OWL 2 Update

Christine Golbreich
<cgolbrei@gmail.com>
OWL 2

- **W3C OWL working group** is developing **OWL 2**
  - see http://www.w3.org/2007/OWL/wiki/
  - **Extends OWL** with a small but useful set of features
  - **Fully backwards compatible** with OWL:
    - Every OWL ontology is a valid OWL 2 ontology
    - Every OWL 2 ontology – not using new features – is a valid OWL ontology

- **A community effort** – features included are those
  - That are needed in applications
  - For which semantics and reasoning techniques are well understood
  - That tool builders are willing and able to support

- Already supported by many popular **OWL tools**
  - Protégé, Swoop, TopBraid Composer, FaCT++, Pellet, OWL API
What’s New in OWL 2?

- **OWL 2** is an update to OWL adding several new features
  - Increased expressive power, e.g., w.r.t. properties
  - Extended support for datatypes
  - Simple metamodelling capabilities
  - Extended annotation capabilities
  - Database style keys

- **OWL 2** also defines several profiles
  - language subsets
    - that may better meet certain performance requirements
    - or may be easier to implement
Increased expressive power

- Qualified cardinality restrictions
  - Minimum, Maximum, or Exact - Object or Data Property - Qualified or not
    E.g., Set of objects bound to at most three Hydrogen
    
    \[
    \text{MaxCardinality}( 3 \ boundTo \ \text{Hydrogen})
    \]
    
    \[
    \text{MinCardinality}( 1 \ \text{hasSSN})
    \]

- Property chain inclusion axioms
  - allows to chain several object properties
    E.g., If x is locatedIn y, and y is partOf z, then x is locatedIn z;
    
    \[
    \text{SubPropertyOf}( \text{PropertyChain}( locatedIn \ \text{partOf } ) \ locatedIn )
    \]
  - provides a means to represent some types of rules under certain global restrictions on axioms for decidability
    E.g., the Uncle rule!
    
    \[
    \text{SubPropertyOf}( \text{PropertyChain}
    
    ( \text{hasParent hasBrother} ) \ \text{hasUncle})
    \]
Increased expressive power

• Reflexive, Irreflexive, Asymmetric
  E.g., each one has the same blood group as himself
    \text{ReflexiveProperty}( \text{hasSameBloodGroup} )
  E.g., Nothing can be a proper part of itself
    \text{IrreflexiveProperty}( \text{proper_part_of} )
  E.g., if x is preceded by y, then y cannot be preceded by x
    \text{AsymmetricProperty}( \text{preceded_by} ) [\text{e.g., process}] 

• Local reflexivity
  E.g., Auto-regulating processes regulate themselves
    \text{SubClassOf}( \text{AutoRegulatingProcess} \text{ExistsSelf}( \text{regulate} ) )

• Disjoint properties
  E.g., no individuals can be both homozygous and heterozygous twins
    \text{DisjointProperties}( \text{homozygousTwin} \text{ heterozygousTwin} )
Syntactic sugar

• DisjointUnion
  E.g., a brain hemisphere is either a left or right hemisphere but not both
  \[\text{DisjointUnion}( \text{BrainHemisphere} \ \text{LeftHemisphere} \ \text{RightHemisphere} )\]

• DisjointClasses
  E.g., Middle and upper, middle and lower, upper and lower lungs are exclusive
  \[\text{DisjointClasses}( \text{MiddleLung} \ \text{UpperLung} \ \text{LowerLung} )\]

• NegativePropertyAssertion
  E.g., This patient is not five years old.
  \[\text{NegativePropertyAssertion}( \text{hasAge} \ \text{ThisPatient} \ 5^{\text{xsd:integer}} )\]
Extended datatypes

- A richer set of datatypes for representing
  - various kinds of **numbers**, adding support of a wider range of XML Schema Datatypes
    E.g.; integer, real, double, float, decimal, …
  - **strings** with a **Language Tag** (or without)
    E.g.; the class with ID 0000003 has label ‘anatomical structure’ in English
    \[\text{EntityAnnotation(Class(CARO:0000003) Label("anatomical structure"@en))}\]
  - **Boolean values, Binary Data, URIs, Time Instants**, etc.

- Datatype restriction
  - **User-defined datatypes** using **facets** from XML Schema Datatypes for range
    E.g.; Individuals that are more than 18
    \[\text{DatatypeRestriction(xsd:integer minInclusive "18"^^xsd:integer)}\]
Simple metamodelling

- Based on **punning**
  - The same name can refer to different types of entities, with certain restrictions
    - E.g., both individual and:
      - class | datatype | object property | data property | annotation property
  - Punning forbidden for
    - ObjectProperty ↔ DatatypeProperty    Class ↔ Datatype

| Declaration( Class( Deprecated_Properties ) ) | Declares `Deprecated_Properties` to be a Class |
| Declaration( ObjectProperty( located_in ) ) | Declares `located_in` to be an ObjectProperty |
| ClassAssertion( Deprecated_Properties located_in ) | states that `located_in` is an Individual of the class `Deprecated_Properties`. |
Extended annotations

- Annotations of axioms as well as entities
  
  E.g., `SubClassOf( Comment("Middle lobe are necessary right lobe.") MiddleLobe RightLobe)`

- Even annotations of annotations

- Value of an annotation can be either
  - a literal (e.g., string, integer, or any other OWL datatype)
    
    E.g. `EntityAnnotation (Class(CARO: anatomical structure) hasId( "0000003"^^xsd:integer ))`
  
  - an ontology entity (such as a class or individual)
  
  - an anonymous individual
Keys

- OWL 2 allows to define **Database style keys** for a given class
- A **HasKey** axiom states that each (named) instance of a class is uniquely identified by a property or a set of properties
  - if two (named) instances coincide on all the values of key properties, then these two individuals are the same.

  E.g., Each person is uniquely identified by his social security number.

  HasKey( Person hasSSN )
Profiles (Tractable Fragments)

- Profile is a subset of vocabulary (fragment)
- OWL 1 defines only one fragment (OWL Lite)
  - And it isn’t very tractable!
- OWL 2 defines several different fragments with
  - Useful *computational* properties
    - E.g., reasoning complexity in range LOGSPACE to PTIME
  - Useful *implementation* possibilities
    - E.g., Smaller fragments implementable using RDBs
- OWL 2 profiles
  - OWL 2 EL, OWL 2 QL, OWL 2 RL
OWL 2 EL

• Useful for applications employing ontologies that contain very large number of properties and/or classes

• Captures expressive power used by many large-scale ontologies E.g.; SNOMED CT, NCI thesaurus

• Features
  – Included: existential restrictions, intersection, subClass, equivalentClass, disjointness, range and domain, object property inclusion possibly involving property chains, and data property inclusion, transitive properties, keys …
  – Missing: include value restrictions, Cardinality restrictions (min, max and exact), disjunction and negation

• Maximal language for which reasoning (including query answering) known to be worst-case polynomial
OWL 2 QL

• Useful for applications that use **very large volumes of data**, and where query answering is the most important task.

• Captures expressive power of simple ontologies like **thesauri**, **classifications**, and (most of) expressive power of ER/UML schemas. E.g., **CIM10, Thesaurus of Nephrology, ...**

• Features
  - **Included**: limited form of existential restrictions, subClass, equivalentClass, disjointness, range & domain, symmetric properties, ...
  - **Missing**: existential quantification to a class, self restriction, nominals, universal quantification to a class, disjunction etc.

• Can be implemented on top of **standard** relational **DBMS**

• Maximal language for which reasoning (including query answering) is known to be worst case **logspace** (same as DB)
OWL 2 RL

- Useful for applications that require **scalable reasoning without sacrificing too much expressive power**, and where query answering is the most important task
- **Support most OWL features but**
  - with restrictions placed on the syntax of OWL 2
  - standard semantics only apply when they are used in a restricted way
- Can be implemented on top of **rule** extended DBMS
  - E.g., Oracle’s OWL Prime implemented using forward chaining rules in Oracle 11g
  - Related to DLP [DLP] and pD* [pD*]
- Allows for scalable (**polynomial**) reasoning using rule-based technologies
OWL 2 Public Working Drafts

• Seven OWL 2 Drafts Published (2008-10-08)

W3C News   [http://www.w3.org/](http://www.w3.org/)

2008-10-08: The OWL Working Group published seven documents relating to the OWL 2 Web Ontology Language:

... 

1. Structural Specification and Functional-Style Syntax
2. Direct Semantics
3. RDF-Based Semantics (First Public Draft)
4. Mapping to RDF Graphs
5. XML Serialization
6. Profiles
7. Conformance and Test Cases (First Public Draft)
OWL 2 Public Working Drafts

- Seven OWL 2 Drafts Published (2008-10-08)

  - First three documents form the technical **core of OWL 2** specifying its
    1. **Syntax**: both the structure of the language and its functional-style syntax
    2. **Semantics**: both a traditional "direct" and a new "RDF-based" semantics
  - Documents 4 & 5 specify two different serializations for OWL ontologies
    * one based on a **Mapping to RDF** and one using **XML** more directly
  - Document 6 defines the **Profiles**
  - Document 7 specifies **Conformance** and will later enumerate **Test cases**
  - Five other documents are under development
Thank you for listening

Thanks to Ian Horrocks (slides) & OWL WG (work)

OWL 2 Public Working Drafts on Wiki

http://www.w3.org/2007/OWL/wiki/OWL_Working_Group#Deliverables
Any questions?