Textbook (required)

Pascal Hitzler, Markus Krötzsch, Sebastian Rudolph

Foundations of Semantic Web Technologies

Chapman & Hall/CRC, 2010

Choice Magazine Outstanding Academic Title 2010 (one out of seven in Information & Computer Science)

http://www.semantic-web-book.org
Today: Description Logics
Today’s Session: DLs (towards OWL)

1. Basic Ideas
2. The Description Logic SROIQ(D)
3. Different Perspectives
4. Class Project
5. Class Presentations
OWL Building Blocks

- **individuals (written as URIs)**
  - also: constants (FOL), resources (RDF), instances
  - http://example.org/sebastianRudolph
  - we write these lowercase and abbreviated, e.g. "sebastianRudolph"

- **classes (also written as URIs!)**
  - also: concepts, unary predicates (FOL)
  - we write these uppercase, e.g. "Father"

- **properties (also written as URIs!)**
  - also: roles (DL), binary predicates (FOL)
  - we write these lowercase, e.g. "hasDaughter"
DL syntax                    RDFS syntax

• Person(mary)                       • :mary rdf:type :Person .

• Woman ⊆ Person                       • :Woman rdfs:subClassOf :Person .
  – Person ≡ HumanBeing (class equivalence):
     Person ⊆ HumanBeing AND
     HumanBeing ⊆ Person

• hasWife(john,mary)                     • :john :hasWife :mary .

• hasWife ⊆ hasSpouse                   • :hasWife rdfs:subPropertyOf :hasSpouse
  – hasSpouse ≡ marriedWith (property equivalence)
DL syntax  

- Person(mary)  
- Woman ⊆ Person  
  - Person ≡ HumanBeing (class equivalence)  
- hasWife(john, mary)  
- hasWife ⊆ hasSpouse  
  - hasSpouse ≡ marriedWith (property equivalence)

FOL syntax  

- Person(mary)  
- ∀x (Woman(x) → Person(x))  
- Person ≡ HumanBeing (class equivalence)  
- hasWife(john, mary)  
- ∀x ∀y (hasWife(x, y) → hasSpouse(x, y))  
- hasSpouse ≡ marriedWith (property equivalence)
Special classes and properties

- **owl:Thing** (RDF syntax)
  - DL-syntax: $\top$
  - contains everything
- **owl:Nothing** (RDF syntax)
  - DL-syntax: $\bot$
  - empty class
- **owl:topProperty** (RDF syntax)
  - DL-syntax: $U$
  - every pair is in $U$
- **owl:bottomProperty** (RDF syntax)
  - empty property
Class constructors

• conjunction
  – Mother ≡ Woman □ Parent
    „Mothers are exactly those who are women and parents.“

\[ ∀x (\text{Mother}(x) ⇔ \text{Woman}(x) \land \text{Parent}(x)) \]

• disjunction
  – Parent ≡ Mother ▪ Father
    „Parents are exactly those who are mothers or fathers.“

\[ ∀x (\text{Parent}(x) ⇔ \text{Mother}(x) \lor \text{Father}(x)) \]

• negation
  – ChildlessPerson ≡ Person □ ¬Parent
    „ChildlessPersons are exactly those who are persons and who are not parents.“

\[ ∀x (\text{ChildlessPerson}(x) ⇔ \text{Person}(x) \land \neg \text{Parent}(x)) \]
Class constructors

• existential quantification
  – only to be used with a role – also called a property restriction
  – Parent $\equiv \exists$hasChild.Person
    „Parents are exactly those who have at least one child which is a Person.“

\[
\forall x \ (\text{Parent}(x) \leftrightarrow \exists y \ (\text{hasChild}(x,y) \land \text{Person}(y)))
\]

• universal quantification
  – only to be used with a role – also called a property restriction
  – Person $\cap$ Happy $\equiv \forall$hasChild.Happy
    „A (person which is also happy) is exactly (something all children of which are happy).“

\[
\forall x \ (\text{Person}(x) \land \text{Happy}(x) \leftrightarrow \forall y \ (\text{hasChild}(x,y) \rightarrow \text{Happy}(y)))
\]

• Class constructors can be nested arbitrarily
Today’s Session: DLs (towards OWL)

1. Basic Ideas
2. The Description Logic SROIQ(D)
3. Different Perspectives
4. Class Project
5. Class Presentations
The description logic ALC

- **ABox expressions:**
  - Individual assignments: Father(john)
  - Property assignments: hasWife(john, mary)

- **TBox expressions:**
  - Subclass relationships: $\sqsubseteq$
  - Equivalence: $\equiv$
  - Conjunction: $\sqcap$
  - Disjunction: $\sqcup$
  - Negation: $\neg$
  - Property restrictions: $\forall$, $\exists$

**Complexity:** ExpTime

Also: $\top$, $\bot$
The Description Logic ALC

• Set of *individuals* a, b, c, ...
• Set of *atomic classes* (class names) A, B, ...
• Set of *role names* R, S, ...

• *(Complex) class expressions* are constructed as:

\[
C, D ::= A \mid \top \mid \bot \mid \neg C \mid C \cap D \mid C \cup D \mid \forall R.C \mid \exists R.C
\]

• A *TBox* is a set of statements of the form \( C \equiv \overline{D} \) or \( C \sqsubseteq D \), where C and D are class expressions. They are called *general inclusion axioms*.

• An *ABox* consists of statements of the form \( C(a) \) or \( R(a, b) \), where \( C \) is a class expression, \( R \) is a role, and \( a, b \) are individuals.
Human ⊑ ∃hasParent.Human

Orphan ⊑ Human ⊓ ∀hasParent.¬Alive

Orphan(harrypotter)

hasParent(harrypotter, jame southern)
Understanding SROIQ(D)

ALC + role chains = SR

- $\text{hasParent} \circ \text{hasBrother} \subseteq \text{hasUncle}$. 
  $$\forall x \forall y (\exists z ((\text{hasParent}(x,z) \land \text{hasBrother}(z,y)) \rightarrow \text{hasUncle}(x,y)))$$
  - includes top property and bottom property

- includes $S = \text{ALC} + \text{transitivity}$
  - $\text{hasAncestor} \circ \text{hasAncestor} \subseteq \text{hasAncestor}$

- includes $\text{SH} = S + \text{role hierarchies}$
  - $\text{hasFather} \subseteq \text{hasParent}$
Understanding SROIQ(D)

• O – nominals (closed classes)
  – MyBirthdayGuests ≡ \{bill, john, mary\}
  – Note the difference to
    MyBirthdayGuests(bill)
    MyBirthdayGuests(john)
    MyBirthdayGuests(mary)

• Individual equality and inequality (no unique name assumption!)
  – bill = john
    • \{bill\} ≡ \{john\}
  – bill ≠ john
    • \{bill\} ∩ \{john\} ≡ ⊥
Understanding SROIQ(D)

- **I** – inverse roles
  - hasParent \equiv \text{hasChild}^{-1}
  - Orphan \equiv \forall \text{hasChild}^{-1}.\text{Dead}

- **Q** – qualified cardinality restrictions
  - \leq 4 \text{hasChild}.\text{Parent}(\text{john})
  - HappyFather \equiv \geq 2 \text{hasChild}.\text{Female}
  - Car \subseteq (\leq 4 \text{hasTyre}. \top)

- Complexity SHIQ, SHOQ, SHIO: ExpTime.
  Complexity SHOIQ: NExpTime
  Complexity SROIQ: N^2\text{ExpTime}
Understanding SROIQ(D)

Properties can be declared to be

- **Transitive**
  hasAncestor \( R(a,b) \) and \( R(b,c) \) \( \rightarrow \) \( R(a,c) \)

- **Symmetric**
  hasSpouse \( R(a,b) \) \( \rightarrow \) \( R(b,a) \)

- **Asymmetric**
  hasChild \( R(a,b) \) \( \rightarrow \) not \( R(b,a) \)

- **Reflexive**
  hasRelative \( R(a,a) \) for all \( a \)

- **Irreflexive**
  parentOf not \( R(a,a) \) for any \( a \)

- **Functional**
  hasHusband \( R(a,b) \) and \( R(a,c) \) \( \rightarrow \) \( b=c \)

- **InverseFunctional**
  hasHusband \( R(a,b) \) and \( R(c,b) \) \( \rightarrow \) \( a=c \)

called *property characteristics*
(D) – datatypes

- so far, we have only seen properties with individuals in second argument, called \textit{object properties} or \textit{abstract roles} (DL)

- properties with datatype literals in second argument are called \textit{data properties} or \textit{concrete roles} (DL)

- In OWL allowed are many XML Schema datatypes, including \texttt{xsd:integer}, \texttt{xsd:string}, \texttt{xsd:float}, \texttt{xsd:boolean}, \texttt{xsd:anyURI}, \texttt{xsd:dateTime}

and also e.g. \texttt{owl:real}
Understanding SROIQ(D)

(D) – datatypes

- hasAge(john, "51"^^xsd:integer)

- additional use of constraining facets (from XML Schema)
  - e.g. Teenager $\equiv$ Person $\cap$ $\exists$hasAge.(xsd:integer: $\geq$12 and $\leq$19)

  note: this is not standard DL notation! It‘s really only used in OWL.
Understanding SROIQ(D)

further expressive features

• Self
  – PersonCommittingSuicide ≡ ∃kills.Self
• Keys (not really in SROIQ(D), but in OWL)
  – set of (object or data) properties whose values uniquely identify an object
• disjoint properties
  – Disjoint(hasParent,hasChild)
• explicit anonymous individuals
  – as in RDF: can be used instead of named individuals
SROIQ(D) constructors – overview

- ABox assignments of individuals to classes or properties
- ALC: $\subseteq, \equiv$ for classes
  $\cap, \cup, \neg, \exists, \forall$
  $\top, \bot$
- SR: + property chains, property characteristics, role hierarchies $\sqsubseteq$
- SRO: + nominals $\{o\}$
- SROI: + inverse properties
- SROIQ: + qualified cardinality constraints
- SROIQ(D): + datatypes (including facets)

- + top and bottom roles (for objects and datatypes)
- + disjoint properties
- + Self
- + Keys (not in SROIQ(D), but in OWL)
SROIQ(D) is essentially *(semantically)* the same as OWL.

Available in OWL (see later) as syntactic sugar for DL axioms:

- **disjoint classes**
  - `Apple ∩ Pear ⊆ ⊥`

- **disjoint union**
  - `Parent ≡ Mother ∪ Father`
  - `Mother ∩ Father ⊆ ⊥`

- **negative property assignments (also for datatypes)**
  - `¬hasAge(jack, "53"^^xsd:integer)`
Two Global Restrictions

- arbitrary property chain axioms lead to undecidability
- restriction: set of property chain axioms has to be regular
  - there must be a strict linear order $\prec$ on the properties
  - every property chain axiom has to have one of the following forms:
    - $R \circ R \sqsubseteq R$
    - $S^{-} \sqsubseteq R$
    - $S_1 \circ S_2 \circ \ldots \circ S_n \sqsubseteq R$
    - $R \circ S_1 \circ S_2 \circ \ldots \circ S_n \sqsubseteq R$
    - $S_1 \circ S_2 \circ \ldots \circ S_n \circ R \sqsubseteq R$
  - thereby, $S_i \prec R$ for all $i = 1, 2, \ldots, n$.

- Example 1: $R \circ S \sqsubseteq R$  $S \circ S \sqsubseteq S$  $R \circ S \circ R \sqsubseteq T$
  $\rightarrow$ regular with order $S \prec R \prec T$
- Example 2: $R \circ T \circ S \sqsubseteq T$
  $\rightarrow$ not regular because form not admissible
- Example 3: $R \circ S \sqsubseteq S$  $S \circ R \sqsubseteq R$
  $\rightarrow$ not regular because no adequate order exists
Two Global Restrictions

- combining property chain axioms and cardinality constraints may lead to undecidability

- restriction: use only *simple* properties in cardinality expressions (i.e. those which cannot be – directly or indirectly – inferred from property chains)

- technically:
  - for any property chain axiom $S_1 \circ S_2 \circ \ldots \circ S_n \subseteq R$ with $n>1$, $R$ is non-simple
  - for any subproperty axiom $S \subseteq R$ with $S$ non-simple, $R$ is non-simple
  - all other properties are simple

- Example: $Q \circ P \subseteq R$, $R \circ P \subseteq R$, $R \subseteq S$, $P \subseteq R$, $Q \subseteq S$
  - non-simple: $R$, $S$
  - simple: $P$, $Q$
Today’s Session: DLs (towards OWL)

1. Basic Ideas
2. The Description Logic SROIQ(D)
3. Different Perspectives
4. Class Project
5. Class Presentations
OWL – Extralogical Features

- OWL ontologies have URIs and can be referenced by others via
  - import statements
- Namespace declarations
- Entity declarations (must be done)
- Versioning information etc.

- Annotations
  - Entities and axioms (statements) can be endowed with annotations, e.g. using rdfs:comment.
  - OWL syntax provides *annotation properties* for this purpose.

Note: We still have to give a syntax for OWL – forthcoming.
The modal logic perspective

• Description logics can be understood from a modal logic perspective.

• Each pair of $\forall R$ and $\exists R$ statements give rise to a pair of modalities.

• Essentially, some description logics are multi-modal logics.
The RDFS perspective

RDFS semantics is weaker

- :mary rdf:type :Person .
- :Mother rdfs:subClassOf :Woman .
- :john :hasWife :Mary .
- :hasWife rdfs:subPropertyOf :hasSpouse
  - Person(mary)
  - Mother ⊆ Woman
  - hasWife(john,mary)
  - hasWife ⊆ hasSpouse

- :hasWife rdfs:range :Woman .
- :hasWife rdfs:domain :Man .
  - T ⊆ ∀hasWife.Woman
  - T ⊆ ∀hasWife⁻.Man or ∃hasWife.T ⊆ Man

RDFS also allows to

- make statements about statements → only possible through annotations in OWL (not present in SROID(D))
- mix class names, individual names, property names (they are all URIs) → punning in OWL
Punning

• Description logics impose \textit{type separation}, i.e. names of individuals, classes, and properties must be disjoint.

• In OWL 2 Full, type separation does not apply.

• In OWL 2 DL, type separation is relaxed, but a class \( X \) and an individual \( X \) are interpreted semantically as if they were different.

• \texttt{Father(john)}  
  \texttt{SocialRole(Father)}

• See further below on the two different types/semantics for OWL: OWL DL and OWL Full.
Today’s Session: DLs (towards OWL)

1. Basic Ideas
2. The Description Logic SROIQ(D)
3. Different Perspectives
4. Class Project
5. Class Presentations
Class project: next step

• none this time.
Today’s Session: DLs (towards OWL)

1. Basic Ideas
2. The Description Logic SROIQ(D)
3. Different Perspectives
4. Class Project
5. Class Presentations
Class presentations – first topics

• SPARQL 1.1 entailment regimes:
  http://www.w3.org/2009/sparql/docs/entailment/xmlspec.xml

• Aidan Hogan, Andreas Harth, Axel Polleres: SAOR: Authoritative Reasoning for the Web. ASWC 2008: 76-90

• Jacopo Urbani, Spyros Kotoulas, Jason Maassen, Frank van Harmelen, Henri E. Bal: OWL Reasoning with WebPIE: Calculating the Closure of 100 Billion Triples. ESWC (1) 2010: 213-227

• Yuan Ren, Jeff Z. Pan, Yuting Zhao: Soundness Preserving Approximation for TBox Reasoning. AAAI 2010

• Franz Baader, Sebastian Brandt, Carsten Lutz: Pushing the EL Envelope. IJCAI 2005: 364-369
Thursday 13\textsuperscript{th} of January: RDFS Part I
Tuesday 18\textsuperscript{th} of January: Exercise Session
Thursday 20\textsuperscript{th} of January: RDF and RDFS Semantics
Tuesday 25\textsuperscript{th} of January: RDF and RDFS Semantics
Thursday 27\textsuperscript{th} of January: Description Logics
Tuesday 8\textsuperscript{th} of February: Description Logic Semantics
Thursday 10\textsuperscript{th} of February: Exercises
Tuesday 15\textsuperscript{th} of February: OWL syntax

Estimated breakdown of sessions:
Intro + XML: 2          RDF: 4
OWL and Logic: 5        Class Presentations: 3
Exercise sessions: 3