18th International Conference on Information Fusion
Washington, DC - July 6-9, 2015

Scalable Uncertainty Treatment Using Triplestores and the OWL 2 RL Profile

Laécio L. Santos*, Rommel N. Carvalho*†, Marcelo Ladeira*, Li Weigang*,
Kathryn B. Laskey‡, and Paulo C. G. Costa‡
*Department of Computer Science
University of Brasília, Brasília, Brazil
Email: laecio@cic.unb.br, [rommelnc,mladeira,weigang]@unb.br
†Department of Research and Strategic Information
Brazil's Office of the Comptroller General, Brasilia, Brazil
Email: rommel.carvalho@cgu.gov.br
‡Center of Excellence in C4I / The Sensor Fusion Lab
George Mason University, Fairfax, VA 22030-4444 U.S.A.
Email: [klaskey,pcosta]@gmu.edu

***Abstract*—The probabilistic ontology language PR-OWL (Probabilistic OWL) uses Multi-Entity Bayesian Networks (MEBN), an extension of Bayesian networks with first-order logic, to add the ability to deal with uncertainty to OWL, the main language of the Semantic Web. A second version, PR-OWL 2, was proposed to allow the construction of hybrid ontologies, containing deterministic and probabilistic parts. Existing PR-OWL implementations cannot deal with very large assertive databases. This limitation is a main obstacle for applying the language in real domains, such as Maritime Domain Awareness (MDA). This paper proposes a PR-OWL extension using RDF triplestores and the OWL 2 RL profile, based on rules, in order to allow dealing with uncertainty in ontologies with millions of assertions. We illustrate our ideas with an MDA ontology built for the PROGNOS (PRobabilistic OntoloGies for Net-centric Operation Systems) project.**

***Index Terms*—probabilistic ontology, uncertainty reasoning, PR-OWL, MEBN, OWL 2 RL, triplestores.**

I. INTRODUCTION

The advance of Big Data and the Internet of Things has brought unprecedented growth in available information from from different sensors and devices. This creates a major challenge in how to select, combine, and use the data to derive new knowledge and insights for decision making. Fusion of information from multiple sources promises to overcome the limitations of each source individually and achieve synergy.

Low-Level fusion (LLF) concerns locating and identifying individual entities by fusing information from multiple sources [1]. LLF has been widely studied and is considered a mature field [2]. Success in LLF has naturally led to increased interest in high-Level fusion (HLF), the problem of identifying the situation, understanding the relationships among the involved entities, and calculating the impact of actions being considered. HLF technologies require the use of expressive representational frameworks capable of explicitly representing the semantics of the domain [3].

Maritime Domain Awareness (MDA) is the effective understanding of anything associated with the maritime domain that could impact security, safety, economy or environment. Achieving MDA requires integrating information from multiple sources in order to provide a global vision of the maritime environment. MDA is essential for identifying and planning preventive action in response to threats such as piracy, terrorism, and transport of illegal cargo. A probabilistic ontology (PO) for the MDA domain was developed for PROGNOS (PRobabilistic OntoloGies for Net-centric Operation Systems) [4], [5], a naval predictive situation awareness system devised to work within the context of U.S. Navy's FORCENet. The PROGNOS PO combines data from several sources and reasons with it in order to provide predictive information.

The PROGNOS PO was built using PR-OWL (Probabilistic OWL) [6], an extension of OWL (Ontology Web Language) that permits representing and reasoning with uncertain information. OWL is the W3C's standard for defining ontologies for the Semantic Web. PR-OWL is based on Multi-Entity Bayesian Networks (MEBN), an extension of Bayesian networks with first-order expressive power, and can represent uncertainty associated with aspects of an ontology.

Current PR-OWL reasoners do not deal with very large assertive databases, since they require that the data must be loaded into memory during inference time. PROGNOS was used only on a simulation platform, and was tested with only about 2,000 entities. Real applications may require millions of statements, demanding a scalable implementation of PR-OWL.

Triplestores can store RDF (Resource Description Framework) triples in databases optimized to work with graphs making it possible to work with ontologies that have a large assertive base. RDF Oracle Spatial and Graph and AllegroGraph, two commercial triplestores, are able to handle trillions of RDF triples[1], showing the scalability of this alternative to relational databases. Some triplestores, such as GraphDB, allow processing and inference with restrictive versions of OWL, such as the profile OWL 2 RL, which is based on rules.

[1] https://www.w3.org/wiki/LargeTripleStores

The objective of this work is to extend PR-OWL to work with triplestores and with the OWL 2 RL profile, allowing treatment of uncertainty in ontologies with millions of assertions. This paper also proposes an implementation of this new language within UnBBayes, a framework for probabilistic reasoning [7], developed by the Group of Artificial Intelligence of the University of Brasilia. We use the MDA probabilistic ontology of the PROGNOS project to illustrate our ideas.

This paper is organized as follows: Section II presents MEBN, PR-OWL, and its implementation in UnBBayes. Section III presents the OWL 2 RL profile and how it is implemented by triplestores. Section IV discuss the problems of scalability of the current PR-OWL implementations in UnBBayes. Section V presents PR-OWL 2 RL, a language that joins PR-OWL with the OWL 2 RL profile implemented in triplestores. Finally, Section VI presents concluding remarks.

II. PR-OWL AND MEBN

This section presents MEBN and PR-OWL as well as their implementation in the UnBBayes framework.

A. Multi-Entity Bayesian Networks

Multi-Entity Bayesian Network (MEBN) is a language for representing first-order probabilistic knowledge bases [8]. MEBN extends Bayesian networks by incorporating the expressiveness of first-order logic (FOL), addressing an important limitation of Bayesian networks: the inability to represent problems in which the number of uncertain variables is unknown and may vary from situation to situation.

MEBN represents domain knowledge as a collection of model templates called MEBN Fragments, or MFrags. Each MFrag represents a modular element of knowledge about a set of related entities, including uncertainty about their attributes and relationships. MFrags contain arguments to be filled in by specific domain entities. A set of MFrags satisfying consistency constraints ensuring the existence of a unique probability distribution is called an MTheory [8]. As information arrives about the specific entities involved in a given situation, MFrags are retrieved and assembled into a problem-specific model called a situation-specific Bayesian network, which is used to reason about the situation.

Figure 1 illustrates the MFrag `Meeting`, from the MDA ontology, presented in [5]. Resident nodes and input nodes represent properties and relationships of entities, with arguments (ordinary variables) to be filled during model instantiation. A resident node has its local probability distribution (LPD) defined into its home MFrag, while an input node has its LPD defined in another MFrag, in which it is a resident node. Context nodes are formulas in first-order logic that define constraints that must be satisfied for the LPD to be used (otherwise, its default distribution will be used).

Inference in MEBN is performed by constructing a Situation-Specific Bayesian Network (SSBN), a minimal Bayesian network sufficient to compute the answer to a query. A query consists of obtaining the posterior distribution for a set of query random variables given a set of evidence and

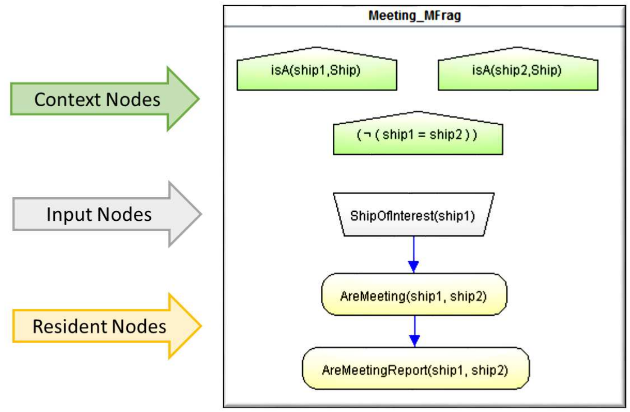

Fig. 1. Meeting MFrag, from the Maritime Domain Awareness ontology

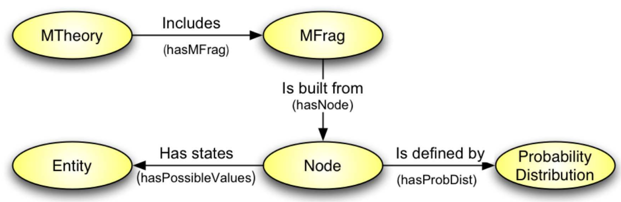

Fig. 2. PR-OWL main concepts

context variables [9]. A bottom-up algorithm for generating a SSBN is presented in [8].

B. PR-OWL and PR-OWL 2

The PR-OWL language, proposed by Costa in 2005, adds uncertainty support to OWL. PR-OWL consists of a set of classes, subclasses, and properties that collectively form a framework for building probabilistic ontologies [6].

Figure 2, extracted from [6], shows how PR-OWL models the main concepts involved in defining a MEBN theory. The user models the probabilistic part of the ontology using the MTheory class, composed of a set of MFrags (connected to the MTheory through the hasMFrag property) which collectively should form a consistent MTheory. The MFrags are built from nodes. The nodes represent random variables. Each node has an associated probabilistic distribution and an exhaustive set of possible states, where states are individuals from the Entity class.

In 2011, Carvalho proposed PR-OWL 2 [10], an extension of PR-OWL that solves two of its major limitations: the lack of formal mapping between the random variables from PR-OWL and concepts defined in OWL; and the lack of compatibility between PR-OWL and OWL types [11].

The first limitation is addressed by defining the relationship `definesUncertaintyOf`, that links random variables from PR-OWL to OWL properties. Additionally, the relationships `isSubjectIn` and `isObjectIn` are used to define the domain and range of the random variable using

OWL concepts. Though this feature, PR-OWL facilitates the construction of hybrid ontologies, containing interrelated probabilistic and deterministic statements.

To solve the second limitation, PR-OWL 2 maps PR-OWL types to the types already present in OWL. The class `ObjectEntity`, for example, is replaced by the class `Thing`, while the class `CategoricalRVStates` is replaced by `DataOneOf`, to enumerate *data types*, or `ObjectOneOf` for objects [11].

PR-OWL 2 also contains other improvements, such as the ability to work with polymorphism in random variables.

C. The UnBBayes MEBN/PR-OWL Plugins

UnBBayes is a framework developed to work with probabilistic reasoning. Created by the Artificial Intelligence Group from the University of Brasilia (GIA-UnB), and written in Java, UnBBayes is open-source software, distributed under the GPL licence, available through the Source Forge[2] site. UnBBayes implements several formalisms, including Bayesian networks, Influence Diagrams, Multiply-Sectioned Bayesian Networks (MSBN), Object-Oriented Bayesian Networks (OOBN), Hybrid Bayesian Networks (HBN), and Multi-Entity Bayesian Networks (MEBN). The current architecture of UnBBayes is based on plug-ins [7], facilitating the implementation of new formalisms.

Support for MEBN and PR-OWL was first developed in 2008 [12], [13]. This was the first implementation of MEBN. This implementation includes a graphical user interface for visual modeling of an MTheory; a knowledge representation and reasoning system; and an algorithm for generating an SSBN, based on the algorithm proposed in [8]. PR-OWL is used as the format for persisting MTheories.

The UnBBayes PR-OWL implementation uses PowerLoom [3], a Knowledge Representation and Reasoning (KR&R) system, to perform logical reasoning. A PowerLoom Knowledge Base is composed of a TBox, containing the terminological vocabulary, and an ABox, containing the assertive base, over the defined TBox. PowerLoom is used for storing and querying the assertion database (ABox), containing the entities and evidence for a specific situation, and for evaluating the FOL expressions in the context nodes. PowerLoom uses as representation language a variant of KIF (Knowledge Interchange Format), a language developed for the exchange of knowledge between different computer systems.

The Pr-OWL knowledge base is loaded in two parts: first, the TBox is created and loaded into PowerLoom, translating the MTheory model to KIF; then, the ABox, containing entities, properties, and relationships, is either loaded from a KIF file specified by the user or specified directly through the UnBBayes GUI. The user can record different KIF files containing different sets of findings, but any of these must be compatible with the TBox. Since PowerLoom has support for first-order logic, the mapping of MEBN elements to PowerLoom elements is straightforward. An extra format, ubf (UnBBayes File), is used to store graphical features (*e.g.*, size and position of the nodes in the canvas), since PR-OWL does not provide support for storing this type of information.

The PR-OWL 2 plug-in was implemented in 2011 [14]. Since PR-OWL 2 allows the modeling of hybrid ontologies, adding uncertainty information to a deterministic ontology, it was important to allow users to work simultaneously with both representations. Protégé[4] is a popular and mature open source framework for representing and editing ontologies. It has been integrated into UnBBayes to allow the modeling of the deterministic part. Protégé was encapsulated as an internal panel in UnBBayes, allowing the user to access all its functionality. The user models the deterministic part of the ontology in Protégé, and the probabilistic part in UnBBayes' MEBN GUI. A panel allows the user to link an OWL property to a resident node in the MTheory, indicating that the latter represents the uncertainty related to the former. This link is mapped to a `definesUncertaintyOf` PR-OWL 2 property.

In the PR-OWL 2 implementation, the knowledge base is saved in OWL 2, eliminating the need for a separate KIF file to store the ABox. HermiT [15], the default reasoner in Protégé, is used to search for information in the assertive database and to evaluate the context nodes. HemiT is based on description logic, offering support for OWL 2 DL. Some restrictions were necessary in the context node formulas since PR-OWL 2 accepts all valid first-order expressions, while HermiT is only able to support a subset of first-order logic. These restrictions are discussed in Section IV.

III. The OWL 2 RL Profile

OWL (Web Ontology Language) is a representation language based on formal logic, used for constructing complex ontologies. In 2004, OWL was adopted as a W3C Recommendation for the development of ontologies in the Semantic Web. In 2012, the revised OWL 2 specification was adopted. OWL is built on RDF, and typically uses RDF/XML for persistence. RDF is a formal language used for describing structured information [16], representing statements through RDF triples, in which predicates link subject nodes to object nodes. An RDF graph is a set of triples in which common nodes are unified.

OWL 2 has a DL version, based on the description logic $\mathcal{SROIQ}$(D), and a Full version, which is undecidable. It includes three profiles, or sub-languages (syntactic subsets), developed for specific applications that possess properties that allow the development of efficient algorithms: OWL 2 EL, based on description logic $\mathcal{EL}++$, recommended for applications involving a large number of properties and classes [17]; OWL 2 QL, based on conjunctive queries, recommended for applications with large amounts of data where the answer to the query is the most important task [18]; and OWL 2 RL, based on rules, recommended for applications that need a scalable reasoning without much sacrifice in expressiveness [18].

[2]http://sourceforge.net/projects/unbbayes
[3]http://www.isi.edu/isd/LOOM/PowerLoom/
[4]http://protege.stanford.edu/

The OWL 2 RL profile allows the implementation of algorithms in polynomial time relative to the size of the ontology for the standard types of inference: ontology consistency, class expression satisfiability, class expression subsumption, instance checking, and conjunctive query answering [18].

OWL 2 RL is based on pD* and DLP (Description Logic Programs). pD*, proposed by ter Horst, extends both RDFS and D* entailments. It is largely defined by means of IF conditions, and it applies to a property-related subset of the OWL vocabulary [19]. Different from SWRL [20] (Semantic Web Rule Language), a more expressive approach which integrates OWL DL with rules, pD* is decidable. DLP consists in a language formed by a subset of OWL DL added to Datalog, being less expressive than both formalisms [16]. A datalog rule is a logical implication that may only contain conjunctions, constant symbols, and universally quantifiers variables, but no disjunctions, negations, existential quantifier, or function symbols [16].

W3C established a partial axiomatization, consisting of a set of implication rules based on RDF semantics. This axiomatization can be used to implement the OWL 2 RL profile using rule-based technologies operating over RDF databases [17]. This set of rules is presented as OWL 2 RL/RDF [18], and can be applied to OWL ontologies serialized in RDF. The rules are given as universally quantified first-order implications over a ternary predicate `T` [18], where `T` represents an RDF triple in the format `T(s, p, o)`, where `s` is the subject, `p` is the predicate and `o` is the object. Using this set of rules, the inference can be accomplished by materialization. Materialization consists of expanding the rules in loading time by calculating all new expressions that emerge from the set of added expressions.

RDF triplestores implementing the OWL 2 RL profile are becoming popular [21]. RDF triplestores are databases that store information structured using RDF triples, forming graphs. Triplestores use a flexible ontological schemata where data is processed by an inference engine according to well-defined semantics [22]. Besides the RDF semantics, most triplestores implement RDF(S), and some of them work with restrictive versions of OWL, such as the profile OWL 2 RL, making it possible to work with ontologies that have large assertive bases. Examples of commercial triplestores that implement fully or partially the OWL 2 RL profile include GraphDB [22], Oracle Spatial and Graph and AllegroGraph.

IV. Scalability of UnBBayes/MEBN

Neither the PR-OWL and PR-OWL 2 implementations in UnBBayes is adequate to work with ontologies with very large assertive databases. This is due to the fact that PowerLoom and HermiT, inference engines used respectively in PR-OWL and PR-OWL 2, work only with data stored in memory, and they consume large amounts of time to evaluate formulas in ontologies with many assertions.

OWL DL reasoners, like HermiT, are good for working with complex ontologies, offering complete answers, but present scalability problems. HermiT makes inferences in OWL 2 DL using the Direct Semantics, based on Description Logic. Tableau reasoners, like Pellet, Racer, and FaCT++, perform consistency tests trying to build a model for knowledge base [15]. HermiT implements a calculus "hipertableau" which reduces greatly the number of possible models to be considered [15], but nevertheless scalability remains a problem. Moreover, these reasoners are limited to the available memory of the computational resource used, since the full database needs to be loaded into memory to allow inference.

TABLE I
Size of LUBM test bases

	Size	Inst. Classes	Inst. Properties
LUBM 1	8.02 MB	20,659	82,415
LUBM 10	102 MB	263,427	1,052,895
LUBM 100	1.06 GB	2,779,262	11,096,694
LUBM 500	12 GB	13,839,128	55,240,636

The main reasoning problems for OWL 2 DL (ontology consistency, class expression satisfiability, class expression subsumption and instance checking) have complexity N2EXPTIME-complete: they are in the class of problems solvable by nondeterministic algorithm in time that is at most double exponential in the size of the input [18]. This limits the performance and scalability of the possible reasoners: query time grows intractably as the knowledge base becomes large. Therefore, implementations of PR-OWL 2 that use OWL 2 DL Reasoners are inherently non-scalable.

In order to evaluate the size of the ontology the user could use in the implementation of PR-OWL in UnBBayes, we ran some tests using LUBM (Lehigh University Benchmark) [23], a benchmark widely used for performance tests in OWL reasoners and RDF triplestores. LUBM consists of an OWL Lite ontology that models an academic domain, an automatic generator of test bases that allows the creation of ABox bases with varying amounts of assertive statements, and a set of fourteen queries of different complexities. Table I shows the number of statements and the physical size of some test bases. Using UnBBayes, running on a machine i5 with 6Gb of memory (3Gb dedicated to the JVM process) it was possible to load only the LUBM 10 version, containing 263,427 instances of classes and 1,052,895 instances of properties. This test case has only 102 MB, making it clear that the structure used adds a great overhead to the implementation.

There are also some practical limitations on the plug-in developed, such as the need for new instances and relationships to be manually entered by the user, using Protégé's interface. This task tends to be slow and repetitive when there are many instances and relationships. Working with ontologies containing thousands of assertions is only possible through the use of API and tools that automate this insertion. It is not clear how the user would work with different sets of assertions, since the implementation of PR-OWL 2 saves the ABox and TBox in the same file. These limitations make the modeling and use of POs in UnBBayes possible only for simple domains.

Furthermore, the PR-OWL 2 plug-in has limitations on expressiveness due to restrictions in the evaluation of context nodes. The implementation of PR-OWL uses PowerLoom to evaluate context nodes, having full support for first-order logic, including quantifiers. The implementation of PR-OWL 2, however, required simplifications in the formats of accepted formulas, since HermiT, the reasoner used, is based on the description logic OWL 2 DL, which is a fragment of first-order logic. According to [14]:

> The DL natively implements the `builtInRV` (operations like `and`, `or`, `not`, `forAll`, and `exists` are implemented natively). However, because of expressive differences between FOL (used in expressions of the context nodes formulas) and DL, the formulas of the context nodes can not be directly mapped to queries in ontology PR-OWL 2, mainly because DL queries may not be done for several ordinary variables simultaneously.

Table II presents the restrictions in the context node formula formats that were made in the implementation of PR-OWL 2. In the table, `ov` refers to an ordinary variable, which will be filled with an entitiy during the evaluation of an MFrag, `BooleanRV` refers to boolean random variable, `nonBooleanRV` refers to a non-Boolean random variable, and `CONST` refers to a constant. According to the table, only simple formulas are allowed, without the use of connectives and quantifiers.

V. PR-OWL 2 RL

The alternative proposed in this work to address the scalability problem is extend PR-OWL 2 to use triplestore RDF together with the profile OWL 2 RL, for storing OWL data and for performing inference. This extension makes it possible to work with probabilistic ontologies that have assertive databases large enough to preclude the use of traditional OWL DL reasoners. The OWL 2 RL profile has polynomial time for the main types of reasoning encountered in our applicaitions, as compared with the N2EXPTIME-complete of OWL 2 DL. Furthermore, triplestores support only versions of OWL with limited expressiveness, making it impossible to work with OWL 2 DL. The OWL 2 RL profile is implemented by several triplestores.

The proposed language, PR-OWL 2 RL, maps the PR-OWL 2 language to be represented in OWL 2 RL, respecting the restrictions of the profile. The new language also has restrictions on the valid formats for context nodes, making them evaluable by triplestores using the SPARQL query language. A plug-in will be developed in UnBBayes in order to allow representation and inference of POs designed in PR-OWL 2 RL.

We performed tests with the triplestores Sesame [5], Jena TDB [6], and GraphDB Lite [7]. These were selected because they are free and popular in the Semantic Web community. We used the LUBM benchmark for checking the storage capacity and reasoning of these alternatives. Although all versions were able to load the LUBM 100, only GraphDB correctly answered the fourteen queries proposed in the benchmark. We could not load larger versions of the assertive base of LUBM because, in the version used in the tests (GraphDB Lite), all the data are loaded in memory (we ran the tests with only 3 GB dedicated to the JVM running the GraphDB server).

Based on these tests, we choose GraphDB Lite, from Ontotext, for implementing our plug-in. The implementation of the profile OWL 2 RL of the GraphDB triplestore is based on the rule set OWL RL/RDF defined by W3C [22], following RDF semantics. Inference is performed by total materialization: all entailment statements are computed at load time [22]. The reasoner uses predominantly forward-chaining to apply the selected inference rules directly to RDF triples [24].

Deterministic inference in PR-OWL elements, using the semantic repository, suffer the limitations of rule-based reasoning. However, it is still possible use an OWL DL reasoner to make inferences over the terminological component (TBox). In fact, the knowledge engineer can use the UnBBayes' PR-OWL 2 plug-in to build and test the terminological component of the ontology, and use the new plug-in only when it is necessary to process a large assertive database (ABox). Note that in our implementation, the TBox has to be mapped to a triplestore prior to loading the assertion data. At this point, the TBox should already be complete, since the introduction of new terminology statements is costly to the database, which, using materialization, has to redo the inferences to instances of ABox preloaded.

The interface between UnBBayes and the triplestore is achieved through the SAIL (Storage and Inference Layer) Sesame API, that abstracts the details of storage and inference. This API is implemented by several triplestores, allowing the use of alternatives to GraphDB.

The user workflow with the proposed tool is described below:

1) The TBox, containing both the probabilistic and deterministic parts in UnBBayes, is defined. Modeling of the deterministic part is done in Protégé, incorporated into UnBBayes, and must hold the restrictions of the OWL 2 RL profile. Modeling of the probabilistic part is done through the creation of an MTheory using the UnBBayes GUI, leaving to the modeler the link between PR-OWL to OWL properties.
2) The TBox is loaded into the triplestore.
3) The assertive data (ABox) triplestore is populated with individuals and their properties occurring in the domain.
4) UnBBayes connects to the RDF triplestore using the Sesame SAIL API.
5) Probabilistic queries are made using the UnBBayes interface, using the SSBN derived from the MTheory and knowledge base.
6) Deterministic queries may be performed using the interface provided by the semantic repository. Alternatively, GUI for SPARQL queries could be constructed in UnBBayes, as it is connected to the triplestore.

[5] http://rdf4j.org/
[6] https://jena.apache.org/
[7] http://ontotext.com/products/ontotext-graphdb/

TABLE II
FORMATS OF VALID CONTEXT NODE FORMULAS IN THE PR-OWL 2 IMPLEMENTATION

Formula	Negation
$ov_1 = ov_2$	NOT ($ov_1 = ov_2$)
booleanRV(ov_1 [, ov_2 , ...])	NOT booleanRV(ov_1 [, ov_2, ...])
ov_0 = nonBooleanRV(ov_1)	NOT (ov_0 = nonBooleanRV(ov_1))
ov_0 = nonBooleanRV(ov_1 [, ov_2, ...])	
CONST = nonBooleanRV(ov_1 [, ov_2 , ...])	
nonBooleanRV(ov_1 [, ov_2 , ...]) = CONST	
nonBooleanRV(ov_1) = ov_0	NOT (nonBooleanRV (ov_1) = ov_0)
nonBooleanRV(ov_1 [, ov_2 , ...]) = ov_0	

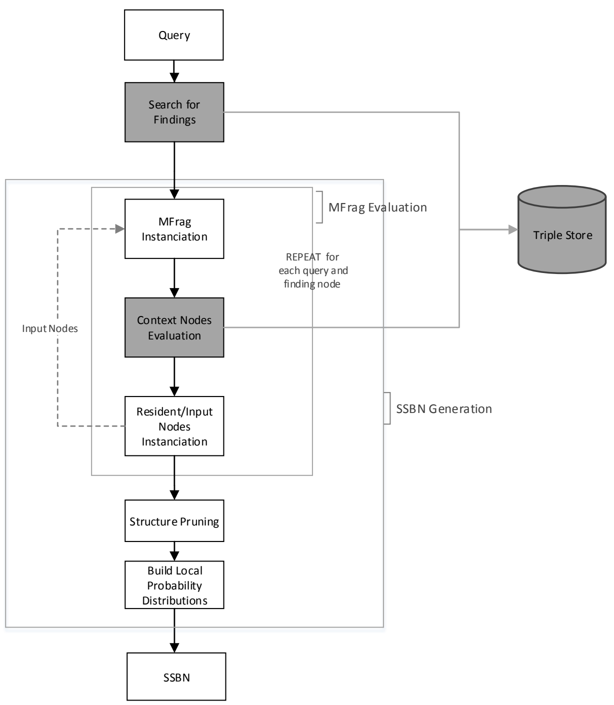

Fig. 3. Generation of SSBN from a user's query

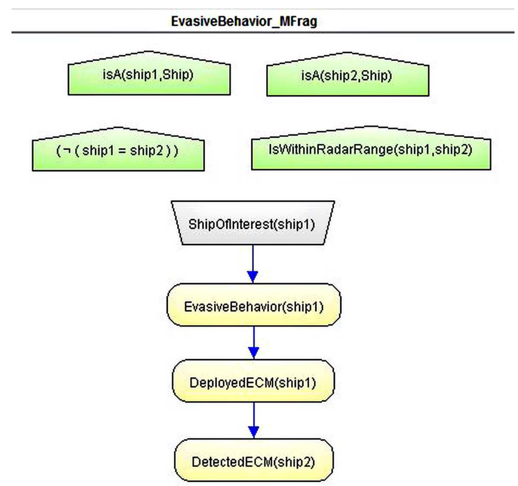

Fig. 4. EvasiveBehavior MFrag

The SSBN construction algorithm implemented in UnBBayes will be changed to search for evidence (findings) and to evaluate the context nodes using the triplestore, as illustrated in Figure 3. The gray elements are the parts of the algorithm that will be modified for using the triplestore. The process starts with a query. Then, the knowledge base (in this case, in a triplestore) is searched for evidence. For each query node and each finding node, there will be an MFrag evaluation. In it, the MFrag will be instantiated and the context node formulas will be evaluated, followed by instantiation of resident and input nodes with the ordinary variables filled with the values of entities that satisfy the context nodes. The input nodes cause the evaluation of MFrags in which they are resident nodes. The SSBN algorithm implemented in UnBBayes is based on the algorithm proposed in [8], where the interested reader can find more details.

The search for evidence and evaluation of context nodes is performed using queries in SPARQL. The code below shows the SPARQL query for evaluating context nodes in the `EvasiveBehavior` MFrag (Figure 4), from the MDA ontology, in a situation where we know that the ordinary variable `ship1` is filled with the entity `ShipA`. The evaluation in this case is made using the comand SELECT.

```
SELECT ?ship2
WHERE  ?ship2 rdf:type Ship           ,
       ShipA isWithinRadarRange ?ship2,
       FILTER(ShipA != ?ship2)}
```

When we know the values of the variables and only want to evaluate a formula, the command ASK is used. The code below shows the evaluation of the same context node of the previous example when we know that ship2 is `ShipA`. This query will return `TRUE` or `FALSE`.

```
ASK
WHERE  ShipB rdf:type Ship    ,
       ShipA isWithinRadarRange ShipB,
       FILTER(ShipA != ShipB)}
```

Since inference in most triplestores is performed by materialization at loading time (the approach used by GraphDB), queries will execute quickly. However, queries correspond to searches of the database, and for this reason can not require evaluation of complex logical expressions. The code showed for the context nodes of the `EvasiveBehavior` MFrag illustrate how an AND and a NOT EQUAL expression can be evaluated, utilizing, respectively, the SPARQL comma operator and the "!=" (different of) operator inside a FILTER. Similarly, the OR can be evaluated using the command UNION. The BNF grammar below shows the formats of context nodes evaluable using these rules. These constructions make it possible to work with the probabilistic ontology for MDA presented in [4].

Listing 1. BNF Grammar for FOL Expressions

```
<atom>::= ov1 == ov2 |
     booleanRV(ov1, [,ov2 ...])  |
     nonBooleanRV(ov1, [,ov2 ...])  = ov0 |
     ov0 = nonBooleanRV(ov1, [,ov2 ...]) |
     nonBooleanRV(ov1, [,ov2 ...]) = CONST |
     CONST = nonBooleanRV(ov1, [,ov2 ...])
<negation>::= NOT <atom>
<conjunction>::= <atom> [AND <atom>]+
<disjunction>::= <atom> [OR <atom>]+
<formula>:= <atom> |
    <negation> |
    <conjunction> |
    <disjunction>
```

Table III compares the two versions of PR-OWL with the characteristics of the proposed PR-OWL 2 RL language.

The first version of PR-OWL was written in OWL DL[8]. PR-OWL 2 was developed in OWL 2 DL, and made several compromises in order to stay within OWL 2 DL. PR-OWL 2 RL will be written using the OWL 2 RL profile, based on RDF semantic.

The expressiveness of context nodes in both versions of PR-OWL is full first-order logic. Nevertheless, their respective implementations have to provide some mechanism to actually evaluate them, since the common reasoners developed for OWL do not deal with FOL. The formats of valid formulas in the context nodes in PR-OWL 2 RL will have restrictions to suit the OWL 2 RL profile. The gain with this approach is that the language will be easily implemented using full implementations already available in current OWL 2 RL technologies.

The first version of PR-OWL only allows probabilistic reasoning, while PR-OWL 2 allows hybrid reasoning, involving probabilistic and deterministic reasoning. This feature will be maintained in PR-OWL 2 RL, through the use of properties such as `definesUncertaintyOf`, `isSubjectIn`, and `isObjectIn`. The use of built-in OWL data types and the support for polymorphism will also remain in PR-OWL 2 RL.

Table IV compares the implementations available for PR-OWL and PR-OWL 2 in the UnBBayes framework with the proposed implementation of PR-OWL 2 RL.

In the PR-OWL implementation, the KIF representation language used in PowerLoom was used in addition to OWL DL, since the assertive database must be expressed in KIF format, separate from the original OWL ontology (represented in OWL DL). The OWL DL language is used to express the terminological database, whereas the assertive database uses the KIF representation language because this is the language used by PowerLoom. The PR-OWL language has constructs to store the evidence using the OWL format, but these are not used in the implementation, making it difficult to work in real domains, expressed in standardized formats of the Semantic Web. In PR-OWL 2 this limitation has been removed: both the TBox and the ABox are stored using OWL. PR-OWL 2 RL keeps this feature, storing the ABox in the triplestore using OWL, serialized as RDF.

The context nodes in the implementation of PR-OWL accept formulas in first-order logic, evaluated by PowerLoom, with restrictions on the evaluation of formulas with variables not filled at evaluation time. In the PR-OWL 2 implementation only formulas in the formats presented in Table II are allowed. The formats of valid formulas in PR-OWL 2 RL are based in OWL 2 RL/RDF and how queries are made using SPARQL.

Of the new characteristics proposed in PR-OWL 2, one that was not fully implemented in UnBBayes was polymorphism, a feature that will be implemented in PR-OWL 2 RL.

The major improvement provided by PR-OWL 2 RL is with respect to scalability, since it is designed to work with millions of RDF triples. The current triplestores available allow work with billions of RDF triples; however, the software that we will use in the prototype, the Lite version of GraphDB, loads the database in memory, which limits the size of the database. The user can work with billions of triples as long as he has a commercial version of GraphDB instead Lite, since the SAIL interface is used in both versions.

VI. Conclusion

PR-OWL and PR-OWL 2 are probabilistic extensions of OWL that use Multi-Entity Bayesian Network (MEBN) for representation and inference under uncertainty. Both languages were implemented in the UnBBayes framework, but these implementations are not scalable enough to be used in real cases with large assertive databases (*e.g.*, in the Maritime Domain Awareness probabilistic ontology developed for the PROGNOS project when used with databases of realistic size). This paper presented an approach to solve this problem, by integrating PR-OWL 2 with RDF triplestores, using the OWL 2 RL profile, a restrictive version of OWL that has polynomial time complexity for most types of reasoning tasks. Future work involves defining the valid constructs for the context node, implementing a plug-in tool for the formalism in UnBBayes, using the GraphDB triplestore, and making more scalability tests with the MDA ontology.

References

[1] J. Esteban, A. Starr, R. Willetts, P. Hannah, and P. Bryanston-Cross, "A review of data fusion models and architectures: towards engineering guidelines," *Neural Computing & Applications*, vol. 14, no. 4, pp. 273–281, 2005.

[8] PR-OWL was proposed when OWL 1 was still the standard for creating ontologies

TABLE III
COMPARATIVE AMONG THE VERSIONS OF PR-OWL

	PR-OWL	PR-OWL 2	PR-OWL 2 RL
Representation Language	OWL DL	OWL 2 DL	OWL 2 RL
Context Nodes	First Order Logic	First Order Logic	Restrict
Hybrid reasoning	No	Yes	Yes
Uncertainty in OWL Properties	No	Yes	Yes
Use of OWL's primitive types	No	Yes	Yes
Polimorphism support	No	Yes	Yes

TABLE IV
COMPARATIVE AMONG THE IMPLEMENTATIONS OF PR-OWL IN UNBBAYES

	PR-OWL	PR-OWL 2	PR-OWL 2 RL
Representation Language	OWL DL/KIF	OWL 2 DL	OWL 2 RL
Reasoner type	KR&R System	DL *Reasoner*	*Triplestore*
Reasoner	PowerLoom	HermiT	GraphDB
Context Nodes	First Order Logic	Restrict - Ad Hoc	Restrict
Hybrid reasoning	No	Yes	Yes
Uncertainty in OWL Properties	No	Yes	Yes
Use of OWL's primitive types	No	Yes	Yes
Polimorphism support	No	No	Yes
Scalability	Thousands	Thousands	Millions

[2] D. Smith and S. Singh, "Approaches to multisensor data fusion in target tracking: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 18, no. 12, pp. 1696–1710, 2006. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109/TKDE.2006.183

[3] P. C. G. Costa, R. N. Carvalho, K. B. Laskey, and C. Y. Park, "Evaluating uncertainty representation and reasoning in HLF systems," in *Information Fusion (FUSION), 2011 Proceedings of the 14th International Conference on*. IEEE, 2011, pp. 1–8.

[4] R. N. Carvalho, P. C. G. da Costa, K. B. Laskey, and K. Chang, "PROGNOS: predictive situational awareness with probabilistic ontologies," in *13th Conference on Information Fusion, FUSION 2010, Edinburgh, UK, July 26-29, 2010*, 2010, pp. 1–8. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5711970

[5] R. N. Carvalho, R. Haberlin, P. C. G. da Costa, K. B. Laskey, and K. Chang, "Modeling a probabilistic ontology for maritime domain awareness," in *Proceedings of the 14th International Conference on Information Fusion, FUSION 2011, Chicago, Illinois, USA, July 5-8, 2011*, 2011, pp. 1–8. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5977514

[6] P. C. G. da Costa, K. B. Laskey, and K. J. Laskey, "PR-OWL: A bayesian ontology language for the semantic web," in *International Semantic Web Conference, ISWC 2005, Galway, Ireland, Workshop 3: Uncertainty Reasoning for the Semantic Web, 7 November 2005*, 2005, pp. 23–33.

[7] S. Matsumoto, R. N. Carvalho, M. Ladeira, P. C. G. Costa, L. L. Santos, D. Silva, M. Onishi, E. Machado, and K. Cai, "UnBBayes: a Java framework for probabilistic models in AI," *Java in Academia and Research. iConcept Press. http://unbbayes. sourceforge. net*, 2011.

[8] K. B. Laskey, "MEBN: a language for First-Order Bayesian knowledge bases," *Artificial Intelligence*, vol. 172, no. 2-3, pp. 140–178, 2008. [Online]. Available: http://portal.acm.org/citation.cfm?id=1327646

[9] S. M. Mahoney and K. B. Laskey, "Constructing Situation Specific Belief Networks," in *UAI '98: Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, University of Wisconsin Business School, Madison, Wisconsin, USA, July 24-26, 1998*, 1998, pp. 370–37. [Online]. Available: http://uai.sis.pitt.edu/displayArticleDetails.jsp?mmnu=1&smnu=2&article_id=269&proceeding_id=14

[10] R. N. Carvalho, K. B. Laskey, and P. C. G. Costa, "PR-OWL 2.0–bridging the gap to OWL semantics," in *Uncertainty Reasoning for the Semantic Web II*. Springer, 2013, pp. 1–18.

[11] R. N. Carvalho, "Probabilistic ontology: Representation and modeling methodology," PhD, George Mason University, Fairfax, VA, USA, 2011.

[12] R. N. Carvalho, M. Ladeira, L. L. Santos, S. Matsumoto, and P. C. G. Costa, "A GUI tool for plausible reasoning in the semantic web using MEBN," in *Innovative Applications in Data Mining*, 2009, pp. 17–45. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-88045-5_2

[13] P. C. G. Costa, M. Ladeira, R. N. Carvalho, K. B. Laskey, L. L. Santos, and S. Matsumoto, "A first-order Bayesian tool for probabilistic ontologies." in *FLAIRS Conference*, D. Wilson and H. C. Lane, Eds. AAAI Press, 2008, pp. 631–636. [Online]. Available: http://dblp.uni-trier.de/db/conf/flairs/flairs2008.html#CostaLCLSM08

[14] S. Matsumoto, "Framework based in plug-ins for reasoning with probabilistic ontologies," M.Sc., University of Brasília, Brasilia, Brazil, 2011.

[15] R. Shearer, B. Motik, and I. Horrocks, "HermiT: A highly-efficient OWL reasoner." in *OWLED*, ser. CEUR Workshop Proceedings, C. Dolbear, A. Ruttenberg, and U. Sattler, Eds., vol. 432. CEUR-WS.org, 2008.

[16] P. Hitzler, M. Krtzsch, and S. Rudolph, *Foundations of Semantic Web Technologies*, 1st ed. Chapman and Hall/CRC Press, 2010.

[17] W3C, "OWL 2 Web Ontology Language document overview," http://www.w3.org/TR/owl2-overview/, 2012, acesso em 01/02/2015. [Online]. Available: http://www.w3.org/TR/owl2-overview/

[18] ——, "OWL 2 Web Ontology Language profiles," http://www.w3.org/TR/owl2-profiles/, 2012, acesso em 01/02/2015. [Online]. Available: http://www.w3.org/TR/owl2-profiles/

[19] H. ter Horst, "Completeness, decidability and complexity of entailment for RDF schema and a semantic extension involving the OWL vocabulary," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 3, no. 2-3, pp. 79–115, 2005.

[20] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, M. Dean *et al.*, "Swrl: A semantic web rule language combining owl and ruleml," *W3C Member submission*, vol. 21, p. 79, 2004.

[21] Y. Zhou, B. C. Grau, I. Horrocks, Z. Wu, and J. Banerjee, "Making the most of your triple store: query answering in OWL 2 using an RL reasoner." in *WWW*, D. Schwabe, V. A. F. Almeida, H. Glaser, R. A. Baeza-Yates, and S. B. Moon, Eds. International World Wide Web Conferences Steering Committee / ACM, 2013, pp. 1569–1580. [Online]. Available: http://dblp.uni-trier.de/db/conf/www/www2013.html\#ZhouGHWB13

[22] B. Bishop, A. Kiryakov, D. Ognyanoff, I. Peikov, Z. Tashev, and R. Velkov, "OWLIM: A family of scalable semantic repositories." *Semantic Web*, vol. 2, no. 1, pp. 33–42, 2011. [Online]. Available: http://dblp.uni-trier.de/db/journals/semweb/semweb2.html#BishopKOPTV11

[23] Y. Guo, Z. Pan, and J. Heflin, "LUBM: A benchmark for OWL knowledge base systems," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 3, no. 2, pp. 158–182, 2005.

[24] B. Bishop and S. Bojanov, "Implementing OWL 2 RL and OWL 2 QL rule-sets for OWLIM." in *OWLED*, ser. CEUR Workshop Proceedings, M. Dumontier and M. Courtot, Eds., vol. 796. CEUR-WS.org, 2011. [Online]. Available: http://dblp.uni-trier.de/db/conf/owled/owled2011.html#BishopB11