Function-Free Horn Clauses Are Hard to Approximate

Richard Nock and Pascal Jappy

Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier,
161, rue Ada,
34392 Montpellier, France
{nock,jappy}@lirmm.fr

Abstract. In this paper, we show two hardness results for approximating the best function-free Horn clause by an element of the same class. Our first result shows that for some constant $k > 0$, the error rate of the best k-Horn clause cannot be approximated in polynomial time to within any constant factor by an element of the same class. Our second result is much stronger. Under some frequently encountered complexity hypothesis, we show that if we replace the constant number of Horn clauses by a small, poly-logarithmic number, the constant factor blows up exponentially to a quasi-polynomial factor $n^{\log^k n}$, where n is the number of predicates of the problem, a measure of its complexity. Our main result links the difficulty of error approximation with the number of clauses allowed. We finally give an outline of the incidence of our result on systems that learn using ILP (Inductive Logic Programming) formalism.

1 Introduction and motivation

ILP is an active research branch at the crossroads of of Machine Learning and Logics. It aims at learning concepts expressed as (variously) restricted Horn Clause Programs from examples, and in the presence of background knowledge. Many experimental applications are available, that have been applied to domains such as biology, chess playing and natural langage analysis. Theoretical work has allowed to establish learnability results for some subclasses of first order Horn clauses. Early studies were undertaken in the Identification in the limit model [7], but most work has focused on Approximately Correct (PAC) learnability [15], [10] which is thought to better quantify the complexity of learning in terms of computational effort and number of examples required. In ILP, this latter problem is intractable for very general classes such as unconstrained Horn clauses (see [11] for a detailed presentation of computational hardness results). So, in order to achieve positive results, several restrictions of Horn Clause programs have been considered [13], [4], [5], and [6].

However, conflicts between PAC results and practical ones have led researchers to look for other learnability models [12]. In a previous paper, we highlighted divergences between PAC and robust learning [8] results for some of the main

ILP classes. Whereas PAC learning makes the strong assumption that any target concept can be represented in the hypothesis class $\mathcal{H}$, (which is very rarely acceptable in practice), robust learning studies the degradation in prediction performance of a hypothesis class $\mathcal{H}$ when it is not known a priori whether it contains the target concept's class. This makes this model a stricter one but it is closer to practical requirements. The commonpoint to both PAC and robust learning models is the sufficiency of worst case analyses to obtain negative results. Our result in [9] states that, even when considering a simple subclass of ILP formalism and even when looking for a single Horn clause, no polynomial-time algorithm can produce a formula whose error comes close to the error of the optimal single Horn clause. In this paper, we go further in worst-case analyses. We show that the condition on the error can be replaced by a much weaker one without losing negative results. We show that no polynomial-time algorithm can produce a formula approximating the error of the optimal one to within very large factors. The rest of this paper is organised as follows: in section 2, we present the ILP background we need for our results, and the link between ILP and structural complexity. In section 3 and 4 we prove that approximating function-free Horn clause is hard. Finally, in section 5, we highligh some relevant subclasses of ILP formalism for which our results are valid.

2 An ILP approximation problem

For a complete formalization of the ILP background needed for this article, we refer the reader to [9]. Given a Horn clause langage $\mathcal{L}$ and a correct inference relation on $\mathcal{L}$, an ILP learning problem can be formalized as follows. Assume a background knowledge $\mathcal{BK}$ expressed in a langage $\mathcal{LB} \subseteq \mathcal{L}$, and a set of examples $\mathcal{E}$ in a langage $\mathcal{LE} \subseteq \mathcal{L}$. The goal is to produce a hypothesis h in a hypothesis class $\mathcal{H} \subseteq \mathcal{L}$ consistent with $\mathcal{BK}$ and $\mathcal{E}$ such that h and the background knowledge cover all positive examples and none of the negative ones. The choice of the representation langages for the background knowledge and the examples, and the inference relation greatly influence the complexity (or decidability) of the learning problem. A common restriction for both $\mathcal{BK}$ and $\mathcal{E}$ is to use ground facts. As in [11], we use θ-subsumption as inference relation. Its main drawback being that it does not allow the use of background knowledge, other subsumption relations have been defined to do so, in particular generalized subsumption [2], and are thus preferred in ILP. We now state a useful lemma

Lemma 1 *Learning a Horn clause program from a set of ground background knowledge $\mathcal{BK}$ and ground examples $\mathcal{E}$, the inference relation being generalized subsumption, is equivalent to learning the same program with θ-subsumption, and empty background knowledge and examples defined as ground Horn clauses of the form $e \leftarrow b$, where $e \in \mathcal{E}$ and $b \in \mathcal{BK}$.*

This lemma allows us to incorporate the background knowledge in the new examples (and is thus empty). Examples and clauses are defined by predicates. To the variables of these predicates that are in the clauses built correspond constant symbols in the examples. θ-subsumption relative to the examples aims at finding adequate substitutions of variables with constant symbols. It can be defined in a general way as follows.

Definition 1 **(θ-subsumption)** *A clause C_1 θ-subsumes a clause C_2 if there exists a substitution θ such that $C_1\theta \subseteq C_2$.*

Our ILP problem can be presented as an optimization problem as follows, respecting the formalization of [8] :

Name : *Opt*(Weighted-Approx(g(.)-function-free Horn clauses)).
Instance : A set of examples $\mathcal{E} = S^+ \cup S^-$, an integer weight $w(x_i) > 0$ for each example $x_i \in S^+ \cup S^-$.
Feasible Solutions : $h \in g$(.)-function-free Horn clause.
Cost Function : $\sum_{(x \in S^+ \wedge h(x)=0) \vee (x \in S^- \wedge h(x)=1)} w(x)$.

g(.) is a function defining the maximum size (clause number) of the function-free Horn clauses constructed. It is worthwile remarking that a machine learning algorithm is ran practically on a set of examples often called the "learning sample", and aims at finding a low-error formula, without prior knowledge on the concept from which these examples were taken. Therefore, what does such an algorithm is trying to find a feasible solution to the previous problem having a low cost. Proving lower bounds on the costs of polynomial-time algorithms for this problem is therefore of practical interest.

3 Result on k-function-free Horn clauses

In this section we state and prove a first non-approximability result, dealing with small-sized function-free Horn clauses.

Theorem 1 *If $NP \not\subset P$, $\forall k \geq 3$, Opt(Weighted-Approx(k-function-free Horn clauses)) is not approximable to within any constant $d > 0$.*

We make a reduction from a minimization problem previously studied in [8]:

Name : *Opt*(Aggravated 3-SAT).
Instance : A set of variables $U = \{x_1, \overline{x}_1, ..., x_n, \overline{x}_n\}$, a collection of 3-clauses over U, a subset $U' \subseteq U$, an assignment satisfying all clauses.
Feasible Solutions : An assignation of the variables of U satisfying all clauses.
Cost Function : The number of variables from U' assigned to **True** .

The satisfiability constraint implies that a solution always exists; the difficulty of the problem therefore relies only is the minimization of the cost function, and not in finding feasible solutions, which would be an artifact of the problem's hardness. We use the following result on *Opt*(Aggravated 3-SAT):

Theorem 2 [8] *If $NP \not\subset P$, Opt(Aggravated 3-SAT) is not approximable to within any constant $d > 0$.*

In order to prove our result, we need to obtain an intermediate inapproximability result for the following minimization problem:

Name : *Opt*(Aggravated k-Colorability).
Instance : A graph $G = (X, E)$. A subset $X' \subset X$. A vertex $x \in X \backslash X'$. A trivial k-Colorability of G.
Feasible Solutions : A valid k-Colorability of G.
Cost Function : The number of elements of X' having the same color as x.

Due to space limitations, undetailed proofs can be found in [14].

Theorem 3 *If $NP \not\subset P$, Opt(Aggravated k-Colorability) is not approximable to within any constant $d > 0$.*

Proof sketch. The reduction is made from the problem *Opt*(Aggravated 3-SAT). $C = \{C_1, ..., C_p\}$ denotes the set of 3-clauses instance of *Opt*(Aggravated 3-SAT). We transform it into a new set $C' = \{C'_1, ..., C'_p, C'_{p+1}, C'_{p+2}, C'_{p+3}, C'_{p+4}\}$ in the following way : let $L = U = \{x_1, \overline{x}_1, ..., x_n, \overline{x}_n\}$ stands for the variables set of the 3-SAT instance. Let $L' = \{x_1, \overline{x}_1, ..., x_n, x_{n+1}, \overline{x}_{n+1}, x_{n+2}, \overline{x}_{n+2}, x_{n+3}, \overline{x}_{n+3}\}$ be our new set of variables, and define $\forall 1 \leq i \leq p, C'_i = C_i \vee x_{n+1}$, $C'_{p+1} = \overline{x}_{n+1} \vee x_{n+2} \vee x_{n+3}$, $C'_{p+2} = \overline{x}_{n+2} \vee x_{n+3}$, $C'_{p+3} = x_{n+2} \vee \overline{x}_{n+3}$, and $C'_{p+4} = \overline{x}_{n+2} \vee \overline{x}_{n+3}$. The length of each new clause in C' is either 2, 3 or 4. We now state a number of facts useful to prove theorem 3. We let $[n]$ denote the set $\{1, 2, ..., n\}$.

Fact 1 *C' satisfies two properties : (1) if C' is satisfiable, x_{n+1}, x_{n+2} and x_{n+3} are all* **False** *; (2) C is satisfiable iff C' is satisfiable.*

The graph G we construct, instance of *Opt*(Aggravated k-Colorability), has the modular structure which we now describe.

Step 1: each of the $2n + 6$ variables of L' is represented by a vertex in G:

$$\{x_1, \overline{x}_1, ..., x_n, \overline{x}_n, x_{n+1}, \overline{x}_{n+1}, x_{n+2}, \overline{x}_{n+2}, x_{n+3}, \overline{x}_{n+3}\}$$

We call these "variable vertices". We then place $n + 3$ edges $(x_i, \overline{x}_i), \forall i \in [n + 3]$ in G. Each couple of variable vertices $(x_i, \overline{x}_i), i \in [n]$ represent the two possible truth assignment of the variables $x_i, \overline{x}_i$ of U: either **True** for x_i (and **False** for $\overline{x}_i$), or **False** for x_i (and **True** for $\overline{x}_i$).

Step 2: for each clause $C'_i, i \in [p + 4]$, we create a corresponding subgraph H_i. Each of these subgraphs uses a basic buiding block shown in figure 1 (K_k is the complete graph on k vertices). In this figure, at least either of a or b is a variable vertex, and y is a new vertex. $\forall i \in [p + 4]$, each clause C'_i of size 2, 3 or 4 is represented in G by a subgraph H_i using 1, 2 or 3 basic building blocks respectively. As shown in figure 1, to each clause C'_i corresponds a distinct vertex noted $y_{6,i}$. The only vertices shared by these subgraphs $H_i, i \in [p+4]$ are variable vertices.

Step 3: for each clause $C'_i, i \in [p + 1]$, we create a corresponding subgraph H'_i isomorphic to H_i. $\forall i \in [p + 1]$, the only vertices shared by H_i and H'_i are the variable vertices corresponding to C'_i. Vertices $y_{6,i}$ in H_i are renamed $y'_{6,i}$ in H'_i. Let $\chi(x)$ denote the color of any vertex x of G.

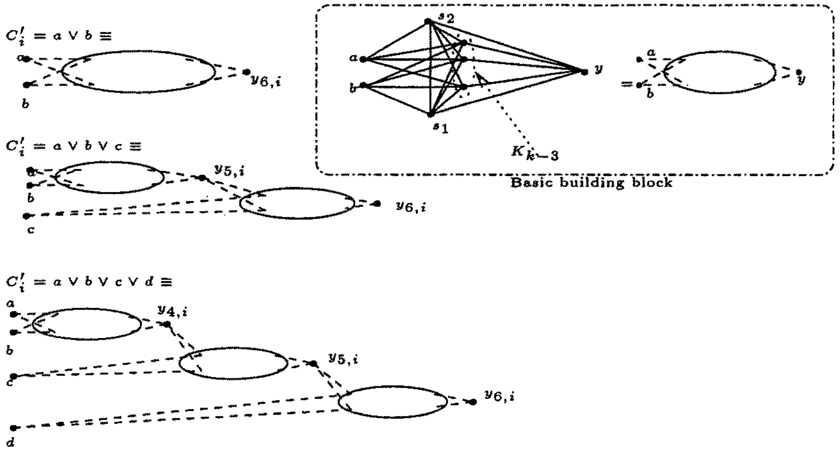

Fig. 1. Basic building block for $H_i, i \in [p+4]$, and subgraphs H_i generated by 2-, 3- and 4-clauses.

Fact 2 *$\forall i \in [p+4], \forall j \in [p+1]$, for any valid k-coloring of the vertices of H_i (resp. H'_j), there exists one which gives to $y_{6,i}$ (resp. $y'_{6,j}$) a color shared by either*

- *a or b if C'_i (resp. C'_j) is of size 2.*
- *a or b or c if C'_i (resp. C'_j) is of size 3.*
- *a or b or c or d if C'_i (resp. C'_j) is of size 4.*

Fact 3 *$\forall i \in [p+4], \forall j \in [p+1]$, any valid k-coloring of the vertices of H_i (resp. H'_j) assigning the same color to each variable vertex of the clause C'_i (resp. C'_j), forces $y_{6,i}$ (resp. $y'_{6,j}$) to have this color.*

Step 4: The subgraphs constructed in steps 1, 2 and 3 are linked to a subgraph W according to figure 2. This terminates the construction of G which now contains $6pk + 8k + 2n + 6$ vertices and is therefore of polynomial size. We have:

Fact 4 *$\chi(x_{n+1}) = \chi(v_2)$ and $\chi(\overline{x}_{n+1}) = \chi(v_1)$.*

Fact 5 *In a valid k-coloring of G, let $\chi_1, ..., \chi_{k-2}$ denote the set of colors used to color K_{k-2} in W, and $\chi_1, ..., \chi_k$ the total set of colors. We have*

$$\forall i \in [n+3], \{\chi(x_i), \chi(\overline{x}_i)\} = \{\chi_{k-1}, \chi_k\} = \{\chi(v_1), \chi(v_2)\}$$

Fact 6 *$\forall i \in [p]$, let $C'_i = x_{i_1} \vee x_{i_2} \vee x_{i_3} \vee x_{n+1}$ denote a 4-clause of C'. We have*

$$(\chi(x_{i_1}) = \chi(v_1)) \vee (\chi(x_{i_2}) = \chi(v_1)) \vee (\chi(x_{i_3}) = \chi(v_1))$$

We now let X' in the instance of Opt(Aggravated k-Colorability) denote the set of variable vertices built from U'. Recall that to one variable in U' corresponds one variable vertex. Therefore, X' does not contain any of the variables from

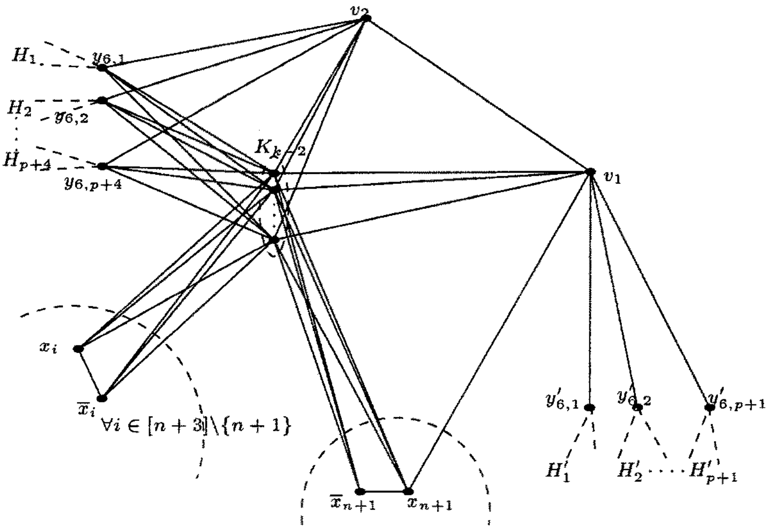

Fig. 2. The subgraph W of G.

the set $\{x_i, \overline{x}_i\}_{i\in\{p+1,p+2,p+3,p+4\}}$. We fix the special vertex to v_1. Proof of the reduction of theorem 3 can be found in [14]. □

We now prove theorem 1 using the following reduction. For any graph G instance of *Opt*(Aggravated k-Colorability), we create a set of examples $\mathcal{E}$ described over a set of predicates $a_i(.), i \in [n]$, and constant symbols $\{l_i\}, i \in [n]$; $\{m_{i,j}\}, (i,j) \in E$; $\{s_{i,x}\}, i \in X'$ (there are $n + |E| + |X'|$ constant symbols). We let x denote the special vertex of the instance of *Opt*(Aggravated k-Colorability).

- $S^+ = \{q(l_i) \leftarrow \wedge_{j \neq i} a_j(l_i) : 1 \leq i \leq n\}$. The weight of these positive examples is $w^+ = n$.
- $S_1^- = \{q(m_{i,j}) \leftarrow \wedge_{k \notin \{i,j\}} a_k(m_{i,j}) : (i,j) \in E\}$. The weight of these negative examples is $w^- = w^+ = n$.
- $S_2^- = \{q(s_{i,x}) \leftarrow \wedge_{k \notin \{i,x\}} a_k(s_{i,j}) : (i,x) \notin E \wedge i \in X'\}$. The weight of these negative examples is $w'^- = 1$.

The proof of theorem 1 follows from the proof of the two following propositions:

Proposition 1 *From any k-function-free Horn clauses making t errors on $S^+ \cup S_1^- \cup S_2^-$, we can build in polynomial time a feasible solution to Opt(Aggravated k-Colorability) which gives the same color as x to at most t elements of X'.*

Proof. Suppose that $t \geq n$. In that case, we can use the trivial coloring of the instance of *Opt*(Aggravated k-Colorability). Since $X' \subset X$, there are trivially at most $t \geq n$ elements of X' colored by the same color as x.
Suppose now that $t < n$. In that case, any element of weight n is well classified. Let $\{h_1, h_2, ..., h_k\}$ denote the set of clauses solution to *Opt*(Weighted-Approx(k-function-free Horn clauses)). We can suppose without loss of generality that any

predicate is absent of at most one clause. Otherwise, we can add this predicate to all clauses but one (in which it is absent), and this does not increase the number of errors since it only forces all positive examples to θ-subsume exactly one clause among the k. The colorability assigned is the following one : $\forall i \in [n], \forall j \in [k]$, if $a_i(.) \not\subset h_j$, then give color j to vertex x_i.
This is a valid k-colorability, otherwise some examples of weight n would be misclassified. The t errors are made on examples of weight 1. These examples represent distinct vertices having the same color as x. Note that a single clause is responsible for all the errors : $h_{\chi(x)}$.

Proposition 2 *Any feasible solution to Opt(Aggravated k-Colorability) which gives the same color as x to at most t elements of X' can be transformed in polynomial time into a feasible solution to Opt(Weighted-Approx(k-function-free Horn clauses)) which makes at most t errors over the examples.*

Proof. The k clauses of the solution to *Opt*(Weighted-Approx(k-function-free Horn clauses)) are defined by:

$$\forall j \in [k], h_j \equiv q(X) \leftarrow \wedge_{i\in[n]:\chi(x_i)\neq j} a_i(X)$$

These clauses do not make errors on examples of weight n. The only errors made are on examples of weight 1 corresponding to vertices having the same color as x. Note that in our construction, only one clause makes all the errors : $h_{\chi(x)}$.

Theorem 1 now follows from propositions 1 and 2.

4 Result on function-free Horn clauses having polylogarithmic size

Let QP denote the class of problems admitting quasi-polynomial time deterministic algorithms. A function of n is quasi-polynomial iff $f(n) \leq n^{\log^c n}$ for some constant c. Many results have introduced the class QP, such as for example [1], to point out the fact that hard-to-solve or approximate problems (such as *Opt*(Aggravated 3-SAT), and therefore *Opt*(Weighted-Approx(k-function-free Horn clauses))) might not even admit quasi-polynomial time approximation algorithms. We are going to use this fact to prove our next result. We now prove theorem 4 below.

Theorem 4 *If $NP \not\subset QP$, $\forall d > 0$ a constant, Opt(Weighted-Approx($\log^{d+2} n$-function-free Horn clauses)) is not approximable to within $n^{\log^d n}$.*

In order to do this, we highlight a correlation between the size of the formula and its error rate. To that effect, we multiply $\log^{d+2} n = K$ instances of *Opt*(Weighted-Approx(k-function-free Horn clauses)) by concatenating the tail of the examples to form new ones, thus described over a set of $n \times K$ predicates (plus the inferred predicate $q(.)$). Each predicate taken from the initial set serves to create exactly K new predicates. This can be viewed as making set products among the set of tails of the examples of the instance of *Opt*(Weighted-Approx(k-function-free Horn clauses)) we used to prove theorem 1. One of these examples could be represented *e.g.* as follows:

$$q(l_i') \leftarrow \left(\wedge_{j=1;j\neq i}^{j=n} a_{j,1}(l_i')\right) \wedge \left(\wedge_{j=1;j\neq i}^{j=n} a_{j,2}(l_i')\right) \wedge ... \wedge \left(\wedge_{j=1;j\neq i}^{j=n} a_{j,K}(l_i')\right)$$

The subscript in $a_{i,j}(.)$ denotes the j^{th} copy of the initial predicate $a_i(.)$. The example represented has each of its K predicate parts coming from the copy of an initial example from S^+ (we shall write that this example comes from $(S^+)^K$). The new examples are described as follows :

- S_K^+ is the subset $(S^+)^K$. Their weight is n^K.
- $S_{1,K}^- = (S^+ \cup S_1^-)^K \backslash S_K^+$. their weight is n^K.
- $S_{2,K}^- = (S_2^-)^K$. their weight is 1.

Due to the increase in the number of examples, the initial set of $n + |E| + |X'|$ constant symbols is replaced by a new larger set of size $(n + |E|)^K + |X'|^K$. Note that this new set of examples is created in quasi-polynomial time. For the sake of brevity, we let c_* denote the minimal error of a feasible solution to *Opt*(Weighted-Approx(k-function-free Horn clauses)), and c'_* the minimal error of a feasible solution to *Opt*(Weighted-Approx(K-function-free Horn clauses)).

Proposition 3 $c_* < n$

Proof follows from the fact that the instance of *Opt*(Aggravated k-Colorability) which serves to build the instance of *Opt*(Weighted-Approx(k-function-free Horn clauses)) is always k-colorable. Proposition 3 comes from the construction technique of proposition 2 : any example of weight n is well classified, and there are at most $n-1$ examples of weight 1. According to propositions 1 and 2, we can suppose that each predicate is absent from at most one clause. Let $c_*(c_{\chi(x)})$ denote the minimal error of the clause that does not contain the predicate $a_x(.)$ corresponding to the special vertex of the instance of *Opt*(Aggravated k-Colorability). From propositions 1 and 2, proposition 3 can be refined :

Proposition 4 $c_* = c_*(h_{\chi(x)})$

Proposition 5 $c'_* \geq (c_*)^K$.

Proof. Whenever an example from the set $S_K^+ \cup S_{1,K}^-$ is badly classified, proposition 3 gives the result: $c'_* \geq n^K > (c_*)^K$. Suppose that all examples from the set $S_K^+ \cup S_{1,K}^-$ are well classified. Any error is necessarily due to an example of the set $S_{2,K}^-$. The only type of clause that can cause these errors is a clause of type $q(X) \leftarrow P_1 \wedge P_2 ... \wedge P_K$ where $\forall j \in [K]$, P_j is a clause described over the set of predicates $\{a_{i,j}\}_{i\in[n]}$ such that P_j makes errors on S_2^-. Since the error of each P_j is at least $c_*(h_{\chi(x)})$, which is c_* (proposition 4), the error of the conjuction $(\wedge)$ is at least the product of the minimal error of each part, $(c_*)^K$. The overall error of the set of clauses is thus at least equal to this quantity.

Proposition 6 *Let h denote a feasible solution to Opt(Weighted-Approx(K-function-free Horn clauses)), whose error is a. Then we can find a feasible solution to Opt(Weighted-Approx(k-function-free Horn clauses)) whose error is no more than $a^{\frac{1}{K}}$.*

Proof. Recall that the graph we constructed from the instance of the problem *Opt*(Aggravated 3-SAT) is always k colorable. Therefore, there always exist a set of k-function-free Horn clauses consistent with $S^+ \cup S_1^-$ (proposition 2), having error $< n$. In h, whenever an example from the set $S_K^+ \cup S_{1,K}^-$ is badly classified, proposition 3 gives the result since the error of h is at least $c'_* \geq n^K$. Suppose that all examples from the set $S_K^+ \cup S_{1,K}^-$ are well classified. Any error is necessarily due to an example of the set $S_{2,K}^-$. The only type of clause that can cause these errors is a clause of type $q(X) \leftarrow P_1 \wedge P_2 ... \wedge P_K$ where $\forall j \in [K]$,

1. P_j is a subset of predicates described over the set of predicates $\{a_{i,j}\}_{i \in [n]}$, and
2. the clause isomorphic to $q(X) \leftarrow P'_j$ described over the set $\{a_i\}_{i \in [n]}$, obtained by replacing each $a_{l,j} \in P_j$ by a_l in P'_j, makes errors on S_2^-.

Note that the error of the conjuction ($\wedge$) is the product of errors of each part $P_j, j \in [K]$ on S_2^-. So, the part P_* over $P_1, ..., P_K$ leading to the least number of errors on S_2^- makes an error that is at most $a^{\frac{1}{K}}$. Now, construct the set h' of $(k-1)$-function-free Horn clauses of proposition 2 with all clauses except $h_{\chi(x)}$, and add (for $h_{\chi(x)}$) the clause corresponding to the part P_* (it is $q(X) \leftarrow P'_*$ as described in point 2 above). The overall error of h' does not exceed $a^{\frac{1}{K}}$.

Proposition 7 $c'_* \leq (c_*)^K$

The proof of this proposition follows simply if we calculate the K-time cross-product of the solution realizing the cost c_*. We obtain a set of k^K clauses, and the construction can be realized in quasi-polynomial time.

From this, it comes that $c'_* = (c_*)^K$. We now prove theorem 4 *ad absurdum.* Suppose that *Opt*(Weighted-Approx($\log^{d+2} n$-function-free Horn clauses)) is approximable to within $n^{\log^d n}$. In quasi-polynomial time, from any instance of *Opt*(Weighted-Approx(k-function-free Horn clauses)), we build an instance of *Opt*(Weighted-Approx(K-function-free Horn clauses)) following the procedure described at the beginning of this section. We can find an element of K-function-free Horn clauses whose error does not exceed $(Kn)^{\log^d(Kn)} c'_*$ (hypothesis). Thus, we can find a solution to *Opt*(Weighted-Approx(k-function-free Horn clauses)) whose cost is approximately no more than $\left((Kn)^{\log^d(Kn)} c'_*\right)^{\frac{1}{K}}$ (proposition 6). But (using propositions 5 and 7)

$$\left((Kn)^{\log^d(Kn)} c'_*\right)^{\frac{1}{K}} = \left((Kn)^{\log^d(Kn)} c'_*\right)^{\frac{1}{\log^{d+2} n}} \leq (Kn)^{\frac{1}{\log(Kn)}} c_* = \mathcal{O}(c_*)$$

This contradicts theorem 1, since we obtain an approximation of *Opt*(Weighted-Approx(k-function-free Horn clauses)) to within a constant factor.

5 Consequences on learnability

In this paper, we have essentially presented two structural complexity results. Their purpose, is to prove that Horn Clauses display very severe error rates when used for learning in complex domains, and they extend our previous results [9]. It should be noted that as in this previous work, in order to obtain general properties, we have studied general function-free Horn-clauses but the proofs of our theorems are made in such a manner (using simple ILP formalisms) as to remain valid for the more specialized classes encoutered in ILP. So, both our non-approximability results also apply to many subsets that have led to theoretical studies in the PAC-learning model. Classes for which this result applies are subclasses of the following classes where the number of clauses is limited to the values of theorems 1 and 4 : ij-determinate non recursive Horn clauses [13] where i and j are any integer constants satisfying $i \geq 0$ and $j > 0$, and l-local Horn clauses [3] where l is any integer constant satisfying $l > 0$.

References

1. S. Arora. Probabilistic checking of proofs and hardness of approximation problems. Technical Report CS-TR-476-94, Princeton University, 1994.
2. W. Buntine. Generalized subsumption and its applications to induction and redundancy. *Artificial Intelligence*, 36:149–176, 1988.
3. W.W. Cohen. Pac-learning nondeterminate clauses. In *Proceedings of the Twelfth National Conference on Artificial Intelligence, AAAI'94*, pages 676–681, 1994.
4. W.W. Cohen. Pac-learning recursive logic programs: Efficient algorithms. *Journal of Artificial Intelligence Research*, 2:501–539, 1995.
5. W.W. Cohen. Pac-learning recursive logic programs: Negative results. *Journal of Artificial Intelligence Research*, 2:541–571, 1995.
6. S. Dzerovski, S.H. Muggleton, and S. Russel. Pac-learnability of determinate logic programs. In *Proceedings of COLT-92*, pages 128–137, 1992.
7. E.M. Gold. Language indentification in the limit. *Information and Control*, 10:447–474, 1967.
8. K-U. Höffgen and H.U. Simon. Robust trainability of single neurons. In *Proc. of the 5 th International Conference on Computational Theory*, 1992.
9. P. Jappy, R. Nock, and O. Gascuel. Negative robust learning results for horn clause programs. In *Proceedings of ICML'96*, pages 258–265, 1996.
10. M. Kearns, M. Li, L. Pitt, and L.G. Valiant. On the learnability of boolean formulae. In *Proceedings of STOCS'87*, pages 285–294, 1987.
11. J.U. Kietz and S. Dzeroski. Inductive logic programming and learnability. *Sigart Bulletin*, 5:22–32, 1994.
12. S.H. Muggleton. Bayesian inductive logic programming. In *Proceedings of the Seventh Workshop on COmputational Learning Theory*, 1994.
13. S.H. Muggleton and C. Feng. *Efficient induction of logic programs*. Inductive Logic Programming. Academic Press, New York, 1992.
14. R. Nock and P. Jappy. On the hardness of approximating function-free horn clauses. Technical Report LIRMM-RR-98017, LIRMM, 1998.
15. L.G. Valiant. A theory of the learnable. *Association for Computing Machinery Communications*, 27:1134–1142, 1984.