Ontologies and the Semantic Web

Ian Horrocks
<horrocks@cs.man.ac.uk>
Information Management Group
School of Computer Science
University of Manchester
The Semantic Web
Today’s Web

- Distributed hypertext/hypermedia
- Information accessed via (keyword based) search and browse
- Browser tools render information for human consumption
What is the Semantic Web?

- Web was “invented” by Tim Berners-Lee (amongst others), a physicist working at CERN
- His vision of the Web was much more ambitious than the reality of the existing (syntactic) Web:

  “… a set of connected applications … forming a consistent logical web of data …”

  “… an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation …”

- This vision of the Web has become known as the Semantic Web
Hard Work using “Syntactic Web”

Find images of Peter Patel-Schneider, Frank van Harmelen and Alan Rector…

Impossible (?) using “Syntactic Web”

• Complex queries involving *background knowledge*
  – Find information about “animals that use sonar but are neither bats nor dolphins”, *e.g.*, *Barn Owl*

• Locating information in *data repositories*
  – Travel enquiries
  – Prices of goods and services
  – Results of human genome experiments

• Finding and using “*web services*”
  – Given a DNA sequence, identify its genes, determine the proteins they can produce, and the biological processes they control

• Delegating complex tasks to web *agents*
  – Book me a holiday next weekend somewhere warm, not too far away, and where they speak either French or English
What is the Problem?

Consider a typical web page:

- Markup consists of:
  - rendering information (e.g., font size and colour)
  - Hyper-links to related content
- Semantic content is accessible to humans, but not (easily) to computers…
What is the (Proposed) Solution?

• Add semantic annotations to web resources

Dr. Alan Rector, Professor of Computer Science, University of Manchester

Giving Semantics to Annotations

- External agreement on meaning of annotations
  - Agree on meaning of a set of annotation tags
    - E.g., Dublin Core
    - Limited flexibility and extensibility
    - Limited number of things can be expressed
  - Use Ontologies to specify meaning of annotations
    - Agree on language used to describe meaning
    - Meanings of vocabularies of terms given by ontologies
      - New terms can be formed by combining existing ones
      - Meaning of such terms is formally specified
      - Can combine/relate terms in multiple ontologies
Ontologies
Ontology: Origins and History

- In Philosophy, fundamental branch of metaphysics
  - Studies “being” or “existence” and their **basic categories**
  - Aims to find out what **entities** and **types of entities** exist
Ontology in Information Science

• An ontology is an engineering artefact consisting of:
  – A **vocabulary** used to describe (a particular view of) some domain
  – An **explicit specification** of the **intended meaning** of the vocabulary.
    • Often includes classification based information
  – Constraints capturing **background knowledge** about the domain

• Ideally, an ontology should:
  – Capture a **shared understanding** of a domain of interest
  – Provide a **formal** and **machine manipulable** model
Example Ontology (Protégé)
Applications of Ontologies

- **e-Science**, e.g., Bioinformatics
  - Open Biomedical Ontologies Consortium (GO, MGED)
  - Used e.g., for “in silico” investigations relating theory and data
    - E.g., relating data on phosphatases to (model of) biological knowledge
Applications of Ontologies

- Medicine
  - Building/maintaining terminologies such as Snomed, NCI & Galen
Applications of Ontologies

• Organising complex and semi-structured information
  – UN-FAO, NASA, Ordnance Survey, General Motors, Lockheed Martin, …
Applications of Ontologies

• **Military/Government**
  - DARPA, NIST, SAIC, Department of Homeland Security, …

• The **Semantic Web** and so-called **Semantic Grid**
Ontology Languages
Ontology Languages for the Web

• Semantic Web effort led to development of “resource description” language(s)
  – E.g., RDF, and later RDF Schema (RDFS)

• RDFS is recognisable as an ontology language
  – Classes and properties
  – Sub/super-classes (and properties)
  – Range and domain (of properties)

• But RDFS too weak to describe resources in sufficient detail, e.g.:
  – No existence/cardinality constraints
  – No transitive, inverse or symmetrical properties
  – No localised range and domain constraints
  – ...

• And RDF(S) has “higher order flavour” with non-standard semantics
  – Difficult to provide reasoning support
From RDFS to OWL

• Two languages developed to address deficiencies & problems of RDFS:
  – **OIL**: developed by group of (largely) European researchers
  – **DAML-ONT**: developed by group of (largely) US researchers

• Efforts merged to produce **DAML+OIL**
  – Development carried out by “Joint EU/US Committee on Agent Markup Languages”

• DAML+OIL submitted to **W3C** as basis for standardisation
  – Web-Ontology (**WebOnt**) Working Group formed
  – WebOnt developed **OWL** language based on DAML+OIL
  – OWL now a W3C **recommendation** (i.e., a standard)

• **OIL**, DAML+OIL and OWL based on **Description Logics**
  – OWL is effectively a “Web-friendly” syntax for **SHOIN**
What Are Description Logics?

• A family of logic based Knowledge Representation formalisms
  – Descendants of semantic networks and KL-ONE
  – Describe domain in terms of concepts (classes), roles (properties, relationships) and individuals
  – Operators allow for composition of complex concepts
  – Names can be given to complex concepts, e.g.:

\[ \text{HappyParent} \equiv \text{Parent} \sqcap \neg \text{hasChild.}(\text{Intelligent} \sqcap \text{Athletic}) \]
Semantics and Reasoning

• Distinguished by:
  
  – **Formal semantics** (typically model theoretic)
    
    • Decidable fragments of FOL (often contained in $C_2$)
    
    • Closely related to Propositional Modal & Dynamic Logics, and to Guarded Fragment

[Quillian, 1967]
Semantics and Reasoning

• Distinguished by:
  – **Formal semantics** (typically model theoretic)
    • Decidable fragments of FOL (often contained in $C_2$)
    • Closely related to Propositional Modal & Dynamic Logics, and to Guarded Fragment
  – Provision of **inference services**
    • Decision procedures for key problems (satisfiability, subsumption, etc)
    • Implemented systems (highly optimised)
Why Description Logic?

- OWL exploits results of 15+ years of DL research
  - Well defined (model theoretic) semantics
Why Description Logic?

• OWL exploits results of 15+ years of DL research
  – Well defined (model theoretic) semantics
  – Formal properties well understood (complexity, decidability)

I can’t find an efficient algorithm, but neither can all these famous people.

Why Description Logic?

• OWL exploits results of 15+ years of DL research
  – Well defined (model theoretic) semantics
  – Formal properties well understood (complexity, decidability)
  – Known reasoning algorithms

\[\begin{align*}
\Box\text{-rule} & \quad \text{if } 1. \ (C_1 \cap C_2) \in \mathcal{L}(v), \ v \text{ is not indirectly blocked, and} \\
& \quad 2. \ \{C_1, C_2\} \not\subseteq \mathcal{L}(v) \\
& \quad \text{then } \mathcal{L}(v) \rightarrow \mathcal{L}(v) \cup \{C_1, C_2\}. \\
\sqcup\text{-rule} & \quad \text{if } 1. \ (C_1 \cup C_2) \in \mathcal{L}(v), \ v \text{ is not indirectly blocked, and} \\
& \quad 2. \ \{C_1, C_2\} \cap \mathcal{L}(v) = \emptyset \\
& \quad \text{then } \mathcal{L}(v) = \mathcal{L}(v) \cup \{E\} \text{ for some } E \in \{C_1, C_2\}. \\
\exists\text{-rule} & \quad \text{if } 1. \ \exists r.C \in \mathcal{L}(v_1), \ v_1 \text{ is not blocked, and} \\
& \quad 2. \ c_1 \text{ has no safe } r\text{-neighbour } v_2 \text{ with } C \in \mathcal{L}(v_1) \\
& \quad \text{then } \mathcal{L}(v_2) = \{C\} \text{ and } \mathcal{L}(v_1, v_2) = \{r\}. \\
\forall\text{-rule} & \quad \text{if } 1. \ \forall r.C \in \mathcal{L}(v_1), \ v_1 \text{ is not indirectly blocked, and} \\
& \quad 2. \ \text{there is an } r\text{-neighbour } v_2 \text{ of } v_1 \text{ with } C \not\in \mathcal{L}(v_2) \\
& \quad \text{then } \mathcal{L}(v_2) = \mathcal{L}(v_2) \cup \{C\}. \\
\forall_+\text{-rule} & \quad \text{if } 1. \ \forall r.C \in \mathcal{L}(v_1), \ v_1 \text{ is not indirectly blocked, and} \\
& \quad 2. \ \text{there is some role } r' \text{ with } \text{Trans}(r') \text{ and } r' \subseteq r \\
& \quad 3. \ \text{there is an } r'\text{-neighbour } v_2 \text{ of } v_1 \text{ with } \exists r'.C \not\in \mathcal{L}(v_2) \\
& \quad \text{then } \mathcal{L}(v_2) = \mathcal{L}(v_2) \cup \{\forall r'.C\}. \\
\text{choose-rule} & \quad \text{if } 1. \ \exists n r.C \in \mathcal{L}(v_1), \ v_1 \text{ is not indirectly blocked, and} \\
& \quad 2. \ \text{there is an } r\text{-neighbour } v_2 \text{ of } v_1 \text{ with } \{C, \exists C\} \cap \mathcal{L}(v_2) = \emptyset \\
& \quad \text{then } \mathcal{L}(v_2) = \mathcal{L}(v_2) \cup \{E\} \text{ for some } E \in \{C, \exists C\}. \\
\geq\text{-rule} & \quad \text{if } 1. \ \geq n r.C \in \mathcal{L}(v), \ v \text{ is not blocked, and} \\
& \quad 2. \ \text{there are not } n \text{ safe } r\text{-neighbours } v_1, \ldots, v_n \text{ of } v \\
& \quad \text{with } C \in \mathcal{L}(v) \text{ and } v_i \neq v_j \text{ for } 1 \leq i < j \leq n.
\end{align*}\]
Why Description Logic?

• OWL exploits results of 15+ years of DL research
  – Well defined (model theoretic) **semantics**
  – **Formal properties** well understood (complexity, decidability)
  – Known **reasoning algorithms**
  – **Implemented systems** (highly optimised)
Why Description Logic?

• Foundational research was **crucial** to design of OWL
  
  – Informed Working Group decisions at every stage, e.g.:
    
    • “Why not extend the language with feature \( x \), which is clearly harmless?”

  
  • “Adding \( x \) would lead to undecidability - see proof in […]”
Why the Strange Names?

- Description Logics are a **family** of KR formalisms
  - Mainly distinguished by available operators

- **Available operators** indicated by letters in name, e.g.,
  - $S$: basic DL ($\mathcal{ALC}$) plus transitive roles (e.g., $\text{ancestor} \in R_+$)
  - $H$: role hierarchy (e.g., $\text{hasDaughter} \vee \text{hasChild}$)
  - $O$: nominals/singleton classes (e.g., $\{\text{Italy}\}$)
  - $I$: inverse roles (e.g., $\text{isChildOf} \dashv \text{hasChild}^{-}$)
  - $N$: number restrictions (e.g., $\geq 2 \text{hasChild}$, $6 \geq 3 \text{hasChild}$)

- Basic DL + role hierarchy + nominals + inverse + NR = **SHOIN**
  - **SHOIN** is the basis for W3C’s **OWL** Web Ontology Language

- **SHOIN** is very expressive, but still decidable (just)
# Class/Concept Constructors

<table>
<thead>
<tr>
<th>Constructor</th>
<th>DL Syntax</th>
<th>Example</th>
<th>FOL Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>intersectionOf</td>
<td>$C_1 \cap \ldots \cap C_n$</td>
<td>Human $\cap$ Male</td>
<td>$C_1(x) \land \ldots \land C_n(x)$</td>
</tr>
<tr>
<td>unionOf</td>
<td>$C_1 \cup \ldots \cup C_n$</td>
<td>Doctor $\cup$ Lawyer</td>
<td>$C_1(x) \lor \ldots \lor C_n(x)$</td>
</tr>
<tr>
<td>complementOf</td>
<td>$\neg C$</td>
<td>$\neg$ Male</td>
<td>$\neg C(x)$</td>
</tr>
<tr>
<td>oneOf</td>
<td>${x_1} \sqcup \ldots \sqcup {x_n}$</td>
<td>${john} \sqcup {mary}$</td>
<td>$x = x_1 \lor \ldots \lor x = x_n$</td>
</tr>
<tr>
<td>allValuesFrom</td>
<td>$\forall P.C$</td>
<td>$\forall$ hasChild.Doctor</td>
<td>$\forall y.P(x, y) \rightarrow C(y)$</td>
</tr>
<tr>
<td>someValuesFrom</td>
<td>$\exists P.C$</td>
<td>$\exists$ hasChild.Lawyer</td>
<td>$\exists y.P(x, y) \land C(y)$</td>
</tr>
<tr>
<td>maxCardinality</td>
<td>$\leq n P$</td>
<td>$\leq 1$ hasChild</td>
<td>$\exists y.P(x, y)$</td>
</tr>
<tr>
<td>minCardinality</td>
<td>$\geq n P$</td>
<td>$\geq 2$ hasChild</td>
<td>$\exists y.P(x, y)$</td>
</tr>
</tbody>
</table>

$C$ is a concept (class); $P$ is a role (property); $x$ is an individual name.
Knowledge Base / Ontology

- A **TBox** is a set of “schema” axioms (sentences), e.g.:

  \[
  \{\text{Parent} \lor \text{Person} \land \exists \text{hasChild}, \\
  \text{HappyParent} \land \text{Parent} \land \exists \text{hasChild}.(\text{Intelligent} \land \text{Athletic})\}
  \]

- An **ABox** is a set of “data” axioms (ground facts), e.g.:

  \[
  \{\text{John:HappyParent}, \\
  \text{John hasChild Mary}\}
  \]

- An OWL ontology is just a **SHOIN KB**
E.g., Parent \cup \exists \text{hasChild.(Intelligent} \land \text{Athletic)}:

```xml
<owl:Class>
  <owl:intersectionOf rdf:parseType=" collection">
    <owl:Class rdf:about="#Parent"/>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasChild"/>
      <owl:allValuesFrom>
        <owl:unionOf rdf:parseType=" collection">
          <owl:Class rdf:about="#Intelligent"/>
          <owl:Class rdf:about="#Athletic"/>
        </owl:unionOf>
      </owl:allValuesFrom>
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>
```
Why Ontology Reasoning?

• Given key role of ontologies in many applications, it is essential to provide **tools** and **services** to help users:
  – Design and maintain high quality ontologies, e.g.:
    • **Meaningful** — all named classes can have instances
Why Ontology Reasoning?

- Given key role of ontologies in many applications, it is essential to provide **tools** and **services** to help users:
  - Design and maintain high quality ontologies, e.g.:
    - **Meaningful** — all named classes can have instances
    - **Correct** — captures intuitions of domain experts
Why Ontology Reasoning?

• Given key role of ontologies in many applications, it is essential to provide **tools** and **services** to help users:
  
  – Design and maintain high quality ontologies, e.g.:
    
    • **Meaningful** — all named classes can have instances
    • **Correct** — captures intuitions of domain experts
    • **Minimally redundant** — no unintended synonyms
Why Ontology Reasoning?

• Given key role of ontologies in many applications, it is essential to provide tools and services to help users:
  – Design and maintain high quality ontologies, e.g.:
    • **Meaningful** — all named classes can have instances
    • **Correct** — captures intuitions of domain experts
    • **Minimally redundant** — no unintended synonyms
  – Answer queries over ontology classes and instances, e.g.:
    • Find more general/specific classes
    • Retrieve individuals/tuples matching a given query
Research Challenges
Increasing Expressive Power

- **Complex role inclusion axioms** [Horrocks&Sattler, IJCAI-03]
  - E.g., hasLocation $\pm$ partOf $\lor$ hasLocation

- **Concrete domains/datatypes**, e.g., [Lutz, IJCAI-99; Pan et al, ISWC-03]
  - E.g., value comparison (income > expenditure)

- **Database style keys** [Lutz et al, JAIR 2004]
  - E.g., make + model + chassis-number is a key for Vehicles

- **Rule language extensions**
  - First order extensions (e.g., SWRL) [Horrocks et al, JWS, 2005]
  - Hybrid language extensions, e.g., [Eiter et al, KR-04; Motik et al, ISWC-04]
  - LP/F-Logic/Common Logic [Chen et al, JLP, 1993; de Bruijn et al, WWW-05]
Improving Scalability

- **Optimisation techniques**
  - Improve performance of DL reasoners, e.g., [Sirin et al, KR-06]

- **Reduction to disjunctive Datalog** [Motik et al, KR-04]
  - Transform DL ontology to Datalog\(\subseteq\) rules
  - Use LP techniques to deal with large numbers of ground facts

- **Hybrid DL-DB systems** [Horrocks et al, CADE-05]
  - Use DB to store “Abox” (individual) axioms
  - Cache inferences and use DB queries to answer/scope logical queries

- **Polynomial time algorithms** for sub-\(\text{ALC}\) logics [Baader et al, IJCAI-05]
  - Graph based techniques for subsumption computation
Tools and Infrastructure

- Editors/environments
  - Oiled, Protégé, Swoop, Construct, Ontotrack, …
Tools and Infrastructure

- **Editors/environments**
  - Oiled, Protégé, Swoop, Construct, Ontotrack, …

- **Reasoning systems**
  - Cerebra, FaCT++, Kaon2, Pellet, Racer, …
Tools and Infrastructure

• **Editors/environments**
  – Oiled, Protégé, Swoop, Construct, Ontotrack, …

• **Reasoning systems**
  – Cerebra, FaCT++, Kaon2, Pellet, Racer, …

• **Non-standard inferences**
  – Explanation, matching, least common subsumer, …
Tools and Infrastructure

- **Editors/environments**
  - Oiled, Protégé, Swoop, Construct, Ontotrack, …

- **Reasoning systems**
  - Cerebra, FaCT++, Kaon2, Pellet, Racer, …

- **Non-standard inferences**
  - Explanation, matching, least common subsumer, …

- **Design methodologies**
  - Foundational ontologies, modularisation, etc.
Summary

- **Semantic Web** aims to make web content more accessible to automated processes
  - Adds semantic annotations to web resources
- **Ontologies** provide vocabulary for annotations
  - Terms have well defined meaning
- **OWL** ontology language based on (description) logic
  - Exploits results of basic research on complexity, reasoning, etc.
- Many **research challenges** remain
  - Including expressive power, scalability and tools
Acknowledgements

Thanks to my many friends in the DL and Semantic Web communities, in particular:

– Alan Rector
– Franz Baader
– Uli Sattler
Resources

• FaCT++ system (open source)
  – http://owl.man.ac.uk/factplusplus/

• Protégé
  – http://protege.stanford.edu/plugins/owl/

• W3C Web-Ontology (WebOnt) working group (OWL)
  – http://www.w3.org/2001/sw/WebOnt/

• DL Handbook, Cambridge University Press
  – http://books.cambridge.org/0521781760.htm
Thank you for listening

Any questions?

DL & KR, Windermere, 30th May — 5th June