Probabilistic Constraint Logic Theories

Marco Alberti[1], Elena Bellodi[2], Giuseppe Cota[2], Evelina Lamma[2], Fabrizio Riguzzi[1], and Riccardo Zese[2]

[1] Dipartimento di Matematica e Informatica – University of Ferrara
Via Saragat 1, I-44122, Ferrara, Italy
[2] Dipartimento di Ingegneria – University of Ferrara
Via Saragat 1, I-44122, Ferrara, Italy
name.surname@unife.it

Abstract. Probabilistic logic models are used ever more often to deal with the uncertain relations typical of the real world. However, these models usually require expensive inference procedures. Very recently the problem of identifying tractable languages has come to the fore. In this paper we consider the models used by the learning from interpretations ILP setting, namely sets of integrity constraints, and propose a probabilistic version of them. A semantics in the style of the distribution semantics is adopted, where each integrity constraint is annotated with a probability. These probabilistic constraint logic models assign a probability of being positive to interpretations. This probability can be computed in a time that is logarithmic in the number of ground instantiations of violated constraints. This formalism can be used as the target language in learning systems and for declaratively specifying the behavior of a system. In the latter case, inference corresponds to computing the probability of compliance of a system's behavior to the model.

1 Introduction

Probabilistic logic models are gaining popularity due to their successful application in a variety of fields, such as natural language processing, information extraction, bioinformatics, semantic web, robotics and computer vision.

However, these models usually require expensive inference procedures. Very recently the problem of identifying tractable languages has come to the fore. Proposals such as Tractable Markov Logic [9], Tractable Probabilistic Knowledge Bases [25,17] and fragments of probabilistic logics [24,16] strive to achieve tractability by limiting the form of sentences.

In the ILP field, the learning from interpretation setting [8,2,7] offers advantages in terms of tractability with respect to the learning from entailment setting. In learning from interpretations, the logic theories are sets of integrity constraints and the examples are interpretations. The coverage problem there consists in verifying whether the constraints are satisfied in the interpretations. This problem is simpler than checking whether an atom follows from a logic program because the constraints can be considered in isolation: the interpretation satisfies the constraints iff it satisfies all of them individually. A first attempt

to this problem was presented in [11,10] where authors described the algorithm LFI-ProbLog for learning ProbLog programs from partial interpretations.

Our aim is to consider a probabilistic version of sets of integrity constraints with a semantics in the style of the distribution semantics [23]. Each integrity constraint is annotated with a probability and a model assigns a probability of being positive to interpretations. This probability can be computed in a time that is logarithmic in the number of groundings of the constraints that are violated.

The formalism we propose, Probabilistic Constraint Logic Theories (PCLT), has a variety of applications. It can be used as the target language of a learning system, thus lifting the learning from interpretations ILP setting to the probabilistic case.

It is also useful for system verification or the problem of checking whether a system's behaviour is compliant to a specification [14].

The system's specification can be given in a number of ways. Specifications based on logic are appropriate to many applications since they provide a non-ambiguous semantics. Additionally, Computational Logic frameworks come with operational semantics with formal correctness properties, which can be used for verification; such a feature has motivated the mapping of heterogeneous specification formalisms onto ones based on computational logic [15]. For instance, the $\mathcal{S}$CIFF framework [1], applied to verification of multi-agent systems, medical guidelines, electronic commerce, is composed of a language for specification with an abductive logic programming semantics, and a sound and complete proof procedure, so that behaviours found correct by the proof procedure are indeed correct according to the declarative semantics, and vice-versa.

The essence of many logic-based specification languages (including $\mathcal{S}$CIFF) is to provide a form of integrity constraints: logic formulas that represent protocols, rules or guidelines, and are required to be satisfied by the system's behaviour.

A binary notion of compliance (where the behaviour is compliant if it satisfies all the integrity constraints, and simply non-compliant otherwise) may not be suited to applications where some amount of non-compliance is inevitable, and quantifying non-compliance is more important than just detecting it. One form of flexibility is to allow the designer to specify the integrity constraints as optional, and to express how important each constraint is. PCTL allows precisely this.

The paper is organized as follows. In Sect. 2, we recall the notion of Constraint Logic Theory, and in Sect. 3 we define PCLT. In Sect. 4 we show how to compute the probability of compliance given the compliance to each integrity constraint. Sect. 5 discusses PCLTs in more detail while Sect. 6 introduces an extended version of CLTs. Sect. 7 discusses related work. Finally, Sect. 8 concludes the paper with some remarks on future work.

2 Constraint Logic Theories

A Constraint Logic Theory (CLT) [7] T is a set of integrity constraints (ICs) C of the form

$$L_1, \ldots, L_b \rightarrow A_1; \ldots; A_h \tag{1}$$

where the L_is are logical literals. Their conjunction $L_1, \ldots, L_b$ is called the *body* of the IC and is indicated with $Body(C)$. The A_j are logical atoms, where the semicolon stands for disjunction, thus $A_1; \ldots; A_h$ is a disjunction of atoms called the *head* of the IC and indicated with $Head(C)$.

Together with a CLT T, we may have a background knowledge B on the domain which is a normal logic program that can be used to represent domain-specific knowledge.

CLTs can be used to classify Herbrand interpretations, i.e., sets of ground facts, that represent for example the behaviour of the system undergoing verification. Given a Herbrand interpretation I (in the following it will be called simply interpretation), the given background knowledge B is used to complete the information in I. Basically, instead of simply considering I, we consider a model $M(B \cup I)$ which follows the Prolog semantics (i.e. Clark completion [4]) where I is interpreted as a set of ground facts. In this way, all the facts of I are true in $M(B \cup I)$, moreover $M(B \cup I)$ can contain new facts derived from I using B.

Given an interpretation I, a background knowledge B and a CLT T we can ask whether T is true in I given B. Formally, an IC C is *true in an interpretation I given a background knowledge B*, written $M(B \cup I) \models C$, if for every substitution θ for which $Body(C)$ is true in $M(B \cup I)$, there exists a disjunct in $Head(C)$ that is true in $M(B \cup I)$. If $M(B \cup I) \models C$ we say that I *satisfies the constraint C given B*; if $M(B \cup I) \not\models C$ we say that I does not satisfy C. If every IC of a CLT T is true in it, then T is *true in an interpretation I given B* and we write $M(B \cup I) \models T$. We also say I *satisfies T given B* or that I *is positive given T and B*.

If all the variables that appear in the head also appear in the body, then the logical clause is *range-restricted*. As shown in [5], the truth of a range-restricted IC in an interpretation I with range-restricted background knowledge B can be tested by asking the goal

$$? - Body(C), \neg Head(C).$$

against a Prolog database containing the atoms of I as facts together with the rules of the normal program B. By $\neg Head(C)$ we mean $\neg A_1, \ldots, \neg A_h$ so the query is

$$? - L_1, \ldots, L_b, \neg A_1, \ldots, \neg A_h. \tag{2}$$

If the query fails, C is true in I given B, otherwise C is false in I given B. If B is range-restricted, every answer to a query Q against $B \cup I$ completely instantiates Q, i.e., it produces an element of $M(B \cup I)$. So the queries $\neg A_j$ are ground when they are called and no floundering occurs.

Example 1 (Bongard Problems). Introduced by the Russian scientist M. Bongard in his book [3], the Bongard Problems consist of a number of pictures, some positive and some negative, and is usually used in learning problems aimed at learning a description which correctly classify the most figures, i.e., at discriminating between the two classes.

The pictures contain different shapes with different properties, such as small, large, pointing down, ... and different relationships between them, such as inside, above, ... Figure 1 shows some of these pictures.

Each picture can be described by an interpretation. Consider the left picture. It consists of a large triangle that includes a small square that, in turn, includes a small triangle. This picture can be described using the interpretation

$$I_l = \{triangle(0), large(0), square(1), small(1), inside(1,0), \\ triangle(2), inside(2,1)\}$$

Moreover, suppose you are given the background knowledge B:

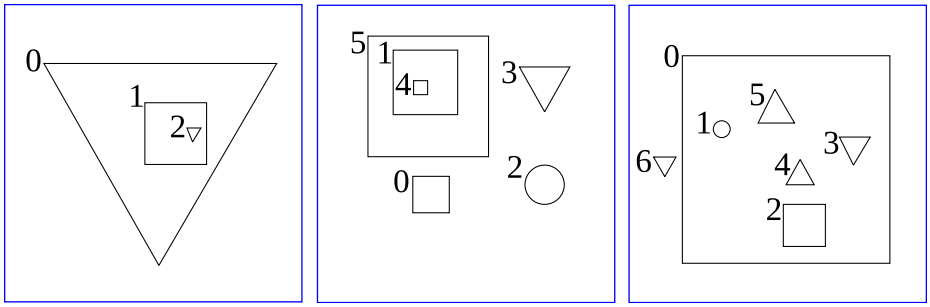


Fig. 1. Bongard pictures.

$$in(A,B) \leftarrow inside(A,B).$$
$$in(A,D) \leftarrow inside(A,C), in(C,D).$$

Thus $M(B \cup I_l)$ will contain the atoms $in(1,0)$, $in(2,1)$ and $in(2,0)$. Given the IC

$$C_1 = triangle(T), square(S), in(T,S) \rightarrow false$$

stating that a figure satisfying the IC cannot contain a triangle inside a square, C_1 is false in I_l given B because triangle 2 is inside square 1.

In the central picture instead C is true given B because the only triangle is outside any square, while in the rightmost picture C_1 is false again because of the presence of triangles 3, 4 and 5 inside square 0.

3 Probabilistic Constraint Logic Programming

A Probabilistic Constraint Logic Theory (PCLT) is a set of probabilistic integrity constraints (PICs) of the form

$$p_i \ :: \ L_1, \ldots, L_b \rightarrow A_1; \ldots; A_h \tag{3}$$

Each constraint C_i is associated with a real value in $[0,1]$ which defines its probability. A PCLT T is sometimes defined also as a set $\{(C_1, p_1), \ldots, (C_n, p_n)\}$. A PCLT T defines a probability distribution on ground constraint logic theories called worlds in this way: for each grounding of each IC, we include the grounding in a world with probability p_i and we assume all groundings to be independent. The notion of world as a theory is similar to the notion of world in ProbLog [6] where a world is a normal logic program. Let us assume that constraint C_i has n_i groundings called $C_{i1}, \ldots, C_{in_i}$. Let us call the ICs C_{ij} *instantiations* of C_i. Thus, the probability of a world w is given by the product:

$$P(w) = \prod_{i=1}^{m} \prod_{C_{ij} \in w} p_i \prod_{C_{ij} \notin w} (1 - p_i)$$

where m is the number of PICs. $P(w)$ so defined is a probability distribution over the set of worlds W. The probability $P(\oplus|w, I)$ of the positive class given an interpretation I, a background knowledge B and a world w is defined as the probability that I satisfies w given B[3]. Its value is $P(\oplus|w, I) = 1$ if $M(B \cup I) \models w$ and 0 otherwise. The probability $P(\oplus|I)$ of the positive class given an interpretation I and a background B is the probability of I satisfying a PCLT T given B. From now on we always assume B as given and we do not mention it again. $P(\oplus|I)$ is given by

$$P(\oplus|I) = \sum_{w \in W} P(\oplus, w|I) = \sum_{w \in W} P(\oplus|w, I) P(w|I) = \sum_{w \in W, M(B \cup I) \models w} P(w) \quad (4)$$

The probability $P(\ominus|I)$ of the negative class given an interpretation I is the probability of I not satisfying T and is given by $1 - P(\oplus|I)$.

Example 2 (Example 1 continued). Consider the PCLT

$$\{C_1 = 0.5 \ :: \ triangle(T), square(S), in(T, S) \rightarrow false\}$$

In the left picture of Figure 1, considering the interpretation

$$I_l = \{triangle(0), large(0), square(1), small(1), inside(1, 0), \\ triangle(2), inside(2, 1)\}$$

There are two different instantiations for the IC C_1:

$$C_{11} = (C_1, \{T/0, S/1\}) \\ C_{12} = (C_1, \{T/2, S/1\})$$

Under I_l there are thus four possible worlds

$$\{\emptyset, \{C_{11}\}, \{C_{12}\}, \{C_{11}, C_{12}\}\}$$

[3] B is omitted from the formula for the sake of brevity.

and for the first two of them $M(B \cup I_l) \models w_i$, thus $P(\oplus|I_l) = P(w_1) + P(w_2) = 0.25 + 0.25 = 0.5$. In the central picture there are four different instantiations for C_1, thus we can build 16 worlds. The interpretation I_c is verified in all of them since the constraint is never violated irrespective of the instantiation, thus the probability is $P(\oplus|I_c) = 1$. Finally, the third figure has 8 different instantiations for IC C_1 and so 256 different worlds. Only 32 worlds satisfy the interpretation I_r, and the probability is $P(\oplus|I_r) = 0.125$.

4 Inference with Probabilistic Constraint Logic Theories

Computing $P(\oplus|I)$ with Formula (4) is impractical as there is an exponential number of worlds. In fact, as seen in Example 2, the number of worlds is exponential in the number of instantiations because, in order to build each world, we must decide whether to include each instantiation of every IC in the world.

Practically, we can associate a Boolean random variable X_{ij} to each instantiated constraint C_{ij}: if C_{ij} is included in the world X_{ij} takes on value 1. Moreover, $P(X_{ij}) = P(C_{ij}) = p_i$ and $P(\overline{X_{ij}}) = 1 - P(C_{ij}) = 1 - p_i$. Let $\mathbf{X}$ be the set of the X_{ij} variables. These variables are all mutually independent. A valuation ν is an assignment of a truth value to all variables in $\mathbf{X}$. There is clearly a one to one correspondence between worlds and valuations. A valuation can be represented as a set containing X_{ij} (if C_{ij} is included in the corresponding world) or $\overline{X_{ij}}$ (if C_{ij} is not included in the corresponding world) for each X_{ij}, and corresponds to the Boolean formula ϕ_ν:

$$\phi_\nu = \bigwedge_{i=1}^{m} \bigwedge_{X_{ij} \in \nu} X_{ij} \bigwedge_{\overline{X_{ij}} \in \nu} \overline{X_{ij}}.$$

Since all the X_{ij} variables are independent, the probability of ϕ_ν being true is

$$P(\phi_\nu) = \prod_{i=1}^{m} \prod_{C_{ij} \in w} p_i \prod_{C_{ij} \notin w} (1 - p_i).$$

As seen above, we can assign to each world w a valuation ν_w of $\mathbf{X}$ in this way: $X_{ij} \in \nu_w$ iff $C_{ij} \in w$ and $\overline{X_{ij}} \in \nu_w$ iff $C_{ij} \notin w$.
Suppose a ground IC C_{ij} is violated in I. The worlds where X_{ij} holds in the respective valuation are thus excluded from the summation in Formula (4). We must keep only the worlds where $\overline{X_{ij}}$ holds in the respective valuation for all ground constraints C_{ij} violated in I. So I satisfies all the worlds where the formula

$$\phi = \bigwedge_{i=1}^{m} \bigwedge_{M(B \cup I) \not\models C_{ij}} \overline{X_{ij}}$$

is true in the respective valuations, so

$$P(\oplus|I) = P(\phi) = \prod_{i=1}^{m} (1 - p_i)^{n_i} \tag{5}$$

where n_i is the number of instantiations of C_i that are not satisfied in I, since the random variables are all mutually independent. Since computing a^b is $O(\log b)$, $P(\oplus|I)$ can be computed in a time that is logarithmic in the number of groundings of constraints that are violated.

Each constraint may have a different number of violated groundings, which may result in a larger weight associated with constraints with many groundings. However, the parameters should be learned from data in order to maximize the likelihood, so the parameters should be adjusted to take into account the number of groundings.

Example 3 (Example 2 continued). Consider the PCLT of Example 2. In the left picture of Figure 1 the body of C_1 is true for the single substitution $T/2$ and $S/1$ thus $n_1 = 1$ and $P(\oplus|I_l) = 0.5$. In the right picture of Figure 1 the body of C_1 is true for three couples (triangle, square) thus $n_1 = 3$ and $P(\oplus|I_r) = 0.125$. These results clearly correspond to those seen in Example 2.

5 Discussion

PCLT can be seen as defining a conditional probability distribution over a random variable C representing the class, positive (+) or negative (-), given the value of the random variables $A_1, \ldots, A_n$ representing the Herbrand base.

In other words, we do not want to model the dependence among atoms of the Herbrand base but only the conditional dependence of the class given the value of the atoms, i.e., given an interpretation. Our aim is to build a discriminative model, rather than a generative model, similarly to what is done with conditional random fields [12] that focus on the relationship between class variables and input variables and do not model the relationship among input variables.

A PCLT defines a Bayesian network of the form shown in Figure 2, with the variables associated to ground atoms that are all parents of the class variable. This model differs from a naive Bayes model because there the input variables (ground atoms) are all children of the class variable. This is a significant difference because the model in Figure 2 can have up to 2^n parameters if n is the number of ground atoms.

The assumption of independence of the constraints may seem restrictive but PCLT can model any conditional probabilistic relationship between the class variable and the ground atoms. For example, suppose you want to model a general conditional dependence between the class atom and a Herbrand base containing two atoms: a and b. This dependence can be represented with the Bayesian network of Figure 3, where the conditional probability table (CPT) has four parameters, $p_1, \ldots, p_4$, so it is the most general. Let us call P' the distribution defined by this network.

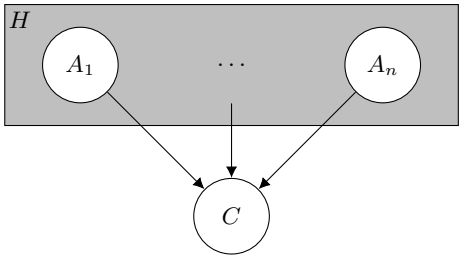
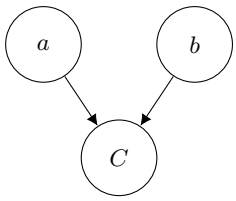


Fig. 2. Bayesian Network representing the dependence between the class of an interpretation and the Herbrand base H.



$P'(C\|a,b)$		C	
a	b	$-$	$+$
0	0	$1-p_1$	p_1
0	1	$1-p_2$	p_2
1	0	$1-p_3$	p_3
1	1	$1-p_4$	p_4

Fig. 3. Bayesian Network representing the dependence between class C and a, b.

This model can be represented with the following PCLT

$$C_1 = 1 - p_1 :: \neg a, \neg b \rightarrow \mathit{false} \tag{6}$$
$$C_2 = 1 - p_2 :: \neg a, b \rightarrow \mathit{false} \tag{7}$$
$$C_3 = 1 - p_3 :: a, \neg b \rightarrow \mathit{false} \tag{8}$$
$$C_4 = 1 - p_4 :: a, b \rightarrow \mathit{false} \tag{9}$$

In fact, consider the interpretation $\{\}$ that assigns value false to each atom of the Herbrand base. The probability that the class variable assumes value $+$ is

$$P(C = +|\neg a, \neg b) = 1 - (1 - p_1) = p_1$$

since only constraint C_1 is violated, so $P(C = +|\neg a, \neg b) = P'(C = +|\neg a, \neg b)$. Similarly we can show that for the other possible interpretations, the probability assigned to the positive class by the above PCLT coincide with the one assigned by the Bayesian network of Figure 3.

Modeling the dependence between C and a, b with the above PCLT is equivalent to representing the Bayesian network of Figure 3 with the Bayesian network of Figure 4, where a Boolean variable X_i represents whether constraint C_i is included in the world (i.e., if it is enforced) and a Boolean variable Y_i whether constraint C_i is violated. Let us call P'' the distribution defined by this network. The conditional probability tables for nodes X_is are $P''(X_i = 1) = 1 - p_i$, those

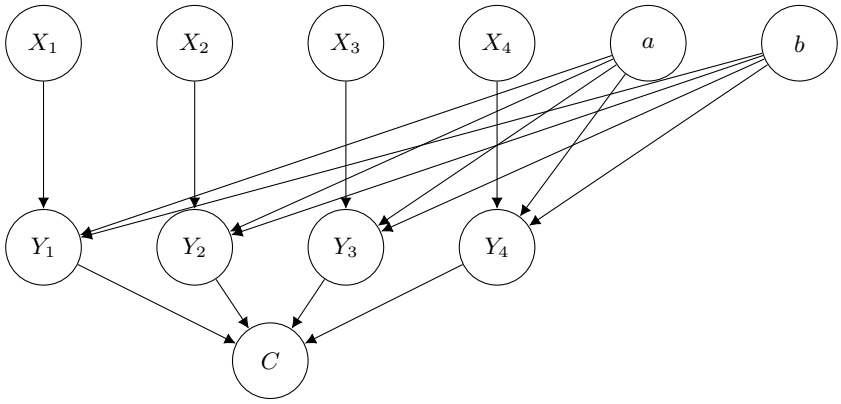


Fig. 4. Bayesian Network modeling the distribution P'' over C, a, b, $X_1, \ldots X_4$, $Y_1, \ldots Y_4$.

for nodes Y_is encode the deterministic functions

$$\begin{aligned} Y_1 &= X_1 \wedge \neg a \wedge \neg b \\ Y_2 &= X_2 \wedge \neg a \wedge b \\ Y_3 &= X_3 \wedge a \wedge \neg b \\ Y_4 &= X_4 \wedge a \wedge b \end{aligned}$$

and that for C encodes the deterministic function

$$C = \neg Y_1 \wedge \neg Y_2 \wedge \neg Y_3 \wedge \neg Y_4$$

where C is interpreted as a Boolean variable with 1 corresponding to + and 0 to -. If we want to compute $P''(C|\neg a, \neg b)$ we get

$$\begin{aligned} P''(C|\neg a \neg b) &= \sum_{\mathbf{Y},\mathbf{X}} P''(X_1) \ldots P''(X_4) P''(Y_1|X_1, \neg a, \neg b) \ldots P''(Y_4|X_4, \neg a, \neg b) \\ &\quad P''(C|Y_1, Y_2, Y_3, Y_4) = \\ &= p_1 \sum_{X_2,X_3,X_4,Y_2,Y_3,Y_4} P''(X_2) \ldots P''(X_4) P''(C|Y_1 = 0, Y_2, Y_3, Y_4) \\ &\quad P''(Y_2|X_2, \neg a, \neg b) \ldots P''(Y_4|X_4, \neg a, \neg b) = \\ &= p_1 \sum_{X_2,X_3,X_4} P''(X_2) \ldots P''(X_4) \\ &\quad P''(C|Y_1 = 0, Y_2 = 0, Y_3 = 0, Y_4 = 0) \\ &\quad P''(Y_2 = 0|X_2, \neg a, \neg b) \ldots P''(Y_4 = 0|X_4, \neg a, \neg b) = \\ &= p_1 \sum_{X_2,X_3,X_4} P''(X_2) \ldots P''(X_4) = \\ &= p_1 \end{aligned}$$

where $\mathbf{X} = \{X_1, \ldots, X_4\}$ and $\mathbf{Y} = \{Y_1, \ldots, Y_4\}$. Similarly, it is possible to show that P and P'' coincide for the other possible interpretations. If we look at the network in Figure 4 we see that the $\mathbf{X}$ variables are mutually unconditionally independent, showing that it is possible to represent any conditional dependence of C from the Herbrand base by using independent random variables. Of course, not assuming independence may result in a finer modeling of the domain. However, this would preclude PCLTs' nice computational properties. Achieving tractability requires approximations and we think that constraint independence is a reasonable assumption, similar to the independence among probabilistic choices in the distribution semantics for PLP.

Moreover, PCLT can compactly encode the dependence because they can take advantage of context specific independences [19]. For example, in the CPT in Table 1 the probability of $C = +$ does not depend on b when a is true. This

$P'(C\|a,b)$		C	
a	b	$-$	$+$
0	0	$1-p_1$	p_1
0	1	$1-p_2$	p_2
1	0	$1-p_3$	p_3
1	1	$1-p_3$	p_3

Table 1. A CPT with context specific independence.

dependence can be encoded with

$$C_1 = 1 - p_1 :: \neg a, \neg b \rightarrow \mathit{false}$$
$$C_2 = 1 - p_2 :: \neg a, b \rightarrow \mathit{false}$$
$$C_3 = 1 - p_3 :: a \rightarrow \mathit{false}$$

PCLT are also related to Markov Logic Networks (MLNs) [20]: similarly to MLNs, PCLT encode constraints on the possible interpretations and the probability of an interpretation depends on the number of violated constraints. However, MLNs encode the joint distribution of the ground atoms and the class, while we concentrate on the conditional distribution of the class given the ground atoms. Given a PCLT, it is possible to obtain an equivalent MLN. For example,

the MLN equivalent to the PCLT (6)-(9) is

$$\begin{aligned}
\ln(1-p_1)\ & \neg a \wedge \neg b \wedge \neg C \\
\ln(p_1)\ & \neg a \wedge \neg b \wedge C \\
\ln(1-p_2)\ & \neg a \wedge b \wedge \neg C \\
\ln(p_2)\ & \neg a \wedge b \wedge C \\
\ln(1-p_3)\ & a \wedge \neg b \wedge \neg C \\
\ln(p_3)\ & a \wedge \neg b \wedge C \\
\ln(1-p_4)\ & a \wedge b \wedge \neg C \\
\ln(p_4)\ & a \wedge b \wedge C
\end{aligned}$$

where C is an atom representing the class. If we compute the conditional probability of C given an interpretation I, we get the same results of the PCLT. In fact, consider the empty interpretation and call P''' the distribution defined by the MLN. We get

$$\begin{aligned}
P'''(C=+|a,b) &= P'''(C=+,a,b)/P'''(a,b) = \\
&= \frac{\frac{e^{\ln(p_1)}}{Z}}{\frac{e^{\ln(1-p_1)}+e^{\ln p_1}}{Z}} = \frac{e^{\ln(p_1)}}{e^{\ln(1-p_1)}+e^{\ln p_1}} = \\
&= \frac{p_1}{1-p_1+p_1} = p_1
\end{aligned}$$

where Z is the partition function. Similarly for the other interpretations. So PCLT are a specialization of MLNs that, by focusing on a simpler problem, allow better performance of inference algorithms.

6 Extensions of CLTs

Integrity constraints can be extended to logical formulas of the form

$$L_1, \ldots, L_b \rightarrow \exists\ (ConjP_1); \ldots; \exists\ (ConjP_n);\ \forall\ \neg(ConjN_1); \ldots; \forall\ \neg(ConjN_m) \quad (10)$$

where the L_is are logical literals and their conjunction $L_1, \ldots, L_b$ represents the body of the IC (as in Section 2), while $ConjP_i$ $(i = 1, \ldots, n)$ and $ConjN_j$ $(j = 1, \ldots, m)$ are conjunctions of literals of the form $A_1, \ldots, A_k$ [13]. The semicolon stands for disjunction, thus $\exists(ConjP_1); \ldots; \exists(ConjP_n); \forall\neg(ConjN_1); \ldots;$ $\forall\neg(ConjN_m)$ is a disjunction of conjunctions of literals.

We will use $Body(C)$ to indicate the body of the IC and $Head(C)$ to indicate the formula $\exists(ConjP_1); \ldots; \exists(ConjP_n); \forall\neg(ConjN_1); \ldots; \forall\neg(ConjN_m)$ and call them respectively the *body* and the *head* of C. All the formulas $ConjP_j$ in $Head(C)$ can also be referred to as P *disjuncts* and all the formulas $ConjN_j$ in $Head(C)$ as N *disjuncts*.

An IC C is *true in an interpretation I given a background knowledge B*, written $M(B \cup I) \models C$, if for every substitution θ for which $Body(C)$ is true in $M(B \cup I)$, there exists a disjunct in $Head(C)$ that is true in $M(B \cup I)$.

Variables in the body are implicitly universally quantified with scope the entire formula; the quantifiers in the head apply to all the variables not appearing in the body.

The truth of an extended IC C in an interpretation I can be tested by running the query $?-Body(C), \neg ConjP_1, \ldots, \neg ConjP_n, ConjN_1, \ldots, ConjN_m.$ against a Prolog database containing the clauses of B and the atoms of I as facts. If B is range-restricted, every answer to an atomic query Q against $B \cup I$ completely instantiates Q, i.e., it produces an element of $M(B \cup I)$. If the query finitely fails the IC is true in I. If the query succeeds, the IC is false in I.

This language extends clausal logic by allowing more complex formulas as disjuncts in the head of clauses. The ICs are more expressive than logical clauses, as can be seen from the query used to test them: for ICs we have the negation of conjunctions, while for clauses we have only the negation of atoms.

7 Related Work

The approach presented here refers to the distribution semantics [23]: a probabilistic theory defines a distribution over non-probabilistic theories by assuming independence among the choices in probabilistic constructs. The distribution semantics has emerged as one of the most successful approaches in Probabilistic Logic Programming and underlies many languages such as Probabilistic Horn Abduction, Independent Choice Logic, PRISM, Logic Programs with Annotated Disjunctions and ProbLog.

In the distribution semantics, the aim is to compute the probability that a ground atom is true. However, performing such inference requires an expensive procedure that is usually based on knowledge compilation. For example, ProbLog [6] and PITA [21,22] build a Boolean formula and compile it into a Binary Decision Diagram from which the computation of the probability is linear in the size of the diagram. However, the compilation procedure is #P in the number of variables. On the contrary, computing the probability of the positive class given an interpretation in a PCLT is logarithmic in the number of variables. This places PCLTs in the recent line of research committed to identifying tractable probabilistic languages.

In addition, a probabilistic program in one of the languages under the distribution semantics defines a probability distribution over normal logic programs called worlds. The distribution is extended to queries and the probability of a query is obtained by marginalizing the joint distribution of the query and the programs. Instead, PCLTs define a conditional probability distribution over a random variable C representing the class, given the value of a set of atoms (an interpretation).

8 Conclusions

We have proposed a probabilistic extension of constraint logic theories for which the computation of the probability of an interpretation being positive is logarithmic in the number of falsified constraints.

In the future we are going to develop a system for learning such probabilistic integrity constraints. A possible way is to exploit Limited-memory BFGS (L-BFGS) [18] for tuning the parameters and constraint refinements for finding good structures. L-BFGS is an optimization algorithm in the family of quasi-Newton methods that approximates the BroydenFletcherGoldfarbShanno (BFGS) algorithm using a limited amount of computer memory.

Acknowledgement This work was supported by the "GNCS-INdAM".

References

1. Alberti, M., Chesani, F., Gavanelli, M., Lamma, E., Mello, P., Torroni, P.: Verifiable agent interaction in abductive logic programming: the *S*CIFF framework. ACM T. Comput. Log. 9(4) (2008), iF: 2.766
2. Blockeel, H., De Raedt, L., Jacobs, N., Demoen, B.: Scaling up inductive logic programming by learning from interpretations. Data Min. Knowl. Discov. 3(1), 59–93 (1999)
3. Bongard, M.M.: Pattern Recognition. Hayden Book Co., Spartan Books (1970)
4. Clark, K.L.: Negation as failure. In: Logic and Data Bases. pp. 293–322 (1977)
5. De Raedt, L., Dehaspe, L.: Clausal discovery. Machine Learning 26(2-3), 99–146 (1997)
6. De Raedt, L., Kimmig, A., Toivonen, H.: ProbLog: A probabilistic Prolog and its application in link discovery. In: 20th International Joint Conference on Artificial Intelligence, Hyderabad, India (IJCAI-05). vol. 7, pp. 2462–2467. AAAI Press, Palo Alto, California USA (2007)
7. De Raedt, L., Van Laer, W.: Inductive constraint logic. In: Proceedings of the 6th Conference on Algorithmic Learning Theory (ALT 1995). LNAI, vol. 997, pp. 80–94. Springer, Fukuoka, Japan (1995)
8. De Raedt, L., Dzeroski, S.: First-order jk-clausal theories are pac-learnable. Artif. Intell. 70(1-2), 375–392 (1994)
9. Domingos, P., Webb, W.A.: A tractable first-order probabilistic logic. In: Hoffmann, J., Selman, B. (eds.) 26th National Conference on Artificial Intelligence, AAAI'12, Toronto, Ontario, Canada. AAAI Press (2012)
10. Fierens, D., den Broeck, G.V., Renkens, J., Shterionov, D.S., Gutmann, B., Thon, I., Janssens, G., De Raedt, L.: Inference and learning in probabilistic logic programs using weighted boolean formulas. Theor. Pract. Log. Prog. 15(3), 358–401 (2015)
11. Gutmann, B., Thon, I., De Raedt, L.: Learning the parameters of probabilistic logic programs from interpretations. In: Gunopulos, D., Hofmann, T., Malerba, D., Vazirgiannis, M. (eds.) European Conference on Machine Learning and Knowledge Discovery in Databases. LNCS, vol. 6911, pp. 581–596. Springer (2011)
12. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: 18th International Conference on Machine Learning. vol. 1, pp. 282–289 (2001)

13. Lamma, E., Mello, P., Riguzzi, F., Storari, S.: Applying inductive logic programming to process mining. In: Proceedings of the 17th International Conference on Inductive Logic Programming, ILP 2007. pp. 132–146. No. 4894 in Lecture Notes in Artificial Intelligence, Springer, Heidelberg, Germany (2008), http://dx.doi.org/10.1007/978-3-540-78469-2_16
14. Ly, L.T., Maggi, F.M., Montali, M., Rinderle-Ma, S., van der Aalst, W.M.: Compliance monitoring in business processes: Functionalities, application, and tool-support. Inform. Syst. 54, 209 – 234 (2015)
15. Montali, M.: Specification and Verification of Declarative Open Interaction Models: a Logic-Based Approach, LNBIP, vol. 56. Springer (2010)
16. Niepert, M., den Broeck, G.V.: Tractability through exchangeability: A new perspective on efficient probabilistic inference. In: Brodley, C.E., Stone, P. (eds.) 28th National Conference on Artificial Intelligence, AAAI'14, Québec City, Québec, Canada. pp. 2467–2475. AAAI Press (2014)
17. Niepert, M., Domingos, P.: Tractable probabilistic knowledge bases: Wikipedia and beyond. In: AAAI-14 Workshop on Statistical Relational Artificial Intelligence. AAAI Workshops, vol. WS-14-13. AAAI Press (2014)
18. Nocedal, J.: Updating quasi-newton matrices with limited storage. Mathematics of Computation 35(151), 773–782 (1980)
19. Poole, D., Zhang, N.L.: Exploiting contextual independence in probabilistic inference. J. Artif. Intell. Res. 18, 263–313 (2003)
20. Richardson, M., Domingos, P.: Markov logic networks. Mach. Learn. 62(1-2), 107–136 (2006)
21. Riguzzi, F., Swift, T.: The PITA system: Tabling and answer subsumption for reasoning under uncertainty. Theor. Pract. Log. Prog. 11(4–5), 433–449 (2011)
22. Riguzzi, F., Swift, T.: Well-definedness and efficient inference for probabilistic logic programming under the distribution semantics. Theor. Pract. Log. Prog. 13(Special Issue 02 - 25th Annual GULP Conference), 279–302 (2013)
23. Sato, T.: A statistical learning method for logic programs with distribution semantics. In: Sterling, L. (ed.) 12th International Conference on Logic Programming, Tokyo, Japan. pp. 715–729. MIT Press, Cambridge, Massachusetts (1995)
24. Van den Broeck, G.: On the completeness of first-order knowledge compilation for lifted probabilistic inference. In: Shawe-Taylor, J., Zemel, R.S., Bartlett, P.L., Pereira, F.C.N., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems. pp. 1386–1394 (2011)
25. Webb, W.A., Domingos, P.: Tractable probabilistic knowledge bases with existence uncertainty. In: AAAI-13 Workshop on Statistical Relational Artificial Intelligence. AAAI Workshops, vol. WS-13-16. AAAI Press (2013)