On Logical Foundation of the Semantic Web

An institution-based approach

Dorel Lucanu

dlucanu@info.uaic.ro

“Alexandru Ioan Cuza” University, Romania
Outline

motivation
institutions
Semantic Web (SW) languages
  RDF and RDF Schema
  OWL
  SWRL
  SWRL FOL
relationships between SW languages
conclusion
Motivation

- an integrating mathematical structure for Semantic Web languages
- soundness of the reasoners for Web ontologies
- translating Web ontologies into other formalisms

“NOTE: There is a strong correspondence between the semantics for OWL DL defined in this section (RDF-based) and the Direct Model-Theoretic Semantics defined in . . . . If, however, any conflict should ever arise between these two forms, then the Direct Model-Theoretic Semantics takes precedence.” (OWL Web Ontology Language Semantics and Abstract Syntax Section 5. RDF-Compatible Model-Theoretic Semantics, http://www.w3.org/TR/owl-semantics/rdfs.html)
Institutions

- formalize the notion of "a logic"
- study the properties of a logic
  - representation
  - implementation
  - translation of logics
Institutions: Ingredients

- **signatures** \( \Sigma \)
- formalize vocabularies

Example: (Many Sorted) First Order Logic with Equality (FOLEQ)

<table>
<thead>
<tr>
<th>Booleans</th>
<th>Integers</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>sorts</strong></td>
<td></td>
</tr>
<tr>
<td><strong>constants</strong></td>
<td></td>
</tr>
<tr>
<td>false, true : Bool</td>
<td>0 : Int</td>
</tr>
<tr>
<td><em>and</em> : Bool Bool → Bool</td>
<td>succ, pred : Int → Int</td>
</tr>
<tr>
<td><em>×</em> : Int Int → Int</td>
<td></td>
</tr>
</tbody>
</table>
Institutions: Ingredients

signatures (continued)

are organized as categories, where the signature morphisms formalizes the translations between vocabularies

\[ \phi : \text{Booleans} \to \text{Integers} \]

\[
\begin{align*}
\text{Bool} & \leftrightarrow \text{Int} \\
false & \mapsto 0, \ true \mapsto \text{succ}(0) \\
\_\text{and}_\_ & \mapsto \_\times\_
\end{align*}
\]
Institutions: Ingredients

sentences
- abstracts the notion of formula
- is formalized as a functor
  \[ \text{sen} : \text{Sign} \to \text{Set} \]
  \[ \text{sen}(\Sigma) = \text{the set of well-formed first-order formulas built over } \Sigma \]
  \[ (\forall y : \text{Int}) \; y \times 0 = 0 \]
  \[ (\forall y : \text{Bool}) \; y \text{ and false } = \text{false} \]
- if \( \phi : \Sigma \to \Sigma' \) then \( \text{sen}(\phi) : \text{sen}(\Sigma) \to \text{sen}(\Sigma') \)
- \( \text{sen}(\phi)(y \text{ and false } = \text{false}) = (y \times 0 = 0) \)

Notation: \( \text{sen}(\phi)(s) \overset{\text{not}}{=} \phi(s) \)
Institutions: Ingredients

- models are interpretations of the syntactical constructs
- are parameterized over signatures: $\text{Mod}(\Sigma) = \text{the category of the } \Sigma\text{-models (interpretations of the vocabulary } \Sigma\text{)}$
- are formalized as a functor:
  $$\text{Mod} : \text{Sign}^{\text{op}} \to \text{Cat}$$
- if $\phi : \Sigma \to \Sigma'$ then $\text{Mod}(\phi^{\text{op}}) : \text{Mod}(\Sigma') \to \text{Mod}(\Sigma)$

$\mathbb{Z}$ is a model for Integers
$\text{Mod}(\phi^{\text{op}})(\mathbb{Z})$ is $\mathbb{Z}$ viewed as a model for Booleans

Notation: $\text{Mod}(\phi^{\text{op}})(M) \overset{\text{not}}{=} M|_{\phi}$
Institutions: Ingredients

satisfaction relation

- relates the models and the sentences: \( M \models_{\Sigma} s \)
  where \( M \) is \( \Sigma \)-model and \( s \) is a \( \Sigma \)-sentence

- it is the subject of the **satisfaction condition** which expresses the invariance of truth under change of notation

\[
M' \models_{\Sigma'} \phi(s) \text{ iff } M' \models_{\phi} \models_{\Sigma} s
\]

where \( \phi : \Sigma \to \Sigma' \), \( M' \) is a \( \Sigma' \)-model, and \( s \) is a \( \Sigma \)-sentence

\[
\mathbb{Z} \models_{\text{Integers}} y \times 0 = 0 \text{ iff } \\
\mathbb{Z} \models_{\phi} \models_{\text{Booleans}} y \text{ and } \text{false} = \text{false}
\]
Institutions: Specifications and Theories

- **a specification** is a pair \((\Sigma, S)\), where \(\Sigma\) is a signature and \(S\) is a set of sentences

- semantical consequences: \((\Sigma, S) \models s\) iff 
  \[
  (\forall M) (M \models_{\Sigma} S \Rightarrow M \models_{\Sigma} s)
  \]

- **a theory** is a specification \((\Sigma, S)\) s.t.
  \[
  (\forall s) ((\Sigma, S) \models s \Rightarrow s \in S)
  \]

- the inclusion \(\text{Th} \rightarrow \text{Spec}\) is an equivalence of categories

- theoroidal institutions:
  - signatures are theories
  - a \((\Sigma, S)\)-sentence is a \(\Sigma\)-sentence
  - \((\Sigma, S)\)-models are \(\Sigma\)-models satisfying \(S\)
  - \(M \models_{(\Sigma, S)} s\) iff \(M \models_{\Sigma} s\)
Institutions: Properties of interest

theory colimits
- the module expressions are evaluated as colimits of theories

model amalgamation
- expresses the possibility of amalgamation of consistent models for different specification modules

liberality
- expresses the possibility of free constructions generalizing the principle of “initial semantics”
Relating Institutions

- **morphism**: capture the way in which a “richer” institution is built over a “simpler” one
- **comorphism**: capture the way in which a “simpler” institution is embedded (encoded) into a “richer” one
- both are the subject of a corresponding satisfaction condition
- there exist a variety of definitions for morphisms and variety of definitions for comorphisms in literature
- a prover from the target logic can be used to prove properties from the source logic only if certain conditions are fulfilled
Institutions: Main references

- Introducing Institutions, by J. Goguen and R. Burstall, 1984
- Institutions: Abstract model theory for specification and programming, by J. Goguen and R. Burstall, 1992
- Structuring theories on consequence, by J. Fiadeiro and A. Sernadas - 1988
- May I Borrow Your Logic?, by M. Cerioli and J. Meseguer, 1993
- Moving Between Logical Systems, Andrzej Tarlecki, 1995
- Institution Morphisms, by J. Goguen and Gr. Rosu, 2002
- Grothendieck Institutions, by R. Diaconescu
Semantic Web

From Semantic Web talk by Tim Berners-Lee at XML 2000
RDF

- proposed in October 1997
- in February 1999 becomes a W3C recommendation
- it is a standard for representing information in the Web
- a expression in RDF is a collection of triples, each consisting of a subject, a property (predicate), and an object
RDF - example

```xml
<rdf:Description rdf:about="http://www-cs-faculty.stanford.edu/~knuth/">
  <hasName rdf:resource="Donald Knuth" />
</rdf:Description>

<rdf:Description rdf:about="http://www.amazon.com/exec/.../104-3442396-7552717">
  <hasAuthor rdf:resource="http://www-cs-faculty.stanford.edu/~knuth/" />
</rdf:Description>
```
We consider given a datatype $\mathbb{D}$

- signatures: $\Sigma = (\mathbb{RR}, \mathbb{BN})$
- sentences: $F ::= (s, p, o) \mid u \equiv v \mid F \land F \mid \neg F \mid (\forall y)F$
- models: $A = (\text{Res}_A, \text{Prop}_A, res_A, [\_]_A)$, where
  - $\text{Res}_A$ a set of resources
  - $\text{Prop}_A$ a set of properties (assume that $\text{Prop}_A \subseteq \text{Res}_A$)
  - $res_A : \mathbb{RR} \to \text{Res}_A$
  - $[\_]_A : \text{Prop}_A \to \mathcal{P}(\text{Res}_A \times (\text{Res}_A \cup \mathbb{D})))$

- satisfaction:
  - $A \models (s, p, o)$ iff $res_A(p) \in \text{Prop}_A$ and $(res_A(s), res_A(o)) \in [res_A(p)]_A$
  - $A \models u \equiv v$ iff $res_A(u) = res_A(v)$
Interpretation of the blank nodes

\[ \Sigma = (RR, BN) \]

A \( \Sigma \)-model \( A \) and a \( \Sigma \)-sentence \( s \)

\[ \phi : \Sigma \to \Sigma' = (RR \cup BN, \emptyset) \]

\( A \models_{\Sigma} s \) iff there is a \( \Sigma' \)-model \( A' \) s.t.

\[ A' \models_{\phi} = A \] and \( A' \models_{\Sigma'} s \)
Interpretation of the blank nodes

\[ \Sigma = (RR, BN) \]

a \( \Sigma \)-model \( A \) and a \( \Sigma \)-sentence \( \phi \)

\[ \phi : \Sigma \rightarrow \Sigma' = (RR \cup BN, \emptyset) \]

\( A \models \Sigma \phi \) iff there is a \( \Sigma' \)-model \( A' \) s.t.

\[ A' \models \phi = A \text{ and } A' \models \Sigma' \phi \]

The satisfaction of the RDF graphs:

a RDF graf is a set \( S \) of triples

\( A \models \Sigma S \) iff \( A \models \Sigma \land_{s \in S} \phi \)
Interpretation of the blank nodes

\( \Sigma = (RR, BN) \)

a \( \Sigma \)-model \( A \) and a \( \Sigma \)-sentence \( s \)

\( \phi: \Sigma \rightarrow \Sigma' = (RR \cup BN, \emptyset) \)

\( A \models_{\Sigma} s \) iff there is a \( \Sigma' \)-model \( A' \) s.t.

\( A' \models_{\phi} A \) and \( A' \models_{\Sigma'} s \)

The satisfaction of the RDF graphs:

a RDF graf is a set \( S \) of triples

\( A \models_{\Sigma} S \) iff \( A \models_{\Sigma} \bigwedge_{s \in S} s \)

which is not always the same with saying that \( A \) satisfies all the sentences in \( S \)
A specification in \textit{RDF++}

\[(\Sigma, S) = \{
\{ bk, dkhp, hasAuthor, hasName \}, \emptyset \},
(bk, hasAuthor, dkhp),
(dkhp, hasName, "Donald Knuth")\]
**RDF++**: properties

- **Sign**$_{\text{RDF++}}$ has colimits

  - e.g., merge:
    
    $$
    (\text{RR}_1 \cap \text{RR}_2, \emptyset) \rightarrow (\text{RR}_1, \text{BN}_1)
    $$
    
    $$
    (\text{RR}_2, \text{BN}_2) \rightarrow (\text{RR}_1 \cup \text{RR}_2, \text{BN}_1 \sqcup \text{BN}_2)
    $$

- **RDF++** is liberal (free constr. is a generalized Herbrand constr.)

- **RDF++** has amalgamation property ($\text{Mod}(\text{RDF++})$ preserves finite limits)
The specification $\text{RDFV}(\text{RDF Vocabulary})$

\[\text{RR}(\text{RDF}) = \{ \text{rdf: type, rdf: Property, rdf: list, rdf: nil, ...} \}\]

\[\text{BN}(\text{RDF}) = \emptyset\]

\[\text{S}(\text{RDF}) = \{
\begin{align*}
(rdf: \text{type}, rdf: \text{type}, rdf: \text{Property}), \\
(rdf: \text{nil}, rdf: \text{type}, rdf: \text{List}), \\
(\forall s, p, o)(s, p, o) \rightarrow (p, rdf: \text{type}, rdf: \text{Property}), \\
\ldots
\end{align*}
\} \]
RDF Schema

- proposed in March 1999
- is a standard which describes how to use RDF to describe RDF vocabularies
- it is claimed that it is a semantical extension of RDF
- introduces the basic primitives for ontology modeling:
  - classes, subclasses
  - subproperties
  - domain, range
  - ...
RDF Schema: Example

```xml
<rdfs:Class rdf:about="Book"/>

<rdfs:Class rdf:about="Person"/>

<rdfs:Class rdf:about="Author">
    <rdfs:subClassOf rdf:resource="#Person"/>
</rdfs:Class>

<rdf:Property rdf:about="hasAuthor">
    <rdfs:domain rdf:resource="Book"/>
    <rdfs:range rdf:resource="Author"/>
</rdf:Property>
```
RDFS is a theory!

\[ \mathsf{RR}(\text{RDFS}) = \mathsf{RR}(\text{RDF}) \cup \{ \text{rdfs: Class, rdfs: subClassOf,} \\
\text{rdfs: subPropertyOf, rdfs: domain, ...} \} \]

\[ \mathsf{BN}(\text{RDFS}) = \text{those used in sentences} \]

\[ \mathsf{S}(\text{RDFS}) = \mathsf{S}(\text{RDF}) \cup \]

\[
\left\{ \\
(\text{rdf: type, rdfs: domain, rdfs: Resource}), \\
(\text{rdfs: domain, rdfs: domain, rdf: Property}), \\
(\forall u, v, x, y)(x, \text{rdf: domain, } y) \land (u, x, v) \rightarrow (u, \text{rdf: type, } y) \\
(\forall x, y)(x, \text{rdfs: subClassOf, } y) \rightarrow (x, \text{rdf: type, rdfs: Class}), \\
(\forall x, y)(x, \text{rdfs: subClassOf, } y) \rightarrow (y, \text{rdf: type, rdfs: Class}), \\
(\forall u, x, y)(x, \text{rdfs: subClassOf, } y) \land (u, \text{rdf: type, } x) \rightarrow (u, \text{rdf: type, } y), \\
\ldots \\
\right\}
\]
The institution \textbf{\textit{RDFS}}

- signatures: theory morphisms \( \text{RDFS} \to (\Sigma, S) \)
  \( \text{Sign}(\text{RDFS}) \) is a comma category
  \( \text{RDFS} \to (\Sigma, S) \) how use RDF to describe RDF vocabularies

- sentences: \( \Sigma \)-sentences

- models: \( (\Sigma, S) \)-models
  \( \text{Mod}(\Sigma, S) \to \text{Mod}(\text{RDFS}) \to \text{Mod}_{\text{RDF}} \) semantical extension

- satisfaction: \( A \models_{\text{RDFS}} (\Sigma, S) \) \( S \) iff \( A \models_{\Sigma} S \)

- semantics of a class:
  \( \llbracket C \rrbracket_A = \{ x \mid A \models (x, \text{rdf}: \text{type}, C) \} \)

There is a simple theoroidal comorphism from \( \text{RDFS} \) to \( \text{RDF}^{++} \).
OWL

- proposed in March 2002
- a language used to describe Web ontologies
- has three levels: OWL LITE, OWL DL, OWL Full
- includes RDF Schema
- new items:
  - makes distinction between individual-valued properties and data-valued properties
  - cardinality restrictions
  - operations with classes
  - restrictions on properties
  - ontology imports
  - ...
OWL: Example

- each book has at least one author

```xml
<owl:Class rdf:ID="Author">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource=""#hasAuthor"" />
      <owl:minCardinality rdf:datatype=""#&xsd;nonNegativeInteger"">1
    </owl:minCardinality>
  </owl:Restriction>
</rdfs:subClassOf>
</owl:Class>
```
OWL is also a theory!

\[ \text{RR(OWL)} = \text{RR(RDFS)} \cup \{ \text{owl: Thing, owl: Class, owl: subClassOf,} \]
\[ \text{owl: ObjectPropertyO, owl: DatatypeProperty, \ldots} \} \]
\[ \text{BN(OWL)} = \text{those used in sentences} \]
\[ S(\text{OWL}) = S(\text{RDFS}) \cup \]
\[ \{ \]
\[ (\text{owl: Nothing, rdf: type, owl: Class}), \]
\[ (\text{owl: Thing, rdf: type, owl: Class}), \]
\[ (\text{owl: Nothing, rdf: type, owl: Class}), \]
\[ \neg (\exists x)(x, \text{rdf: type, owl: Nothing}), \]
\[ (\forall x, C)(x, \text{rdf: type, C}) \land (C, \text{rdf: type, owl: Class}) \rightarrow \]
\[ (x, \text{rdf: type, owl: Thing}), \ldots \]
\[ \} \]

There is a forgetful morphism from \( \overset{\text{OWL}}{\longrightarrow} \) to \( \overset{\text{RDFS}}{\longrightarrow} \).

There is a simple theoroidal comorphism from \( \overset{\text{OWL}}{\longrightarrow} \) to \( \overset{\text{RDF++}}{\longrightarrow} \).
OWL DL hides some vocabulary items

\( \Sigma(\text{OWLDL}) = \text{hide} \)

- rdf:type, rdf: Property, ...
- owl: TransitiveProperty,...

in

\( \Sigma(\text{OWL}) \)

...and adds some new constraints:

\( \text{DLCONSTR} = \{ \)

\( (\forall x, C)(x, \text{rdf: type}, \text{owl: Thing}) \land (C, \text{rdf: type}, \text{owl: Class}) \rightarrow \)

\( \neg(x \equiv C), \)

...

\( \} \)

\( \phi : \Sigma(\text{OWLDL}) \leftrightarrow \Sigma(\text{OWL}) \)

\( \text{Mod}(\text{OWLDL}) = \{ A \models_\phi \mid A \in \text{Mod}(\text{OWL}) \land A \models \text{DLCONSTR} \} \)
The institution $\text{OWL}$

- signatures: theory morphisms $\text{OWL} \rightarrow (\Sigma, S)$
  ($\text{Sign}(\text{OWL})$ is a comma category)
- sentences: $\Sigma$-sentences
- models: $(\Sigma, S)$-models
- satisfaction: $A \models (\Sigma, S) S$ iff $A \models \Sigma S$

There is a forgetful morphism from $\text{OWL}$ to $\text{RDFS}$.
There is a simple theoroidal comorphism from $\text{OWL}$ to $\text{RDF++}$.
OWL: problems with amalgamation

can we amalgamate a PETS-model $A_1$ and BOOKS-model $A_2$ in a PETS+BOOKS model?

NO if $[[\text{owl: Thing}]]_{A_1} \neq [[\text{owl: Thing}]]_{A_2}$
OWL: problems with amalgamation

- solution: transform such a diagram into a pushout

\[ \begin{array}{ccc}
\text{PETS} & \rightarrow & \text{BOOKS} \\
\downarrow & & \downarrow \\
\varnothing & & \varnothing \\
\end{array} \]

we have to consider a \( \emptyset \)-model \( A_0 \)

- \( A_1 \) and \( A_2 \) are consistent iff \( A_1 \models \varphi_1 = A = A_2 \models \varphi_2 \)
SWRL

- proposed in November 2004
- extends OWL with Horn rules
- example: citation implies not self-citation

```xml
<ruleml:imp>
  <ruleml:_body>
    <swrlx:individualPropertyAtom swrlx:property="writtenBy">
      <ruleml:var>x1</ruleml:var>
      <ruleml:var>x2</ruleml:var>
    </swrlx:individualPropertyAtom>
    ...
  </ruleml:_body>
  <ruleml:_head>
    ...
  </ruleml:_head>
</ruleml:imp>
```
The institution $\text{SWRL}$

- signatures: OWL signatures
- sentences:
  \[
  \text{writtenBy}(x_1, x_2) \land \text{citedBy}(x_1, x_3) \rightarrow x_2 \neq x_3.
  \]
- models: OWL models
- satisfaction: as in OWL and HornLog

There is a forgetful morphism from $\text{SWRL}$ to $\text{OWL}$.

There is a simple theoroidal comorphism from $\text{SWRL}$ to $\text{RDF++}$. 
SWRL FOL

- proposed in November 2004
- extends OWL with first-order formulas
- example: any cited author has written a paper which is cited by someone else

<Assertion owlx:name="Example">
  <Forall>
    <ruleml:var>x1</ruleml:var>
    <Implies>
      <swrlx:classAtom owlx:name="CitedAuthor">
        <owlx:Class owlx:name="CitedAuthor" />
        <ruleml:var>x1</ruleml:var>
      </swrlx:classAtom>
      <Exists>
        ...
      </Exists>
    </Implies>
  </Forall>
</Assertion>
The institution **SWRLFOL**

- signatures: OWL signatures
- sentences:

\[(\forall x_1) \text{CitedAuthor}(x_1) \rightarrow (\exists x_2, x_3) \text{writtenBy}(x_2, x_1) \land \text{citedBy}(x_2, x_3)\]

- models: OWL models
- satisfaction: as in OWL and FOL

There is a forgetful morphism from **SWRLFOL** to **SWRL**.

There is a simple theoroidal comorphism from **SWRLFOL** to **RDF++**.
Relationships between SW logics

SWRLFOL $\rightarrow$ RDF++

SWRL $\rightarrow$ RDF++

OWL $\rightarrow$ RDF++

RDFS $\rightarrow$ RDF++
Conclusion

- contributions
  - RDF++, RDFS, OWL, SWRL, and SWRLFOL are in fact theories in RDF++
  - the institutions RDFS, OWL, SWRL, and SWRLFOL defined as particular theoroidal institutions
  - the relationships between these institutions

- advantages:
  - a rigurous and systematic approach of the logics underlying SW languages
  - an important step towards structuring and re-using ontology parts
  - a solid framework for relating SW languages with other formalisms and for proving the soundness of the reasoners
Questions?

Thank you!