Semantic Web Technologies

Relations between Semantic Web Languages

Jos de Bruijn
debruijn AT inf.unibz.it

KRDB Research Group
Free University of Bolzano, Italy

10 December 2007

Outline

RDF Recap

- RDF
  - Graph-based data model
  - RDF statements are triples (subject, predicate, object)
  - URIs as identifiers for resources and properties
  - Literals as concrete data values
  - RDF/XML as “default” syntax
  - RDF vocabulary: rdf:type, rdf:Property, ...

RDF Recap

- RDFS (RDF Vocabulary Description)
  - Vocabulary for describing RDF classes and properties
  - rdfs:Class, rdfs:Resource, rdfs:domain, rdfs:range, ...
  - Simple ontology language
  - Allows additional entailments:
    - (?x rdf:type ?y), (?y rdfs:subClassOf ?z) ⇒ (?x rdf:type ?z)
    - (?x rdfs:subClassOf ?y), (?y rdfs:subClassOf ?z) ⇒ (?x rdfs:subClassOf ?z)
    - (?x ?p ?y), (?p rdfs:subPropertyOf ?q) ⇒ (?x ?q ?y)
    - (?p rdfs:domain ?x), (?y ?p ?z) ⇒ (?y rdf:type ?x)
    - (?p rdfs:range ?x), (?y ?p ?z) ⇒ (?z rdf:type ?x)

Layering on top of RDF(S)

- RDF(S) bottom layer in Semantic Web stack
- Higher languages **layer** on top of RDFS

Syntactic Layering

- **Every valid RDF statement is a valid statement in a higher language**
- This **includes** triples containing keywords of these languages(!)

Semantic Layering

For RDFS graph G and higher-level language L:

If $G \models_{RDFS} G'$ then $G \models_L G'$, and **ideally**

if $G \models_L G'$ then $G \models_{RDFS} G'$

Web Ontology Language OWL

- OWL Lite
  - Layered on subset of RDFS
    - Symmetric, transitive, inverse properties
    - Complete class definitions
    - Limited types of restrictions
  - Based on Description Logic $\mathcal{SHIN}(\mathbf{D})$
- OWL DL
  - Full-fledged Description Logic $\mathcal{SHOIN}(\mathbf{D})$
    - Negation
    - Disjunction
    - Full cardinality
- OWL Full
  - Extension of RDFS, OWL DL
    - Meta-modeling
    - Statements about language primitives
  - Every RDF graph is valid OWL Full

Syntactically layering OWL on RDF(S)

OWL Lite, OWL DL

- OWL Lite, OWL DL subsets of RDF
- Allowed triples defined through mapping from abstract syntax
- **Partial** layering:
  - **every** OWL Lite/DL ontology is an RDF graph
  - **some** RDF graphs are OWL Lite/DL ontologies

OWL Full

- OWL Full encompasses RDF
- **Complete** layering:
  - **every** OWL Full is an RDF graph
  - **all** RDF graphs are OWL Full ontologies

Semantically layering OWL on RDF(S)

OWL Lite, OWL DL

- OWL Lite/DL semantics **not** related to RDFS semantics
- Redefine semantics of RDFS keywords, e.g., `rdfs:subClassOf`
- Work ongoing to describe correspondence between subset of RDFS and OWL Lite/DL

OWL Full

- OWL Full semantics is **extension** of RDFS semantics
- OWL Full is undecidable
- OWL Full semantics hard to understand

OWL Lite/DL vs. RDF

- RDF Graph defined through translation from Abstract Syntax
- Example:

  Class(Human partial Animal
  restriction(hasLegs cardinality(2))
  restriction(hasName allValuesFrom(xsd:string)))

Human	rdf:type	owl:Class
Human	rdfs:subClassOf	Animal
Human	rdfs:subClassOf	_:X1
_:X1	rdf:type	owl:Restriction
_:X1	owl:onProperty	hasLegs
_:X1	owl:cardinality	"2"8sd:nonNegativeInteger
Human	rdfs:subClassOf	_:X2
_:X2	rdf:type	owl:Restriction
_:X2	owl:onProperty	hasName
_:X2	owl:allValuesFrom	xsd:string

OWL Lite/DL vs. RDF

- Not every RDF graph is OWL Lite/DL ontology
- Example:

  A rdf:type **A**
- How to check whether an RDF graph **G** is OWL DL?
  1. Construct an OWL ontology **O** in Abstract Syntax
  2. Translate to RDF graph **G'**
  3. If **G**=**G'**, then **G** is OWL DL
     - Otherwise, go to step (1)

Modeling style

- Modeling Triples
  - Statements of the form ⟨A,B,C⟩
  - Special vocabulary for ontology modeling
    - ⟨A,rdf:type,C⟩ (class membership)
    - ⟨A,rdfs:subClassOf,C⟩ (class hierarchy)
    - ⟨A,rdfs:subPropertyOf,C⟩ (property hierarchy)
    - ⟨A,rdfs:domain,C⟩ (domain restrictions)
    - ⟨A,rdfs:range,C⟩ (range restrictions)
  - No restrictions on use of vocabulary
    - Language constructs in the language
- Modeling DL
  - Class axioms: $C \sqsubseteq D$
  - Property axioms: $P \sqsubseteq Q$
  - Local property restrictions: $C \sqsubseteq \forall R.D$
  - Separation of vocabulary
    - class, property, individual identifiers disjoint
    - No use of language constructs in language

Expressiveness

- OWL DL allows
  - Arbitrary class azioms
  - Local property restrictions
  - Cardinality restrictions
  - (in)equality statements
  - inverse, transitive, symmetric properties
  - disjunction
  - disjointness, negation
- RDFS allows
  - Using classes as instances
  - Modifying language
    - Using language constructs in language

Translating RDF to F-Logic

- Translate triples to attribute molecules
- $\langle$A,B,C$\rangle \mapsto$ `A[B ->> C]`
- Axiomatize RDF semantics
  - Add facts for axiomatic triples
  - Add rules for entailment rules
- Treat bNodes as skolem constants
  - "rigid" bNodes
  - anonymous resources

bNodes as Anonymous Resources

- Unnumbered anonymous resource: _#
  - Globally unique constant identifier
  - Every occurrence is **fresh** identifier
- Numbered anonymous resource: `_#1, _#2, ...`
  - Resource has **scope**: formula
  - Within scope, different occurrences denote the same resource
- Anonymous resources are **not** existential variables
- Thus, semantics of bNodes cannot be captured completely

Expressiveness

- F-Logic allows
  - Arbitrary rules
  - Functional (single-valued) attributes
  - Local attributes, rather than global properties
- RDFS allows
  - Modifying language
    - Using language constructs in language
  - Global property restrictions, property hierarchy
    - Can be simulated using rules
  - bNodes as existentials
    - Form of non-ground entailment

Expressiveness of DL vs LP

- Description Logics allow
  - Existential quantification
  - Disjunction
  - Classical negation
  - Non-ground entailment
  - Equality
- Logic Programming allows
  - *n*-ary predicates
  - Chaining variables avoer predicates
  - Default negation (negation-as-failure)

Ontology Modeling in OWL DL vs F-Logic

- F-Logic allows quantification over class, attribute names, e.g `john:X, john[X -> mary]`
- F-Logic allows modeling classes-as-instances: `b747:airplane, x567:b747`
- In F-Logic, attributes "belong" to a class: `person[hasChild =>> person]` vs. $person \sqsubseteq \forall hasChild.person$
- Interpretation of classes, properties
  - In DLs, classes, attributes are interpreted **directly** as sets, binary relations, e.g. $person^{\mathcal{I}} \subseteq \Delta$
  - In F-Logic
    - Terms interpreted as elements of the domain, e.g. $\mathbf{I}_F(person) = a$
    - Elements of the domain are **associated with** sets, functions, relations, e.g. $\{k \mid k \in_U a\} \subseteq U$

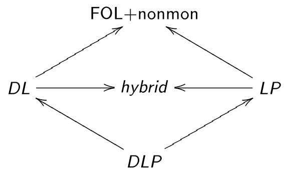
$A \sqsubseteq B$ vs. `A::B`

- Compare subsumption in DLs with subclass statements in F-Logic
- Subsumption in Description Logics: $A \sqsubseteq B$
  - A, B may be arbitrary concept descriptions
  - For interpretation $\mathcal{I}$, $\mathcal{I} \models A \sqsubseteq B$ iff $A^{\mathcal{I}} \subseteq B^{\mathcal{I}}$
  - Interpretaton of A, B directly as sets
  - $A \sqsubseteq B$ is entailed if $A^{\mathcal{I}} \subseteq B^{\mathcal{I}}$ for all models
- Subclass in F-Logic: `A::B`
  - A, B are terms
  - For interpretation $\mathbf{I}$, $\mathbf{I} \models A :: B$ iff $\mathbf{I}_F(A) \preceq_U \mathbf{I}_F(B)$
  - `A::B` is entailed if `A::B` or `A::....::B` explicitly asserted
    - Recall: **if** $a \in_U b$ **and** $b \preceq_U c$ **then** $a \in_U c$ (only one direction!)
  - Can be checked with ground entailment (Logic Programming)

Minimizing the cardinality of the domain

- Restricting size of the domain
- F-Logic requires objects for classes, attributes
- If domain is restricted, classes, attributes are "forced" to be interpreted the same
- Example: $\{\forall x(x = a); a : b\} \models a : c$
  - domain of every interpretation has cardinality one
  - thus, a, b, c always interpreted the same
- Description Logics: $\{\top \sqsubseteq \{a\}, a \in B\} \not\models a \in C$
  - domain of every interpretation has cardinality one
  - however, B, C may still be interpreted differently

Integration of DLs and rules



- Extension of DLP
  - Interoperation only for inexpressive ontologies
- Unified language
  - Undecidable; little research has been done
- Hybrid integration
  - Full DL and LP
  - Several existing approaches

DLs and F-Logic can interoperate in certain restricted cases

Summary

RDF and OWL
- Layering OWL on top of RDF
- Ontology Modeling in RDFS vs. OWL DL

RDF and F-Logic
- Encoding RDF in F-Logic
- Ontology Modeling in RDFS vs. F-Logic

OWL DL and F-Logic Programming

Further reading

- Ian Horrocks, Peter F. Patel-Schneider: Three theses of representation in the semantic web. WWW 2003: 39–47.
- J. de Bruijn, A. Polleres, R. Lara, and D. Fensel. OWL DL vs. OWL Flight: Conceptual modeling and reasoning on the semantic web. In WWW2005, Chiba, Japan, 2005. ACM.
- J. de Bruijn and S. Heymans. Translating ontologies from predicate-based to frame-based languages. In RuleML-2006, Athens, Georgia, USA, November 10–11 2006. IEEE.