ELOG: A Probabilistic Reasoner for OWL EL

Jan Noessner and Mathias Niepert

KR & KM Research Group,
Universität Mannheim
Mannheim, Germany
{jan,mathias}@informatik.uni-mannheim.de

Abstract. Log-linear description logics are probabilistic logics combining several concepts and methods from the areas of knowledge representation and reasoning and statistical relational AI. We describe some of the implementation details of the log-linear reasoner ELOG. The reasoner employs database technology to dynamically transform inference problems to integer linear programs (ILP). In order to lower the size of the ILPs and reduce the complexity we employ a form of cutting plane inference during reasoning.

1 Introduction

Many problems involve both uncertain and certain knowledge. There are several lines of research that attempt to combine the diverse concepts and methods from uncertainty in AI and logical reasoning. Some resulting approaches are probabilistic description logics [4, 5] and statistical relational languages [3]. While the former have mainly been studied on a theoretical level, statistical relational languages have been proven useful for a number of practical applications such as data integration [7]. A particular line of work that has regained importance with the prevalence of semantic web research is the combination of probabilistic models with description logics [5].

In this paper, we present the probabilistic-logical reasoner ELOG[1]. ELOG is a reasoner for log-linear description logics [8] which combine log-linear models and lightweight description logics. Log-linear description logics allow us to model both probabilistic and deterministic dependencies between DL axioms. Fortunately, there exist a number of sophisticated algorithms for inference and parameter learning for log-linear models which we adapt to our needs. Log-linear models form the basis of several languages in the realm of statistical relational learning such as Markov logic [10] for which numerous algorithms for inference exist [9, 11, 6].

At this point, we support probabilistic reasoning for the description logic $\mathcal{EL}^{++}$ which captures the expressivity of numerous ontologies in the life sciences and other application domains. $\mathcal{EL}^{++}$ is also the description logic on which the web ontology language profile OWL 2 EL is based [2].

[1] ELOG is available at `http://code.google.com/p/elog-reasoner/`.

After briefly describing the syntax and semantics of log-linear description logics, the focus on the reduction of the inference problem to instances of integer linear programming. In this paper, we are concerned with the maximum a-posteriori (MAP) query which computes *most probable coherent* ontologies. Furthermore, we will describe how cutting plane inference can be applied to significantly increase the efficiency.

2 Log-Linear Description Logics

Log-linear description logics combine description logics and probabilistic log-linear models. We focus on the log-linear description logic based on $\mathcal{EL}^{++}$ without nominals and concrete domains which we denote as $\mathcal{EL}^{++}$-LL. A $\mathcal{EL}^{++}$-LL CBox $\mathcal{C} = (\mathcal{C}^{\mathsf{D}}, \mathcal{C}^{\mathsf{U}})$ is a pair consisting of a *deterministic* $\mathcal{EL}^{++}$ CBox $\mathcal{C}^{\mathsf{D}}$ and an *uncertain* CBox $\mathcal{C}^{\mathsf{U}} = \{(c, w_c)\}$ which is a set of pairs (c, w_c) with each c being a $\mathcal{EL}^{++}$ axiom and w a real-valued weight assigned to c. While the *deterministic* CBox contains axioms that are known to be true the *uncertain* CBox contains axioms for which we only have a *degree of confidence*. Intuitively, the greater the weight of an axiom the more likely it is true. Every axiom can either be part of the deterministic or the uncertain CBox but not part of both. The deterministic CBox is assumed to be coherent and satisfiable.

Computing answers for probabilistic queries over a $\mathcal{EL}^{++}$-LL CBox implies logical reasoning and coherence checking. We achieve both via a mapping to a first-order representation leading to a one-to-one correspondence between models of this representation on one side and classified and coherent $\mathcal{EL}^{++}$ CBoxes on the other. In particular, we implicitly maintain three sets of ground formulas that form the basis for the probabilistic reasoning algorithms: A set $\mathcal{K}$ containing the transformed axioms of the deterministic CBox; a set $\mathcal{F}^{\mathsf{N_U}}$ containing grounded formulas partially based on the completion rules for the DL $\mathcal{EL}^{++}$[2] (see Table 1); and a set $\mathcal{G}$ containing weighted formulas representing the axioms in the uncertain CBox. The mapping allows us to adapt existing algorithms from the area of statistical relational AI. The mapping is defined as follows.

Definition 1 (CBox Mapping). *Let* $\mathsf{N_C}$ *and* $\mathsf{N_R}$ *be sets of concept and role names and let* $\mathsf{N_U} \subseteq \mathsf{N_C} \cup \mathsf{N_R}$ *be a finite set. Let* $\mathcal{T}$ *be the set of normalized* $\mathcal{EL}^{++}$ *axioms constructible from* $\mathsf{N_U}$*. Moreover, let* $\mathcal{H}$ *be the Herbrand base of* $\mathcal{F}$ *with respect to* $\mathsf{N_U}$*. The function* $\varphi : \wp(\mathcal{T}) \rightarrow \wp(\mathcal{H})$ *maps normalized CBoxes to subsets of* $\mathcal{H}$ *as follows:* $(\varphi(\mathcal{C}) = \bigcup_{c \in \mathcal{C}} \varphi(c))$

$$
\begin{array}{ll}
C_1 \sqsubseteq D & \mapsto sub(C_1, D) \\
C_1 \sqcap C_2 \sqsubseteq D & \mapsto int(C_1, C_2, D) \\
C_1 \sqsubseteq \exists r.C_2 & \mapsto rsup(C_1, r, C_2) \\
\exists r.C_1 \sqsubseteq D & \mapsto rsub(C_1, r, D) \\
r \sqsubseteq s & \mapsto psub(r, s) \\
r_1 \circ r_2 \sqsubseteq r_3 & \mapsto pcom(r_1, r_2, r_3).
\end{array}
$$

The predicates in Definition 1 are typed, meaning that $r, s, r_i, (1 \leq i \leq 3)$, are role names, C_1, C_2 basic concept descriptions, and D basic concept descriptions

Table 1. The set of first-order formulas $\mathcal{F}$. Instantiations of formulas have to be compatible with the types of the predicates previously defined. $\perp$ and $\top$ are constant symbols representing the bottom and top concept.

F_1	$\forall c : sub(c, c)$
F_2	$\forall c : sub(c, \top)$
F_3	$\forall c, c', d : sub(c, c') \wedge sub(c', d) \Rightarrow sub(c, d)$
F_4	$\forall c, c_1, c_2, d : sub(c, c_1) \wedge sub(c, c_2) \wedge int(c_1, c_2, d) \Rightarrow sub(c, d)$
F_5	$\forall c, c', r, d : sub(c, c') \wedge rsup(c', r, d) \Rightarrow rsup(c, r, d)$
F_6	$\forall c, r, d, d', e : rsup(c, r, d) \wedge sub(d, d') \wedge rsub(d', r, e) \Rightarrow sub(c, e)$
F_7	$\forall c, r, d, s : rsup(c, r, d) \wedge psub(r, s) \Rightarrow rsup(c, s, d)$
F_8	$\forall c, r_1, r_2, r_3, d, e : rsup(c, r_1, d) \wedge rsup(d, r_2, e) \wedge pcom(r_1, r_2, r_3) \Rightarrow rsup(c, r_3, e)$
F_9	$\forall c : \neg sub(c, \perp)$

or the bottom concept. Every $\mathsf{N_U}$ can be transformed into a normalized CBox $\mathcal{T}$ by applying a finite set of normalization rules and introducing new concept and role names [2].

The semantics of $\mathcal{EL}^{++}$-LL is fully captured by the set of first-order formulas $\mathcal{F}$ partially derived from the $\mathcal{EL}^{++}$ completion rules (see Table 1). We construct the first-order representation of a $\mathcal{EL}^{++}$-LL CBox as follows: $\mathcal{G}$ is a set of *weighted* ground formulas carrying the uncertain information and is derived from the axioms in the uncertain CBox $\mathcal{C}^{\mathsf{U}}$. For every pair $(c, w_c) \in \mathcal{C}^{\mathsf{U}}$ we add the conjunction of ground atoms

$$\bigwedge_{\mathsf{g} \in \varphi(\mathsf{norm}(\{c\}))} \mathsf{g}$$

to $\mathcal{G}$ with weight w_c. The set $\mathcal{K}$ is constructed analogously from the deterministic CBox $\mathcal{C}^{\mathsf{D}}$ except that we do not associate weights with the ground formulas. Finally $\mathcal{F}^{\mathsf{N_U}}$ is the set of all instantiations of formulas in $\mathcal{F}$ (see Table 1) relative to $\mathsf{N_U}$. For a detailed justification we refer the reader to [8].

Example 1. Let $\mathcal{C}^{\mathsf{D}} = \{C \sqsubseteq D, D \sqsubseteq E\}$ and $\mathcal{C}^{\mathsf{U}} = \{\langle B \sqcap C \sqcap D \sqsubseteq E, 0.5\rangle\}$. Then, $\mathsf{norm_{LL}}(\mathcal{C}) = \mathsf{norm}(\{C \sqsubseteq D\}) \cup \mathsf{norm}(\{D \sqsubseteq E\}) \cup \mathsf{norm}(\{B \sqcap C \sqcap D \sqsubseteq E\}) = \{C \sqsubseteq D, D \sqsubseteq E, C \sqcap D \sqsubseteq A, B \sqcap A \sqsubseteq E\}$ with A a new concept name. Furthermore, $\varphi(\mathsf{norm}(\{C \sqsubseteq D\})) = \{sub(c, d)\}$, $\varphi(\mathsf{norm}(\{D \sqsubseteq E\})) = \{sub(d, e)\}$ and $\varphi(\mathsf{norm}(\{B \sqcap C \sqcap D \sqsubseteq E\})) = \{int(c, d, a), int(b, a, e)\}$. Hence, $\mathcal{K} = \{sub(c, d), sub(d, e)\}$ and $\mathcal{G} = \{\langle int(c, d, a) \wedge int(b, a, e), 0.5\rangle\}$.

3 Reasoning as Combinatorial Optimization

After having defined the syntax and semantic of $\mathcal{EL}^{++}$-LL, in this section we show that inference in the language can be formulated as an optimization problem. Under the given syntax and semantics the MAP query answers the following question: *"Given a $\mathcal{EL}^{++}$-LL CBox, what is a most probable coherent $\mathcal{EL}^{++}$ CBox over the same concept and role names?"*

3.1 Transformation to ILP

ELOG solves MAP queries by transforming the inference problem into an integer linear program (ILP). In principle, it maximizes the sum of confidence values of the axioms subject to the coherence of the ontology. For each ground atom g_i occurring in a formula in $\mathcal{G}, \mathcal{K}$, or $\mathcal{F}^{\mathsf{N_U}}$ we associate a binary variable $x_i \in \{0, 1\}$. Let C_j^G be the set of indices of ground atoms in formula $G_j \in \mathcal{G}$, let C_j^K be the set of indices of ground atoms in formula $K_j \in \mathcal{K}$, and let C_j^F ($\bar{C}_j^F$) be the set of indices of unnegated (negated) ground atoms in clause $F_j \in \mathcal{F}^{\mathsf{N_U}}$. With each formula $G_j \in \mathcal{G}$ with weight w_j we associate a variable $z_j \in \{0, 1\}$. Then, the ILP is stated as follows

$$\max \sum_{j=1}^{|\mathcal{G}|} w_j z_j \qquad \text{subject to}$$

$$\sum_{i \in C_j^G} x_i \geq |C_j^G| z_j,\ \forall j \quad \textit{(type 1)} \ \text{ and } \sum_{i \in C_j^K} x_i \geq |C_j^K|,\ \forall j \quad \textit{(type 2)}$$

$$\text{and} \sum_{i \in C_j^F} x_i + \sum_{i \in \bar{C}_j^F} (1 - x_i) \geq 1,\ \forall j \quad \textit{(type 3)}.$$

Every solution $\{x_i\}$ of the ILP corresponds (via the function φ) to a most probable classified and coherent CBox over $\mathsf{N_U}$ that entails $\mathcal{C}^{\mathsf{D}}$.

Example 2. Let z_1 be associated with the weighted formula $\langle int(c, d, a) \wedge int(b, a, e), 0.5\rangle$ from Example 1. Then, the objective is max $0.5z_1$ with linear constraints $x_{int(c,d,a)} + x_{int(b,a,e)} \geq 2z_1$ *(type 1)*, $x_{sub(c,d)} \geq 1$, $x_{sub(d,e)} \geq 1$ *(type 2)*, and $\forall \alpha, \beta, \gamma, \delta \in \{a, b, c, d, e\}$, we obtain $x_{sub(\alpha,\alpha)} \geq 1$ (*type 3* F_1), $x_{sub(\alpha,\top)} \geq 1$ (*type 3* F_2), $(1 - x_{sub(\alpha,\beta)}) + (1 - x_{sub(\beta,\gamma)}) + x_{sub(\alpha,\gamma)} \geq 1$ (*type 3* F_3), $(1 - x_{sub(\alpha,\beta)}) + (1 - x_{sub(\alpha,\gamma)}) + (1 - x_{int(\beta,\gamma,\delta)}) + x_{sub(\alpha,\delta)} \geq 1$ (*type 3* F_4). Since there are no roles in the example, F_5-F_8 need not be considered.

This example illustrates that even for a small number of concepts the number of *type 3* constraints explode. For the 5 concepts we have $5 + 5 + 5^3 + 5^4 = 760$ *type 3* linear constraints.

3.2 Cutting Plane Inference

As we have seen, for most realistic problems the resulting optimization problems will become very large and potentially intractable especially because of the *type 3* constraints. For that reason, we will apply cutting plane inference (CPI) on the problem. We will empirical confirm that CPI significantly reduces the runtime of the ELOG reasoner, although the problem of finding the maximum a-posteriori state for $\mathcal{EL}^{++}$-LL CBoxes is NP-hard [8].

More precisely, we employ a variant of the CPI algorithm first proposed in the context of Markov logic [11]. We first solve the optimization problem *without* constraints of *type 3*. Given a solution to the partial problem we determine in polynomial time (using conjunctive queries) those constraints of *type 3* that are violated by this solution, add those to the formulation, and solve the updated problem. This is repeated until no violated constraints remain. We apply inner joins to retrieve the groundings of forumlas that are violated in the current solution. Since joins in relational databases have a polynomial data complexity [1] the construction of the ILP is performed in polynomial time.

Example 3. We first solve the problem with the objective $\max 0.5 z_1$ and the constraints $x_{int(c,d,a)} + x_{int(b,a,e)} \geq 2z_1$ (*type 1*) and $x_{sub(c,d)} \geq 1$ (*type 2*). This results in $x_{sub(c,d)} = x_{sub(d,e)} = x_{int(c,d,a)} = x_{int(b,a,e)} = 1$. Then, we add the violated constraints of *type 3*: $(1 - x_{sub(c,d)}) + (1 - x_{sub(d,e)}) + x_{sub(c,e)} \geq 1$, and solve the ILP again. The new results is $x_{sub(c,d)} = x_{sub(d,e)} = x_{int(c,d,a)} = x_{int(b,a,e)} = x_{sub(c,e)} = 1$. No type 3 constraints are violated and we have determined the final solution.

The example demonstrates the benefit of cutting plane inference: the number of *type 3* constraints could be reduced from 760 to only 1. Although more often than not more than one CPI *iteration* is needed the resulting ILPs are much more compact. We employ standard database systems for computing the CPI queries. The temporary solution is stored in relational tables one for each predicate. Furthermore, tables `concept` and `property` are created which contain all concept and property names. In each CPI iteration every query (one for each formula $F_i, i \in \{1, 2, ..., 8\}$) returns the violated constraints which are then added to the ILP. The constraint is violated if the formula evaluates to $false$, or in other words, if the *negated* formula evaluates to *true*. Consequently, we first negate the formula F_i and build the database joins based on this negated formula $\bar{F_i}$. For every positive predicate in the formula $\bar{F_i}$ we perform an inner join while negative predicates are subtracted from the current results. The following example illustrates how the queries are build.

Example 4. The negated formula of F_3 is $\bar{F}_3 \equiv \forall c, d, e : sub(c, d) \wedge sub(d, e) \wedge \neg sub(c, e)$. From this formula we construct the following SQL query (simplified) which returns the concrete values for c, d and e[2]:

```
SELECT xx.c, xx.d, xx.e FROM
( SELECT c.col0 as c, d.col0 as d, e.col0 as e
  FROM concept c, concept d, concept e, sub sub0, sub sub1
  WHERE sub0.col0 = c.col0 AND sub0.col1 = d.col0
    AND sub1.col0 = d.col0 AND sub1.col1 = e.col0
) as xx LEFT JOIN sub sub2 ON sub2.col0 = xx.c AND sub2.col1 = xx.e
WHERE sub2.col0 IS NULL
```

4 Experiments

In our preliminary experiments we compare the runtime of the reasoner with and without cutting plane inference. We use the dataset and experimental set-

[2] `y.colX` symbolizes the column number `x` in table `y`.

Table 2. Runtimes in seconds.

Algorithm	Preparation Time	Reasoning Time	Overall Time
With CPI	4.7	3.7	8.4
Without CPI	5.0	> 10,000	> 10,000

up from [8] where confidence values for axioms were generated using a crowdsourcing service. The gold standard ontology consists of 75 classes, 33 object properties, 148 subsumption, and 4,598 disjointness axioms when materialized. All experiments were conducted on a desktop PC with Intel Core2 Processor P8600 with 2.4GHz and 2GB RAM.

Table 2 summarizes the runtimes of the algorithms. The preparation time for both algorithm took around 5 seconds where most of the time is needed to read the ontology files. The naïve algorithm without CPI did not terminate within 10,000 seconds because the large ILP could not be solved within the time. In fact, the *type 3* formula F_4 resulted in $75^4 = 31,640,625$ linear constraints. ELOG's cutting plane inference method needed only 3.7 seconds and 7 iterations to classify the ontology and to solve the maximum a-posteriori query. The results indicate that using cutting-plane-inference is more efficient. In our case, the problem has even been intractable without CPI. These results are in line with results in the context of Markov logic [11].

References

1. Aho, A., Beeri, C., Ullman, J.: The theory of joins in relational databases. ACM Transactions on Database Systems (TODS) 4(3), 297–314 (1979)
2. Baader, F., Brandt, S., Lutz, C.: Pushing the $\mathcal{EL}$ envelope. In: Proceedings of IJCAI (2005)
3. Getoor, L., Taskar, B.: Introduction to Statistical Relational Learning. MIT Press (2007)
4. Koller, D., Levy, A., Pfeffer, A.: P-classic: A tractable probabilistic description logic. In: Proceedings of the 14th AAAI Conference on Artificial Intelligence (1997)
5. Lukasiewicz, T., Straccia, U.: Managing uncertainty and vagueness in description logics for the semantic web. J. of Web Sem. 6 (2008)
6. Niepert, M.: A Delayed Column Generation Strategy for Exact k-Bounded MAP Inference in Markov Logic Networks. In: Proceedings of UAI (2010)
7. Niepert, M., Meilicke, C., Stuckenschmidt, H.: A Probabilistic-Logical Framework for Ontology Matching. In: Proceedings of AAAI (2010)
8. Niepert, M., Noessner, J., Stuckenschmidt, H.: Log-Linear Description Logics. In: Proceedings of IJCAI (2011)
9. Poon, H., Domingos, P.: Sound and efficient inference with probabilistic and deterministic dependencies. In: Proceedings of AAAI (2006)
10. Richardson, M., Domingos, P.: Markov logic networks. Machine Learning 62(1-2) (2006)
11. Riedel, S.: Improving the accuracy and efficiency of map inference for markov logic. In: Proceedings of UAI (2008)