Lemma 8, it must be that $\alpha^\bullet \subseteq \alpha^*$, but the removal of implications of type (1) guarantees that $\alpha^\bullet \subset \alpha^*$, thus we have that $\text{BITS}(\alpha^\bullet) \not\models \alpha^\bullet \rightarrow \alpha^*$ and Condition 1 of left-saturation is satisfied. Now let $\beta^\bullet \rightarrow \beta^*$ be any other implication in H' . We need to show that $\text{BITS}(\alpha^\bullet) \models \beta^\bullet \rightarrow \beta^*$. Assume by way of contradiction that this is not so, and $\text{BITS}(\alpha^\bullet) \models \beta^\bullet$ but $\text{BITS}(\alpha^\bullet) \not\models \beta^*$. That is, $\beta^\bullet \subseteq \alpha^\bullet$ but $\beta^* \not\subseteq \alpha^*$. If $\beta^\bullet = \alpha^\bullet$, then $\beta^* = \alpha^*$, contradicting the fact that both implications have survived type (2) of removal of implications in the algorithm. Thus, $\beta^\bullet \subset \alpha^\bullet$, and therefore $\beta^* \subseteq \alpha^*$ must hold as well. It cannot be that $\beta^* = \alpha^*$ because we would have that $\alpha^\bullet \rightarrow \alpha^*$ is subsumed by $\beta^\bullet \rightarrow \beta^*$ and thus removed from the output H' during removal of implications of type (3) (and it is not). Thus, it can only be that $\beta^\bullet \subset \alpha^\bullet$ and $\beta^* \subset \alpha^*$. But if $\beta^* \subset \alpha^*$, Lemma 8 and the fact that $H \models \beta^\bullet \rightarrow \beta^*$ (notice that

⁴ No generality is lost since right-saturating clauses (that is, replacing right-hand-sides of clauses with their closure) does not change the function represented.

saturating clauses does not change the logical value of the resulting formula) guarantee that $\beta^* \subseteq \alpha^\bullet$ contradicting our assumption that $\beta^* \not\subseteq \alpha^\bullet$. It follows that H' is saturated as required. \square

It is clear that $GD(H)$ has at most as many implications as H . Thus, if H is of minimum size, then so is $GD(H)$. This, together with the fact that the GD basis is unique, implies:

Theorem 5. [8] *The GD basis of a definite Horn function is of minimum implicational size.* \square

4 The Guigues-Duquenne Basis in Query Learning

The classic query learning algorithm by Angluin, Frazier, and Pitt [2] is able to learn Horn CNF with membership and equivalence queries. It was proved in [2] that the outcome of the algorithm is always equivalent to the target concept. However, the following questions remain open: (1) which of the Horn CNF, among the many equivalent candidates, is output? And (2) does this output depend on the specific counterexamples given to the equivalence queries? Indeed, each query depends on the counterexamples received so far, and intuitively the final outcome should depend on that as well.

Our main result from this section is that, contrary to our first intuition, the output is always the same Horn CNF: namely, the GD basis of the target Horn function. This section assumes that the target is definite Horn, further sections in the paper lift the “definite” constraint.

4.1 The AFP Algorithm for Definite Horn CNF

We recall some aspects of the learning algorithm as described in [4], which bears only slight, inessential differences with the original in [2]. The algorithm maintains a set P of all the positive examples seen so far. The fact that the target is definite Horn allows us to initialize P with the positive example 1^n . The algorithm maintains also a sequence $N = (x_1, \dots, x_t)$ of representative negative examples (these become the antecedents of the clauses in the hypotheses). The argument of an equivalence query is prepared from the list $N = (x_1, \dots, x_t)$ of negative examples combined with the set P of positive examples. The query corresponds to the following intuitive bias: everything is assumed positive unless some (negative) $x_i \in N$ suggests otherwise, and everything that some x_i suggests negative is assumed negative unless some positive example $y \in P$ suggests otherwise. This is exactly the intuition in the hypothesis constructed by the AFP algorithm.

For the set of positive examples P , denote $P_x = \{y \in P \mid x \leq y\}$. The hypothesis to be queried, given the set P and the list $N = (x_1, \dots, x_t)$, is denoted $H(N, P)$ and is defined as $H(N, P) = \{\text{ONES}(x_i) \rightarrow \text{ONES}(\bigwedge P_{x_i}) \mid x_i \in N\}$.

A positive counterexample is treated just by adding it to P . A negative counterexample y is used to either refine some x_i into a smaller negative example, or to add x_{t+1} to the list. Specifically, let

$$i := \min(\{j \mid MQ(x_j \wedge y) \text{ is negative, and } x_j \wedge y < x_j\} \cup \{t+1\})$$

and then refine x_i into $x'_i = x_i \wedge y$, in case $i \leq t$, or else make $x_{t+1} = y$, subsequently increasing t . The value of i is found through membership queries on all the $x_j \wedge y$ for which $x_j \wedge y < x_j$ holds.

```

AFP()
1  N ← ()           ▷ /* empty list */
2  P ← {1^n}       ▷ /* top element */
3  t ← 0
4  while EQ(H(N,P)) = ("no", y)           ▷ /* y is the counterexample */
5      do if y ≠ H(N,P)
6          then add y to P
7          else find the first i such that ▷ /* N = (x_1, ..., x_t) */
8                  x_i ∧ y < x_i, and     ▷ /* that is, x_i ≰ y */
9                  x_i ∧ y is negative    ▷ /* use membership query */
10         if found
11             then x_i ← x_i ∧ y         ▷ /* replace x_i by x_i ∧ y in N */
12             else t ← t + 1; x_t ← y   ▷ /* append y to end of N */
13 return H(N,P)

```

Fig. 2. The AFP learning algorithm for definite Horn CNF

The AFP algorithm is described in Figure 2. In order to prove that its output is indeed the GD basis, we need the following lemmas from [4]:

Lemma 10 (Lemma 2 from [4]). *Along the running of the AFP algorithm, at the point of issuing the equivalence query, for every x_i and x_j in N with $i < j$ there exists a positive example z such that $x_i \wedge x_j \leq z \leq x_j$. \square*

Lemma 11 (Variant of Lemma 1 from [4]). *Along the running of the AFP algorithm, at the point of issuing the equivalence query, for every x_i and x_j in N with $i < j$ and $x_i \leq x_j$, it holds that $\bigwedge P_{x_i} \leq x_j$.*

Proof. At the time x_j is created, we know it is a negative counterexample to the current query, for which it must be therefore positive. That query includes the implication $\text{ONES}(x_i) \rightarrow \text{ONES}(\bigwedge P_{x_i})$, and x_j must satisfy it, and then $x_i \leq x_j$ implies $\bigwedge P_{x_i} \leq x_j$. From that point on, further positive examples may enlarge P_{x_i} and thus reduce $\bigwedge P_{x_i}$, keeping the inequality. Further negative examples y may reduce x_i , again possibly enlarging P_{x_i} and keeping the inequality; or may reduce x_j into $x_j \wedge y$. If $x_i \not\leq x_j \wedge y$ anymore, then there is nothing left to prove. Finally, if $x_i \leq x_j \wedge y$, then $x_i \leq y$, and y is again a negative counterexample

that must satisfy the implication $\text{ONES}(x_i) \rightarrow \text{ONES}(\bigwedge P_{x_i})$ as before, so that $\bigwedge P_{x_i} \leq x_j \wedge y$ also for the new value of x_j . \square

Our key lemma for our next main result is:

Lemma 12. *All hypotheses $H(N, P)$ output by the AFP learning algorithm in equivalence queries are saturated.*

Proof. Recall that $H(N, P) = \{\text{ONES}(x_i) \rightarrow \text{ONES}(\bigwedge P_{x_i}) \mid x_i \in N\}$, where $P_{x_i} = \{y \in P \mid x_i \leq y\}$. Let $\alpha_i = \text{ONES}(x_i)$ and $\beta_i = \text{ONES}(\bigwedge P_{x_i})$ for all i so that $H(N, P) = \{\alpha_i \rightarrow \beta_i \mid 1 \leq i \leq t\}$.

First we show that $H(N, P)$ is left-saturated. To see that $x_i \not\models \alpha_i \rightarrow \beta_i$ it suffices to note that $x_i < \bigwedge P_{x_i}$ since x_i is negative but $\bigwedge P_{x_i}$ is positive by Lemma 1, being an intersection of positive examples; thus, these two assignment must be different.

Now we show that $x_i \models \alpha_j \rightarrow \beta_j$, for all $i \neq j$. If $x_i \not\models \alpha_j$, then clearly $x_i \models \alpha_j \rightarrow \beta_j$. Otherwise, $x_i \models \alpha_j$ and therefore $x_j \leq x_i$. If $i < j$, then by Lemma 10 we have that $x_i \wedge x_j \leq z \leq x_j$. Then, $x_i \wedge x_j = x_j \leq z \leq x_j$, so that $x_j = z$, contradicting the fact that x_j is negative whereas z is positive. Otherwise, $j < i$. We apply Lemma 11: it must hold that $\bigwedge P_{x_j} \leq x_i$. Thus, in this case, $x_i \models \alpha_j \rightarrow \beta_j$ as well because $x_i \models \beta_j = \text{ONES}(\bigwedge P_{x_j})$.

It is only left to show that $H(N, P)$ is right-saturated. Clearly, $H(N, P)$ is consistent with N and P , that is, $x \not\models H(N, P)$ for all $x \in N$ and $y \models H(N, P)$ for all $y \in P$. Take any $x \in N$ contributing the implication $\text{ONES}(x) \rightarrow \text{ONES}(\bigwedge P_x)$ to $H(N, P)$. We show that it is right-saturated, i.e., $\bigwedge P_x = x^*$, where the closure is taken with respect to $H(N, P)$. We note first that $H(N, P) \models \text{ONES}(x) \rightarrow (\text{ONES}(x))^*$ since the closure is taken w.r.t. implications in $H(N, P)$. By the construction of $H(N, P)$, all examples $y \in P_x$ must satisfy it, hence they must satisfy the implication $\text{ONES}(x) \rightarrow (\text{ONES}(x))^*$ as well. Therefore, since $y \models \text{ONES}(x)$ we must have that $y \models (\text{ONES}(x))^*$, or equivalently, that $x^* \leq y$. This is true for every such y in P_x and thus $x^* \leq \bigwedge P_x$. On the other hand, it is obvious that $\bigwedge P_x \leq x^*$ since the implication $\text{ONES}(x) \rightarrow \text{ONES}(\bigwedge P_x)$ of $H(N, P)$ guarantees that all the variables in $\bigwedge P_x$ are included in the forward chaining process in the final x^* . So we have $x^* \leq \bigwedge P_x \leq x^*$ as required. \square

Putting Theorem 4 and Lemma 12 together, we obtain:

Theorem 6. *AFP, run on a definite Horn target, always outputs the GD basis of the target concept.* \square

5 A Canonical Basis for General Horn

Naturally, we wish to extend the notion of saturation and GD basis to general Horn functions. We do this via a prediction-with-membership reduction [3] from general Horn to definite Horn, and use the corresponding intuitions to define a GD basis for general Horn. We use this reduction to generalize our AFP

algorithm to general Horn CNF, and as a consequence one obtains that the generalized AFP always outputs a saturated version of the target function. Indeed, for the generalized AFP it is also the case that the output is only dependent on the target, and not on the counterexamples received along the run. Finally, we construct strong polynomial certificates for general Horn functions directly in terms of the generalized GD basis, thus generalizing our earlier result of [4].

5.1 Reducing General Horn CNF to Definite Horn CNF

In this section we describe the intuition of the representation mapping, which we use in the next section to obtain a canonical basis for general Horn functions.

For any Horn H over n propositional variables, e.g. $X = \{x_i \mid 1 \leq i \leq n\}$, we construct a definite Horn H' over the set of $n + 1$ propositional variables $X' = X \cup \{\mathbf{f}\}$, where \mathbf{f} is a new “dummy” variable; in essence \mathbf{f} represents the *false* (that is, empty) consequent of the negative clauses in H . The relationship between the assignments for H and H' are as follows: for assignments of $n + 1$ variables $x\mathbf{b}$ where x assigns to the variables in X and \mathbf{b} is the truth value assigned to \mathbf{f} , $x0 \models H'$ if and only if $x \models H$, whereas $x1 \models H'$ if and only if $x = 1^n$.

Define the implication $C_{\mathbf{f}}$ as $\mathbf{f} \rightarrow X'$. Let H_d be the set of definite Horn clauses in H , and $H_n = H \setminus \{H_d\}$ the negative ones. Define the mapping g as

$$g(H) = H_d \cup \{\neg C \rightarrow X' \mid C \in H_n\} \cup \{C_{\mathbf{f}}\}.$$

That is, $g(H)$ includes the definite clauses of H , the special implication $C_{\mathbf{f}}$, and the clauses C that are negative are made definite by forcing all the positive literals, including \mathbf{f} , into them. Clearly, the resulting $g(H)$ is definite Horn. Observe that that the new implication $C_{\mathbf{f}}$ is saturated and the ones coming from H_n are right-saturated. Observe also that g is injective: given $g(H)$, we recover H by removing the implication $C_{\mathbf{f}}$, and by removing all positive literals from any implications containing \mathbf{f} . Clearly, $g^{-1}(g(H)) = H$, since g^{-1} is removing all that g adds.

5.2 Constructing a GD-like Basis for General Horn CNF

The notion of left-saturation translates directly into general Horn CNF:

Definition 4. *Let $B = \{\alpha_i \rightarrow \beta_i\}_i$ be a basis for some Horn function. Notice that now β_i can possibly be empty (it is empty for the negative clauses). Then, B is left-saturated if the following two conditions hold:*

1. $\text{BITS}(\alpha_i) \not\models \alpha_i \rightarrow \beta_i$, for all i
2. $\text{BITS}(\alpha_i) \models \alpha_j \rightarrow \beta_j$, for all $i \neq j$

Notice that now in the second condition, if $\beta_j = \emptyset$, i.e., we are dealing with a negative clause, then $\text{BITS}(\alpha_i) \models \alpha_j \rightarrow \beta_j$ translates directly into $\text{BITS}(\alpha_i) \not\models \alpha_j$ (equivalently, $\alpha_i \not\supseteq \alpha_j$) since it could never happen that $\text{BITS}(\alpha_i) \models \alpha_j$ but $\text{BITS}(\alpha_i) \models \square$, where \square is the (unsatisfiable) empty clause. The conditions can be more explicitly stated as follows:

1. $\text{BITS}(\alpha_i) \not\models \alpha_i \rightarrow \beta_i$, for all i
2. $\text{BITS}(\alpha_i) \models \alpha_j \rightarrow \beta_j$, for all $i \neq j$ such that $\beta_j \neq \emptyset$
3. $\alpha_j \not\subseteq \alpha_i$, for all $i \neq j$ such that $\beta_j = \emptyset$

Notice that the third condition guarantees that no negative clause is a subset of any other clause. If this were not so, we clearly would have a redundant clause.

For a definite Horn CNF H , right-saturating a clause $\alpha \rightarrow \beta$ essentially means that we add to its consequent everything that is implied by its antecedent, namely α^* . This can no longer be done in the case of general Horn CNF, since we need to take special care of the negative clauses. If $\beta = \emptyset$, we cannot set β to α^* without changing the underlying boolean function being represented. The closure x^* of an assignment x is defined as the closure with respect to all definite clauses in the general Horn CNF. It is useful to continue to partition assignments x in the boolean hypercube according to their closures x^* ; however, in the general Horn case, we distinguish a new class (the negative class) of closed assignments that are actually negative, that is, it is possible now that $x^* \not\models H$. These assignments are exactly those that satisfy all definite clauses of H but violate negative ones. Based on this, the negative clauses (those with antecedent α such that $\text{BITS}(\alpha^*) \not\models B$) should be left unmodified, and the definite clauses (those whose antecedents α are such that $\text{BITS}(\alpha^*) \models B$) should be right-saturated. Thus, the definition is:

Definition 5. Let $B = \{\alpha_i \rightarrow \beta_i\}_i$ be a basis for some general Horn function. Then, B is right-saturated if, for all i , $\beta_i = \emptyset$ if $\alpha_i^* \not\models B$, and $\beta_i = \alpha_i^*$ otherwise.

As for the definite case, “saturated” means that the Horn CNF in question is both left- and right-saturated. We must see that this is the “correct” definition in some sense:

Lemma 13. A basis H is saturated iff $H = g^{-1}(GD(g(H)))$.

Proof. First let us note that the expression $g^{-1}(GD(g(H)))$ is well-defined. We can always invert g on $GD(g(H))$, since saturating $g(H)$ does not modify $C_{\mathbf{f}}$ (already saturated) and it does not touch the positive literals of implications containing \mathbf{f} since these are right-saturated. Therefore, we can invert it since the parts added by g are left untouched by the construction of $GD(g(H))$.

We prove first that if H is saturated then $H = g^{-1}(GD(g(H)))$. Assume, then, that H is saturated but $H \neq g^{-1}(GD(g(H)))$. Applying g , which is injective, this can only happen if $GD(g(H)) \neq g(H)$, namely, $g(H)$, as a definite Horn CNF, differs from its own GD basis and, hence, it is not saturated: it must be because some implications other than $C_{\mathbf{f}}$ is not saturated, since this last one is saturated by construction. Also the ones containing \mathbf{f} in their consequents are right-saturated, so no change happens in the right-hand-sides of these implications when saturating $g(H)$. This means that when saturating we must add a literal different from \mathbf{f} to the right-hand-side of an implication not containing \mathbf{f} or to the left-hand-side of an implication. In both cases, this means that the original H could not be saturated either, contradicting our assumption.

It is only left to show that an H such that $H = g^{-1}(GD(g(H)))$ is indeed saturated. By way of contradiction, assume that H is not saturated but $H = g^{-1}(GD(g(H)))$. Applying g to both sides, we must have that $g(H) = GD(g(H))$ so that $g(H)$ is actually saturated. Notice that the only difference between H and $g(H)$ is in the implication C_f and the right-hand-sides of negative clauses in H ; $g(H)$ being left-saturated means that so must be H since the left-hand-sides of H and $g(H)$ coincide exactly (ignoring C_f naturally). Therefore, H is left-saturated as well. It must be that H is not right-saturated, that is, it is either missing some variable in some non-empty consequent, or some clause that should be negative is not. In the first case, then $g(H)$ is missing it, too, and it cannot be saturated. In the second case, then there is a redundant clause in H contradicting the fact that H is left-saturated (see Lemma 2). In both cases we arrive at a contradiction, thus the lemma follows. \square

Notice that this last lemma also gives us a way to compute the saturation (that is, the GD basis) of a given general Horn CNF:

Theorem 7. *Horn functions have a unique saturated basis. This basis, which we denote $GD(H)$, can be computed by $GD(H) = g^{-1}(GD(g(H)))$.*

Proof. If H is saturated then $H = g^{-1}(GD(g(H)))$. The uniqueness of such an H follows from the following facts: first, $g(H)$ and $g(H')$ are equivalent whenever H and H' are equivalent; second, $GD(g(H))$ is unique for the function represented by H (Theorem 4) and third, g^{-1} is univocally defined since g is injective. \square

Example 5. Let H be the Horn CNF $\{a \rightarrow b, a \rightarrow c, abc \rightarrow \emptyset\}$. Then,

- $g(H) = \{a \rightarrow b, a \rightarrow c, abc \rightarrow abcf, \mathbf{f} \rightarrow abcf\}$
- $GD(g(H)) = \{a \rightarrow abcf, \mathbf{f} \rightarrow abcf\}$
- $GD(H) = g^{-1}(GD(g(H))) = \{a \rightarrow \emptyset\}$

Similarly to the case of definite Horn functions, $GD(H)$ does not increase the number of new implications, and therefore if H is of minimum size, $GD(H)$ must be of minimum size as well. This, together with the uniqueness of saturated representation implies that:

Theorem 8. *[8] The GD basis of a general Horn function is of minimum implicational size.* \square

5.3 The AFP Algorithm for General Horn CNF

We study now the AFP algorithm operating on general Horn CNF, by following a detour: we obtain it via reduction to the definite case.

We consider, therefore, an algorithm that, for target a general Horn function H , simulates the version of AFP algorithm from Figure 2 on its definite transformation $g(H)$, where g is the representation transformation from Section 5.1. It has to simulate the membership and equivalence oracles for definite Horn

CNF that the underlying algorithm expects, by using the oracles that it has for general Horn.

Initially, we set $P = \{1^{n+1}\}$, and $N = (0^n 1)$ since we know that $g(H)$ is definite and must contain the implication $\mathbf{f} \rightarrow X \cup \{\mathbf{f}\}$ by construction. In essence, the positive assignment $1^{n+1} = \mathbf{f}^\star$ and the negative $0^n 1 = \mathbf{f}^\bullet$ guarantee that the implication $C_{\mathbf{f}}$ is included in every hypothesis $H(N, P)$ that the simulation outputs as an equivalence query. The resulting algorithm for general Horn CNF is described in Figure 3.

In order to deal with the queries, we use two transformations: we must map examples over the $n + 1$ variables, asked as membership queries, into examples over the original example space over n variables, although in some cases are able to answer the query directly as we shall see. Upon asking $x0$ as membership query for $g(H)$, we pass on to H the membership query about x . Membership queries of the form $x1$ are answered always negatively, except for 1^{n+1} which is answered positively (in fact query 1^{n+1} never arises anyway, because that example is in P from the beginning). Conversely, n -bit counterexamples x from the equivalence query with H are transformed into $x0$. The equivalence queries themselves are transformed according to g^{-1} . It is readily checked that all equivalence queries belong indeed to the image set of g since $C_{\mathbf{f}} \in H(N, P)$.

All together, these functions constitute a prediction-with-membership (pwm) reduction from general Horn to definite Horn, in the sense of [3].

It is interesting to note that if we unfold the simulation, we end up with the original algorithm by Angluin, Frazier and Pitt [2] (obviously, with no explicit reference to our “dummy” \mathbf{f}). In fact, the only difference between our algorithm in Figure 3 is in the construction of hypotheses: ours uses the reduction described in Section 5.1 and the construct $H(N, P)$, whereas the algorithm AFP of [2] directly build their hypotheses from the negative x_i and positive examples received. These negative examples, in fact coincide exactly for both algorithms, as the mechanism that maintains N is the same. AFP [2] builds the hypotheses in the following way. Let X be the set of propositional variables. Each $x_i \in N$ generates an implication $\text{ONES}(x_i) \rightarrow \mathbf{F}$ if no positive counterexample y received is such that $y \geq x_i$, where [2] use the variable \mathbf{F} to denote the always *false* variable (this is analogue to our \mathbf{f}). Otherwise, x_i generates the implication $\text{ONES}(x_i) \rightarrow \beta$, where β is the set of variables in X that are not in $\text{ONES}(x_i)$ and are set to 1 in *all* positive counterexamples y such that $y \geq x_i$.

To see that $H(N, P)$ contains the same implications, notice that if there is no $y \geq x_i$, then the implication included in $H(N, P)$ in our version is $\text{ONES}(x_i) \rightarrow X \cup \{\mathbf{f}\}$ (because 1^{n+1} is used to construct the right-hand-side). When constructing the inverse image g^{-1} all the literals in the consequent are removed and thus the corresponding clause is $\text{ONES}(x_i) \rightarrow \square$, precisely the one constructed by [2]. On the other hand, if there are positive y s.t. $y \geq x_i$, then our algorithm constructs the implication $\text{ONES}(x_i) \rightarrow \text{ONES}(\bigwedge P_{x_i})$. Notice that since $P_{x_i} = \{y \mid y \geq x_i, y \in P\}$, variables in $\text{ONES}(\bigwedge P_{x_i})$ are precisely those that are set to 1 in all $y \in P_{x_i}$. Thus, the only difference between our clauses

and the ones in [2] is the fact that ours are right-saturated (the ones in [2] are only missing the variables in $\text{ONES}(x_i)$ from their right-hand-sides).

```

AFP()
1   $N \leftarrow (0^n 1)$ 
2   $P \leftarrow \{1^{n+1}\}$ 
3  ▷ /* simulate AFP */
4  if AFP asks an equivalence query with  $H(N, P)$ 
5      then pass  $g^{-1}(H(N, P))$  to our equivalence oracle
6      if answer is "Yes"
7          then return  $g^{-1}(H(N, P))$  and stop
8      elseif answer is "No" and  $x$  is counterexample received
9          then pass  $f(x) = x0$  as a counterexample to AFP
10 elseif AFP asks a membership query  $MQ(x')$ 
11     then if  $h(x') = 0$            ▷ /* membership query transformation */
12         then answer the query with a "No"
13     elseif  $h(x') = 1$ 
14         then answer the query with a "Yes"
15     elseif  $h(x') = x \notin \{0, 1\}$ 
16         then answer query using our own membership oracle on  $x$ 

```

Fig. 3. The AFP learning algorithm for general Horn CNF

Therefore, the outcome of AFP on a general Horn target H comes univocally determined by the outcome of AFP on the corresponding definite Horn function $g(H)$; combining this fact with Theorems 6 and 7 leads to:

Theorem 9. *The AFP algorithm always outputs the GD basis of the target concept.* \square

5.4 Certificates for General Horn CNF

The certificate dimension of a given concept class is closely related to its learnability in the model of learning from membership and equivalence queries [10, 9, 1, 6]. Informally, a certificate for a class \mathcal{C} of concepts of size at most m is a set of (labeled) assignments that proves that concepts consistent with it must be outside \mathcal{C} . The polynomial $q(m, n)$ used below quantifies the cardinality of the certificate set in term of m , the size of the class, and n , the number of variables in the class. The polynomial $p(m, n)$ quantifies the expansion in size allowed in the hypotheses. In this paper, $p(m, n) = m$ and thus we construct *strong* certificates. Formally,

Definition 6. *Let \mathcal{C} be a class of representations of a boolean concepts in $\{0, 1\}^n$, and let $\mathcal{C}^{\leq m}$ be the subclass of concepts in \mathcal{C} of size at most m . The class \mathcal{C} has certificates of size $q(m, n)$ for representation expansion $p(m, n)$ if for every $m, n > 0$ and for every boolean function $f \notin \mathcal{C}^{\leq p(m, n)}$ (i.e., $|f| > p(m, n)$ or $f \notin \mathcal{C}$), there is a set $Q_f \subseteq \{0, 1\}^n$ satisfying the following:*

1. $|Q_f| \leq q(m, n)$, and
2. for every $g \in \mathcal{C}^{\leq m}$, there is some $x \in Q_F$ s.t. $g(x) \neq f(x)$. In other words, no function in $\mathcal{C}^{\leq m}$ is consistent with f over Q_f .

If $p(m, n)$ and $q(m, n)$ are polynomials then we say that \mathcal{C} has polynomial size certificates.

In [4] we show how to build strong certificates for definite Horn CNF. Here, we extend this to general Horn CNF, and describe the certificates directly in terms of the generalized GD basis. First we present a lemma that is useful in the final proof.

Lemma 14. *Let f be a Horn function and x_i^\bullet, x_j^\bullet two left-hand-sides of saturated clauses $\alpha_i \rightarrow \beta_i, \alpha_j \rightarrow \beta_j$ of f (i.e., $x_i^\bullet = \text{BITS}(\alpha_i)$ and $x_j^\bullet = \text{BITS}(\alpha_j)$). Then, $x_i^\bullet \wedge x_j^\bullet \not\models f$ if and only if $x_i^\bullet \leq x_j^\bullet$ or $x_j^\bullet \leq x_i^\bullet$.*

Proof. The direction from right to left is clear: if $x_i^\bullet \leq x_j^\bullet$ (or $x_j^\bullet \leq x_i^\bullet$), then $x_i^\bullet \wedge x_j^\bullet = x_i^\bullet$ (or $x_i^\bullet \wedge x_j^\bullet = x_j^\bullet$). In either case, it is clear that $x_i^\bullet \not\models f$ and $x_j^\bullet \not\models f$ so that $x_i^\bullet \wedge x_j^\bullet \not\models f$.

For the other direction, assume that $x_i^\bullet \wedge x_j^\bullet \not\models f$. Other implications $\alpha \rightarrow \beta$ of f that are neither $\alpha_i \rightarrow \beta_i$ nor $\alpha_j \rightarrow \beta_j$ are satisfied by $x_i^\bullet \wedge x_j^\bullet$, since Horn implications are closed under intersection and both $x_i^\bullet \models \alpha \rightarrow \beta$ and $x_j^\bullet \models \alpha \rightarrow \beta$ by left-saturation. Therefore, it must be that either $x_i^\bullet \wedge x_j^\bullet \not\models \alpha_i \rightarrow \beta_i$ or $x_i^\bullet \wedge x_j^\bullet \not\models \alpha_j \rightarrow \beta_j$. Assume w.l.o.g. that $x_i^\bullet \wedge x_j^\bullet \not\models \alpha_i \rightarrow \beta_i$. This implies that $\alpha_i \leq x_i^\bullet \wedge x_j^\bullet$ so that $\alpha_i \leq x_j^\bullet$. Since $x_j^\bullet \models \alpha_i \rightarrow \beta_i$, it must hold that $x_j^\bullet \models \beta_i$. Thus, $\beta_i \neq \emptyset$ and $\beta_i = \alpha_i^*$ by right-saturation of the implication. We obtain that $x_i^\bullet \leq x_i^* \leq x_j^\bullet$ as required. \square

Finally, our certificate result is:

Theorem 10. *The class of general Horn CNF has strong polynomial certificates with $p(m, n) = m$ and $q(m, n) = \binom{m+1}{2} + m + 1 = \binom{m+2}{2}$.*

Proof. The argumentation follows, essentially, the same steps as the analogous proof in [4], because, by Lemma 13, the GD basis in the general case is saturated, and therefore all required facts carry over to the general case. Let f be a boolean function that cannot be represented with m Horn implications. If f is not Horn, then three assignments $x, y, x \wedge y$ such that $x \models f, y \models f$ but $x \wedge y \not\models f$ suffice. Otherwise, f is a general Horn CNF of implicational size strictly greater than m . Assume that f contains at least $m + 1$ non-redundant and possibly negative implications $\{\alpha_i \rightarrow \beta_i\}$. We define the certificate for f as follows:

$$\begin{aligned} Q_f = & \{x_i^\bullet, x_i^* \mid 1 \leq i \leq m + 1, x_i = \text{BITS}(\alpha_i), \beta_i \neq \emptyset\} \\ & \cup \{x_i^\bullet \mid 1 \leq i \leq m + 1, x_i = \text{BITS}(\alpha_i), \beta_i = \emptyset\} \\ & \cup \{x_i^\bullet \wedge x_j^\bullet \mid 1 \leq i < j \leq m + 1\} \end{aligned}$$

It is illustrative to note the relation between this set of certificates for f and its GD basis: the assignments x_i^\bullet and x_i^* correspond exactly to the left and

right-hand-sides of the (saturated) definite implications in $GD(f)$. For negative clauses, only the (saturated) left-hand-side of the implication x_i^\bullet matters.

We prove that Q_f is indeed a certificate set. Take any Horn function g of implicational size at most m . By way of contradiction, assume that g is consistent with f over all the elements in Q_f .

By construction, the assignments x_i^\bullet are negative for f . Since there are at least $m + 1$ such assignments in Q_f , there must be one particular implication $\alpha \rightarrow \beta$ in g violated by both x_i^\bullet and x_j^\bullet for some i, j s.t. $1 \leq i < j \leq m + 1$.

Clearly $x_i^\bullet \wedge x_j^\bullet \not\models \alpha \rightarrow \beta$; therefore $x_i^\bullet \wedge x_j^\bullet \not\models g$.

Assume w.l.o.g. that the clauses in f are saturated; it must be that $x_i^\bullet \models \alpha_j \rightarrow \beta_j$, and $x_j^\bullet \models \alpha_i \rightarrow \beta_i$.

If $x_i^\bullet \leq x_j^\bullet$, then $x_j^\bullet \models \alpha_i$ and therefore $x_j^\bullet \models \beta_i$, $\beta_i \neq \emptyset$, $\beta_i = \text{BITS}(x_i^\bullet)$ so that $x_i^\bullet \leq x_j^\bullet$. The fact that $\beta_i \neq \emptyset$ implies that $x_i^\bullet \models f$ and $x_i^\bullet \in Q_f$. If x_i^\bullet is positive for f then it must be positive for g as well, thus $x_i^\bullet \models \alpha \rightarrow \beta$ and hence $x_i^\bullet \models \beta$ since $x_i^\bullet \models \alpha$ and $x_i^\bullet \leq x_i^\bullet$. But this contradicts the fact that $x_j^\bullet \not\models \alpha \rightarrow \beta$ because we have obtained that $x_i^\bullet \leq x_j^\bullet$ and $x_i^\bullet \models \beta$. The case where $x_j^\bullet \leq x_i^\bullet$ is analogous. Otherwise, x_i^\bullet and x_j^\bullet are incomparable. In this case, Lemma 14 guarantees that $x_i^\bullet \wedge x_j^\bullet \models f$, thus g and f differ on $x_i^\bullet \wedge x_j^\bullet \in Q_f$.

We conclude that Q_f is a certificate set. Its cardinality appears to be $\binom{m+1}{2} + 2(m+1)$ in the worst case that f is definite Horn (no negative clauses). Notice however that the proof above only uses x_i^\bullet when there is a x_j^\bullet s.t. $x_i^\bullet \leq x_j^\bullet$. Moreover, in this case, it is clear that $x_i^\bullet \wedge x_j^\bullet = x_i^\bullet$ so the intersection is already in Q_f and we do not need to count it. So, if we remove from Q_f as described above those x_i^\bullet for which no x_j^\bullet exists s.t. $x_i^\bullet \leq x_j^\bullet$, we end up with a certificate set of the desired cardinality. The final certificate set is:

$$\begin{aligned} Q_f = & \{x_i^\bullet \mid 1 \leq i \leq m + 1, x_i = \text{BITS}(\alpha_i)\} \\ & \cup \{x_i^\bullet \mid 1 \leq i, j \leq m + 1, x_i = \text{BITS}(\alpha_i), x_i^\bullet \leq x_j^\bullet \text{ for some } j \neq i\} \\ & \cup \{x_i^\bullet \wedge x_j^\bullet \mid 1 \leq i < j \leq m + 1, x_i^\bullet \not\leq x_j^\bullet, x_j^\bullet \not\leq x_i^\bullet\} \end{aligned}$$

□

The certificates of [5] for Horn CNF use $p(m, n) = m(n+1)$. In contrast, here we only require that $p(m, n) = m$ thus applying to the more stringent model of strong proper learning [10]. On the other hand, in [5] m counts the number of clauses whereas here m is the number of comparable pairs, which translates to using implications in the Horn logic setting. The authors feel that the right thing to count in the case of Horn expressions is the number of implications, especially in the context of analyzing the AFP algorithm that learns using the implicational representation.

Notice also that this upper bound matches *exactly* the lower bound from [5] in the case that $m < n$. If $m > n$, there is still a gap between the lower bound of $\Omega(mn)$ in [5] and our upper bound of $O(m^2)$. Regarding the consequences of this certificate size on the query complexity of the problem, see the discussions in [1, 5].

References

1. D. Angluin. Queries revisited. *Theoretical Computer Science*, 313:175–194, 2004.
2. D. Angluin, M. Frazier, and L. Pitt. Learning conjunctions of Horn clauses. *Machine Learning*, 9:147–164, 1992.
3. Dana Angluin and Michael Kharitonov. When won't membership queries help? *Journal of Computer and System Sciences*, 50(2):336–355, April 1995.
4. M. Arias and José L. Balcázar. Query learning and certificates in lattices. In *ALT*, pages 303–315, 2008.
5. Marta Arias, Aaron Feigelson, Roni Khardon, and Rocco A. Servedio. Polynomial certificates for propositional classes. *Inf. Comput.*, 204(5):816–834, 2006.
6. José L. Balcázar, Jorge Castro, David Guijarro, Johannes Köbler, and Wolfgang Lindner. A general dimension for query learning. *J. Comput. Syst. Sci.*, 73(6):924–940, 2007.
7. Chin-Liang Chang and Richard Char-Tung Lee. *Symbolic Logic and Mechanical Theorem Proving*. Academic Press, Inc., Orlando, FL, USA, 1973.
8. J.L. Guigues and V. Duquenne. Familles minimales d'implications informatives resultants d'un tableau de donnees binaires. *Math. Sci. Hum.*, 95:5–18, 1986.
9. T. Hegedüs. On generalized teaching dimensions and the query complexity of learning. In *Proceedings of the Conference on Computational Learning Theory*, pages 108–117, New York, NY, USA, 1995. ACM Press.
10. Lisa Hellerstein, Krishnan Pillaipakkamnatt, Vijay Raghavan, and Dawn Wilkins. How many queries are needed to learn? *Journal of the ACM*, 43(5):840–862, September 1996.
11. A. Horn. On sentences which are true of direct unions of algebras. *J. of Symbolic Logic*, 16:14–21, 1956.
12. Roni Khardon and Dan Roth. Reasoning with models. *Artificial Intelligence*, 87(1-2):187 – 213, 1996.
13. J. C. C. McKinsey. The decision problem for some classes of sentences without quantifiers. *J. Symbolic Logic*, 8:61–76, 1943.
14. H. Wang. Toward mechanical mathematics. *IBM Journal for Research and Development*, 4:2–22, 1960.
15. Marcel Wild. A theory of finite closure spaces based on implications. *Advances in Mathematics*, 108:118–139, 1994.