



TECHNISCHE
UNIVERSITÄT
DRESDEN

FOUNDATIONS OF SEMANTIC WEB TECHNOLOGIES

OWL 2 – Syntax and Semantics

Sebastian Rudolph

Dresden, 10 May 2013

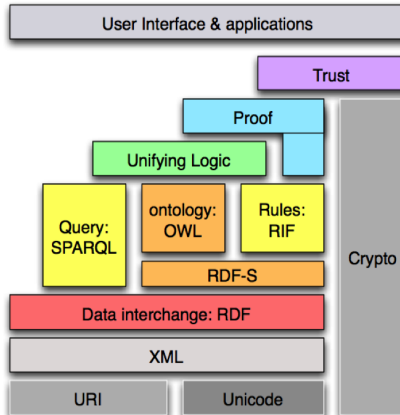


DRESDEN
UNIVERSITÄT
TECHNISCHE UNIVERSITÄT
DRESDEN

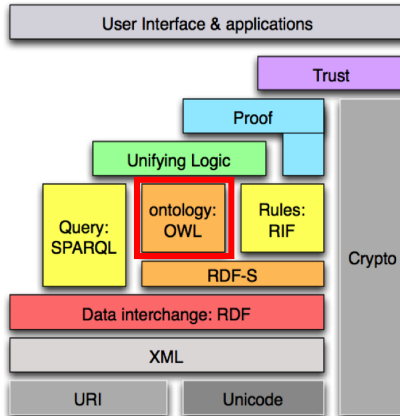
Content

Overview & XML	9 APR DS2	Hypertableau II	7 JUN DS5
Introduction into RDF	9 APR DS3	Tutorial 5	11 JUN DS2
RDFS – Syntax & Intuition	12 APR DS5	SPARQL Syntax & Intuition	11 JUN DS3
RDFS – Semantics	16 APR DS2	SPARQL – Semantics	14 JUN DS2
RDFS Rule-based Reasoning	16 APR DS3	SPARQL 1.1	14 JUN DS3
Tutorial 1	19 APR DS5	SPARQL Entailment	14 JUN DS5
OWL – Syntax & Intuition	26 APR DS5	Tutorial 6	18 JUN DS2
Tutorial 2	3 MAY DS5	SPARQL Implementation	18 JUN DS3
OWL – Syntax & Intuition ctd.	7 MAY DS2	Ontology Editing	2 JUL DS2
OWL & Description Logics	7 MAY DS3	Ontology Engineering	2 JUL DS3
OWL 2	10 MAY DS5	Tutorial 7	9 JUL DS2
Tutorial 3	14 MAY DS2	Linked Data	9 JUL DS3
Tableau I	14 MAY DS3	Applications	12 JUL DS5
Tableau II	17 MAY DS5	Test Exam	16 JUL DS2
Tutorial 4	7 JUN DS2	Test Exam Evaluation	16 JUL DS3
Hypertableau I	7 JUN DS3	Q&A Session	19 JUL DS5

OWL 2



OWL 2



Agenda

- Recap OWL & Overview OWL 2
- The Description Logic *SROIQ*
- Inferencing with *SROIQ*
- OWL 2 DL
- OWL 2 Profiles
- OWL 2 Full
- Summary

Agenda

- Recap OWL & Overview OWL 2
- The Description Logic *SROIQ*
- Inferencing with *SROIQ*
- OWL 2 DL
- OWL 2 Profiles
- OWL 2 Full
- Summary

Insufficiencies of OWL

OWL still too weak for certain tasks

Insufficiencies of OWL

OWL still too weak for certain tasks

- OWL insufficient as query language
 ↪ conjunctive queries, SPARQL for OWL

Insufficiencies of OWL

OWL still too weak for certain tasks

- OWL insufficient as query language
↪ conjunctive queries, SPARQL for OWL
- OWL insufficient as ontology language
↪ FOL-based rule extensions, SWRL & RIF

Insufficiencies of OWL

OWL still too weak for certain tasks

- OWL insufficient as query language
 ↪ conjunctive queries, SPARQL for OWL
- OWL insufficient as ontology language
 ↪ FOL-based rule extensions, SWRL & RIF

Should the OWL standard itself be extended?

Insufficiencies of OWL

OWL still too weak for certain tasks

- OWL insufficient as query language
↪ conjunctive queries, SPARQL for OWL
- OWL insufficient as ontology language
↪ FOL-based rule extensions, SWRL & RIF

Should the OWL standard itself be extended?

↪ OWL 2

Development of OWL 2

OWL 2 as “next version” of OWL

extensions due to practical experiences with OWL 1.0:

- additional expressivity due to new ontological axioms
- extralogical extensions (syntax, metadata, ...)
- revision of the OWL variants (Lite/DL/Full)

goals:

- most extensive compatibility with the existing OWL standard
- preservation of decidability of OWL DL
- correction of problems in the OWL 1.0 standard

Agenda

- Recap OWL & Overview OWL 2
- The Description Logic *SROIQ*
- Inferencing with *SROIQ*
- OWL 2 DL
- OWL 2 Profiles
- OWL 2 Full
- Summary

From *SHOIN* to *SROIQ*

OWL DL based on DL *SHOIN(D)*:

- axioms:
 - TBox: subclass relationships $C \sqsubseteq D$
 - RBox: subrole relationships $R \sqsubseteq S$ (\mathcal{H}), inverse roles R^- (\mathcal{I}), transitivity
 - ABox: class assertions $C(a)$, role assertions $R(a, b)$, equality $a \approx b$, inequality $a \not\approx b$
- class constructors:
 - conjunction $C \sqcap D$, disjunction $C \sqcup D$, negation $\neg C$ of classes
 - role restrictions: universal $\forall R.C$ and existential $\exists R.C$
 - number restrictions (\mathcal{N}): $\leq n R$ and $\geq n R$ (n non-negative integer)
 - nominals (\mathcal{O}): $\{a\}$
- datatypes (D)

OWL 2 extends this to *SROIQ(D)*

ABox

SHOIN supports different ABox assertions:

- class membership $C(a)$ (C complex class),
- special case: negated class membership $\neg C(a)$ (C complex class),
- equality $a \approx b$,
- inequality $a \not\approx b$
- role membership $R(a, b)$

ABox

SHOIN supports different ABox assertions:

- class membership $C(a)$ (C complex class),
- special case: negated class membership $\neg C(a)$ (C complex class),
- equality $a \approx b$,
- inequality $a \not\approx b$
- role membership $R(a, b)$
- negated role membership?

ABox

SHOIN supports different ABox assertions:

- class membership $C(a)$ (C complex class),
- special case: negated class membership $\neg C(a)$ (C complex class),
- equality $a \approx b$,
- inequality $a \not\approx b$
- role membership $R(a, b)$
- negated role membership?

\rightsquigarrow *SROIQ* allows **negated roles** in der ABox: $\neg R(a, b)$

Number Restrictions

SHOIN supports only unqualified number restrictions (\mathcal{N}):

Person $\sqcap \geq 3$ hasChild

"class of all persons with 3 or more children"

Number Restrictions

SHOIN supports only unqualified number restrictions (\mathcal{N}):

Person $\sqcap \geq 3$ hasChild

"class of all persons with 3 or more children" \rightsquigarrow *SROIQ* also allows **qualified**

number restrictions (\mathcal{Q}):

Person $\sqcap \geq 3$ hasChild.(Woman \sqcap Professor)

"class of all persons with 3 or more daughters who are professors"

The Self “Concept”

modeling task: “Every human knows himself/herself.”

The Self “Concept”

modeling task: “Every human knows himself/herself.”

- *SHOIN*:

knows(tom, tom) knows(tina, tina) knows(udo, udo) ...

The Self “Concept”

modeling task: “Every human knows himself/herself.”

- *SHOIN*:

knows(tom, tom) knows(tina, tina) knows(udo, udo) ...

↪ not generally applicable

- *SROIQ*: specific notation **Self**

Human $\sqsubseteq \exists \text{knows.Self}$

Role Axioms in *SHOIN*

SHOIN provides few role axioms:

- $\text{Trans}(r)$, `owl:TransitiveProperty`: r is transitive
Example: `Trans(locatedIn)`

Role Axioms in *SHOIN*

SHOIN provides few role axioms:

- $\text{Trans}(r)$, `owl:TransitiveProperty`: r is **transitive**
Example: `Trans(locatedIn)`
- $\text{Sym}(r)$, `owl:SymmetricProperty`: r is **symmetric**
Example: `Sym(relativeOf)`
also: $r \sqsubseteq r^{-}$

Role Axioms in *SHOIN*

SHOIN provides few role axioms:

- $\text{Trans}(r)$, `owl:TransitiveProperty`: r is **transitive**
Example: `Trans(locatedIn)`
- $\text{Sym}(r)$, `owl:SymmetricProperty`: r is **symmetric**
Example: `Sym(relativeOf)`
also: $r \sqsubseteq r^{-}$
- $\text{Func}(r)$, `owl:FunctionalProperty`: r is **functional**
Example: `Func(hasFather)`
also: $\top \sqsubseteq \leq 1r$

Role Axioms in *SHOIN*

SHOIN provides few role axioms:

- $\text{Trans}(r)$, `owl:TransitiveProperty`: r is **transitive**
Example: `Trans(locatedIn)`
- $\text{Sym}(r)$, `owl:SymmetricProperty`: r is **symmetric**
Example: `Sym(relativeOf)`
also: $r \sqsubseteq r^{-}$
- $\text{Func}(r)$, `owl:FunctionalProperty`: r is **functional**
Example: `Func(hasFather)`
also: $\top \sqsubseteq \leq 1r$
- $\text{InvFunc}(r)$, `owl:InverseFunctionalProperty`: r is **inverse functional**
Example: `InvFunc(isFatherOf)`
also $\top \sqsubseteq \leq 1r^{-}$ or `Func(r^{-})`

Role Axioms in *SROIQ*

SROIQ provides additional statements about roles:

- $\text{Ref}(r)$, `owl:ReflexiveProperty`: r is **reflexive**, $(x, x) \in r^{\mathcal{I}}$ for all domain individuals x
Example: $\text{Ref}(\text{knows})$

Role Axioms in *SROIQ*

SROIQ provides additional statements about roles:

- $\text{Ref}(r)$, `owl:ReflexiveProperty`: r is **reflexive**, $(x, x) \in r^{\mathcal{I}}$ for all domain individuals x
Example: $\text{Ref}(\text{knows})$
- $\text{Irr}(r)$, `owl:IrreflexiveProperty`: r is **irreflexive**, $(x, x) \notin r^{\mathcal{I}}$ for all domain individuals x
Example: $\text{Irr}(\text{hasChild})$

Role Axioms in *SROIQ*

SROIQ provides additional statements about roles:

- $\text{Ref}(r)$, `owl:ReflexiveProperty`: r is **reflexive**, $(x, x) \in r^{\mathcal{I}}$ for all domain individuals x
Example: $\text{Ref}(\text{knows})$
- $\text{Irr}(r)$, `owl:IrreflexiveProperty`: r is **irreflexive**, $(x, x) \notin r^{\mathcal{I}}$ for all domain individuals x
Example: $\text{Irr}(\text{hasChild})$
- $\text{Asym}(r)$, `owl:AsymmetricProperty`: r is **asymmetric**, $(x, y) \in r^{\mathcal{I}}$ implies $(y, x) \notin r^{\mathcal{I}}$
Example: $\text{Asym}(\text{hasChild})$

Role Axioms in *SROIQ*

SROIQ provides additional statements about roles:

- $\text{Ref}(r)$, `owl:ReflexiveProperty`: r is **reflexive**, $(x, x) \in r^{\mathcal{I}}$ for all domain individuals x
Example: $\text{Ref}(\text{knows})$
- $\text{Irr}(r)$, `owl:IrreflexiveProperty`: r is **irreflexive**, $(x, x) \notin r^{\mathcal{I}}$ for all domain individuals x
Example: $\text{Irr}(\text{hasChild})$
- $\text{Asym}(r)$, `owl:AsymmetricProperty`: r is **asymmetric**, $(x, y) \in r^{\mathcal{I}}$ implies $(y, x) \notin r^{\mathcal{I}}$
Example: $\text{Asym}(\text{hasChild})$
- $\text{Dis}(r, s)$, `owl:propertyDisjointWith, , owl:AllDisjointProperties`: r and s are **disjoint**, $(x, y) \notin r^{\mathcal{I}} \cap s^{\mathcal{I}}$ for all x, y
Example: $\text{Dis}(\text{hasFather}, \text{hasSon})$

The Universal Role

SROIQ provides the universal role:

- **universal role** U (`owl:TopObjectProperty`):
 $(x, y) \in U^{\mathcal{I}}$ for all x, y

Example

$\top \sqsubseteq \leq 7\,000\,000\,000 \ U.Human$
(not recommended!)

- ↔ U is mainly comfortable as a counterpart for \top , e.g. as root element in a graphically displayed role hierarchy
- the converse `owl:BottomObjectProperty` has been introduced in OWL, but has no corresponding syntactic element in DLs
- for datatype properties analog `owl:TopDataProperty` and `owl:BottomDataProperty`

Complex Role Inclusion

"The friends of my friends are my friends."

↪ can be expressed in *SHOIN*:
hasFriend is transitive

"The enemies of my friends are my enemies."

↪ Cannot be expressed in *SHOIN*!

Complex Role Inclusion

"The friends of my friends are my friends."

↪ can be expressed in *SHOIN*:
hasFriend is transitive

"The enemies of my friends are my enemies."

↪ Cannot be expressed in *SHOIN*!

complex role inclusion

- RBox-expressions of the form $r_1 \circ r_2 \circ \dots \circ r_n \sqsubseteq s$
- semantics: $(x_0, x_1) \in r_1^{\mathcal{I}}, (x_1, x_2) \in r_2^{\mathcal{I}}, \dots, (x_{n-1}, x_n) \in r_n^{\mathcal{I}}$,
implies $(x_0, x_n) \in s^{\mathcal{I}}$

Complex Role Inclusions – Example

Example

$\text{hasFriend} \circ \text{hasEnemy} \sqsubseteq \text{hasEnemy}$:
if $(x, y) \in \text{hasFriend}^{\mathcal{I}}$ and $(y, z) \in \text{hasEnemy}^{\mathcal{I}}$,
then also holds $(x, z) \in \text{hasEnemy}^{\mathcal{I}}$

further examples

$\text{partOf} \circ \text{belongsTo} \sqsubseteq \text{belongsTo}$

$\text{hasBrother} \circ \text{hasChild} \sqsubseteq \text{isUncleOf}$

Expressivity of Complex Role Inclusions

How complicated are complex role inclusions?

RBoxes allow for encoding formal languages:

grammar for language of words $ab, aabb, aaabbb, \dots$:

$L ::= ab$
 $L ::= aLb$

becomes the following RBox

$r_a \circ r_b \sqsubseteq \ell$
 $r_a \circ \ell \circ r_b \sqsubseteq \ell$

In fact, this way, all context-free languages can be encoded. This even enables us to encode the emptiness problem for intersection of two context-free languages into KB satisfiability.

\rightsquigarrow OWL with (unrestricted) role inclusions is undecidable.

Regular RBoxes

Can complex role inclusion be restricted in order to retain decidability?

- RBoxes correspond to grammars for context-free languages
 - intersection of these problematic
- ↪ restriction to regular languages!

Regularity Conditions for RIAs

in order to guarantee decidability of inferencing, the set of role inclusions has to be **regular**

- there has to be a strict linear order \prec over the roles such that every RIA has one of the following forms (with $s_i \prec r$ for all $1 \leq i \leq n$):
 - $r \circ r \sqsubseteq r$
 - $r^- \sqsubseteq r$
 - $s_1 \circ s_2 \circ \dots \circ s_n \sqsubseteq r$
 - $r \circ s_1 \circ s_2 \circ \dots \circ s_n \sqsubseteq r$
 - $s_1 \circ s_2 \circ \dots \circ s_n \circ r \sqsubseteq r$

Regularity Conditions for RIAs

- Example 1: $ros \sqsubseteq r \quad sos \sqsubseteq s \quad rosor \sqsubseteq t$

Regularity Conditions for RIAs

- Example 1: $r \circ s \sqsubseteq r \quad s \circ s \sqsubseteq s \quad r \circ s \circ r \sqsubseteq t$
 \rightsquigarrow regular with order $s \prec r \prec t$
- Example 2: $r \circ t \circ s \sqsubseteq t$

Regularity Conditions for RIAs

- Example 1: $r \circ s \sqsubseteq r \quad s \circ s \sqsubseteq s \quad r \circ s \circ r \sqsubseteq t$
 \rightsquigarrow regular with order $s \prec r \prec t$
- Example 2: $r \circ t \circ s \sqsubseteq t$
 \rightsquigarrow not regular, form not allowed
- Example 3: $r \circ s \sqsubseteq s \quad s \circ r \sqsubseteq r$

Regularity Conditions for RIAs

- Example 1: $r \circ s \sqsubseteq r \quad s \circ s \sqsubseteq s \quad r \circ s \circ r \sqsubseteq t$
 \rightsquigarrow regular with order $s \prec r \prec t$
- Example 2: $r \circ t \circ s \sqsubseteq t$
 \rightsquigarrow not regular, form not allowed
- Example 3: $r \circ s \sqsubseteq s \quad s \circ r \sqsubseteq r$
 \rightsquigarrow not regular, since no appropriate order exists

Revisiting the Definition of Simple Roles

- simple roles in *SHOIN* = roles without transitive subroles
- in *SROIQ* we need to take RIAs into account

Revisiting the Definition of Simple Roles

simple roles are all roles. . .

- that do not occur on the rhs of a role inclusion,
- that are inverses of other simple roles,
- that occur only on the rhs of RIAs where the lhs consists of a length-one chain with a simple role.

(Caution: inductive definition)

↔ non-simple are roles that can be derived from a chain of roles with length at least 2

Revisiting the Definition of Simple Roles

simple roles are all roles. . .

- that do not occur on the rhs of a role inclusion,
- that are inverses of other simple roles,
- that occur only on the rhs of RIAs where the lhs consists of a length-one chain with a simple role.

(Caution: inductive definition)

↪ non-simple are roles that can be derived from a chain of roles with length at least 2

Expressions $\leq_n r.C$, $\geq_n r.C$, $\text{lrr}(r)$, $\text{Dis}(r, s)$, $\exists r.\text{Self}$, $\neg r(a, b)$
are only allowed for simple roles r and s !

(Reason: ensure decidability)

Overview *SRDIQ* – TBoxes

class expressions

class names	A, B
conjunction	$C \sqcap D$
disjunction	$C \sqcup D$
negation	$\neg C$
existential role restriction	$\exists r.C$
universal role restriction	$\forall r.C$
Self	$\exists s.\text{Self}$
atleast restriction	$\geq n s.C$
atmost restriction	$\leq n s.C$
nominals	$\{a\}$

TBox (class axioms)

inclusion	$C \sqsubseteq D$
equivalence	$C \equiv D$

Overview *SROIQ* – RBoxes & ABoxes

Roles

roles	r, s, t
simple roles	s, t
universal role	u

ABox (assertions)

class membership	$C(a)$
role membership	$r(a, b)$
neg. role membership	$\neg s(a, b)$
equality	$a \approx b$
inequality	$a \not\approx b$

RBox (role axioms)

inclusion	$r_1 \sqsubseteq r_2$
complex role inclusion	$r_1 \circ \dots \circ r_n \sqsubseteq r$
transitivity	$\text{Trans}(r)$
symmetry	$\text{Sym}(r)$
reflexivity	$\text{Ref}(r)$
irreflexivity	$\text{Irr}(s)$
disjointness	$\text{Dis}(s, t)$

Agenda

- Recap OWL & Overview OWL 2
- The Description Logic *SROIQ*
- Inferencing with *SROIQ*
- OWL 2 DL
- OWL 2 Profiles
- OWL 2 Full
- Summary

How complicated is *SROIQ*?

recap: *SHOIN* (OWL DL) is very complex (NExpTime)

How complicated is *SHOIQ*?

recap: *SHOIN* (OWL DL) is very complex (NExpTime)

observation: some modeling features are not really necessary (“syntactic sugar”)

- $\text{Trans}(r)$ can be expressed as $r \circ r \sqsubseteq r$
- $\text{Sym}(r)$ can be expressed as $r^- \sqsubseteq r$
- $\text{Asym}(r)$ can be expressed as $\text{Dis}(r, r^-)$
- $\text{Irr}(s)$ can be expressed as $\top \sqsubseteq \neg \exists S.\text{Self}$
- ABox can be represented by TBox axioms with nominals, e.g. $r(a, b)$ becomes $\{a\} \sqsubseteq \exists r.\{b\}$

qualifizierte number restrictions do not cause problems (known and implemented before)

↪ main problem: role axioms (RBox)

Role Inclusions , Languages, Automata

How to deal with RBoxes?

- RBox inclusions resemble formal grammars
- every role r defines a regular language:
the language of role chains from which it follows
- regular languages \equiv regular Expressions \equiv finite automata

↔ approach: tableau methods are extended by "RBox automata"

Decidability of *SROIQ*

tableau method for *SROIQ* shows decidability

- algorithm has a good adaptation behaviour: modeling features that are not used do hardly impede computation (“pay as you go”)
- tableau method not useful for complexity considerations
- *SROIQ* 2-NExpTime-complete
 - *RIQ* and *SROIQ* are Harder than *SHOIQ*. Yevgeny Kazakov. In Gerhard Brewka and Jérôme Lang, editors, KR 2008. Pages 274-284. AAAI Press. 2008
 - lower bound: encoding of a 2Exp tiling problem
 - upper bound: exponential translation into the 2-variable fragment of FOL with counting quantifiers, C_2 , for which satisfiability checking is known to be NExpTime-complete)

Agenda

- Recap OWL & Overview OWL 2
- The Description Logic *SROIQ*
- Inferencing with *SROIQ*
- OWL 2 DL
- OWL 2 Profiles
- OWL 2 Full
- Summary

OWL 2 DL: Further Aspects

SROIQ is “only” logical foundation of OWL 2 DL

further non-logical aspects:

- Syntax (extension necessary)
- datatype declarations and datatype functions, new datatypes?
- metamodeling: “punning”
- comments and ontological metadata
- invers-functional concrete roles (datatype properties): Keys?
- mechanisms for ontology import?
- ...

↪ diverse smaller changes

Metamodeling

Metamodeling

specification of ontological knowledge about elements of the ontology (including classes, roles, axioms).

Examples:

- “The class Person was created on the 30.1.2008 by bglimm.”
- “For the class City, we recommend the property numberOfCitizens.”
- “The statement ‘Dresden was founded in 1206’ was extracted automatically with a confidence of 85%.”

(Compare Reification in RDF Schema)

Punning in OWL

Metamodeling in expressive logics is dangerous and expensive!

OWL 2 currently supports the simple form of metamodeling:

Punning

- the names for classes, roles, individuals do not have to be disjoint
- no logical relationship between class, individual and role of the same name
- only relevant for pragmatic interpretation

Example:

```
Person(Birte)  classCreatedBy(Person, bglimm)
```

Comments and Metadata

punning supports simple metadata with (weak) semantic meaning

How can one make purely syntactic comments in an ontology?

- comments in XML files: `<!-- comment -->`

Comments and Metadata

punning supports simple metadata with (weak) semantic meaning

How can one make purely syntactic comments in an ontology?

- comments in XML files: `<!-- comment -->`
↔ no relation to the OWL axioms in this file
- non-logical annotations in OWL 2: `owl:AnnotationProperty`

Comments and Metadata

punning supports simple metadata with (weak) semantic meaning

How can one make purely syntactic comments in an ontology?

- comments in XML files: `<!-- comment -->`
↪ no relation to the OWL axioms in this file
- non-logical annotations in OWL 2: `owl:AnnotationProperty`
↪ attached to (semantic) ontological element

Syntactic Aspects

new/extended syntaxes:

- RDF/XML: extension by OWL 2 elements
- functional-style syntax: replaces “abstract syntax” in OWL 1
- OWL/XML: syntax for simpler processing in XML tools
- Turtle: RDF triple syntax
- Manchester syntax: syntax that is easier to read for humans

Quo vadis, OWL Lite?

Quo vadis, OWL Lite?

OWL Lite as a Failure:

- almost as complex as OWL DL
- complicated syntax that does not provide direct access to actual modeling power
- use in ontologies today only "by accident", not deliberately

original goal:

capture the part of OWL that is easy and efficiently implementable

Quo vadis, OWL Lite?

OWL Lite as a Failure:

- almost as complex as OWL DL
- complicated syntax that does not provide direct access to actual modeling power
- use in ontologies today only "by accident", not deliberately

original goal:

capture the part of OWL that is easy and efficiently implementable

↔ OWL 2 Profiles

Agenda

- Recap OWL & Overview OWL 2
- The Description Logic *SROIQ*
- Inferencing with *SROIQ*
- OWL 2 DL
- OWL 2 Profiles
- OWL 2 Full
- Summary

OWL 2 Profiles

OWL 2 defines three fragments where automated inferencing can be done in PTime

- OWL EL
 - computation of the class hierarchy (all subclass relationships) in PTime

OWL 2 Profiles

OWL 2 defines three fragments where automated inferencing can be done in PTime

- OWL EL
 - computation of the class hierarchy (all subclass relationships) in PTime
- OWL QL
 - conjunctive queries in AC_0 (data complexity) \rightsquigarrow reducible to SQL

OWL 2 Profiles

OWL 2 defines three fragments where automated inferencing can be done in PTime

- OWL EL
 - computation of the class hierarchy (all subclass relationships) in PTime
- OWL QL
 - conjunctive queries in AC_0 (data complexity) \rightsquigarrow reducible to SQL
- OWL RL
 - can be used as an extension of RDFS or as a fragment of OWL DL (OWL Direct Semantics)
 - complexity PTime

OWL 2 EL

- An (almost maximal) fragment of OWL 2 such that
 - satisfiability can be checked in PTime (PTime-complete)
 - data complexity for ABox queries also PTime-complete
- class hierarchy (all subsumption relationships between atomic classes) can be computed in one pass
- uses a saturation method that was developed for the description logic \mathcal{EL}

OWL 2 EL

- allowed:
 - subclass axioms with conjunction, existential restriction, \top , \perp , singleton nominals
 - complex RIAs, range restrictions (under certain conditions)
- not allowed:
 - negation, disjunction, universal restrictions, inverse roles

OWL 2 QL

- an (almost maximal) fragment of OWL 2 such that
 - data complexity of conjunctive query answering is in AC^0
- queries can be rewritten such that no terminological knowledge has to be taken into account
 - ⇒ standard RDBMS can be used for storage and querying

OWL 2 QL

- allowed:
 - simple role hierarchies, domain & range axioms
 - subclass axioms with
 - lhs: class name or existential restriction with \top
 - rhs: conjunction of class names, existential restriction and negation of lhs expressions
- supports RDFS with “well-formed” graphs

OWL 2 RL

- An (almost maximal) fragment of OWL 2 such that
 - automated inferencing is PTime-complete (consistency, satisfiability of classes, subsumption, class membership checks)
 - automated inferencing is correct (sound & complete) if the given RDF graph satisfies certain requirements
 - otherwise the automated reasoning may be sound but incomplete.
- can operate directly on RDF triples in order to enrich instance data (materialization, forward chaining for facts)
- automated inferencing can be implemented via a set of rules (using a rule engine that supports equality)

Agenda

- Recap OWL & Overview OWL 2
- The Description Logic *SROIQ*
- Inferencing with *SROIQ*
- OWL 2 DL
- OWL 2 Profiles
- OWL 2 Full
- Summary

What to do with OWL Full?

Goal of OWL 2 DL: make many OWL Full 1.0 ontologies interpretable as OWL DL (cf. e.g. punning)

What to do with OWL Full?

Goal of OWL 2 DL: make many OWL Full 1.0 ontologies interpretable as OWL DL (cf. e.g. punning)

- extension of OWL Full by OWL 2 features is required by a few practitioners
- allows to work on all kinds of RDF graphs
- despite undecidability: many FOL verification tools do not guarantee termination and are still useful
- alternative implementation techniques can be used, which might be faster (but do not guarantee termination)

Crucial Differences in the Semantics

- annotations do not have a semantics in the direct semantics (which is used for OWL DL), but they do in the RDF-based semantics (which is used for OWL Full)
- import commands are only parser commands in the direct semantics, but do have a presence as triple in the RDF-based Semantics
- in the RDF-based semantics, classes are individuals, that are endowed with an extension \rightsquigarrow semantic conditions are only applicable to those classes that have an individual representant

Crucial Differences in the Semantics

Example

- $C(a)$
- query for all instances of the class $C \sqcup D$

Crucial Differences in the Semantics

Example

- $C(a)$
- query for all instances of the class $C \sqcup D$
- RDF-based semantics: \emptyset , direct semantics: a

Crucial Differences in the Semantics

Example

- $C(a)$
 - query for all instances of the class $C \sqcup D$
 - RDF-based semantics: \emptyset , direct semantics: a
- ↪ under the RDF-based semantics, we only have the guarantee that the union of the extensions of C and D do exist as subsets of the domain, however it is not ensured that an element exists which has this set as extension.
- ↪ contrarily, in the direct semantics class names “directly” represent sets and not domain elements
- ↪ the answer coincides for both semantics after adding $E \equiv C \sqcup D$

Agenda

- Recap OWL & Overview OWL 2
- The Description Logic *SROIQ*
- Inferencing with *SROIQ*
- OWL 2 DL
- OWL 2 Profiles
- OWL 2 Full
- Summary

Summary

OWL 2 as first extension of the OWL standard

- standardized 27.10.2009
- logical extension based on description logic *SROIQ*
- new modeling features, most notably complex RIAs, qualified number restrictions
- non-logical extensions: punning, comments, datatypes, etc.
- profiles with polynomial reasoning procedures