Title: Web Ontology Language (OWL)

Name: Kunal Sengupta[1], Pascal Hitzler[1]

Affil./Addr.: Wright State University, Data Semantics Laboratory

465 Joshi Research Center, 3640 Colonel Glenn Highway, Dayton, OH 45435, U.S.A

Phone: (937)775-5217

E-mail: kunal.nes@gmail.com, pascal.hitzler@wright.edu

Web Ontology Language (OWL)

Synonyms

Knowledge Representation Language

Description Logics $\mathcal{SROIQ}$

Semantic Web Modelling Language

Glossary

KR: Knowledge Representation.

Ontology: A set of facts and axioms using a KR language.

OWL: Web Ontology Language.

Inference: Derived knowledge from an ontology.

Expressivity: The level of detail to which data can be modeled.

Definition

Web Ontology Language (OWL) is a core world wide web consortium [W3C] standard knowledge representation language for the Semantic Web. The term knowledge representation, in general, refers to the method of modeling the knowledge about real world entities and relations. OWL is a highly expressive, flexible and efficient knowledge representation language, that can be used to model background knowledge about domains e.g. Health-care, Social Network, Automobiles. OWL is derived from a well-known family of logics called description logics (DLs) [Baader, 2007], and therefore offers a well-defined semantics to the language. A key advantage of using OWL and other logic based modeling languages is that these languages support reasoning services. Reasoning is a method of processing (explicit) background knowledge and infer (implicit) information. E.g. consider the statements, "every Father is a Male" and "Alex is a Father", then an OWL reasoner can reason with this knowledge and infer that "Alex is a Male". OWL semantics supports the open world assumption (OWA), that follows the notion that knowledge about the world at any point of time is incomplete, in other words things that are not known to be true are not necessarily false. OWA is suitable for applications on the world wide web setting where the information is ever increasing. The current version of OWL called OWL 2, became a W3C standard in 2009, it is more expressive than its predecessor OWL 1 (2004). [Motik, 2009] provides more information about the differences between OWL 1 and OWL 2 versions. The following section presents an overview of the OWL 2 syntax.

Introduction

In this Chapter, we provide an overview of OWL including the the syntax, semantics and a brief coverage on the notion of reasoning and how formal logic based languages like OWL can be useful to perform automated reasoning. The syntax of a language is described using a Grammar which provides rules using which legal sentences in the language can be formed. The semantics of a language describes how the statements in the language should interpreted in an unambiguous manner. We also provide an overview of the profiles supported by OWL, these profiles mainly differ in their expressive power and the efficiency of performing reasoning tasks.

Key Points

The key points covered in this Chapter are as follows:

- Basic OWL constructs including classes and properties
- Complex class expressions supported in OWL
- Expressing property relations in OWL
- Reasoning tasks supported in OWL
- OWL 2 has four major profiles that vary in expressiveness and scalability

Historical Background

In the 1990s, many researchers were exploring the idea of using knowledge representation languages used in artificial intelligence to be used as an ontological language for the web. Many of these languages evolved from frame-based languages. In the early 2000s, a language based on description logics, called DAML + OIL [Horrocks, 2002], was chosen to be developed into the web ontology language. In 2004, the first version of OWL was released (OWL 1), and in 2009 the more expressive version OWL 2 was released which became a W3C recommendation.

OWL 2 overview

This section provides an insight to the reader about the OWL 2 syntax and to some extent covers the various constructs available in OWL 2. This by no means is an exhaustive coverage of OWL syntax and we advise the reader to look at [Motik, 2009, Hitzler, 2010] to obtain further details. It should be noted that there are many syntaxes available for OWL 2 and tools that translate from one syntax to another. RDF/XML is the most common and recommended syntax for OWL 2 documents, which will be used througout this article to demonstrate some of the constructs.

Basic constructs

The most fundamental element of an OWL 2 ontology is an IRI (International resource identifier), each real world entity is represented by an IRI. Most often IRIs are quiet long, RDF/XML syntax (and other syntaxes) provides a method to abbreviate the IRIs in the beginning of the document such that the abbreviation can be used to represent the entities in the rest of the document for convenient authoring and ease of readability. Classes, properties (or roles), individuals and datatypes are the basic building blocks of OWL 2. Classes represent the conceptual entities in a domain, e.g. Author, Paper, Contributor, etc. Instances of classes are called individuals e.g. Mark is an Individual that belongs to the class Author. Properties are binary relations, there are two kinds of OWL 2 properties – OWLDataProperty, that represents relationships between (individual, datatype) pairs e.g. hasName, hasTitle, and OWLObjectProperty, that represents relationships between (individual, individual) pairs e.g. hasAuthor is a relationship between instances of Paper and instances of Author. Below are some examples of basic constructs in OWL 2.

```
<owl:Class    rdf:about="Author" />                        (1)
<owl:Class    rdf:about="Paper" />                         (2)
<owl:ObjectProperty      rdf:about="hasAuthor" />     (3)
<owl:DatatypeProperty    rdf:about="hasName" />       (4)
<Author       rdf:about="Bob"/>                            (5)
<rdf:Description rdf:about="Bob">
```

```
        <rdf:type   rdf:resource="Author" />          (6)
    </rdf:Description>
```

Axioms (1) and (2) state that `Author` and `Paper` are concepts represented by OWL 2 Class. Whereas, axiom (3) and (4) state that `hasAuthor` is an object property and `hasName` is a data property. Axiom (5) and (6) are two different ways to assert that the individual `Bob` is an instance of class `Author`. Note that in RDF/XML serailization there are several ways to write an axiom, in this document we will choose the shortest possible form.

Class relations and constructors

For many domains taxonomy is an essential modeling requirement. A taxonomical hierarchy can be generated by modeling simple relations between classes using subclass, equivalent class, disjoint class axioms, and between properties using sub property, equivalent property and disjoint property axioms. These constructs help in modeling statements like (1) every `Author` is a `Contributor` (i.e `Author` is subclass of `Contributor`), (2) every `Author` is a `Writer` and vice versa (i.e `Author` and `Writer` are equivalent classes), and (3) an `Author` is not a `Subscriber` and vice versa (i.e. `Author` and `Subscriber` are disjoint classes). Some predefined classes are offered in OWL 2 which are useful to define hierarchies. `<owl:Thing>` is defined as the topmost class, i.e. all classes are subclasses of this class. Its complementary class `<owl: Nothing` is the bottom class, i.e. it is the subclass of all classes. Likewise, top and bottom properties are defined for both object and data properties.

OWL 2 constructs are not limited to defining taxonomies, constructs known as *complex class expressions* that are very useful to expressively describe classes in terms of a combination of other classes having certain properties. Following are the basic OWL 2 complex class constructors, C, D, and E should be read as class names:

- <owl:intersectionOf> (conjunction) – Used to define a class C that is the intersection of two other classes D and E. Semantically, members of class C are members of both the classes D and E.
- <owl:unionOf> (disjunction) – Used to define a class C in terms of two other classes D and E such that, it contains all the members that belong to either class D or class E or both.
- <owl:complementOf> (negation) – Used to define a class C in terms of another class D such that, members of class C are not the members of class D.

Even more complex class expressions can be formulated by using the *Property Restrictions* in conjunction with the above constructors. As the name suggests these restrictions are used to impose constraints on the property that a class may have. Property restrictions are useful in expressing statements of the form "every Paper should have at least one Author." OWL 2 provides the following property restrictions:

- <owl:allValuesFrom>
- <owl:someValuesFrom>
- <owl:maxQualifiedCardinality>
- <owl:minQualifiedCardinality>
- <owl:qualifiedCardinality>

The example below is used to demonstrate the usage of <owl:someValuesFrom> property restriction. In this example, a paper is defined to be a subclass of an anonymous class which is defined using property restriction (enclosed between <owl:Restriction> and </owl:Restriction>). It is the OWL 2 syntax representation of the sentence "every paper should have at least one author". Other property restrictions can be used using similar syntax.

```
<owl:Class rdf:about="Paper">
```

```
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource = "hasAuthor" />
            <owl:someValuesFrom rdf:resource = "Author" />
        </owl:Restriction>
    <rdfs:subClassOf>
</owl:Class>
```

Property relations and characteristics

Properties play an important role in OWL ontologies, OWL 2 vocabulary consists of several terms which can be used to describe relationships between properties. Properties can also be related to one another using the terms `<rdfs:subPropertyOf>`, `<rdfs:equivalentProperty>`, `<rdf:inverseOf>`. Semantically, if property $p1$ is subproperty of $p2$, then all pairs for which $p1$ holds, $p2$ also holds. If $p1$ is equivalent to $p2$ then $p1$ is subproperty of $p2$ and $p2$ is subproperty of $p1$. If property $p1$ is inverse of property $p2$ then if $p1$ holds for a pair (x, y) of individuals then $p2$ holds for (y, x). In OWL roles constructors like conjunction, disjunction and negation are not available, however, OWL 2 provides a construct called property chains (or role chains) which is often useful in modeling complex properties. Using property chains we can model statements of the form "if an individual x has father as individual y and y has brother as individual z, then x has uncle as individual z.

Properties in OWL can be declared to have some characteristics like domain, range, transitive, reflexive, asymmetric and irreflexive.

Reasoning in OWL

As mentioned before, OWL 2 inherently supports reasoning with background knowledge (explicitly described in an ontology) and infer more (implicit) knowledge. An OWL 2 reasoner is a piece of software that performs the reasoning on an OWL ontology using the well defined semantics of OWL 2. Most reasoners support the following reasoning services:

- Ontology satisfiability: The reasoner checks if the input ontology is consistent (free of contradiction), a contradiction occurs when an ontology has two statements that are inconsistent taken together. e.g. author1 is an instance of class Author and author1 is an instance of negation of class Author.
- Instance checking: Given a class C and individual a, check whether a is an instance of class C.
- Class satisfiability: Given a class C, check whether it has any instance. A class is inconsistent if it has no instance.
- Subsumption: Given two classes C and D check whether C is a subclasss of D.
- Classification: Given an ontology, generate all subclass relationships.

OWL 2 DL profiles

OWL 2 provides flexibility to the ontology curators to choose from the profiles (or sub-languages) of OWL 2, on the basis of expressivity and scalability requirements. The more expressive profiles of OWL 2 are theoretically proven to have higher computational complexity than those with lower expressive power. The most expressive OWL 2 profile with guaranteed decidability of reasoning task is the OWL 2 DL profile for which the worst case computational complexity of reasoning tasks fall in the complexity class N2EXP-TIME-complete. Some lesser but sufficiently useful expressive profiles of OWL 2 provide polynomial time computation guarantee for the reasoning tasks. Below is the listing of the OWL 2 profiles:

- OWL 2 EL: The computational complexity of standard inference tasks in OWL 2 EL is polynomial time. It is mainly useful for applications that have large class and property hierarchies and don't require the more complex OWL constructs. SNOMED CT is a medical ontology which comes under the OWL 2 EL profile.
- OWL 2 QL: This profile was mainly designed to support conjunctive query answering on relational database systems. The standard inference tasks in OWL 2 QL also have the worst case computational complexity of polynomial time.
- OWL 2 RL: This profile allows for rule based systems to perform the reasoning in polynomial time. It has been designed for systems that are implemented using rule based engines. It also provides some inter-operability with other rule base KR languages.
- OWL 2 DL: Is the profile with maximum expressivity while retaining computational decidability, soundness and completeness. The formal foundations of OWL 2 DL is based on description logics $\mathcal{SRIOQ}(\mathcal{D})$ [Horrocks, 2006] language.
- OWL 2 FULL: This is the most expressive OWL 2 profile which comprises of all of OWL 2 DL and RDF(S) constructs with no restrictions. However, there is no guarantee that the reasoning process on an ontology written in this profile would terminate.

Key Applications

Of course, the major application of OWL is to build ontologies to formally describe domains. Once an OWL knowledge base is curated then multiple applications can be created to consume the data like query answering systems and semantic search systems. There is a plethora of real world ontologies published through the Linked Open Data initiative. The Earth cube initiative is another example of real world applications of OWL, where scientists from various research groups are building ontologies to represent geosciences data to be consumed to build applications for search and discovery, visualization etc. For more information we refer the reader to [Hitzler, 2010].

Future Directions

Some of the possible future extensions of OWL may include more expressive languages, identifying more expressive fragments of OWL that are tractable, and adding closed world features to OWL. OWL follows the open world assumption which means that knowledge is always assumed to be incomplete and the absence of a fact from the knowledge base does not entail its falsity. However, closed world assumption on the other hand allows us to assume that a fact is false if doesn't exist in the knowledge base. There are many applications where a combination of closed world and open world assumptions would be useful, therefore we think it will be included in one of the future revisions of OWL.

Conclusion

In this article we have introduced OWL 2 starting with the definition of the subject, then a brief overview of OWL 2 syntax, and some coverage on reasoning and OWL 2 profiles. The content of this article is at the introductory level and we suggest the reader to use the Recommended Reading section as a guide to obtain further insight on this topic.

Cross-References

- Description Logics (00108)
- Linked Open Data (00111)
- RDF (00114)
- Reasoning (00115)
- RIF: The Rule Interchange Format (00118)
- Semantic Annotation (00119)

Acknowledgements

This work was supported by the National Science Foundation under award 1017225 "III: Small: TROn—Tractable Reasoning with Ontologies,"

References

[W3C] World Wide Web Consortium – http://www.w3.org/. Accessed, 30 July 2016.

[Baader, 2007] Baader F, Calvanese D, McGuinness D, Nardi D, Patel-Schneider, eds. (2007) The Description Logic Handbook. Cambridge University Press.

[Motik, 2009] Boris Motik, Peter F. Patel-Schneider, Bijan Parsia, eds. (2009) OWL 2 Web Ontology Language: Structural Specification and Functional-Style, http://www.w3.org/TR/2009/REC-owl2-syntax-20091027/. Latest version available at http://www.w3.org/TR/owl2-syntax/. Accessed, 30 July 2016.

[Hitzler, 2010] Hitzler P, Krötzsch M, and Rudolph S (2010) Foundations of Semantic Web Technologies. Chapman & Hall/CRC.

[Horrocks, 2006] Horrocks I, Kutz O, and Sattler U (2006) The Even More Irresistible SROIQ. In Proc. of the 10th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2006). AAAI Press, 2006.

[NSF, 2011] The Earth Cube Initiative, http://earthcube.org/. Accessed, 30 July 2016.

[Horrocks, 2002] Horrocks I (2002) DAML+OIL: a Description Logic for the Semantic Web. IEEE Data Engineering Bulletin 25 (2002), 4–9.

Recommeded Reading

- The main website for OWL – http://www.w3.org/2007/OWL. Accessed, 30 July 2017.
- OWL 2 overview page (with details about differnces between OWL 1 and OWL 2) – http://www.w3.org/TR/2009/WD-owl2-overview-20090327/. Accessed, 30 July 2016.
- The OWL 2 primer – http://www.w3.org/TR/owl2-primer/. Accessed, 30 July 2017.
- More detailed discussion on OWL 2 profiles can be found on the website at the URL http://www.w3.org/TR/owl2-profiles/. Accessed, 30 July 2017.
- [Hitzler, 2010], provides thorough coverage of OWL syntax and semantics.
- [Horrocks, 2006] is the landmark paper on $\mathcal{SROIQ}$, which forms the basis for the OWL 2 language specs.