Information Integration with Bayesian Description Logic Programs

Livia Predoiu
Digital Enterprise Research Institute
Technikerstr. 21a
Innsbruck, Austria
livia.predoiu@deri.org

ABSTRACT

In this paper, we present a novel approach for information integration usable in many different web and network environments. The knowledge representation formalism of Description Logic Programs (DLPs) is well known in the Semantic Web community as a qualified Information Integration language. This is due to its property of being the intersection between the established knowledge representation formalisms of Description Logics and Logic Programming. In this paper, we take the DLPs a step further and extend them with probabilities. Furthermore, we suggest a framework for Information Integration by using these so-called *Bayesian Description Logic Programs* (Bayesian DLPs). As most ontology mapping tools are inexact, a knowledge representation formalism that considers the uncertainty of mappings enhances the automatism of information integration and overcomes the worst bottleneck in Information Integration, namely the need for human intervention.

Categories and Subject Descriptors

D.I.2.4 [**Knowledge Representation Formalisms and Methods**]: Miscellaneous

General Terms

Languages, Theory

Keywords

Ontology Mediation, Information Integration, Description Logics, Logic Programming, Probabilistic Logic Programming

1. INTRODUCTION

Information Integration is crucial for an effective and efficient utilization of the huge amount of information that is available in the World Wide Web. The representation of information provided by web sites and databases connected to the Internet is not bound to a specific representation language or representation structure, and similar or even the same information might be represented in different systems. These different representations need to be aligned in order to enable the utilization of this information in a coherent manner. Thus, the heterogeneity of the systems connected to the internet and the heterogeneity of information representations available herein necessitates good means for Information Integration. As ontologies capture the semantics of data and thus the information inside, they are not only very useful for representing information in general but also very promising for semantically integrating information distributed on the web or within heterogeneous ontology networks.

The languages that can be found currently in the World Wide Web context range usually around the Description Logics and Logic Programming knowledge representation paradigms. While OWL which represents the Description Logics paradigm (OWL-Lite and OWL-DL closely correspond to $\mathcal{SHIF}(D)$ and $\mathcal{SHOIN}(D)$, respectively, cf. [8]) is already a W3C recommendation [1], Logic Programming paradigms gain more and more attention and support by the community. Some examples for Logic Programming languages intended for the usage on the World Wide Web are Rule-ML Logic Programs[1], WRL[2] and TRIPLE [3]. There is also a W3C working group charter which works on a rule interchange format[3] which shows the huge amount of interest to use Logic Programming for the specification of information for the Internet.

Hence, the ontologies that need to be integrated are expected to be represented mainly by some Description Logics or some Logic Programming variants. In [7], a knowledge representation formalism called *Description Logic Programs* (DLPs) has been introduced to the Semantic Web community. The knowledge representation formalism of DLPs represents the expressive intersection between Description Logics and Logic Programming. This language is therefore well-suited for information integration purposes. Information Integration can be performed by means of Description Logic Programs by translating the heterogeneous ontologies to be integrated into the knowledge representation formalism of DLPs. Note that such a translation requires the parts being translated to be syntactical variants of DLPs, i.e. lying semantically in the intersection of LP and DL. The parts of the ontologies that lie outside of this intersection cannot be translated and can therefore not be integrated. If the ontologies are represented syntactically within the same knowledge representation formalism, they can be queried by reasoning formalisms developed for the knowledge representation formalism at hand.

After translating the ontologies into the same knowledge representation formalism, in our case Description Logic Pro-

[1]http://www.ruleml.org
[2]http://www.w3.org/Submission/WRL/
[3]http://www.w3.org/2005/rules/wg

grams, mappings between the entities of the ontologies can be more easily discovered. It is beneficial to have mappings between the entities of the ontologies represented within the same knowledge representation formalism like the ontologies themselves because the mappings and the ontologies can be unified into one big ontology and fed into a reasoner. In the ideal case, the mappings should be discovered by means of an automatic mapping discovery tool. Most current approaches for ontology mapping (discovery) are semi-automatic and require human intervention (cf. [11]). The need for human intervention does not scale for the purpose of Information Integration within the setting of the World Wide Web with its huge amount of resources. Thus, a knowledge representation formalism that takes into account uncertainty helps to overcome the worst bottleneck of Information Integration, the need of human intervention.

In this paper, we present a probabilistic extension of DLPs. This extension is a subset of a knowledge representation formalism called *Bayesian Logic Programs* (BLPs) [13]. We define a restriction of BLPs which is usable for Information Integration of information represented by heterogeneous networks of Description Logics and Logic Programming ontologies. We call this knowledge representation formalism *Bayesian Description Logic Programs* (BDLPs) and provide a framework for integrating ontologies by means of BDLPs.

The rest of the paper is organized as follows. We present a short survey on Description Logic Programs in section 2. In section 3, we introduce Bayesian Description Logic Programs by first surveying Bayesian Logic Programs in section 3.1 which are a formal probabilistic and logical formalism on which Bayesian Description Logic Programs are grounded. Then we introduce Bayesian Description Logic Programs in section 3.2. Afterwards, in section 4, we present a framework for information integration by means of Bayesian Description Logic Programs. In section 5 we discuss related work. Finally, we present our conclusions in section 6.

2. DESCRIPTION LOGIC PROGRAMS

Semantically, the knowledge representation formalism of Description Logic Programs (DLPs) [7] is the expressive intersection of the knowledge representation formalisms of Description Logics (DL) [2] and of Logic Programming (LP) [16] which are the prevalent logical formalisms used within the Semantic Web community. *Syntactically*, the DLP language corresponds to Datalog without negation and without equality. The difference between Datalog and function-free definite clause logic (i.e. the function-free Horn-Logic fragment of First-Order Logic) is that only fact-form conclusions can be drawn. Thus, the DLP fragment is called the expressive *f-subset* of equality-free and function-free definite clause Logic. This means that it is a mild weakening regarding entailment power as it permits only fact-form conclusions. In [7], this mild weakening concerning entailment power is called *f-weakening.*

Syntactically and semantically, DLPs correspond to Datalog without negation and without equality. Correspondingly, a Description Logic Program consists of a set of *rules* and a set of *facts*. The set of facts is usually called *extensional database* or *EDB*. The set of facts that can be derived only by the rules are called *intensional database* or *IDB*. The union of the EDB and the IDB is the least Herbrand model of the DLP.

Each rule has the form $H \leftarrow B_1 \wedge \ldots \wedge B_m$, where H and the B_i are atomic formulae and it holds that $m \geq 0$.

An atomic formula consists of a predicate symbol p followed by a bracketed n-tuple of terms t_i, $p(t_1, \ldots, t_n)$ with $n \geq i \geq 0$. A term can be either a constant (i.e. an instance) or a variable (i.e. a placeholder for an instance). If all terms in an atomic formula are constants, the atomic formula is called a *ground atom*. The left hand side of a rule, H, is called *head* and the right-hand side of a rule, $B_1 \wedge \ldots \wedge B_m$, is called *body*. All variables in rules are universally quantified, although this is not explicitly written. For $i = 0$, the body is empty and the arrow is omitted. Such a rule with an empty body is called a *fact*. We allow only ground atoms as facts.

In [7], it has been shown how to perform one direction of the so-called *DLP-fusion*, i.e. the bidirectional mapping from the DLP fragment of DL to LP and vice versa from the DLP fragment of LP to DL. Only the translation from Description Logics ontologies to DLP ontologies has been detailed. As DLPs resemble syntactically rather the LP formalism, the translation of DL to LP has been detailed. However, as can be seen later in our framework for Information Integration, we need only the translation from DL ontologies into the DLP fragment of LP. The Bayesian Description Logic Programming formalism which we present in the next section, is built on definite clause logic. Furthermore, we agree with the authors of [7] that the LP community provides much more mature, efficient and scalable algorithms and systems for reasoning than the DL community. We believe also, that in the Information Integration setting, we rather need support for queries that are typical for the LP paradigm and that are much better supported by LP reasoning engines. Such queries correspond to ABox reasoning in DL terminology. ABox reasoning with DL reasoning engines is not fast[4].

Bear in mind that information specified within the language difference sets of Description Logics and Logic Programming ontologies is lost and cannot be integrated because there is no means to translate information specified in either of them. Another knowledge representation formalism that would be well-suited for Information Integration is the least superset containing the union of the DL and the LP knowledge representation formalisms. However, currently it is not yet clear how to combine the Description Logics paradigm (in which negation is monotonic) with the Logic Programming paradigm (in which negation is nonmonotonic). Furthermore, reasoning with the much less expressive DLP language is decidable and has a much lower complexity than a knowledge representation formalism which is so expressive that it unifies DL and LP. As function symbols are common in the LP paradigm, it can be expected that the knowledge representation formalism unifying LP and DL will be at least semi-decidable.

3. BAYESIAN DLPS

In [13], the knowledge representation formalism *Bayesian Logic Programs* (BLPs) have been introduced. BLPs are a knowledge representation formalism with a formal semantics that extends definite clause logic with probabilities. They

[4]Note that KAON2 which translates DL ontologies into disjunctive Datalog outperforms other common DL reasoners in ABox reasoning [18]. However, KAON2 has not been compared yet with common LP reasoners.

also correspond to an extension of Bayesian Networks [10] to first-order definite clause logic. As Bayesian Networks correspond to a probabilistic extension of sentential definite clause logic, BLPs extend them in a natural way to first-order logic. We chose BLPs for the purpose of Information Integration because they unify the paradigms of definite clause logic and Bayesian Networks in a complete and coherent manner. Both definite clause logic and Bayesian Networks are subsets of BLPs. We can therefore easily define a subset of BLPs that extends DLPs with probabilities.

In subsection 3.1, we provide an overview on BLPs and how inference can be done with BLPs. In subsection 3.2, we define the subset of BLPs that correspond to a probabilistic extension of DLPs that can be used for Information Integration on the Web and within networks of DL and LP ontologies.

3.1 Bayesian Logic Programs

Bayesian Logic Programs (BLPs) have been introduced in [13]. The knowledge representation formalism of BLPs combines the principle of Logic Programming with Bayesian Networks (BN) [10]. Bayesian Networks are a very important, efficient and elegant framework for representing and reasoning with probabilistic models. Thus, there are mature and efficient inference engine implementations for Bayesian Networks available that can be exploited also for Information Integration purposes. A Bayesian Network is a directed, acyclic graph where the nodes correspond to random variables and the arcs correspond to direct influences. To each node in the Bayesian Network, a conditional probability density is attached which specifies the probability of each of the states of the random variable corresponding to the node given each of the states of it's immediate parents.

In section 3.1.1, we provide a short survey on BLPs and in section 3.1.2, we describe how queries to Bayesian Logic Programs are processed and answered. For a more detailed description of BLPs, we refer the interested reader to [13].

3.1.1 The Logical Formalism of BLPs

Bayesian Logic Programs unify first-order definite clause logic with Bayesian Networks. To each rule, a conditional probability density is attached that specifies the probability of the head atom given the body atoms. A fact is associated with an a-priori probability. If the (conditional or a-priori) probability densities are not considered, the BLP corresponds to a common Prolog program without negation and without equality. In [13], the set of logical definite clauses corresponding to the set of clauses in a Bayesian Logic Program B is called the *corresponding logic program* $\tilde{B}$.

Compared to DLPs, the logical language used to build up the corresponding logic programs matches to Description Logic Programs, extended with the addition of function symbols in term composition. This means that in BLPs, terms can be built recursively in a more complex manner with *constants* being valid terms, *variables* being valid terms and *function symbols* followed by a bracketed n-tuple of terms t_i ($f(t_1, \ldots, t_n)$ with $n \geq i \geq 0$) are also valid terms. I.e. they correspond rather to Prolog without negation and without equality than to Datalog like DLPs do. Another difference is that instead of the arrow, the symbol "|" is used in order to hint at the idea of conditional probability densities.

If additionally to the language of the corresponding logic programs the (conditional or a-priori) probability densities are considered as well, a BLP B can be seen to encode a Bayesian Network in the same way, a set of clauses in first-order definite clause logic encodes a set of clauses in sentential definite clause logic. Hence, BLPs can be seen from two perspectives, one corresponding to pure definite clause logic and one corresponding to pure Bayesian Networks.

Another important difference between BLPs and DLPs is that the atoms $p(t_1, \ldots, t_n)$ are *bayesian* which means that they are not boolean but are associated to an arbitrary finite domain $D(p)$. To each predicate p_i of the BLP, a unique domain $D(p_i)$ is associated. All ground atoms inherit the domain that belongs to their predicate and this basically means that a a ground atom can have a state from an arbitrary set of states, the domain $D(p)$ that belongs to its predicate p. Seen from the perspective of Bayesian Networks, a ground atom corresponds to a random variable and thus to a node in the Bayesian Network. Hence, a non ground atom in a Bayesian Logic Program can be called *bayesian atom* and generically represents a set of random variables.

We introduce now the probabilistic aspect in BLPs. As mentioned above, each rule and fact that is contained within a BLP has a conditional or a-priori probability density attached. The probability of each of the possible states of the head atom is conditioned on each of the states of the body atoms. Seen from the perspective of a Bayesian Network, a clause with a nonempty body in a BLP encodes a direct influence relationship, i.e. if the clause is ground and valid, each atom corresponds to a node in the Bayesian Network and there is an arc from each of the body atoms to the head atom.

Let r be a rule in a BLP: $r : h(t_{h1}, \ldots, t_{hn})|b_1(t_{11}, \ldots, t_{1n}), \ldots, b_n(t_{n1}, \ldots, t_{nn})$. Let $D(h) = \{h_1, \ldots, h_m\}$ be the domain that the ground atoms containing the predicate h are associated to. Then, for each combination of states the ground body atoms can have $(e_1, \ldots, e_n) \in D(b_1) \times \ldots \times D(b_n)$ and each state of the ground head atom h_i a function $cpd(r)(h_i|e_1, \ldots, e_n) : D(h) \mapsto [0, 1]$ is given. This function is the conditional probability density of each of the random variables that are represented by the direct influence relationship between ground atoms encoded by such a rule. Note that rules with empty bodies are facts and for a fact f with $f \equiv h(t_{h1}, \ldots, t_{hn})$ the a-priori probability density is given in the same way.

The last important structural element in the knowledge representation formalism of Bayesian Logic Programs is the construct of so-called *combining rules*. Combining rules are needed in BLPs because there is a one-to-one correspondence between ground atoms and random variables or nodes in the Bayesian Network, respectively. If we have e.g. two rules with the same predicate in the head atom, it might be possible to unify the head atoms of both rules with the same ground atom. So, if the bodies of the two rules can be completely grounded in a valid way, the ground body atoms of both rules are direct parents of the ground head atom in the corresponding Bayesian Network. Then, the conditional probability density of the random variable that corresponds to the ground head atom needs to consider the possible states of all body atoms at once.

A combining rule is an algorithm that maps a finite set of conditional probability densities $\{p(h_i|a_{i1}, \ldots, a_{in_i})|m \geq i \geq 1, n_i \leq 0\}$, $m \geq 1$, to the conditional probability densities $p(h|b_1, \ldots, b_n)$ with $\{b_1, \ldots, b_n\} \subseteq \cup_{i=1}^{m}\{a_{i1}, \ldots, a_{in_i}\}$. As explained above, the combining rules are important to ensure that random variables that get e.g. by means of a reasoning process more direct parents, get also a valid conditional probability density. Combining rules can be different algorithms. [13] mentions as most simple combining rule e.g. the usage of the maximum of the former probability densities.

Semantically, a BLP B corresponds to a compressed representation of a Bayesian Network. The proofs for the correspondance of BLPs to Bayesian Networks can be found in [13].

3.1.2 *Reasoning with BLPs*

The processing of queries posed to BLPs consists of two steps of a so-called *knowledge-based model construction* policy. First, an LP reasoner has to be used to build the least Herbrand Model (or only the relevant part of it by means of top-down reasoning or Backward-Chaining) of the corresponding logic program $\tilde{B}$ of an BLP B. Then, by means of substituting the ground atoms of the least Herbrand Model in all possible but valid ways into the rules, the corresponding Bayesian Network is created.

A *probabilistic BLP query* is defined in [13] as an expression of the form $? - Q_1, \ldots, Q_n|E_1 = e_1, \ldots, E_m = e_m$. This expression queries for the conditional probability density $p(Q_1, \ldots, Q_n|E_1 = e_1, \ldots, E_m = e_m)$. It has to hold that $\{Q_1, \ldots, Q_n, E_1, \ldots, E_m\} \subseteq HB(B)$. Here, $HB(B)$ is the Herbrand model of the BLP B. Clearly, an answer is only defined if and only if $\{Q_1, \ldots, Q_n, E_1, \ldots, E_m\} \subseteq LHB(B)$ with $LH(B)$ the least Herbrand Model of B.

An inference engine for BLPs exists already and is called *Balios* [12]. It is written in Java and calls Sicstus Prolog to perform logical inference and a BN inference engine (e.g. HUGIN or ELVIRA) to perform probabilistic inference.

3.2 Bayesian Description Logic Programs

We define Bayesian Description Logic Programs (Bayesian DLPs or even shorter BDLPs) as a subset of BLPs. As the knowledge representation formalism of Description Logic Programs does not contain function symbols in the syntax and we want to extend DLPs only with probabilities, we restrict the syntax of BLPs first and disallow the usage of function symbols. By this means we ensure that the language of corresponding logic programs of Bayesian Description Logic Programs corresponds to the knowledge representation formalism of DLPs. Note that the only difference in the representation lies in the usage of the"|" symbol instead of the arrow "←" typical in Logic Programming.

Furthermore, we restrict the domain of each predicate to a boolean set containing just the two states *true* and *false*. By this means we ensure that we yield a logic that corresponds to common logics.

There is no restriction on the combining rule that goes further than to obey to the fact that only boolean states are allowed for the ground atoms of a BLP in order to lie in the Bayesian DLP fragment. As a combining rule we can thus use a very simple one like the already mentioned maximum or even the *noisy or* (for details cf. [13]).

There is also no restriction on the probability densities that can be used.

4. INFORMATION INTEGRATION WITH BAYESIAN DLPS

A way for integrating data sources described by Logic Programming and Description Logic ontologies is to first translate the Description Logic ontologies to Logic Programming, i.e., more specifically, into Description Logic Programs. This can be done only with the part of the Description Logic ontologies that lie in this fragment. The Logic Programming ontologies need to be rectified from elements that go syntactically beyond Description Logic Programs. This can be done e.g. by simply deleting the rules that employ such elements. Afterwards, appropriate mappings between the entities of the ontologies need to be discovered by means of some automatic mapping discovery tool, preferable one that employs a machine learning approach with a bayesian method. The reason for such a preference is that a quantitative measure of the certainty for the results of the mapping discovery process is computed by such a method as well and can be directly used. Another requirement for the mappings is, that they can be or are already represented in the logical formalism of DLPs or a fragment of DLPs. Then, the ontologies and the mappings can be seen as one huge ontology that can be queried in a coherent manner by the same reasoning methods.

We assume that the discovered mappings are rules that have a conditional probability distribution attached which indicates the certainty of the mappings. Such a mapping represented by means of a Bayesian DLP clause is for example the rule:
$o2 : woman(X)|o1 : mammal(X) \wedge o1 : female(X)$,
i.e. each instance of ontology $o1$ that is a $mammal$ and as well $female$, is a $woman$ in the second ontology $o2$. The probability $p(o2 : woman = true|o1 : mammal = true, o1 : female = true) = 0{,}8$ might for example also hold. Then, according the laws of probability theory, also $p(o2 : woman = false|o1 : mammal = true, o1 : female = true) = 0{,}2$ holds.

A framework for information integration based on Bayesian DLPs needs to be based on similar reasoning algorithms like the ones that are used for BLPs. Reasoning with BLPs bases on a so-called *knowledge-based model construction* approach. I.e. by means of a Logic Programming reasoning algorithm, the least Herbrand model (or the part of the least Herbrand Model which is relevant for the query) of the corresponding logic program is computed. The ground atoms of the Herbrand model correspond to the nodes or random variables, respectively, in the Bayesian Network that needs to be built up in order to perform probabilistic inference. For each valid and exhaustive ground atom substitution of a rule, the grounded rule corresponds to a child (ground head atom) and parents (ground body atoms) relationship in the corresponding Bayesian Network. A directed arc is included into the Bayesian Network from each of the ground body atoms to the ground head atom of the grounded rule at hand. If two different valid and ground rules exist that have the same ground head atom but different ground body atoms, the random variable that corresponds to the ground head atoms has all the ground body atoms of the rules as parents in the corresponding Bayesian Network.

The discovered mapping rules are associated with a conditional probability distribution. In order to reason with the mappings and the original ontologies in a coherent manner, we need to add a-priori and conditional probabilities to the original ontologies that have been transferred into pure Description Logic programs. This means that we need to transfer the DLP ontologies into BDLP ontologies.

The easiest is the assigment of a-priori probabilities to the facts of the program, i.e. the EDB. We simply assign for each fact $f(c)$ in the EDB the a-priori probability $p(f(c) = true) = 1$. Then, clearly, the probability of the fact being false has to be $p(f(c) = false) = 0$.

The assignment of conditional probabilities to the rules cannot be performed before the least Herbrand model (or the part of it which is relevant for the query) has been derived. As the probabilities are needed only for the inference with the Bayesian Network, this is no problem. We suggest the following algorithm for computing the conditional probabilities for ground atoms (or in the terminology of Bayesian Networks: random variables) which are intensional (or in the terminology of Bayesian Networks: which have parents): First, the least Herbrand Model (or the part of it which is relevant for the query) of the DLP ontologies and the corresponding logical program $\tilde{B}$ of the BDLP mapping program B is inferred in a coherent manner (i.e. as if the ontologies and $\tilde{B}$ are one huge ontology O). Then, the ground atoms of the inferred least Herbrand Model (or part of it, as case may be) have to be unified with the atoms of the rules in O in all possible ways such that the resulting ground rules are valid regarding satisfiability relative to O. For each such rule $r : h|b_1, \ldots, b_n$, we can assess the conditional probability as $p(h = true|b_1 = true, \ldots, b_n = true) = 1$. Hence, it holds $p(h = false|b_1 = true, \ldots, b_n = true) = 0$. A lot of state combinations of the ground body atoms are not considered in this way, because it is not possible to infer directly the head atom by these state combinations. For these state combinations the probability of $h = true$ given these states is 0 and the probability for $h = false$ given these states is 1. In this way, we assign to each relevant rule valid conditional probabilities that can be used in the Bayesian Network that corresponds to O or just to the part of O which is relevant for the query. Note that if a ground head atom can be inferred by different ground rules, the combining rule *maximum* [13] can be applied.

As an example consider the following rule of the corresponding logic program of a BDLP B: $Parent(X) \leftarrow hasChild(X, Y)$. Say the atoms $hasChild(John, Peter)$, $hasChild(John, Linda)$ and $Parent(John)$ are ground atoms contained in the least Herbrand model of the corresponding logic program $\tilde{B}$. All these ground atoms correspond to random variables in the corresponding Bayesian Network which will be set up for the reasoning and query ansering purposes. By means of these ground atoms, we can build the following ground rules:

- $Parent(John) \leftarrow hasChild(John, Peter)$
- $Parent(John) \leftarrow hasChild(John, Linda)$

Thus, the following conditional probabilities will be assigned: $p(Parent(John) = true|hasChild(John, Peter) = true) = 1$ and $p(Parent(John) = true|hasChild(John, Linda) = true) = 1$. The complementary states of the head atoms given the same states of the body atoms can be computed easily by means of the laws of probability theory. The application of the combining rule *maximum* yields finally $p(Parent(John) = true|hasChild(John, Peter) = true, hasChild(John, Linda) = oneOf(false, true)) = 1$ and as well $p(Parent(John) = true|hasChild(John, Peter) = oneOf(false, true), hasChild(John, Linda) = true) = 1$. Note that the function *oneOf* returns one item of its parameter set.

After setting up the corresponding Bayesian Network or the part of it which is relevant to the query, the query can be processed by a typical Bayesian Network inference engine. Possible queries can be e.g. to ask for the probability $p(o2 : father(X)|o1 : male(X) = false, o1 : parent(X) = true)$. By means of the Logic Programming reasoner, all possbile valid substitutions for X can be computed and the ground queries can be posed to the corresponding Bayesian Network. We can also ask simply for $p(o2 : father(X))$ and not give any evidence. Note that the query needs not to be taken into account before the Logic Programming reasoning process starts, although by considering it in advance, a smaller portion of the least Herbrand Model and thus the Bayesian Network can be built. The whole least Herbrand Model can be inferred and by this means the whole Bayesian Network be built. However, we do believe that it is much more efficient to consider the query in advance, because the least Herbrand Model of several ontologies that need to be integrated can be quite huge and thus the resulting Bayesian Network as well.

It is very likely that cycles appear in the mappings and thus also in the corresponding Bayesian Network. In [19], however, it has been shown that approximate inference for Bayesian Networks with cycles often converges and if it does, a good approximation of the correct marginals is given.

5. RELATED WORK

Different kinds of probabilistic logics have been developed, especially since the early nineties of the last century. In particular, the growth in understanding of Bayesian Networks promoted the emergence of different flavours of probabilistic logics. E.g. in [15] a probabilistic Description Logic based on Bayesian Networks has been introduced. In [20], [14] and [9] approaches have been presented that focus rather at the Logic Programming and Relational Logic paradigm. However, our intention within this paper is to present an elegant way of integrating the knowledge representation formalism of Description Logic Programs with probabilities. The knowledge representation formalism of BLPs allows to separate the logic programming paradigm from the probabilistic modelling and reasoning paradigm and thus is particularly appropriate for our aim of extending DLPs with probabilities for Information Integration purposes. As can be seen, BLPs allow a probabilistic extension of DLPs in a very straightforward and intuitive way. This is not possible with the other probabilistic logics introduced in Artificial Intelligence and Knowledge Representation. A survey on probabilistic logics that focus on the logic programming and the relational logic paradigm can be found in [21].

In [5], Datalog$_P$, a probabilistic extension of Datalog has been presented. One difference to our approach is that Datalog with stratified negation has been extended. As we want to integrate DL and LP ontologies, we do not need negation. Another difference is the probabilistic model behind Datalog$_P$ which does not extend Logic Programming in such a straightforward way like it is done in BDLPs. E.g. in Datalog$_P$ only facts have a probability attached. The facts which can be derived get constraints on the probabilities.

The semantics considers a set of possible worlds and a probability distribution on this set. In contrast, our approach views each ground fact as a random variable which can be either true or false.

Within the context of the Semantic Web, a probabilistic extension of $\mathcal{SHOQ}$ [6] and a probabilistic extension of OWL [4] have been introduced. Only in [4] a Bayesian Network has been used as underlying probabilistic reasoning mechanism. However, the knowledge representation formalism of Logic Programming has not been considered in either of these works.

An approach for Ontology Mediation by means of Bayesian Networks has been presented in [17]. However, the authors rather want to enhance existing mappings (cf. the name of the system **O**ntology **M**apping **EN**hancer). They do not provide an integrated framework on reasoning with the mappings and the ontologies. Furthermore, the language they use for capturing the mappings is very simple and very similar to RDF Schema.

6. CONCLUSIONS

In this paper, we have presented a framework for probabilistic Information Integration of information described by means of Description Logics and Logic Programming ontologies. For this purpose, we have introduced a logical formalism that is a probabilistic extension of Description Logic Programs [7]. The logical formalism we present is a subset of a logical formalism called Bayesian Logic Programs (BLPs) [13]. We restrict the knowledge representation formalism of BLPs in order to enable automatic, probabilistic Information Integration by means of a probabilistic extension of Description Logic Programs. In [7], it has been shown that Description Logic Programs are the most appropriate knowledge representation formalism for the integration of Description Logics and Logic Programming ontologies. The rationale is their feature of being the expressive intersection of both knowledge representation formalisms, DL and LP which are the prevalent knowledge representation formalisms with a formal semantical grounding in the Semantic Web community.

Our framework shows how to overcome the worst bottleneck of Information Integration, the need of human intervention. We show how automatic discovered mappings can be used to reason in an integrated and coherent manner with the information represented within the original ontologies.

7. REFERENCES

[1] OWL Web Ontology Language Reference. http://www.w3.org/TR/owl-ref/, 10 February, 2004. W3C Recommendation.

[2] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications.* Cambridge University Press, 2003.

[3] S. Decker, M. Sintek, A. Billig, N. Henze, P. Dolog, W. Nejdl, A. Harth, A. Leicher, S. Busse, J. L. Ambite, M. Weathers, G. Neumann, and U. Zdun. TRIPLE - an RDF Rule Language with Context and Use Cases. In *Rule Languages for Interoperability*, 2005.

[4] Z. Ding and Y. Peng. A Probabilistic Extension to Ontology Language OWL. In *Proceedings of HICSS '04*, page 40111.1, Washington, DC, USA, 2004. IEEE Computer Society.

[5] N. Fuhr. Probabilistic Datalog: a logic for powerful retrieval methods. In *Proceedings of SIGIR-95, 18th ACM International Conference on Research and Development in Information Retrieval*, pages 282–290, Seattle, US, 1995.

[6] R. Giugno and T. Lukasiewicz. P-$\mathcal{SHOQ}(\mathcal{D})$: A Probabilistic Extension of $\mathcal{SHOQ}(\mathcal{D})$ for Probabilistic Ontologies in the Semantic Web. In *JELIA*, pages 86–97, 2002.

[7] B. N. Grosof, I. Horrocks, R. Volz, and S. Decker. Description Logic Programs: combining logic programs with description logic. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 48–57, New York, NY, USA, 2003. ACM Press.

[8] I. Horrocks, P. F. Patel-Schneider, and F. van Harmelen. From $\mathcal{SHIQ}$ and RDF to OWL: the making of a Web Ontology Language. *J. Web Sem.*, 1(1):7–26, 2003.

[9] M. Jaeger. Relational Bayesian networks. In M. Kaufmann, editor, *Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence*, pages 266–273, 1997.

[10] F. V. Jensen. *Introduction to Bayesian Networks.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1996.

[11] Y. Kalfoglou and W. M. Schorlemmer. Ontology Mapping: The State of the Art. In *Semantic Interoperability and Integration*, 2005.

[12] K. Kersting and U. Dick. Balios - The Engine for Bayesian Logic Programs. In *PKDD*, pages 549–551, 2004.

[13] K. Kersting and L. D. Raedt. Bayesian Logic Programs. Technical report, Albert-Ludwigs University, Freiburg, 2001.

[14] D. Koller. Probabilistic relational models. volume 1634, pages 3–13, 1999. Invited paper.

[15] D. Koller, A. Y. Levy, and A. Pfeffer. P-CLASSIC: A Tractable Probablistic Description Logic. In *AAAI/IAAI*, pages 390–397, 1997.

[16] J. W. Lloyd. *Foundations of Logic Programming, 2nd Edition.* Springer, 1987.

[17] P. Mitra, N. F. Noy, and A. Jaiswal. OMEN: A Probabilistic Ontology Mapping Tool. In *International Semantic Web Conference*, pages 537–547, 2005.

[18] B. Motik and U. Sattler. Practical DL reasoning over Large ABoxes with KAON2. *Submitted for publication*, 2006.

[19] K. P. Murphy, Y. Weiss, and M. I. Jordan. Loopy Belief Propagation for Approximate Inference: An Empirical Study. In M. Kaufmann, editor, *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, 1999.

[20] L. Ngo and P. Haddawy. Probabilistic logic programming and bayesian networks. In *Asian Computing Science Conference*, pages 286–300, 1995.

[21] L. D. Raedt and K. Kersting. Probabilistic logic learning. *SIGKDD Explorations*, 5(1):31–48, 2003.