Ontology-mediated Queries over Probabilistic Data via Probabilistic Logic Programming

Timothy van Bremen
timothy.vanbremen@cs.kuleuven.be
KU Leuven, Belgium

Anton Dries
anton.dries@cs.kuleuven.be
KU Leuven, Belgium

Jean Christoph Jung
jeanjung@uni-bremen.de
Universität Bremen, Germany

ABSTRACT

We study ontology-mediated querying over probabilistic data for the case when the ontology is formulated in $\mathcal{ELH}^{dr}$, an expressive member of the $\mathcal{EL}$ family of description logics. We leverage techniques that have been developed (i) for classical ontology-mediated querying and (ii) for probabilistic logic programming and provide an implementation based on our findings. We include both theoretical considerations and an experimental evaluation of our approach.

ACM Reference Format:
Timothy van Bremen, Anton Dries, and Jean Christoph Jung. 2019. Ontology-mediated Queries over Probabilistic Data via Probabilistic Logic Programming. In *The 28th ACM International Conference on Information and Knowledge Management (CIKM'19), November 3–7, 2019, Beijing, China.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3357384.3358168

1 INTRODUCTION

In many domains, data is inherently probabilistic due to uncertainty in the measurement or extraction process: for example, temperature readings from an unreliable sensor, or information extracted from the Internet using an imperfect NLP system. We nevertheless may be in possession of some deterministic domain knowledge which can be used to enrich the data. For instance, we may know that it cannot be snowing if we record the weather as being hot, or that every person extracted with the NLP system must have a date of birth. This idea of accessing uncertain data enriched with deterministic knowledge (an ontology) has led to the framework of *ontology-mediated querying over probabilistic data (OMQPD)* [12]. The main reasoning task in this setting is to evaluate a query over a probabilistic dataset, modelled as a tuple-independent probabilistic database, in the presence of an ontology.

Existing work on OMQPD has focused mainly on lightweight ontology languages from the *DL-Lite* family [4]. One observation [16] of particular interest was that the well-known technique of computing *first-order rewritings* [4] serves as a useful tool for implementing OMQPD systems. Intuitively, instead of evaluating a query in the presence of an ontology, the ontology is compiled into the query; the result is then evaluated (without ontology) over the data.

In this paper we focus on the ontology language $\mathcal{ELH}^{dr}$, an expressive member of the $\mathcal{EL}$ family of description logics [5], which forms the basis of the OWL 2 EL profile [1]. In contrast to *DL-Lite*, first-order rewritings are not a complete method for ontology-mediated querying with ontologies formulated in $\mathcal{EL}$ (or extensions thereof). That is, there exist query/ontology pairs for which no equivalent first-order rewriting exists [7].

In order to have a complete tool, we show how the *combined approach* to ontology-mediated querying over deterministic data [13] can be lifted to the probabilistic case. This results in a rewriting of the query, ontology, and probabilistic data into a *probabilistic logic program (PLP)*, which is a logic program augmented with uncertainty; see [9] for a recent survey. Often based on the distribution semantics, PLPs feature a combination of uncertainty and deterministic rules similar to the OMQPD setting. From a practical perspective, our rewriting technique allows us to take advantage of the extensive research conducted in the probabilistic logic programming community, which has resulted in several powerful inference systems, e.g., PRISM [15], cplint [2] and ProbLog [14]. As a proof-of-concept, we implemented the rewriting approach using ProbLog as an inference system and evaluated it on a probabilistic variant of the Lehigh University Benchmark [11]. From a theoretical perspective our approach also yields a polynomial time reduction from OMQPD to *weighted model counting* over propositional formulas, a popular approach for probabilistic inference [8].

2 BACKGROUND

We briefly review the description logic $\mathcal{ELH}^{dr}$. Fix disjoint countably infinite sets of concept and role names N_C and N_R, respectively. Then $\mathcal{EL}$-concepts are formed according to the syntax rule

$$C ::= \top \mid A \mid C \sqcap C \mid \exists r.C$$

where $A \in N_C$ and $r \in N_R$. An $\mathcal{ELH}^{dr}$*-ontology* (hereafter ontology) is a set of *concept inclusions* $C \sqsubseteq D$, *role inclusions* $r \sqsubseteq s$, *domain restrictions* $\mathsf{dom}(r) \sqsubseteq C$, and *range restrictions* $\mathsf{ran}(r) \sqsubseteq C$, where C and D are $\mathcal{EL}$-concepts and $r, s \in N_R$. An ABox is a finite set of concept assertions $A(a)$ and role assertions $r(a, b)$ where $A \in N_C$, $r \in N_R$, and a, b range over a countably infinite set of individual names N_I. We denote with $\mathsf{Ind}(\mathcal{A})$ the set of all individual names that occur in $\mathcal{A}$. The semantics of $\mathcal{ELH}^{dr}$ is defined as usual in terms of interpretations $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$; we elide a full description here and instead refer the reader to Baader et al. [6] for details. Following [5, 13], we assume without loss of generality that *(i)* for every role name r, there is exactly one range restriction $\mathsf{ran}(r) \sqsubseteq A$ in $\mathcal{T}$, and we denote the concept name A with $\mathsf{Ran}(r)$, *(ii)* if $\mathcal{T} \models r \sqsubseteq s$, then $\mathcal{T} \models \mathsf{Ran}(r) \sqsubseteq \mathsf{Ran}(s)$, *(iii)* there are no $r \neq s$ with $\mathcal{T} \models s \sqsubseteq r$ and $\mathcal{T} \models r \sqsubseteq s$, *(iv)* for every domain restriction $\mathsf{dom}(r) \sqsubseteq C$, we have $C \in N_C$, and *(v)* every $C \sqsubseteq D$ in $\mathcal{T}$ takes one of the following forms, for concept names A, A', B:

$$A \sqcap A' \sqsubseteq B, \quad \exists r.A \sqsubseteq B, \quad A \sqsubseteq \exists r.B.$$

CIKM '19, November 3–7, 2019, Beijing, China

ACM ISBN 978-1-4503-6976-3/19/11…$15.00
https://doi.org/10.1145/3357384.3358168

Ontology-mediated Querying over Probabilistic Data. Let N_V denote a countably infinite set of variables disjoint from N_I. Then $N_T = N_V \cup N_I$ forms a set of terms. A *conjunctive query (CQ)* φ is a first-order formula of the form $\varphi(\mathbf{x}) = \exists \mathbf{y}.\psi(\mathbf{x}, \mathbf{y})$, where $\mathbf{x}$ and $\mathbf{y}$ are tuples of variables in N_V, and $\phi(\mathbf{x}, \mathbf{y})$ is a conjunction of atoms over signature $N_C \cup N_R$ using terms from N_T. We drop the free variables $\mathbf{x}$ of $\varphi(\mathbf{x})$ whenever no confusion can arise. An *ontology-mediated query (OMQ)* is a pair $(\mathcal{T}, \varphi)$ of ontology $\mathcal{T}$ and CQ φ. Given an ABox $\mathcal{A}$, and an OMQ $(\mathcal{T}, \varphi)$, we say that a tuple $\mathbf{a}$ of individuals from $\mathcal{A}$ is a *certain answer for* $(\mathcal{T}, \varphi)$ *over* $\mathcal{A}$ if $(\mathcal{T}, \mathcal{A}) \models \varphi(\mathbf{a})$, that is, every model $\mathcal{I}$ of $\mathcal{T}$ and $\mathcal{A}$ satisfies $\mathcal{I} \models \varphi(\mathbf{a})$. The set of all certain answers to $(\mathcal{T}, \varphi)$ is denoted by $\mathsf{cert}_{\mathcal{A}}(\mathcal{T}, \varphi)$.

Following [12], we use *assertion-independent probabilistic ABoxes (ipABoxes)* to model uncertain data. An ipABox is a pair $(\mathcal{A}, p)$ where $\mathcal{A}$ is a classical ABox and $p : \mathcal{A} \to [0, 1]$ assigns a probability to every assertion in $\mathcal{A}$. An ipABox $(\mathcal{A}, p)$ induces a distribution $p(\cdot)$ over possible ABoxes $\mathcal{A}' \subseteq \mathcal{A}$, which is defined by taking

$$p(\mathcal{A}') = \Pi_{\alpha \in \mathcal{A}'} p(\alpha) \cdot \Pi_{\alpha \in \mathcal{A} \setminus \mathcal{A}'} (1 - p(\alpha)), \tag{1}$$

for every $\mathcal{A}' \subseteq \mathcal{A}$. The *probability of an answer* $\mathbf{a}$ *to an OMQ* $(\mathcal{T}, \varphi)$ *over an ipABox* $(\mathcal{A}, p)$ is then defined as:

$$Pr_{\mathcal{A},p}(\mathcal{T}, \varphi, \mathbf{a}) = \sum_{\mathcal{A}' \subseteq \mathcal{A}, \mathbf{a} \in \mathsf{cert}_{\mathcal{A}'}(\mathcal{T}, \varphi)} p(\mathcal{A}').$$

The prime inference task here is to *compute answer probabilities*, that is, given an ipABox $(\mathcal{A}, p)$ and an OMQ $(\mathcal{T}, \varphi)$ with answer candidate $\mathbf{a}$, compute $Pr_{\mathcal{A},p}(\mathcal{T}, \varphi, \mathbf{a})$.

Probabilistic Logic Programs. We introduce a variant of probabilistic logic programs that is sufficient for our purposes, though some systems support more features. A *probabilistic logic program (PLP)* is a triple $(\mathcal{F}, p, \Pi)$ where $\mathcal{F}$ is a set of facts, $p : \mathcal{F} \to [0, 1]$ assigns a probability to every fact, and Π is a *stratified logic program* consisting of rules of the form:

$$H \leftarrow B_1, \ldots, B_m, \neg B_{m+1}, \ldots, \neg B_n$$

where H and all B_i are relational atoms over variables and individual names. The semantics of PLPs $(\mathcal{F}, p, \Pi)$ is defined as follows. The pair $(\mathcal{F}, p)$ induces a probability distribution $p(\cdot)$ over subsets $\mathcal{F}' \subseteq \mathcal{F}$ just as in Equation (1). Moreover, given a set of facts $\mathcal{F}$ and a set of rules Π, we denote with $\Pi(\mathcal{F})$ the *minimal supported model* of $\mathcal{F} \cup \Pi$, obtained via the iterated fixed point construction of [3]. The prime inference task for PLPs is *marginal inference*, that is, compute the *probability of a ground fact* G *under the PLP* $(\mathcal{F}, p, \Pi)$, which is:

$$Pr_{\mathcal{F},p,\Pi}(G) = \sum_{\mathcal{F}' \subseteq \mathcal{F}, G \in \Pi(\mathcal{F}')} p(\mathcal{F}').$$

3 COMBINED APPROACH OVER IPABOXES

Let us briefly review the *combined approach* to OMQs over non-probabilistic ABoxes [13]. For simplicity, we use a slightly different presentation than in the original paper, but they are easily seen to be equivalent. In a nutshell, given $(\mathcal{T}, \varphi)$ and $\mathcal{A}$, one computes in polynomial time an ABox $\mathcal{A}_\mathcal{T} \supseteq \mathcal{A}$ and a set of rules $\Pi_{\mathcal{T},\varphi}$ such that, for some distinguished relation name Goal, we have

$$\mathsf{cert}_{\mathcal{A}}(\mathcal{T}, \varphi) = \{\mathbf{a} \mid \mathsf{Goal}(\mathbf{a}) \in \Pi_{\mathcal{T},\varphi}(\mathcal{A}_\mathcal{T})\}.$$

Thus, the computation of certain answers is reduced to computing $\mathcal{A}_\mathcal{T}$ and the minimal model $\Pi_{\mathcal{T},\varphi}(\mathcal{A}_\mathcal{T})$. We will give more details on how $\mathcal{A}_\mathcal{T}$ and $\Pi_{\mathcal{T},\varphi}$ are constructed when they are needed to prove correctness of our adaptation. Note that the combined approach is not directly applicable to answering an OMQ $(\mathcal{T}, \varphi)$ over an ipABox $(\mathcal{A}, p)$ since for every $\mathcal{A}' \subseteq \mathcal{A}$, the ABox $\mathcal{A}'_\mathcal{T}$ might be different. We solve this by providing a set of rules $\Pi_\mathcal{T}$ such that:

$$\text{for every } \mathcal{A}' \subseteq \mathcal{A}, \text{ we have } \Pi_{\mathcal{T},\varphi}(\Pi_\mathcal{T}(\mathcal{A}')) = \Pi_{\mathcal{T},\varphi}(\mathcal{A}'_\mathcal{T}). \quad (*)$$

Given $(*)$, it is not hard to verify that we can use $\Pi_{\mathcal{T},\varphi} \cup \Pi_\mathcal{T}$ to lift the combined approach to answering OMQs over ipABoxes. More precisely, the following Lemma is an immediate consequence of the definition of marginal probabilities and $(*)$:

Lemma 1. *For every OMQ* $(\mathcal{T}, \varphi)$*, ipABox* $(\mathcal{A}, p)$*, and answer candidate* $\mathbf{a}$*, we have:* $Pr_{\mathcal{A},p}(\mathcal{T}, \varphi, \mathbf{a}) = Pr_{\mathcal{A},p,\Pi_{\mathcal{T},\varphi} \cup \Pi_\mathcal{T}}(\mathsf{Goal}(\mathbf{a}))$.

In the construction of $\Pi_\mathcal{T}$, we use fresh individuals $a_{A,B}$, where A, B are concept names that occur in $\mathcal{T}$, and a fresh concept name Aux to mark these individuals as auxiliary. Then, $\Pi_\mathcal{T}$ is the collection of the following rules (which do not use negation):

- $A(a_{A,B})$, $B(a_{A,B})$, and $\mathsf{Aux}(a_{A,B})$, for every fresh $a_{A,B}$;
- $B(x) \leftarrow A(x),\ A'(x)$, for every $A \sqcap A' \sqsubseteq B \in \mathcal{T}$;
- $B(x) \leftarrow A(y),\ r(x, y)$, for every $\exists r.A \sqsubseteq B \in \mathcal{T}$;
- $r(x, a_{B,\mathsf{Ran}(r)}) \leftarrow A(x)$, for every $A \sqsubseteq \exists r.B \in \mathcal{T}$;
- $s(x, y) \leftarrow r(x, y)$, for every $r \sqsubseteq s \in \mathcal{T}$;
- $A(x) \leftarrow r(y, x)$, for every $\mathsf{ran}(r) \sqsubseteq A \in \mathcal{T}$;
- $A(x) \leftarrow r(x, y)$, for every $\mathsf{dom}(r) \sqsubseteq A \in \mathcal{T}$.

In order to prove $(*)$, we recall the construction of $\mathcal{A}_\mathcal{T}$ and the relevant properties of the PLP $\Pi_{\mathcal{T},\varphi}$ as used in the combined approach. The extension $\mathcal{A}_\mathcal{T}$ of $\mathcal{A}$ contains the following assertions:

- $\mathsf{Aux}(a_{A,B})$, for every fresh $a_{A,B}$;
- $B(a)$, for all $a \in \mathsf{Ind}(\mathcal{A})$ and B such that $\mathcal{T} \cup \mathcal{A} \models B(a)$;
- $A'(a_{A,B})$, for all $a_{A,B}$ such that $\mathcal{T} \models A \sqcap B \sqsubseteq A'$;
- $r(a, b)$ for all $a, b \in \mathsf{Ind}(\mathcal{A})$ with $s(a, b) \in \mathcal{A}$ and $\mathcal{T} \models s \sqsubseteq r$;
- $r(a, a_{A,B})$ for all $a \in \mathsf{Ind}(\mathcal{A})$ such that $\mathcal{T} \cup \mathcal{A} \models \exists s.A(a)$, $B = \mathsf{Ran}(s)$, and $\mathcal{T} \models s \sqsubseteq r$;
- $r(a_{A,B}, a_{A',B'})$ for all A, B, A', B' such that $\mathcal{T} \models A \sqcap B \sqsubseteq \exists s.A'$, $B' = \mathsf{Ran}(s)$, and $\mathcal{T} \models s \sqsubseteq r$.

It is routine to verify that $\Pi_\mathcal{T}(\mathcal{A}') = \mathcal{A}'_\mathcal{T}$, for every $\mathcal{A}' \subseteq \mathcal{A}$. Thus, in order to prove Property $(*)$ it suffices to note that the rules in $\Pi_{\mathcal{T},\varphi}$ constructed in the classical combined approach do not use symbols from $\mathcal{T}$, $\mathcal{A}$, φ in the head.

Lemma 1 has two important consequences. First, we can use any probabilistic logic programming system to compute answer probabilities for $\mathcal{ELH}^{dr}$ OMQs over ipABoxes. Second, it provides a polynomial time reduction to *weighted model counting* over propositional formulas [8], which is interesting from both a theoretical and practical perspective. Our proof is similar to what has been done in [10]. We use standard notation for propositional formulas. A *weight function* W for a propositional formula χ over variables $x_1, \ldots, x_n$, assigns a value $W(\ell)$ to every literal ℓ over $x_1, \ldots, x_n$. The *weight of a variable assignment* π, denoted $W(\pi)$, is defined as $\Pi_{i=1}^n W(\ell_i)$ where $\ell_i = x_i$ if $\pi(x_i) = 1$ and $\ell_i = \neg x_i$, otherwise. The *weight of a formula*, denoted $W(\chi)$ is then the sum of the weights of all satisfying assignments for χ.

Lemma 2. *For every* $\mathcal{ELH}^{dr}$ *OMQ* $(\mathcal{T}, \varphi)$*, ipABox* $(\mathcal{A}, p)$*, and possible answer* $\mathbf{a}$*, one can compute in polynomial time a propositional formula* χ *and a weight function* W *such that* $W(\chi) = Pr_{\mathcal{A},p}(\mathcal{T}, \varphi, \mathbf{a})$.

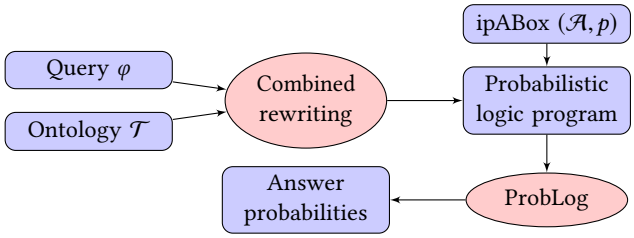

Figure 1: The architecture of our approach.

Proof. Let Π be the PLP $\Pi_{\mathcal{T},\varphi} \cup \Pi_{\mathcal{T}}$ from Lemma 1; its size is polynomial in the size of $\mathcal{T}$ and independent from $\mathcal{A}$. Let further $\Pi_1, \ldots, \Pi_k$ denote the strata of Π. We first construct the *groundings* $G_1, \ldots, G_k$ of $\Pi_1, \ldots, \Pi_k$. Introduce a propositional variable x_f for every ground fact f that can be obtained by instantiating an atom in Π with individuals from $\mathsf{Ind}(\mathcal{A})$. Then, G_i is the set of clauses

$$x_{g(H)} \leftarrow x_{g(B_1)} \wedge \ldots \wedge x_{g(B_m)} \wedge \neg x_{g(B_{m+1})} \wedge \ldots \wedge \neg x_{g(B_n)}$$

for all $H \leftarrow B_1, \ldots, B_m, \neg B_{m+1}, \ldots, \neg B_n$ in Π_i with free variables $\mathbf{x}$, and all mappings $g : \mathbf{x} \to \mathsf{Ind}(\mathcal{A})$. By construction, $G := \bigcup_i G_i$ is a stratified propositional Horn formula with strata $G_1, \ldots, G_k$, which satisfies that, for every $\mathcal{A}' \subseteq \mathcal{A}$, $\Pi(\mathcal{A}')$ corresponds precisely to the minimal supported model π of $G \cup \{x_f \mid f \in \mathcal{A}'\}$.

To arrive at χ, we "simulate" the computation of the minimal supported model of G. For every $j \in \{1, \ldots, k\}$, let H_j be the set of variables that occur as head in G_j, and create copies x^i, $i \in \{1, \ldots, |H_j|\}$, for every $x \in H_j$. Then, $\chi = \chi_0 \wedge \chi_1 \wedge \chi_2$ where $\chi_0 = x^N_{\mathrm{Goal}(\mathbf{a})}$ with $N := |H_k|$, and χ_1, χ_2 are defined as follows. Formula χ_1 uses additional variables y_f, for every $f \in \mathcal{A}$, and is defined as $\chi_1 = \bigwedge_{f \in \mathcal{A}} (x^1_f \leftrightarrow y_f)$. Formula χ_2 has a conjunct χ_f for every variable $x_f \in H_j$ for some $j \in \{1, \ldots, k\}$. To define χ_f, let $r_1, \ldots, r_k$ be the bodies of all rules in G_j with head x_f. For some rule body $r \in \{r_1, \ldots, r_k\}$, we denote with r^{i-1} the variant of r where every variable x with $x \in H_j$ is replaced with x^{i-1}, and every variable x with $x \in H_{j'}$, $j' < j$ is replaced with x^N where $N = |H_{j'-1}|$. Now, χ_f is defined as:

$$\chi_f = \bigwedge_{i=2}^{N} \left(x^i_f \leftrightarrow (x^{i-1}_f \vee \bigvee_{j=1}^{k} r^{i-1}_j) \right)$$

Intuitively, x^i_f with $x_f \in H_j$ is forced to be true iff it can be derived using at most i applications of rules from G_j. That is, it is true from the beginning (via χ_1), or it can be derived using some rule that depends only on variables with lower subscript $i-1$ (via χ_2). It remains to define the weight function W as follows:

- $W(y_f) = p(f)$ and $W(\neg y_f) = 1 - p(f)$, for every $f \in \mathcal{A}$, and
- $W(x) = W(\neg x) = 1$, for all other variables x used in χ.

It can be verified that χ and W are as required. □

4 IMPLEMENTATION AND EXPERIMENTS

Lemma 1 immediately gives rise to an implementation of OMQPD with $\mathcal{ELH}^{dr}$ on top of a probabilistic logic programming system supporting marginal inference. More precisely, on input $(\mathcal{T}, \varphi)$, $(\mathcal{A}, p)$ and $\mathbf{a}$, we construct the PLP $\mathcal{P} = (\Pi_{\mathcal{T},\varphi} \cup \Pi_{\mathcal{T}}, \mathcal{A}, p)$, feed the system with $\mathcal{P}$, and query the marginal probability of $\mathrm{Goal}(\mathbf{a})$.

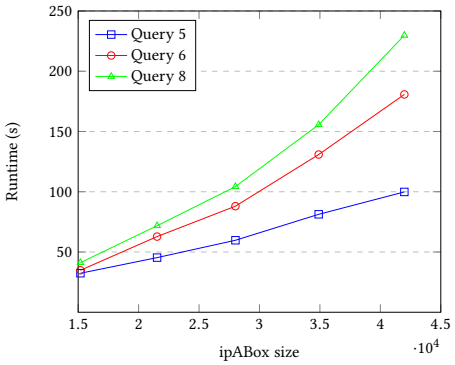

Figure 2: Total inference time with the combined approach on various ipABox sizes, using classic inference.

As a probabilistic logic programming system, we use ProbLog [14], which has a sufficiently expressive language with the required semantics. Moreover, it supports marginal inference via a variety of different algorithms. The overall architecture of our approach is depicted in Figure 1. For our experiments, we used two different inference methods: (i) the "classic" ProbLog inference approach of cycle-breaking and compilation to sentential decision diagrams (SDDs) [18], and (ii) T_P-compilation to SDDs, which avoids the cycle-breaking step altogether through forward inference [17]. Regardless of the method used, ProbLog first computes the ground program relevant to the query.

We conducted our experiments on a probabilistic version of the Lehigh University Benchmark (LUBM) [11]. LUBM is a benchmark for measuring the performance of semantic knowledge base systems in a consistent manner, comprising an ontology, data generation tool, and a set of test queries. For the purposes of our experiments, we dropped transitive and inverse role declarations from the ontology in order to obtain a valid $\mathcal{ELH}^{dr}$-ontology. We set the parameters of the original data generation tool to generate an ABox of cardinality 15189. Of this, 12260 statements were role assertions and the remainder were concept assertions.

We wrote scripts to transform the assertions generated by the data generation tool to probabilistic facts in ProbLog. As the data from the tool is deterministic by default, we enriched the output by associating each ABox statement with an indepedent, uniformly drawn probability $X_i \sim \mathcal{U}(0, 1)$ to obtain an ipABox. Finally, we computed the combined rewritings of each of the LUBM queries with respect to the ontology. As first-order rewritings exist for all of the queries we considered, we manually computed these rewritings for comparison purposes. Queries 11, 12, and 13 were deliberately omitted as they are specifically designed to test reasoning with inverse and transitive role declarations, which as mentioned earlier are unsupported in our ontology language. The results of applying ProbLog to the rewritten LUBM queries can be found in Table 1.

Interestingly, we see that, particularly in the combined approach, most of the time is spent in the grounding step rather than the knowledge compilation step for each query. These steps correspond to the (deterministic) query answering phase and probability computation phase, respectively. This means that a large amount of

Table 1: Grounding and compilation runtime for the Lehigh University Benchmark queries. All times are in seconds.

	Combined approach				First-order rewriting			
			Classic inference				Classic inference	
Query	Grounding	T_P-compilation	Cycle-breaking	Compilation	Grounding	T_P-compilation	Cycle-breaking	Compilation
1	0.00	0.00	0.00	0.00	0.04	0.05	0.00	0.00
2	70.14	5.17	0.00	0.00	28.82	0.11	0.00	0.00
3	0.03	0.00	0.00	0.00	0.59	0.67	0.00	0.00
4	25.60	5.73	0.02	0.03	0.88	0.95	0.02	0.03
5	28.24	28.04	1.60	2.53	2.39	5.66	0.40	1.05
6	25.61	71.23	2.92	6.30	4.09	50.12	2.23	5.67
7	78.49	6.26	0.04	0.05	4.53	5.44	0.02	0.05
8	30.24	92.90	3.46	7.47	6.19	71.90	2.54	6.91
9	Timeout	–	–	–	Timeout	–	–	–
10	27.28	4.85	0.00	0.00	4.35	4.63	0.01	0.03
14	0.32	0.12	0.01	0.03	0.20	0.13	0.00	0.00

"Timeout" indicates that the procedure took over ten minutes to run.

time is taken in the computation of the relevant ground program, which is based on SLD-resolution. As SLD-resolution is, theoretically, not a hard task, we believe this to be the result of inefficiencies in ProbLog's implementation of grounding which become apparent when dealing with large programs like the ones here.

The classic ProbLog inference method of cycle-breaking and compilation to SDDs consistently outperforms T_P-compilation. We also observe that first-order rewritings seem to have somewhat better inference times overall, as a trade-off for the incompleteness of this approach. We conclude that in practice, it may be best to first test the first-order rewritability of the query before resorting to the combined approach as a second option.

Finally, to get an indication of how our method scales, we examined the total inference time on different ipABox sizes for a subset of the queries in Table 1 for which inference appeared non-trivial. The total inference time here is the sum of grounding, cycle-breaking, and SDD compilation time. The results are shown in Figure 2.

5 CONCLUSION AND FUTURE WORK

We established a connection between OMQPD with ontologies formulated in $\mathcal{ELH}^{dr}$ and probabilistic logic programming, inspired by the combined approach known from classical ontology-mediated querying. We evaluated our approach with promising first results. There are a number of possible next steps. The results suggest that further work on ProbLog's grounding engine is needed to scale to real-world database sizes; e.g. one could use the grounding constructed in Lemma 2. One could also investigate whether our approach extends to different ontology languages. Finally, it is interesting to see whether other inference capabilities of ProbLog, such as learning, can be transferred to the OMQPD setting.

ACKNOWLEDGMENTS

This work has received funding from the Research Foundation - Flanders (grant G042815N), and from the European Research Council under the European Union's Horizon 2020 research and innovation programme (grant 694980).

REFERENCES

[1] 2012. *OWL 2 Web Ontology Language Profiles* (2 ed.). W3C Recommendation. W3C. http://www.w3.org/TR/2012/REC-owl2-profiles-20121211/.
[2] Marco Alberti, Elena Bellodi, Giuseppe Cota, Fabrizio Riguzzi, and Riccardo Zese. 2017. cplint on SWISH: Probabilistic Logical Inference with a Web Browser. *Intelligenza Artificiale* 11, 1 (2017), 47–64.
[3] Krzysztof R. Apt, Howard A. Blair, and Adrian Walker. 1988. Towards a Theory of Declarative Knowledge. In *Foundations of Deductive Databases and Logic Programming*, Jack Minker (Ed.). Morgan Kaufmann, 89–148.
[4] Alessandro Artale, Diego Calvanese, Roman Kontchakov, and Michael Zakharyaschev. 2009. The DL-lite Family and Relations. *J. Artif. Int. Res.* 36, 1 (2009), 1–69.
[5] Franz Baader, Sebastian Brandt, and Carsten Lutz. 2005. Pushing the EL Envelope. In *Proceedings of IJCAI 2005*. 364–369.
[6] Franz Baader, Ian Horrocks, Carsten Lutz, and Ulrike Sattler. 2017. *An Introduction to Description Logic*. Cambridge University Press.
[7] Meghyn Bienvenu, Peter Hansen, Carsten Lutz, and Frank Wolter. 2016. First Order-Rewritability and Containment of Conjunctive Queries in Horn Description Logics. In *Proceedings of IJCAI 2016*. 965–971.
[8] Mark Chavira and Adnan Darwiche. 2008. On probabilistic inference by weighted model counting. *Artif. Intell.* 172, 6-7 (2008), 772–799.
[9] Luc De Raedt and Angelika Kimmig. 2015. Probabilistic (logic) programming concepts. *Machine Learning* 100, 1 (2015), 5–47.
[10] Daan Fierens, Guy Van den Broeck, Joris Renkens, Dimitar Shterionov, Bernd Gutmann, Ingo Thon, Gerda Janssens, and Luc De Raedt. 2015. Inference and learning in probabilistic logic programs using weighted Boolean formulas. *TPLP* 15, 3 (2015), 358–401.
[11] Yuanbo Guo, Zhengxiang Pan, and Jeff Heflin. 2005. LUBM: A benchmark for OWL knowledge base systems. *J. Web Semant.* 3, 2-3 (2005), 158–182.
[12] Jean Christoph Jung and Carsten Lutz. 2012. Ontology-Based Access to Probabilistic Data with OWL QL. In *Proceedings of ISWC 2012*. Springer, 182–197.
[13] Carsten Lutz, David Toman, and Frank Wolter. 2009. Conjunctive Query Answering in the Description Logic EL Using a Relational Database System. In *Proceedings of IJCAI 2009*. 2070–2075.
[14] Luc De Raedt, Angelika Kimmig, and Hannu Toivonen. 2007. ProbLog: A Probabilistic Prolog and Its Application in Link Discovery. In *Proceedings of IJCAI 2007*. 2462–2467.
[15] Taisuke Sato and Yoshitaka Kameya. 1997. PRISM: a language for symbolic-statistical modeling. In *Proceedings of IJCAI 1997*. 1330–1335.
[16] Joerg Schoenfisch and Heiner Stuckenschmidt. 2015. Towards Large-Scale Probabilistic OBDA. In *Proceedings of SUM 2015*. Springer, 106–120.
[17] Jonas Vlasselaer, Guy Van den Broeck, Angelika Kimmig, Wannes Meert, and Luc De Raedt. 2016. T_P-Compilation for inference in probabilistic logic programs. *Int. J. Approx. Reasoning* 78 (2016), 15–32.
[18] Jonas Vlasselaer, Joris Renkens, Guy Van den Broeck, and Luc De Raedt. 2014. Compiling probabilistic logic programs into sentential decision diagrams. In *Workshop on Probabilistic Logic Programming (PLP), Vienna*.