Computing Inferences for Credal $\mathcal{ALC}$ Terminologies

Rodrigo B. Polastro[a], Fabio G. Cozman[a],
Felipe I. Takiyama[a], and Kate C. Revoredo[b]

[a]Univ. de São Paulo - Av. Prof. Mello Moraes 2231, São Paulo, SP - Brazil
[b]Dept. Informática Aplicada, Unirio - Av. Pasteur, 458, Rio de Janeiro, RJ - Brazil
rodrigopolastro@gmail.com, fgcozman@usp.br, felipe.takiyama@usp.br,
katerevoredo@gmail.com

Abstract. We describe a package that performs inferences for the probabilistic description logic CR$\mathcal{ALC}$: given a terminology consisting of a set of sentences in CR$\mathcal{ALC}$, and a set of assertions, the package computes the probability of additional assertions using an approximate variational method. We briefly review the essentials of CR$\mathcal{ALC}$, mention some recent applications, and describe the package. We then describe our current efforts to incorporate lifted inference into the package.

1 Introduction

This paper focuses on a particular probabilistic description logic, Credal $\mathcal{ALC}$ (referred to as CR$\mathcal{ALC}$). This logic adds some probabilistic operators to the popular logic $\mathcal{ALC}$ [1] and combines these operators with independence assumptions inspired by the theory of relational Bayesian networks [4]. One can see CR$\mathcal{ALC}$ as a language to express ontologies with probabilistic assessments, or simply as a language to describe relational Bayesian networks. Applications in mobile robotics [2], automatic construction of ontologies [7], and analysis of social networks [8] have benefited from the use of CR$\mathcal{ALC}$, often coupled with machine learning techniques. We summarize the main features of CR$\mathcal{ALC}$, and some of its applications, in Section 2. Alas, so far there has been no simple way to produce inferences in CR$\mathcal{ALC}$ — here an *inference* means the computation of a probability value for a given assertion conditional on other observed assertions, using a probabilistic terminology as background knowledge.

In Section 3 we introduce a package, coded by the first author, that accepts sentences and assertions in CR$\mathcal{ALC}$, and that produces inferences using an approximate variational algorithm. We then discuss our current efforts in developing exact *lifted* inference methods that can be added to the package.

2 cr$\mathcal{ALC}$: A summary, and applications

As usual with description logics, we have *individuals*, *concepts*, and *roles*. The semantics is given by a *domain* $\mathcal{D}$ (a set that we assume *finite* in this paper) and an *interpretation* $\cdot^{\mathcal{I}}$ (a functor). Each *concept* is interpreted as a subset of a domain $\mathcal{D}$. Each *role* is interpreted as a binary relation on the domain.

Many probabilistic descriptions logics have appeared in the literature [6]. Several consider probabilities over the interpretations. For example, one interprets $P(\mathsf{Professor}(\mathsf{John})) = 0.001$ as assigning 0.001 to be the probability of the set of interpretations where John is a $\mathsf{Professor}$. The logic CR$\mathcal{ALC}$ is a probabilistic extension of the description logic $\mathcal{ALC}$ that adopts such an interpretation-based semantics [3]. It keeps all constructors of $\mathcal{ALC}$, but only allows concept names on the left hand side of inclusions/definitions. Additionally, in CR$\mathcal{ALC}$ one can have probabilistic inclusions such as $P(C|D) = \alpha$ or $P(r) = \beta$ for concepts C and D, and for role r. If the interpretation of D is the whole domain, then we simply write $P(C) = \alpha$. The semantics of these inclusions is roughly (a formal definition can be found in [3]) given by:

$$\forall x \in \mathcal{D} \ : \ P(C(x)|D(x)) = \alpha, \qquad \forall x \in \mathcal{D}, y \in \mathcal{D} \ : \ P(r(x,y)) = \beta.$$

We assume that every terminology is acyclic; no concept uses itself. This assumption allows one to represent any terminology $\mathcal{T}$ through a directed acyclic graph. Such a graph, denoted by $\mathcal{G}(\mathcal{T})$, has each concept name and role name as a node, and if a concept C directly uses concept D, then D is a *parent* of C in $\mathcal{G}(\mathcal{T})$. Each existential restriction $\exists r.C$ and value restriction $\forall r.C$ is added to the graph $\mathcal{G}(\mathcal{T})$ as nodes, with an edge from r and C to each restriction directly using it. Each restriction node is a *deterministic* node in that its value is completely determined by its parents. We then assume a Markov condition on this graph, similar to the Markov condition on Bayesian networks; with a few additional assumptions concerning uniqueness of names and values, this guarantees that any probability distribution over interpretations factorizes as a Bayesian network over grounded concepts and roles [3].

Inferences, such as $P(\mathsf{A}_o(\mathsf{a}_0)|\mathcal{A})$ for an ABox $\mathcal{A}$, can be computed by grounding a set of sentences into a possibly large Bayesian network. As this may be too complex in practice, an alternative is to run approximate schemes, for instance schemes based on approximate variational approximations [3].

Recent work has explored the use of probabilistic terminologies in CR$\mathcal{ALC}$ in several applications [2, 7, 8]. These applications require the computation of many inferences; thus it is important to have a package that can perform inference in CR$\mathcal{ALC}$ terminologies.

3 A package

This section describes a software package that handles CR$\mathcal{ALC}$ terminologies and assertions, and that produces inferences (either by producing relational Bayesian networks that can be further processed, or by running approximate variational inference). The package has been coded by the first author using the Java language, and can work either from the command prompt or through a graphical user interface (depicted in Figure 1).

The first design decision was the input language. We have chosen to adapt the Knowledge Representation System Specification (KRSS). The standard complete specification of KRSS can be found at http://dl.kr.org/krss-spec.ps. We use

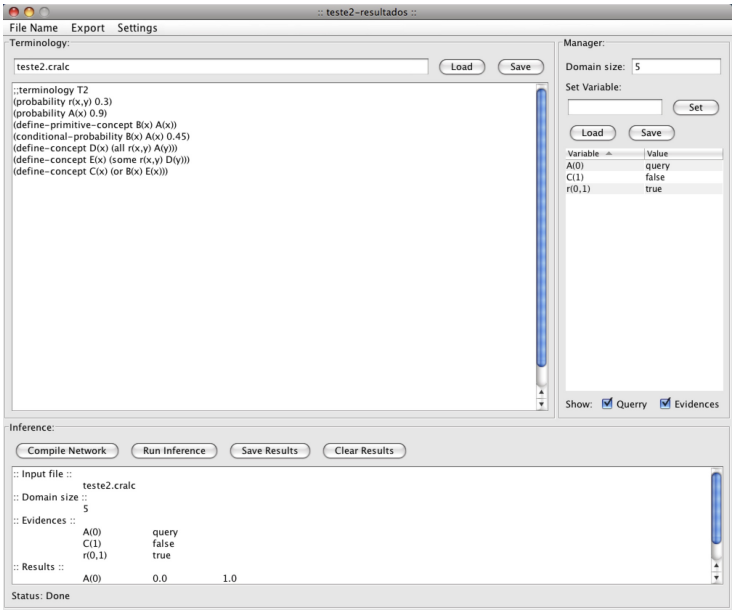

Fig. 1. Terminologies are written in the larger panel, while assertions are set in the right panel; the lower panel reports on inferences.

the following constructs: `(and C1...Cn)` for conjunction; `(or C1...Cn)` for disjunction; `(not C)` for complement; `(all r C)` to indicate the quantifier $\forall r.C$; `(some r C)` to indicate the quantifier $\exists r.C$; `(define-concept C D)` for $C \equiv D$; and `(define-primitive-concept C D)` for $C \sqsubseteq D$.

Probabilistic inclusions are specified as follows: `(probability B` α`)` denotes $P(B) = \alpha$; `(conditional-probability B A` α`)` for $P(A|B) = \alpha$. An example of valid input file is:

```
(probability A(x) 0.7)       (probability B(x) 0.4)
(define-concept C(x) (and A(x) (not B(x))))
```

Assertions can be represented through written files as well; inference results can be exported to files. Alternatively, the graphical user interface depicted in Figure 1 can be used to load/save files, to specify the size of the domain and the assertions, to ask for inferences, and to check results. The package is freely available at http://sites.poli.usp.br/pmr/ltd/Software/CRALC/index.html.

Approximate inferences are produced by generating a set of grounded Bayesian networks, one for each individual mentioned in the query and in the evidence, plus an additional Bayesian network for a "generic" individual [3]. Exact Bayesian network inference is performed in each one of these networks (the package assumes that such exact inference is feasible) and messages are exchanged between the networks using a loopy-propagation scheme. A relatively small number of message-passing iterations seems to generate good approximations; the cost

of running an approximate inference is then the number of allowed iterations times the sum of inference costs for each one of the grounded networks plus the "generic" individual network.

4 Conclusion

Efficient inference for probabilistic description logics is a key enabler of technologies that must deal with uncertainty and semantic information. Currently there are many proposals for probabilistic description logics but relatively few implemented inference engines. In this short paper we have described our modest efforts in providing easier ways to represent and process sentences in probabilistic description logics. The software package we have presented still requires much development, but it is a step in a direction we feel has not received enough attention.

Our current effort is to implement exact *lifted* inference; that is, inference that does not require grounding concepts and roles for the entire domain. We are using recently developed methods for lifted inference in graphical models [5], and plan to report on the results soon.

Acknowledgements

The first author was supported by FAPESP. The second author is partially supported by CNPq. The work reported here has received substantial support by FAPESP grant 2008/03995-5.

References

1. F. Baader, D. Calvanese, D.L. McGuinness, D. Nardi, and P.F. Patel-Schneider. *Description Logic Handbook*. Cambridge University Press, 2003.
2. F. Correa, F. G. Cozman, and J. Okamoto Jr. Collective classification in semantic mapping with a probabilistic description logic. In *Int. Workshop on Description Logics*, pages 455–465, Barcelona, Spain, 2011.
3. F. G. Cozman and R. B. Polastro. Complexity analysis and variational inference for interpretation-based probabilistic description logics. In *Uncertainty in Artificial Intelligence*, pages 117–125, Corvallis, Oregon, 2009. AUAI Press.
4. M. Jaeger. Relational Bayesian networks. In *Uncertainty in Artificial Intelligence*, pages 266–273, San Francisco, California, 1997. Morgan Kaufmann.
5. J. Kisynski and D. Poole. Lifted aggregation in directed first-order probabilistic models. Int. Joint Conf. on Artifical Intelligence, pages 19221929, 2009.
6. T. Lukasiewicz and U. Straccia. Managing uncertainty and vagueness in description logics for the semantic web. *Journal of Web Semantics*, 6:291–308, 2008.
7. J. E. Ochoa Luna, K. Revoredo, and F. G. Cozman. Learning probabilistic description logics: A framework and algorithms. In *Mexican Int. Conf. on Artificial Intelligence, Lecture Notes in Artificial Intelligence*, volume 7094 Part I, pages 28–39. Springer, 2011.
8. K. Revoredo, J. E. Ochoa-Luna, and F. G. Cozman. Semantic link prediction through probabilistic description logic. In *Workshop on Uncertainty Reasoning for the Semantic Web*, pages 87–97, Bonn, Germany, 2011.