A First-Order Bayesian Tool for Probabilistic Ontologies

Paulo C. G. Costa[1], **Marcelo Ladeira**[2], **Rommel N. Carvalho**[2],
Kathryn B. Laskey[1], **Laécio L. Santos**[2], **Shou Matsumoto**[2]

[1]George Mason University
4400 University Drive
Fairfax, VA 22030-4400 USA
{pcosta, klaskey}@gmu.edu

[2]University of Brasilia
Campus Universitário Darcy Ribeiro
Brasilia – DF 70910-900 Brazil
mladeira@unb.br, {rommel.carvalho, laecio, cardialfly}@gmail.com

Abstract

One of the major weaknesses of current research on the Semantic Web (SW) is the lack of proper means to represent and reason with uncertainty. A number of recent efforts from the SW community, the W3C, and others have recently emerged to address this gap. Such efforts have the positive side effect of bringing together two fields of research that have been apart for historical reasons, the artificial intelligence and the SW communities. One example of the potential research gains of this convergence is the current development of Probabilistic OWL (PR-OWL), an extension of the OWL Web Ontology Language that provides a framework to build probabilistic ontologies, thus enabling proper representation and reasoning with uncertainty within the SW context. PR-OWL is based on Multi-Entity Bayesian Networks (MEBN), a first-order probabilistic logic that combines the representational power of first-order logic (FOL) and Bayesian Networks (BN). However, PR-OWL and MEBN are still in development, lacking a software tool that implements their underlying concepts. The development of UnBBayes-MEBN, an open source, Java-based application that is currently in alpha phase (public release March 08), addresses this gap by providing both a GUI for building probabilistic ontologies and a reasoner based on the PR-OWL/MEBN framework. This work focuses on the major challenges of UnBBayes-MEBN implementation, describes the features already implemented, and provides an overview of the major algorithms, mainly the one used for building a Situation Specific Bayesian Network (SSBN) from a MEBN Theory.

Introduction

Uncertainty is an important element of many actions or processes in our world. Therefore, any representational scheme designed to model such actions and processes should be able to cope with the effects of non-deterministic output and its consequences. This is particularly true in an environment such as the Semantic Web. However, until recently, too little attention has been devoted to probabilistic representation and reasoning for the SW. This paper begins with a brief introduction of the factors that led to this current state of affairs. Then, background information on the technologies being implemented by UnBBayes-MEBN is provided. Finally, the main section of this paper provides an overview of the major algorithms implemented in UnBBayes-MEBN.

A Deterministic Ontology-based Web

One of the main technical differences between the current World Wide Web and the Semantic Web is that while the first relies on syntactic-only protocols such as HTTP and HTML, the latter adds meta-data annotations as a means to convey shared, precisely defined terms. That is, semantic awareness is exploited to improve interoperability among Web resources. Semantic interoperability requires shared repositories of precisely defined concepts. Such repositories are called ontologies.

Ontologies. Ontologies are used in the SW to provide a common set of terms for describing and representing a domain in which automated tools might perform more accurate Web searches, allow intelligent software agents to operate and afford better knowledge management. For historical reasons (see Costa, 2005), initial SW research was limited to deterministic representations and reasoning. These are adequate to solve problems for which all the necessary information is available and complete. However, many of the SW reasoning challenges cannot assume a closed world environment, and thus need to deal with incomplete data to perform necessary tasks. When faced with incomplete information, deterministic reasoners typically apply default rules to enable conclusions to be drawn with incomplete information, but such defaults typically depend on ad hoc heuristics. In the open world case, logical systems may represent phenomena such as exceptions and unknown states with generic labels such as "other", but will lose the ability to draw strong conclusions. To overcome this limitation, some specific areas of Web research are already employing AI techniques, such as the Web Intelligence Community (see http://wi-consortium.org/ for further information). In probabilistic systems, uncertain phenomena would receive probability assessments, providing greater flexibility and more defensible results. There remain many important open research issues regarding open-world probabilistic reasoning (see Laskey, 1994), but Bayesian decision theory provides a principled foundation that is lacking in standard default reasoners.

The realization of the need for proper uncertainty representation and reasoning has already inspired many research efforts from the SW community. Examples include the URSW workshop series held at the ISWC (e.g. Costa et al., 2005; 2006), and the W3C's URW3 Incubator Group (Laskey *et al*., 2007). Still, there is a lack of standards and tools with support for either representing or reasoning with uncertain, incomplete information. UnBBayes-MEBN (Carvalho *et al*., 2007) is an initial step to change this situation, providing an application that ensures full support for uncertainty in the field of ontology engineering and, as a consequence, to the Semantic Web. Before embarking on the specifics of the application, some background information is needed.

Background Information

UnBBayes-MEBN implements the PR-OWL probabilistic ontology language (Costa, 2005), a probabilistic extension of the OWL based on the Multi-Entity Bayesian Networks logic – MEBN (Laskey, 2007).

MEBN integrates First Order Predicate Calculus with Bayesian probability, providing: (1) a means of expressing a globally consistent joint distribution over models of any consistent, finitely axiomatizable FOL theory; (2) a proof theory capable of identifying inconsistent theories in finitely many steps and converging to correct responses to probabilistic queries; and (3) a built in mechanism for adding sequences of new axioms and refining theories in the light of observations. Knowledge is expressed in MEBN as a collection of MEBN fragments (MFrags) organized into MEBN Theories (MTheories). An MFrag represents a conditional probability distribution of the instances of its resident random variables (RVs) given the values of instances of their parents in the fragment graphs and given the context constraints. RVs are graphically represented in an MFrag either as resident nodes, which have distributions defined in their home fragment, or as input nodes, which have distributions defined elsewhere. Context nodes are the third type of MFrag nodes, and represent conditions assumed for definition of the local distributions. A collection of MFrags represents a joint probability distribution of an unbounded, possibly infinite number of instances of its random variables. The joint distribution is specified by means of the local distributions together with the conditional independence relationships implied by the fragment graphs. Context terms are used to specify constraints under which the local distributions in an MFrag apply. A collection of MFrags that satisfies consistency constraints ensuring the existence of a unique joint probability distribution over its random variables is called an MTheory. MTheories can express probability distributions over truth-values of arbitrary FOL sequences and can be used to express domain-specific ontologies that capture statistical regularities in a particular domain of application.

Figure 1 depicts the Starship MTheory used by Costa (2005) as a toy example based on the famous Paramount Star Trek™ series. Each of its 11 MFrags represents the probability information about a group of their respective random variables. Collectively, the group implicitly expresses a JPD over truth-values of sets of FOL sentences. That is, probability distributions are specified locally over small groups of hypotheses and composed into globally consistent probability distributions over sets of hypotheses. MEBN theories extend ordinary Bayesian networks to provide an inner structure for random

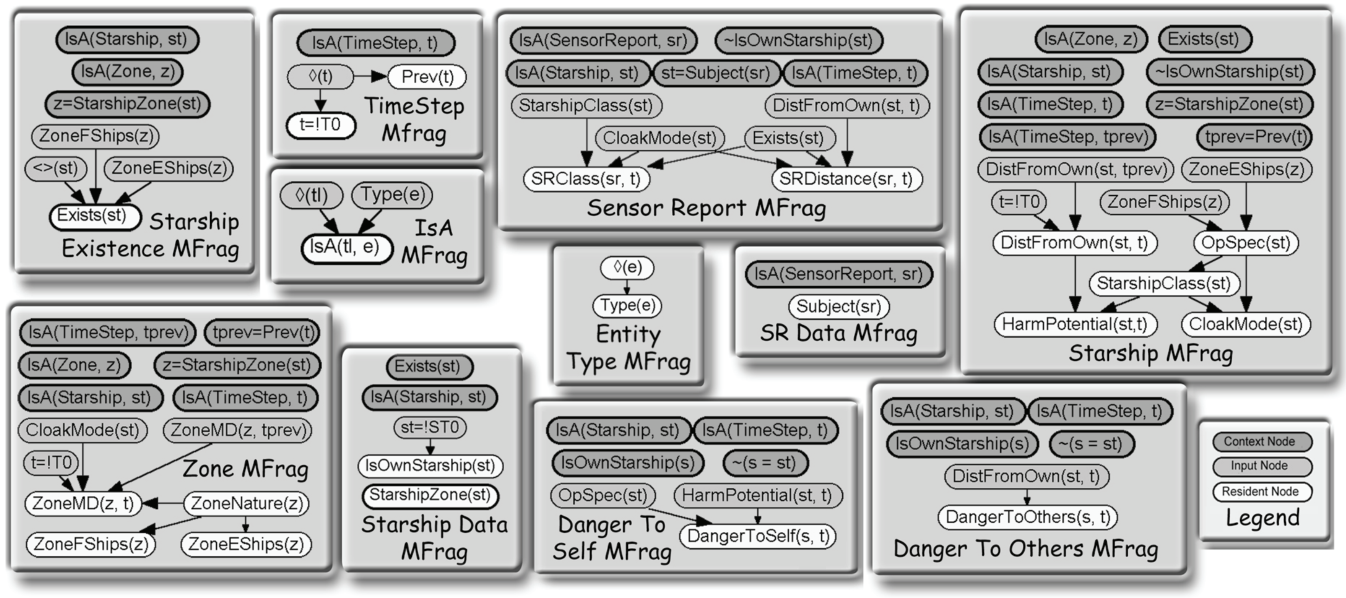


Figure 1 - The Starship MTheory

variables. Random variables in MEBN theories take arguments that refer to entities in the domain of application. As an example from the Sensor Report MFrag of Figure 1, the predicate StarshipClass(*st*) might represent the class of the starship designated by the variable *st*. To refer to the class of the starship instance labeled !Avenger, one would fill in a value for *st* to obtain an instance StarshipClass(!Avenger) of the StarshipClass(*st*) random variable. Note that, in MEBN syntax, an exclamation mark starts a string when it is used as an instance label. A given situation might involve any number of instances of the StarshipClass(*st*) random variable, referring to different starships. A comprehensive explanation of MEBN logic is outside the scope of this work, but the interested reader is directed to (Laskey, 2007) and (Costa & Laskey, 2005).

PR-OWL

The usual workaround for representing probabilities in deterministic languages like OWL is to show probability information as annotations. This means that numerical information is stored as text strings. Because this solution does not convey the structural features of a probabilistic domain theory, it is no more than a palliative. This is no minor shortcoming. Researchers have stressed the importance of structural information in probabilistic models (see Schum, 1994). For instance, Shafer (1988, pages 5-9) stated that probability is more about structure than it is about numbers.

Probabilistic Ontologies. A major concept behind PR-OWL is that of probabilistic ontologies, which goes beyond simply annotating ontologies with probabilities to provide a means of expressing all relevant uncertainties about the entities and relationships that exist in a domain in a logically coherent manner. This not only provides a consistent representation of uncertain knowledge that can be reused by different probabilistic systems, but also allows applications to perform plausible reasoning with that knowledge. PR-OWL uses the following definition of a probabilistic ontology (Costa, 2005): a probabilistic ontology (PO) is an explicit, formal knowledge representation that expresses knowledge about a domain of application. This includes: (a) types of entities that exist in the domain; (b) properties of those entities; (c) relationships among entities; (d) processes and events that happen with those entities; (e) statistical regularities that characterize the domain; (f) inconclusive, ambiguous, incomplete, unreliable, and dissonant knowledge related to entities of the domain; and (g) uncertainty about all the above forms of knowledge; where the term entity refers to any concept (real or fictitious, concrete or abstract) that can be described and reasoned about within the domain of application.

Probabilistic ontologies are used for the purpose of comprehensively describing knowledge about a domain and the uncertainty associated with that knowledge in a principled, structured and sharable way. PR-OWL was developed as an extension enabling OWL ontologies to represent complex Bayesian probabilistic models in a way that is flexible enough to be used by diverse Bayesian probabilistic tools based on different probabilistic technologies (e.g. PRMs, BNs, etc.). More specifically, PR-OWL is an upper ontology (i.e. an ontology that represents fundamental concepts that cross disciplines and applications) for probabilistic systems that is expressive enough to represent even the most complex probabilistic models. It consists of a set of classes, subclasses and properties that collectively form a framework for building POs.

Currently, the first step toward building a probabilistic ontology as defined above is to import the PR-OWL ontology (available at http://www.pr-owl.org/pr-owl.owl) into an ontology editor (e.g. OntoEdit, Protégé, Swoop, etc) and start constructing the domain-specific concepts, using the PR-OWL definitions to represent uncertainty about their attributes and relationships. Using this procedure, a knowledge engineer is not only able to build a coherent generative MTheory and other probabilistic ontology elements, but also make it compatible with other ontologies that use PR-OWL concepts. However, building MFrags this way is a manual, error prone, and tedious process that requires deep knowledge of the logic and of the data structures of PR-OWL in order to avoid errors or inconsistencies. UnBBayes-MEBN changes all that by providing a GUI-based editing process for building POs based on the PR-OWL upper ontology on probabilistic models (Carvalho *et al*., 2007).

The major advantages of using PR-OWL are its flexibility and representational power, both inherited from the fact that the language is based on MEBN, a full integration of First-Order Logic and probability that merges the expressiveness of the former with the inferential power of the latter. UnBBayes-MEBN leverages that power with a built-in MEBN reasoner that implements both the SSBN creation process and its respective evaluation. The next section provides an overall view of the current state of that tool. The prospective reader can find additional details on PR-OWL at http://www.pr-owl.org.

A First-Order Bayesian Tool

Implementing a complex logic such as MEBN while focusing on the usability requirements of an (probabilistic) ontology editor requires making trade-offs between performance, decidability, expressivity, and ease of use. In other words, the complexity of the logic and the fact that it is still in development imply that any implementation has to include alternative algorithms and optimizations to make a working, feasible tool. UnBBayes-MEBN is no exception to this rule, and many of the design decisions were based on the above-cited constraints.

POs in UnBBayes are saved in PR-OWL format (*.owl file), while application-specific data is stored in a text file with the *.ubf extension. Support for MEBN input/output operations is provided via the Protégé-OWL API, (http://protege.stanford.edu/plugins/owl/api/index.html), which is based on the class JenaOWLModel. By using a common API, UnBBayes-MEBN ensures that MTheories created using its GUI can be opened and edited in Protégé (and vice-versa). This compatibility is important because it ensures that files created in UnBBayes can be opened and edited not only in Protégé, but also in any OWL-compliant application (although these applications will not be able to understand the ontology's probabilistic characteristics). In addition, ontologies that have already been defined using an OWL-compliant editor can be extended to the PR-OWL format in a quick and direct way. All that is needed is to open the OWL file in UnBBayes and create an MTheory for this ontology and save the result.

Creating a MEBN Theory

Figure 2 depicts the Starship MFrag from the Starship MTheory of Figure 1 as displayed in UnBBayes-MEBN. The GUI not only facilitates the entry of context, input, and resident nodes, but also has a built-in consistency check that prevents many possible mistakes that would be easy to make in a manual editing process.

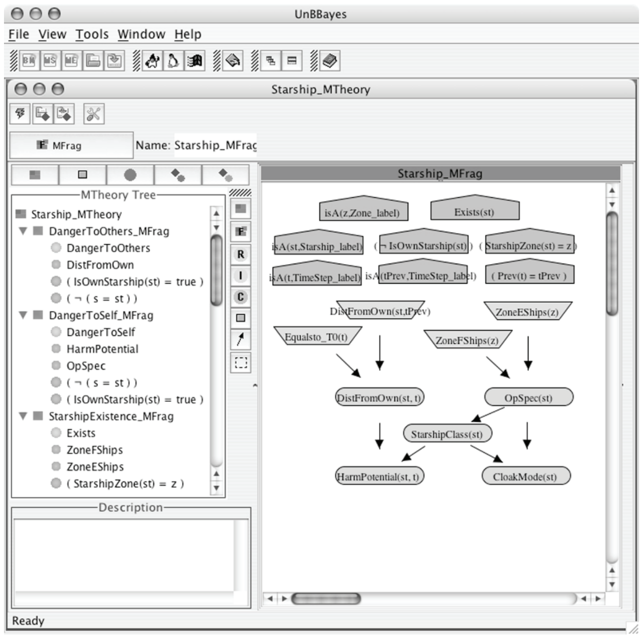


Figure 2 – Building the Starship MFrag in UnBBayes-MEBN

UnBBayes was designed to allow building POs in an intuitive way without having to rely on a deep knowledge of the PR-OWL specification. In the example, a click on the "R" icon and another click any-where in the editing panel will create a resident node, for which a description can be inserted in the text area at the lower left part of the screen. Clicking on the arrow icon would allow one to graphically define the probabilistic relations of that resident node with other nodes, as much as it would be done in current Bayesian packages such as Hugin™. All those actions would result in the software creating the respective PR-OWL tags (syntactic elements that denote particular parts of a PR-OWL ontology) in the background.

Logical Functions

MEBN, as a first-order Bayesian logic, poses the implementation challenge of how to evaluate FOL sentences. UnBBayes-MEBN implements all logical functionality via PowerLoom, an open-source API that is a Knowledge Representation and Reasoning (KR&R) tool available at http://www.isi.edu/isd/LOOM/PowerLoom. It provides a highly expressive, logic-based KR&R system with multiple built-in deductive reasoning capabilities including a query processor, a description classifier, and a context mechanism. Figure 3 shows the context node formula editor, in which the user chooses graphically the formula arguments and operators, while the GUI uses PowerLoom in the background to build and evaluate the resulting logical expression.

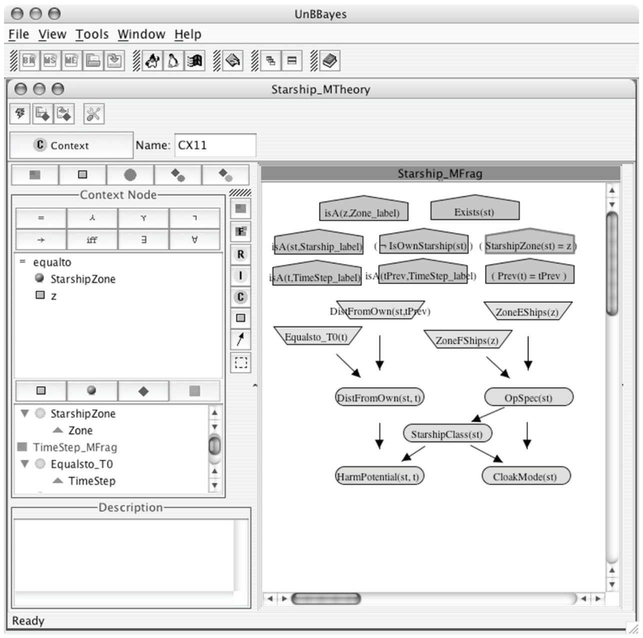


Figure 3 – The Formula Editor for context nodes

SSBN Construction Algorithm

UnBBayes-MEBN also includes a knowledge base (KB) to store domain information. When a query is submitted, the KB is searched for information to answer the query. If the available information does not suffice, then the KB and the generative MTheory are used to construct a BNto answer the query. This process is called SSBN construction.

In the current implementation, a query consists of a single random variable (RV) instance, which is not allowed to have any evidence below it. The following procedure takes a node name and a list of entity instances as arguments. It is called initially with the query node and its arguments.

PROCEDURE SSBN-CNSTR(NODE, ENTITY-LIST):

(i) For the RV instance NODE(ENTITY-LIST), search for evidence in the KB. If there is a finding for this given entry, finish.

(ii) Search for the resident node that has the name NODE and get its MFrag. Once NODE(OV-LIST) is found, verify if the type of ENTITY-LIST is the same as OV-LIST (where OV-LIST is the list of ordinary variable arguments for NODE in its home MFrag).

(iii) Verify in the KB which context nodes refer to the OVs in OV-LIST, replacing each OV by the appropriate instance in ENTITY-LIST. If any context variable is false, mark the MFrag to use the default distribution.

(iv) If the truth-value of the context node in (iii) is not determined, make it a parent of NODE.

(v) For each parent of NODE, identify any instance of the parent that can be constructed by replacing the OVs by the known entities (contained in the query or KB), and has not yet been added to the SSBN. For each such parent instance, call procedure SSBN-CNSTR for the parent node and its arguments.

(vi) Create the NODE's CPT.

(vii) Finish.

This algorithm is easily enhanced to allow multiple query nodes and evidence below query nodes. These enhancements are currently under development.

A few performance issues had to be considered in the implementation of UnBBayes-MEBN. Depending on the complexity of the domain, the algorithm may reach a context node that cannot be immediately evaluated. This happens when all ordinary variables in the parents set of a resident random variable term do not appear in the resident term itself. In this case, there may be an arbitrary, possibly infinite number of instances of a parent for any given instance of the child. For example, in the Starship MFrag depicted in Figures 1 and 2, if the zone where a starship is located is uncertain, the number of enemies and friends (ZoneEStarships(*z*) and ZoneFStarships(*z*)) in any zone it might be located is relevant to the distribution of the OpSpec(*s*t) random variable. If time step *t* has previous time steps, then more than one distance (DistanceFromOwn(*st*, *tprev*)) must be evaluated, which makes the distance measured in all time steps relevant to the distribution of the DistFromOwn(*st*, *tprev*) random variable in time t. Thus, any number of instances of the ZoneEShips(*z*), ZoneFShips(*z*), and DistFromOwn(*st*, *tprev*) random variables might be relevant to the distributions of the OpSpec(*st*) and DistFromOwn(*st*, *tprev*) random variables in time step *t*. In this case, the local distribution for a random variable must specify how to combine influences from all relevant instances of its parents.

However, especially in complex formulas this may have a strong impact in the performance of the algorithm, so the designed solution involves asking the user for more information. In the current implementation, if one does not provide such information the algorithm will just halt. Another design option was to restrict memory usage in a way that a possible memory overload triggers a warning to the user and stops the algorithm. In step (iii), a design optimization over the general SSBN algorithm in (Laskey, 2007), only the necessary context nodes for a given MFrag are evaluated, in contrast with the original solution of revising all the context nodes for that MFrag. Although the implementation addressed other optimization issues, for the sake of conciseness only the most relevant are listed here.

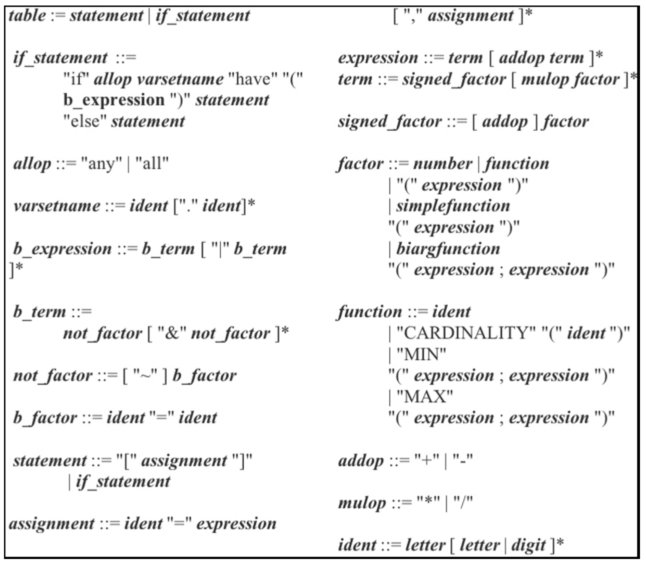
Modifications to the Original PR-OWL

PR-OWL was designed as a general formalism for building POs, without a focus on implementing an actual tool such as UnBBayes-MEBN. Thus, some contributions from this work were actually introduced into the revised PR-OWL specification. One example is the grammar defined for writing formulas to dynamically build CPTs, which addresses a key aspect of reasoning process: the generation of the combined probability tables (CPT) for nodes with dynamically defined distributions. More specifically, when a query is posed to the system, it triggers the algorithm for building the Situation Specific Bayesian Network (SSBN) that will answer the query by instantiating all random variables that can add information towards its solution. In this process, some nodes may have an unknown number of parents, so instead of a static, previously defined CPT there will be a formula for dynamically generating the CPT given the number of parents in that specific situation. Although the original work on the PR-OWL language does not contain a rigid specification for representing formulas that perform the dynamic definition of probability distributions of an instantiated MFrag's nodes, it presents pseudo code (Costa, 2005, page 64) that was used as a basis for specifying conditional probabilistic tables (CPT) in UnBBayes-MEBN. The implementation of this CPT generator is a key aspect for the SSBN construction, as many real world models include nodes with no previously defined number of parents. In UnBBayes-MEBN, a grammar and a complete new compiler, featuring a lexical, syntactic, and semantic analyzer were designed from the ground up. During the SSBN construction algorithm, they act by evaluating dynamic probability distribution formulas and then building the CPT for the respective node.

Figure 4 shows a sample of the grammar's pseudo-code, which can be understood as a sequence of probability assignment blocks conditioned by *if-else* clauses. The clause *else* has a special purpose: it not only declares non-specified probabilities, but also establishes the *default* distribution (i.e. the one used when a context node is not satisfied).

Evaluation of the pseudo-code is performed via syntactical analysis by a recursive descent parser, which is a top-down technique in which each production rule is implemented via a procedure. Commands and additional data structures were also added to the grammar as a means to allow for a semantic analysis of the code and for the generation of intermediate code. The latter is composed of specific blocks that are evaluated once all random variable conditioning cases are known, and is responsible for performing the final SSBN CPT generation. As an example of an element of the grammar, the production rule *varsetname* declares how the pseudo-code references a given set of parent nodes. Parent node instances can be divided into subsets that have similar predominant arguments. To better understand the concept, suppose that all parent nodes that have arguments *st* and *z* as their respective predominant arguments form a subset of parent nodes. Then, a hypothetical condition "*if* any ***st.z***" would be applied only to members of that subset. Currently, non-predominant arguments (weak arguments) are the recursive ordinary variables (e.g. ordinary variable *t* in DangerToSelf(*s*, *t*)).

Another contribution to PR-OWL made during the development of the tool was to include information on global exclusivity for a node's state. This is necessary in situations when only one finding is allowed for a specific node in a given state. For instance, in Starship Data MFrag of Figure 1's MTheory, the IsOwnStarship(*st*) has the state True as possible for just one starship *st*. That is, the state True has globally exclusive with respect to the random variable IsOwnStarship(*st*). Global Exclusivity was accepted as a contribution by the PR-OWL team, and was inserted in PR-OWL version 1.05 (www.pr-owl.org).



```
table := statement | if_statement

if_statement ::=
        "if" allop varsetname "have" "("
        b_expression ")" statement
        "else" statement

allop ::= "any" | "all"

varsetname ::= ident ["." ident]*

b_expression ::= b_term [ "|" b_term
]*

b_term ::=
        not_factor [ "&" not_factor ]*

not_factor ::= [ "~" ] b_factor

b_factor ::= ident "=" ident

statement ::= "[" assignment "]"
        | if_statement

assignment ::= ident "=" expression
        [ "," assignment ]*

expression ::= term [ addop term ]*
term ::= signed_factor [ mulop factor ]*

signed_factor ::= [ addop ] factor

factor ::= number | function
        | "(" expression ")"
        | simplefunction
        "(" expression ")"
        | biargfunction
        "(" expression ; expression ")"

function ::= ident
        | "CARDINALITY" "(" ident ")"
        | "MIN"
        "(" expression ; expression ")"
        | "MAX"
        "(" expression ; expression ")"

addop ::= "+" | "-"

mulop ::= "*" | "/"

ident ::= letter [ letter | digit ]*
```

Figure 4 – Grammar used for dynamically generating a CPT

Conclusion

Proper representation and reasoning with uncertainty is a topic of growing interest within the Semantic Web community. However, there are no established standards and most of the research in this direction does not provide knowledge engineers with appropriate tools. The PR-OWL/MEBN formalism provides a promising framework to address the lack of standards, but so far no implementation was available to people interested in using it. This work presented UnBBayes-MEBN, an open source, Java-based software distributed under GNU GPL license that addresses this problem, providing support for domain modeling and reasoning with POs based on this formalism. Due to the state-of-the-art status of our tool, we just have preliminary, anecdotal evaluation of its results. Thus, our focus was on the solutions we've employed to the most daunting tasks in devising a MEBN/PR-OWL reasoner and GUI, while also emphasizing its potential applications within the interest of the FLAIRS audience. Nonetheless, it represents a contribution to the SW community and, more specifically, to the current work of the URW3-XG Incubator Group, created by the W3C to better define the challenge of reasoning with and representing uncertain information available through the World Wide Web and related technologies.

References

Carvalho, R. N., Santos, L. L., Ladeira, M., and Costa, P. C. G. 2007. A Tool for Plausible Reasoning in the Semantic Web using MEBN. In *Proceedings of the Seventh International Conference on Intelligent Systems Design and Applications*, 381-386. IEEE Press.

Costa, P. C. G. 2005. Bayesian Semantics for the Semantic Web. PhD Diss. Department of Systems Engineering and Operations Research, George Mason University. 315p, July 2005, Fairfax, VA, USA.

Costa, P. C. G., and Laskey, K. B. 2005. Multi-Entity Bayesian Networks without Multi-Tears. Draft, Department of Systems Engineering and Operations Research, George Mason University, Fairfax, VA, USA, 2005. Available at http://hdl.handle.net/1920/456.

Costa, P. C. G., Laskey, K. B.; Laskey, K. J., Pool, M. eds., 2005. Proceedings of the First ISWC Workshop on Uncertainty Reasoning for the Semantic Web. Galway, Ireland.

Costa, P. C. G., Fung, F., Laskey, K. B., Laskey, K. J., Pool, M. eds. 2006. Proceedings of the Second ISWC Workshop on Uncertainty Reasoning for the Semantic Web. Athens, GA, USA.

Laskey, K.B., MEBN: A Language for First-Order Bayesian Knowledge Bases, Artificial Intelligence, 172(2-3), 2007

Laskey, K.B., and Lehner, P.E. 1994. Meta Reasoning and the Problem of Small Worlds. *IEEE Transactions on Systems, Man and Cybernetics*, 24(11), 1643-1652.

Laskey, K.J., Laskey, K.B. & Costa, P.C.G. eds. 2007. Uncertainty Reasoning for the World Wide Web Incubator Group Charter (W3C Incubator Activity). Available at http://www.w3.org/2005/Incubator/urw3/charter.

Shafer, G. 1988. Combining AI and OR. University of Kansas School of Business, Working Paper No. 195. April.

Spiegelhalter, D. J., Thomas, A., and Best, N. 1996. Computation on Graphical Models. *Bayesian Statistics*, 5: 407-425.

Schum, D.A. 1994. Evidential Foundations of Probabilistic Reasoning, John Wiley & Sons, Inc., New York, NY.