The Web Ontology Language

A(n Over)View not entirely
without bias

The Semantic Web

- The “next generation”, “machine processable” Web
  - – Interlinked information for programs
- The “original vision” and “ultimate destiny”
- “Webized” Knowledge Represenation
  - – *To do for KR what the Web did for hypertext*
  - – The Web as giant Semantic Net
  - – Like most Semantic Net tech, it has been *logicized*
    - Though, like Semantic Nets, there is still resistence

Web identifiers: URIs

- Core bit of Web Architecture
  - Constraint: Identify with URIs
  - Principle of URI assignment
    - URIfy resources others wish to refer to
  - Good Practice: avoid aliases
  - Good Practice: avoid ambiguity
- In Web Hyperlinks, explicit URIs primarily name link *targets*
  - Either in a source, e.g., <a href=…
  - Or in a target, e.g., <a name=…
  - But the source tends to be a target for others
  - Link arcs tend to remain unnamed

From Links to Assertions

- <a href=“http://www.example.com/test” />
- Three parts
  - Source = The containing document
  - Arc = href
  - Target = http://www.example.com/test
- href(Source, Target), that is, a *relation*
- Name the relation with a URI
  - Name the *predicate*
  - Now we have an RDF triple

RDF Graphs

- A RDF graph is a collection of triples
  - A conjunction of assertions
  - A chunk of the Semantic Web/Net
  - A set of resources connected to other resource in a certain pattern
    - A *view* of the Web
- RDF graphs are entirely asserted
  - There are no *quoted* triples
  - There may be *encoded* triples (e.g., reification)

RDF Syntax

- Many syntactic quirks, but the triple lives:

```
<rdf:Description rdf:ID=“Jen”>
     <hasTitle rdf:resource=“#AdjunctProfessor”/>
</rdf:Description>
```

- Many syntax alternatives
  - Some of you may have heard of N3
- Even the raw triples have assertional weight
  - It’s logic all the way down, not data structures
  - Ground RDF graphs are the closest
  - Consider bnodes

RDF Inference

- What is a KR with out licenced and encouraged inferences?
  - Not much of one!
  - See: What is a Knowledge Representation?
    - http://medg.lcs.mit.edu/ftp/psz/k-rep.html
- Not a lot in RDF alone
  - The predicate rdf:type is special
  - Bnodes are significant (instance lemma)
  - Simple entailment: S entails E iff some subgraph(S) is an instance of E
  - Few other things

From Facts to Taxonomies

- RDF graphs are close to relational database
  - Bit other, actually, but close
- RDF alone lacks *structuring*
  - We know things are rdf:Properties and rdf:XMLLiteral, but not much more
  - Things can be members of classes (rdf:type), but we don’t know what classes are
  - We don’t have any class/class relations
    - Except for membership
- We can do better!
  - Add more standard meaning to certain triples

RDF Schema

- Classes, Properties, and Resources
  - New classes: rdfs:Class, rdfs:Resource
  - Certain relations between
    - classes and classes: rdfs:subClassOf
    - properties and properties: rdfs:subPropertyOf
    - properties and classes: rdfs:range and rdfs:domain
  - New inferences:
    - a rdf:type B, B rdfs:subClassOf C
      - a rdf:type C
- But very weak!
  - Only explicit connected class and property hierarchies
  - Inference (without negation), not validation!
    - a prop b, a rdf:type C, prop rdfs:range D
      - a rdf:type C *and* a rdf:type D

From Taxonomies to Ontologies

- Taxonomies are useful
  - Dmoz, Yahoo, et al.
- But they tend to be inexpressive
  - The are defined in very weak languages
  - The relation between class and class, and class and instance, and everything is explicit (mostly)
  - Fall prey to the Metacrap problems
- Rough rule:
  - If you need to classify, you have an ontology

What is an Ontology?

- Ontologies in CS
  - We philosophers weep!
  - Shared formalization of a conceptualization
  - A logical *theory* encoded as input to an automated *reasoner* (or other program
- Thus, an ontology (typically)
  - is a collection of *axioms* (and other assertions)
  - is connected with human intentions and understanding
  - is connected to program behavior
- So, the ontology language should be
  - For people (epistemically adequate)
  - For programs (sensibly computable)

What is an Ontology? II

- something rdf:type owl:Ontology?
- An application/rdf+xml document?
- A collection of classes and properties (and individuals?)
  - With some degree of axiomization?

Desiderata for a Web Ontology Language

- Expressive
  - Negation, cardinality restrictions, class construction, property features
  - Classes as instances and other metamodeling
  - Self-axiomatizable
- Web centered
  - Use URIs
  - Use the Web (owl:imports anyone?)
- Implementable
  - The reasoning procedures should be “practical”

Two varieties of OWL

- OWL DL
  - – OWL Lite
- OWL Full

Class Expression Constructors

- Boolean (and, or, not, enumerated)
  - owl:intersectionOf
  - owl:unionOf
  - owl:complementOf
  - owl:oneOf
- Quantification (*restrictions*)
  - Universal (owl:allValuesfrom)
  - Existential (owl:someValuesFrom)
  - Nominals (owl:hasValue)
  - Number restrictions (owl:minC/maxC/cardinality)

Class Axioms

- These relate two *class expressions*
  - Class names (that is, URIs)
  - Class expressions
    - Anything formed by the Class Expression Constructors
  - Class names to class expressions
- subClassOf
  - From name to expression == *necessary* conditions
  - From expression to name == *sufficient* conditions
  - equivalentClass == both
- Class expressions on both sides:
  - General constraints

Property Axioms

- subPropertyOf and equivalentProperty
- inverseOf
- FunctionalProperty and InverseFunctionalProperty
- SymmetricProperty
- range and domain
- AnnotationProperty, OntologyProperty, DatatypeProperty, ObjectProperty

Facts

- Class expressions, and Class and Property Axioms are encoded in *triples*

```
<rdf:Description rdf:ID="HedgeHog">
    <rdfs:subClassOf rdf:resource="#Pet"/>
</rdf:Description>
<owl:Class rdf:ID="#Pet"/>
```

- The rest of the triples are facts
  - The syntax triples are (in OWL Full) facts too!

Ontology Headers

- All the properties hanging off a rdf:type owl:Ontology
  - Includes metadata
    - owl:versionInfo
    - rdfs:comment
- And a key modularity/webizing feature
  - *owl:imports*

```
<owl:Ontology rdf:about="“”>
   <owl:imports>
        <owl:Ontology rdf:about=“http://www.someotherontology.org/…”/>
   </owl:imports>
</owl:Ontology>
```

Multiple Ontologies: owl:imports

- owl:imports is either obvious or mysterious
  - Key members of WG (e.g., Pat Hayes and Dan Connelly) have claimed not to understand it
  - Operational meaning (roughly):
    - include all the axioms and facts of the imported ontology (which includes all the axioms and facts of *its* imported ontologies, etc.)
  - This is (barring syntax tricks like entities) the only way external meanings get into your ontology
    - That is, just *using* a URI from some other URI space is not enough
    - Transclusion: owl:imports is like <img src=…
    - Link: URI use is (a bit) like <a href=…
    - Work underway (C-OWL, E-connections, PECs, syntactic, etc.)
  - The imports closure is *flat*

Metamodeling

- *Everything* is an element of the domain
  - In common FOL, only individuals
    - Classes, properties, syntax are not
    - Quantifying over classes, etc. moves to second order
  - OWL (Full)
    - Classes are individuals
      - owl:Class rdf:type owl:Class
    - Properties are individuals
      - ex:trueLove rdf:type owl:SymmetricProperty
    - And syntax
      - owl:unionOf rdf:type rdf:Property

Metamodeling II

- OWL Full has second order syntax and first order semantics
  - Axiomatics (LBase, DAML+OIL)
    - Everything is a triple
    - Rules express consequences of those triples
  - Model theory
    - Classes are objects with relations to their extentions
    - Following HiLog, SKIF
- This is very expressive! Perhaps Web like
  - Everything really *is* a resource
  - Anyone can say anything about anything
  - Self-describing, partial/bootstrap understanding

Difficulties with OWL Full

- Expressive, but not expressive enough
- Inference procedures not well worked out
- Semantics are non-traditional
  - Some common metamodeling schemes (e.g., UML) are stratified
  - Have to reinvent a lot
- No complete implementation
  - Not clear what strategies are best

OWL DL

- The Description Logic SHION(Dn) (plus a bit)
  - A decidable subset of FOL (and of OWL Full)
  - Metalogical terms
    - Sound
    - Complete
    - Decidable (has a decision procedure)
      - Semi-decidable: every yes question answering terminates
  - High complexity (NExpTime, ExpTime for major subsets)
  - “Practical”, highly optimized algorithms
  - Several implementations and lots of experience
  - Strong user, theory, implementor communities
  - A more traditional FOL fragment

Description Logics

- Semantic Nets were Scruffy
  - Coming out linguistic and cognative modeling
  - Lots of pointer chasing
    - And other implementation dependent moves
    - Some notable innovations
    - Some notable “confusions” (isa/instanceOf)
  - “Intuitively” Object Oriented
- Send in the neats
  - Pat Hayes: “The logic of frames”
  - Brachman & friends, KL-One
  - Trade expressivity for tractability
  - Trade tractability for “practical” expressivity

Traditional Semantics

- Classes are 1-place predicates
  - Person == Person(x)
  - bob rdf:type Person == Person(bob)
- Properties are 2-place predicates
- Axioms (typically) are conditionals
  - C subClassOf D == if C then D
- Syntactic distinctions between categories
  - ObjectProperty disjointWith DatatypeProperty
  - Class disjointWith Thing (and ObjectProperty, etc.)
  - Syntax triples disappear when converting to abstract syntax
  - Hard to enforce in RDF graphs!
- Then inherit the semantic of FOL

Annotation Properties

- We regain a bit of syntactic higher order
  - Classes, Ontologies, Properties, etc. can have *AnnotationProperties*
  - Within the ontology, AnnotationProperties obey a lot of restrictions
    - Disjoint from all other Property types, classes, etc.
    - Cannot participate in axioms
  - AnnotationProperties are *invisible* to the reasoner
- More like structured comments

Difficulties with OWL DL

- Pretty expressive, but not *that* expressive
  - Can’t define uncleOf!
- Pretty scalable, but not *that* scalable
  - This ain’t no database
  - Pathological cases are crippling
    - It remains to be seen if the Semantic Web normal case is pathological
  - Simple implementations fair quite poorly
  - No known (or published) OWL DL algorithm
- Layered DL in RDF/XML is painful to write

Problems with OWL

- Closed vs. open world
- Inference vs. data validation
- No Unique Names Assumption
- Scalability
- Rules
- Other Expressivity

SWRL (neé OWL Rules)

- *Very* expressive superset of OWL
- Extends the permissible material conditionals
  - Allows class and property *atoms* with arbitrary number of variables
  - Used in OWL-S for precondition/effect formulas
- Relies on OWL for much of its expressivity
- Not very closed world/Datalog/Prolog friendly