Logic Programming Techniques for Reasoning with Probabilistic Ontologies

Riccardo Zese, Elena Bellodi, Evelina Lamma and Fabrizio Riguzzi

University of Ferrara, Italy

riccardo.zese@unife.it
Outline

1. Introduction
2. Representing Uncertainty
3. Probabilistic Ontologies under the DISPONTE semantics
4. BUNDLE
5. From BUNDLE to TRILL
6. TRILL
7. TRILL\(^P\)
8. TRILL-on-SWISH
9. Experiments
10. Conclusions
Semantic Web
- Aims at making information available in a form that is understandable by machines
- Web Ontology Language (OWL)
  - Based on Description Logics

Reasoners
- Most DL reasoners use a tableau algorithm for doing inference
- Most of them are implemented in a procedural language
  - Example: Pellet, RacerPro, FaCT++
Uncertainty Representation

**Semantic Web**
- Incompleteness or uncertainty are intrinsic of much information on the World Wide Web
- Most common approaches: probability theory, Fuzzy Logic

**Logic Programming**
- Uncertain relationships among entities characterize many complex domains
- Most common approach: probability theory → **Distribution Semantics** [Sato, 1995].
  - It underlies many languages (ICL, PRISM, ProbLog, LPADs),...
  - They define a probability distribution over normal logic programs, called worlds
  - The distribution is extended to a joint distribution over worlds and queries
  - The probability of a query is obtained from this distribution by summing out worlds
Idea: **annotate axioms of an ontology with a probability** and assume that the axioms are pairwise independent

\[ 0.6 :: \text{Cat} \sqsubseteq \text{Pet} \]

A probabilistic ontology defines thus a distribution over normal theories (worlds) obtained by including an axiom in a world with a probability given by the annotation.
Atomic choice: a pair \((E_i, k)\), where \(E_i\) is the \(i\)th probabilistic axiom and \(k \in \{0, 1\}\) indicates whether \(E_i\) is chosen to be included in a world \((K = 1)\) or not \((K = 0)\).

Selection \(\sigma\): set of one atomic choice for each probabilistic axiom.

\(\sigma\) identifies a world \(w_\sigma\)

\[P(w_\sigma) = \prod_{(E_i,1) \in \sigma} p_i \prod_{(E_i,0) \in \sigma} (1 - p_i)\]

Probability of a query \(Q\) given a world \(w\): \(P(Q | w) = 1\) if \(w \models Q\), 0 otherwise

Probability of \(Q\)

\[P(Q) = \sum_w P(Q,w) = \sum_w P(Q | w)P(w) = \sum_{w: w \models Q} P(w)\]
Inference and Query answering

- The probability of a query $Q$ can be computed according to the distribution semantics by first finding the explanations for $Q$ in the knowledge base.

- **Explanation**: subset of axioms of the KB that is sufficient for entailing $Q$.

- All the explanations for $Q$ must be found, corresponding to all ways of proving $Q$.

- Probability of $Q$ → probability of the DNF formula

  $$ F(Q) = \bigvee_{e \in E_Q} \bigwedge_{F_i \in e} X_i $$

  where $E_Q$ is the set of explanations and $X_i$ as a Boolean random variable associated to axiom $F_i$.

- We exploit Binary Decision Diagrams for efficiently computing the probability of a DNF formula.
BUNDLE performs inference over DISPONTE knowledge bases

It exploits an underlying ontology reasoner able to return all explanations for a query, such as *Pellet* [Sirin et al., 2007].

Explanations for a query in the form of a *set of sets of axioms*

BUNDLE uses a *tableau algorithm*

Each tableau expansion rule updates a *tracing function* $\tau$, which associates sets of axioms with nodes and edges of the tableau.
Non-determinism

- **Problem**: some tableau expansion rules are non-deterministic
  - Reasoners implement a search strategy in a or-branching space
  - We want to find all the possible explanations for a query
    - The algorithm has to explore all the non-deterministic choices done
Why Prolog?

- The reasoners implemented using procedural languages have to implement also a backtracking algorithm to find all the possible explanations
  - Example: Pellet uses an hitting set algorithm that repeatedly removes an axiom from the KB and then computes again a new explanation
- Reasoners written in Prolog can exploit Prolog’s backtracking facilities for performing the search
TRILL implements the tableau algorithm using Prolog

It resolves the axiom pinpointing problem in which we are interested in the set of explanations that entail a query

Thea2 library for converting OWL DL ontologies to Prolog:
  - each OWL axiom is translated into a Prolog fact

It applies all the possible expansion rules, first the non-deterministic ones then the deterministic ones

It returns the set of the explanations
Deterministic rules are implemented by predicates that take as input a tableau and return a new single tableau.

Non-deterministic rules are implemented by predicates that take as input a tableau and return a list of tableaux from which one is non-deterministically chosen.
TRILL$^P$ resolves the axiom pinpointing problem by computing a *pinpointing formula*.

[Baader and Peñaloza, 2010a, Baader and Peñaloza, 2010b]

1. We associate a Boolean variable to each axiom of the KB.
2. The pinpointing formula is a monotone Boolean formula on these variables that compactly encodes the set of all explanations.
TRILL$^P$ - Tableau Reasoner for description Logics in proLog exploiting Pinpointing formula

- Deterministic and non-deterministic rules are implemented in the same way of TRILL’s expansion rules
- They associate a pinpointing formula to the labels of the nodes instead of a set of explanations
Computing the probability

- The pinpointing formula is a Boolean formula which can be directly translated into a BDD.
- We can compute the probability of the query from the BDD as in BUNDLE.
Example

\[
F_1 = \text{fluffy} : \text{Cat} \\
F_2 = \text{tom} : \text{Cat} \\
0.6 :: F_3 = \text{Cat} \sqsubseteq \text{Pet} \\
0.5 :: F_4 = \exists \text{hasAnimal}. \text{Pet} \sqsubseteq \text{NatureLover} \\
F_5 = (\text{kevin}, \text{fluffy}) : \text{hasAnimal} \\
F_6 = (\text{kevin}, \text{tom}) : \text{hasAnimal}
\]

Let \( Q = \text{kevin} : \text{NatureLover} \) be the query,
the set of explanations is \( \{ \{ F_5, F_1, F_3, F_4 \}, \{ F_6, F_2, F_3, F_4 \} \} \),
the pinpointing formula is \( ((F_5 \land F_1) \lor (F_6 \land F_2)) \land F_3 \land F_4 \).
the probability is \( P = 0.3 \)
A Web Interface for TRILL: TRILL-on-SWISH

- SWISH [Lager and Wielemaker, 2014]
  - a recently proposed Web framework for logic programming
  - based on various features and packages of SWI-Prolog
  - allows the user to write Prolog programs and ask queries in the browser

- TRILL-on-SWISH allows users to write a KB in the RDF/XML format directly in the web page or load it from a URL, and specify queries that are answered by TRILL running on the server.

- Available at http://trill.lamping.unife.it.
<xml version="1.0">

This knowledge base is inspired by the people+pets ontology from
Patel-Schneider, P. F., Horrocks, I., and Bechhofer, S. 2003. Tutorial on OWL.
The knowledge base indicates that the individuals that own an animal which is a pet are:


Doctoral Consortium of the 30th International Conference on Logic Programming (ICLP 2014)

/** <examples>
13  ?- prob_instanceOf('natureLover', 'Kevin', Prob).
14  ?- instanceOf('natureLover', 'Kevin', ListExpl).

*/
</xml>

Experiments

• Comparison between TRILL, TRILL$^P$ and BUNDLE

• We consider four datasets:
  1. BRCA that models the risk factor of breast cancer;
  2. An extract of DBPedia;
  3. Biopax level 3 that models metabolic pathways;
  4. Vicodi that contains information on European history.

Table: Average time for computing the probability of queries in seconds.

<table>
<thead>
<tr>
<th>DATASET</th>
<th>TRILL TIME (S)</th>
<th>TRILL$^P$ TIME (S)</th>
<th>BUNDLE TIME (S)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BRCA</td>
<td>27.87</td>
<td>4.74</td>
<td>6.96</td>
</tr>
<tr>
<td>DBPedia</td>
<td>51.56</td>
<td>4.67</td>
<td>3.79</td>
</tr>
<tr>
<td>Biopax level 3</td>
<td>0.12</td>
<td>0.12</td>
<td>1.85</td>
</tr>
<tr>
<td>Vicodi</td>
<td>0.19</td>
<td>0.19</td>
<td>1.12</td>
</tr>
</tbody>
</table>
Conclusions

- We presented a semantics for modeling probabilistic DL KBs
- We presented three reasoners which can compute the probability of queries under the DISPONTE semantics
- We presented a web interface for TRILL, one of the reasoners presented in the paper
- The results we obtained show that:
  1. Prolog is a viable language for implementing DL reasoning algorithms
  2. TRILL’s and TRILL^P’s performances are comparable with those of a state-of-art reasoner
Thanks.

Questions?
References I

Automata-based axiom pinpointing.

Axiom pinpointing in general tableaux.

Pengines: Web logic programming made easy.
*TPLP*, 14(4-5):539–552.

A statistical learning method for logic programs with distribution semantics.

Pellet: A practical OWL-DL reasoner.