

OntoBayes: An Ontology-Driven Uncertainty Model

Yi Yang and Jacques Calmet

Institute for Algorithms and Cognitive Systems (IAKS)

University of Karlsruhe (TH)

76131 Karlsruhe, Germany

{yiyang,calmet}@ira.uka.de

Abstract—This paper describes an ontology-driven model, which integrates Bayesian Networks (BN) into the Ontology Web Language (OWL) to preserve the advantages of both. This model makes use of probability and dependency-annotated OWL to represent uncertain information in BN structures. These extensions enhance knowledge representation in OWL and enable agents to act under uncertainty and complex structured open environments at the same time. This paper presents the underlying principles and scratches the surface of the decision theoretic agent system design based on “OntoBayes”.

I. INTRODUCTION

A key reason for using ontologies in intelligent systems is that they enable the representation of background knowledge about a domain in a machine understandable form. An intelligent agent is capable of flexible autonomous action in order to meet its design objectives [1]. Agents need to develop an environmental model which permits them to reason on how the actions they perform impact their environment and how their effects lead them to reach their goals. To construct these models lightweight ontologies provide the conceptual framework which can describe not only the type and properties of entities but also the relations between them [2]. By comparison, heavyweight ontologies possess additional axioms and constraints [3].

Ontologies can formally and explicitly specify a shared conceptualization [4]. They can excellently represent the organizational structure of large complex domains, but their application is bounded because of their inability to deal with uncertainty [5]. Uncertainty is an inevitable feature in most environments. Agents do not act within a static and close environment but within a dynamic and open one. The available information is mostly incomplete and often imprecise, because agents almost never have access to the whole truth implied by their environment. Agents must, therefore act under uncertainty [6].

In order to allow agents to work under uncertainty, an extension of ontologies, which has the capability of capturing uncertainty knowledge about concepts, properties and relations in domains and of supporting reasoning with inaccurate information, is mandatory. Along this direction, researchers have attempted in the past to use different non-probabilistic and probabilistic methodologies for ontology definition and reasoning. In this paper we propose an ontology-driven uncertainty model, which makes use of Web Ontology Language (OWL) [7] as underlying ontology modeling language and annotates it with Bayesian probability as well as with dependency relationships.

OWL is one of various ontology modeling languages (e.g. [8], [9], [10], [11]) and it is supported by W3C for defining ontologies in semantic web. It is based on the DARPA Agent Markup Language (DAML) [10] and the Ontology Inference Layer (OIL) [9] but with simpler primitives.

Probability is interpreted as a numerical degree of belief (between 0 and 1) in Bayesian probability [12]. Whereas frequency probability assigns probabilities to random events only according to their relative frequencies of occurrence, Bayesian probability allows the assignment of probabilities to any other kind of statement. For example, the latter might assign a 1/10,000 probability to a personal belief in the proposition that there is a tenth planet in the solar system. This assertion is made without intending to assert anything about relative frequencies.

The annotation of dependency relations in OWL facilitates an explicit specification of Bayesian random variables in ontologies and structure them into Bayesian Networks (BNs) [13]. BN is a widely used graphic model for probabilistic knowledge representation under uncertainty. It is very limited because of its inability to represent complex structured domains. This is one of the key reasons for proposing OntoBayes, an ontology-driven Bayesian model for uncertain knowledge representation.

The paper is structured as follows. In section II we remind the relevant concepts of Bayesian probability and we summarize the basic features and notations of OWL. This section also introduces a working example that is used to illustrate our model. Section III is devoted to an overview of OntoBayes. In Section IV we survey existing approaches. Section V deals with concluding remarks and future works.

II. BAYESIAN PROBABILITY AND OWL

In order to set up a general framework for agent systems, a formal language for representing uncertain but structured knowledge is required. We remind first the basics of Bayesian probability.

A. Basic Bayesian probability notation

The basic element of Bayesian probability is the *random variable* which is considered as referring to a state (or an event) of the initially unknown environment of agents. Depending on the type of domain values, random variables are typically divided into three kinds: boolean, discrete and continuous random variables [6]. In this paper we only use

discrete random variables, which include boolean random variables as a special case, with values from an enumerable domain. We use an uppercase letter as the first letter of random variables and lowercase ones for their values. For instance, the domain of *Price* and *Premium* might be {low, middle, high}.

In order to describe what is known about a variable A in the absence of any other evidence Bayesian probability uses the *prior* (or unconditional) probability. It is often called simply the prior and written as $P(A)$. A prior is normally the purely subjective assessment of an experienced expert. To express the probabilities of all combinations of the values on a set of random variables we need the *joint probability distribution* of these variables. The prior probability of two random variable A and B in conjunction can be written $P(A \wedge B)$ or $P(A, B)$. In the case of $P(\textit{Price}, \textit{Premium})$ it can be represented by a 3×3 table of probabilities. This is called the joint probability distribution of *price* and *Premium*.

Once agents have observed some evidence which has influence over the previously random variables, prior probabilities are no longer appropriate. Instead, we use *conditional* (or posterior) probabilities. The notation used is $P(A|B)$, where A and B are random variables, and is read as "the probability of A given B ". Conditional probabilities can be defined in terms of unconditional probabilities:

$$P(A|B) = \frac{P(A \wedge B)}{P(B)} \quad (1)$$

where $P(B) > 0$. For example, $P(\textit{Price} = \textit{low} | \textit{Premium} = \textit{high}) = 0.1$ indicates that if the Premium of an insurance product is observed to be high and no other information is yet available, then the probability that the product has a low price will be 0.1. Equation (1) can also be written as

$$P(A \wedge B) = P(A|B)P(B) \quad (2)$$

which is called the *product rule* [6]. Starting from (1) and (2) we can derive the Bayes' theorem:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \quad (3)$$

The conditional probability $P(A|B)$ indicates the dependency or causal relation between variables A and B : A depends on B (or B causes A). For instance $P(\textit{price} | \textit{Premium})$ expresses that the price of an insurance product depends on its Premium.

B. Basic features and notations of OWL

OWL is a semantic markup language for publishing and sharing ontologies on the World Wide Web. It is intended to provide a language that can be used to

- conceptualize domains by defining classes and properties of those classes,
- construct individuals and assert properties about them, and
- reason about these classes and individuals [7].

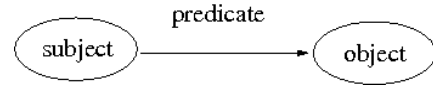


Fig. 1. The graphical representation for a generic RDF triple

OWL is developed as a vocabulary extension of the Resource Description Framework (RDF) [14]. The underlying structure of any expression in RDF is a collection of triples, each consisting of a subject, a predicate (also called a property) and an object. Such triples can also be represented as the "RDF graph" in which nodes corresponds to the "subject" and "object" and the directed arc corresponds to the "predicate" as shown in Figure 1. OWL and RDF have much in common, but OWL is a language with better descriptive power than RDF.

OWL has three increasingly expressive sublanguages: OWL Lite, OWL DL and OWL Full [15]. We introduce some basic primitives of them as follows:

- `<owl:Class>`: This primitive is used to create ontology concepts as classes. Each class will be named with a class identifier. For example, "`<owl:Class rdf:ID='Location'>`" defines a class instance "Location".
- `<owl:DatatypeProperty>`: This primitive is used to express relations between instances of classes and RDF literals [16] and XML Schema datatypes [17].
- `<owl:ObjectProperty>`: This primitive is used to describe relations between instances of two classes.
- `<owl:Restriction>`: This primitive is used to specify the constraints on ontology concepts.
- `<owl:onProperty>`: This primitive indicates which property is restricted.

A detailed information about features and primitives of the language is to be founded in [7], [15], [18]. In the next subsection we show how to use this language to model an ontology for applications in the insurance industry.

C. An application example

Market-based means of managing natural disaster risks are emerging according to a recent article in Food Policy [19]. With the help of capital market (e.g. insurance industry) the catastrophic consequence could be significantly reduced. From the perspective of insurance a decision support system is required to help making decisions when a client invokes an insurance against natural disasters. From the perspective of users, it is meaningful to have a system to help them assess insurance purchase decisions and to manage catastrophic risks from natural disasters. One of the important components of such a decision support system is the knowledge base, particularly the knowledge representation. We have developed an ontology for modeling the knowledge base. We extract some encoding from it, to show how OWL works practically.

The most basic concepts in the insurance and natural disaster domain should correspond to classes that are the roots of various taxonomic trees. In our ontology we define a class "Insurance" and "Natural_Disaster" as roots in

```

<owl:Class rdf:ID="#Customer">
  <rdfs:subClassOf>
    <owl:Class rdf:resource="#Marketing"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="#Product">
  <rdfs:subClassOf>
    <owl:Class rdf:resource="#Marketing"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="Marketing">
  <rdfs:subClassOf rdf:resource="#Insurance"/>
</owl:Class>

```

Fig. 2. Encoding for class *Customer*, *Marketing* and *Insurance*

```

<owl:DatatypeProperty rdf:ID="Price">
  <rdfs:range>
    <owl:DataRange>
      <owl:oneOf rdf:parseType="Resource">
        <rdf:first rdf:datatype="xsd:string">
          low</rdf:first>
        <rdf:rest rdf:parseType="Resource">
          <rdf:rest rdf:parseType="Resource">
            <rdf:first rdf:datatype="xsd:string">
              middle</rdf:first>
          </rdf:rest>
          <rdf:first rdf:datatype="xsd:string">
            high</rdf:first>
          </rdf:rest>
        </owl:oneOf>
      </owl:DataRange>
    </rdfs:range>
    <rdfs:domain rdf:resource="#Product"/>
  </owl:DatatypeProperty>

```

Fig. 3. Encoding for a datatype property *Price*

domain insurance and natural disaster respectively. Figure 2 presents three hierarchical classes in insurance domain: “Customer” and “Product” are two subclasses of “Marketing” and “Marketing” is a subclass of “Insurance” in turn. We also define properties for classes, for example, Figure 3 defines a datatype property “Price” for class “Product”. By reason of only using discrete random variables we require price’s value to be one the of domain values {low, middle, high}. Figure 4 presents an object property “Buy” for class “Customer”, which means “Customer” buys “Product”. We can also build some constraints on concepts as illustrated in Figure 5, which specify the cardinality of class “Product” on property “Price”: “Product” has only one “Price”.

III. THE ONTOBAYES MODEL

In the last section we have described an application example for insurance and natural disaster management. The typical characteristic of these domains is risk management. The normal ontological model is not able to express the domain specific uncertainty or risks. To overcome this shortcoming we propose an ontology-driven uncertainty model: OntoBayes Model. This model uses extended OWL to integrate the Bayesian approach into ontologies, and preserves the advantages of both.

```

<owl:ObjectProperty rdf:ID="Buy">
  <rdfs:range rdf:resource="#Product"/>
  <rdfs:domain rdf:resource="#Customer"/>
</owl:ObjectProperty>

```

Fig. 4. Encoding for an object property *Buy*

```

<owl:Class rdf:about="#Product">
  <owl:Restriction>
    <owl:cardinality rdf:datatype="xsd:int">1
  </owl:cardinality>
  <owl:onProperty>
    <owl:DatatypeProperty rdf:ID="Price"/>
  </owl:onProperty>
</owl:Restriction>
</owl:Class>

```

Fig. 5. Encoding for a cardinality restriction of class *Product* on property *Price*

A. Probability-annotated OWL

The first integration step is probabilistic extension. To express uncertain information or the risk of actions we need an extension of OWL which enables to specify probability-annotated classes or properties. For this purposes we define three OWL classes: “PriorProb”, “CondProb” and “FullProbDist”. The first two classes are defined to identify the prior probability and conditional probability respectively. They have a same datatype property “ProbValue”, which can express the probabilistic value between 0 and 1. The last one is used to specify the full disjoint probability distribution. It has two disjoint object properties: “hasPrior” or “hasCond”. The property “hasPrior” specifies the relation between classes “FullProbDist” and “PriorProb”. It indicates that the instances of “PriorPro” are elements of one instance of “FullProbDist”. The property “hasCond” describes the relation between classes “FullProbDist” and “CondProb” and it has a similar semantic meaning as “hasPrior”. These two properties are disjoint, because any instance of “FullProbDist” can only have one probability type: either prior or conditional.

Table I states a full prior probability distribution for $P(Price)$. With the help of probability-annotated OWL we can encode it in Figure 6.

Table II shows an example of a full conditional probability distribution for $P(Price|Premium)$. The relevant encoding will be presented in Figure 7. But this encoding is not complete because it contains only one of the nine conditional probabilities: $P(Price = low|Premium = low) = 0.25$. The other eight can be similarly encoded.

The proposed encoding scheme is influenced by the approaches in [20], [21], but their scope is limited because of

TABLE I
EXAMPLE OF $P(Price)$

Price=low	0.2
Price=middle	0.5
Price=high	0.3

```

<FullProbDist rdf:ID="P(Price)">
  <hasPriorProb rdf:resource=
    "#P(Price=low)" />
  <hasPriorProb rdf:resource=
    "#P(Price=middle)" />
  <hasPriorProb rdf:resource=
    "#P(Price=high)" />
</FullProbDist>

<PriorProb rdf:ID="P(Price=low)">
  <ProbValue>0.2</ProbValue>
</PriorProb>
<PriorProb rdf:ID="P(Price=middle)">
  <ProbValue>0.5</ProbValue>
</PriorProb>
<PriorProb rdf:ID="P(Price=high)">
  <ProbValue>0.3</ProbValue>
</PriorProb>

```

Fig. 6. The probability-annotated encoding for Table I

TABLE II
EXAMPLE OF $P(\text{Price}|\text{Premium})$

	Premium=low	Premium=middle	Premium=high
Price=low	0.25	0.2	0.5
Price=middle	0.5	0.2	0.5
Price=high	0.3	0.2	0.5

their inability to represent multivalued random variables and to specify the full disjoint probability distributions over a multivalued variable. Our scheme overcomes these limitations by incorporating probabilistic representations in OWL.

B. Dependency-annotated OWL

The probabilistic extension of OWL alone is not enough for modeling our ontology-driven BN. It is required to specify the dependency relations between the random variables explicitly, because of the following reasons.

- 1) The dependency relations are not automatically specified by modeling ontologies.
- 2) The dependency modeling can indicate which random variables are dependent. It means that we can construct BNs based on this specification.
- 3) A common dependency modeling method is more applicable than a domain specific method as proposed in Ding’s work [20] that uses BNs to overlap different ontologies over a single domain. In order to reach this goal Ding and Peng design some set-theoretic approach based rules for dependency modeling within the original language. However these rules can not satisfy the requirements of our model, which aims at facilitating probabilistic reasoning to support decision making. We need a more generic dependency modeling than this set-theoretic approach.

In order to solve the problems just mentioned we introduce an additional property element `<rdfs:dependsOn>` to markup dependency information in an OWL ontology. Before we give a formal definition of dependency, we introduce some notations which are influenced by the Object Oriented Programming (OOP) approach. We denote an object property s between

```

<FullProbDist rdf:ID="P(Price|Premium)">
  <hasCondProb rdf:resource=
    "#P(Price=low|Premium=low)" />
  <hasCondProb rdf:resource=
    "#P(Price=middle|Premium=low)" />
  <hasCondProb rdf:resource=
    "#P(Price=high|Premium=low)" />
  <hasCondProb rdf:resource=
    "#P(Price=low|Premium=middle)" />
  <hasCondProb rdf:resource=
    "#P(Price=middle|Premium=middle)" />
  <hasCondProb rdf:resource=
    "#P(Price=high|Premium=middle)" />
  <hasCondProb rdf:resource=
    "#P(Price=low|Premium=high)" />
  <hasCondProb rdf:resource=
    "#P(Price=middle|Premium=high)" />
  <hasCondProb rdf:resource=
    "#P(Price=high|Premium=high)" />
</FullProbDist>

<CondProb rdf:ID="P(Price=low|Premium=low)">
  <ProbValue>0.25</ProbValue>
</CondProb>
.....

```

Fig. 7. The partial encoding for Table II

domain class X and range class Y as $s(X, Y)$. It is considered as an available operation of the subject class X to object class Y . A datatype property d of class X will be denoted as $X.d$, where d is considered as an attribute of X . Now we can define the dependency between the properties as follow.

Definition 3.1: A dependency is a pair $X \rightarrow Y$, where each of X and Y is either a datatype property $X.d$ or an object property $s(X, Y)$. It is read as “ X depends on Y ”.

This definition clearly points out that each random variable in OntoBayes modeled BN is either an object property or a datatype property. We do not specify the dependency relation between classes because this avoids possible errors when extracting a Bayesian structure from ontologies. For instance, in our application example for insurance we model two object properties “LiveIn” and “LocatedAt” for class “Person”. These properties have a same range class “Location”. The property “LiveIn” indicates the permanent official address while “LocatedAt” stores the present address. The other class “Natural_Disaster” has a datatype property “Occurrence_probability”. If we define that the property “Occurrence_probability” depends on “Location”, then insurance companies can not distinguish the occurrence probability of natural disaster between customer’s resident location and actual location. In fact they only care about the occurrence probability at resident location of customer. In order to avoid such confusion by modeling we decide to specify dependency between properties, not between classes. Therefore the correct modeling for the property “Occurrence_probability” should be that “Occurrence_probability” depends on “LiveIn”. Figure 8 shows the encoding for the dependency denotation $Buy(\text{Product}, \text{Customer}) \rightarrow \text{Product.Price}$ in OWL.

To adapt the probability annotation to our dependency definition, we simply need to change accordingly the value of

```

<owl:ObjectProperty rdf:ID="Buy">
  <rdfs:range rdf:resource="#Product"/>
  <rdfs:domain rdf:resource="#Customer"/>
  <rdfs:dependsOn rdf:resource="#Price"/>
</owl:ObjectProperty>

```

Fig. 8. Encoding for $Buy(Product, Customer) \rightarrow Product.Price$

the property element “rdf:ID”. For instance, we should change $\langle FullProbDist \text{ rdf:ID}="P(Price|Premium)" \rangle$ in Figure 7 to $\langle FullProbDist \text{ rdf:ID}="P(Person.Price|Person.Premium)" \rangle$.

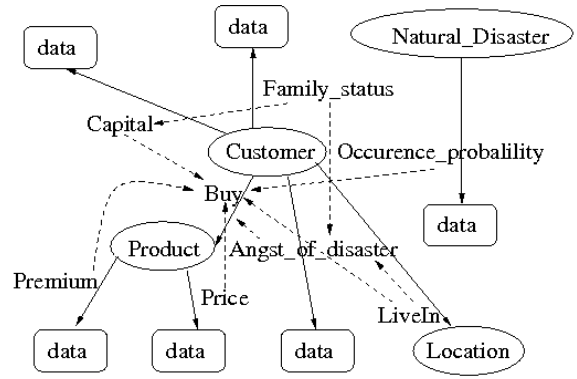
C. The graphical representation of OntoBayes

The presentation of OntoBayes was so far done through some sort of markup language. There is, in addition, a graphical representation. There are in fact two graphical models in OntoBayes: the OWL and the Bayesian graph (or network) models. The former is a directed graph which is built on the graph data model of RDF, as illustrated in Figure 1, and additionally has a markup of the dependency relation between properties. This graph model can visualize all possible information of a specified ontology such as the class hierarchy or the dependency relation for instance. But it exhibits so many different relations that it is challenging to visualize any significant overview of such graphs in realistic cases. Therefore we extract the Bayesian graph model from this model, in order to clearly show the dependency relations. These two models can be also distinguished through their nodes. Indeed, while the nodes in the OWL graph model consist of classes and datatypes, the nodes in the Bayesian graph model are properties.

The underlying structure of any expression for BNs in OntoBayes is a collection of triples, each consisting of a subject, a predicate and an object, where the predicate is constantly the primitive $\langle rdfs:dependsOn \rangle$ and the subject and object are properties. Such a triple is called as *dependency triple*. There are two differences from an RDF triple in graphical representation. The first one is that we omit the predicate as a label of an arc because the predicate here is unique. The second one is that the arrow of arcs is not from subject to object, but from object to subject, because the intuitive meaning of an arrow in a properly constructed network is usually that X has a direct influence on Y [6]. We illustrate it in Figure 9 as opposed to a generic RDF triple in Figure 1. We can simply construct the OWL graph model with the help of RDF and dependency triples. Figure 10 shows some concepts and their properties as well as the relations in the domain of insurance and natural disaster (as mentioned in Section II-C). The ellipse is a graphical notation for the OWL classes and the rectangle is used for any data type. A label on an

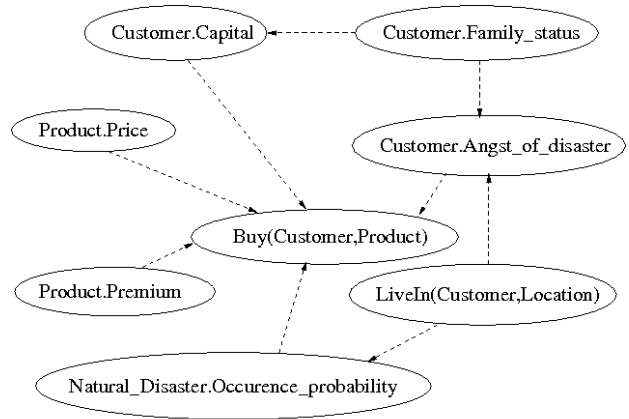


Fig. 9. The graphical representation for a generic dependency triple



legend:
 -----> influence relation
 -----> owl property
 [data] set of domain value
 () owl class

Fig. 10. The OWL graph model for the insurance ontology



legend: () random variable -----> influence relation

Fig. 11. The Bayesian graph model for an insurance ontology

arrow line refers to a property. The dashed line models the influence relation. For example $Buy(Customer, Product) \rightarrow Product.Price$.

Using the dependency triples we can build a Bayesian graph model by the following simple rules.

- First we extract all dependency triples from an OntoBayes ontology and represent them separately according to Figure 9.
- Next, all triples will be merged: all nodes with a same identifier are composed into one single node. For example, if there are two triples $A \rightarrow B$ and $B \rightarrow C$, they can be merge into a Bayesian Network with only one node B such as $A \rightarrow B \rightarrow C$.

In Figure 11 we illustrate the same encoding shown in Figure 10 but with a Bayesian Network. The ellipses are nodes and the dashed lines specify the influence relations.

IV. RELATED WORKS

There has been a number of attempts to define non-probabilistic and probabilistic methodologies to represent uncertainty inclusion ontologies, such as possibility theory [22], fuzzy logic [23], [24], rough sets [25], [26] and probability [5], [20], [21], [27], [28] approaches.

Ding [20] presents a probabilistic extension to OWL and the translated BN is associated with a joint probability distribution over the application domain. But the proposed method is not suitable to deal with multivalued random variable. Therefore, the main purpose of the work is ontology overlapping with the help of BN. It is very different from our approach. Based on their probabilistic extension Gu [21] lightly modified the extended annotations, to markup arbitrary conditional probability. But they didn't overcome the limitation of two-valued random variable. Although the translation rules from ontologies to BN looks similar to ours, the actual transformation of properties is done quite differently in Gu's work.

The approach that seems closest to ours is the old work of Koller and Pfeffer [5]. They integrate BN into a frame-based system to preserve the advantages of both. We have a similar approach but use totally different languages and methods. Besides, the translation mechanism is also very different. Furthermore, they didn't propose a formal notation to show how the integration syntactically and semantically works.

V. CONCLUSION AND FUTURE WORK

In this paper, we have proposed an approach to introduce uncertain knowledge in decision making systems. Our goal is to design and implement a working system for the management of risks linked to natural disasters in the insurance industry. The system extends Bayesian Networks and OWL. We have defined additional OWL classes which can be used to markup probabilities and dependencies in OWL files. Our model allows multivalued random variable representation in OWL and full joint probability distribution over them.

From the perspective of ontologies, the OntoBayes model preserves the ability to express meaningful knowledge in very large complex domains and extent ontologies to probability-annotated OWL to facilitate meaningful knowledge representation in uncertain systems. From the perspective of probabilistic modeling, our model take the advantage of powerful knowledge representation ability from ontologies, to scale up our ability to do uncertain reasoning.

Our model is also meaningful for the design of decision theoretic agents [6]. A feature of OntoBayes is that agents can reason about the effects of their actions enabling them to interact with an incompletely known world. The next task is thus to design the decision making mechanism of OntoBayes. Then, the problem of the syntactical and semantical validation of these decisions will be investigated.

ACKNOWLEDGEMENT

One author (Yi Yang) thanks the graduate college "Natural Disasters" at the university of Karlsruhe (TH) for their financial support.

REFERENCES

- [1] M. Wooldridge, "Intelligent agents," in *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, W. Gerhard, Ed. The MIT Press, 1999, ch. 1, pp. 27–78.
- [2] K. Sycara and M. Paolucci, "Ontologies in agent architectures," in *Handbook on Ontologies*, S. Staab and R. Studer, Eds. Springer, 2004, ch. 17, pp. 343–365.
- [3] A. Gómez-Pérez, M. Fernández-López, and O. Corcho, *Ontological engineering*. Springer, 2004.
- [4] T. Gruber, "A translation approach to protable ontology specifications," *Knowledge Acquisition*, vol. 5, no. 2, pp. 199–220, 1993.
- [5] D. Koller and A. Pfeffer, "Probabilistic frame-based systems," in *Proc. AAAI National Conference on Artificial Intelligence (AAAI'98)*, Madison, Wisconsin, USA, July, 1998, pp. 580–587.
- [6] S. J. Russell and P. Norvig, *Artificial intelligence*, 2nd ed. Prentice Hall, 2003.
- [7] Owl web ontology language guide. W3C. [Online]. Available: <http://www.w3.org/TR/owl-guide/>
- [8] M. Uschold and M. Grüninger, "Ontologies: Principles, methods and applications," *Knowledge Engineering Review*, vol. 11, no. 2, pp. 93–155, 1996.
- [9] Ontology inference layer. [Online]. Available: <http://www.ontoknowledge.org/oil/>
- [10] Darpa agent markup language. [Online]. Available: <http://www.daml.org/>
- [11] Simple html ontology extensions (shoe). [Online]. Available: <http://www.cs.umd.edu/projects/plus/SHOE/>
- [12] J. M. Bernardo and A. F. M. Smith, *Bayesian theory*, paperback ed. Wiley, 2000.
- [13] F. V. Jensen, *An introduction to Bayesian networks*. UCL Press, 1996.
- [14] Resource description framework (rdf): Concepts and abstract syntax. W3C. [Online]. Available: <http://www.w3.org/TR/rdf-concepts/>
- [15] Owl web ontology language semantics and abstract syntax. W3C. [Online]. Available: <http://www.w3.org/TR/owl-semantics/>
- [16] Rdf/xml syntax specification (revised). W3C. [Online]. Available: <http://www.w3.org/TR/rdf-syntax-grammar/>
- [17] Xml schema part 2: Datatypes. W3C. [Online]. Available: <http://www.w3.org/TR/xmlschema-2/>
- [18] Owl web ontology language reference. W3C. [Online]. Available: <http://www.w3.org/TR/owl-ref/>
- [19] J. Skees, "A role for capital markets in natural disasters: a piece of the food security puzzle," *Food Policy*, vol. 25, pp. 365–378, 2000.
- [20] Z. Ding and Y. Peng, "A probabilistic extension to ontology language owl," in *Proc. International Conference on System Sciences (HICSS'04)*, vol. 4. Washington, DC, USA: IEEE Computer Society, Jan.5–8, 2004, p. 10.
- [21] H. K. P. Tao Gu and D. Q. Zhang, "A bayesian approach for dealing with uncertain contexts," in *Proc. International Conference on Pervasive Computing (Pervasive'04)*, Austria, Apr., 2004.
- [22] D. Dubois and H. M. Prade, *Possibility theory*. Plenum Pr., 1988.
- [23] R. A. Angryk and F. E. Perty, "Consistent fuzzy concept hierarchies for attribute generalization," in *Proc. International Conference on Information and Knowledge Sharing (IKS'03)*, Scottsdale, AZ, USA, Nov.17–19, 2003.
- [24] G. J. Klir and T. A. Folger, *Fuzzy sets, uncertainty, and information*. Prentice Hall, 1988.
- [25] H. Stuckenschmidt and U. Visser, "Semantic translation based on approximate reclassification," in *Proc. Workshop on Semantic Approximation, Granularity and Vagueness*, Breckenridge, Colorado, USA, 2000.
- [26] Z. Pawlak, *Rough sets*. Kluwer, 1991.
- [27] J. Pearl, *Probabilistic reasoning in intelligent systems*, 2nd ed. Morgan Kaufmann, 1997.
- [28] M. Holli and E. Hyvnen, "Probabilistic information retrieval based on conceptual overlap in semantic web ontologies," in *Proc. Finnish Artificial Intelligence Conference (FAIS'04)*, vol. 2, Finland, 2004.