DAML+OIL: a Reason-able Web Ontology Language

Ian Horrocks
horrocks@cs.man.ac.uk

University of Manchester
Manchester, UK
Talk Outline

The Semantic Web
Web Ontology Languages
DAML+OIL Language
Reasoning with DAML+OIL
OilEd Demo
Description Logic Reasoning
Research Challenges
Summary
The Semantic Web
The Semantic Web Vision

- Web made possible through established **standards**
  - **TCP/IP** for transporting bits down a wire
  - **HTTP & HTML** for transporting and rendering hyperlinked text
- **Applications** able to exploit this common infrastructure
  - Result is the WWW as we know it
- **1st generation** web mostly handwritten HTML pages
- **2nd generation** (current) web often machine generated/active
- Both intended for direct human processing/interaction
- In **next generation** web, **resources** should be more accessible to automated processes
  - To be achieved via **semantic markup**
  - **Metadata** annotations that describe content/function
- Coincides with Tim Berners-Lee’s vision of a **Semantic Web**
Realising the Semantic Web

- Semantic web vision is extremely ambitious
- Even partial realisation will be a major undertaking
- Input will be required from many communities (inc. AI and Database)
- Topics covered at ISWC include:
  - Agents
  - Database technologies
  - Digital libraries
  - e-business
  - e-science and the Grid
  - Integration, mediation and storage
  - Knowledge representation and reasoning
  - Languages and infrastructure
  - Metadata (inc. generation and authoring)
  - Multimedia data
  - Natural language
  - Ontologies
  - Searching and querying
  - Services and service description
  - Trust and meaning
  - User interfaces
  - Visualisation and modelling
  - Web mining
Ontologies

- Semantic markup must be **meaningful** to automated processes
- Ontologies will play a key role
  - Source of **precisely defined** terms (vocabulary)
  - Can be **shared** across applications (and humans)
- Ontology typically consists of:
  - **Hierarchical** description of important **concepts** in domain
  - Descriptions of the **properties** of each concept
- Degree of formality can be quite variable (NL–logic)
- Increased formality and regularity facilitates machine understanding
- Ontologies can be used, e.g.:
  - To facilitate buyer–seller communication in **e-commerce**
  - In semantic based **search**
  - To provide richer **service descriptions** that can be more flexibly interpreted by intelligent agents
Web Ontology Languages
Web Languages

- Web languages already extended to facilitate **content description**
  - XML Schema (XMLS)
  - RDF and RDF Schema (RDFS)
- RDFS recognisable as an **ontology language**
  - Classes and properties
  - Range and domain
  - Sub/super-classes (and properties)
- But RDFS not a suitable foundation for Semantic Web
  - **Too weak** to describe resources in sufficient detail
- Requirements for web ontology language:
  - **Compatible** with existing Web standards (XML, RDF, RDFS)
  - **Easy to understand** and use (based on common KR idioms)
  - **Formally specified** and of “adequate” expressive power
  - Possible to provide **automated reasoning** support
Two languages developed to satisfy above requirements

- **OIL**: developed by group of (largely) European researchers (several from OntoKnowledge project)
- **DAML-ONT**: developed by group of (largely) US researchers (in DARPA DAML programme)

Efforts merged to produce DAML+OIL

- Development was overseen by joint EU/US committee
- Now submitted to W3C as basis for standardisation
- WebOnt working group developing language standard
- New standard may be called **OWL** (Ontology Web Language)
DAML+OIL

- DAML+OIL **layered** on top of RDFS
  - RDFS based **syntax**
  - **Inherits** RDFS ontological primitives (subclass, range, domain)
  - Provides **much** richer set of primitives (equality, cardinality, . . .)

- DAML+OIL designed to describe **structure** of domain (**schema**)
  - **Object oriented**: classes (concepts) and properties (roles)
  - DAML+OIL ontology consists of set of **axioms** asserting characteristics of classes and properties
  - E.g., Person is **kind of** Animal whose parents are Persons

- RDF used for class/property membership assertions (**data**)
  - E.g., John is an **instance of** Person; ⟨John, Mary⟩ is an instance of parent
DAML+OIL equivalent to very expressive Description Logic
  • But don’t tell anyone!

More precisely, DAML+OIL is (extension of) SHIQ DL

DAML+OIL benefits from many years of DL research
  • Well defined semantics
  • Formal properties well understood (complexity, decidability)
  • Known reasoning algorithms
  • Implemented systems (highly optimised)

DAML+OIL classes can be names (URI’s) or expressions
  • Various constructors provided for building class expressions

Expressive power determined by
  • Kinds of constructor provided
  • Kinds of axiom allowed
## DAML+OIL Class Constructors

<table>
<thead>
<tr>
<th>Constructor</th>
<th>DL Syntax</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>intersectionOf</td>
<td>$C_1 \sqcap \ldots \sqcap C_n$</td>
<td>Human $\sqcap$ Male</td>
</tr>
<tr>
<td>unionOf</td>
<td>$C_1 \sqcup \ldots \sqcup C_n$</td>
<td>Doctor $\sqcup$ Lawyer</td>
</tr>
<tr>
<td>complementOf</td>
<td>$\neg C$</td>
<td>$\neg$Male</td>
</tr>
<tr>
<td>oneOf</td>
<td>${x_1 \ldots x_n}$</td>
<td>${\text{john, mary}}$</td>
</tr>
<tr>
<td>toClass</td>
<td>$\forall P.C$</td>
<td>$\forall$hasChild.Doctor</td>
</tr>
<tr>
<td>hasClass</td>
<td>$\exists P.C$</td>
<td>$\exists$hasChild.Lawyer</td>
</tr>
<tr>
<td>hasValue</td>
<td>$\exists P.{x}$</td>
<td>$\exists$citizenOf.{USA}$</td>
</tr>
<tr>
<td>minCardinalityQ</td>
<td>$\geq nP.C$</td>
<td>$\geq 2$hasChild.Lawyer</td>
</tr>
<tr>
<td>maxCardinalityQ</td>
<td>$\leq nP.C$</td>
<td>$\leq 1$hasChild.Male</td>
</tr>
<tr>
<td>cardinalityQ</td>
<td>$= n P.C$</td>
<td>$= 1$hasParent.Female</td>
</tr>
</tbody>
</table>

- XMLS **datatypes** as well as classes
- Arbitrarily complex **nesting** of constructors
  - E.g., Person $\sqcap \forall$hasChild.(Doctor $\sqcup \exists$hasChild.Doctor)
### DAML+OIL Axioms

<table>
<thead>
<tr>
<th>Axiom</th>
<th>DL Syntax</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>subClassOf</td>
<td>$C_1 \sqsubseteq C_2$</td>
<td>Human $\sqsubseteq$ Animal $\sqcap$ Biped</td>
</tr>
<tr>
<td>sameClassAs</td>
<td>$C_1 \equiv C_2$</td>
<td>Man $\equiv$ Human $\sqcap$ Male</td>
</tr>
<tr>
<td>subPropertyOf</td>
<td>$P_1 \sqsubseteq P_2$</td>
<td>hasDaughter $\sqsubseteq$ hasChild</td>
</tr>
<tr>
<td>samePropertyAs</td>
<td>$P_1 \equiv P_2$</td>
<td>cost $\equiv$ price</td>
</tr>
<tr>
<td>sameIndividualAs</td>
<td>${x_1} \equiv {x_2}$</td>
<td>${\text{President_Bush}} \equiv {\text{G_W_Bush}}$</td>
</tr>
<tr>
<td>disjointWith</td>
<td>$C_1 \sqsubseteq \neg C_2$</td>
<td>Male $\sqsubseteq \neg$ Female</td>
</tr>
<tr>
<td>differentIndividualFrom</td>
<td>${x_1} \sqsubseteq \neg {x_2}$</td>
<td>${\text{john}} \sqsubseteq \neg {\text{peter}}$</td>
</tr>
<tr>
<td>inverseOf</td>
<td>$P_1 \equiv P_2^{-}$</td>
<td>hasChild $\equiv$ hasParent$^{-}$</td>
</tr>
<tr>
<td>transitiveProperty</td>
<td>$P^+ \sqsubseteq P$</td>
<td>ancestor$^+$ $\sqsubseteq$ ancestor</td>
</tr>
<tr>
<td>uniqueProperty</td>
<td>$\top \sqsubseteq \leq_1 P$</td>
<td>$\top \sqsubseteq \leq_1 \text{hasMother}$</td>
</tr>
<tr>
<td>unambiguousProperty</td>
<td>$\top \sqsubseteq \leq_1 P^{-}$</td>
<td>$\top \sqsubseteq \leq_1 \text{isMotherOf}^{-}$</td>
</tr>
</tbody>
</table>

* Axioms (mostly) reducible to subClass/PropertyOf
<daml:Class>
  <daml:intersectionOf rdf:parseType="daml:collection">
    <daml:Class rdf:about="#Person"/>
    <daml:Restriction>
      <daml:onProperty rdf:resource="#hasChild"/>
      <daml:toClass>
        <daml:unionOf rdf:parseType="daml:collection">
          <daml:Class rdf:about="#Doctor"/>
          <daml:Restriction>
            <daml:onProperty rdf:resource="#hasChild"/>
            <daml:hasClass rdf:resource="#Doctor"/>
          </daml:Restriction>
        </daml:unionOf>
      </daml:toClass>
    </daml:Restriction>
  </daml:intersectionOf>
</daml:Class>
XML Datatypes in DAML+OIL

- DAML+OIL supports the full range of XML Schema datatypes
  - Primitive (e.g., decimal) and derived (e.g., integer sub-range)
- Clean separation between “object” classes and datatypes
  - Disjoint interpretation domains: John$^T \neq$ (int 5)$^T$
  - Object properties disjoint from datatype properties
- Philosophical reasons:
  - Datatypes structured by built-in predicates
  - Not appropriate to form new datatypes using ontology language
- Practical reasons:
  - Ontology language remains simple and compact
  - Semantic integrity of ontology language not compromised
  - Implementability not compromised—can use hybrid reasoner
- In practice, DAML+OIL implementations can choose to support subset of XML Schema datatypes.
Reasoning with DAML+OIL
Why Provide Reasoning Services?

- **Understanding** closely related to reasoning
  - Semantic Web aims at machine understanding
- Reasoning useful at all stages of ontology life-cycle
- Ontology *design and maintenance*
  - Check class consistency and (unexpected) implied relationships
  - Particularly important with large ontologies/multiple authors
- Ontology *integration*
  - Assert inter-ontology relationships
  - Reasoner computes integrated class hierarchy/consistency
- Ontology *deployment*
  - Determine if set of facts are consistent w.r.t. ontology
  - Determine if individuals are instances of ontology classes
Why Decidable Reasoning?

- DAML+OIL constructors/axioms restricted so reasoning is **decidable**
- Consistent with Semantic Web’s **layered architecture**
  - XML provides syntax **transport layer**
  - RDF(S) provides basic **relational language** and simple ontological primitives
  - DAML+OIL provides powerful but still decidable **ontology language**
  - Further layers (e.g., **rules**) will extend DAML+OIL
  - **Extensions** will almost certainly be undecidable
- Facilitates provision of **reasoning services**
  - Known “practical” **algorithms**
  - Several implemented **systems**
  - Evidence of **empirical tractability**
- Understanding dependent on **reliable & consistent** reasoning
Basic Inference Problems

☞ **Consistency** — check if knowledge is meaningful
  - Is \( \mathcal{O} \) consistent? There exists some model \( \mathcal{I} \) of \( \mathcal{O} \)
  - Is \( C \) consistent? \( C^\mathcal{I} \neq \emptyset \) in some model \( \mathcal{I} \) of \( \mathcal{O} \)

☞ **Subsumption** — structure knowledge, compute taxonomy
  - \( C \sqsubseteq_\mathcal{O} D \) \( \iff \) \( C^\mathcal{I} \subseteq D^\mathcal{I} \) in all models \( \mathcal{I} \) of \( \mathcal{O} \)

☞ **Equivalence** — check if two classes denote same set of instances
  - \( C \equiv_\mathcal{O} D \) \( \iff \) \( C^\mathcal{I} = D^\mathcal{I} \) in all models \( \mathcal{I} \) of \( \mathcal{O} \)

☞ **Instantiation** — check if individual \( i \) instance of class \( C \)
  - \( i \in_\mathcal{O} C \) \( \iff \) \( i \in C^\mathcal{I} \) in all models \( \mathcal{I} \) of \( \mathcal{O} \)

☞ **Retrieval** — retrieve set of individuals that instantiate \( C \)
  - set of \( i \) s.t. \( i \in C^\mathcal{I} \) in all models \( \mathcal{I} \) of \( \mathcal{O} \)

☞ Problems all **reducible** to consistency (satisfiability):
  - \( C \sqsubseteq_\mathcal{O} D \) iff \( D \cap \neg C \) not consistent w.r.t. \( \mathcal{O} \)
  - \( i \in_\mathcal{O} C \) iff \( \mathcal{O} \cup \{ i \in \neg C \} \) is **not** consistent
Highly Optimised Implementation

- Naive implementation $\rightarrow$ effective non-termination
- Modern systems include **MANY** optimisations
- Optimised **classification**
  - Use enhanced traversal (exploit information from previous tests)
  - Use structural information to select classification order
- Optimised **subsumption** testing
  - Normalisation and simplification of concepts
  - Absorption (simplification) of general axioms
  - Davis-Putnam style semantic branching search
  - Dependency directed backtracking
  - Caching of satisfiability results and (partial) models
  - Heuristic ordering of propositional and modal expansion
  - ...
Research Challenges
Research Challenges

- **Increased expressive power**
  - Existing DL systems implement (at most) $SHIQ$
  - DAML+OIL extends $SHIQ$ with datatypes and nominals

- **Scalability**
  - Very large KBs
  - Reasoning with (very large numbers of) individuals

- **Other reasoning tasks**
  - Querying
  - Matching
  - Least common subsumer
  - ... 

- **Tools and Infrastructure**
Increased Expressive Power: Datatypes

- **DAML+OIL** has simple form of datatypes
  - Unary predicates plus disjoint object-class/datatype domains
- Well understood *theoretically*
  - Existing work on *concrete domains* [Baader & Hanschke, Lutz]
  - Algorithm already known for $SHOQ(D)$ [Horrocks & Sattler]
  - Can use *hybrid reasoning* (DL reasoner + datatype “oracle”)
- May be *practically* challenging
  - All XMLS datatypes supported (?)
- Already seeing some (partial) *implementations*
  - Cerebra system (Network Inference), Racer system (Hamburg)
Increased Expressive Power: Nominals

- DAML+OIL `oneOf` constructor equivalent to hybrid logic nominals
  - Extensionally defined concepts, e.g., \( \text{EU} \equiv \{ \text{France, Italy, \ldots} \} \)

- Theoretically very challenging
  - Resulting logic has known high complexity (NExpTime)
  - No known “practical” algorithm
  - Not obvious how to extend tableaux techniques in this direction
    - Loss of tree model property
    - Spy-points: \( T \sqsubseteq \exists R. \{ S_{py} \} \)
    - Finite domains: \( \{ S_{py} \} \sqsubseteq \leq n R^{-} \)
  - Promising research on automata based algorithms

- Standard solution is weaker semantics for nominals
  - Treat nominals as (disjoint) primitive classes
  - Loose some inferential power, e.g., w.r.t. max cardinality
Scalability

- Reasoning **hard** — even without nominals (i.e., SHIQ)
- Web ontologies may grow **very large**
- Good **empirical evidence** of scalability/tractability for DL systems
  - E.g., 5,000 (complex) classes – 100,000+ (simple) classes
- But evidence mostly w.r.t. SHF (no inverse)
- **Problems** can arise when SHF extended to SHIQ
  - Important **optimisations** no longer (fully) work
- Reasoning with **individuals**
  - Deployment of web ontologies will mean reasoning with (possibly very large numbers of) individuals/tuples
  - Unlikely that standard **Abox** techniques will be able to cope
  - Necessary to employ **database** technology
Other Reasoning Tasks

**Querying**
- Retrieval and instantiation wont be sufficient
- Minimum requirement will be *conjunctive query language* [Tessaris & Horrocks]
- May also need “what can I say about $x$?” style of query [Bechhofer & Horrocks]

**Explanation** [McGuinness, Borgida et al]
- To support ontology design
- Justifications and proofs

**LCS** and/or **matching** [Baader, Küsters & Molitor]
- To support ontology integration
- To support “bottom up” design of ontologies
Summary

- **Semantic Web** aims to make web resources accessible to automated processes
- **Ontologies** will play key role by providing vocabulary for semantic markup
- **DAML+OIL** is an ontology language designed for the web
  - Exploits existing standards: XML, RDF(S)
  - Formal rigor of Description Logic
  - KR idioms from object oriented and frame systems
- **Popular** combination of features—already being widely adopted
- **Challenges** remain
  - Reasoning with full language
  - Demonstration of scalability
Acknowledgements

- Members of the OIL and DAML+OIL development teams, in particular Dieter Fensel and Frank van Harmelen (Amsterdam) and Peter Patel-Schneider (Bell Labs)
- Franz Baader, Uli Satler and Stefan Tobies (Dresden — formerly Aachen)
- Carole Goble and other members of the Information Management Group (University of Manchester)
Resources

Slides from this talk

http://www.cs.man.ac.uk/~horrocks/Slides/edbt02.pdf

FaCT system (open source)

http://www.cs.man.ac.uk/FaCT/

OilEd (open source)

http://oiled.man.ac.uk/

OIL

http://www.ontoknowledge.org/oil/

DAML+OIL

http://www.w3c.org/Submission/2001/12/