The Distribution Semantics for Normal Programs with Function Symbols

Fabrizio Riguzzi*

Dipartimento di Matematica e Informatica, Università di Ferrara
Via Saragat 1, I-44122, Ferrara, Italy

Abstract

The distribution semantics integrates logic programming and probability theory using a possible worlds approach. Its intuitiveness and simplicity has made it the most widely used semantics for probabilistic logic programming, with successful applications in many domains. When the program has function symbols, the semantics was defined for special cases: either the program has to be definite or the queries must have a finite number of finite explanations. In this paper we show that it is possible to define the semantics for all programs. We also show that this definition coincides with that of Sato and Kameya on positive programs. Moreover, we highlight possible approaches for inference, both exact and approximate.

Keywords: Distribution Semantics, Function Symbols, ProbLog, Probabilistic Logic Programming

1. Introduction

The distribution semantics has been proposed independently in [1, 2, 3] and has been rediscovered many times by various authors. It is a particularly appealing approach for assigning a semantics for probabilistic logic programs (PLP) because of its clear reference to possible worlds that make programs readable and intuitive.

In the last few years many languages have been proposed that are based on the distribution semantics, such as Probabilistic Horn Abduction [2], PRISM [3], Independent Choice Logic [4], Logic Programs with Annotated Disjunctions [5], ProbLog [6] and CP-logic [7]. These languages have been successfully applied in many domains such as natural language processing, biology, chemistry and medicine.

The definition of the distribution semantics can be given quite simply in the case of no function symbols in the program: a probabilistic logic program under

*Corresponding author
Email address: `fabrizio.riguzzi@unife.it` (Fabrizio Riguzzi)

the distribution semantics defines a probability distribution over normal logic programs called *worlds* and the probability of a ground query can be obtained by marginalizing the joint distribution of the worlds and the query. In the case where the program has function symbols, however, this simple definition does not work as the probability of individual worlds is zero.

A definition of the distribution semantics for programs with function symbols was proposed in [2, 8] but restricted to definite programs. The case of normal programs was taken into account in [4] where the semantics required that the programs are acyclic. A looser condition was proposed in [9] but still required each goal to have a finite set of finite explanations.

In this paper we show that the distribution semantics can be defined for all programs, thus also for programs that have goals with an infinite number of possibly infinite explanations. We do so by adapting the definition of the well-founded semantics in terms of iterated fixpoints of [10] to the case of ProbLog, similarly to the way in which the T_P operator has been adapted in [11, 12] to the case of stratified ProbLog programs using parameterized interpretations. In the case of an infinite number of possibly infinite explanations, we show that the probability of queries is defined in the limit and the limit always exists.

We consider the case of ProbLog but the results are equally applicable to all other languages under the distribution semantics, as there are linear transformations from one language to another that preserve the semantics.

The possibility of having goals with an infinite number of explanations is useful for modeling domains with recursive definitions or chains such as those that are found in model checking or grammar parsing. Some examples of these domains are illustrated in Section 7.

Moreover, we discuss possible approaches for performing inference, i.e., computing the probability of goals having an infinite number of explanations. Exact inference algorithms have been presented in [13, 14, 15] but they impose limitations to the form of programs. Approximate inference based on Monte Carlo sampling can be applied to a larger class of programs directly, as the probability of following an infinite explanation is null. The web system cplint on SWISH [16][1] contains many example programs that can be run online with Monte Carlo inference.

The paper is organized as follows. Section 2 presents preliminary material on fixpoints and the well-founded semantics. Section 3 introduces the distribution semantics for programs without function symbols. Section 4 discusses the definition of the distribution semantics with function symbols in the case of finite set of finite explanations. Section 5 represents the main contribution of this paper and discusses the case of infinite set of infinite explanations. Section 6 illustrates the relationship with the definition of [8]. Section 7 shows some examples where the definition applies and Section 8 discusses approaches for inference. Finally, Section 9 concludes the paper.

[1] `http://cplint.lamping.unife.it/`

2. Preliminaries

2.1. Partial Orders, Complete Lattices, Fixpoints

A relation on a set S is a *partial order* if it is reflexive, antisymmetric and transitive. In the following, let S be a set with a partial order $\leqslant$. $a \in S$ is an *upper bound* of a subset X of S if $x \leqslant a$ for all $x \in X$. Similarly, $b \in S$ is a *lower bound* of X if $b \leqslant x$ for all $x \in X$.

An element $a \in S$ is the *least upper bound* of a subset X of X if a is an upper bound of X and, for all upper bounds a' of X, we have $a \leqslant a'$. Similarly, $b \in S$ is the *greatest lower bound* of a subset X of S if b is a lower bound of X and, for all lower bounds b' of X, we have $b' \leqslant b$. The least upper bound of X is unique, if it exists, and is denoted by $lub(X)$. Similarly, the greatest lower bound of X is unique, if it exists, and is denoted by $glb(X)$.

A partially ordered set L is a *complete lattice* if $lub(X)$ and $glb(X)$ exist for every subset X of L. We let $\top$ denote the top element $lub(L)$ and $\bot$ denote the bottom element $glb(L)$ of the complete lattice L.

Let L be a complete lattice and $T : L \to L$ be a mapping. We say T is *monotonic* if $T(x) \leqslant T(y)$, whenever $x \leqslant y$. We say that $a \in L$ is a *fixpoint* of T if $T(a) = a$. We say that $a \in L$ is the *least fixpoint* of T if a is a fixpoint and, for all fixpoints b of T, we have $a \leqslant b$. Similarly, we define *greatest fixpoint*.

Let L be a complete lattice and $T : L \to L$ be monotonic. Then we define $T \uparrow 0 = \bot$; $T \uparrow \alpha = T(T \uparrow (\alpha - 1))$, if α is a successor ordinal; $T \uparrow \alpha = lub(\{T \uparrow \beta | \beta < \alpha\})$, if α is a limit ordinal; $T \downarrow 0 = \top$; $T \downarrow \alpha = T(T \downarrow (\alpha - 1))$, if α is a successor ordinal; $T \downarrow \alpha = glb(\{T \downarrow \beta | \beta < \alpha\})$, if α is a limit ordinal.

Proposition 1. *Let L be a complete lattice and $T : L \to L$ be monotonic. Then T has a lest fixpoint, $lfp(T)$ and a greatest fixpoint $gfp(T)$.*

2.2. Logic Programming

A normal program P is a set of normal rules. A normal rule has the form

$$r = h \leftarrow b_1, \ldots, b_n, \neg c_1, \ldots, \neg c_m \tag{1}$$

where $h, b_1, \ldots, b_n, c_1, \ldots, c_m$ are atoms.

The set of ground atoms that can be built with the symbols of a program P is called the *Herbrand base* and is denoted as $\mathcal{B}_P$.

A *two-valued interpretation* I is a subset of $\mathcal{B}_P$. I is the set of true atoms, so a is true in I if $a \in I$ and is false if $a \notin I$. The set $Int2$ of two-valued interpretations for a program P forms a complete lattice where the partial order $\leqslant$ is given by the subset relation $\subseteq$. The least upper bound and greatest lower bound are defined as $lub(X) = \bigcup_{I \in X} I$ and $glb(X) = \bigcap_{I \in X} I$. The bottom and top element are respectively $\varnothing$ and $\mathcal{B}_P$.

A *three-valued interpretation* $\mathcal{I}$ is a pair $\langle I_T; I_F \rangle$ where I_T and I_F are subsets of $\mathcal{B}_P$ and represent respectively the set of true and false atoms. So a is true in $\mathcal{I}$ if $a \in I_T$ and is false in $\mathcal{I}$ if $a \in I_F$. A *consistent* three-valued interpretation $\mathcal{I} = \langle I_T; I_F \rangle$ is such that $I_T \cap I_F = \varnothing$. The union of two three-valued interpretations

$\langle I_T, I_F\rangle$ and $\langle J_T, J_F\rangle$ is defined as $\langle I_T, I_F\rangle \cup \langle J_T, J_F\rangle = \langle I_T \cup J_T, I_F \cup J_F\rangle$. The intersection of two three-valued interpretations $\langle I_T, I_F\rangle$ and $\langle J_T, J_F\rangle$ is defined as $\langle I_T, I_F\rangle \cap \langle J_T, J_F\rangle = \langle I_T \cap J_T, I_F \cap J_F\rangle$.

The set $Int3$ of three-valued interpretations for a program P forms a complete lattice where the partial order $\leqslant$ is defined as $\langle I_T, I_F\rangle \leqslant \langle J_T, J_F\rangle$ if $I_T \subseteq J_T$ and $I_F \subseteq J_F$. The least upper bound and greatest lower bound are defined as $lub(X) = \bigcup_{I\in X} I$ and $glb(X) = \bigcap_{I\in X} I$. The bottom and top element are respectively $\langle\varnothing,\varnothing\rangle$ and $\langle\mathcal{B}_P,\mathcal{B}_P\rangle$.

The well-founded semantics (WFS) assigns a three-valued model to a program, i.e., it identifies a consistent three-valued interpretation as the meaning of the program. The WFS was given in [17] in terms of the least fixpoint of an operator that is composed by two sub-operators, one computing consequences and the other computing unfounded sets. We give here the alternative definition of the WFS of [10] that is based on a different iterated fixpoint.

Definition 1. For a normal program P, sets Tr and Fa of ground atoms, and a 3-valued interpretation $\mathcal{I}$ we define the operators $OpTrue^P_{\mathcal{I}} : Int2 \rightarrow Int2$ and $OpFalse^P_{\mathcal{I}} : Int2 \rightarrow Int2$ as

$OpTrue^P_{\mathcal{I}}(Tr) = \{a|a$ is not true in $\mathcal{I}$; and there is a clause $b \leftarrow l_1, ..., l_n$ in P, a grounding substitution θ such that $a = b\theta$ and for every $1 \leqslant i \leqslant n$ either $l_i\theta$ is true in $\mathcal{I}$, or $l_i\theta \in Tr\}$;

$OpFalse^P_{\mathcal{I}}(Fa) = \{a|a$ is not false in $\mathcal{I}$; and for every clause $b \leftarrow l_1, ..., l_n$ in P and grounding substitution θ such that $a = b\theta$ there is some i $(1 \leqslant i \leqslant n)$ such that $l_i\theta$ is false in $\mathcal{I}$ or $l_i\theta \in Fa\}$.

In words, the operator $OpTrue^P_{\mathcal{I}}(Tr)$ extends the interpretation $\mathcal{I}$ to add the new true atoms that can be derived from P knowing $\mathcal{I}$ and true atoms Tr, while $OpFalse^P_{\mathcal{I}}(Fa)$ computes new false atoms in P by knowing $\mathcal{I}$ and false atoms Fa. $OpTrue^P_{\mathcal{I}}$ and $OpFalse^P_{\mathcal{I}}$ are both monotonic [10], so they both have least and greatest fixpoints. An iterated fixpoint operator builds up *dynamic strata* by constructing successive three-valued interpretations as follows.

Definition 2 (Iterated Fixed Point). For a normal program P, let $IFP^P : Int3 \rightarrow Int3$ be defined as $IFP^P(\mathcal{I}) = \mathcal{I} \cup \langle lfp(OpTrue^P_{\mathcal{I}}), gfp(OpFalse^P_{\mathcal{I}})\rangle$.

IFP^P is monotonic [10] and thus as a least fixed point $lfp(IFP^P)$. Moreover, the well-founded model $WFM(P)$ of P is in fact $lfp(IFP^P)$. Let δ be the smallest ordinal such that $WFM(P) = IFP^P \uparrow \delta$. We refer to δ as the *depth* of P. The *stratum* of atom a is the least ordinal β such that $a \in IFP^P \uparrow \beta$ (where a may be either in the true or false component of $IFP^P \uparrow \beta$). Undefined atoms of the well-founded model do not belong to any stratum – i.e. they are not added to $IFP^P \uparrow \delta$ for any ordinal δ.

3. The Distribution Semantics for Programs without Function Symbols

We present the distribution semantics for the case of ProbLog [6] as it is the language with the simplest syntax. A ProbLog program $\mathcal{P}$ is composed by a set of normal rules $\mathcal{R}$ and a set $\mathcal{F}$ of probabilistic facts. Each *probabilistic fact* is of the form $p_i :: f_i$ where $p_i \in [0,1]$ and f_i is an atom[2], meaning that each ground instantiation $f_i\theta$ of f_i is true with probability p_i and false with probability $1 - p_i$. Each world is obtained by selecting or rejecting each grounding of all probabilistic facts.

An *atomic choice* indicates whether grounding $f\theta$ of a probabilistic fact $F = p :: f$ is selected or not. It is represented with the triple (f, θ, i) where $i \in \{0, 1\}$. A set κ of atomic choices is *consistent* if it does not contain two atomic choices (f, θ, i) and (f, θ, j) with $i \neq j$ (only one alternative is selected for a ground probabilistic fact). The function *consistent*(κ) returns true if κ is consistent. A *composite choice* κ is a consistent set of atomic choices. The probability of composite choice κ is $P(\kappa) = \prod_{(f_i,\theta,1)\in\kappa} p_i \prod_{(f_i,\theta,0)\in\kappa} 1-p_i$ where p_i is the probability of the i-th probabilistic fact F_i. A *selection* σ is a total composite choice, i.e., it contains one atomic choice for every grounding of every probabilistic fact. A *world* w_σ is a logic program that is identified by a selection σ. The world w_σ is formed by including the atom corresponding to each atomic choice $(f, \theta, 1)$ of σ.

The probability of a world w_σ is $P(w_\sigma) = P(\sigma)$. Since in this section we are assuming programs without function symbols, the set of groundings of each probabilistic fact is finite, and so is the set of worlds $W_\mathcal{P}$. Accordingly, for a ProbLog program $\mathcal{P}$, $W_\mathcal{P} = \{w_1, \ldots, w_m\}$. Moreover, $P(w)$ is a distribution over worlds: $\sum_{w\in W_\mathcal{P}} P(w) = 1$. We call *sound* a program for which every world has a two-valued well-founded model. We consider only sound programs, as the uncertainty should be handled by the choices rather than by the semantics of negation.

Let q be a query in the form of a ground atom. We define the conditional probability of q given a world w as: $P(q|w) = 1$ if q is true in w and 0 otherwise. Since the program is sound, q can be only true or false in a world. The probability of q can thus be computed by summing out the worlds from the joint distribution of the query and the worlds: $P(q) = \sum_w P(q, w) = \sum_w P(q|w)P(w) = \sum_{w\models q} P(w)$.

4. The Distribution Semantics for Programs with Function Symbols

When a program contains functions symbols there is the possibility that its grounding may be infinite. If so, the number of atomic choices in a selection that defines a world is countably infinite and there is an uncountably infinite

[2]With an abuse of notation, sometimes we use $\mathcal{F}$ to indicate the set containing the atoms f_is. The meaning of $\mathcal{F}$ will be clear from the context.

number of worlds. The probability of each individual world is given by an infinite product. We recall the following result from [18, page 218].

Lemma 1. *If $p_i \in [0, b]$ for all $i = 1, 2, \dots$ with $b \in [0, 1)$, then the infinite product $\prod_{i=1}^{\infty} p_i$ converges to 0.*

Each factor in the infinite product giving the probability of a world is bounded away from one, i.e., it belongs to$[0, b]$ for a $b \in [0, 1)$. To see this it is enough to pick b as the maximum of all the probabilistic parameters that appear in the program. This is possible if the program does not have *flexible probabilities* [19] or probabilities that depend on values computing during program execution.

So if the program does not contain flexible probabilities, the probability of each individual world is zero and the semantics of Section 3 is not well-defined.

Example 1. *Consider the program*

$$\begin{array}{lll} p(0) \leftarrow u(0). & t \leftarrow \neg s. & F_1 = a :: u(X). \\ p(s(X)) \leftarrow p(X), u(X). & s \leftarrow r, \neg q. & F_2 = b :: r. \\ & q \leftarrow u(X). & \end{array}$$

The set of worlds is infinite and uncountable. In fact, each world can be put in a one to one relation with a selection and a selection can be represented as a countable sequence of atomic choices of which the first involves fact f_2, the second $f_1/\{X/0\}$, the third $f_1/\{X/s(0)\}$ and so on. The set of selections can be shown uncountable by Cantor's diagonal argument. Suppose the set of selections is countable. Then the selections could be listed in order, suppose from top to bottom. Suppose the atomic choices of each selection are listed from left to right. We can pick a composite choice that differs from the first selection in the first atomic choice (if $(f_2, \emptyset, k)$ is the first atomic choice of the first selection, pick $(f_2, \emptyset, 1-k)$), from the second selection in the second atomic choice (similar to the case of the first atomic choice) and so on. In this way we have obtained a selection that is not present in the list because it differs from each selection in the list for at least an atomic choice. So it is not possible to list the selections in order.

Example 2. *Consider the game of dice proposed in [5]: the player repeatedly throws a six-sided die. When the outcome is six, the game stops. A version of this game where the die has three sides is:*

$$\begin{array}{l} F_1 = 1/3 :: one(X). \\ F_2 = 1/2 :: two(X). \\ on(0,1) \leftarrow one(0). \\ on(0,2) \leftarrow \neg one(0), two(0). \\ on(0,3) \leftarrow \neg one(0), \neg two(0). \\ on(s(X),1) \leftarrow on(X,\_), \neg on(X,3), one(s(X)). \\ on(s(X),2) \leftarrow on(X,\_), \neg on(X,3), \neg one(s(X)), two(s(X)). \\ on(s(X),3) \leftarrow on(X,\_), \neg on(X,3), \neg one(s(X)), \neg two(s(X)). \end{array}$$

If we add the clauses

$$\begin{array}{l} at\_least\_once\_1 \leftarrow on(\_, 1). \\ never\_1 \leftarrow \neg at\_least\_once\_1. \end{array}$$

we can ask for the probability that at least once the die landed on face 1 and that the die never landed on face 1. As in Example 1, this program has an infinite and uncountable set of worlds.

We now present the definition of the distribution semantics for programs with function symbols following [4]. The semantics for a probabilistic logic program $\mathcal{P}$ with function symbols is given by defining a *probability measure* μ over the set of worlds $W_{\mathcal{P}}$. Informally, μ assigns a probability to a set of *subsets* of $W_{\mathcal{P}}$, rather than to every element of (the infinite set) $W_{\mathcal{P}}$. The approach dates back to [20] who defined a probability measure μ as a real-valued function whose domain is a σ*-algebra* Ω on a set $\mathcal{W}$ called the *sample space.* Together $\langle \mathcal{W}, \Omega, \mu \rangle$ is called a *probability space.*

Definition 3. [21, Section 3.1] The set Ω of subsets of $\mathcal{W}$ is a σ*-algebra* on the set $\mathcal{W}$ iff (σ-1) $\mathcal{W} \in \Omega$; (σ-2) Ω is closed under complementation, i.e., $\omega \in \Omega \rightarrow (\mathcal{W} \backslash \omega) \in \Omega$; and ($\sigma$-3) Ω is closed under countable union, i.e., if $\omega_i \in \Omega$ for $i = 1, 2, \ldots$ then $\bigcup_i \omega_i \in \Omega$.

The elements of Ω are called *measurable sets* and $(\mathcal{W}, \Omega)$ is called a *measurable space.*

Importantly, for defining the distribution semantics for programs with function symbols, not every subset of $\mathcal{W}$ need be present in Ω.

Definition 4. [20] Given a sample space $\mathcal{W}$ and a σ-algebra Ω of subsets of $\mathcal{W}$, a *probability measure* is a function $\mu : \Omega \rightarrow \mathbb{R}$ that satisfies the following axioms: (μ-1) $\mu(\omega) \geqslant 0$ for all $\omega \in \Omega$; (μ-2) $\mu(\mathcal{W}) = 1$; (μ-3) μ is countably additive, i.e., if $O = \{\omega_1, \omega_2, \ldots\} \subseteq \Omega$ is a countable collection of pairwise disjoint sets, then $\mu(\bigcup_{\omega \in O}) = \sum_i \mu(\omega_i)$.

We first consider the finite additivity version of probability spaces. In this stronger version, the σ-algebra is replaced by an algebra.

Definition 5. [21, Section 3.1] The set Ω of subsets of $\mathcal{W}$ is an *algebra* on the set $\mathcal{W}$ iff it respects conditions (σ-1), (σ-2) and condition (a-3): Ω is closed under finite union, i.e., $\omega_1 \in \Omega, \omega_2 \in \Omega \rightarrow (\omega_1 \cup \omega_2) \in \Omega$

The probability measure is replaced by a finitely additive probability measure.

Definition 6. Given a sample space $\mathcal{W}$ and an algebra Ω of subsets of $\mathcal{W}$, a *finitely additive probability measure* is a function $\mu : \Omega \rightarrow \mathbb{R}$ that satisfies axioms (μ-1) and (μ-2) of Definition 4 and axiom (m-3): μ is finitely additive, i.e., $\omega_1 \cap \omega_2 = \varnothing \rightarrow \mu(\omega_1 \cup \omega_2) = \mu(\omega_1) + \mu(\omega_2)$ for all $\omega_1, \omega_2 \in \Omega$.

Towards defining a suitable algebra given a probabilistic logic program $\mathcal{P}$, we define the *set of worlds* ω_κ *compatible with* a composite choice κ as $\omega_\kappa = \{w_\sigma \in W_\mathcal{P} | \kappa \subseteq \sigma\}$. Thus a composite choice identifies a set of worlds. For programs without function symbols $P(\kappa) = \sum_{w \in \omega_\kappa} P(w)$.

Given a *set* of composite choices K, the *set of worlds* ω_K *compatible with* K is $\omega_K = \bigcup_{\kappa \in K} \omega_\kappa$. Two composite choices κ_1 and κ_2 are *incompatible* if their union is not consistent. A set K of composite choices is *pairwise incompatible* if for all $\kappa_1 \in K, \kappa_2 \in K$, $\kappa_1 \neq \kappa_2$ implies that κ_1 and κ_2 are incompatible.

Regardless of whether a probabilistic logic program has a finite number of worlds or not, obtaining pairwise incompatible sets of composite choices is an important problem. This is because the *probability of a pairwise incompatible set* K *of composite choices* is defined as $P(K) = \sum_{\kappa \in K} P(\kappa)$ which is easily computed. Two sets K_1 and K_2 of finite composite choices are *equivalent* if they correspond to the same set of worlds: $\omega_{K_1} = \omega_{K_2}$.

One way to assign probabilities to a set K of composite choices is to construct an equivalent set that is pairwise incompatible; such a set can be constructed through the technique of *splitting*. More specifically, if $f\theta$ is an instantiated fact and κ is a composite choice that does not contain an atomic choice (f, θ, k) for any k, the *split* of κ on $f\theta$ is the set of composite choices $S_{\kappa,f\theta} = \{\kappa \cup \{(f, \theta, 0)\}, \kappa \cup \{(f, \theta, 1)\}\}$. It is easy to see that κ and $S_{\kappa,f\theta}$ identify the same set of possible worlds, i.e., that $\omega_\kappa = \omega_{S_{\kappa,f\theta}}$, and that $S_{\kappa,f\theta}$ is pairwise incompatible. The technique of splitting composite choices on formulas is used for the following result [22].

Theorem 1 (Existence of a pairwise incompatible set of composite choices [22])**.** *Given a finite set K of composite choices, there exists a finite set K' of pairwise incompatible composite choices such that K and K' are equivalent.*

Proof. Given a finite set of composite choices K, there are two possibilities to form a new set K' of composite choices so that K and K' are equivalent:

1. **removing dominated elements**: if $\kappa_1, \kappa_2 \in K$ and $\kappa_1 \subset \kappa_2$, let $K' = K \backslash \{\kappa_2\}$.
2. **splitting elements**: if $\kappa_1, \kappa_2 \in K$ are compatible (and neither is a superset of the other), there is a $(f, \theta, k) \in \kappa_1 \backslash \kappa_2$. We replace κ_2 by the split of κ_2 on $f\theta$. Let $K' = K \backslash \{\kappa_2\} \cup S_{\kappa_2,f\theta}$.

In both cases $\omega_K = \omega_{K'}$. If we repeat this two operations until neither is applicable we obtain a splitting algorithm that terminates because K is a finite set of composite choices. The resulting set K' is pairwise incompatible and is equivalent to the original set. □

Theorem 2 (Equivalence of the probability of two equivalent pairwise incompatible finite set of finite composite choices [23])**.** *If K_1 and K_2 are both pairwise incompatible finite sets of finite composite choices such that they are equivalent then $P(K_1) = P(K_2)$.*

Proof. Consider the set D of all instantiated facts $f\theta$ that appear in an atomic choice in either K_1 or K_2. This set is finite. Each composite choice in K_1 and

K_2 has atomic choices for a subset of D. For both K_1 and K_2, we repeatedly replace each composite choice κ of K_1 and K_2 with its split $S_{\kappa,f_i\theta_j}$ on an $f_i\theta_j$ from D that does not appear in κ. This procedure does not change the total probability as the probabilities of $(f_i, \theta_j, 0)$ and $(f_i, \theta_j, 1)$ sum to 1.

At the end of this procedure the two sets of composite choices will be identical. In fact, any difference can be extended into a possible world belonging to ω_{K_1} but not to ω_{K_2} or vice versa. □

For a probabilistic logic program $\mathcal{P}$, we can thus define a unique probability measure $\mu : \Omega_\mathcal{P} \to [0, 1]$ where $\Omega_\mathcal{P}$ is defined as the set of sets of worlds identified by finite sets of finite composite choices: $\Omega_\mathcal{P} = \{\omega_K | K$ is a finite set of finite composite choices $\}$.

Theorem 3. *$\Omega_\mathcal{P}$ is an algebra over $W_\mathcal{P}$.*

Proof. $W_\mathcal{P} = \omega_K$ with $K = \{\varnothing\}$. The complement ω_K^c of ω_K where K is a finite set of finite composite choice is $\omega_{\overline{K}}$ where $\overline{K}$ is a finite set of finite composite choices. In fact, $\overline{K}$ can obtained with the function $duals(K)$ of [22] that performs Reiter's hitting set algorithm over K, generating an element κ of $\overline{K}$ by picking an atomic choice (f, θ, k) from each element of K and inserting in κ the atomic choice $(f, \theta, 1 - k)$. After this process is performed in all possible ways, inconsistent sets of atom choices are removed obtaining $\overline{K}$. Since the possible choices of the atomic choices are finite, so is $\overline{K}$. Finally, condition (a-3) holds since the union of ω_{K_1} with ω_{K_2} is equal to $\omega_{K_1 \cup K_2}$ for the definition of ω_K. □

The corresponding measure μ is defined by $\mu(\omega_K) = P(K')$ where K' is a pairwise incompatible set of composite choices equivalent to K.

Theorem 4. *$\langle W_\mathcal{P}, \Omega_\mathcal{P}, \mu\rangle$ is a finitely additive probability space according to Definition 6.*

Proof. $\mu(\omega_{\{\varnothing\}})$ is equal to 1. Moreover, $\mu(\omega_K) \geqslant 0$ for all K and if $\omega_{K_1} \cap \omega_{K_2} = \varnothing$ and K'_1 (K'_2) is pairwise incompatible and equivalent to K_1 (K_2), then $K'_1 \cup K'_2$ is pairwise incompatible and

$$\mu(\omega_{K_1} \cup \omega_{K_2}) = \sum_{\kappa \in K'_1 \cup K'_2} P(\kappa) = \sum_{\kappa_1 \in K'_1} P(\kappa_1) + \sum_{\kappa_2 \in K'_2} P(\kappa_2) = \mu(\omega_{K_1}) + \mu(\omega_{K_2}).$$

□

Given a query q, a composite choice κ is an *explanation* for q if $\forall w \in \omega_\kappa : w \models q$. A set K of composite choices is *covering* with respect to q if every world in which q is true belongs to ω_K

Definition 7. For a probabilistic logic program $\mathcal{P}$, the probability of a ground atom q is given by $P(q) = \mu(\{w | w \in W_\mathcal{P}, w \models q\})$.

If q has a finite set K of finite explanations such that K is covering then $\{w|w \in W_{\mathcal{P}} \wedge w \models q\} = \omega_K \in \Omega_T$ and we say that $P(q)$ is *finitely well-defined* for the distribution semantics. A program $\mathcal{P}$ is *finitely well-defined* if the probability of all ground atoms in the grounding of $\mathcal{P}$ is finitely well-defined.

Example 3. *Consider the program of Example 1. The set* $K = \{\kappa\}$ *with* $\kappa = \{(f_1, \{X/0\}, 1), (f_1, \{X/s(0)\}, 1)\}$ *is a pairwise incompatible finite set of finite explanations that are covering for the query* $p(s(0))$*. Definition 7 therefore applies, and* $P(p(s(0))) = P(\kappa) = a^2$

Example 4. *Now consider Example 2. The set* $K = \{\kappa_1, \kappa_2\}$ *with*

$$\begin{aligned} \kappa_1 &= \{(f_1, \{X/0\}, 1), (f_1, \{X/s(0)\}, 1)\} \\ \kappa_2 &= \{(f_1, \{X/0\}, 0), (f_2, \{X/0\}, 1), (f_1, \{X/s(0)\}, 1)\} \end{aligned}$$

is a pairwise incompatible finite set of finite explanations that are covering for the query $on(s(0), 1)$*. According to Definition 7 then*

$$P(on(s(0), 1)) = P(K) = 1/3 \times 1/3 + 2/3 \times 1/2 \times 1/3 = 2/9.$$

5. Infinite Covering Set of Explanations

In this section we go beyond [4] and we remove the requirement of the finiteness of the covering set of explanations and of each explanation for a query q.

Example 5. *In Example 1, the query* s *has the pairwise incompatible covering set of explanations*

$$K^s = \{\kappa_0^s, \kappa_1^s, \ldots\}$$

with

$$\kappa_i^s = \{(f_2, \varnothing, 1), (f_1, \{X/0\}, 1), \ldots, (f_1, \{X/s^{i-1}(0)\}, 1), (f_1, \{X/s^i(0)\}, 0)\}$$

where $s^i(0)$ *is the term where the functor* s *is applied* i *times to 0. So* K^s *is countable and infinite. A covering set of explanations for* t *is*

$$K^t = \{\{(f_2, \varnothing, 0)\}, \kappa^t\}$$

where κ^t *is the infinite composite choice*

$$\kappa^t = \{(f_2, \varnothing, 1), (f_1, \{X/0\}, 1), (f_1, \{X/s(0)\}, 1), \ldots\}$$

Example 6. *In Example 2, the query* $at\_least\_once\_1$ *has the pairwise incompatible covering set of explanations*

$$K^+ = \{\kappa_0^+, \kappa_1^+, \ldots\}$$

with

$$\begin{array}{rcl}
\kappa_0^+ & = & \{(f_1, \{X/0\}, 1)\} \\
\kappa_1^+ & = & \{(f_1, \{X/0\}, 0), (f_2, \{X/0\}, 1), (f_1, \{X/s(0)\}, 1)\} \\
\dots & & \\
\kappa_i^+ & = & \{(f_1, \{X/0\}, 0), (f_2, \{X/0\}, 1), \dots, (f_1, \{X/s^{i-1}(0)\}, 0), \\
 & & (f_2, \{X/s^{i-1}(0)\}, 1), (f_1, \{X/s^i(0)\}, 1)\} \\
\dots & &
\end{array}$$

So K^+ is countable and infinite. The query never_1 has the pairwise incompatible covering set of explanations

$$K^- = \{\kappa_0^-, \kappa_1^-, \dots\}$$

with

$$\begin{array}{rcl}
\kappa_0^- & = & \{(f_1, \{X/0\}, 0), (f_2, \{X/0\}, 0)\} \\
\kappa_1^- & = & \{(f_1, \{X/0\}, 0), (f_2, \{X/0\}, 1), (f_1, \{X/s(0)\}, 0), \\
 & & (f_2, \{X/s(0)\}, 0)\} \\
\dots & & \\
\kappa_i^- & = & \{(f_1, \{X/0\}, 0), (f_2, \{X/0\}, 1), \dots, (f_1, \{X/s^{i-1}(0)\}, 0), \\
 & & (f_2, \{X/s^{i-1}(0)\}, 1), (f_1, \{X/s^i(0)\}, 0), (f_2, \{X/s^i(0)\}, 0)\} \\
\dots & &
\end{array}$$

For a probabilistic logic program $\mathcal{P}$, we can define the probability measure $\mu : \Omega_{\mathcal{P}} \to [0, 1]$ where $\Omega_{\mathcal{P}}$ is defined as the set of sets of worlds identified by countable sets of countable composite choices: $\Omega_{\mathcal{P}} = \{\omega_K | K$ is a countable set of countable composite choices $\}$.

Before showing that $\Omega_{\mathcal{P}}$ is a σ-algebra, we need some definitions and results regarding sequences of sets. For any sequence of sets $\{A_n | n \geqslant 1\}$ define [24, page 2]

$$\begin{array}{rcl}
\underline{\lim}_{n\to\infty} A_n & = & \bigcup_{n=1}^{\infty} \bigcap_{k=n}^{\infty} A_n \\
\overline{\lim}_{n\to\infty} A_n & = & \bigcap_{n=1}^{\infty} \bigcup_{k=n}^{\infty} A_n
\end{array}$$

Note that [24, page 2]

$$\begin{array}{rcl}
\overline{\lim}_{n\to\infty} A_n & = & \{a | a \in A_n \text{ i.o.}\} \\
\underline{\lim}_{n\to\infty} A_n & = & \{a | a \in A_n \text{ for all but a finite number of indices } n\}
\end{array}$$

where i.o. denotes infinitely often. The two definitions differ because an element a of $\overline{\lim}_{n\to\infty} A_n$ may be absent from A_n for an infinite number of indices n, provided that there is a disjoint, infinite set of indices n for which $a \in A_n$. For each $a \in \underline{\lim}_{n\to\infty} A_n$, instead, there is a $m \geqslant 1$ such that $\forall n \geqslant m, a \in A_n$.

Then $\underline{\lim}_{n\to\infty} A_n \subseteq \overline{\lim}_{n\to\infty} A_n$. If $\underline{\lim}_{n\to\infty} A_n = \overline{\lim}_{n\to\infty} A_n = A$, then A is called the *limit of the sequence* and we write $A = \lim_{n\to\infty} A_n$.

A sequence $\{A_n | n \geqslant 1\}$ is *increasing* if $A_{n-1} \subseteq A_n$ for all $n = 2, 3, \ldots$. If a sequence $\{A_n | n \geqslant 1\}$ is increasing, the limit $\lim_{n\to\infty} A_n$ exists and is equal to $\bigcup_{n=1}^{\infty} A_n$ [24, page 3].

Lemma 2. *$\Omega_{\mathcal{P}}$ is a σ-algebra over $W_{\mathcal{P}}$.*

Proof. (σ-1) is true as in the algebra case. To see that the complement ω_K^c of ω_K is in $\Omega_{\mathcal{P}}$, let us prove that the dual $\overline{K}$ of K is a countable set of countable composite choices and then that $\omega_K^c = \omega_{\overline{K}}$. Let us consider first the case where K is finite, i.e., let K be $K_n = \{\kappa_1, \ldots, \kappa_n\}$. We will prove the thesis by induction. In the base case, if $K_1 = \{\kappa_1\}$, then we can obtain $\overline{K_1}$ by picking each atomic choice (f, θ, k) of κ_1 and inserting in $\overline{K_1}$ the composite choice $\{(f, \theta, 1-k)\}$. As there is a finite or countable number of atomic choices in κ_1, $\overline{K_1}$ is a finite or countable set of composite choices each with one atomic choice.

In the inductive case, assume that $K_{n-1} = \{\kappa_1, \ldots, \kappa_{n-1}\}$ and that $\overline{K_{n-1}}$ is a finite or countable set of composite choices. Let $K_n = K_{n-1} \cup \{\kappa_n\}$ and $\overline{K_{n-1}} = \{\kappa'_1, \kappa'_2, \ldots\}$. We can obtain $\overline{K_n}$ by picking each κ'_i and each atomic choice (f, θ, k) of κ_n. If $(f, \theta, k) \in \kappa'_i$, we discard κ'_i, else if $(f, \theta, k') \in \kappa'_i$ with $k' \neq k$, we insert κ'_i in $\overline{K_n}$. Otherwise we generate the composite choice κ''_i where $\kappa''_i = \kappa'_i \cup \{(f, \theta, 1-k)\}$ and insert it in $\overline{K_n}$. Doing this for all atomic choices (f, θ, k) in κ_n generates a finite set of finite composite choices if κ_n is finite and a countable number of finite composite choices if κ_n is countable. Doing this for all κ'_i we obtain that $\overline{K_n}$ is a countable union of countable sets which is a countable set [25, page 3]. $\omega_K^c = \omega_{\overline{K}}$ because all composite choices of $\overline{K}$ are incompatible with each world of ω_K, as they are incompatible with each composite choice of K. So $\omega_K^c \in \Omega_{\mathcal{P}}$.

If K is not finite, then let $K = \{\kappa_1, \kappa_2, \ldots\}$. Consider the subsets K_n of the form $K_n = \{\kappa_1, \ldots, \kappa_n\}$. Using the construction above build $\overline{K_n}$ for all n and consider the set $\underline{\lim}_{n\to\infty} \overline{K_n}$ and $\overline{\lim}_{n\to\infty} \overline{K_n}$. A κ' belongs to $\underline{\lim}_{n\to\infty} \overline{K_n}$ if there exists an integer $m \geqslant 1$ such that $\kappa' \in \overline{K_n}$ for all $n \geqslant m$. Consider a κ' that belongs to $\overline{\lim}_{n\to\infty} \overline{K_n}$. Suppose $\kappa' \in \overline{K_j}$ and $\kappa' \notin \overline{K_{j+1}}$. This means that κ' was removed because $\kappa' \subseteq \kappa_{j+1}$. Then κ' will never be re-added to a $\overline{K_n}$ with $n > j+1$ because otherwise ω_{K_n} and $\omega_{\overline{K_n}}$ would have a non-empty intersection. So for a composite choice to appear infinitely often there must exist an integer $m \geqslant 1$ such that $\kappa' \in \overline{K_n}$ for all $n \geqslant m$. In other words, κ' must belong to $\overline{\lim}_{n\to\infty} \overline{K_n}$ for all but a finite number of indices n. Therefore $\underline{\lim}_{n\to\infty} \overline{K_n} = \overline{\lim}_{n\to\infty} \overline{K_n} = \lim_{n\to\infty} \overline{K_n}$. Let us call $\overline{K}$ this limit.

$\overline{K}$ is countable as it is a countable union of countable sets. Moreover, each composite choice of $\overline{K}$ is incompatible with each composite choice of K. So $\omega_K^c = \omega_{\overline{K}}$ and $\omega_K^c \in \Omega_{\mathcal{P}}$.

(σ-3) is true as in the algebra case. □

Consider the sequence $\{K_n | n \geqslant 1\}$ where $K_n = \{\kappa_1, \ldots, \kappa_n\}$. Since K_n is an increasing sequence, the limit $\lim_{n\to\infty} K_n$ exists and is equal to K. Each K_n is a finite set of composite choices and we can compute an equivalent finite pairwise incompatible set of composite choices K'_n. For each K'_n we can compute the probability $P(K'_n)$, noting that the probability of infinite composite choices is 0.

According to [24, Corollay 2, page 23],

$$P(K) = P(\lim_{n\to\infty} K'_n) = \lim_{n\to\infty} P(K'_n)$$

because the limit $\lim_{n\to\infty} K'_n$ exists. So we can define measure μ as

$$\mu(\omega_K) = \lim_{n\to\infty} P(K'_n)$$

.

Theorem 5. *$\langle W_{\mathcal{P}}, \Omega_{\mathcal{P}}, \mu\rangle$ is a probability space according to Definition 4.*

Proof. (μ-1) and (μ-2) hold as for the finite case and for (μ-3) let

$$O = \{\omega_{L_1}, \omega_{L_2}, \ldots\}$$

be a countable set of subsets of $\Omega_{\mathcal{P}}$ such that the ω_{L_i}s are pairwise disjoint. Let L'_i be the pairwise incompatible set equivalent to L_i and let $\mathcal{L}$ be $\bigcup_{i=1}^{\infty} L'_i$. Since the ω_{L_i}s are pairwise disjoint, then $\mathcal{L}$ is pairwise incompatible. $\Omega_{\mathcal{P}}$ is a σ-algebra, so $\mathcal{L}$ is countable. Let $\mathcal{L}$ be $\{\kappa_1, \kappa_2, \ldots\}$ and let K'_n be $\{\kappa_1, \ldots, \kappa_n\}$. Then $\mu(O) = \lim_{n\to\infty} P(K'_n) = \lim_{n\to\infty} \sum_{\kappa\in K'_n} P(\kappa) = \sum_{\kappa\in\mathcal{L}} P(\kappa)$. Since $\mathcal{L} = \bigcup_{i=1}^{\infty} L'_i$, by rearranging the terms in the last summation we get

$$\mu(O) = \sum_{\kappa\in\mathcal{L}} P(\kappa) = \sum_{n=1}^{\infty} P(L'_n) = \sum_{n=1}^{\infty} \mu(\omega_{L_n}).$$

□

For a probabilistic logic program $\mathcal{P}$, the probability of a ground atom q is again given by $P(q) = \mu(\{w | w \in W_{\mathcal{P}}, w \models q\})$. If q has a countable set K of explanations such that K is covering then $\{w | w \in W_{\mathcal{P}} \wedge w \models q\} = \omega_K \in \Omega_{\mathcal{P}}$ and we say that $P(q)$ is *well-defined* for the distribution semantics. A program $\mathcal{P}$ is well-defined if the probability of all ground atoms in the grounding of $\mathcal{P}$ is well-defined.

Example 7. *Consider Example 5. Since the explanations in K^s are pairwise incompatible the probability of s can be computed as*

$$P(s) = b(1-a) + ba(1-a) + ba^2(1-a) + \ldots = \frac{b(1-a)}{1-a} = b.$$

since the sum is a geometric series. K^t is also pairwise incompatible and $P(\kappa^t) = 0$ so $P(t) = 1 - b + 0 = 1 - b$ which is what we intuitively expect.

Example 8. *In Example 6, the explanations in K^+ are pairwise incompatible so the probability of at_least_once_1 is given by*

$$\begin{aligned} P(at\_least\_once\_1) &= 1/3 + 2/3 \times 1/2 \times 1/3 + (2/3 \times 1/2)^2 \times 1/3 + \ldots \\ &= 1/3 + 1/9 + 1/27 \ldots \\ &= \frac{1/3}{1-1/3} = \frac{1/3}{2/3} = 1/2. \end{aligned}$$

since the sum is a geometric series.

For the query never_1, the explanations in K^- are pairwise incompatible so the probability of never_1 can be computed as

$$\begin{aligned} P(never\_1) &= 2/3/2 + 2/3 \times 1/2 \times 2/3 \times 1/2 + (2/3 \times 1/2)^2 \times 2/3 \times 1/2 + \ldots \\ &= 1/3 + 1/9 + 1/27 \ldots = 1/2. \end{aligned}$$

This expected as $never\_1 = \neg at\_least\_once\_1$.

We now want to show that every program has a countable set of countable explanations that is covering for each query. In the following, we consider only ground programs that however may be countably infinite, thus they can be the result of grounding a program with function symbols.

Given two sets of composite choices K_1 and K_2, define the conjunction $K_1 \otimes K_2$ of K_1 and K_2 as $K_1 \otimes K_2 = \{\kappa_1 \cup \kappa_2 | \kappa_1 \in K_1, \kappa_2 \in K_2, consistent(\kappa_1 \cup \kappa_2)\}$

Similarly to [11, 12], we define parameterized interpretations and a $IFPP_{\mathcal{P}}$ operator. Differently from [11, 12], here parameterized interpretations associate to each atom a set of composite choices instead of a Boolean formula.

Definition 8. A *parameterized positive two-valued interpretation Tr* of a ground probabilistic logic program $\mathcal{P}$ with Herbrand base $\mathcal{B}_{\mathcal{P}}$ is a set of pairs (a, K_a) with $a \in atoms$ and K_a a set of composite choices. A *parameterized negative two-valued interpretation Fa* of a ground probabilistic logic program P with Herbrand base $\mathcal{B}_{\mathcal{P}}$ is a set of pairs $(a, K_{\neg a})$ with $a \in \mathcal{B}_{\mathcal{P}}$ and $K_{\neg a}$ a set of composite choices.

Parameterized two-valued interpretations form a complete lattice where the partial order is defined as $I \leqslant J$ if $\forall (a, K_a) \in I, (a, L_a) \in J : \omega_{K_a} \subseteq \omega_{L_a}$. The least upper bound and greatest lower bound always exist and are $lub(X) = \{(a, \bigcup_{(a,K_a) \in I, I \in X} K_a) | a \in \mathcal{B}_{\mathcal{P}}\}$ and $glb(X) = \{(a, \bigotimes_{(a,K_a) \in I, I \in X} K_a) | a \in \mathcal{B}_{\mathcal{P}}\}$. The top element $\top$ is

$$\{(a, \{\emptyset\}) | a \in \mathcal{B}_{\mathcal{P}}\}$$

and the bottom element $\bot$ is

$$\{(a, \emptyset) | a \in \mathcal{B}_{\mathcal{P}}\}.$$

Definition 9. A *parameterized three-valued interpretation $\mathcal{I}$* of a ground probabilistic logic program $\mathcal{P}$ with Herbrand base $\mathcal{B}_{\mathcal{P}}$ is a set of triples $(a, K_a, K_{\neg a})$ with $a \in \mathcal{B}_{\mathcal{P}}$ and K_a and $K_{\neg a}$ sets of composite choices. A consistent parameterized three-valued interpretation $\mathcal{I}$ is such that $\forall (a, K_a, K_{\neg a}) \in \mathcal{I}$, $\omega_{K_a} \cap \omega_{K_{\neg a}} = \emptyset$.

Parameterized three-valued interpretations form a complete lattice where the partial order is defined as $I \leqslant J$ if $\forall (a, K_a, K_{\neg a}) \in I, (a, L_a, L_{\neg a}) \in J : \omega_{K_a} \subseteq \omega_{L_a}$ and $\omega_{K_{\neg a}} \subseteq \omega_{L_{\neg a}}$. The least upper bound and greatest lower bound always exist and are $lub(X) = \{(a, \bigcup_{(a,K_a,K_{\neg a}) \in I, I \in X} K_a, \bigcup_{(a,K_a,K_{\neg a}) \in I, I \in X} K_{\neg a}) | a \in \mathcal{B}_{\mathcal{P}}\}$ and $glb(X) = \{(a, \bigotimes_{(a,K_a,K_{\neg a}) \in I, I \in X} K_a, \bigotimes_{(a,K_a,K_{\neg a}) \in I, I \in X} K_{\neg a}) | a \in \mathcal{B}_{\mathcal{P}}\}$. The top element $\top$ is

$$\{(a, \{\emptyset\}, \{\emptyset\}) | a \in \mathcal{B}_{\mathcal{P}}\}$$

and the bottom element $\bot$ is

$$\{(a, \emptyset, \emptyset) | a \in \mathcal{B}_{\mathcal{P}}\}.$$

Definition 10. For a ground program $\mathcal{P}$, a two-valued parameterized positive interpretation Tr with pairs (a, L_a), a two-valued parameterized negative interpretation Fa with pairs $(a, M_{\neg a})$ and a three-valued parameterized interpretation $\mathcal{I}$ with triples $(a, K_a, K_{\neg a})$, we define $OpTrueP^{\mathcal{P}}_{\mathcal{I}}(Tr) = \{(a, L'_a) | a \in \mathcal{B}_{\mathcal{P}}\}$ where

$$L'_a = \begin{cases} \{\{(a, \emptyset, 1)\}\} & \text{if } a \in \mathcal{F} \\ \bigcup_{a \leftarrow b_1,\ldots,b_n,\neg c_1,\ldots,c_m \in \mathcal{R}} ((L_{b_1} \cup K_{b_1}) \otimes \ldots & \text{if } a \in \mathcal{B}_{\mathcal{P}} \backslash \mathcal{F} \\ \otimes (L_{b_n} \cup K_{b_n}) \otimes K_{\neg c_1} \otimes \ldots \otimes K_{\neg c_m}) & \end{cases}$$

$OpFalseP^{\mathcal{P}}_{\mathcal{I}}(Fa) = \{(a, M'_a) | a \in \mathcal{B}_{\mathcal{P}}\}$ where

$$M'_{\neg a} = \begin{cases} \{\{(a, \emptyset, 0)\}\} & \text{if } a \in \mathcal{F} \\ \bigotimes_{a \leftarrow b_1,\ldots,b_n,\neg c_1,\ldots,c_m \in \mathcal{R}} ((M_{\neg b_1} \otimes K_{\neg b_1}) \cup \ldots & \text{if } a \in \mathcal{B}_{\mathcal{P}} \backslash \mathcal{F} \\ \cup (M_{\neg b_n} \otimes K_{\neg b_n}) \cup K_{c_1} \cup \ldots \cup K_{c_m}) & \end{cases}$$

Proposition 2. *$OpTrueP^{\mathcal{P}}_{\mathcal{I}}$ and $OpFalseP^{\mathcal{P}}_{\mathcal{I}}$ are monotonic.*

Proof. Let us consider $OpTrueP^{\mathcal{P}}_{\mathcal{I}}$. We have to prove that if $Tr_1 \leqslant Tr_2$ then $OpTrueP^{\mathcal{P}}_{\mathcal{I}}(Tr_1) \leqslant OpTrueP^{\mathcal{P}}_{\mathcal{I}}(Tr_2)$. $Tr_1 \leqslant Tr_2$ means that

$$\forall (a, L_a) \in Tr_1, (a, M_a) \in Tr_2 : L_a \subseteq M_a.$$

Let (a, L'_a) be the elements of $OpTrueP^{\mathcal{P}}_{\mathcal{I}}(Tr_1)$ and (a, M'_a) the elements of $OpTrueP^{\mathcal{P}}_{\mathcal{I}}(Tr_2)$. We have to prove that $L'_a \subseteq M'_a$

If $a \in \mathcal{F}$ then $L'_a = M'_a = \{\{(a, \theta, 1)\}\}$. If $a \in \mathcal{B}_{\mathcal{P}} \backslash \mathcal{F}$, then L'_a and M'_a have the same structure. Since $\forall b \in \mathcal{B}_{\mathcal{P}} : L_b \subseteq M_b$, then $L'_a \subseteq M'_a$

We can prove similarly that $OpFalseP^{\mathcal{P}}_{\mathcal{I}}$ is monotonic. □

Since $OpTrueP^{\mathcal{P}}_{\mathcal{I}}$ and $OpFalseP^{\mathcal{P}}_{\mathcal{I}}$ are monotonic, they have a least fixpoint and a greatest fixpoint.

Definition 11 (Iterated Fixed Point). For a ground program $\mathcal{P}$, let $IFPP^{\mathcal{P}}$ be defined as $IFPP^{\mathcal{P}}(\mathcal{I}) = \{(a, K_a, K_{\neg a}) | (a, K_a) \in lfp(OpTrueP^{\mathcal{P}}_{\mathcal{I}}), (a, K_{\neg a}) \in gfp(OpFalseP^{\mathcal{P}}_{\mathcal{I}})\}$.

Proposition 3. *$IFPP^{\mathcal{P}}$ is monotonic.*

Proof. We have to prove that if $\mathcal{I}_1 \leqslant \mathcal{I}_2$ then $\mathit{IFPP}^{\mathcal{P}}(\mathcal{I}_1) \leqslant \mathit{IFPP}^{\mathcal{P}}(\mathcal{I}_2)$. $\mathcal{I}_1 \leqslant \mathcal{I}_2$ means that $\forall (a, L_a, L_{\neg a}) \in \mathcal{I}_1, (a, M_a, M_{\neg a}) \in \mathcal{I}_2 : L_a \subseteq M_a, L_{\neg a} \subseteq M_{\neg a}$. Let $(a, L'_a, L'_{\neg a})$ be the elements of $\mathit{IFPP}^{\mathcal{P}}(\mathcal{I}_1)$ and $(a, M'_a, M'_{\neg a})$ the elements of $\mathit{IFPP}^{\mathcal{P}}(\mathcal{I}_2)$. We have to prove that $L'_a \subseteq M'_a$ and $L'_{\neg a} \subseteq M'_{\neg a}$. This follows from the monotonicity of $\mathit{OpTrueP}^{\mathcal{P}}_{\mathcal{I}_1}$ and $\mathit{OpFalseP}^{\mathcal{P}}_{\mathcal{I}_2}$ in $\mathcal{I}_1$ and $\mathcal{I}_2$ respectively, which can be proved as in Proposition 2. $\square$

So $\mathit{IFPP}^{\mathcal{P}}$ has a least fixpoint. Let $\mathit{WFMC}(\mathcal{P})$ denote $\mathit{lfp}(\mathit{IFPP}^{\mathcal{P}})$, and let δ the smallest ordinal such that $\mathit{IFPP}^{\mathcal{P}} \uparrow \delta = \mathit{WFMC}(\mathcal{P})$. We refer to δ as the *depth* of $\mathcal{P}$.

Let us now prove that $\mathit{OpTrueP}^{\mathcal{P}}_{\mathcal{I}}$ and $\mathit{OpFalseP}^{\mathcal{P}}_{\mathcal{I}}$ are sound.

Lemma 3. *For a ground probabilistic logic program $\mathcal{P}$ with probabilistic facts $\mathcal{F}$, rules $\mathcal{R}$ and Herbrand base $\mathcal{B}_{\mathcal{P}}$, let L^{α}_a be the formula associated with atom a in $\mathit{OpTrueP}^{\mathcal{P}}_{\mathcal{I}} \uparrow \alpha$. For every atom a, total choice σ and iteration α, we have:*

$$w_\sigma \in \omega_{L^{\alpha}_a} \rightarrow \mathit{WFM}(w_\sigma|\mathcal{I}) \models a$$

where $w_\sigma|\mathcal{I}$ is obtained by adding to w_σ the atoms a for which $(a, K_a, K_{\neg a}) \in \mathcal{I}$ and $w_\sigma \in K_a$ as facts and by removing all the rules with a in the head for which $(a, K_a, K_{\neg a}) \in \mathcal{I}$ and $w_\sigma \in K_{\neg a}$.

Proof. Let us prove the lemma by transfinite induction: let us assume the thesis for all $\beta < \alpha$ and let us prove it for α. If α is a successor ordinal, then it is easily verified for $a \in \mathcal{F}$. Otherwise assume $w_\sigma \in \omega_{L^{\alpha}_a}$ where

$$L^{\alpha}_a = \bigcup_{a \leftarrow b_1, \dots, b_n, \neg c_1, \dots, c_m \in \mathcal{R}} ((L^{\alpha-1}_{b_1} \cup K_{b_1}) \otimes \dots \otimes (L^{\alpha-1}_{b_n} \cup K_{b_n}) \otimes K_{\neg c_1} \otimes \dots \otimes K_{\neg c_m})$$

This means that there is rule $a \leftarrow b_1, \dots, b_n, \neg c_1, \dots, c_m \in \mathcal{R}$ such that $w_\sigma \in \omega_{L^{\alpha-1}_{b_i} \cup K_{b_i}}$ for $i = 1, \dots, n$ and $w_\sigma \in \omega_{K_{\neg c_j}}$ for $j = 1 \dots, m$. By the inductive assumption and because of how $w_\sigma|\mathcal{I}$ is built then $\mathit{WFM}(w_\sigma|\mathcal{I}) \models b_i$ and $\mathit{WFM}(w_\sigma|\mathcal{I}) \models \neg c_j$ so $\mathit{WFM}(w_\sigma|\mathcal{I}) \models a$.

If α is a limit ordinal, then

$$L^{\alpha}_a = \mathit{lub}(\{L^{\beta}_a | \beta < \alpha\}) = \bigcup_{\beta < \alpha} L^{\beta}_a$$

If $w_\sigma \in \omega_{L^{\alpha}_a}$ then there must exist a $\beta < \alpha$ such that $w_\sigma \in \omega_{L^{\beta}_a}$. By the inductive assumption the hypothesis holds. $\square$

Lemma 4. *For a ground probabilistic logic program $\mathcal{P}$ with probabilistic facts $\mathcal{F}$, rules $\mathcal{R}$ and Herbrand base $\mathcal{B}_{\mathcal{P}}$, let $M^{\alpha}_{\neg a}$ be the set of composite choices associated with atom a in $\mathit{OpFalseP}^{\mathcal{P}}_{\mathcal{I}} \downarrow \alpha$. For every atom a, total choice σ and iteration α, we have:*

$$w_\sigma \in \omega_{M^{\alpha}_{\neg a}} \rightarrow \mathit{WFM}(w_\sigma|\mathcal{I}) \models \neg a$$

where $w_\sigma|\mathcal{I}$ is built as in Lemma 3.

Proof. Similar to the proof of Lemma 3. □

The following Lemma shows that $\mathit{IFPP}^{\mathcal{P}}$ is sound.

Lemma 5. *For a ground probabilistic logic program $\mathcal{P}$ with probabilistic facts $\mathcal{F}$, rules $\mathcal{R}$ and Herbrand base $\mathcal{B}_{\mathcal{P}}$, let K_a^α and $K_{\neg a}^\alpha$ be the formulas associated with atom a in $\mathit{IFPP}^{\mathcal{P}} \uparrow \alpha$. For every atom a, total choice σ and iteration α, we have:*

$$w_\sigma \in \omega_{K_a^\alpha} \rightarrow \mathit{WFM}(w_\sigma) \models a$$

$$w_\sigma \in \omega_{K_{\neg a}^\alpha} \rightarrow \mathit{WFM}(w_\sigma) \models \neg a$$

Proof. Let us first prove that for all α, $\mathit{WFM}(w_\sigma) = \mathit{WFM}(w_\sigma|\mathit{IFPP}^{\mathcal{P}} \uparrow \alpha)$. We can prove it by transfinite induction. Consider the case of α a successor ordinal. Consider an atom b. If $w_\sigma \notin \omega_{K_b^\alpha}$ and $w_\sigma \notin \omega_{K_{\neg b}^\alpha}$ then the rules for b in w_σ and $w_\sigma|\mathit{IFPP}^{\mathcal{P}} \uparrow \alpha$ are the same. If $w_\sigma \in \omega_{K_b^\alpha}$ then b is a fact in $w_\sigma|\mathit{IFPP}^{\mathcal{P}} \uparrow \alpha$ but, according to Lemma 3, $\mathit{WFM}(w_\sigma|\mathit{IFPP}^{\mathcal{P}} \uparrow (\alpha-1)) \models b$. For the inductive hypothesis $\mathit{WFM}(w_\sigma) \models b$ so b has the same truth value in $\mathit{WFM}(w_\sigma)$ and $\mathit{WFM}(w_\sigma|\mathit{IFPP}^{\mathcal{P}} \uparrow \alpha)$. Similarly, if $w_\sigma \in \omega_{K_{\neg b}^\alpha}$, then $\mathit{WFM}(w_\sigma) \models \neg b$ and b has the same truth value in $\mathit{WFM}(w_\sigma)$ and $\mathit{WFM}(w_\sigma|\mathit{IFPP}^{\mathcal{P}} \uparrow \alpha)$. So overall $\mathit{WFM}(w_\sigma) = \mathit{WFM}(w_\sigma|\mathit{IFPP}^{\mathcal{P}} \uparrow \alpha)$.

If α is a limit ordinal, then $K_b^\alpha = \bigcup_{\beta<\alpha} K_b^\beta$ and $K_{\neg b}^\alpha = \bigcup_{\beta<\alpha} K_b^\beta$. So if $w_\sigma \in \omega_{K_b^\alpha}$ there is a β such $w_\sigma \in \omega_{K_b^\beta}$ and for the inductive hypothesis $\mathit{WFM}(w_\sigma) \models b$ so b has the same truth value in $\mathit{WFM}(w_\sigma)$ and $\mathit{WFM}(w_\sigma|\mathit{IFPP}^{\mathcal{P}} \uparrow \alpha)$. Similarly if $w_\sigma \in \omega_{K_{\neg b}^\alpha}$.

We can now prove the lemma by transfinite induction. Consider the case of α a successor ordinal. Since $(a, K_a^\alpha) \in \mathit{lfp}(\mathit{OpTrueP}^{\mathcal{P}}_{\mathit{IFPP}\uparrow(\alpha-1)})$, by Lemma 3

$$w_\sigma \in \omega_{K_a^\alpha} \rightarrow \mathit{WFM}(w_\sigma|\mathit{IFPP}^{\mathcal{P}} \uparrow (\alpha-1)) \models a$$

Since $\mathit{WFM}(w_\sigma|\mathit{IFPP}^{\mathcal{P}} \uparrow (\alpha-1)) = \mathit{WFM}(w_\sigma)$, (5) is proved.

Since $(a, K_{\neg a}^\alpha) \in \mathit{gfp}(\mathit{OpFalseP}^{\mathcal{P}}_{\mathit{IFPP}^{\mathcal{P}}\uparrow(\alpha-1)})$, by Lemma 4

$$w_\sigma \in \omega_{K_{\neg a}^\alpha} \rightarrow \mathit{WFM}(w_\sigma|\mathit{IFPP}^{\mathcal{P}} \uparrow (\alpha-1)) \models \neg a$$

Since $\mathit{WFM}(w_\sigma|\mathit{IFPP}^{\mathcal{P}} \uparrow (\alpha-1)) = \mathit{WFM}(w_\sigma)$, (5) is proved.

If α is a limit ordinal, $K_a^\alpha = \bigcup_{\beta<\alpha} K_a^\beta$ and $K_{\neg a}^\alpha = \bigcup_{\beta<\alpha} K_a^\beta$. If $w_\sigma \in \omega_{K_a^\alpha}$ there is a β such that $w_\sigma \in \omega_{K_b^\alpha}$ and by the inductive hypothesis (5) is proved. Similarly for (5). □

The following Lemma shows that $\mathit{IFPP}^{\mathcal{P}}$ is complete.

Lemma 6. *For a ground probabilistic logic program $\mathcal{P}$ with probabilistic facts $\mathcal{F}$, rules $\mathcal{R}$ and Herbrand base $\mathcal{B}_{\mathcal{P}}$, let K_a^α and $K_{\neg a}^\alpha$ be the formulas associated with atom a in $\mathit{IFPP}^{\mathcal{P}} \uparrow \alpha$. For every atom a, total choice σ and iteration α, we have:*

$$a \in \mathit{IFP}^{w_\sigma} \uparrow \alpha \rightarrow w_\sigma \in K_a^\alpha$$

$$\neg a \in \mathit{IFP}^{w_\sigma} \uparrow \alpha \rightarrow w_\sigma \in K_{\neg a}^\alpha$$

Proof. Let us prove it by double transfinite induction. If α is a successor ordinal, assume that

$$a \in \mathit{IFP}^{w_\sigma} \uparrow (\alpha - 1) \rightarrow w_\sigma \in K_a^{\alpha-1}$$

$$\neg a \in \mathit{IFP}^{w_\sigma} \uparrow (\alpha - 1) \rightarrow w_\sigma \in K_{\neg a}^{\alpha-1}$$

Let us perform transfinite induction on the iterations of $\mathit{OpTrue}^{\mathcal{P}}_{\mathit{IFPP}^{\mathcal{P}}\uparrow(\alpha-1)}$. Let us consider a successor ordinal δ: assume that

$$a \in \mathit{OpTrue}^{w_\sigma}_{\mathit{IFP}^{w_\sigma}\uparrow(\alpha-1)} \uparrow (\delta - 1) \rightarrow w_\sigma \in L_a^{\delta-1}$$

$$\neg a \in \mathit{OpFalse}^{w_\sigma}_{\mathit{IFP}^{w_\sigma}\uparrow(\alpha-1)} \downarrow (\delta - 1) \rightarrow w_\sigma \in M_{\neg a}^{\delta-1}$$

and prove that

$$a \in \mathit{OpTrue}^{w_\sigma}_{\mathit{IFP}^{w_\sigma}\uparrow(\alpha-1)} \uparrow \delta \rightarrow w_\sigma \in L_a^{\delta}$$

$$\neg a \in \mathit{OpFalse}^{w_\sigma}_{\mathit{IFP}^{w_\sigma}\uparrow(\alpha-1)} \downarrow \delta \rightarrow w_\sigma \in M_{\neg a}^{\delta}$$

Consider a. If $a \in \mathcal{F}$ then it is easily proved.

For other atoms a, $a \in \mathit{OpTrue}^{w_\sigma}_{\mathit{IFP}^{w_\sigma}\uparrow(\alpha-1)} \uparrow \delta$ means that there is a rule $a \leftarrow b_1, \ldots, b_n, \neg c_1, \ldots, c_m$ such that for all $i = 1, \ldots, n$ $b_i \in \mathit{OpTrue}^{w_\sigma}_{\mathit{IFP}^{w_\sigma}\uparrow(\alpha-1)} \uparrow (\delta - 1)$ and for all $j = 1, \ldots, m$ $\neg c_j \in \mathit{IFP}^{w_\sigma} \uparrow (\alpha - 1)$. For the inductive hypothesis $\forall i : w_\sigma \in L_{b_i}^{\delta-1} \vee w_\sigma \in K_{b_i}^{\alpha-1}$ and $\forall j : w_\sigma \in K_{\neg c_j}^{\alpha-1}$ so, for the definition of $\mathit{OpTrue}^{w_\sigma}_{\mathit{IFP}^{w_\sigma}\uparrow(\alpha-1)}$, $w_\sigma \in L_a^\delta$. Analogously for $\neg a$.

If δ is a limit ordinal, then $L_a^\delta = \bigcup_{\mu<\delta} L_a^\mu$ and $M_{\neg a}^\delta = \bigotimes_{\mu<\delta} M_{\neg a}^\mu$. For the inductive hypothesis for all $\mu < \delta$

$$a \in \mathit{OpTrue}^{w_\sigma}_{\mathit{IFP}^{w_\sigma}\uparrow(\alpha-1)} \uparrow \mu \rightarrow w_\sigma \in L_a^{\mu}$$

$$\neg a \in \mathit{OpFalse}^{w_\sigma}_{\mathit{IFP}^{w_\sigma}\uparrow(\alpha-1)} \downarrow \mu \rightarrow w_\sigma \in M_{\neg a}^{\mu}$$

If $a \in \mathit{OpTrue}^{w_\sigma}_{\mathit{IFP}^{w_\sigma}\uparrow(\alpha-1)} \uparrow \delta$, then there exists a $\mu < \delta$ such that $a \in \mathit{OpTrue}^{w_\sigma}_{\mathit{IFP}^{w_\sigma}\uparrow(\alpha-1)} \uparrow \mu$. For the inductive hypothesis, $w_\sigma \in L_a^\delta$.

If $\neg a \in \mathit{OpFalse}^{w_\sigma}_{\mathit{IFP}^{w_\sigma}\uparrow(\alpha-1)} \downarrow \delta$, then, for all $\mu < \delta$, $\neg a \in \mathit{OpFalse}^{w_\sigma}_{\mathit{IFP}^{w_\sigma}\uparrow(\alpha-1)} \downarrow \mu$. For the inductive hypothesis, $w_\sigma \in M_a^\delta$.

Consider a limit α. Then $K_a^\alpha = \bigcup_{\beta<\alpha} K_a^\beta$ and $K_{\neg a}^\alpha = \bigcup_{\beta<\alpha} K_{\neg a}^\beta$. The inductive hypothesis is

$$a \in \mathit{IFP}^{w_\sigma} \uparrow \beta \rightarrow w_\sigma \in K_a^{\beta}$$

$$\neg a \in \mathit{IFP}^{w_\sigma} \uparrow \beta \rightarrow w_\sigma \in K_{\neg a}^{\beta}$$

If $a \in \mathit{IFP}^{w_\sigma} \uparrow \alpha$, then there exists a $\beta < \alpha$ such that $a \in \mathit{IFP}^{w_\sigma} \uparrow \beta$. For the inductive hypothesis $w_\sigma \in K_a^\beta$ so $w_\sigma \in K_a^\alpha$. Similarly for $\neg a$. □

We can now prove that $\mathit{IFPP}^{\mathcal{P}}$ is sound and complete.

Theorem 6. *For a ground probabilistic logic program $\mathcal{P}$ with Herbrand base $\mathcal{B}_\mathcal{P}$, let K_a^α and $K_{\neg a}^\alpha$ be the formulas associated with atom a in $\mathit{IFPP}^\mathcal{P} \uparrow \alpha$. For every atom a and total choice σ, there is an iteration α_0 such that for all $\alpha > \alpha_0$ we have:*

$$w_\sigma \in \omega_{K_a^\alpha} \leftrightarrow \mathit{WFM}(w_\sigma) \models a$$

$$w_\sigma \in \omega_{K_{\neg a}^\alpha} \leftrightarrow \mathit{WFM}(w_\sigma) \models \neg a$$

Proof. The $\rightarrow$ direction is Lemma 5. In the other direction, $\mathit{WFM}(w_\sigma) \models a$ implies $\exists \alpha_0 \forall \alpha \geqslant \alpha_0 : \mathit{IFP}_{w_\sigma} \uparrow \alpha \models a$. For Lemma 6, $w_\sigma \in \omega_{K_a^\alpha}$. $\mathit{WFM}(w_\sigma) \models \neg a$ implies $\exists \alpha_0 \forall \alpha \geqslant \alpha_0 : \mathit{IFP}^{w_\sigma} \uparrow \alpha \models \neg a$. For Lemma 6, $w_\sigma \in \omega_{K_{\neg a}^\alpha}$. □

We can also prove that every query for every program has a countable set of countable explanations that is covering.

Theorem 7. *For a ground probabilistic logic program $\mathcal{P}$, let K_a^α and $K_{\neg a}^\alpha$ be the formulas associated with atom a in $\mathit{IFPP}^\mathcal{P} \uparrow \alpha$. For every atom a and every iteration α, K_a^α and $K_{\neg a}^\alpha$ are countable sets of countable composite choices.*

Proof. It can be proved by observing that each iteration of $\mathit{OpTrueP}^\mathcal{P}_{\mathit{IFPP}^\mathcal{P} \uparrow \beta}$ and $\mathit{OpFalseP}^\mathcal{P}_{\mathit{IFPP}^\mathcal{P} \uparrow \beta}$ generates countable sets of countable explanations since the set of rules is countable. □

So the probability measure $\mu(\{w | w \in W_\mathcal{P}, w \models a\})$ for a ground atom a is well defined. Moreover, since the program is sound, for all atoms a, $\omega_{K_a^\delta} = \omega^c_{K_{\neg a}^\delta}$ where δ is the depth of the program, as in each world a is either true or false and $\mathit{lfp}(\mathit{IFPC}^\mathcal{P})$ is a consistent parameterized three-valued interpretation.

6. Comparison with Sato and Kameya's Definition

Sato and Kameya [8] define the distribution semantics for definite programs, i.e., programs without negative literals. They build a probability measure on the set of Herbrand interpretations from a collection of finite distributions. Let $\mathcal{F}$ be $\{f_1, f_2, \ldots\}$ and let X_i be a random variable associated to f_i whose domain is $\{0, 1\}$.

They define $V_\mathcal{F}$ as a topological space with the product topology such that each $\{0, 1\}$ is equipped with the discrete topology.

In order to clarify this definition, let us introduce some topology terminology. A *topology* on a set $\mathcal{V}$ [26, page 23] is a collection Ψ of subsets of $\mathcal{V}$, called the *open sets*, satisfying: (t-1) any union of elements of Ψ belongs to Ψ, (t-2) any finite intersection of elements of Ψ belongs to Ψ, (t-3) $\varnothing$ and $\mathcal{V}$ belong to Ψ. We say that $(\mathcal{V}, \Psi)$ is a *topological space*. The *discrete topology* of a set $\mathcal{V}$ [27, page 41] is the powerset $\mathbf{P}(\mathcal{V})$ of $\mathcal{V}$.

The *infinite Cartesian product* of sets ψ_i for $i = 1, 2, \ldots$ is

$$\rho = \bigtimes_{i=1}^{\infty} \psi_i = \{(s_1, s_2, \ldots) | s_i \in \psi_i, i = 1, 2, \ldots\}$$

A *product topology* [26, page 53] on the infinite Cartesian product $\times_{i=1}^{\infty} \mathcal{V}_i$ is obtained by taking all possible unions of open sets of the form $\times_{i=1}^{\infty} \nu_i$ where (p-1) ν_i is open $\forall i$ and (p-2) for all but finitely many i, $\nu_i = \mathcal{V}_i$. Sets satisfying (p-1) and (p-2) are called *cylinder sets*. There exists a countable number of them.

So $V_{\mathcal{F}} = \times_{i=1}^{\infty}\{0,1\}$, i.e., it is an infinite Cartesian product with $\mathcal{V}_i = \{0,1\}$ for all i. Sato and Kameya [8] define a probability measure $\eta_{\mathcal{F}}$ over the sample space $V_{\mathcal{F}}$ from a collection of finite joint distributions $P_{\mathcal{F}}^{(n)}(X_1 = k_1, \ldots, X_n = k_n)$ for $n \geqslant 1$ such that

$$\begin{cases} 0 \leqslant P_{\mathcal{F}}^{(n)}(X_1 = k_1, \ldots, X_n = k_n) \leqslant 1 \\ \sum_{k_1,\ldots,k_n} P_{\mathcal{F}}^{(n)}(X_1 = k_1, \ldots, X_n = k_n) = 1 \\ \sum_{k_{n+1}} P_{\mathcal{F}}^{(n+1)}(X_1 = k_1, \ldots, X_{n+1} = k_{n+1}) = P_{\mathcal{F}}^{(n)}(X_1 = k_1, \ldots, X_n = k_n) \end{cases} \tag{2}$$

The last equation is called the compatibility condition. It can be proved [24] from the compatibility condition that there exists a probability space $(V_{\mathcal{F}}, \Psi_{\mathcal{F}}, \eta_{\mathcal{F}})$ where $\eta_{\mathcal{F}}$ is a unique probability measure on $\Psi_{\mathcal{F}}$, the minimal σ-algebra containing open sets of $V_{\mathcal{F}}$ such that for any n,

$$\eta_{\mathcal{F}}(X_1 = k_1, \ldots, X_n = k_n) = P_T^{(n)}(X_1 = k_1, \ldots, X_n = k_n). \tag{3}$$

$P_{\mathcal{F}}^{(n)}(X_1 = k_1, \ldots, X_n = k_n)$ is defined as $P_{\mathcal{F}}^{(n)}(X_1 = k_1, \ldots, X_n = k_n) = \pi_1 \ldots \pi_n$ where $\pi_i = p_i$ if $k_i = 1$ and $\pi_i = 1 - p_i$ if $k_i = 0$, with p_i the annotation of fact f_i. This definition clearly satisfies the properties in (2).

The distribution $P_{\mathcal{F}}^{(n)}(X_1 = k_1, \ldots, X_n = k_n)$ is then extended to a probability measure over the set of Herbrand interpretations of the whole program. Let $\mathcal{B}_{\mathcal{P}}$ be $\{a_1, a_2, \ldots\}$ and let Y_i be a random variable associated to a_i whose domain is $\{0,1\}$. Moreover, let $a^k = a$ if $k = 1$ and $a^k = \neg a$ if $k = 0$. $V_{\mathcal{P}}$ is the infinite Cartesian product $V_{\mathcal{P}} = \times_{i=1}^{\infty}\{0,1\}$.

Measure $\eta_{\mathcal{F}}$ is extended to $\eta_{\mathcal{P}}$ by introducing a series of finite joint distributions $P_{\mathcal{P}}^{(n)}(Y_1 = k_1, \ldots, Y_n = k_n)$ for $n = 1, 2, \ldots$ by

$$[a_1^{k_1} \wedge \ldots \wedge a_n^{k_n}]_{\mathcal{F}} = \{v \in V_{\mathcal{F}} | \mathit{lhm}_{\mathcal{P}}(v) \models a_1^{k_1} \wedge \ldots \wedge a_n^{k_n}\}$$

where $\mathit{lhm}_{\mathcal{P}}(v)$ is the least Herbrand model of $\mathcal{R} \cup \mathcal{F}_v$, with $\mathcal{F}_v = \{f_i | v_i = 1\}$.

Then let

$$P_{\mathcal{P}}^{(n)}(Y_1 = k_1, \ldots, Y_n = k_n) = \eta_{\mathcal{F}}([a_1^{k_1} \wedge \ldots \wedge a_n^{k_n}]_{\mathcal{F}})$$

Sato and Kameya state that $[a_1^{k_1} \wedge \ldots \wedge a_n^{k_n}]_{\mathcal{F}}$ is $\eta_{\mathcal{F}}$-measurable and that, by definition, $P_{\mathcal{P}}^{(n)}$ satisfy the compatibility condition

$$\sum_{k_{n+1}} P_{\mathcal{P}}^{(n+1)}(Y_1 = k_1, \ldots, Y_{n+1} = k_{n+1}) = P_{\mathcal{P}}^{(n)}(Y_1 = k_1, \ldots, Y_n = k_n)$$

Hence there exists a unique probability measure $\eta_{\mathcal{P}}$ over $\Psi_{\mathcal{P}}$ which is an extension of $\eta_{\mathcal{F}}$.

In order to relate this definition with ours, we need to introduce some more terminology on σ-algebras.

Definition 12 ([24, page 6]). The minimal σ-algebra Σ' containing a nonempty set Σ of subsets of $\mathcal{W}$, is a σ-algebra such that (m-1) $\Sigma' \supseteq \Sigma$, and (m-2) $\Sigma'' \supseteq \Sigma'$ whenever $\Sigma'' \supseteq \Sigma$ and Σ'' is a σ-algebra. Such a minimal σ-algebra Σ' containing Σ is also called the *σ-algebra generated by* Σ and is denoted by $\sigma(\Sigma)$.

The minimal σ-algebra of a set Σ $\sigma(\Sigma)$ always exists and is unique [24, page 7].

Definition 13 (Product Space). For any measurable spaces $(\mathcal{W}_i, \Omega_i)$, $i = 1, 2, \ldots$, define

$$\mathcal{G} = \bigcup_{m=1}^{\infty} \{ \bigtimes_{i=1}^{\infty} \omega_i | \omega_i \in \Omega_i, 1 \leqslant i \leqslant m \text{ and } \omega_i = \mathcal{W}_i, i > m \}$$

$$\bigtimes_{i=1}^{\infty} \Omega_i = \sigma(\mathcal{G})$$

$$\bigtimes_{i=1}^{\infty} (\mathcal{W}_i, \Omega_i) = (\bigtimes_{i=1}^{\infty} \mathcal{W}_i, \bigtimes_{i=1}^{\infty} \Omega_i)$$

Then $\bigtimes_{i=1}^{\infty}(\mathcal{W}_i, \Omega_i)$ is called the *infinite-dimensional product-measurable space* and $\bigtimes_{i=1}^{\infty} \Omega_i$ is the *product σ-algebra.*

It is clear that if $\mathcal{W}_i = \{0, 1\}$ and $\Omega_i = \mathbf{P}(\{0, 1\})$ for all i, then $\mathcal{G}$ is the set of all possible unions of cylinder sets so it is the product topology on $\bigtimes_{i=1}^{\infty} \mathcal{W}_i$ and

$$\bigtimes_{i=1}^{\infty} (\mathcal{W}_i, \Omega_i) = (V_{\mathcal{F}}, \Psi_{\mathcal{F}})$$

i.e, the infinite-dimensional product-measurable space and the minimal σ-algebra containing open sets of $V_{\mathcal{F}}$ coincide. Moreover, according to [24, Exercise 1.3.6], $\Psi_{\mathcal{F}}$ is the minimal σ-algebra generated by cylinder sets.

An infinite Cartesian product $\rho = \bigtimes_{i=1}^{\infty} \nu_i$ is *consistent* if it is different from the empty set, i.e., if $\nu_i \neq \varnothing$ for all $i = 1, 2, \ldots$. We can establish a bijective map between infinite consistent Cartesian products $\rho = \bigtimes_{i=1}^{\infty} \nu_i$ and composite choices: $\gamma_{\mathcal{F}}(\bigtimes_{i=1}^{\infty} \nu_i) = \{(f_i, \varnothing, k_i) | \nu_i = \{k_i\}\}$.

Lemma 7. *Each element of $\Psi_{\mathcal{F}}$ can be written as a countable union of consistent possibly infinite Cartesian products:*

$$\psi = \bigcup_{j=1}^{\infty} \rho_j \tag{4}$$

where $\rho_j = \bigtimes_{i=1}^{\infty} \nu_i$ and $\nu_i \in \{\{0\}, \{1\}, \{0, 1\}\}$ for all $i = 1, 2, \ldots$.

Proof. We will show that $\Psi_{\mathcal{F}}$ and Φ, the set of all elements of the form (4), coincide. Given a ψ for the form (4), each ρ_j can be written as a countable union of cylinder sets so it belongs to $\Psi_{\mathcal{F}}$. Since $\Psi_{\mathcal{F}}$ is a σ-algebra and ψ is a countable union, then $\psi \in \Psi_{\mathcal{F}}$ and $\Phi \subseteq \Psi_{\mathcal{F}}$.

We can prove that Φ is a σ-algebra using the same technique of Lemma 2, where Cartesian products replace composite choices. Φ contains cylinder sets: even if each ρ_j must be consistent, inconsistent sets are empty set of worlds, so they can be removed from union (4). As $\Psi_{\mathcal{F}}$ is the minimal σ-algebra containing cylinder sets, then $\Psi_{\mathcal{F}} \subseteq \Phi$. $\square$

So each element ψ of $\Psi_{\mathcal{F}}$ can be written as a countable union of consistent possibly infinite Cartesian products.

Lemma 8. *Consider the function* $\Gamma_{\mathcal{F}} : \Psi_{\mathcal{F}} \to \Omega_{\mathcal{P}}$ *defined by* $\Gamma_{\mathcal{F}}(\psi) = \omega_K$ *where* $\psi = \bigcup_{j=1}^{\infty} \rho_j$ *and* $K = \bigcup_{j=1}^{\infty}\{\gamma_{\mathcal{F}}(\rho_j)\}$. *Then* $\Gamma_{\mathcal{F}}$ *is bijective.*

Proof. Immediate because $\gamma_{\mathcal{F}}$ is bijective. $\square$

Theorem 8. *Probability measure* μ *coincides with* $\eta_{\mathcal{P}}$ *for definite programs.*

Proof. Consider $\langle X_1 = k_1, \ldots, X_n = k_n\rangle$ and let K be

$$\{\{(f_1, \emptyset, k_1), \ldots, (f_n, \emptyset, k_n)\}\}.$$

Then $K = \Gamma_{\mathcal{F}}(\langle X_1 = k_1, \ldots, X_n = k_n\rangle)$ and μ assigns probability $\pi_1 \ldots \pi_n$ to K, where $\pi_i = p_i$ if $k_i = 1$ and $\pi_i = 1 - p_i$ otherwise.

So μ is in accordance with $P_{\mathcal{F}}^{(n)}$. But $P_{\mathcal{F}}^{(n)}$ can be extended in only one way to a probability measure $\eta_{\mathcal{F}}$ and there is a bijection between $\Psi_{\mathcal{F}}$ and $\Omega_{\mathcal{P}}$, so μ is in accordance with $\eta_{\mathcal{F}}$ on all $\Psi_{\mathcal{F}}$.

Now consider $C = a_1^{k_1} \wedge \ldots \wedge a_n^{k_n}$. Since $\mathit{IFPP}^{\mathcal{P}} \uparrow \delta$ is such that K_a^{δ} and $K_{\neg a}^{\delta}$ are countable sets of countable composite choices for all atoms a, we can compute a covering set of explanations K for C by taking a finite conjunction of countable sets of composite choices, so K is a countable set of countable composite choices.

Clearly $P_{\mathcal{P}}^{(n)}(Y_1 = k_1, \ldots, Y_n = k_n)$ coincides with $\mu(\omega_K)$. But $P_{\mathcal{P}}^{(n)}$ can be extended in only one way to a probability measure $\eta_{\mathcal{P}}$, so μ is in accordance with $\eta_{\mathcal{P}}$ on all $\Psi_{\mathcal{P}}$ when $\mathcal{P}$ is a definite program. $\square$

7. Examples

Let us now see some examples where there are goals that have an infinite number of explanations from a normal probabilistic logic program. These examples and those that will be mentioned below in Section 8 show that the extended semantics allows users to write programs that solve concrete problems.

We first consider the game of dice proposed in [5] and discussed in Example 2: the player repeatedly throws a die. When the outcome is six, the game stops. This game can be modeled with the following Logic Program with Annotated Disjunctions (LPAD) [5]:

```
on(0,1):1/6;on(0,2):1/6;on(0,3):1/6;
on(0,4):1/6;on(0,5):1/6;on(0,6):1/6.

on(X,1):1/6;on(X,2):1/6;on(X,3):1/6;
on(X,4):1/6;on(X,5):1/6;on(X,6):1/6:-
  X1 is X-1,X1>=0,on(X1,_),
  \+ on(X1,6).
at_least_once_1:- on(_,1).
never_1:- \+ at_least_once_1.
```

Note that PLP languages under the distribution semantics are equally expressive, there are linear transformations from one language to another that preserve the semantics. We use an LPAD here instead of ProbLog as in Example 2 for its more general syntax that allows a more succinct program and to give an example of the transformation between ProbLog and LPADs. This programs has function symbols because it uses integer arithmetics and integers can be represented with function symbols.

Another example is a prefix parser for probabilistic context free grammars, from [13, 14]. The program below computes the probability that a string is a prefix of a string generated by the grammar. Again the program is presented as an LPAD for succinctness.

```
% grammar
% 0.4:S->SS
% 0.3:S->a
% 0.3:S->b
pre_pcfg(L):- pre_pcfg(['S'],L,[]).

pre_pcfg([A|R],L0,[]):-
  rule(A,L0,RHS),      % A is a nonterminal
  pre_pcfg(RHS,L0,[]). % pseudo success, R is discarded

pre_pcfg([A|R],L0,L2):-
  rule(A,L0,RHS),      % rule A->RHS is selected
  pre_pcfg(RHS,L0,L1). % recursion
  pre_pcfg(R,L1,L2).   % recursion

pre_pcfg([A|R],[A|L1],L2):-
  \+ rule(A,_,_),     % A is a terminal, consume A
  pre_pcfg(R,L1,L2).

pre_pcfg([],L,L).      % termination

rule('S',L,['S','S']):0.4; rule('S',L,[a]):0.3;
rule('S',L,[b]):0.3.
```

Clearly the number of explanations for the query `pre_cdg(L)` with `L` a string may be infinite as there may be an infinite number of strings that start with `L`.

Another interesting example involves handling domains with a variable number of objects. BLOG [28] is a system designed for handling unknown objects and the paper presents an example where the blips on a radar screen are used to probabilistically infer the presence of aircrafts, of which there are an unknown number.

This example has been represented in logic programming in [29] and we report below an LPAD version

```
numObj(N, N) :-
  \+ more(N).

numObj(N, N2) :-
  more(N),
  N1 is N + 1,
  numObj(N, N2).

more(N):0.3.

aircraft(I):-
 numObj(0,N),
 between(1, N, I).

blip(X, Y, T):-
  aircraft(I),
  inGrid(I, T),
  xpos(I, T, X),
  ypos(I, T, Y).
  producesBlip(I, T).

producesBlip(I, T):0.1.
```

In this example, the number of objects follows a geometric distribution with parameter 0.3: for each possible number of aircrafts, there is probability 0.3 that there is one more. A blip on the radar screen may be generated by an aircraft `I` where `I` varies from 1 to the number of aircrafts. Clearly, each possible number of aircrafts has a non-zero probability so the number of possible explanations for goal of the form `blip(X,Y,T)` is infinite.

A model checker for a fragment of Probabilistic Computation Tree Logic (PCTL) is presented in [15]. PCTL an extension of computation tree logic which allows for probabilistic quantification. The LPAD version of the model checker is

```
% State Formulae
models(S, prop(A)) :-
  holds(S,A).
models(S, neg(A)) :-
  \+ models(S,A).
```

```
models(S, and(SF1, SF2)):-
  models(S, SF1),
  models(S, SF2).
models(S, pr(PF, gt, B)) :-
  prob(pmodels(S, PF), P),
  P > B.
models(S, pr(PF, geq, B)) :-
  prob(pmodels(S, PF), P),
  P >= B.

trans(s0,I,s0):0.5; trans(s0,I,s1):0.3;
trans(s0,I,s2):0.2.
trans(s1,I,s1):0.4; trans(s1,I,s3):0.1;
trans(s1,I,s4):0.5.
trans(s4,_,s3).

% Path Formulae
pmodels(S, PF) :-
  pmodels(S, PF, _).
pmodels(S, until(SF1, SF2), H) :-
  models(S,SF2).
pmodels(S, until(SF1, SF2), H) :-
  models(S, SF1),
  trans(S, H, T),
  pmodels(T, until(SF1, SF2), next(H)).
pmodels(S, next(SF), H) :-
  trans(S, H, T),
  pmodels(T, SF).
```

The logic partitions formulas into state formulas and path formulas. The first are checked with goals of the form `models(S, SF)` that return true if state formula `SF` is true in state `S`. Path formulas are checked with goals of the form `pmodels(S,PF)` that return true if path formula `PF` is true in state `S`. Note that path formula take into account state transitions, that follow a Markov chain model where a transition from a state depends only on the current state. The probability of a path formula at a state requires taking into account all paths starting from the state in which the formula holds. Since the paths may be infinite, queries to this program may have an infinite number of explanations.

Note that this program makes also use of nested probability computations, as the last two clauses for `models/2` have the predicate `prob/2` in the body that computes the probability of a query. While we haven't defined a semantics for this case, it is not difficult to define one provided the nested calls are well-founded, i.e., that there are no circular dependencies, please see [19, 30].

8. Inference

Recently, a number of papers have started to appear that present techniques for exact inference when the number of explanations is infinite. In [13, 14] the authors extended PRISM by considering programs under the *generative exclusiveness condition*: at any choice point in any execution path of the top-goal, the choice is done according to a value sampled from a PRISM probabilistic switch. The generative exclusiveness condition implies that every disjunction is exclusive and originates from a probabilistic choice made by some switch.

In this case, a *cyclic explanation graph* can be computed that encodes the dependence of atoms on probabilistic switches. The graph is composed of formulas of the form

$$H \Leftrightarrow (\texttt{msw}(id_H, v_1) \wedge \beta_1) \vee \ldots \vee (\texttt{msw}(id_H, v_M) \wedge \beta_M)$$

where H is a ground atom and the β_is for $i = 1, \ldots, M$ are conjunctions of ground atoms and switch atoms. The authors show that from explanations graphs a system of equations can be obtained of the form

$$P(H) = P_{DB}(\texttt{msw}(id_H, v_1))P(\beta_1) + \ldots + P_{DB}(\texttt{msw}(id_H, v_M))P(\beta_M)$$

where $P(H)$ is a variable, $P_{DB}(\texttt{msw}(id_H, v_i)$ is a constant (the probability of the switch value from the program) and $P(\beta_i)$ is a product of some switch probabilities and variables.

The authors of [13, 14] show that by first assigning all atoms probability 0 and repeatedly applying the equations to compute updated values results in a process that converges to a solution of the system of equations. For some program, such as those computing the probability of prefixes of strings from Probabilistic Context Free Grammars, the system is linear, so solving it is even simpler. In general, this provides an approach for performing inference when the number of explanations is infinite but under the generative exclusiveness condition.

In [15] the authors present the algorithm PIP (for Probabilistic Inference Plus), that is able to peform inference even when explanations are not necessarily mutually exclusive and the number of explanations is infinite. They require the programs to be *temporally well-formed*, i.e., that one of the arguments of predicates can be interpreted as a time that grows from head to body. In this case the explanations for an atom can be represented succinctly by Definite Clause Grammars (DCGs). Such DCGs are called *explanation generators* and are used to build Factored Explanation Diagrams (FED) that have a structure that closely follows that of a Binary Decision Diagram (BDD). While the internal nodes of a BDD are Boolean variables, a FED contains two kinds of internal nodes: one representing terminal symbols of explanations (`msws`), and the other representing non-terminal symbols of explanations. FEDs can be used to obtain a system of polynomial equations that is monotonic and thus convergent as in [13, 14]. So, even when the system is non linear, a least solution can be computed to within an arbitrary approximation bound by an iterative procedure.

Turning to approximate inference, we can observe that Monte Carlo sampling approaches such as ProbLog [31] and MCINTYRE [32], while not developed for this purpose, can handle as well goals with an infinite number of possibly infinite explanations. In fact, these approaches work by running the query many times to a program where the probabilistic choices are sampled independently in each run. The proportion of times that the query succeeded gives an estimate of the probability of the query. Each run/derivation corresponds naturally to an explanation and the probability of a run is the same as the probability of the corresponding explanation. The risk is that of incurring an infinite explanation. But infinite explanations have probability 0 so the probability that the computation goes down such a path and does not terminate is 0 as well. As a consequence, Monte Carlo inference can be used on programs with an infinite number of possibly infinite of explanations. As examples of applications of MCINTRYE to these programs see the web system cplint on SWISH [16] that allows the user to query LPADs using exact and approximate inference. cplint on SWISH contains the following examples that have infinite explanations:

- parser for a Probabilistic Context Free Left Recursive Grammar[3];
- prefix parser for PCFGs[4] [14];
- prefix parser for Probabilistic Left-Corner Grammars[5] [14];
- model checker of a Markov chain[6] [15];
- model checker of the Synchronous Leader Election Protocol expressed in PCTL[7] [15];
- model checker for fuzzy formulas in generalized probabilistic logic [8] [15];
- model checker for Recursive Markov Chains expressed in generalized probabilistic logic [9] [15];
- random arithmetic functions [10][33];
- program for handling unknown objects[11] [28, 29];
- game of dice [5][12].

[3]`http://cplint.lamping.unife.it/example/inference/pcfglr.pl`
[4]`http://cplint.lamping.unife.it/example/inference/prefix.pl`
[5]`http://cplint.lamping.unife.it/example/inference/pre_plcg.pl`
[6]`http://cplint.lamping.unife.it/example/inference/markov_chain.pl`
[7]`http://cplint.lamping.unife.it/example/inference/pctl_slep.pl`
[8]`http://cplint.lamping.unife.it/example/inference/gpl.pl`
[9]`http://cplint.lamping.unife.it/example/inference/rmc.pl`
[10]`http://cplint.lamping.unife.it/example/inference/arithm.pl`
[11]`http://cplint.lamping.unife.it/example/inference/var_obj.pl`
[12]`http://cplint.lamping.unife.it/example/inference/threesideddicemc.pl`

Another form of approximate inference achieves a lower bound on the probability of the query by imposing a depth bound on the derivations of an exact inference algorithm computing explanations. In this way, infinite derivations are avoided at the cost of possibly cutting legitimate long explanations. This has been implemented in the PITA exact inference algorithm [34, 35] available in cplint on SWISH. Many of the above examples can be run with this inference approach[13].

9. Conclusions

We have presented a definition of the distribution semantics in terms of an iterated fixpoint operator that allowed us to prove that the semantics is well defined for all programs. In this way we can give a meaning to programs where the queries have an infinite number of possibly infinite explanations. We have shown that there are many interesting problems that benefit from such a feature. We have also discussed approaches for performing inference on these programs. The operator we have presented could also be used for developing an exact forward inference algorithm similarly to [11, 12], complementing recent work on inference [13, 14, 15].

Acknowledgements

Fabrizio Riguzzi is also member of the Gruppo Nazionale per il Calcolo Scientifico of the Istituto Nazionale di Alta Matematica “F. Severi”.

[1] E. Dantsin, Probabilistic logic programs and their semantics, in: Russian Conference on Logic Programming, Vol. 592 of LNCS, Springer, 1991, pp. 152–164.

[2] D. Poole, Logic programming, abduction and probability - a top-down anytime algorithm for estimating prior and posterior probabilities, New Gen. Comp. 11 (3) (1993) 377–400.

[3] T. Sato, A statistical learning method for logic programs with distribution semantics, in: L. Sterling (Ed.), 12th International Conference on Logic Programming, Tokyo, Japan, MIT Press, Cambridge, Massachusetts, 1995, pp. 715–729.

[4] D. Poole, The Independent Choice Logic for modelling multiple agents under uncertainty, Artif. Intell. 94 (1997) 7–56.

[5] J. Vennekens, S. Verbaeten, M. Bruynooghe, Logic programs with annotated disjunctions, in: 20th International Conference on Logic Programming, Vol. 3131 of LNCS, Springer, Berlin Heidelberg, Germany, 2004, pp. 195–209.

[13]`http://cplint.lamping.unife.it/example/inference/pcfglrdb.pl`, `http://cplint.lamping.unife.it/example/inference/markov_chaindb.pl`, `http://cplint.lamping.unife.it/example/inference/var_objdb.pl`

[6] L. De Raedt, A. Kimmig, H. Toivonen, ProbLog: A probabilistic Prolog and its application in link discovery, in: 20th International Joint Conference on Artificial Intelligence, Hyderabad, India (IJCAI-05), Vol. 7, AAAI Press, Palo Alto, California USA, 2007, pp. 2462–2467.

[7] J. Vennekens, M. Denecker, M. Bruynooghe, CP-logic: A language of causal probabilistic events and its relation to logic programming, Theor. Pract. Log. Prog. 9 (3) (2009) 245–308.

[8] T. Sato, Y. Kameya, Parameter learning of logic programs for symbolic-statistical modeling, J. Artif. Intell. Res. 15 (2001) 391–454.

[9] F. Riguzzi, T. Swift, Terminating evaluation of logic programs with finite three-valued models, ACM T. Comput. Log. 15 (4) (2014) 32:1–32:38. doi:10.1145/2629337.

[10] T. Przymusinski, Every logic program has a natural stratification and an iterated least fixed point model, in: ACM Conference on Principles of Database Systems, 1989, pp. 11–21.

[11] J. Vlasselaer, G. Van den Broeck, A. Kimmig, W. Meert, L. De Raedt, Anytime inference in probabilistic logic programs with Tp-compilation, in: Internation Joint Conference on Artificial Intelligence, 2015, pp. 1852–1858.

[12] J. Vlasselaer, G. Van den Broeck, A. Kimmig, W. Meert, L. De Raedt, Tp-compilation for inference in probabilistic logic programs, Int. J. Approx. Reason. in this issue.

[13] T. Sato, P. Meyer, Tabling for infinite probability computation, in: International Conference on Logic Programming, Vol. 17 of LIPIcs, 2012, pp. 348–358.

[14] T. Sato, P. Meyer, Infinite probability computation by cyclic explanation graphs, Theor. Pract. Log. Prog. 14 (2014) 909–937. doi:10.1017/S1471068413000562.

[15] A. Gorlin, C. R. Ramakrishnan, S. A. Smolka, Model checking with probabilistic tabled logic programming, Theor. Pract. Log. Prog. 12 (4-5) (2012) 681–700.

[16] F. Riguzzi, E. Bellodi, E. Lamma, R. Zese, G. Cota, Probabilistic logic programming on the web, Software Pract. and Exper.doi:10.1002/spe.2386.

[17] A. Van Gelder, K. A. Ross, J. S. Schlipf, The well-founded semantics for general logic programs, J. ACM 38 (3) (1991) 620–650.

[18] K. Knopp, Theory and Application of Infinite Series, Dover Books on Mathematics, Dover Publications, 1951.

[19] L. De Raedt, A. Kimmig, Probabilistic (logic) programming concepts, Mach. Learn. 100 (1) (2015) 5–47. `doi:10.1007/s10994-015-5494-z`.

[20] A. N. Kolmogorov, Foundations of the Theory of Probability, Chelsea Publishing Company, New York, 1950.

[21] S. Srivastava, A Course on Borel Sets, Graduate Texts in Mathematics, Springer, 2013.

[22] D. Poole, Abducing through negation as failure: stable models within the independent choice logic, J. Logic Program. 44 (1-3) (2000) 5–35.

[23] D. Poole, Probabilistic horn abduction and Bayesian networks, Artif. Intell. 64 (1) (1993) 81–129.

[24] Y. Chow, H. Teicher, Probability Theory: Independence, Interchangeability, Martingales, Springer Texts in Statistics, Springer, 2012.

[25] P. Cohn, Basic Algebra: Groups, Rings, and Fields, Springer, 2003.

[26] S. Willard, General Topology, Addison-Wesley series in mathematics, Dover Publications, 1970.

[27] L. Steen, J. Seebach, Counterexamples in Topology, Dover Books on Mathematics, Dover Publications, 2013.

[28] B. Milch, B. Marthi, S. J. Russell, D. Sontag, D. L. Ong, A. Kolobov, BLOG: probabilistic models with unknown objects, in: L. P. Kaelbling, A. Saffiotti (Eds.), IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh, Scotland, UK, July 30-August 5, 2005, Professional Book Center, 2005, pp. 1352–1359.

[29] D. Poole, The Independent Choice Logic and beyond, in: L. De Raedt, P. Frasconi, K. Kersting, S. Muggleton (Eds.), Probabilistic Inductive Logic Programming, Vol. 4911 of LNCS, Springer Berlin Heidelberg, 2008, pp. 222–243.

[30] T. Mantadelis, G. Janssens, Nesting probabilistic inference, CoRR`arXiv:abs/1112.3785`.

[31] A. Kimmig, B. Demoen, L. De Raedt, V. S. Costa, R. Rocha, On the implementation of the probabilistic logic programming language problog, Theor. Pract. Log. Prog. 11 (2-3) (2011) 235–262.

[32] F. Riguzzi, MCINTYRE: A Monte Carlo system for probabilistic logic programming, Fund. Inform. 124 (4) (2013) 521–541. `doi:10.3233/FI-2013-847`.

[33] N. D. Goodman, J. B. Tenenbaum. Probabilistic models of cognition [online, cited 15 April 2016].

[34] F. Riguzzi, T. Swift, The PITA system: Tabling and answer subsumption for reasoning under uncertainty, Theor. Pract. Log. Prog. 11 (4–5) (2011) 433–449. `doi:10.1017/S147106841100010X`.

[35] F. Riguzzi, T. Swift, 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) (2013) 279–302. `doi:10.1017/S1471068411000664`.