Online appendix for the paper

Inference and Learning in Probabilistic Logic Programs using Weighted Boolean Formulas

published in Theory and Practice of Logic Programming

DAAN FIERENS, GUY VAN DEN BROECK, JORIS RENKENS,
DIMITAR SHTERIONOV, BERND GUTMANN, INGO THON,
GERDA JANSSENS, LUC DE RAEDT

Department of Computer Science
KU Leuven
Celestijnenlaan 200A, 3001 Heverlee, Belgium
(*e-mail:* FirstName.LastName@cs.kuleuven.be)

submitted 26 June 2012; revised 4 January 2013; accepted 28 January 2014

Appendix A Proofs

In this appendix we give the proofs of Theorems 1 to 3.

Proof of Theorem 1

To prove Theorem 1 we first give necessary some lemma's. We use $pruneInactive(L, \mathbf{E} = \mathbf{e})$ to denote the result of removing from a ground program L all rules that are inactive under the evidence $\mathbf{E} = \mathbf{e}$.

Lemma 1
Let L be a ground normal logic program and let $L' = pruneInactive(L, \mathbf{E} = \mathbf{e})$. For each world/interpretation ω that is consistent with the evidence $\mathbf{E} = \mathbf{e}$ it holds:

a) for each subset A of atoms: A is an unfounded set with respect to ω under program L if and only if it is so under program L', and
b) ω is the well-founded model of L if and only if it is the well-founded model of L'.

Proof:
Part a: We use the notion of *unfounded set* see Definition 3.1 in Van Gelder et al. (1991). We prove both directions of the 'if and only if'.

- If A is an unfounded set with respect to ω under program L, then this also holds under program L':
  The definition of unfounded set imposes a certain condition on each rule in the program whose head is in the set A, we refer to this as the unfounded rule condition. If we know that this condition holds for all such rules in L, then it also holds for all such rules in L', because the latter set of rules is a subset of the former (L' is the result of removing inactive rules from L).

- If A is an unfounded set with respect to ω under program L', then this also holds under program L:
  The 'if' part of this 'if-then' implies that the unfounded rule condition holds for all rules in L', so to prove the 'then' part we only need to show that the unfounded rule condition also holds for all rules in $L \setminus L'$ (i.e., for all rules in L that were removed because of being inactive under the evidence). Every rule $r \in L \setminus L'$ contains at least one atom in its body that is false according to the evidence (that is what made r inactive). Since this lemma applies only to worlds ω that are consistent with the evidence, we have for every such world ω: every rule $r \in L \setminus L'$ contains in its body at least one atom false in ω. This is a sufficient condition to make the rule r an unfounded rule, see condition '1.' in Definition 3.1 in Van Gelder et al. (1991).

Part b: We can now use *Part a* to prove that, for every evidence-consistent world ω, ω is the well-founded model (WFM) of L if and only if it is the WFM of L'.

The WFM is the fixed point of the W_P operator (Van Gelder et al. 1991). For a program L, this operator is defined as $W_L(\omega) = T_L(\omega) \cup \neg U_L(\omega)$, see Definition 3.3 in Van Gelder et al. (1991). We prove below that, for every evidence-consistent world ω: $T_L(\omega) = T_{L'}(\omega)$ and $U_L(\omega) = U_{L'}(\omega)$. Hence, $W_L(\omega) = W_{L'}(\omega)$, hence their fixed points are identical, hence *Part b* holds.

- For every evidence-consistent world ω: $T_L(\omega) = T_{L'}(\omega)$
  L consists of all rules in L' plus some rules that are inactive under the evidence. For each evidence-consistent ω, the bodies of the inactive rules in L are false under ω and hence these rules cannot 'fire'. Hence these rules play no role in the execution of the T_P operator on ω. Hence $T_L(\omega) = T_{L'}(\omega)$.
- For every evidence-consistent world ω: $U_L(\omega) = U_{L'}(\omega)$:
  $U_L(\omega)$ is the greatest unfounded set with respect to (wrt) ω, which is defined as the union of all unfounded sets wrt ω, see Definition 3.2 in Van Gelder et al. (1991). *Part a* says that any subset A of atoms is an unfounded set wrt ω under program L if and only if it is so under program L'. Hence $U_L(\omega) = U_{L'}(\omega)$.

□

Lemma 2
Let L be a ground ProbLog program and let $L' = pruneInactive(L, \mathbf{E} = \mathbf{e})$. Then $MOD_{\mathbf{E}=\mathbf{e}}(L) = MOD_{\mathbf{E}=\mathbf{e}}(L')$.

Proof: $MOD_{\mathbf{E}=\mathbf{e}}(L)$ is defined as the set of all worlds ω that are consistent with the evidence $\mathbf{E} = \mathbf{e}$ and are models of the ProbLog program L, i.e., for which there exists a total choice C and $WFM(C \cup R) = \omega$, with R the rules in L and $WFM()$ the well-founded model. The previous lemma implies that, for every ω consistent with the evidence, removing inactive rules from a given logic program does not alter whether or not ω is the WFM of that program or not. In other words: $\omega \in MOD_{\mathbf{E}=\mathbf{e}}(L)$ if and only if $\omega \in MOD_{\mathbf{E}=\mathbf{e}}(L')$. Hence $MOD_{\mathbf{E}=\mathbf{e}}(L) = MOD_{\mathbf{E}=\mathbf{e}}(L')$. □

Lemma 3

Let L be a ground ProbLog program and let $L' = pruneInactive(L, \mathbf{E} = \mathbf{e})$. Then $P_L(Q \mid \mathbf{E} = \mathbf{e}) = P_{L'}(Q \mid \mathbf{E} = \mathbf{e})$

Proof: We prove the stronger condition $\forall\omega : P_L(\omega \mid \mathbf{E} = \mathbf{e}) = P_{L'}(\omega \mid \mathbf{E} = \mathbf{e})$ The conditional probability $P_L(\omega \mid \mathbf{E} = \mathbf{e})$ of an interpretation ω according to program L is:

- (case1) if $\omega \in MOD_{\mathbf{E}=\mathbf{e}}(L)$ then

  $$P_L(\omega \mid \mathbf{E} = \mathbf{e}) = \frac{P_L(\omega, \mathbf{E} = \mathbf{e})}{P_L(\mathbf{E} = \mathbf{e})}.$$

  Since ω agrees with $\mathbf{E} = \mathbf{e}$, we have that the combined assignment $\omega, \mathbf{E} = \mathbf{e}$ is simply equal to ω. Hence:

  $$P_L(\omega \mid \mathbf{E} = \mathbf{e}) = \frac{P_L(\omega)}{P_L(\mathbf{E} = \mathbf{e})} = \frac{P_L(\omega)}{\sum_{\omega' \in MOD_{\mathbf{E}=\mathbf{e}}(L)} P_L(\omega')}. \tag{A1}$$

- (case2) if $\omega \notin MOD_{\mathbf{E}=\mathbf{e}}(L)$ then $P_L(\omega \mid \mathbf{E} = \mathbf{e}) = 0$.

We now prove that for every ω, $P_L(\omega \mid \mathbf{E} = \mathbf{e}) = P_{L'}(\omega \mid \mathbf{E} = \mathbf{e})$. The proof consists of two parts.

1. We need to prove that we are in case1 under L if and only if we are in case1 under L'. In other words: for every ω: $\omega \in MOD_{\mathbf{E}=\mathbf{e}}(L)$ if and only if $\omega \in MOD_{\mathbf{E}=\mathbf{e}}(L')$. This follows from the previous lemma.
2. We need to prove that if we are in case1 (i.e. if $\omega \in MOD_{\mathbf{E}=\mathbf{e}}(L)$), then the conditional probability given by the fraction in Equation A1 is the same under L as under L'.
   - The numerator is the same under L and L'. This can be seen as follows. For any $\omega \in MOD_{\mathbf{E}=\mathbf{e}}(L)$, the probability $P(\omega)$ is by definition equal to the probability of ω's total choice. The ProbLog programs L and L' differ in their rules, but they have exactly the same probabilistic facts and hence determine the same probability distribution over total choices. Hence $P(\omega)$ is the same under L as under L'.
   - The sum in the denominator is also the same under L and L'. This can be seen as follows. First, the set $MOD_{\mathbf{E}=\mathbf{e}}(L)$ over which the sum ranges is the same under L as under L' because of the above lemma. Second, each term in the sum is the same under L as under L', i.e. for every $\omega \in MOD_{\mathbf{E}=\mathbf{e}}(L)$ the probability $P(\omega)$ is the same under L as under L' (because of the same reasoning as for the numerator).

This concludes the proof. □

Theorem 1
Let L be a ProbLog program and let L_g be the relevant ground program for L with respect to $\mathbf{Q}$ and $\mathbf{E} = \mathbf{e}$. Then $P_L(Q \mid \mathbf{E} = \mathbf{e}) = P_{L_g}(Q \mid \mathbf{E} = \mathbf{e})$.

Proof: It follows from the grounding semantics of ProbLog that replacing the original program L by its full grounding (w.r.t. the Herbrand base) L_{full} preserves the distribution, i.e., $P_L(\mathbf{Q} \mid \mathbf{E} = \mathbf{e}) = P_{L_{full}}(\mathbf{Q} \mid \mathbf{E} = \mathbf{e})$. The relevant ground program

L_g differs from L_{full} only in that it does not contain inactive rules (with respect to $\mathbf{E} = \mathbf{e}$) or irrelevant rules (with respect to $\mathbf{Q} \cup \mathbf{E}$). The lemma above states that removing inactive rules preserves the distribution $P(\mathbf{Q} \mid \mathbf{E} = \mathbf{e})$. Removing irrelevant rules also preserves this distribution; this can be seen as follows. The probability of an atom being true can be determined from all proofs of the atom and the probabilities of the probabilistic facts appearing in these proofs, see De Raedt et al. (2007). Irrelevant rules are - by definition - rules that are not used in any proof of any atom in $\mathbf{Q} \cup \mathbf{E}$. Hence omitting such irrelevant rules does not alter the distribution $P(\mathbf{Q}, \mathbf{E})$. Hence, also the distribution $P(\mathbf{Q} \mid \mathbf{E} = \mathbf{e})$ is preserved because $P(\mathbf{Q} \mid \mathbf{E} = \mathbf{e})$ can be defined in terms of $P(\mathbf{Q}, \mathbf{E})$, i.e., $P(\mathbf{Q} \mid \mathbf{E} = \mathbf{e}) = \frac{P(\mathbf{Q}, \mathbf{E}=\mathbf{e})}{P(\mathbf{E}=\mathbf{e})} = \frac{P(\mathbf{Q}, \mathbf{E}=\mathbf{e})}{\sum_{\mathbf{q}} P(\mathbf{Q}=\mathbf{q}, \mathbf{E}=\mathbf{e})}$.

Proof of Theorem 2

Theorem 2
Let L_g be the relevant ground program for some ProbLog program with respect to $\mathbf{Q}$ and $\mathbf{E} = \mathbf{e}$. Let $MOD_{\mathbf{E}=\mathbf{e}}(L_g)$ be those models in $MOD(L_g)$ that are consistent with the evidence $\mathbf{E} = \mathbf{e}$. Let φ denote the formula and $w(\cdot)$ the weight function of the weighted formula derived from L_g. Then:

- **(model equivalence)** $SAT(\varphi) = MOD_{\mathbf{E}=\mathbf{e}}(L_g)$,
- **(weight equivalence)** $\forall \omega \in SAT(\varphi)$: $w(\omega) = P_{L_g}(\omega)$, i.e., the weight of ω according to $w(\cdot)$ is equal to the probability of ω according to L_g.

Proof: The proof consists of two parts.
Model equivalence. Consider Lemma 1 (Section 5.2). The lemma is about the formula φ_r that captures the rules but not yet the evidence. The lemma states that $SAT(\varphi_r) = MOD(L_g)$. The present theorem is about the formula $\varphi = \varphi_r \wedge \varphi_e$, where φ_e captures the evidence. The effect of adding φ_e to the formula is that all worlds not consistent with the evidence are ruled out. Hence $SAT(\varphi_r \wedge \varphi_e) = MOD_{\mathbf{E}=\mathbf{e}}(L_g)$.
Weight equivalence. Weight equivalence says that the probability of every model (according to L_g) is equal to the weight of the model (according to our weight function $w(\cdot)$). This follows from the way the probability and the weight function are defined.

- The **probability** of a model of a ProbLog program, according to the distribution semantics, is the probability of the underlying total choice, which in turn is defined as the product of probabilities of each of the atomic choices. Formally, the probability of a model ω is:

$$P(\omega) = \prod_{a \in PA^+(\omega)} p(a) \prod_{a \in PA^-(\omega)} p(\neg a)$$
$$= \prod_{a \in PA^+(\omega)} p(a) \prod_{a \in PA^-(\omega)} (1 - p(a)),$$

  with $PA^+(\omega)$ (respectively $PA^-(\omega)$) being the set of all ground probabilistic atoms that are true (resp. false) in ω and $p(\cdot)$ denoting the probability distribution specified by the probabilistic facts.

- The **weight** of a world ω according to our weight function is the product of the weights of all literals l constituting the world/interpretation ω:

$$w(\omega) = \prod_{l \in \omega} w(l) \cdot$$

The literals/atoms in ω fall into four groups: probabilistic atoms that are true in ω (denoted $PA^+(\omega)$, non-probabilistic or derived atoms that are true in ω (denoted $DA^+(\omega)$), and similar for the atoms that are false in ω ($PA^-(\omega)$ and $DA^-(\omega)$). Hence:

$$\begin{aligned} w(\omega) &= \prod_{l \in \omega} w(l) \\ &= \prod_{a \in PA^+(\omega)} w(a) \prod_{a \in PA^-(\omega)} w(\neg a) \prod_{a \in DA^+(\omega)} w(a) \prod_{a \in DA^-(\omega)} w(\neg a) \cdot \end{aligned}$$

By definition of the weight function, the weight of an atom $a \in PA^+(\omega)$ is $p(a)$, the weight of $a \in PA^-(\omega)$ is $1 - p(a)$, the weight of $a \in DA^+(\omega) \cup DA^-(\omega)$ is 1. Hence:

$$\begin{aligned} w(\omega) &= \prod_{a \in PA^+(\omega)} p(a) \prod_{a \in PA^-(\omega)} (1 - p(a)) \prod_{a \in DA^+(\omega)} 1 \prod_{a \in DA^-(\omega)} 1 \\ &= \prod_{a \in PA^+(\omega)} p(a) \prod_{a \in PA^-(\omega)} (1 - p(a)) = P(\omega) \cdot \end{aligned}$$

This proves weight equivalence. □

Proof of Theorem 3

Theorem 3
Let L_g be the relevant ground program for some ProbLog program with respect to $\mathbf{Q}$ and $\mathbf{E} = \mathbf{e}$. Let $\mathcal{M}$ be the corresponding ground MLN. The distribution $P(\mathbf{Q})$ according to $\mathcal{M}$ is the same as the distribution $P(\mathbf{Q} \mid \mathbf{E} = \mathbf{e})$ according to L_g.

Proof: We prove that **(1)** the set of worlds with non-zero probability according to the MLN is the same as the set of worlds with non-zero probability according to the ProbLog program *and* the evidence; **(2)** for every such world ω, $P_{\mathcal{M}}(\omega) = P_L(\omega \mid \mathbf{E} = \mathbf{e})$
(Part 1) A world has non-zero probability according to an MLN if it satisfies all hard clauses in the MLN. The hard clauses in the MLN are the same as the clauses in the weighted formula φ. Hence the set of worlds with non-zero probability according to the MLN equals $SAT(\varphi)$. Theorem 2 (model equivalence) implies that this set equals $MOD_{\mathbf{E}=\mathbf{e}}(L_g)$, which is exactly the set of worlds with non-zero probability according to the ProbLog program and the evidence.
(Part 2) The probability of a world $\omega \in SAT(\varphi)$ according to an MLN is defined as $P_{\mathcal{M}}(\mathbf{Q}) = W(\omega)/Z$, with $W(\omega)$ the product of exponentiated weights of the soft clauses satisfied in ω, and Z the normalization constant. The probability of ω according to the ProbLog program conditioned on the evidence is $P_L(\omega|\mathbf{E} =$

$\mathbf{e}) = P_L(\omega)/P_L(\mathbf{E} = \mathbf{e})$. We now show that both expressions are the same (i.e. $W(\omega)/Z = P_L(\omega)/P_L(\mathbf{E} = \mathbf{e})$).

- The numerators are the same ($W(\omega) = P_L(\omega)$): The only soft clauses in the MLN are unit clauses, whose weights are derived from the probabilistic facts. The unit clauses are such that, for any given world ω, there is one unit clause per probabilistic atom that is satisfied. $W(\omega)$ is the product of the exponentiated weights of all these clauses. It follows from the way these weights are defined in terms of the weighted formula, and from weight equivalence between the weighted formula and the ProbLog program (Theorem 2), that this product is equal to the probability of the total choice of ω according to the ProbLog program and hence to the numerator $P_L(\omega)$.
- The denominators are the same ($Z = P_L(\mathbf{E} = \mathbf{e})$): The normalization constant Z of the MLN is defined as $\sum_{\omega \in SAT(\varphi)} W(\omega)$. The evidence probability $P_L(\mathbf{E} = \mathbf{e})$ equals $\sum_{\omega \in MOD_{\mathbf{E}=\mathbf{e}}(L)} P(\omega)$. These sums are equal since (a) the sets over which they range are equal due to Theorem 2 (model equivalencce), (b) the summed terms are equal (because of the same reasoning as for the numerator).

This concludes the proof. □

Appendix B Markov Logic

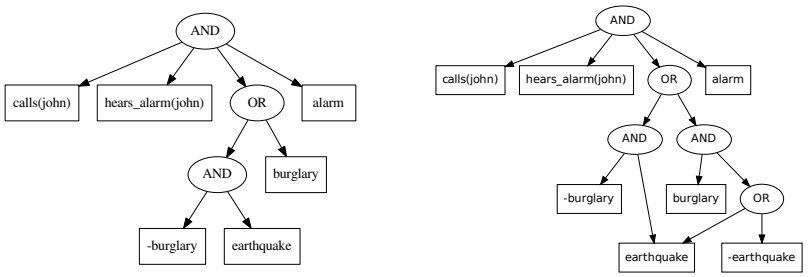
We briefly review Markov Logic (Domingos et al. 2008). While Markov Logic generally works with FOL formulas, we consider only the ground case, as this is sufficient for our paper.

A *Markov Logic Network (MLN)* consists of two parts: a set of 'soft' formulas f_i, which each have an associated weight $w_i \in \mathbb{R}$, and a set of 'hard' formulas. An MLN determines a probability distribution on the set of possible worlds (determined by the Herbrand base). The probability of a world ω is 0 if it violates some hard formula and is $\frac{1}{Z} e^{\sum_i w_i \delta_i(\omega)}$ otherwise, where the sum is over all soft formulas and $\delta_i(\omega)$ is the indicator function being 1 if the soft formula f_i is true in world ω and 0 otherwise. Note that the exponent $\sum_i w_i \delta_i(\omega)$ is the sum of weights of satisfied soft formulas in world ω; the higher this sum, the more likely ω is. The name 'MLN' comes from the fact that this probability distribution can also be written as the distribution of a Markov network.

Appendix C The need for smoothing d-DNNFs

The algorithm we use to compute marginal probabilities requires a smooth d-DNNF. A smooth d-DNNF is a d-DNNF where for every disjunction node all children use exactly the same set of atoms. That is, if $C_1 \cdot \cdot C_n$ are the children of an *OR* node C, then $Atoms(C_i) = Atoms(C_j)$, for $i \neq j$, where $Atoms(C_i)$ is the set of atoms which C_i uses.

Figure C 1a shows a non-smooth d-DNNF, while Figure C 1b shows the corresponding smooth d-DNNF (which we have already shown before in Figure 1.a of



(a) A non smooth d-DNNF (b) A smooth d-DNNF

Fig. C 1. A d-DNNF and the corresponding smooth d-DNNF for the formula $burglary \vee earthquake$.

the paper but we repeat here for convenience). Consider the non-smooth d-DNNF. The `OR` node doesn't satisfy smoothness, since the sets of atoms of its children differ ($\{burglary, earthquake\}$ and $\{burglary\}$; the negation is ignored here). Hence we need to transform this d-DNNF into a smooth d-DNNF, see Figure C 1b. This is done by substituting the `burglary` node by an `AND` node, then adding the `burglary` node as a child to the new `AND` node and creating a new smoothing node for the missing atom `-earthquake`. The smoothing node is an `OR` node which links to `earthquake` and `-earthqake`. It is then linked to the `AND` node.

Let us illustrate how smoothness affects the computation of probabilities using our Alarm running example (so not the restricted version of the example considered above). We have seen the arithmetic circuit (AC) corresponding to the smooth d-DNNF for this example before, recall Figure 2 on p. 22 of the paper. This figure also illustrates how we can compute the probability of the conjunction $P(earthquake = true \wedge calls(john) = true)$. This yields the value 0.14, which is indeed the correct value. In contrast, Figure C 2 shows the same evaluation process on an AC for the non-smooth d-DNNF. This results in an incorrect value (0.196). This shows the need for smoothness of the d-DNNF.

Appendix D Kullback-Leibler Divergence Between ProbLog Programs

The Kullback-Leibler divergence $D(P||Q)$ is a non-symmetric measure for the difference of two probability distributions P and Q (cf. Wasserman (2003)). It is used in probability theory as well as in information theory where it is also known as information gain. The K-L divergence aggregates the difference of the two distributions on all elements of the outcome space. It is only defined if the support of Q is larger than the one of P, that is, for all i where $P(i) > 0$ also $Q(i) > 0$.

We use the K-L divergence to evaluate the LFI-ProbLog learning algorithm (cf. Algorithm 1 of the paper) and measure how close the learned program T_2 is to the ground truth program T_1. We are doing parameter estimation, that is, the structure of the program is fixed and only the fact probabilities change. Hence

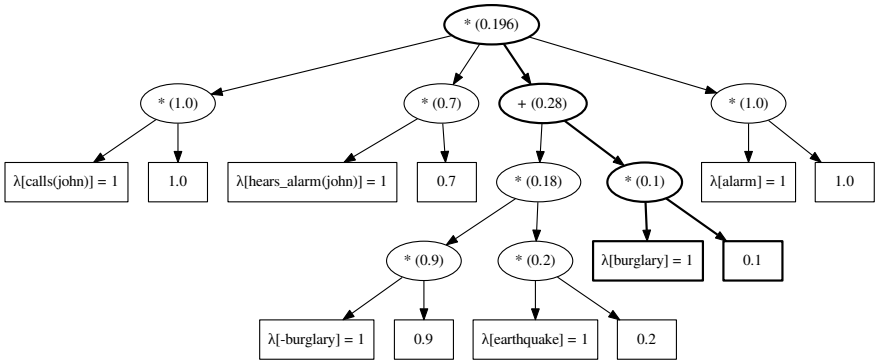


Fig. C 2. The arithmetic circuit corresponding to the non-smooth d-DNNF for the Alarm example.

we can restrict the definition of the K-L divergence to programs that are identical except for the fact probabilities.

Definition 1 (*K-L Divergence*)
Let $T_1 = F_1 \cup R$ and $T_2 = F_2 \cup R$ be ground ProbLog programs such that the probabilistic facts are identical except for the probabilities, that is, $F_1 = \{p_i :: f_i | 1 \leq i \leq n\}$ and $F_2 = \{q_i :: f_i | 1 \leq i \leq n\}$. Let At denote the Herbrand base of T_1 and T_2 (note that they have the same Herbrand base). We denote interpretations as subsets of atoms, i.e., $L \subseteq At$ is the interpretation in which the atoms that are in L are true and the other atoms are false. Then the K-L Divergence between T_1 and T_2 is defined as

$$D(T_1 || T_2) = \sum_{L \subseteq At} P_{T_1}(L) \log \frac{P_{T_1}(L)}{P_{T_2}(L)} \tag{D1}$$

There are exponentially many interpretations $L \subseteq At$, which makes evaluating the K-L divergence as defined above impossible in practice. However, the probabilistic facts in a ProbLog program are independent, which can be exploited to compute the K-L divergence in linear time by looping once over F.

Theorem 4
Let $T_1 = F_1 \cup R$ and $T_2 = F_2 \cup R$ be ground ProbLog programs such that the probabilistic facts are identical except for the probabilities, that is, $F_1 = \{p_i :: f_i | 1 \leq i \leq n\}$ and $F_2 = \{q_i :: f_i | 1 \leq i \leq n\}$. Then the K-L Divergence between T_1 and T_2 can be calculated as

$$D(T_1 || T_2) = \sum_{i=1}^{n} \left(p_i \log \frac{p_i}{q_i} + (1 - p_i) \log \frac{1 - p_i}{1 - q_i} \right) \quad . \tag{D2}$$

It is possible to extend the K-L divergence and the theorem to non-ground facts. To do so, one needs to multiply each summand $p_i \log \frac{p_i}{q_i} + (1 - p_i) \log \frac{1-p_i}{1-q_i}$ with the number of ground instances of the probabilistic fact f_i.

Proof
We prove Theorem 4 by induction over the number of probabilistic facts.

Base case $n = 1$.

$$\begin{aligned}
D(T_1||T_2) &= \sum_{L\subseteq At} P_{T_1}(L)\log\frac{P_{T_1}(L)}{P_{T_2}(L)} \\
&= P_{T_1}(\{f_1\})\log\frac{P_{T_1}(\{f_1\})}{P_{T_2}(\{f_1\})} \\
&\quad + P_{T_1}(\emptyset)\log\frac{P_{T_1}(\emptyset)}{P_{T_2}(\emptyset)} \\
&= p_1\log\frac{p_1}{q_1} + (1-p_1)\log\frac{1-p_1}{1-q_1} \\
&= \sum_{i=1}^{n}\left(p_i\log\frac{p_i}{q_i} + (1-p_i)\log\frac{1-p_i}{1-q_i}\right)
\end{aligned}$$

Inductive case $n \to n+1$. To simplify the notation, we define $T_1^{n+1} = T_1 \cup \{p_{n+1} :: f_{n+1}\}$ and $T_2^{n+1} = T_2 \cup \{q_{n+1} :: f_{n+1}\}$

$$\begin{aligned}
&D(T_1^{n+1}||T_2^{n+1}) \\
&= \sum_{L\subseteq(At\cup\{f_{n+1}\})} P_{T_1^{n+1}}(L)\log\frac{P_{T_1^{n+1}}(L)}{P_{T_2^{n+1}}(L)} \\
&= \left[\sum_{L\subseteq At} P_{T_1^{n+1}}(L\cup\{f_{n+1}\})\log\frac{P_{T_1^{n+1}}(L\cup\{f_{n+1}\})}{P_{T_2^{n+1}}(L\cup\{f_{n+1}\})}\right] + \\
&\qquad \left[\sum_{L\subseteq At} P_{T_1^{n+1}}(L)\log\frac{P_{T_1^{n+1}}(L)}{P_{T_2^{n+1}}(L)}\right]
\end{aligned}$$

Probabilistic facts are independent and thus we can factorize the probabilities

$$\begin{aligned}
&= \left[\sum_{L\subseteq At} p_{n+1}\cdot P_{T_1}(L)\log\frac{p_{n+1}\cdot P_{T_1}(L)}{q_{n+1}\cdot P_{T_2}(L)}\right] + \\
&\qquad \left[\sum_{L\subseteq At} (1-p_{n+1})\cdot P_{T_1}(L)\log\frac{(1-p_{n+1})\cdot P_{T_1}(L)}{(1-q_{n+1})\cdot P_{T_2}(L)}\right]
\end{aligned}$$

using the rules for log and factoring out the constants

$$\begin{aligned}
&= p_{n+1}\left[\sum_{L\subseteq At} P_{T_1}(L)\left(\log\tfrac{p_{n+1}}{q_{n+1}} + \log\tfrac{P_{T_1}(L)}{P_{T_2}(L)}\right)\right] + \\
&\qquad (1-p_{n+1})\left[\sum_{L\subseteq At} P_{T_1}(L)\left(\log\tfrac{1-p_{n+1}}{1-q_{n+1}} + \log\tfrac{P_{T_1}(L)}{P_{T_2}(L)}\right)\right]
\end{aligned}$$

expanding the inner sums and factoring out constants

$$\begin{aligned}
&= p_{n+1}\left(\log\tfrac{p_{n+1}}{q_{n+1}}\right)\left[\sum_{L\subseteq At} P_{T_1}(L)\right] + \\
&\qquad p_{n+1}\left[\sum_{L\subseteq At} P_{T_1}(L)\left(\log\tfrac{P_{T_1}(L)}{P_{T_2}(L)}\right)\right] +
\end{aligned}$$

$$(1-p_{n+1})\left(\log\tfrac{1-p_{n+1}}{1-q_{n+1}}\right)\left[\sum_{L\subseteq At}P_{T_1}(L)\right]+$$
$$(1-p_{n+1})\left[\sum_{L\subseteq At}P_{T_1}(L)\left(\log\tfrac{P_{T_1}(L)}{P_{T_2}(L)}\right)\right]$$
since $\sum_{L\subseteq At}P_{T_1}(L)$ is 1, rearranging yields
$$= \quad p_{n+1}\left(\log\tfrac{p_{n+1}}{q_{n+1}}\right)+(1-p_{n+1})\left(\log\tfrac{1-p_{n+1}}{1-q_{n+1}}\right)+$$
$$\sum_{L\subseteq At}P_{T_1}(L)\log\tfrac{P_{T_1}(L)}{P_{T_2}(L)}$$
using the inductive assumption
$$= \quad p_{n+1}\left(\log\tfrac{p_{n+1}}{q_{n+1}}\right)+(1-p_{n+1})\left(\log\tfrac{1-p_{n+1}}{1-q_{n+1}}\right)+$$
$$\sum_{i=1}^{n}\left(p_i\log\tfrac{p_i}{q_i}+(1-p_i)\log\tfrac{1-p_i}{1-q_i}\right)$$
rearranging the terms
$$= \quad \sum_{i=1}^{n+1}\left(p_i\log\tfrac{p_i}{q_i}+(1-p_i)\log\tfrac{1-p_i}{1-q_i}\right)$$

□

References

De Raedt, L., Kimmig, A., and Toivonen, H. 2007. Problog: a probabilistic prolog and its application in link discovery. In *In Proceedings of 20th International Joint Conference on Artificial Intelligence*. AAAI Press, 2468–2473.

Domingos, P., Kok, S., Lowd, D., Poon, H., Richardson, M., and Singla, P. 2008. *Probabilistic Inductive Logic Programming - Theory and Applications*. Lecture Notes in Computer Science. Springer, Chapter 'Markov Logic'.

Van Gelder, A., Ross, K. A., and Schlipf, J. S. 1991. The well-founded semantics for general logic programs. *Journal of the ACM 38,* 3, 620–650.

Wasserman, L. 2003. *All of Statistics: A Concise Course in Statistical Inference (Springer Texts in Statistics)*. Springer.