Semantic Web

5. RDF-S and OWL

Gerd Gröner, Matthias Thimm

{groener,thimm}@uni-koblenz.de

Institute for Web Science and Technologies (WeST)
University of Koblenz-Landau

July 12, 2013

Recap

- The RDF Model
  - URIs, literals, blank nodes
  - statements, graphs
- Complex statements
  - Typed literals
  - Containers: sequences, bags, alternatives
  - Linked lists
  - Reification
- RDF encodings: Turtle, RDF/XML

Home assignment:
Represent the following statements in RDF (Home assignment):

Jack Smith studies at Koblenz University
Koblenz University has the web site `http://www.uni-koblenz.de`
Jack is a friend of Anna
Anna studies at Boston University
Boston University has the website `http://www.bu.edu`
Anna has a dog called "Snoopy"

RDF vocabulary

- Recap RDF vocabulary:
  - `rdf:XMLLiteral`: XML literal values
  - `rdf:Property`: class of properties
  - `rdf:Statement`: class of RDF statements
  - `rdf:Alt`, `rdf:Bag`, `rdf:Seq`: containers
  - `rdf:List`: class of RDF Lists
  - `rdf:nil`: the empty list
  - `rdf:type`: type of an instance
  - `rdf:first`: first item in a list
  - `rdf:rest`: rest of a list
  - `rdf:value`: for structured values
  - `rdf:subject`: subject of a statement
  - `rdf:predicate` predicate of a statement
  - `rdf:object`: object of a statement

What is missing?

RDF Schema

- RDFS: RDF Schema

RDF Schema

- RDFS: RDF Schema
- Extends the basic RDF vocabulary with *meta-modeling* capabilities

WeST People and Knowledge Networks

RDF Schema

- RDFS: RDF Schema
- Extends the basic RDF vocabulary with *meta-modeling* capabilities
- RDFS defines a basic set of classes and properties, together with their semantics (interpretation) and logic

RDF Schema

- RDFS: RDF Schema
- Extends the basic RDF vocabulary with *meta-modeling* capabilities
- RDFS defines a basic set of classes and properties, together with their semantics (interpretation) and logic
- Some elements:
  - Class, subClassOf, DataType
  - Property, subPropertyOf
  - Domain, range

OWL

- OWL: Web Ontology Language

OWL

- OWL: Web Ontology Language
- Extends RDFS even further

OWL

- OWL: Web Ontology Language
- Extends RDFS even further
- Incorporates aspects of description logics

OWL

- OWL: Web Ontology Language
- Extends RDFS even further
- Incorporates aspects of description logics
- Defines formal semantics for RDF

Outline

RDFS Vocabulary 1/2

- RDFS namespace:
  `xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"`

RDFS Vocabulary 1/2

- RDFS namespace:
  `xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"`
- RDFS classes:
  - `rdfs:Resource`: everything is a resource
  - `rdfs:Literal`: the class of literal values
  - `rdfs:Class`: the class of classes
  - `rdfs:Datatype`: the class of datatypes
  - `rdfs:Container`: the class of containers
  - `rdfs:ContainerMembershipProperty`: rdf:_1, rdf:_2, ...

RDFS Vocabulary 2/2

- RDFS properties:
  - `rdfs:subClassOf`: subclass of a class
  - `rdfs:subPropertyOf`: subproperty of a property
  - `rdfs:domain`: domain of a property
  - `rdfs:range`: range of a property
  - `rdfs:label`: the name
  - `rdfs:comment`: a description of resource
  - `rdfs:member`: a member of a resource
  - `rdfs:seeAlso`: further information about a resource
  - `rdfs:isDefinedBy`: the definition of a resource

WeST People and Knowledge Networks

RDFS Principles

- rdfs:Resource: all resources are instances of rdfs:Resource (like $\top$ in description logics):

  (ex:John, rdf:type, rdfs:Resource)
  (rdfs:Resource, rdf:type, rdfs:Resource)

RDFS Principles

- `rdfs:Resource`: all resources are instances of `rdfs:Resource` (like $\top$ in description logics):

  (`ex:John`, `rdf:type`, `rdfs:Resource`)
  (`rdfs:Resource`, `rdf:type`, `rdfs:Resource`)

- `rdfs:Class`: sets of resources, can be hierarchically structured (like concepts in description logics):

  (`ex:Bird`, `rdf:type`, `rdfs:Class`)
  (`ex:Bird`, `rdfs:subClassOf`, `ex:Animal`)

RDFS Principles

- `rdfs:Resource`: all resources are instances of `rdfs:Resource` (like $\top$ in description logics):

  (`ex:John`, `rdf:type`, `rdfs:Resource`)
  (`rdfs:Resource`, `rdf:type`, `rdfs:Resource`)

- `rdfs:Class`: sets of resources, can be hierarchically structured (like concepts in description logics):

  (`ex:Bird`, `rdf:type`, `rdfs:Class`)
  (`ex:Bird`, `rdfs:subClassOf`, `ex:Animal`)

- `rdfs:Property`: resources that link resources, can appear in the middle-position of triples (like relations in description logics):

  (`ex:Color`, `rdf:type`, `rdfs:Class`)
  (`ex:hasColor`, `rdf:type`, `rdfs:Property`)
  (`ex:hasColor`, `rdfs:range`, `ex:Color`)
  (`ex:hasColor`, `rdfs:domain`, `rdfs:Resource`)

Example

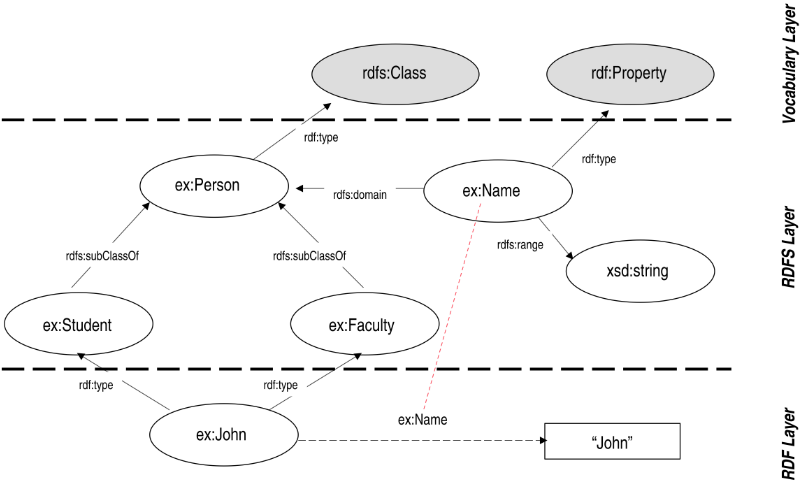

Image taken from Jos de Bruijn, Semantic Web Technologies course, 2008

Semantics 1/3

- RDF-S terms come with a well-defined meaning that allows limited *reasoning* on RDF

Semantics 1/3

- RDF-S terms come with a well-defined meaning that allows limited *reasoning* on RDF
- Rules (informally):
  - If (`X`,`rdfs:domain`,`Y`) and (`U`,`X`,`V`) then (`U`,`rdf:type`,`Y`)

WeST People and Knowledge Networks

Semantics 1/3

- RDF-S terms come with a well-defined meaning that allows limited *reasoning* on RDF
- Rules (informally):
  - If (`X`,`rdfs:domain`,`Y`) and (`U`,`X`,`V`) then (`U`,`rdf:type`,`Y`)
    - If (`ex:motherOf`,`rdfs:domain`,`ex:Human`) and (`ex:Mia`,`ex:motherOf`,`ex:Ann`) then (`ex:Mia`,`rdf:type`,`ex:Human`)

Semantics 1/3

- RDF-S terms come with a well-defined meaning that allows limited *reasoning* on RDF
- Rules (informally):
  - If (`X`,`rdfs:domain`,`Y`) and (`U`,`X`,`V`) then (`U`,`rdf:type`,`Y`)
    - If (`ex:motherOf`,`rdfs:domain`,`ex:Human`) and (`ex:Mia`,`ex:motherOf`,`ex:Ann`) then (`ex:Mia`,`rdf:type`,`ex:Human`)
  - If (`X`,`rdfs:range`,`Y`) and (`U`,`X`,`V`) then (`V`,`rdf:type`,`Y`)

WeST People and Knowledge Networks

Semantics 1/3

- RDF-S terms come with a well-defined meaning that allows limited *reasoning* on RDF
- Rules (informally):
  - If (`X`,`rdfs:domain`,`Y`) and (`U`,`X`,`V`) then (`U`,`rdf:type`,`Y`)
    - If (`ex:motherOf`,`rdfs:domain`,`ex:Human`) and (`ex:Mia`,`ex:motherOf`,`ex:Ann`) then (`ex:Mia`,`rdf:type`,`ex:Human`)
  - If (`X`,`rdfs:range`,`Y`) and (`U`,`X`,`V`) then (`V`,`rdf:type`,`Y`)
    - If (`ex:hasColor`,`rdfs:range`,`ex:Color`) and (`ex:hammer`,`ex:hasColor`,`ex:green`) then (`ex:green`,`rdf:type`,`ex:Color`)

Semantics 2/3

- Rules (informally, cont'd):
  - If (X,`rdfs:subPropertyOf`,Y) and (Y,`rdfs:subPropertyOf`,Z) then (X,`rdfs:subPropertyOf`,Z)

Semantics 2/3

- Rules (informally, cont'd):
  - If (X,rdfs:subPropertyOf,Y) and (Y,rdfs:subPropertyOf,Z) then (X,rdfs:subPropertyOf,Z)
    - If (ex:brotherOf,rdfs:subPropertyOf,ex:SiblingOf) and (ex:SiblingOf,rdfs:subPropertyOf,ex:RelativeOf) then (ex:brotherOf,rdfs:subPropertyOf,ex:RelativeOf)

Semantics 2/3

- Rules (informally, cont'd):
  - If (X,rdfs:subPropertyOf,Y) and (Y,rdfs:subPropertyOf,Z) then (X,rdfs:subPropertyOf,Z)
    - If (ex:brotherOf,rdfs:subPropertyOf,ex:SiblingOf) and (ex:SiblingOf,rdfs:subPropertyOf,ex:RelativeOf) then (ex:brotherOf,rdfs:subPropertyOf,ex:RelativeOf)
  - If (X,rdfs:subPropertyOf,Y) and (U,X,V) then (U,Y,V)

WeST People and Knowledge Networks

Semantics 2/3

- Rules (informally, cont'd):
  - If (X,rdfs:subPropertyOf,Y) and (Y,rdfs:subPropertyOf,Z) then (X,rdfs:subPropertyOf,Z)
    - If (ex:brotherOf,rdfs:subPropertyOf,ex:SiblingOf) and (ex:SiblingOf,rdfs:subPropertyOf,ex:RelativeOf) then (ex:brotherOf,rdfs:subPropertyOf,ex:RelativeOf)
  - If (X,rdfs:subPropertyOf,Y) and (U,X,V) then (U,Y,V)
    - If (ex:Carl,ex:brotherOf,ex:Dave) and (ex:brotherOf,rdfs:subPropertyOf,ex:SiblingOf) then (ex:Carl,ex:SiblingOf,ex:Dave)

Semantics 3/3

- Rules (informally, cont'd):
  - If (X,rdfs:subClassOf,Y) and (Y,rdfs:subClassOf,Z) then (X,rdfs:subClassOf,Z)

Semantics 3/3

- Rules (informally, cont'd):
  - If (X,rdfs:subClassOf,Y) and (Y,rdfs:subClassOf,Z) then (X,rdfs:subClassOf,Z)
    - If (ex:Cat,rdfs:subClassOf,ex:Mammal) and (ex:Mammal,rdfs:subClassOf,ex:Animal) then (ex:Cat,rdfs:subClassOf,ex:Animal)

WeST People and Knowledge Networks

Semantics 3/3

- Rules (informally, cont'd):
  - If (X,rdfs:subClassOf,Y) and (Y,rdfs:subClassOf,Z) then (X,rdfs:subClassOf,Z)
    - If (ex:Cat,rdfs:subClassOf,ex:Mammal) and (ex:Mammal,rdfs:subClassOf,ex:Animal) then (ex:Cat,rdfs:subClassOf,ex:Animal)
  - If (X,rdfs:subClassOf,Y) and (U,rdf:type,X) then (U,rdf:type,Y)

Semantics 3/3

- Rules (informally, cont'd):
  - If (X,rdfs:subClassOf,Y) and (Y,rdfs:subClassOf,Z) then (X,rdfs:subClassOf,Z)
    - If (ex:Cat,rdfs:subClassOf,ex:Mammal) and (ex:Mammal,rdfs:subClassOf,ex:Animal) then (ex:Cat,rdfs:subClassOf,ex:Animal)
  - If (X,rdfs:subClassOf,Y) and (U,rdf:type,X) then (U,rdf:type,Y)
    - If (ex:Cat,rdfs:subClassOf,ex:Mammal) and (ex:Kitty,rdf:type,ex:Cat) then (ex:Kitty,rdf:type,ex:Mammal)

WeST People and Knowledge Networks

Axiomatic Triples

The following triples are assumed to be always true:

```
(rdf:type, rdfs:domain, rdfs:Resource)
(rdfs:domain, rdfs:domain, rdf:Property)
(rdfs:range, rdfs:domain, rdf:Property)
(rdfs:subPropertyOf, rdfs:domain, rdf:Property)
...
(rdf:type, rdfs:range, rdfs:Class)
(rdfs:domain, rdfs:range, rdfs:Class)
(rdfs:range, rdfs:range, rdfs:Class)
(rdfs:subPropertyOf, rdfs:range, rdf:Property)
(rdfs:subClassOf, rdfs:range, rdfs:Class)
...
(rdf:Alt, rdfs:subClassOf, rdfs:Container)
(rdf:Bag, rdfs:subClassOf, rdfs:Container)
...
```

Entailment

- If a triplestore (=data base for RDF triples) implements the RDF/RDFS *entailment regime* the previous rules are applied when answering queries

Entailment

- If a triplestore (=data base for RDF triples) implements the RDF/RDFS *entailment regime* the previous rules are applied when answering queries
- more on queries later ($\rightarrow$ SPARQL)

WeST People and Knowledge Networks

Entailment

- If a triplestore (=data base for RDF triples) implements the RDF/RDFS *entailment regime* the previous rules are applied when answering queries
- more on queries later ($\rightarrow$ SPARQL)
- Small example anyway: Querying

  (`ex:Cat`,`rdfs:subClassOf`,`ex:Mammal`)
  (`ex:Kitty`, `rdf:type`, `ex:Cat`)
  (`ex:Dumbo`, `rdf:type`, `ex:Mammal`)

  with "Get all mammals" returns `Dumbo` and `Kitty`.

Outline

Limitations of RDFS

- RDFS has limited expressive power for several applications
  - No conditional range and domain restrictions
    - The range of `hasChild` should be `Human` when applied to humans and `Elephant` when applied to elephants

Limitations of RDFS

- RDFS has limited expressive power for several applications
  - No conditional range and domain restrictions
    - The range of `hasChild` should be `Human` when applied to humans and `Elephant` when applied to elephants
  - No existence/cardinality constraints
    - All instances of `Person` must have a mother
    - All instances of `Person` must have two parents

Limitations of RDFS

- RDFS has limited expressive power for several applications
  - No conditional range and domain restrictions
    - The range of `hasChild` should be `Human` when applied to humans and `Elephant` when applied to elephants
  - No existence/cardinality constraints
    - All instances of `Person` must have a mother
    - All instances of `Person` must have two parents
  - No constraints on properties
    - `Knows` is symmetric
    - `isAncestor` is transitive

WeST People and Knowledge Networks

OWL - Web Ontology Language

- The OWL family comprises actually three different languages
  - OWL Lite
    - Classification hierarchy
    - Simple constraints

OWL - Web Ontology Language

- The OWL family comprises actually three different languages
  - OWL Lite
    - Classification hierarchy
    - Simple constraints
  - OWL DL
    - Maximal expressiveness
    - Still tractable

OWL - Web Ontology Language

- The OWL family comprises actually three different languages
  - OWL Lite
    - Classification hierarchy
    - Simple constraints
  - OWL DL
    - Maximal expressiveness
    - Still tractable
  - OWL Full
    - Even higher expressiveness
    - Not tractable

WeST People and Knowledge Networks

Features of OWL Language Layers

- OWL Lite
  - (sub)classes, individuals
  - (sub)properties, domain, range
  - conjunction
  - (in)equality
  - cardinality 0/1
  - datatypes
  - inverse, transitive, symmetric properties
  - someValuesFrom
  - allValuesFrom
- OWL DL
  - Negation
  - Disjunction
  - Full cardinality
  - Enumerated types
  - hasValue
- OWL Full
  - Meta-classes
  - More compatible with RDFS

OWL Full

- No restriction on vocabulary
  - use classes as instances

OWL Full

- No restriction on vocabulary
  - use classes as instances
- `owl:Class`≡`rdfs:Class`

OWL Full

- No restriction on vocabulary
  - use classes as instances
- `owl:Class`≡`rdfs:Class`
- RDF-style model theory
  - Reasoning using FOL engine
  - Semantics should correspond to OWL DL for restricted KBs
- based on $\mathcal{SROIQ}$

OWL DL

- Use of vocabulary restricted
  - no classes as instances
  - defined by abstract syntax

OWL DL

- Use of vocabulary restricted
  - no classes as instances
  - defined by abstract syntax
- owl:Class$\sqsubset$rdfs:Class

OWL DL

- Use of vocabulary restricted
  - no classes as instances
  - defined by abstract syntax
- owl:Class$\sqsubset$rdfs:Class
- Standard DL-based model theory
  - Direct correspondence with a DL
  - Reasoning via DL engines
- based on $\mathcal{SHOIN}$

OWL Lite

- No explicit negation or union
- Restricted cardinality (0/1)
- No nominals (oneOf)
- DL-based semantics
  - Reasoning via DL engines

OWL Lite

- No explicit negation or union
- Restricted cardinality (0/1)
- No nominals (oneOf)
- DL-based semantics
  - Reasoning via DL engines
- `owl:Class` $\sqsubset$ `rdfs:Class`

WeST People and Knowledge Networks

OWL Lite

- No explicit negation or union
- Restricted cardinality (0/1)
- No nominals (oneOf)
- DL-based semantics
  - Reasoning via DL engines
- `owl:Class`$\sqsubset$`rdfs:Class`
- Semantically, only small restriction on OWL DL
  - No nominals
  - no arbitrary cardinality
- based on $\mathcal{SHIF}$

OWL concepts

OWL concept	DL	Example
intersectionOf	$C_1 \sqcap \ldots \sqcap C_n$	$Actor \sqcap Politician$
unionOf	$C_1 \sqcup \ldots \sqcup C_n$	$Male \sqcup Female$
complementOf	$\neg C$	$\neg Student$
oneOf	$\{o_1, \ldots, o_n\}$	$\{carl, dave\}$
allValuesFrom	$\forall P.C$	$\forall hasChild.Female$
someValuesFrom	$\exists P.C$	$\exists hasChild.Female$
value	$\exists P.\{v\}$	$\exists hasColor.green$
minCardinality	$\geq_n P.C$	$\geq_3 hasChild.Male$
maxCardinality	$\leq_n P.C$	$\leq_2 hasChild.Male$
cardinality	$=_n P.C$	$=_3 hasChild.Male$

OWL axioms

OWL axiom	DL	Example
SubClassOf	$C_1 \sqsubseteq C_n$	$Platypus \sqsubseteq Mammal \sqcap Oviparous$
EquivalentClasses	$C_1 \equiv C_2$	$Man \equiv Human \sqcap Male$
SubPropertyOf	$P_1 \sqsubseteq P_2$	$hasParent \sqsubseteq hasAncestor$
EquivalentProperties	$P_1 \equiv P_2$	$hasCost \equiv hasPrice$
SameIndividual	$o_1 \equiv o_2$	$jack \equiv jacksmith$
DisjointClasses	$C_1 \sqsubseteq \neg C_2$	$Male \sqsubseteq \neg Female$
inverseOf	$P_1 \equiv P_2^-$	$hasChild \equiv hasParent^-$
Transitive	$P^+ \sqsubseteq P$	$hasAncestor^+ \sqsubseteq hasAncestor$
Symmetric	$P \equiv P^-$	$knows \equiv knows^-$

Example 1/2

In OWL:

```
Class( associateProfessor partial academicStaffMember)
DisjointClasses ( associateProfessor assistantProfessor )
DisjointClasses ( professor associateProfessor )
Class( faculty complete academicStaffMember)
```

In DL:

$$\text{associateProfessor} \sqsubseteq \text{academicStaffMember}$$
$$\text{associateProfessor} \sqsubseteq \neg\text{assistantProfessor}$$
$$\text{professor} \sqsubseteq \neg\text{associateProfessor}$$
$$\text{faculty} \equiv \text{academicStaffMember}$$

WeST People and Knowledge Networks

Example 2/2

In OWL:

```
DatatypeProperty(age range(xsd:nonNegativeInteger))
SubPropertyOf(isTaughtBy involves)
ObjectProperty(teaches inverseOf(isTaughtBy)
   domain(academicStaffMember) range(course))
```

In DL:

$\top \sqsubseteq \forall age.xsd:nonNegativeInteger$

$isTaughtBy \sqsubseteq involves$

$teaches \equiv isTaughtBy^-$

$\top \sqsubseteq \forall teaches^-.academicStaffMember$

$\top \sqsubseteq \forall teaches.course$

WeST People and Knowledge Networks

OWL in RDF

- OWL namespace:
  `xmlns:owl="http://www.w3.org/2002/07/owl"`

OWL in RDF

- OWL namespace:
  `xmlns:owl="http://www.w3.org/2002/07/owl"`
- OWL expressions can be represented in RDF
  - `Class(C)` → `(C, rdf:type, owl:class)`

OWL in RDF

- OWL namespace:
  `xmlns:owl="http://www.w3.org/2002/07/owl"`
- OWL expressions can be represented in RDF
  - `Class(C)` → `(C, rdf:type, owl:class)`
  - `DisjointClasses(C1 C2)` → `(C1, owl:disjointOf, C2)`

WeST People and Knowledge Networks

OWL in RDF

- OWL namespace:
  `xmlns:owl="http://www.w3.org/2002/07/owl"`
- OWL expressions can be represented in RDF
  - `Class(C)` → `(C, rdf:type, owl:class)`
  - `DisjointClasses(C1 C2)` → `(C1, owl:disjointOf, C2)`
  - `SymmetricObjectProperty(P)` → `(P, rdf:type, owl:SymmetricProperty)`

OWL in RDF

- OWL namespace:
  `xmlns:owl="http://www.w3.org/2002/07/owl"`
- OWL expressions can be represented in RDF
  - `Class(C)` → `(C, rdf:type, owl:class)`
  - `DisjointClasses(C1 C2)` → `(C1, owl:disjointOf, C2)`
  - `SymmetricObjectProperty(P)` → `(P, rdf:type, owl:SymmetricProperty)`
  - `SameIndividual(o1 o2)` → `(o1, owl:sameAs, o2)`

OWL in RDF

- OWL namespace:
  `xmlns:owl="http://www.w3.org/2002/07/owl"`
- OWL expressions can be represented in RDF
  - `Class(C)` → `(C, rdf:type, owl:class)`
  - `DisjointClasses(C1 C2)` → `(C1, owl:disjointOf, C2)`
  - `SymmetricObjectProperty(P)` → `(P, rdf:type, owl:SymmetricProperty)`
  - `SameIndividual(o1 o2)` → `(o1, owl:sameAs, o2)`
  - `ObjectAllValuesFrom(P C)` →
    - `(_x, rdf:type, owl:Restriction)`
    - `(_x, owl:onProperty, P)`
    - `(_x, owl:allValuesFrom, C)`

WeST People and Knowledge Networks

OWL in RDF

- OWL namespace:
  `xmlns:owl="http://www.w3.org/2002/07/owl"`
- OWL expressions can be represented in RDF
  - `Class(C)` → `(C, rdf:type, owl:class)`
  - `DisjointClasses(C1 C2)` → `(C1, owl:disjointOf, C2)`
  - `SymmetricObjectProperty(P)` → `(P, rdf:type, owl:SymmetricProperty)`
  - `SameIndividual(o1 o2)` → `(o1, owl:sameAs, o2)`
  - `ObjectAllValuesFrom(P C)` →
    - `(_x, rdf:type, owl:Restriction)`
    - `(_x, owl:onProperty, P)`
    - `(_x, owl:allValuesFrom, C)`
    - Example:

      ```
      (ex:ParentOfOnlyDaughters, owl:equivalentClass, _x)
      (_x, rdf:type, owl:Restriction)
      (_x, owl:onProperty, ex:hasChild)
      (_x, owl:allValuesFrom, ex:Female)
      ```

WeST People and Knowledge Networks

OWL in RDF

- OWL namespace:
  `xmlns:owl="http://www.w3.org/2002/07/owl"`
- OWL expressions can be represented in RDF
  - `Class(C)` → `(C, rdf:type, owl:class)`
  - `DisjointClasses(C1 C2)` → `(C1, owl:disjointOf, C2)`
  - `SymmetricObjectProperty(P)` → `(P, rdf:type, owl:SymmetricProperty)`
  - `SameIndividual(o1 o2)` → `(o1, owl:sameAs, o2)`
  - `ObjectAllValuesFrom(P C)` →
    - `(_x, rdf:type, owl:Restriction)`
    - `(_x, owl:onProperty, P)`
    - `(_x, owl:allValuesFrom, C)`
    - Example:

```
(ex:ParentOfOnlyDaughters, owl:equivalentClass, _x)
(_x, rdf:type, owl:Restriction)
(_x, owl:onProperty, ex:hasChild)
(_x, owl:allValuesFrom, ex:Female)
```

  - ...

WeST People and Knowledge Networks

Outline

WeST People and Knowledge Networks

Summary

- RDF Schema
  - Resources, classes, properties, ...
  - Sub-classes, domain, range, ...
  - Semantics and entailment
- OWL
  - OWL Lite, OWL DL, OWL Full
  - OWL in RDF

Pointers to further reading

- Baader, Franz; Horrocks, Ian; Sattler, Ulrike (2005). Description Logics as Ontology Languages for the Semantic Web. In Hutter, Dieter; Stephan, Werner. Mechanizing Mathematical Reasoning: Essays in Honor of Jörg H. Siekmann on the Occasion of His 60th Birthday. Heidelberg, DE: Springer Berlin. ISBN 978-3-540-25051-7.
- Hitzler, Pascal; Krötzsch, Markus; Rudolph, Sebastian (2009-08-25). Foundations of Semantic Web Technologies. CRCPress. ISBN 1-4200-9050-X.
- Patel-Schneider, Peter F.; Motik, Boris (2009-10-27). OWL 2 Web Ontology Language Mapping to RDF Graphs. `http://www.w3.org/TR/2009/REC-owl2-mapping-to-rdf-20091027/`

WeST People and Knowledge Networks

Exercises

- What can be entailed about “ex:tweety”?

  (ex:tweety,rdf:type,ex:penguin)
  (ex:penguin,rdfs:subClassOf,ex:bird)
  (ex:bird,owl:disjointOf,ex:fish)

Exercises

- What can be entailed about "ex:tweety"?

  `(ex:tweety,rdf:type,ex:penguin)`
  `(ex:penguin,rdfs:subClassOf,ex:bird)`
  `(ex:bird,owl:disjointOf,ex:fish)`

- Represent the following statement in RDF using RDFS and OWL:

  Dave belongs to a group of people who have only sons and no sisters

WeST People and Knowledge Networks

Exercises

- What can be entailed about "ex:tweety"?

  (`ex:tweety`,`rdf:type`,`ex:penguin`)
  (`ex:penguin`,`rdfs:subClassOf`,`ex:bird`)
  (`ex:bird`,`owl:disjointOf`,`ex:fish`)

- Represent the following statement in RDF using RDFS and OWL:

  Dave belongs to a group of people who have only sons and no sisters

- Represent the following statement in RDF using RDFS and OWL:

  Carl belongs to a group of people who like everyone who likes them

WeST People and Knowledge Networks

Exercises

- What can be entailed about "ex:tweety"?

  (`ex:tweety`,`rdf:type`,`ex:penguin`)
  (`ex:penguin`,`rdfs:subClassOf`,`ex:bird`)
  (`ex:bird`,`owl:disjointOf`,`ex:fish`)

- Represent the following statement in RDF using RDFS and OWL:

  Dave belongs to a group of people who have only sons and no sisters

- Represent the following statement in RDF using RDFS and OWL:

  Carl belongs to a group of people who like everyone who likes them

Not possible?