

# 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



UNIVERSITÀ  
DEGLI STUDI  
DI FERRARA  
- EX LABORE FRUCTUS -

# Outline

- 1 Introduction
- 2 Representing Uncertainty
- 3 Probabilistic Ontologies under the DISPONTE semantics
- 4 BUNDLE
- 5 From BUNDLE to TRILL
- 6 TRILL
- 7 TRILL<sup>P</sup>
- 8 TRILL-on-SWISH
- 9 Experiments
- 10 Conclusions

# Introduction

- 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

# DISPONTE: DIstribution Semantics for Probabilistic ONTologiEs

- Idea: **annotate axioms of an ontology with a probability** and assume that the axioms are pairwise independent

$$0.6 :: \textit{Cat} \sqsubseteq \textit{Pet}$$

- 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

# DISPONTE

- **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  $Q \rightarrow$  probability of the DNF formula

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

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

## Binary decision diagrams for Uncertain reasoning on Description Logic theories

- 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 - Tableau Reasoner for description Logics in proLog

- 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

# TRILL - Tableau Reasoner for description Logics in proLog

- 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<sup>P</sup> - Tableau Reasoner for description Logics in proLog exploiting Pinpointing formula

- TRILL<sup>P</sup> 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<sup>P</sup> - 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

# TRILL<sup>P</sup> - Tableau Reasoner for description Logics in proLog

- Example

$$F_1 = \text{fluffy} : \text{Cat}$$

$$F_2 = \text{tom} : \text{Cat}$$

$$0.6 :: F_3 = \text{Cat} \sqsubseteq \text{Pet}$$

$$0.5 :: F_4 = \exists \text{hasAnimal.Pet} \sqsubseteq \text{NatureLover}$$

$$F_5 = (\text{kevin}, \text{fluffy}) : \text{hasAnimal}$$

$$F_6 = (\text{kevin}, \text{tom}) : \text{hasAnimal}$$

- 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  $((F_5 \wedge F_1) \vee (F_6 \wedge F_2)) \wedge F_3 \wedge F_4$ .
- 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>.

## TRILL-on-SWISH

TRILL on SWISH – SWISH


trill.lamping.unife.it/trill\_on\_swish/tos\_example/people+pets.owl

TRILL on SWISH File Edit Examples Help Search

```

1 <?xml version="1.0"?>
2
3 <!--
4
5 This knowledge base is inspired by the people+pets ontology from
6 Patel-Schneider, P, F., Horrocks, I., and Bechhofer, S. 2003. Tutorial on OWL.
7 The knowledge base indicates that the individuals that own an animal which is a pet are
8 Zese, R.: Reasoning with Probabilistic Logics. ArXiv e-prints 1405.0915v3.
9 Doctoral Consortium of the 30th International Conference on Logic Programming (ICLP 2014)
10
11 /** <examples>
12
13 ?- prob_instanceOf('natureLover','Kevin',Prob).
14 ?- instanceOf('natureLover','Kevin',ListExpl).
15
16 */
17 -->
18
19 <!DOCTYPE rdf:RDF [
20
21 <ENTITY owl "http://www.w3.org/2002/07/owl#" >
22 <ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
23 <ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
24 <ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
25 <ENTITY dispointe "https://sites.google.com/a/unife.it/ml/dispointe" >
26
27 ]>
28
29 <rdf:RDF xmlns="http://cohse.semanticweb.org/ontologies/people#"
30 xml:base="http://cohse.semanticweb.org/ontologies/people#"
31 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
32 xmlns:owl="http://www.w3.org/2002/07/owl#"
33 xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
34 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
35 xmlns:dispointe="https://sites.google.com/a/unife.it/ml/dispointe">
36 <owl:Ontology rdf:about="http://cohse.semanticweb.org/ontologies/people"/>
37

```



```

prob_instanceOf('natureLover','Kevin',Prob).
Prob = 0.3

?- prob_instanceOf('natureLover','Kevin',Prob).

```

Examples History Clear table results Run!

# Experiments

- Comparison between TRILL, TRILL<sup>P</sup> 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.

DATASET	TRILL TIME (S)	TRILL <sup>P</sup> TIME (S)	BUNDLE TIME (S)
BRCA	27.87	4.74	6.96
DBPedia	51.56	4.67	3.79
Biopax level 3	0.12	0.12	1.85
Vicodi	0.19	0.19	1.12

# 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<sup>P</sup>'s performances are comparable with those of a state-of-art reasoner

Thanks.  
Questions?

# References I

-  Baader, F. and Peñaloza, R. (2010a).  
Automata-based axiom pinpointing.  
*J. Autom. Reasoning*, 45(2):91–129.
-  Baader, F. and Peñaloza, R. (2010b).  
Axiom pinpointing in general tableaux.  
*J. Log. Comput.*, 20(1):5–34.
-  Lager, T. and Wielemaker, J. (2014).  
Penguins: Web logic programming made easy.  
*TPLP*, 14(4-5):539–552.
-  Sato, T. (1995).  
A statistical learning method for logic programs with distribution semantics.  
In *ICLP 1995*, pages 715–729. MIT Press.
-  Sirin, E., Parsia, B., Cuenca-Grau, B., Kalyanpur, A., and Katz, Y. (2007).  
Pellet: A practical OWL-DL reasoner.  
*J. Web Sem.*, 5(2):51–53.