# A Probabilistic Extension to Ontology Language OWL

Zhongli Ding and Yun Peng
Department of Computer Science and Electrical Engineering
University of Maryland Baltimore County
Baltimore, Maryland 21250
{zding1, ypeng}@cs.umbc.edu
Phone: 410-455-3816
Fax: 410-455-3969

# A Probabilistic Extension to Ontology Language OWL

## Abstract

With the development of the semantic web activity, ontologies become widely used to represent the conceptualization of a domain. However, none of the existing ontology languages provides a means to capture uncertainty about the concepts, properties and instances in a domain. Probability theory is a natural choice for dealing with uncertainty. Incorporating probability theory into existing ontology languages will provide these languages additional expressive power to quantify the degree of the overlap or inclusion between two concepts, support probabilistic queries such as finding the most probable concept that a given description belongs to, and make more accurate semantic integration possible. One approach to provide such a probabilistic extension to ontology languages is to use Bayesian networks, a widely used graphic model for knowledge representation under uncertainty. In this paper, we present our on-going research on extending OWL, an ontology language recently proposed by W3C's Semantic Web Activity. First, the language is augmented to allow additional probabilistic markups, so probabilities can be attached with individual concepts and properties in an OWL ontology. Secondly, a set of translation rules is defined to convert this probabilistically annotated OWL ontology into a Bayesian network. Our probabilistic extension to OWL has clear semantics: the Bayesian network obtained will be associated with a joint probability distribution over the application domain. General Bayesian network inference procedures (e.g., belief propagation or junction tree) can be used to compute $P(C|e)$: the degree of the overlap or inclusion between a concept C and a concept represented by a description *e*. We also provide a similarity measure that can be used to find the most probable concept that a given description belongs to.

## 1. Introduction

With the development of Semantic Web [Berners-Lee 1998], ontologies have become widely used to capture the knowledge about concepts and their relations defined in a domain for information exchange and knowledge sharing. There are different definitions of the term "ontology", from philosophy, linguistics, and artificial intelligence. In our context, ontology can be formally defined as a set of vocabulary to describe the conceptualization of a particular domain [Gruber 1993]. A number of ontology definition languages have been developed over the past years [RDF; SHOE; OIL; DAML; DAML+OIL; OWL]. The most widely used among them is DAML+OIL, which is based on DAML (DARPA Agent Markup Language) and OIL (Ontology Inference Layer). OWL (Web Ontology Language) is a newly emerging standard proposed and supported by W3C for defining ontologies in semantic web. OWL is based on DAML+OIL but with simpler primitives. Like DAML+OIL [Horrocks 2002], OWL is also based on description logics, which is a subset of first-order logic that provides sound and decidable reasoning support. As with traditional crisp logic, any sentence in OWL, being asserted facts, domain knowledge, or reasoning results, must be either true or false and nothing in between. However, most real world domains contain uncertainty knowledge and incomplete or imprecise information that is true only to a certain degree. Ontologies defined by these languages thus cannot quantify the degree of the overlap or inclusion between two concepts, and cannot support reasoning in which only partial information about a concept or individual in the domain can be obtained. Uncertainty becomes more prevalent when more than on ontologies are involved where it is often the case that a concept defined in on ontology can only find partial matches to one or more concepts in another ontology.

To overcome the difficulty arising from the crisp logics, existing ontology languages need to be extended to be able to capture uncertainty knowledge about the concepts, properties and instances in the domain and to support reasoning with partial, imprecise information. Along this direction, researchers in the past have attempted to apply different formalisms such as Fuzzy logic [Zadeh 1965], rough set theory [Pawlak 1982] and Bayesian probability as well as ad hoc heuristics into ontology definition and reasoning (see [Stuckenschmidt 2000] for a brief survey). In this paper, we take Bayesian probability approach and propose a probabilistic extension to ontology language OWL by using Bayesian networks [Pearl 1988] as the underlying probabilistic model and the reasoning mechanism. Our ultimate goal is to develop methodology based on which one can convert an OWL ontology to a Bayesian network. Bayesian networks (BN) have been established as an effective and principled general framework for knowledge representation and inference under uncertainty. In the BN framework, the interdependence relationships among random variables in a domain are represented by the network structure, and the uncertainty of such relationships by the conditional probability table (CPT) associated with each variable. These local CPT collectively and compactly encode the joint probability distribution of all variables in the system. This extension thus provides OWL additional expressive power, support probabilistic queries, and make more accurate semantic integration possible.

In our approach, the OWL language is first augmented to allow additional probabilistic markups so that probability values can be attached to individual concepts and properties as well as their interrelations in an OWL ontology. Secondly, a set of translation rules is defined to convert this probabilistically annotated ontology into a Bayesian network. The Bayesian network obtained will be associated with a joint probability distribution over the application domain, represented by the conditional probability tables associated with individual nodes (concepts and properties). General Bayesian network inference procedures (for example, belief propagation [Pearl 1986], junction tree [Lauritzen and Spiegelhalter 1988]) can be used to compute P(C|$e$): the degree of the overlap or inclusion between a concept C and a concept represented by a description $e$. It can also support other reasoning tasks such as finding the most probable concept that a given description belongs to.

Earliest works have tried to add probabilities into full first-order logic [Bacchus 1990; Halpern 1990], in which syntax of adding probabilities into the logic are defined and semantics of the result formalisms are clarified, but the logic was highly undecidable just as pure first-order logic. An alternative direction is to integrate probabilities into less expressive subsets of first-order logic such as rule-based (for example, probabilistic horn abduction [Poole 1993]) and description logics ([Jaeger 1994; Heinsohn 1994; Koller et.al 1997; Giugno and Lukasiewicz 2002]) systems. Works in the latter category is particularly relevant to our work because 1) description logic, as a subset of first-order logic, is decidable and provides sound inference mechanism; and 2) OWL and several other ontology languages are based on description logic. Works in [Jaeger 1994; Heinsohn 1994] provide a probabilistic extension of the description logic $\mathcal{ALC}$ based on probabilistic logics. P-CLASSIC [Koller et.al 1997] gives an informal probabilistic extension of description logic CLASSIC (with some variations) based on Bayesian networks, in which each probabilistic component is associated with a set $\mathcal{P}$ of p-classes, each p-class $P \in \mathcal{P}$ is represented using a Bayesian network $N_P$. P-*SHOQ*(D) [Giugno and Lukasiewicz 2002] is the probabilistic extension of description logic

*SHOQ*(D) [Horrocks 2001], which is the semantics behind DAML+OIL (without inverse roles), based on the notion of probabilistic lexicographic entailment from probabilistic default reasoning. Among these works, only P-*SHOQ*(D) is able to represent assertional (i.e., Abox) probabilistic knowledge about concept and role instances, while only P-CLASSIC uses Bayesian networks as underlying probabilistic reasoning mechanism. The primary difference between our work and the others such as P-CLASSIC is that we are not aimed at providing additional means to represent uncertainty or probabilistic aspect of the domain but rather at developing formal rules to directly translate an OWL ontology into a Bayesian network.

The rest of this paper is organized as follows. Section 2 provides a brief description of OWL language; Section 3 contains the most of our work on how to represent an OWL ontology by a Bayesian network, including the extension of OWL for probabilistic annotation and a set of rules converting OWL constructors into nodes and arcs of Bayesian network; Section 4 discusses issues concerning the semantics of the Bayesian network generated by the converting rules and how reasoning can be performed over this probabilistic-annotated ontology. The paper concluded in Section 5 with directions for future research.

## 2. Web Ontology Language OWL

In this section, we briefly review OWL, the standard web ontology language proposed by W3C recently. OWL is intended to be used by applications to represent terms and their interrelationships. It is an extension of RDF (the Resource Description Framework [RDF]) and goes beyond its semantics. RDF is a general assertional model to represent the resources availble on the web through RDF triples of "*subject*", "*predicate*" and "*object*". For example, in the following RDF statement,

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
            xmlns:dc="http://purl.org/dc/elements/1.1/">
   <rdf:Description rdf:about=" http://www.w3.org/">
      <dc:title>World Wide Web Consortium</dc:title>
   </rdf:Description>
</rdf:RDF>
```

"http://www.w3.org/" is the "subject", "dc:title" is the "predicate" (where "dc" is an abreviation of "http://purl.org/dc/elements/1.1/"), and "World Wide Web Consortium" is the "object". This triple is read as: "http://www.w3.org/" has a title "World Wide Web Consortium". Each RDF triple can be encoded in XML as shown in the above example. It can also be represented as the "RDF graph" in which nodes correspond to "subject" and "object" and the directed arc corresponds to the "predicate" as shown in Figure 2.1 below:

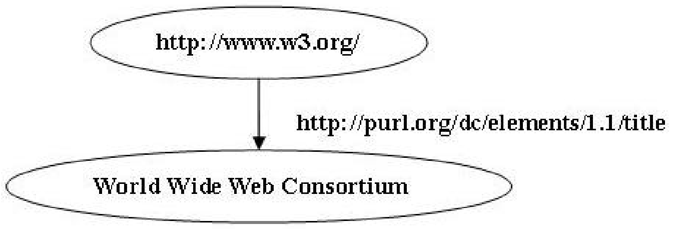

http://www.w3.org/
http://purl.org/dc/elements/1.1/title
World Wide Web Consortium

Figure 2.1 RDF Graph Example

Each triple in RDF makes a distinct assertion, adding any other triples will not change the meaning of the existing triples. A simple datatyping model of RDF called RDF Schema [RDFS] is used to control the set of terms, properties, domains and ranges of properties, and "subClassOf" and "subPropertyOf" relationships used to define resources. However, RDF Schema is not expressive enough to catch all the relationships between classes and properties. OWL provides a richer set of vocabulary by further restricting on the set of triples that can be represented. OWL includes three increasingly complex languages [OWL GUIDE]: OWL Lite, OWL DL and OWL Full. An OWL document can include an optional ontology header and any number of class, property, and individual descriptions or axioms. Next we will describe the set of constructors used in OWL to form descriptions or axioms in brief [OWL REF].

A named class in an OWL ontology can be described by a class indentifier, for example, "<owl:Class rdf:ID="Animal" />" defines a class "Animal" which is an instance of owl:Class. An anonymous class can be described by exhaustively enumerating all the individuals that form the instances of this class (owl:oneOf), by a property restriction (owl:Restriction), or by logical operation on two or more classes (owl:intersectionOf, owl:unionOf, owl:complementOf). Property restrictions include value (owl:allValuesFrom, owl:someValuesFrom, owl:hasValue) and cardinality restrictions (owl:cardinality, owl:maxCardinality, owl:minCardinality). For example, the following description is a value restriction:

```
<owl:Restriction>
   <owl:onProperty rdf:resource="#hasParent"/>
   <owl:allValuesFrom rdf:resource="#Human"/>
</owl:Restriction>
```

It defines an anonymous class of all individuals whose "hasParent" property only has range values of class "Human". In other words, the local range of property "hasParent" is restricted to "Human". Similarly, the following description is a cardinality restriction which defines a class of all individuals that have exactly 1 father:

```
<owl:Restriction>
   <owl:onProperty rdf:resource="#hasFather"/>
   <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1</owl:cardinality>
</owl:Restriction>
```

The three logical operators are corresponding to AND (conjunction), OR (disjunction) and NOT (negation) in logic, they define class of all individuals by standard set-operation: intersection, union, and complement, respectively. OWL also has three class axioms for additional necessary and sufficient conditions of a class: rdfs:subClassOf, owl:equivalentClass, owl:disjointWith. For example,

```
<owl:Class rdf:ID="Female">
  <rdfs:subClassOf rdf:resource="#Animal"/>
  <owl:disjointWith rdf:resource="#Male"/>
</owl:Class>
```

The above statement defines that class "Female" is subclass of class "Animal", and it is disjoint with class "Male" (i.e., these two classes have no individuals in common). There are two kinds of properties can be defined in OWL: object property (owl:ObjectProperty) which links individuals to individuals, and datatype property (owl:DatatypeProperty) which links individuals to data values. Similar to class, "rdfs:subPropertyOf" is used to define that a property is a subproperty of another property. In the following example, "hasParent" is an object property with global domain "Animal", which means an individual with property "hasParent" must be an individual in class "Animal"; and with global range "Animal", which means that the range values of "hasParent" must comes from class "Animal".

```
<owl:ObjectProperty rdf:ID="hasParent">
  <rdfs:domain rdf:resource="#Animal"/>
  <rdfs:range rdf:resource="#Animal"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hasFather">
  <rdfs:subPropertyOf rdf:resource="#hasParent"/>
  <rdfs:range rdf:resource="#Male"/>
</owl:ObjectProperty>
```

"hasFather" is another object property defined as a subproperty of "hasParent", it inherits the domain from "hasParent" and overrides its range to class "Male". Therefore, if an individual *a* can be related to an individual *b* by "hasFather", then *a* must also be related to *b* by "hasParent". Besides these most often used constructors, there are other constructors. For example, constructors owl:equivalentProperty, owl:inverseOf are used to relate two properties;

owl:FunctionalProperty, owl:InverseFunctionalProperty to impose cardinality restrictions on properties; owl:TransitiveProperty, owl:SymmetricProperty to specify logical characteristics of properties; and owl:sameAs, owl:sameIndividualAs, owl:differentFrom, owl:AllDifferent to relate individuals. The semantics of OWL is defined in the way analogous to the semantics of description logic [OWL SEMANTICS]. With the additional set of vocabulary (mostly as described above), one can define an ontology as a set of (restricted) RDF triples which can be represented as a RDF graph.

## 3. Probabilistic Extension to OWL

In the most general form, a BN of $n$ variables consists of a directed acyclic graph ($DAG$) of $n$ nodes and a number of arcs. Nodes $X_i$ in a $DAG$ correspond to variables, and directed arcs between two nodes represent direct causal or influential relation from one node to the other. The uncertainty of the causal relationship is represented locally by the conditional probability table ($CPT$) $P(X_i | p_i)$ associated with each node $X_i$, where $p_i$ is the parent set of $X_i$. Under a conditional independence assumption, the graphic structure of BN allows an unambiguous representation of interdependency between variables, which leads to one of the most important feature of BN: the joint probability distribution of $X = (X_1, \ldots, X_n)$ can be factored out as a product of the CPT in the network: $P(X = x) = {}_{i=1}^{n} P(x_i | p_i)$.

With the joint probability distribution, BN supports, at least in theory, any inference in the joint space. Although it has been proven that the probabilistic inference with general $DAG$ structure is $NP$-hard [Cooper 1990], BN inference algorithms such as belief propagation [Pearl 1986] and junction tree [Lauritzen and Spiegelhalter 1988] have been developed to explore the causal structure in BN for effective computation.

Besides the expressive power and rigorous and efficient probabilistic reasoning capability, we are attracted to BN in this work also by the structural similarity between the DAG of a BN and the RDF graph of OWL ontology: both of them are directed graphs, and direct correspondence exists between many nodes and arcs in the two graphs. However, a number of difficulties exist in converting an OWL ontology to a BN while maintaining its original semantics. Among them, the most noticeable are: 1) how can one encode probabilit y values in OWL; 2) how to generate a correct DAG from OWL graph, especially how to handle the anonymous classes in OWL for object properties; and 3) how to construct CPT from user provided probabilities and logical relations specified in OWL. We report in this section our initial results on overcoming these difficulties.

To focus attention, we in the current stage of our research assume that an OWL ontology uses only constructors corresponding to the terminology part (DL TBox) of description logic and uses only object properties. So, the constructors related to individuals (DL ABox) and datatypes are not considered. Another assumption made is that every object property defined in the ontology has at most one global domain and range, this assumption is made only to ease our illustration of the translation rules, since in the case of multiple domains/ranges, the actual domain/range set is the “intersectionOf” or “unionOf” those individual domains/ranges. Also, in the current stage, only ontologies defined by OWL Lite and OWL DL are considered. This is because, among other things, classes and individuals, properties and data values form disjoint domains in ontologies defined by OWL Lite and OWL DL; while in OWL Full a class can be treated as a collection of individuals and itself as a member at the same time, leading to cycles in the underlying RDF graph that are difficult to convert to the acyclic graph of BN.

We organize this section into 3 subsections. Section 3.1 describes how an OWL ontology is augmented to allow additional probabilistic markups. Section 3.2 defines a set of translation rules used to convert this probabilistically annotated ontology into a Bayesian network structure. Section 3.3 considers the rules for constructing the conditional probability tables.

## 3.1 Encoding requested probabilistic information

The model-theoretic semantics [OWL SEMANTICS] of OWL ontology treats the domain as a non-empty collection of individuals (or instances). If "A", "B" are classes, we interpret "P(A)" as the probability that an arbitrary individual belongs to class "A", "P(A|B)" as the probability that an individual of class "B" also belongs to "A", and "P(A|$\overline{B}$)" as the probability that an individual in the complement set of "B" belongs to "A". To add such uncertainty information into an existing ontology, we treat a probability as a kind of resource, and define three kinds of OWL classes (owl:Class): "PriorProbObj", "CondProbObjT", and "CondProbObjF". A probability of the form "P(A)" is defined as an instance of class "PriorProbObj", which has two mandatory properties: "hasVarible" and "hasProbValue". A probability of the form "P(A|B)" is defined as an instance of class "CondProbObjT" and a probability of the form "P(A|$\overline{B}$)" is defined as an instance of class "CondProbObjF", both have three mandatory properties: "hasCondition", "hasVariable", and "hasProbValue". "hasCondition" and "hasVariable" are object properties and "hasProbValue" is a datatype property. For example, suppose "prob" is the namespace abbreviation of the probability definitions, and "ont" is the namespace abbreviation of the original OWL ontology, if "Animal" is a class defined in the ontology, then "P(Animal) = 0.5", the prior probability that an individual object belongs to class “Animal”, can be expressed as:

```
<prob:PriorProbObj rdf:ID="P(Animal)">
   <prob:hasVariable><rdf:value>&ont;Animal</rdf:value></prob:hasVariable>
   <prob:hasProbValue>0.5</prob:hasProbValue>
</prob:PriorProbObj>
```

And if “Male” is another class defined in the domain, then “P(Male|Animal) = 0.5”, the conditional probability that an individual object belongs to class “Male” given it is an “Animal”, can be expressed as:

```
<prob:CondProbObjT rdf:ID="P(Male|Animal)">
   <prob:hasCondition><rdf:value>&ont;Animal</rdf:value></prob:hasCondition>
   <prob:hasVariable><rdf:value>&ont;Male</rdf:value></prob:hasVariable>
   <prob:hasProbValue>0.5</prob:hasProbValue>
</prob:CondProbObjT>
```

And "P(Female| $\overline{Male}$) = 0.32", the conditional probability that an individual object belongs to class “Female” given it is not a “Male”, can be expressed as:

```
<prob:CondProbObjF rdf:ID="P(Female|(not)Male)">
   <prob:hasCondition><rdf:value>&ont;Male</rdf:value></prob:hasCondition>
   <prob:hasVariable><rdf:value>&ont;Female</rdf:value></prob:hasVariable>
   <prob:hasProbValue>0.32</prob:hasProbValue>
</prob:CondProbObjF>
```

Currently only these three forms of probabilities are allowed to be added into an existing OWL ontology. Note that “$P(\overline{A}) = 1 - P(A)$”, “$P(\overline{A}|B) = 1 - P(A|B)$” and “$P(\overline{A}|\overline{B}) = 1 - P(A|\overline{B})$”, so there is no need to specify them explicitly. In the future, it is possible to extend the above forms to arbitrary conditional probability such as “P(A|P1,P2,…,Pn)”.

## 3.2 Structural translation rules based on set-theoretic approach

The ontology file augmented with probability values as described in the previous subsection will still be an OWL file. This probabilistically annotated ontology can be further translated into a Bayesian belief network by following a set of rules defined. In this subsection, we are focusing on the translation of the network structure, i.e., construction of the DAG from the ontology file, and leave the task of constructing CPT to the next subsection. For simplicity, we ignore those header components in the ontology, such as "owl:imports" (for convenience, we assume the whole ontology is defined in a single OWL file), "owl:versionInfo", "owl:priorVersion", "owl:backwardCompatibleWith", and "owl:incompatibleWith". If the domain of discourse is treated as a non-empty collection of individuals ("owl:Thing"), then every class (either primitive or defined) can be thought as a countable subset (or subclass) of "owl:Thing". We have developed a set of rules to translate a probability-annotated ontology into a Bayesian network structure. The general principle underlying these rules is that all classes (specified as "subjects" and "objects" in RDF triples of the OWL file) are translated into nodes in BN, and an arc is drawn between two nodes in BN only if the corresponding two classes are related by a "predicate" in the OWL file with the direction from the superclass to the subclass if it can be determined. These rules are summarized as follows:

1) Every primitive or defined class "C", or anonymous class "Res" (defined by restriction), is mapped into a two-state (either true or false) variable node in the translated BN.

2) Every object property "P" defined with domain "D" and range "R" is mapped into a two-state (either true or false) variable node "Class_P" in the translated BN, denoting the set of individuals in the domain who have this property "P" (this implies that "Class_P" should be a subclass of the domain class "D").

3) Every "Class_P" node has one "GR_P" node as its child. If the global range class specified by the ontology or inherited from superproperty of "P" is "R", then the "GR_P" node will have two states: "R" and "Irrelevant", or "Thing" if nothing is specified or inherited.

4) Similar rules are given to "Res" nodes. Recall that in OWL a class having property P is specified by an anonymous "Res" class which restricts cardinality and range. Therefore, every "Res" node in the translated BN may have either one "LC_P_Res" node (for local cardinality restriction of property P) or one "LR_P_Res" node (for local range restriction on property P) as its child, its states will correspond to the restriction defined. We call nodes such as "GR_P", "LC_P_Res" and "LR_P_Res" projection nodes, which is used to represent range and cardinality information. Projection nodes can only be leaf nodes in BN, and there is a directed arc from a "Class_P" or "Res" node to its corresponding projection node.

5) There is a directed arc from a superconcept node to a subconcept node (e.g., specified by "rdfs:subClassOf"); there is a directed arc between two mutual exclusive or disjoint concept nodes (e.g., specified by "owl:disjointWith") and the direction may in either way; there is a directed arc between two explicitly dependent concept nodes (e.g., specified by "owl:intersectionOf"), with the direction corresponding to the relation; there is no arc between two independent concept nodes, they should be d-separated with each other; and there is no arc between two implicitly dependent concept nodes, although they are not be de-separated with each other.

If no "rdfs:subClassOf" or "rdfs:subPropertyOf" cycles are allowed to be defined or no any "subClassOf" relation cycles can be inferred, and if mutual disjoint among a set of classes are carefully handled (the direction of the arcs are specified in a way without cycles), using our translation rules, the taxonomy T1 formed by all "C" and "Res" nodes (and their projection nodes) and the taxonomy T2 formed by all "Class_P" nodes (and their projection nodes) will both be acyclic and the translated network is also guaranteed to be acyclic (DAG), which links

T1 and T2 by adding directed arcs from domain class nodes in T1 to their corresponding "Class_P" nodes in T2, and from "Class_P" nodes in T2 to local restriction class nodes "Res" in T1 (note these new adding arcs will not cause cycles since no "subClassOf" relation cycles be detected). One last thing to be noted is that, for each class defined by restriction (or anonymous class) in the original ontology, we require that the ontology provides an "rdf:ID" attribute for it, since we need to name them uniquely in the translated Bayesian network and provide ways to specify probabilities for them.

We illustrate the above rules in detail by some examples. The software system we used to construct Bayesian networks in these examples is Netica from Norsys Software Corp. [NORSYS]. This system also support belief update in BN.

**Example 1:** A primitive concept "Animal" defined by a class identifier as "<owl:Class rdf:ID="Animal"/>" and denoted with a prior probability "P(Animal) = 0.5" can be mapped into a single variable node in the translated BN as:

Figure 3.1 – Single Class Only

**Example 2:** If "hasParent" is an object property with domain and range "Animal", "hasFather" is another object property which is a subproperty of "hasParent" with range "Male", defined in OWL as follows:

```
<owl:ObjectProperty rdf:ID="hasParent">
  <rdfs:domain rdf:resource="#Animal"/>
  <rdfs:range rdf:resource="#Animal"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hasFather">
   <rdfs:subPropertyOf rdf:resource="#hasParent"/>
  <rdfs:range rdf:resource="#Male"/>
</owl:ObjectProperty>
```

Then the above statements can be mapped into a subgraph in the translated BN as in Figure 3.2, there is an arc from "Class_hasParent" to "Class_hasFather" since the set of individuals who has father is a subset of the set of individuals who has parents, and "GR_hasParent" has states "Animal" and "Irrelevant", "GR_hasFather" has states "Male" and "Irrelevant":

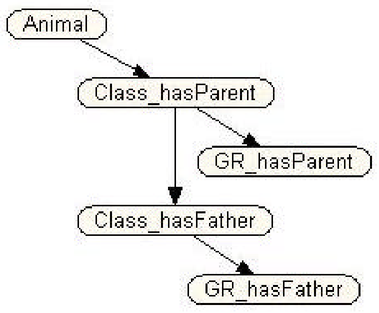


Figure 3.2 – Object Property

**Example 3:** An anonymous class "Res" defined by local value restriction as:

```
<owl:Restriction rdf:ID="Res">
    <owl:onProperty rdf:resource="#hasParent"/>
    <owl:allValuesFrom rdf:resource="#Human"/>
</owl:Restriction>
```

can be mapped into a subgraph in the translated BN as in Figure 3.3. "LR_hasParent_Res" has two states: "Human" and "Other" which is interpreted as the set difference of "Animal" and "Human". There is a directed arc from "Class_hasParent" to "Res" since the set of individuals "Res" whose parents are human is a subset of the set of individuals "Class_hasParent" who has parents. Statements using "owl:someValuesFrom" can be treated in the same way. In the current state, no translation for "owl:hasValue" is considered.

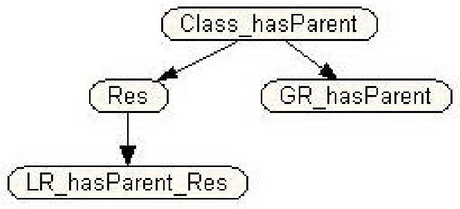


Figure 3.3 – Class by local value restriction

Similarly, an anonymous class "Res" defined by local cardinality restriction as:

```
<owl:Restriction rdf:ID="Res">
   <owl:onProperty rdf:resource="#hasParent"/>
   <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">2</owl:cardinality>
</owl:Restriction>
```

can be mapped into a subgraph in the translated BN as in Figure 3.4. "LC_hasParent_Res" has two states: "N2" (for its cardinality is 2) and "Nother" (for other cardinality). Statements using "owl:maxCardinality", "owl:minCardinality" could be easily translated by similar rules.

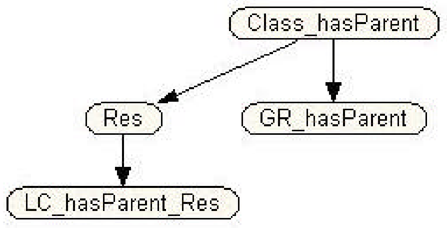


Figure 3.4 – Class by Local Cardinality Restriction

**Example 4:** A class "Man" defined by "owl:intersectionOf" as the logical conjunction of class "Human" and "Male" as follows:

```
<owl:Class rdf:ID="Man">
  <owl:intersectionOf rdf:parseType="Collection">
     <owl:Class rdf:about="#Human"/>
     <owl:Class rdf:about="#Male"/>
   </owl:intersectionOf>
</owl:Class>
```

can be mapped into a subgraph in the translated BN as in Figure 3.5. The CPT of "Man" is determined by the logical relation "AND". Similar translation can be done for "owl:unionOf" statements, with the CPT determined by the logical relation "OR"; and "owl:complementOf" statements, with the CPT determined by the logical relation "NOT".

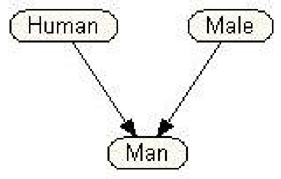


Figure 3.5 – Class by Logical Operator "AND"

**Example 5:** Constructor "rdfs:subClassOf" specifies necessary but not sufficient conditions for class membership. If "Human" is a subclass of "Animal" and "Biped", defined as:

```
<owl:Class rdf:ID="Human">
   <rdfs:subclassOf  rdf:resource="#Animal">
   <rdfs:subclassOf  rdf:resource="#Biped">
</owl:Class>
```

then we can map this statement into the following subgraph in the translated BN:

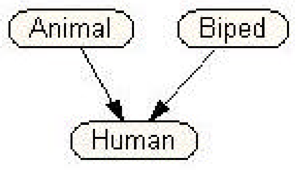


Figure 3.6 – Class by "rdfs:subClassOf" Axiom

**Example 6:** Constructor "owl:disjointWith" is used to assert that the classes involved have no individuals in common. Figure 3.7 below illustrates different situations the "disjointWith" axiom may involve.

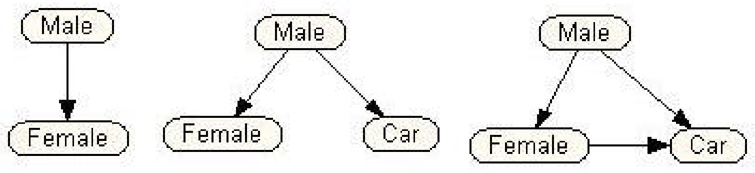
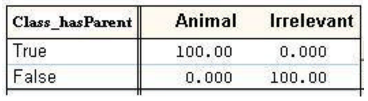


Figure 3.7 – Class by "owl:disjointWith" Axiom

In left figure, "Female" is disjoint with "Male"; in middle figure, "Male" is disjoint with both "Female" and "Car"; and in right figure, all three classes are mutual disjoint with each other.

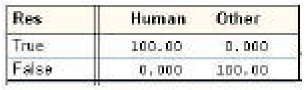
## 3.3 Constructing conditional probability tables

Once we had the network structure, the last step to finish the translation is to assign a conditional probability table (CPT) to each node in the structure. We require that prior probability "P(A)" be attached to a class "A" if it does not have any parent, conditional probability "P(A|B)" be attached to a class "A" if it is a subclass of class "B" (so, P(A| $\overline{B}$ ) = 0), and if necessary "P(A| $\overline{B}$ )" are attached to a class "A" if it is disjoint with class "B" (so, P(A|B) = 0). How to encode these probabilities in OWL file has been discussed earlier in subsection 3.1. In many cases, the CPT of a node can be determined completely from the semantics or logical relation between classes. For projection node such as "GR_P" and "LC_P_Res", its CPT is deterministic; for projection node such as "LR_P_Res", its CPT is deterministic if it is from an "owl:allValuesFrom" statement or specified by user if it is from an "owl:someValuesFrom" statement. Table 3.1 below shows the CPTs for the projection nodes from **Example 3** in last section:



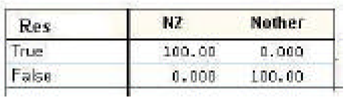
| Class_hasParent | Animal | Irrelevant |
|---|---|---|
| True | 100.00 | 0.000 |
| False | 0.000 | 100.00 |

**"GR_hasParent"**



| Res | Human | Other |
|---|---|---|
| True | 100.00 | 0.000 |
| False | 0.000 | 100.00 |

**"LR_hasParent_Res"**

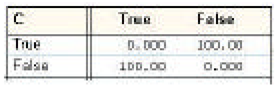


| Res | NZ | Nother |
|---|---|---|
| True | 100.00 | 0.000 |
| False | 0.000 | 100.00 |

**"LC_hasParent_Res"**

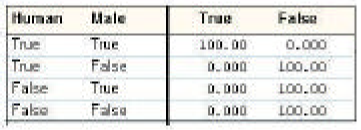
Table 3.1 CPT - Projection Nodes

For class defined by a logical operator (e.g., the class "Man" in **Example 4**), its CPT is determined by the logical relation.



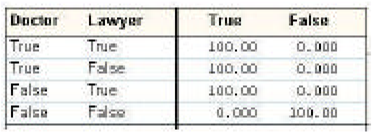
| C | True | False |
|---|---|---|
| True | 0.000 | 100.00 |
| False | 100.00 | 0.000 |

"$\overline{C}$"



| Human | Male | True | False |
|---|---|---|---|
| True | True | 100.00 | 0.000 |
| True | False | 0.000 | 100.00 |
| False | True | 0.000 | 100.00 |
| False | False | 0.000 | 100.00 |

**"Man"**



| Doctor | Lawyer | True | False |
|---|---|---|---|
| True | True | 100.00 | 0.000 |
| True | False | 100.00 | 0.000 |
| False | True | 100.00 | 0.000 |
| False | False | 0.000 | 100.00 |

**"Doctor $\cup$ Lawyer"**

Table 3.2 CPT – Logical NOT, AND, OR

Table 3.2 above shows sample CPT for class defined by "owl:complementOf", "owl:intersectionOf" or "owl:unionOf" statement in order. "Man" is an intersection of

"Human" and "Male", we have P(Man|Human,Male) = 1 and the probabilities of "Man" given all other assignments of "Human" and "Male" as conditions are false (0). "complementOf" and "unionOf" operators can be treated in a similar fashion.

In cases that a class node has several superconcept nodes as parents, computation is needed to get the whole CPT. For example, if a class "A" is a subclass of both "B" and "C", and we are given P(A|B) and P(A|C), then it can be easily shown that P(A|B,C) = P(A|B) / P(C|B), or symmetrically P(A|B,C) = P(A|C) / P(B|C), which can be computed if we also know P(C|B) or P(B|C). However, if a class "C" has $n$ direct superconcepts "S1, S2, … , Sn", it may be impossible to compute P(C|S1,S2,…,Sn) from given P(C|S1), P(C|S2), … , P(C|Sn) unless some strong independence assumption can be made. For such general cases like this, we may choose to model the network as a "noisy-or" BN in which influence from individual parents can be combined in a kind of simple additive way. We can also define some heuristic combination rule such as: P(C|S1,S2,…,Sn) = P(C) / P(S1,S2,…,Sn) = (P(C|S1)*P(S1)) / $\prod_{i=1}^{n} P(S_i \mid p(S_i))$. Another possible solution is to allow the user to specify conditional probability with arbitrary number of conditions, so there is no need to compute P(C|S1,S2,…,Sn) any more.

There are also cases that the arcs into a node in the BN come from different sources. For example, if both "Male" and "Female" are subclasses of "Animal" and they are disjoint, then for the "Female" node, there is an arc from "Animal" and an arc from "Male" into it; given "P(Animal)=0.5", "P(Male|Animal)=0.5" and "P(Female|Animal)=0.48", we can compute:

(1) P(Female) = P(Animal,Female) = P(Female|Animal) * P(Animal) = 0.48 * 0.5 = 0.24;

(2) P(Male) = P(Animal,Male) = P(Male|Animal) * P(Animal) = 0.5 * 0.5 = 0.25;

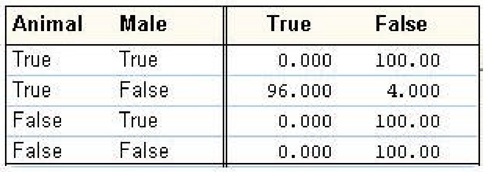
(3) P(Female| $\overline{Male}$) = P(Female, $\overline{Male}$) / P($\overline{Male}$) = P(Female) / (1- P(Male)) = 0.24 / (1- 0.25)

= 0.32;

(4) P(Female|Animal, $\overline{Male}$) = P(Animal, $\overline{Male}$,Female) / P (Animal, $\overline{Male}$)

= P(Female) / (P($\overline{Male}$|Animal) * P(Animal)) = 0.24 / (0.5 * 0.5) = 0.96.

And the CPT of node "Female" is given as below:



| Animal | Male | True | False |
|---|---|---|---|
| True | True | 0.000 | 100.00 |
| True | False | 96.000 | 4.000 |
| False | True | 0.000 | 100.00 |
| False | False | 0.000 | 100.00 |

Table 3.3 CPT – "Female $\subseteq$ Animal $\cap \neg$ Male"

The computation in these cases should be based on how the arcs come into the node are mixed. Note in above example the entry "P(Female| $\overline{Animal}$, Male)" in the CPT is irrelevant (this is a situation that never happens), so we simply assign it to false (0).

An example OWL ontology and its translated Bayesian network, including the CPT for all nodes, can be found at http://www.cs.umbc.edu/~zding1/HICSS37Supp/. Due to the page limit, we only show the DAG of the translated BN here in the figure below.

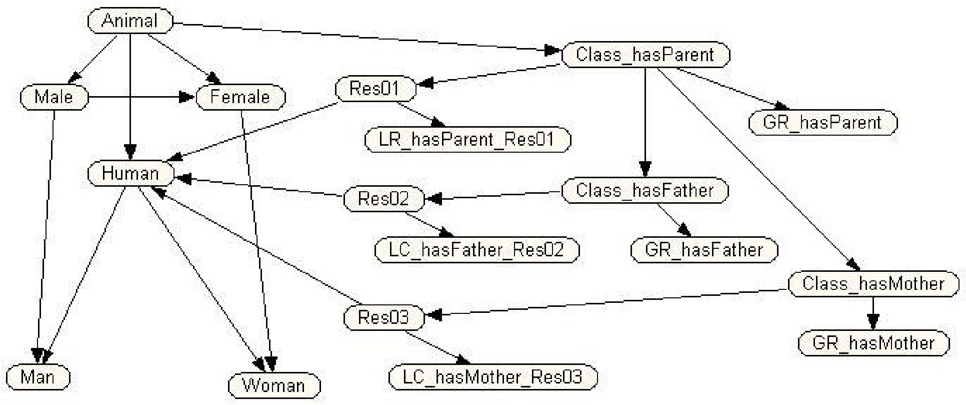


Figure 3.8 – Example Translated BN Ontology

## 4. Discussion on Semantics and Reasoning

In this section, we briefly discuss issues concerning the semantics of the Bayesian network generated by the converting rules and how reasoning can be performed over this network.

## 4.1 Semantics

The semant ics of the Bayesian network obtained can be outlined as follows. It will be associated with a joint probability distribution (as the product of all its CPT as with any BN) over the application domain, on top of the standard description logic semantics. A description logic interpretation $I = (\Delta^I, .^I)$ consists of a non-empty domain of objects $\Delta^I$ and an interpretation function $.^I$. This function maps every concept to a subset of $\Delta^I$, every role and attribute to a subset of $\Delta^I \times \Delta^I$, and every individual to an object of $\Delta^I$. An interpretation $I$ is a model for a concept C if $C^I$ is non-empty, and C is said "satisfiable". Besides this description logic interpretation $I = (\Delta^I, .^I)$, in our semantics, there is a function $P$ to map each object o$\in \Delta^I$ to a value between 0 and 1, $1 \geq P(\text{o}) \geq 0$, and $P$(o) = 1, for all o$\in \Delta^I$. This is the probability distribution over all the domain objects. For a class C: Pr(C) = $P$(o) for all o$\in$ C. If C and D are classes and C $\subseteq$ D, then Pr(C) $\leq$ Pr(D).

## 4.2 Reasoning

The probabilistic-extended ontology supports common ontology-related reasoning tasks as probabilistic inferences. Here we outline how three such tasks can be done in principle. Detailed algorithms are under development.

**Concept Satisfiability**: Given a concept represented by a description $e$, decide if P($e$) = 0 (false)? P($e$) can be computed by applying the chain rule of Bayesian networks.

**Concept Overlapping**: What is the degree of the overlap or inclusion P(C|$e$) between a concept C and a concept represented by a description $e$? P(C|$e$) can be computed by applying general Bayesian network belief update algorithms (e.g.,, belief propagation [Pearl 1986], junction tree [Lauritzen and Spiegelhalter 1988]).

**Concept subsumption**: How to find the most probable concept C that a given description $e$ belongs to? We define a similarity measure "MPC($e$,C)" between $e$ and C by combining *Jaccard Coefficient* [Rijsbergen 1979] and *MSP (Most-Specific-Parent)* [Doan et.al 2002] similarity measures as follows: (1) If P(C|$e$) = 1, let MPC($e$,C) = P($e$|C) = P($e$) / P(C). In this case, $e$ is a subclass of C, and larger P($e$|C) means C is more specific; (2) Otherwise, let MPC($e$,C) = P($e \cap$ C) / P($e \cup$ C) = P($e$,C) / (P($e$) + P(C) – P($e$,C)). It takes lowest value 0 if $e$ and C are disjoint and highest value 1 if $e$ and C are the same, and P($e$,C) = P(C|$e$) * P($e$). Note P($e$), P(C), P(C|$e$) can be computed using general Bayesian network techniques. According to the definition of this similarity measure, the concept node C with highest "MPC($e$,C)" in the BN is the most probable concept that $e$ belongs to.

## 5. Conclusions

In this paper we present our ongoing research on probabilistic extension to OWL. We have defined additional OWL classes (PriorProbObj, CondProbObjT, and CondProbObjF), which can be used to markup probabilities in OWL files. We have also defined a set of rules for translating most of the OWL constructors into Bayesian network subgraphs. We are actively working on resolving remaining issues, including:

- How to construct CPT for all nodes in a more systematic and disciplined way;
- Whether existing BN inference algorithm can be directly applied to our framework, and if not, what new algorithms need to be developed;
- In OWL ontologies, cycles can be formed by equivalent classes (defined by the "owl:equivalentClass" constructor), mutual disjoint among a set of classes, or other inferred dependencies among a set of classes. Since cycles are not allowed in Bayesian networks, how to detect and remove cycles is another issue to be addressed.

Based on successful resolution of these issues and other refinement of our framework, we plan to implement a prototype which can automatically translate a given OWL ontology into a BN and can also support common ontology-based reasoning tasks.

Future research can be conducted in several directions. One can extend this work to include individuals (i.e., Abox in description logics) and develop translation rules for individual related OWL constructors. Our current probabilistic markups can be used only to attach probabilities to otherwise *crisp* logical relations. For example, one can add conditional probability P(A|B) for a relation A $\subseteq$ B (which gives implicitly that P(B|A) = 1). Another direction of research is to extend this work to represent approximate relations and related probabilities (e.g., "A" is almost a subset of "B" with P(B|A) = 0.85). It is also very attractive to investigate how to apply this work to ontology mapping/translation since ontology mappings are often partial, approximate and uncertain. Finally it would be an interesting research topic to apply machine learning or information retrieval techniques to learn those attached probabilities before a domain expert come to verify it.

## Acknowledgement:

This work was supported in part by DARPA contract F30602-97-1-0215.

## References

[Bacchus 1990] Bacchus, F. *Representing and Reasoning with Probabilistic Knowledge*. MIT Press, Cambridge, MA. 1990

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

[Berners-Lee 1998] Berners-Lee, T. http://www.w3.org/DesignIssues/Semantic.html. Semantic Web Road Map, 1998

[Cooper 1990] Cooper, G. 1990. The Computational Complexity of Probabilistic Inference Using Bayesian Belief Network. *Artificial Intelligence*, 42, 393-347.

[Doan et.al 2002] Doan, A. H.; Madhavan, J.; Domingos, P.; Halevy, A. 2002. Learning to Map between Ontologies on the Semantic Web. In *WWW 2002*.

[Giugno and Lukasiewicz 2002] Giugno, R.; Lukasiewicz, T. April 2002. P-*SHOQ*(D): A Probabilistic Extension of *SHOQ*(D) for Probabilistic Ontologies in the Semantic Web. INFSYS Research Report 1843-02-06, Wien, Austria.

[Gruber 1993] Gruber, T. R. 1993. A Translation Approach to Portable Ontology Specifications. In *Knowledge Acquisition*, 5(2):199-220.

[Halpern 1990] Halpern, J.Y. 1990. An Analysis of First-Order Logics of Probability. In *Artificial Intelligence*, 46: 311-350.

[Heinsohn 1994] Heinsohn, J. 1994. Probabilistic Description Logics. In *Proceedings of UAI-94*: 311-318.

[Horrocks 2001] Horrocks, I.; Sattler, U. 2001. Ontology Reasoning in the *SHOQ*(D) Description Logic. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*.

[Horrocks 2002] Horrocks, I. 2002. DAML+OIL: A Description Logic for the Semantic Web.

[Jaeger 1994] Jaeger, M. 1994. Probabilistic Reasoning in Terminological Logics. In *Proceedings of KR-94*: 305-316.

[Koller et.al 1997] Koller, D.; Levy, A.; Pfeffer, A. 1997. P-CLASSIC: A Tractable Probabilistic Description Logic. In *Proceedings of AAAI-97*: 390-397.

[Lauritzen and Spiegelhalter 1988] Lauritzen, S.L.; Spiegelhalter, D.J. 1988. Local Computation with Probabilities in Graphic Structures and Their Applications in Expert Systems. In *J. Royal Statistical Soc. Series B*, 50(2): 157-224.

[Pawlak 1982] Pawlak, J. 1982. Rough Sets. *International Journal of Information and Computers*, 11, 341-356.

[Pearl 1986] Pearl, J. 1986. Fusion, Propagation, and Structuring in Belief Networks. In *Artificial Intelligence*, 29: 241-248.

[Pearl 1988] Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufman, San Mateo, CA.

[Poole 1993] Poole, D. November 1993. Probabilistic Horn Abduction and Bayesian Networks. In *Artificial Intelligence*, 64(1): 81-129.

[Rijsbergen 1979] van Rijsbergen 1979. *Information Retrieval*. Lodon:Butterworths. Second Edition.

[Stuckenschmidt 2000] Stuckenschmidt, H.; Visser. U. 2000. Semantic Translation based on Approximate Re-classification. *Proceedings of the Workshop "Semantic Approximation, Granularity and Vagueness", KR'00.*

[Zadeh, 1965] Zadeh, L. 1965. Fuzzy Sets. *Information and Control*, 8, 338-353.

[DAML] http://www.daml.org/. DAML Homepage.

[DAML+OIL] http://www.daml.org/2001/03/daml+oil-index. DAML+OIL Homepage.

[NORSYS] http://www.norsys.com/. Norsys System Corp.

[OIL] http://www.ontoknowledge.org/oil/. OIL Homepage.

[OWL] http://www.w3.org/2001/sw/WebOnt/. W3C WebOnt Working Group.

[OWL GUIDE] http://www.w3.org/TR/owl-guide/. OWL Web Ontology Language Guide.

[OWL REF ] http://www.w3.org/TR/owl-ref/. OWL Web Ontology Language Reference.

[OWL SEMANTICS] http://www.w3.org/TR/owl-semantics/. OWL Web Ontology Language Semantics and Abstract Syntax.

[RDF] http://www.w3.org/RDF/. W3C RDF Homepage.

[RDFS] http://www.w3.org/TR/rdf-schema/. W3C RDF Schema Specification Homepage.

[SHOE] http://www.cs.umd.edu/projects/plus/SHOE/. SHOE Homepage.