Semantic Web Technologies

OWL DL and RDF Rules

Jos de Bruijn
debruijn AT inf.unibz.it

KRDB Research Group
Free University of Bolzano, Italy

07 November 2008

Outline

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)

OWL DL in RDF

- Mapping from OWL Abstract Syntax to RDF Graphs: given OWL ontology O, return RDF graph $T(O)$
- OWL DL
- Reuses RDF(S) vocabulary as much as possible

  e.g. `rdfs:subClassOf`, `rdf:type`
- Complete translation online at:
  `http://www.w3.org/TR/owl-semantics/mapping.html`
- We skip annotations here

Translating Ontologies

Abstract Syntax

Ontology(O $axiom_1$ ... $axiom_n$)

RDF graph

```
O rdf:type owl:Ontology .
T(axiom1) ... T(axiomn)
```

In case O is not present, a blank node is introduced.

Translating partial Class definition

Abstract Syntax

Class(classID partial $description_1$ ... $description_n$)

RDF graph

```
classID rdf:type owl:Class .
classID rdfs:subClassOf T(description1) . ...
classID rdfs:subClassOf T(descriptionn) .
```


Translating complete Class definition

Abstract Syntax

Class(classID complete $description_1$ ... $description_n$)

RDF graph

```
classID rdf:type owl:Class .
classID owl:intersectionOf T(SEQ description1 ...
  descriptionn) .
```


Translating complete Class definition

T(SEQ description1 ... descriptionn)

RDF graph

```
_:l1 rdf:type rdf:List . [opt]
_:l1 rdf:first T(item1) . _:l1 rdf:rest _:l2 .

...

_:ln rdf:type rdf:List . [opt]
_:ln rdf:first T(itemn) . _:ln rdf:rest rdf:nil .
```


Enumerated class

Abstract Syntax

EnumeratedClass(classID iID1 ... iIDn)

RDF graph

```
classID rdf:type owl:Class .
classID owl:oneOf T(SEQ iID1 ... iIDn) .
```


Disjoint classes

Abstract Syntax

DisjointClasses(description1 ... descriptionn)

RDF graph

```
T(descriptionj) owl:disjointWith T(descriptioni) .
```

$1 \leq i < j \leq n$

Translating Union

Abstract Syntax

unionOf(description1 ... descriptionn)

RDF graph

```
_:x rdf:type owl:Class .
_:x owl:unionOf T(SEQ description1 ...
  descriptionn) .
```

Translating Intersection

Abstract Syntax

intersectionOf(description1 ... descriptionn)

RDF graph

```
_:x rdf:type owl:Class .
_:x owl:intersectionOf T(SEQ description1 ...
  descriptionn) .
```


Translating Restrictions

Abstract Syntax

restriction(ID component1 ... componentn)

(With at least two components)

RDF graph

```
_:x rdf:type owl:Class .
_:x owl:intersectionOf
  T(SEQ(restriction(ID component1) ...
    restriction(ID componentn))) .
```


Translating Restrictions

Abstract Syntax

restriction(ID allValuesFrom(range))

RDF graph

```
_:x rdf:type owl:Restriction .
_:x owl:onProperty T(ID) .
_:x owl:allValuesFrom T(range) .
```


Translating Restrictions

Abstract Syntax

restriction(ID minCardinality(min))

RDF graph

```
_:x rdf:type owl:Restriction .
_:x owl:onProperty T(ID) .
_:x owl:minCardinality "min"^^xsd:nonNegativeInteger .
```


Translating Individuals

Abstract Syntax

Individual(iID
 type(type1) ... type(typen)
 value(pID1 v1) ... value(pIDk vk))

RDF graph

```
iID rdf:type T(type1) . ... iID rdf:type T(typen) .
iID T(pID1) T(v1) . ... iID T(pIDk) T(vk) .
```


RDF Rules: N3

- RDF Rules: extension of Turtle to **rules**
- `rdf:type` may be abbreviated to `a`
- Syntax:
  `{` (*triple pattern* `.`)* `}` `=>` `{` (*triple pattern* `.`)* `}` `.`
- Rule safety:
  - Variables on right-hand side of `=>` (**head**) must occur on left-hand side **body**
- Variables implicitly universally quantified over rule
- `=>` as standard implication in Logic Programming
- Rule with empty head is **integrity constraint**
- Rule with empty body is **fact**

RDF Rules: examples

```
{?A rdfs:subClassOf ?B. ?B rdfs:subClassOf ?A} =>
  {?A owl:equivalentClass ?B}.

{?D owl:complementOf ?C.
  ?D owl:equivalentClass owl:Nothing} =>
  {?C owl:equivalentClass owl:Thing}.

{?A owl:differentFrom ?A} => {}.
```


Rule-based OWL reasoning

- Cannot **exactly** capture OWL DL semantics
- OWL DL semantics can be approximated to large extent, e.g., equality:
  - Equality not really in the (rule) language
  - Can be simulated with `owl:sameAs`
- Hard to characterized which part of Semantics is captured
- Can capture part of OWL Full semantics
- Rule-based RDF reasoning popular on Semantic Web (e.g. Jena, CWM, Euler, ...)

Basic Ideas of DLs

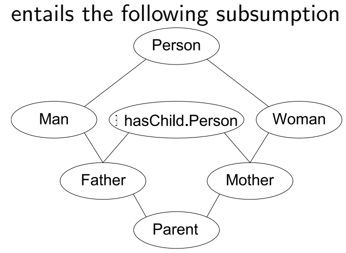
- Basic syntactic building blocks
  - Atomic concepts
  - Atomic roles
  - Individuals
- Limited constructs for building complex concepts, roles
- Implicit knowledge can be inferred automatically
  - Subsumption

Inference of implicit knowledge

Consider the TBox:

$$\begin{aligned} \mathit{Woman} &\equiv \mathit{Person} \sqcap \mathit{Female} \\ \mathit{Man} &\equiv \mathit{Person} \sqcap \neg \mathit{Woman} \\ \mathit{Mother} &\equiv \mathit{Woman} \sqcap \exists \mathit{hasChild}.\mathit{Person} \\ \mathit{Father} &\equiv \mathit{Man} \sqcap \exists \mathit{hasChild}.\mathit{Person} \\ \mathit{Parent} &\equiv \mathit{Mother} \sqcup \mathit{Father} \end{aligned}$$

entails the following subsumption hierarchy:



Review of DL Basics

- Set-based term descriptions
- Intentional knowledge
  - A DL ontology is not a database!
- Main inference procedure: subsumption reasoning
  - If a concept D is more general than a concept C, it **subsumes** that concept:
    $C \sqsubseteq D$
- Efficient TBox reasoning
- ABox reasoning (query answering) not well developed

OWL constructs

OWL Construct	DL	Example
intersectionOf	$C_1 \sqcap \ldots \sqcap C_n$	$\mathit{Human} \sqcap \mathit{Male}$
unionOf	$C_1 \sqcup \ldots \sqcup C_n$	$\mathit{Doctor} \sqcup \mathit{Lawyer}$
complementOf	$\neg C$	$\neg \mathit{Male}$
oneOf	$\{o_1, \ldots, o_n\}$	$\{\mathit{john}, \mathit{mary}\}$
allValuesFrom	$\forall P.C$	$\forall \mathit{hasChild}.\mathit{Doctor}$
someValuesFrom	$\exists P.C$	$\exists \mathit{hasChild}.\mathit{Lawyer}$
value	$\exists P.\{o\}$	$\exists \mathit{citizenOf}.\mathit{USA}$
minCardinality	$\geq nP.C$	$\geq 2\mathit{hasChild}.\mathit{Lawyer}$
maxCardinality	$\leq nP.C$	$\leq 1\mathit{hasChild}.\mathit{Male}$
cardinality	$= nP.C$	$= 1\mathit{hasParent}.\mathit{Female}$

+ XML Schema datatypes: int, string, real, etc...

OWL axioms

OWL Axiom	DL	Example
SubClassOf	$C_1 \sqsubseteq C_2$	$Human \sqsubseteq Animal \sqcap Biped$
EquivalentClasses	$C_1 \equiv \ldots \equiv C_n$	$Man \equiv Human \sqcap Male$
SubPropertyOf	$P_1 \sqsubseteq P_2$	$hasDaughter \sqsubseteq hasChild$
EquivalentProperties	$P_1 \equiv \ldots \equiv P_n$	$cost \equiv price$
SameIndividual	$o_1 = \ldots = o_n$	$President\_Bush = G\_W\_Bush$
DisjointClasses	$C_i \sqsubseteq \neg C_j$	$Male \sqsubseteq \neg Female$
DifferentIndividuals	$o_i \neq o_j$	$john \neq peter$
inverseOf	$P_1 \equiv P_2^-$	$hasChild \equiv hasParent^-$
Transitive	$P^+ \sqsubseteq P$	$ancestor^+ \sqsubseteq ancestor$
Symmetric	$P \equiv P^-$	$connectedTo \equiv connectedTo^-$

More examples

Class(firstYearCourse partial restriction (isTaughtBy allValuesFrom (Professor)))

Class(mathCourse partial restriction (isTaughtBy hasValue (949352)))

Class(academicStaffMember partial restriction (teaches someValuesFrom (undergraduateCourse)))

Class(course partial restriction (isTaughtBy minCardinality (1)))

Class(department partial restriction (hasMember minCardinality(10)) restriction (hasMember maxCardinality(30)))

More examples

In DL syntax:

$firstYearCourse \sqsubseteq \forall isTaughtBy.Professor$
$mathCourse \sqsubseteq \exists isTaughtBy.\{949352\}$
$academicStaffMember \sqsubseteq \exists teaches.undergraduateCourse$
$course \sqsubseteq \geq 1 isTaughtBy$
$department \sqsubseteq \geq 10 hasMember \sqcap \leq 30 hasMember$

More examples

Class(course partial complementOf(staffMember))

Class(peopleAtUni complete unionOf(staffMember student))

Class(facultyInCS complete intersectionOf (faculty restriction (belongsTo hasValue (CSDepartment))))

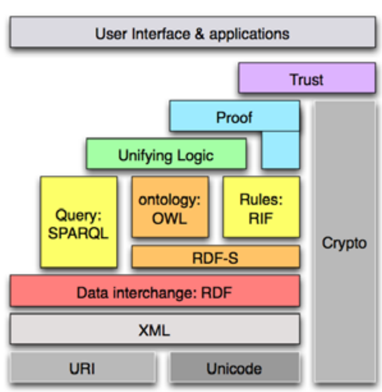
Class(adminStaff complete intersectionOf (staffMember complementOf(unionOf(faculty techSupportStaff))))

More examples

In DL syntax:

$course \sqsubseteq \neg staffMember$
$peopleAtUni \equiv staffMember \sqcup student$
$facultyInCS \equiv faculty \sqcap \exists belongsTo.\{CSDepartment\}$
$adminStaff \equiv staffMember \sqcap \neg(faculty \sqcup techSupportStaff)$

Semantic Web Languages



Why relate DL and LP?

Why not use DL or LP for everything?

- LP: Efficient algorithms and implementations for query answering
- DL: Efficient algorithms and implementations for subsumption reasoning
- Thus: it depends on the reasoning task which language to use
- Many existing implementations of rules systems (e.g., SQL1999 ≈ Datalog with linear recursion)
- Web Ontology Language (OWL) based on DL
- Thus: in order to leverage existing rule implementations for the Semantic Web, we need to translate!

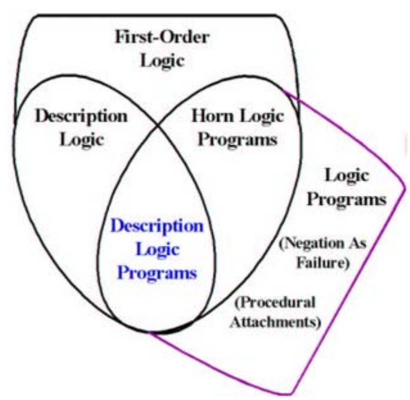
Recap: Logic Programming

- Any FOL term is a term in LP
- Any FOL atomic formula is an atomic formula in LP
- Any Horn formula is a rule in LP (quantification usually omitted)
  - $H \leftarrow B_1 \wedge \ldots \wedge B_n$
- Logic programming is a syntactic subset of FOL
- Note! Negation-as-failure in LP is an **extension** of Horn rules
  - $\neg \neq not$

Entailment

- General First-Order entailment:
  - $\phi \models \psi$ iff for every interpretation $\mathcal{I}$: if $\mathcal{I} \models \phi$ then $\mathcal{I} \models \psi$
  - Thus, the set of models of ϕ $M(\phi)$ is a subset of $M(\psi)$: $M(\phi) \subseteq M(\psi)$
  - e.g., $p(x) \wedge q(x) \models p(x)$
- Ground entailment:
  - $\phi \models \psi_{ground}$ iff for every interpretation $\mathcal{I}$: if $\mathcal{I} \models \phi$ then $\mathcal{I} \models \psi_{ground}$ and $\psi ground$ **does not** contain variables
  - e.g., $(p(x) \rightarrow q(x)) \wedge p(a) \models q(a)$
- Logic Programming only defines ground entailment
- Horn Logic (i.e., Horn subset of FOL) is equivalent to Logic Programming wrt. ground entailment
  - For any set of Horn formulas ϕ and a ground Horn formula ψ_{ground}: $\phi \models_{FOL} \psi_{ground}$ iff $\phi \models_{LP} \psi_{ground}$
  - $\models_{FOL}$ is classical First-Order entailment; $\models_{LP}$ is LP entailment

Relation between DL and LP



Description Logic Programs

- "Intersection" of Description Logics and Logic Programming
- That part of Description Logics (OWL in particular) which can be translated to a Logic Program
- Horn Logic subset of $\mathcal{SHOIN}$, **reduced** to a Logic Program: Description Logic Program: DLP
- General idea:
  1. Translate $\mathcal{SHOIN}$ axiom to First-Order Logic
  2. Rewrite to Horn Logic
     - If rewriting not possible: formula not in DLP
  3. Reduce to Logic Program

Recap: $\mathcal{SHOIN}$

Concept descriptions

$C, D \longrightarrow A \mid$	(atomic concept)
$\top \mid$	(universal concept)
$\bot \mid$	(bottom concept)
$C \sqcap D \mid$	(intersection)
$C \sqcup D \mid$	(disjunction)
$\neg C \mid$	(negation)
$\forall R.C \mid$	(value restriction)
$\exists R.C \mid$	(existential quantification)
$\{o_1, \ldots, o_n\} \mid$	(enumeration)
$\exists R.\{o\} \mid$	(hasValue)
$\geqslant nR \mid$	(minimal cardinality)
$\leqslant nR \mid$	(maximal cardinality)

Recap: $\mathcal{SHOIN}$

Individual assertions

$a \in C$
$\langle a, b \rangle \in R$

Recap: $\mathcal{SHOIN}$

Axioms

$C \sqsubseteq D$	(class subsumption)
$C \equiv D$	(equivalence)
$Q \sqsubseteq R$	(property subsumption)
$R \equiv Q^-$	(inverse roles)
$R \equiv R^-$	(symmetric roles)
$R^+ \sqsubseteq R$	(transitive properties)

Mapping $\mathcal{SHOIN}$ to FOL

A (atomic concept)	$A(x)$
$\top$	$\top$
$\bot$	$\bot$
$C \sqcap D$	$tr(C) \wedge tr(D)$
$C \sqcup D$	$tr(C) \vee tr(C)$
$\neg C$	$\neg tr(C)$
$\forall R.C$	$\forall y : R(x, y) \rightarrow tr(C, y)$
$\exists R.C$	$\exists y : R(x, y) \wedge tr(C, y)$
$\{o_1, ..., o_n\}$	$x = o_1 \vee ... \vee x = o_n$
$\exists R.\{o\}$	$R(x, o)$
$\geqslant nR$	$\exists y_1, \ldots, y_n : \bigwedge R(X, y_i) \wedge \bigwedge y_i \neq y_j$
$\leqslant nR$	$\forall y_1, \ldots, y_{n+1} : \bigwedge R(X, y_i) \rightarrow \bigvee y_i = y_j$

Mapping $\mathcal{SHOIN}$ to FOL

$a \in A$	$A(a)$
$\langle a, b \rangle \in R$	$R(a, b)$

$C \sqsubseteq D$	$\forall x : tr(C, x) \rightarrow tr(D, x)$
$C \equiv D$	$\forall x : tr(C, x) \leftrightarrow tr(D, x)$
$Q \sqsubseteq R$	$\forall x, y : Q(r, y) \rightarrow R(x, y)$
$R \equiv Q^-$	$\forall x, y : R(x, y) \leftrightarrow Q(y, x)$
$R^+ \sqsubseteq R$	$\forall x, y, z : R(x, y) \wedge R(y, z) \rightarrow R(x, z)$

Identifying DLP (1)

- Next step: identify DLP-fragment of $\mathcal{SHOIN}$
- Start with the axioms
  - Easy case: properties

$Q \sqsubseteq R$	$\forall x, y : Q(r, y) \rightarrow R(x, y)$
$R \equiv Q^-$	$\forall x, y : R(x, y) \leftrightarrow Q(y, x)$
$R^+ \sqsubseteq R$	$\forall x, y, z : R(x, y) \wedge R(y, z) \rightarrow R(x, z)$

    All Horn Logic!
- Equivalence axioms: $C \equiv D$
  Reduce to: $C \sqsubseteq D, D \sqsubseteq C$
- Subsumption axioms: $C \sqsubseteq D \mapsto \forall x : tr(C, x) \rightarrow tr(D, x)$

  $tr(C, x)$ is the body of the rule
  $tr(D, x)$ is the head of the rule

Identifying DLP (2)

- $C \sqsubseteq D$
  - **Remember:** the **body** is a conjunction of literals; the **head** is a single literal
  - Thus: C and D look different
  - C is called the **left-hand side** (lhs)
  - D is called the **right-hand side** (rhs)
- We distinguish between:
  - Descriptions allowed on the left-hand side
  - Descriptions allowed on the right-hand side

Identifying DLP (3)

$\mathcal{SHOIN}$	FOL	lhs	rhs
A (atomic concept)	$A(x)$	+	+
$\top$	$\top$	+/-	+
$\bot$	$\bot$	+	-
$C \sqcap D$	$tr(C) \wedge tr(D)$	+	+
$C \sqcup D$	$tr(C) \vee tr(C)$	+	-
$\neg C$	$\neg tr(C)$	-	-
$\forall R.C$	$\forall y : R(x,y) \rightarrow tr(C,y)$	-	+
$\exists R.C$	$\exists y : R(x,y) \wedge tr(C,y)$	+	-
$\{o_1, ..., o_n\}$	$x = o_1 \vee ... \vee x = o_n$	+	-
$\exists R.\{o\}$	$R(x,o)$	+	+
$\geqslant nR$	$\exists y_1, \ldots, y_n : \bigwedge R(X, y_i) \wedge \bigwedge y_i \neq y_j$	+	-
$\leqslant nR$	$\forall y_1, \ldots, y_{n+1} : \bigwedge R(X, y_i) \rightarrow \bigvee y_i = y_j$	-	-

Identifying DLP (4)

- Individual assertions:

$a \in A$	$A(a)$
$\langle a, b \rangle \in R$	$R(a,b)$

  Trivially in DLP!
- Example: $\top \sqcap \exists P.A \sqcap \{o_1, o_2\} \sqcap B \sqsubseteq \forall P.B$

  Step 1: rewrite to FOL:
  $\forall x : \top \wedge (\exists y : P(x,y) \wedge A(y)) \wedge (x = o_1 \vee x = o_2) \wedge B(x) \rightarrow (\forall y : P(x,y) \wedge B(y))$

  Step 2: rewrite to Horn Logic:
  1:$\forall x : \top \wedge (\exists y : P(x,y) \wedge A(y)) \wedge (x = o_1 \vee x = o_2) \wedge B(x) \rightarrow (\forall y : P(x,y) \wedge B(y)) \Leftrightarrow$

Example (cont'd)

2:$\forall x : \neg\top \vee \neg(\exists y : P(x,y) \wedge A(y)) \vee \neg(x = o_1 \vee x = o_2) \vee \neg B(x) \vee (\forall y : P(x,y) \wedge B(y)) \Leftrightarrow$

3:$\forall x : \neg(\exists y : P(x,y) \wedge A(y)) \vee \neg(x = o_1 \vee x = o_2) \vee \neg B(x) \vee (\forall y : P(x,y) \wedge B(y)) \Leftrightarrow$

4:$\forall x : (\forall y : \neg(P(x,y) \wedge A(y))) \vee \neg(x = o_1 \vee x = o_2) \vee \neg B(x) \vee (\forall y : P(x,y) \wedge B(y)) \Leftrightarrow$

5:$\forall x, y : \neg P(x,y) \vee \neg A(y) \vee \neg(x = o_1 \vee x = o_2) \vee \neg B(x) \vee (\forall y : P(x,y) \wedge B(y)) \Leftrightarrow$

Example (cont'd)

6:$\forall x, y : \neg P(x,y) \vee \neg A(y) \vee (\neg x = o_1 \wedge \neg x = o_2) \vee \neg B(x) \vee (\forall y : P(x,y) \wedge B(y)) \Leftrightarrow$

7:$\forall x, y, z : \neg P(x,y) \vee \neg A(y) \vee (\neg x = o_1 \wedge \neg x = o_2) \vee \neg B(x) \vee (P(x,z) \wedge B(z)) \Leftrightarrow$

8:$\forall x, y, z : \neg P(x,y) \vee \neg A(y) \vee \neg x = o_1 \vee \neg B(x) \vee (P(x,z) \wedge B(z))$
$\forall x, y, z : \neg P(x,y) \vee \neg A(y) \vee \neg x = o_2 \vee \neg B(x) \vee (P(x,z) \wedge B(z)) \Leftrightarrow$

Example (cont'd)

9:$\forall x, y, z : \neg P(x,y) \vee \neg A(y) \vee \neg x = o_1 \vee \neg B(x) \vee P(x,z)$
$\forall x, y, z : \neg P(x,y) \vee \neg A(y) \vee \neg x = o_1 \vee \neg B(x) \vee B(z)$
$\forall x, y, z : \neg P(x,y) \vee \neg A(y) \vee \neg x = o_2 \vee \neg B(x) \vee P(x,z)$
$\forall x, y, z : \neg P(x,y) \vee \neg A(y) \vee \neg x = o_2 \vee \neg B(x) \vee B(z)$

Step 3: Reduce to Logic Program

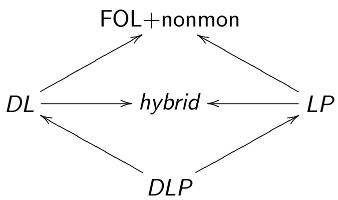
```
P(x,z) :- P(x,y), A(y), x=o1, B(x).
B(z)   :- P(x,y), A(y), x=o1, B(x).
P(x,z) :- P(x,y), A(y), x=o2, B(x).
B(z)   :- P(x,y), A(y), x=o2, B(x).
```

Semantic reduction: only entailment of ground atoms

Integration of DL and LP beyond DLP

- DLP has limited expressiveness
  - No disjunction
  - No existentials
  - No negation
  - No chaining variables over predicates
- DLP allows **extension** to DL or LP, no real **interoperation**

Outlook: Integration beyond DLP



- Extension of DLP
  - Interoperation only for inexpressive ontologies
- Unified language
  - Obviously undecidable
  - Little research has been done
- Hybrid integration
  - Full DL and LP
  - Interaction limited to retain decidability
  - Several existing approaches (e.g. [Eiter, 2004; Rosati, 2006])