Ontology Semantics, Repair and Enrichment
(with a Focus on Linked Data Knowledge Bases)

Jens Lehmann, Lorenz Bühmann

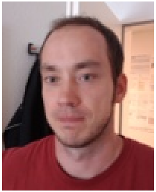
Creating Knowledge out of Interlinked Data

2011-09-15

Personal Introduction

Jens Lehmann

- PhD Uni Leipzig 2006-2010
- Head of Machine Learning and Ontology Engineering Group (MOLE) at AKSW since 2010
- Software/Research-Projects: DL-Learner, DBpedia, ORE, LinkedGeoData, AutoSPARQL, ReDD, SAIM, NLP2RDF



Lorenz Bühmann

- Studied Computer Science at Uni Leipzig 2006 - 2011
- PhD Uni Leipzig since 2011
- Software/Research-Projects: ORE, DL-Learner, AutoSPARQL

Outline

Outline

OWL (2) General

- OWL 1 W3C Recommendation since 2004 and OWL 2 since 2009
- Semantic fragment of FOL

OWL (2) General

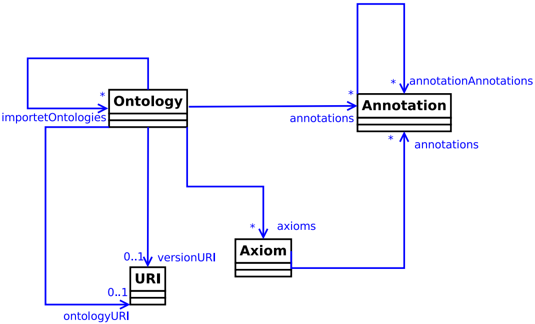
- OWL 1 W3C Recommendation since 2004 and OWL 2 since 2009
- Semantic fragment of FOL
- Variants of OWL 1: OWL Lite $\subseteq$ OWL DL $\subseteq$ OWL Full
- Variants of OWL 2: OWL 2 DL $\subseteq$ OWL 2 Full
- Profiles of OWL 2: OWL 2 QL, OWL 2 EL++, OWL 2-RL DL, OWL 2-RL Full

OWL (2) General

- OWL 1 W3C Recommendation since 2004 and OWL 2 since 2009
- Semantic fragment of FOL
- Variants of OWL 1: OWL Lite $\subseteq$ OWL DL $\subseteq$ OWL Full
- Variants of OWL 2: OWL 2 DL $\subseteq$ OWL 2 Full
- Profiles of OWL 2: OWL 2 QL, OWL 2 EL++, OWL 2-RL DL, OWL 2-RL Full
- OWL DL is decidable and corresponds to description logic $\mathcal{SHOIN}(D)$
- OWL 2 DL corresponds to $\mathcal{SROIQ}(D)$
- W3C documents contain more details than discussed here

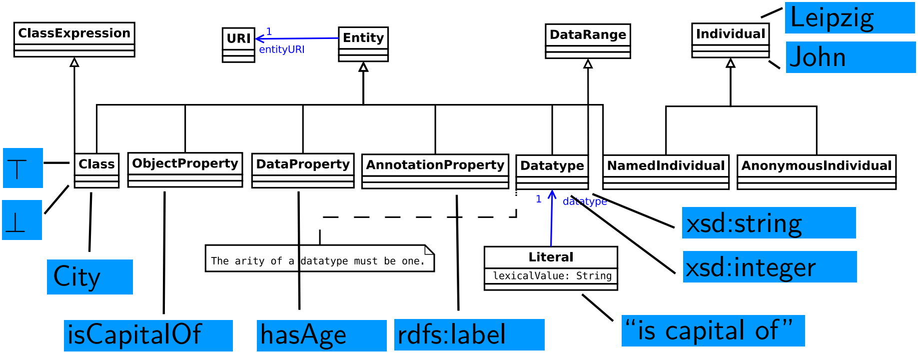
- we use Manchester Syntax and Description Logic (DL) syntax in this presentation
- DL syntax:
  $\texttt{Student} \sqsubseteq \texttt{Person}$
- Manchester syntax (will be introduced when needed):
  `Class: Student SubClassOf: Person`

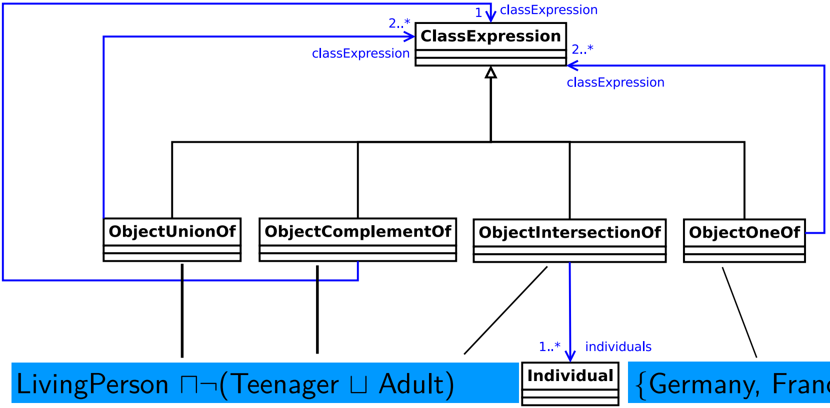
OWL 2 - Ontology Structure



- As in OWL 1:
  - Ontology = Set of axioms (+ Head)
  - 1 Axiom = 1...n RDF Triple
- Physical location has to correspond to version URI (if it exists) and current version has to be found at ontology URI (if it exists)

OWl 2 - Entities

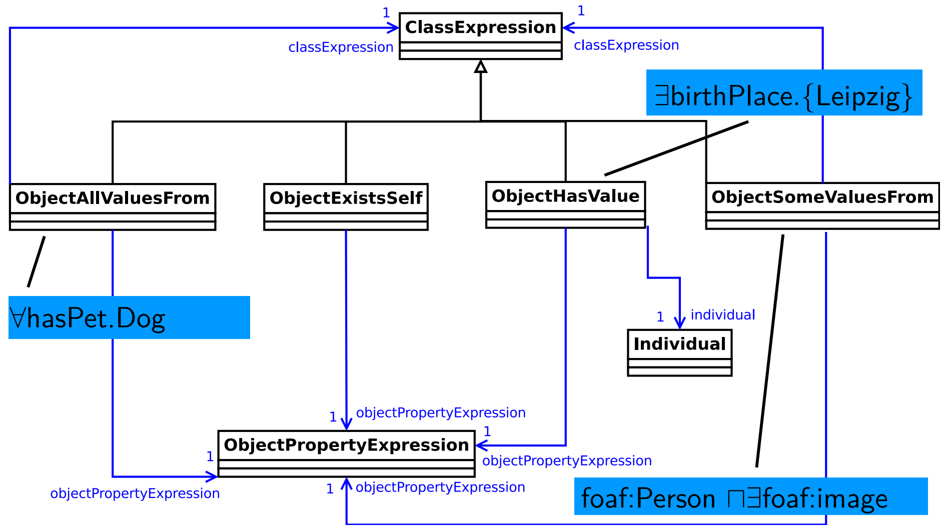




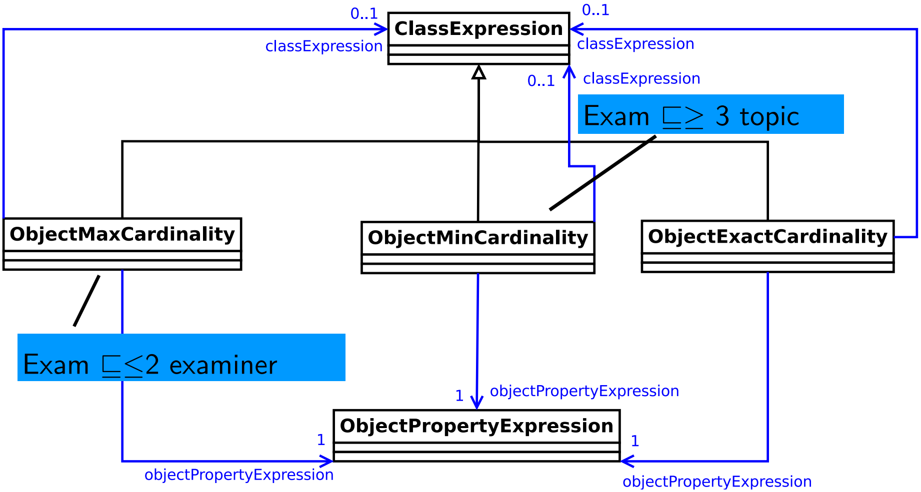
LivingPerson $\sqcap \neg$(Teenager $\sqcup$ Adult)

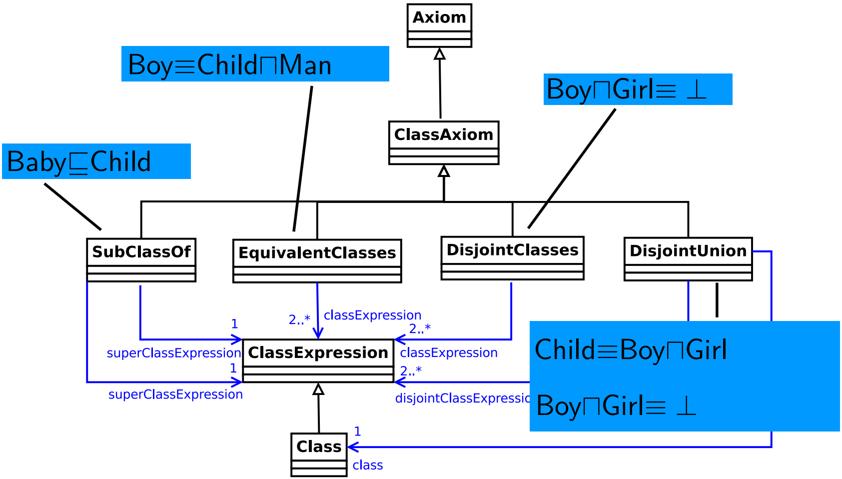
{Germany, France}

OWL 2 – Class Expressions

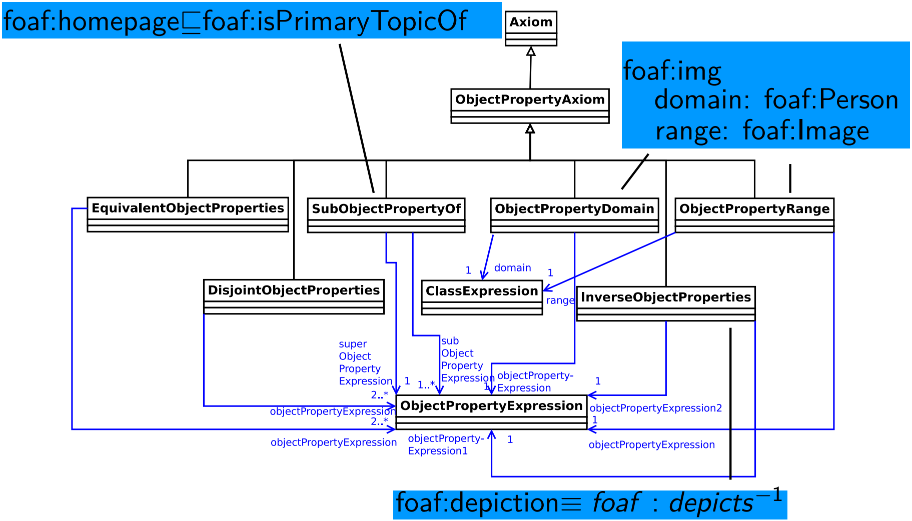


OWL 2 – Class Expressions

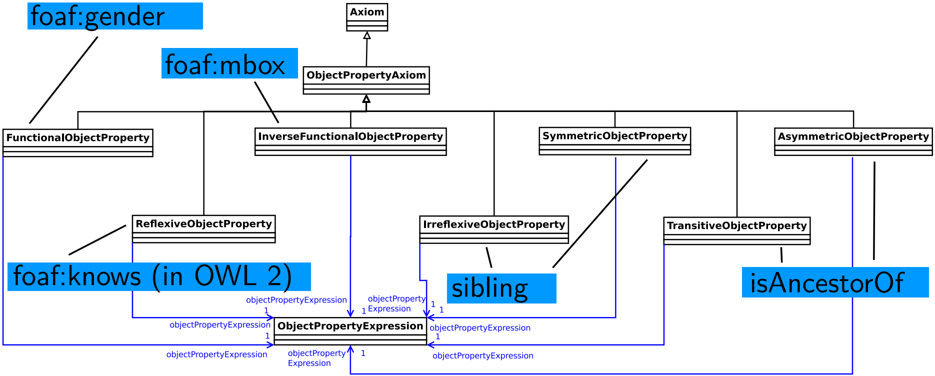



Axiom
ClassAxiom
Boy≡Child⊓Man
Boy⊓Girl≡ ⊥
Baby⊑Child
SubClassOf
EquivalentClasses
DisjointClasses
DisjointUnion
1
2..*
classExpression
2..*
superClassExpression
ClassExpression
classExpression
1
2..*
superClassExpression
disjointClassExpressio
Child≡Boy⊓Girl
Boy⊓Girl≡ ⊥
Class
1
class

OWL 2 - Axioms


foaf:homepage⊑foaf:isPrimaryTopicOf
Axiom
ObjectPropertyAxiom
foaf:img
domain: foaf:Person
range: foaf:Image
EquivalentObjectProperties
SubObjectPropertyOf
ObjectPropertyDomain
ObjectPropertyRange
DisjointObjectProperties
ClassExpression
InverseObjectProperties
domain
range
1
1
super Object Property Expression
sub Object Property Expression
objectProperty- Expression
1
1..*
2..*
1
ObjectPropertyExpression
objectPropertyExpression
objectPropertyExpression2
2..*
objectPropertyExpression
objectProperty- Expression1
1
1
objectPropertyExpression
foaf:depiction≡ foaf : depicts^{-1}

OWL 2 - Axioms



foaf:gender
foaf:mbox
Axiom
ObjectPropertyAxiom
FunctionalObjectProperty
InverseFunctionalObjectProperty
SymmetricObjectProperty
AsymmetricObjectProperty
ReflexiveObjectProperty
IrreflexiveObjectProperty
TransitiveObjectProperty
foaf:knows (in OWL 2)
sibling
isAncestorOf
ObjectPropertyExpression
objectPropertyExpression
1
objectProperty Expression

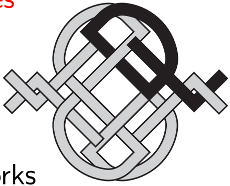
OWL 2 Tool-Support

- APIs: OWL API, KAON2
- Editors: Protégé, TopBraid
- Reasoning:
  - OWL 2 DL: Pellet, FaCT++
  - OWL 2 EL: CEL
  - OWL 2 QL: QuONto, Owlgres
  - OWL 2 RL: Oracle 11g
- ...

Outline

Description Logics (DLs)

- family of formal knowledge representation languages
- fragment of FOL
- mostly decidable
- relatively expressive
- created from attempts to formalize semantic networks
- intuitive syntax
- variable-free



- W3C-Standard OWL DL is based on Description Logic $\mathcal{SHOIN}(\mathcal{D})$ (OWL 2 DL based on $\mathcal{SROIQ}(\mathcal{D})$)
- we discuss $\mathcal{ALC}$ for explaining OWL/DL semantics

Basic components:

- class names (also referred to as concepts)
- role names
- individual names (also referred to as objects)

$\boxed{\text{knowledge base} = \text{set of axioms}}$

Axioms for instance data:

- *Man*(*Bob*)
  $\widehat{=}$ individual *Bob* belongs to class *Man*
- *hasPet*(*Bob*, *Tweety*)
  $\widehat{=}$ *Bob* has a pet *Tweety*

$Child \sqsubseteq Person$

- "Every child is a person."
- corresponds to $(\forall x)(Child(x) \rightarrow Person(x))$
- corresponds to `rdfs:subClassOf`

$Man \equiv MaleHuman$

- "Man are exactly the male humans."
- corresponds to $(\forall x)(Man(x) \leftrightarrow MaleHuman(x))$
- corresponds to `owl:equivalentClass`

$\mathcal{ALC}$ - complex classes

Conjunction $\sqcap$ corresponds to `owl:intersectionOf`

Disjunction $\sqcup$ corresponds to `owl:unionOf`

Negation $\neg$ corresponds to `owl:complementOf`

Example:
$Professor \sqsubseteq$
$(Person \sqcap UniversityMember) \sqcup (Person \sqcap \neg PhDStudent \sqcap Postgraduate))$

Predicate logic: $(\forall x)(Professor(x) \rightarrow ((Person(x) \wedge$
$UniversityMember(x)) \vee Person(x) \wedge \neg PhDStudent(x) \wedge Postgraduate(x))$

$\mathcal{ALC}$ - Quantors on roles

$Exam \sqsubseteq \forall hasExaminor.Professor$

- "Every exam has only professors as examinor."
- $(\forall x)(Exam(x) \rightarrow (\forall y)hasExaminor(x, y) \wedge Professor(y)))$
- corresponds to `owl:allValuesFrom`

$Exam \sqsubseteq \exists hasExaminor.Person$

- "Every exam has at least one examinor."
- $(\forall x)(Exam(x) \rightarrow (\exists y)(hasExaminor(x, y) \wedge Person(y)))$
- corresponds to `owl:someValuesFrom`

$\mathcal{ALC}$ - formal syntax

- The following syntax rules create classes in $\mathcal{ALC}$. Here A is an atomic class and R a role
  $C, D \rightarrow A|\top|\bot|\neg C|C \sqcap D|C \sqcup D|\forall R.C|\exists R.C$
- An $\mathcal{ALC}$-TBox consists of expressions of type $C \sqsubseteq D$ and $C \equiv D$, where C, D are classes.
- An $\mathcal{ALC}$-ABox consists of expressions of type $C(a)$ and $R(a, b)$, where C is a complex class, R is a role and a, b are individuals.
- An $\mathcal{ALC}$- knowledge base consists of a ABox and a TBox.

$\mathcal{ALC}$ - Semantics (Interpretations)

- we define the model-theoretic semantics of $\mathcal{ALC}$ (i.e. entailment is defined by interpretations)
- an interpretation $\mathcal{I} = (\triangle^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of
  - a set $\triangle^{\mathcal{I}}$, called domain and
  - a function $\cdot^{\mathcal{I}}$, which maps from
    - individual names a to elements of the domain $a \in \triangle^{\mathcal{I}}$
    - class names C to set of domain elements $C^{\mathcal{I}} \subseteq \triangle^{\mathcal{I}}$
    - role names R to set of pairs of domain elements $R^{\mathcal{I}} \subseteq \triangle^{\mathcal{I}} \times \triangle^{\mathcal{I}}$

$\mathcal{ALC}$ - Semantics (complex classes)

- will be extended to complex classes:

$\top^{\mathcal{I}} = \triangle^{\mathcal{I}}$

$\bot^{\mathcal{I}} = \emptyset$

$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$

$(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$

$(\forall R.C)^{\mathcal{I}} = \{x \mid \forall (x,y) \in R^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\}$

$(\exists R.C)^{\mathcal{I}} = \{x \mid \exists (x,y) \in R^{\mathcal{I}}$ mit $y \in C^{\mathcal{I}}\}$

$(\neg C)^{\mathcal{I}} = \triangle^{\mathcal{I}} \setminus C^{\mathcal{I}}$

... and finally to axioms:

- $C(a)$ is satisfied in $\mathcal{I}$, if: $a^{\mathcal{I}} \in C^{\mathcal{I}}$
- $R(a, b)$ is satisfied in $\mathcal{I}$, if: $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$
- $C \sqsubseteq D$ is satisfied in $\mathcal{I}$, if: $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
- $C \equiv D$ is satisfied in $\mathcal{I}$, if: $C^{\mathcal{I}} = D^{\mathcal{I}}$

Interpretations, which satisfy an axiom (resp. a set of axioms), are called models.

Interpretations - Examples

$$\text{TBox } \mathcal{T}: \quad \begin{aligned} \texttt{Man} &\equiv \neg\texttt{Woman} \sqcap \texttt{Person} \\ \texttt{Woman} &\sqsubseteq \texttt{Person} \\ \texttt{Mother} &\equiv \texttt{Woman} \sqcap \exists \texttt{hasChild}.\top \end{aligned}$$

$$\text{ABox } \mathcal{A}: \quad \begin{aligned} &\texttt{Man}(\texttt{STEPHEN}). \\ &\neg\texttt{Man}(\texttt{MONICA}). \\ &\texttt{Woman}(\texttt{JESSICA}). \\ &\texttt{hasChild}(\texttt{STEPHEN}, \texttt{JESSICA}). \end{aligned}$$

For all following interpretations $\mathcal{I}$:

- domain $\Delta^{\mathcal{I}} = \{MONICA, JESSICA, STEPHEN\}$
- objects are mapped to themselves ($a^{\mathcal{I}} = a$)

Interpretations - Examples

$$
\begin{aligned}
\text{TBox } \mathcal{T}: \quad & \mathtt{Man} \equiv \neg\mathtt{Woman} \sqcap \mathtt{Person} \\
& \mathtt{Woman} \sqsubseteq \mathtt{Person} \\
& \mathtt{Mother} \equiv \mathtt{Woman} \sqcap \exists \mathtt{hasChild}.\top \\
\text{ABox } \mathcal{A}: \quad & \mathtt{Man}(\mathtt{STEPHEN}). \\
& \neg\mathtt{Man}(\mathtt{MONICA}). \\
& \mathtt{Woman}(\mathtt{JESSICA}). \\
& \mathtt{hasChild}(\mathtt{STEPHEN}, \mathtt{JESSICA}).
\end{aligned}
$$

$$
\begin{aligned}
\mathtt{Man}^{\mathcal{I}_1} &= \{\mathtt{JESSICA}, \mathtt{STEPHEN}\} \\
\mathtt{Woman}^{\mathcal{I}_1} &= \{\mathtt{MONICA}, \mathtt{JESSICA}\} \\
\mathtt{Mother}^{\mathcal{I}_1} &= \emptyset \\
\mathtt{Person}^{\mathcal{I}_1} &= \{\mathtt{JESSICA}, \mathtt{MONICA}, \mathtt{STEPHEN}\} \\
\mathtt{hasChild}^{\mathcal{I}_1} &= \{(\mathtt{STEPHEN}, \mathtt{JESSICA})\}
\end{aligned}
$$

TBox $\mathcal{T}$: $\quad \mathtt{Man} \equiv \neg\mathtt{Woman} \sqcap \mathtt{Person}$

$\mathcal{I}_1 \not\models \mathcal{T}$ $\quad \mathtt{Woman} \sqsubseteq \mathtt{Person}$

$\mathtt{Mother} \equiv \mathtt{Woman} \sqcap \exists \mathtt{hasChild}.\top$

ABox $\mathcal{A}$: $\quad \mathtt{Man}(\mathtt{STEPHEN}).$

$\mathcal{I}_1 \models \mathcal{A}$ $\quad \neg\mathtt{Man}(\mathtt{MONICA}).$

$\mathtt{Woman}(\mathtt{JESSICA}).$

$\mathtt{hasChild}(\mathtt{STEPHEN}, \mathtt{JESSICA}).$

$$
\begin{aligned}
\mathtt{Man}^{\mathcal{I}_1} &= \{\mathtt{JESSICA}, \mathtt{STEPHEN}\} \\
\mathtt{Woman}^{\mathcal{I}_1} &= \{\mathtt{MONICA}, \mathtt{JESSICA}\} \\
\mathtt{Mother}^{\mathcal{I}_1} &= \emptyset \\
\mathtt{Person}^{\mathcal{I}_1} &= \{\mathtt{JESSICA}, \mathtt{MONICA}, \mathtt{STEPHEN}\} \\
\mathtt{hasChild}^{\mathcal{I}_1} &= \{(\mathtt{STEPHEN}, \mathtt{JESSICA})\}
\end{aligned}
$$

Interpretations - Examples

$$
\begin{aligned}
\text{TBox } \mathcal{T}: \quad & \texttt{Man} \equiv \neg\texttt{Woman} \sqcap \texttt{Person} \\
& \texttt{Woman} \sqsubseteq \texttt{Person} \\
& \texttt{Mother} \equiv \texttt{Woman} \sqcap \exists\texttt{hasChild}.\top \\
\text{ABox } \mathcal{A}: \quad & \texttt{Man}(\texttt{STEPHEN}). \\
& \neg\texttt{Man}(\texttt{MONICA}). \\
& \texttt{Woman}(\texttt{JESSICA}). \\
& \texttt{hasChild}(\texttt{STEPHEN}, \texttt{JESSICA}).
\end{aligned}
$$

$$
\begin{aligned}
\texttt{Man}^{\mathcal{I}_2} &= \{\texttt{STEPHEN}\} \\
\texttt{Woman}^{\mathcal{I}_2} &= \{\texttt{JESSICA}, \texttt{MONICA}\} \\
\texttt{Mother}^{\mathcal{I}_2} &= \emptyset \\
\texttt{Person}^{\mathcal{I}_2} &= \{\texttt{JESSICA}, \texttt{MONICA}, \texttt{STEPHEN}\} \\
\texttt{hasChild}^{\mathcal{I}_2} &= \emptyset
\end{aligned}
$$

Interpretations - Examples

$$\begin{array}{ll}
\text{TBox } \mathcal{T}: & \texttt{Man} \equiv \neg\texttt{Woman} \sqcap \texttt{Person} \\
{\color{green}\mathcal{I}_2 \models \mathcal{T}} & \texttt{Woman} \sqsubseteq \texttt{Person} \\
 & \texttt{Mother} \equiv \texttt{Woman} \sqcap \exists\texttt{hasChild}.\top \\
\text{ABox } \mathcal{A}: & \texttt{Man}(\texttt{STEPHEN}). \\
{\color{red}\mathcal{I}_2 \not\models \mathcal{A}} & \neg\texttt{Man}(\texttt{MONICA}). \\
 & \texttt{Woman}(\texttt{JESSICA}). \\
 & \texttt{hasChild}(\texttt{STEPHEN}, \texttt{JESSICA}).
\end{array}$$

$$\begin{aligned}
\texttt{Man}^{\mathcal{I}_2} &= \{\texttt{STEPHEN}\} \\
\texttt{Woman}^{\mathcal{I}_2} &= \{\texttt{JESSICA}, \texttt{MONICA}\} \\
\texttt{Mother}^{\mathcal{I}_2} &= \emptyset \\
\texttt{Person}^{\mathcal{I}_2} &= \{\texttt{JESSICA}, \texttt{MONICA}, \texttt{STEPHEN}\} \\
\texttt{hasChild}^{\mathcal{I}_2} &= \emptyset
\end{aligned}$$

Interpretations - Examples

$$\text{TBox } \mathcal{T}: \quad \mathtt{Man} \equiv \neg\mathtt{Woman} \sqcap \mathtt{Person}$$
$$\mathtt{Woman} \sqsubseteq \mathtt{Person}$$
$$\mathtt{Mother} \equiv \mathtt{Woman} \sqcap \exists\mathtt{hasChild}.\top$$
$$\text{ABox } \mathcal{A}: \quad \mathtt{Man}(\mathtt{STEPHEN}).$$
$$\neg\mathtt{Man}(\mathtt{MONICA}).$$
$$\mathtt{Woman}(\mathtt{JESSICA}).$$
$$\mathtt{hasChild}(\mathtt{STEPHEN}, \mathtt{JESSICA}).$$

$$\mathtt{Man}^{\mathcal{I}_3} = \{\mathtt{STEPHEN}\}$$
$$\mathtt{Woman}^{\mathcal{I}_3} = \{\mathtt{JESSICA}, \mathtt{MONICA}\}$$
$$\mathtt{Mother}^{\mathcal{I}_3} = \{\mathtt{MONICA}\}$$
$$\mathtt{Person}^{\mathcal{I}_3} = \{\mathtt{JESSICA}, \mathtt{MONICA}, \mathtt{STEPHEN}\}$$
$$\mathtt{hasChild}^{\mathcal{I}_3} = \{(\mathtt{MONICA}, \mathtt{STEPHEN}), (\mathtt{STEPHEN}, \mathtt{JESSICA})\}$$

Interpretations - Examples

$$
\begin{array}{rl}
\text{TBox } \mathcal{T}: & \texttt{Man} \equiv \neg\texttt{Woman} \sqcap \texttt{Person} \\
\mathcal{I}_3 \models \mathcal{T} & \texttt{Woman} \sqsubseteq \texttt{Person} \\
& \texttt{Mother} \equiv \texttt{Woman} \sqcap \exists\texttt{hasChild}.\top \\
\text{ABox } \mathcal{A}: & \texttt{Man}(\texttt{STEPHEN}). \\
\mathcal{I}_3 \models \mathcal{A} & \neg\texttt{Man}(\texttt{MONICA}). \\
& \texttt{Woman}(\texttt{JESSICA}). \\
& \texttt{hasChild}(\texttt{STEPHEN}, \texttt{JESSICA}).
\end{array}
$$

$$
\begin{aligned}
\texttt{Man}^{\mathcal{I}_3} &= \{\texttt{STEPHEN}\} \\
\texttt{Woman}^{\mathcal{I}_3} &= \{\texttt{JESSICA}, \texttt{MONICA}\} \\
\texttt{Mother}^{\mathcal{I}_3} &= \{\texttt{MONICA}\} \\
\texttt{Person}^{\mathcal{I}_3} &= \{\texttt{JESSICA}, \texttt{MONICA}, \texttt{STEPHEN}\} \\
\texttt{hasChild}^{\mathcal{I}_3} &= \{(\texttt{MONICA}, \texttt{STEPHEN}), (\texttt{STEPHEN}, \texttt{JESSICA})\}
\end{aligned}
$$

$\mathcal{ALC}$ alternative semantic

- Translation of TBox statements into Predicate Logic by using the mapping function π (right side).
- Let C, D be complex classes, R a role and A an atomic class.

$$
\begin{aligned}
\pi(C \sqsubseteq D) &= (\forall x(\pi_x(C) \rightarrow \pi_x(D)) \\
\pi(C \equiv D) &= (\forall x(\pi_x(C) \leftrightarrow \pi_x(D)) \\
\pi_x(A) &= A(x) \\
\pi_x(\neg C) &= \neg\pi_x(C) \\
\pi_x(C \sqcap D) &= \pi_x(C) \wedge \pi_x(D) \\
\pi_x(C \sqcup D) &= \pi_x(C) \vee \pi_x(D) \\
\pi_x(\forall R.C) &= (\forall y)(R(x,y) \rightarrow \pi_y(C)) \\
\pi_x(\exists R.C) &= (\exists y)(R(x,y) \wedge \pi_y(C)) \\
\pi_y(A) &= A(y) \\
\pi_y(\neg C) &= \neg\pi_y(C) \\
\pi_y(C \sqcap D) &= \pi_y(C) \wedge \pi_y(D) \\
\pi_y(C \sqcup D) &= \pi_y(C) \vee \pi_y(D) \\
\pi_y(\forall R.C) &= (\forall x)(R(y,x) \rightarrow \pi_x(C)) \\
\pi_y(\exists R.C) &= (\exists x)(R(y,x) \wedge \pi_x(C))
\end{aligned}
$$

OWL und $\mathcal{ALC}$

The following OWL DL language constructs are representable in $\mathcal{ALC}$:

- classes, roles, individuals
- class assertion, role assertion
- `owl:Thing` und `owl:Nothing`
- class inclusion, equivalence and disjointness
- `owl:intersectionOf`, `owl:unionOf`
- `owl:complementOf`
- `owl:allValuesFrom`, `owl:someValuesFrom`
- `rdfs:range` und `rdfs:domain`

Important inference problems

Global consistency of the knowledge base — $\mathcal{K} \models \text{false}?$

- Does the knowledge base make sense? (Semantic: Exists a model for $\mathcal{K}$?)

Class consistency — $C \equiv \bot?$

- Has class C to be empty?

Class inclusion (Subsumption) — $C \sqsubseteq D?$

- Structuring of the knowledge base

Class equivalence — $C \equiv D?$

- Are two classes the same?

Class disjointness — $C \sqcap D = \bot?$

- Are two classes disjoint?

Class assertion — $C(a)$?

- Does individual a belong to class C?

Instance generation (Retrieval) "find all x with $C(x)$"

- Find all (known!) individuals of class C.

Outline

Motivation

- increasing number of knowledge bases in the Semantic Web (see e.g. LOD cloud)
- maintenance of knowledge bases with expressive semantics is challenging

⇝ ORE simplifies this task (as part of the LOD2 stack)

Creating Knowledge out of Interlinked Data

Features of the ORE Tool

- open-source tool for repairing and extending knowledge bases
- state-of-the-art inconsistency detection, ranking, and repair methods
- use of supervised machine learning
- support for very large knowledge bases available as SPARQL endpoints

Basics - Ontology

- OWL ontology $\mathcal{O}$ consists of a set of axioms

```
capitalOf SubPropertyOf: locatedIn
BEIJING capitalOf CHINA
```

(note: Manchester OWL Syntax)

Basics - Ontology

- OWL ontology $\mathcal{O}$ consists of a set of axioms

  `capitalOf SubPropertyOf: locatedIn`
  `BEIJING capitalOf CHINA`

  (note: Manchester OWL Syntax)
- semantics of OWL allow to draw entailments

  `BEIJING locatedIn CHINA`

Basics - Ontology

- OWL ontology $\mathcal{O}$ consists of a set of axioms

  `capitalOf SubPropertyOf: locatedIn`
  `BEIJING capitalOf CHINA`

  (note: Manchester OWL Syntax)
- semantics of OWL allow to draw entailments

  `BEIJING locatedIn CHINA`
- justification $J \subseteq \mathcal{O}$ of an entailment is a minimal set of axioms from which the entailment can be drawn

Basics - Inconsistency

$\mathcal{O}$ is inconsistent if it does not have a model ($=$ contains a contradiction):

```
City and population some int[>10000000]
  SubClassOf: capitalOf some Country
SHANGHAI Types: City,
                not(capitalOf some Country)
         Facts: population 13831900
```

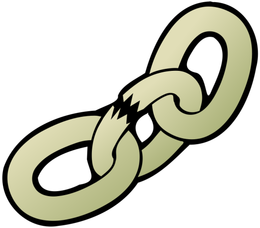
Basics - Unsatisfiable

a class c in $\mathcal{O}$ is unsatisfiable if $C^{\mathcal{I}} = \emptyset$ for all models I of $\mathcal{O}$ ($=$ the class cannot have an instance):

```
ISWCConference SubClassOf: SmallConference
SmallConference SubClassOf:
  not (hasEvent some (Tutorial or Workshop))
ISWConference SubClassOf:
  (hasEvent some Tutorial) and
  (hasEvent some Workshop)
```

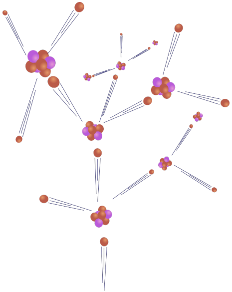
Problems

- there can be many justifications for a single entailment
- there can be several unsatisfiable classes
- due to ontological relations, several problems can be intertwined and are difficult to separate



Root Unsatisfiability

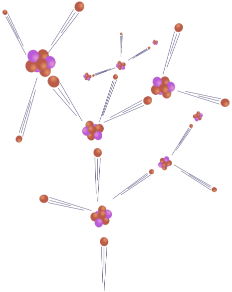
- idea: separate between root and derived unsatisfiable classes
- derived unsatisfiable class has justification, which contains a justification of another unsatisfiable class
- fixing root problems may resolve further problems



"Debugging Unsatisfiable Classes in OWL Ontologies", Kalyanpur, Parsia, Sirin, Hendler, J. Web Sem, 2005.

Root Unsatisfiability

- idea: separate between root and derived unsatisfiable classes
- derived unsatisfiable class has justification, which contains a justification of another unsatisfiable class
- fixing root problems may resolve further problems



- approach 1: compute all justifications for each unsatisfiable class and apply the definition → computationally often too expensive
- approach 2: heuristics for structural analysis of axioms
- ORE uses sound but incomplete variant of approach 2

"Debugging Unsatisfiable Classes in OWL Ontologies", Kalyanpur, Parsia, Sirin, Hendler, J. Web Sem, 2005.

Axiom Relevance

- resolving justification requires to delete or edit axioms
- ranking methods highlight the most probable causes for problems
- methods:
  - frequency
  - syntactic relevance
  - semantic relevance

Axiom Relevance

- resolving justification requires to delete or edit axioms
- ranking methods highlight the most probable causes for problems
- methods:
  - frequency
  - syntactic relevance
  - semantic relevance
- ORE supports those metrics and an aggregation of them

Axiom	F	U	Σ
Lumber SubClassOf ForestProduct	2	15	0.88
Lumber SubClassOf ManufacturedProduct	1	25	0.54
ForestProduct DisjointWith ManufacturedProduct	2	35	0.71

Repair Consequences

- after repairing process, axioms have been deleted or modified

→ desired entailments may be lost or new entailments obtained (including inconsistencies!)

- ORE allows to preview new or lost entailments

→ user can decide to preserve them

Retained	Olive **SubClassOf** OrganicObject
Lost	EdibleNut **DisjointWith** Fruit

SPARQL Endpoint Support

- ORE supports using SPARQL endpoints
- implements an incremental load procedure
- knowledge base is loaded in small chunks:
  - count number of axioms by type
  - priority based loading procedure
  - e.g. disjointness axioms have higher priority than class assertion axioms
- uses Pellet incremental reasoning

"Learning of OWL Class Descriptions on Very Large Knowledge Bases", Hellmann, Lehmann, Auer, Int. Journal Semantic Web Inf. Syst, 2009

SPARQL Endpoint Support II

- algorithm performs sanity checks, e.g. SPARQL queries which probe for typical inconsistent axiom sets
- can fetch additional Linked Data
- different termination criteria

SPARQL Endpoint Support II

- algorithm performs sanity checks, e.g. SPARQL queries which probe for typical inconsistent axiom sets
- can fetch additional Linked Data
- different termination criteria
- overall:
  - ORE allows to apply state-of-the-art ontology debugging methods on a larger scale than was possible previously
  - aims at stronger support for the "web aspect" of the Semantic Web and the high popularity of Web of Data initiative

Outline

induction algorithms can derive axioms about a class by using its instances as positive examples

Basics - Learning

induction algorithms can derive axioms about a class by using its instances as positive examples

Data:

```
ISWC2003 Types: ISWCConference
         Facts: hasTopic SemanticBrokering,
                hasTopic Ontologies
                ...
                takesPlaceIn Florida
```

induction algorithms can derive axioms about a class by using its instances as positive examples

Data:

```
ISWC2003 Types: ISWCConference
         Facts: hasTopic SemanticBrokering,
                hasTopic Ontologies
                ...
                takesPlaceIn Florida
```

Learned:

```
ISWCConference SubClassOf: hasTopic some Ontologies
```

Basics - Learning

induction algorithms can derive axioms about a class by using its instances as positive examples

Data:

```
ISWC2003 Types: ISWCConference
         Facts: hasTopic SemanticBrokering,
                hasTopic Ontologies
                ...
                takesPlaceIn Florida
```

Learned:

```
ISWCConference SubClassOf: hasTopic some Ontologies

ISWCConference SubClassOf: takesPlaceIn some
   (Europe or Asia or NorthAmerica)
```

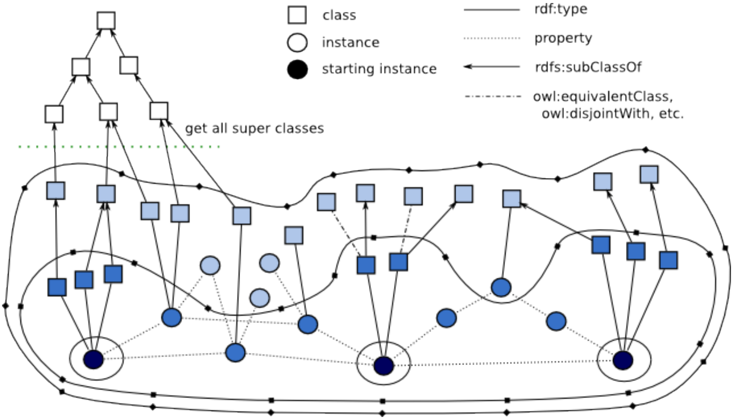
Enrichment - Basics

- DL-Learner is used as machine learning framework for enrichment
- ORE uses DL-Learner (provides an ontology enrichment user interface on top of it)
- ORE supports enriching an ontology with super class axioms and definitions (other algorithms will be supported in the future)
- uses supervised CELOE machine learning algorithm implemented in DL-Learner
- use class instances as positive examples

"DL-Learner: Learning Concepts in Description Logics",
J. Lehmann, Journal of Machine Learning Research, 2009

"Concept Learning in Description Logics Using Refinement Operators",
J. Lehmann, P. Hitzler, Machine Learning journal, 2010

- have to deal with very large knowledge bases → fragment extraction is used



Enrichment - Consequences

- often no perfectly accurate axioms in SW scenarios
- ORE can handle consequences of adding such axioms

Example:

- ORE/CELOE suggests that a "car" always has an "engine" and a "manufacturer":
  `Class: Car SubClassOf: hasPart SOME Engine AND hasManufacturer SOME Company`
- but 3% of the cars do not have that information

→ ORE shows those instances and allows to complete information

Enrichment - Other OWL Axioms

What about other axioms besides complex super classes and definitions?

Enrichment - Other OWL Axioms

What about other axioms besides complex super classes and definitions?

It is possible to learn e.g. the property hierarchy, domains, ranges etc?

How can this be done efficiently?

Enrichment - Other OWL Axioms(2)

General method:

1. Use SPARQL queries to obtain general/schema information about the knowledge base, in particular we retrieve axioms, which allow to construct the class hierarchy.
2. Obtain data via SPARQL, which is relevant for the learning the considered axiom.
3. Compute appropriate score of axiom candidates and return results.

Enrichment - Other OWL Axioms(3)

```
@prefix dbpedia: <http://dbpedia.org/resource/> .
@prefix dbo:     <http://dbpedia.org/ontology/> .
dbpedia:Luxembourg dbo:currency    dbpedia:Euro ;
                   rdf:type        dbo:Country .
dbpedia:Ecuador    dbo:currency    dbpedia:United_States_dollar ;
                   rdf:type        dbo:Country .
dbpedia:Ifni       dbo:currency    dbpedia:Spanish_peseta ;
                   rdf:type        dbo:PopulatedPlace .
dbo:Country        rdfs:subClassOf dbo:PopulatedPlace.
```

Enrichment - Other OWL Axioms(3)

```
@prefix dbpedia: <http://dbpedia.org/resource/> .
@prefix dbo:     <http://dbpedia.org/ontology/> .
dbpedia:Luxembourg dbo:currency     dbpedia:Euro ;
                   rdf:type         dbo:Country .
dbpedia:Ecuador    dbo:currency     dbpedia:United_States_dollar ;
                   rdf:type         dbo:Country .
dbpedia:Ifni       dbo:currency     dbpedia:Spanish_peseta ;
                   rdf:type         dbo:PopulatedPlace .
dbo:Country        rdfs:subClassOf dbo:PopulatedPlace.
```

Query in Phase 2

```
PREFIX dbo: <http://dbpedia.org/ontology/>
SELECT ?type COUNT(DISTINCT ?ind) WHERE {
        ?ind dbo:currency ?o.
        ?ind a ?type.
} GROUP BY ?type
```

Enrichment - Other OWL Axioms(3)

Example: Domain of object property dbo:currency

Query

```
PREFIX dbo: <http://dbpedia.org/ontology/>
SELECT ?type COUNT(DISTINCT ?ind) WHERE {
        ?ind dbo:currency ?o.
        ?ind a ?type.
} GROUP BY ?type
```

```
@prefix dbpedia: <http://dbpedia.org/resource/> .
@prefix dbo:     <http://dbpedia.org/ontology/> .
dbpedia:Luxembourg dbo:currency     dbpedia:Euro ;
                   rdf:type         dbo:Country .
dbpedia:Ecuador    dbo:currency
dbpedia:United_States_dollar ;
                   rdf:type         dbo:Country .
dbpedia:Ifni       dbo:currency
dbpedia:Spanish_peseta ;
                   rdf:type
dbo:PopulatedPlace .
dbo:Country
rdfs:subClassOf dbo:PopulatedPlace.
```

- Score(dbo:Country) = 2/3 = 66,7%
- Score(dbo:PopulatedPlace) = 3/3 = 100% (33,3% without inference)

Example: Domain of object property dbo:currency

Query

```
PREFIX dbo: <http://dbpedia.org/ontology/>
SELECT ?type COUNT(DISTINCT ?ind) WHERE {
        ?ind dbo:currency ?o.
        ?ind a ?type.
} GROUP BY ?type
```

```
@prefix dbpedia: <http://dbpedia.org/resource/> .
@prefix dbo:     <http://dbpedia.org/ontology/> .
dbpedia:Luxembourg dbo:currency     dbpedia:Euro ;
                   rdf:type         dbo:Country .
dbpedia:Ecuador    dbo:currency
dbpedia:United_States_dollar ;
                   rdf:type         dbo:Country .
dbpedia:Ifni       dbo:currency
dbpedia:Spanish_peseta ;
                   rdf:type
dbo:PopulatedPlace .
dbo:Country
rdfs:subClassOf dbo:PopulatedPlace.
```

- Score(dbo:Country) = 2/3 = 66,7%
- Score(dbo:PopulatedPlace) = 3/3 = 100% (33,3% without inference)

Problem: Straightforward score doesn't take the support for an axiom in the knowledge base into account!

Enrichment - Other OWL Axioms(4)

Score method: Average of the 95% confidence interval boundaries (can be computed efficiently using improved Wald method)

Score method: Average of the 95% confidence interval boundaries (can be computed efficiently using improved Wald method)

95% confidence interval (Wald method)

Assume we have m observations out of which s were successful, then the approximation of the 95% confidence interval is as follows:

$$\max(0, p' - 1.96 \cdot \sqrt{\frac{p' \cdot (1 - p')}{m + 4}}) \text{ to } \min(1, p' + 1.96 \cdot \sqrt{\frac{p' \cdot (1 - p')}{m + 4}})$$

$$\text{with } p' = \frac{s + 2}{m + 4}$$

Enrichment - Other OWL Axioms(4)

Score method: Average of the 95% confidence interval boundaries (can be computed efficiently using improved Wald method)

95% confidence interval (Wald method)

Assume we have m observations out of which s were successful, then the approximation of the 95% confidence interval is as follows:

$$\max(0, p' - 1.96 \cdot \sqrt{\frac{p' \cdot (1-p')}{m+4}}) \text{ to } \min(1, p' + 1.96 \cdot \sqrt{\frac{p' \cdot (1-p')}{m+4}})$$

$$\text{with } p' = \frac{s+2}{m+4}$$

- toy example: score(`dbo:Country`) = 57.3% (2 of 3)
- toy example: score(`dbo:PopulatedPlace`) = 69.1% (3 of 3)

Enrichment - Other OWL Axioms(4)

Score method: Average of the 95% confidence interval boundaries (can be computed efficiently using improved Wald method)

95% confidence interval (Wald method)

Assume we have m observations out of which s were successful, then the approximation of the 95% confidence interval is as follows:

$$\max(0, p' - 1.96 \cdot \sqrt{\frac{p' \cdot (1 - p')}{m + 4}}) \text{ to } \min(1, p' + 1.96 \cdot \sqrt{\frac{p' \cdot (1 - p')}{m + 4}})$$

$$\text{with } p' = \frac{s + 2}{m + 4}$$

- toy example: score(`dbo:Country`) = 57.3% (2 of 3)
- toy example: score(`dbo:PopulatedPlace`) = 69.1% (3 of 3)
- DBpedia Live: score(`dbo:Country`) = 98.5% (603 of 610)
- DBpedia Live: score(`dbo:PopulatedPlace`) = 97.1% (594 of 610)

Outline

To install DL-Learner, perform the following steps:

- install Java version 6 or higher (`http://www.java.com/en/download/`)
- go to `http://dl-learner.org` and press “download”
- extract the downloaded archive
- run `cli/enrichment -?` (Unix) or `cli/enrichment.bat -?` (Windows) on the command line in the extracted directory

DL-Learner - Enrichment - Usage

```
Option                        Description
------                        -----------
-?, -h, --help                Show help.
-e, --endpoint <URL>          SPARQL endpoint URL to be used.
-f, --format                  Format of the generated output (plain,
                                rdf/xml, turtle, n-triples).
                                (default: plain)
-g, --graph [URI]             URI of default graph for queries on
                                SPARQL endpoint.
-i, --inference [Boolean]     Specifies whether to use inference. If
                                yes, the schema will be loaded into
                                a reasoner and used for computing
                                the scores. (default: true)
-o, --output [File]           Specify a file where the output can be
                                written.
-r, --resource [URI]          The resource for which enrichment
                                axioms should be suggested.
-t, --threshold [Double]      Confidence threshold for suggestions.
                                Set it to a value between 0 and 1.
                                (default: 0.7)
```

DL-Learner - Enrichment - Usage (2)

Obtain enrichment suggestions for the `currency` property in DBpedia Live:

```
-e http://live.dbpedia.org/sparql -g http://dbpedia.org
-r http://dbpedia.org/ontology/currency
```

Output those enrichments to a file `results.txt`:

```
-e http://live.dbpedia.org/sparql -g http://dbpedia.org
-r http://dbpedia.org/ontology/currency -o results.txt
```

Write the enrichments in Turtle syntax in a file using the enrichment ontology:

```
-e http://live.dbpedia.org/sparql -g http://dbpedia.org
-r http://dbpedia.org/ontology/currency -o results.ttl
-f turtle
```

Do the same task with an increased threshold and without inference

```
-e http://live.dbpedia.org/sparql -g http://dbpedia.org
-r http://dbpedia.org/ontology/currency -o results.ttl
-f turtle -t 0.9 -i false
```

DL-Learner - Enrichment - Usage (3)

lorenz@lorenz-ThinkPad-W510: ~/Downloads/dllearner-1.0-alpha-1

Datei Bearbeiten Ansicht Suchen Terminal Hilfe

```
lorenz@lorenz-ThinkPad-W510:~/Downloads/dllearner-1.0-alpha-1$ ./bin/enrichment -e http://live.dbpedia.org/sparql -g
 http://dbpedia.org -r http://dbpedia.org/ontology/currency -i false -t 0.7
http://dbpedia.org/ontology/currency appears to be an object property. Running appropriate algorithms.

Applying disjoint objectproperty axiom learner on http://dbpedia.org/ontology/currency ... done in 274 ms
suggested axioms and their score in percent:
 100,0% dbo:currency DisjointWith: dbo:academicAdvisor
 100,0% dbo:currency DisjointWith: dbo:academyAward
 100,0% dbo:currency DisjointWith: dbo:actingHeadteacher
 100,0% dbo:currency DisjointWith: dbo:administrativeCollectivity
 100,0% dbo:currency DisjointWith: dbo:administrativeDistrict
 100,0% dbo:currency DisjointWith: dbo:administrator
 100,0% dbo:currency DisjointWith: dbo:affiliation
 100,0% dbo:currency DisjointWith: dbo:afiAward
 100,0% dbo:currency DisjointWith: dbo:aircraftAttack
 100,0% dbo:currency DisjointWith: dbo:aircraftBomber

Applying equivalent objectproperty axiom learner on http://dbpedia.org/ontology/currency ... done in 56 ms
  no axiom suggested

Applying functional objectproperty axiom learner on http://dbpedia.org/ontology/currency ... done in 34 ms
suggested axioms and their score in percent:
  75,9% Functional(dbo:currency)

Applying inversefunctional objectproperty axiom learner on http://dbpedia.org/ontology/currency ... done in 31 ms
  no axiom suggested

Applying objectproperty domain axiom learner on http://dbpedia.org/ontology/currency ... done in 1603 ms
suggested axioms and their score in percent:
  98,5% Domain(dbo:currency, dbo:PopulatedPlace)
  98,5% Domain(dbo:currency, dbo:Place)
  97,1% Domain(dbo:currency, dbo:Country)

Applying objectproperty range learner on http://dbpedia.org/ontology/currency ... done in 183 ms
  no axiom suggested

Applying object subPropertyOf axiom learner on http://dbpedia.org/ontology/currency ... done in 53 ms
```

ORE - Demo

- go to `http://web.ore-tool.net` [Warning: alpha version!]

ORE - Demo

- go to `http://web.ore-tool.net` [Warning: alpha version!]
- (1) click on `Bookmark` >> `Koala` and `Action` >> `Debug`
- have a look at the generated justifications

ORE - Demo

- go to `http://web.ore-tool.net` [Warning: alpha version!]
- (1) click on `Bookmark` >> `Koala` and `Action` >> `Debug`
- have a look at the generated justifications
- (2) click on `Bookmark` >> `Swore` and `Action` >> `Learn`
- click on `CustomerRequirement` and have a look at the learned definitons

ORE - Demo

- go to `http://web.ore-tool.net` [Warning: alpha version!]
- (1) click on `Bookmark >> Koala and Action >> Debug`
- have a look at the generated justifications
- (2) click on `Bookmark >> Swore and Action >> Learn`
- click on `CustomerRequirement` and have a look at the learned definitons
- (3) click on `File >> Connect to SPARQL Endpoint`
- try `http://live.dbpedia.org/sparql` with default graph `http://dbpedia.org`
- note: only CELOE integrated in ORE at the moment, many algorithms on our TODO list ...

Inconsistency in DBpedia Live:

```
Individual: dbr:Purify_(album)
  Facts: dbo:artist dbr:Axis_of_Advance
Individual: dbr:Axis_of_Advance
  Types: dbo:Organisation
Class: dbo:Organisation
  DisjointWith dbo:Person
ObjectProperty: dbo:artist
  Range: dbo:Person
```

Inconsistency in DBpedia in combination with WGS84 (Linked Data):

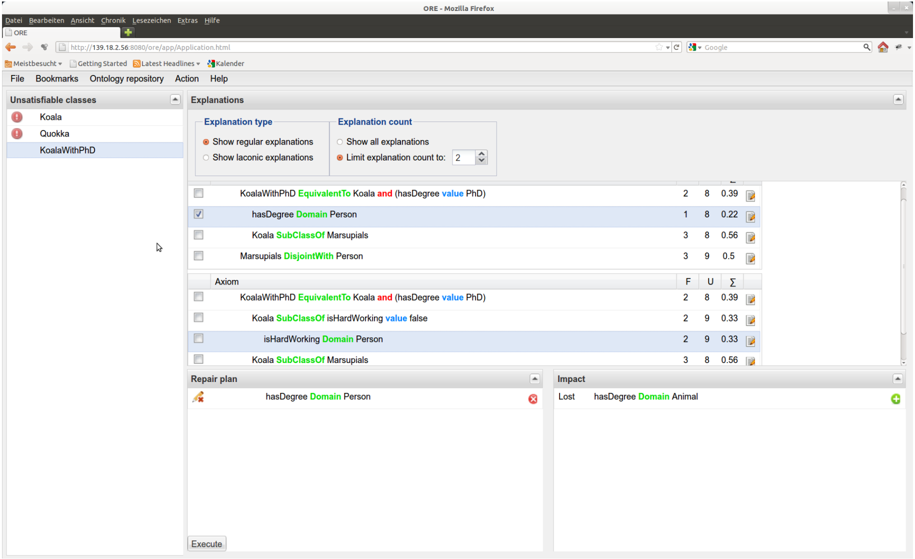
```
Individual: dbr:WKWS Facts: geo:long -81.76833343505859
                     Types: dbo:Organisation
DataProperty: geo:long Domain: geo:SpatialThing
Class: dbo:Organisation DisjointWith: geo:SpatialThing
```

Inconsistency in OpenCyc:

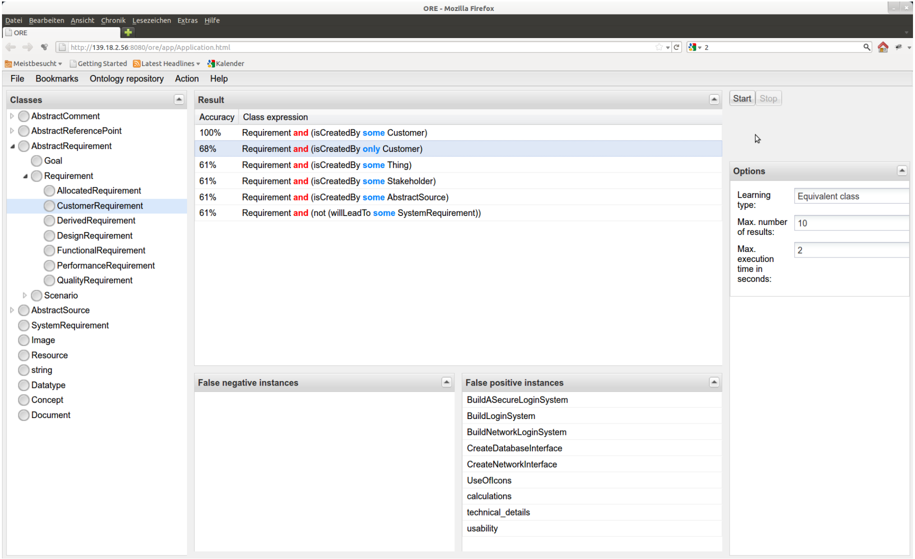
```
Individual: 'PopulatedPlace'
  Types: 'ArtifactualFeatureType', 'ExistingStuffType'
Class: 'ArtifactualFeatureType'
  SubClassOf: 'ExistingObjectType'
Class: 'ExistingObjectType'
  DisjointWith: 'ExistingStuffType'
```

OpenCyc.org

ORE - Screenshot



ORE - Screenshot



Outline

Related Tools

- Swoop
  - can compute justifications for unsatisfiability of classes and offers repair mode
  - fine-grained justification computation algorithm is incomplete
  - can also compute justifications for an inconsistent ontology, but does not offer repair mode in this case
  - does not extract locality-based modules, which leads to lower performance for large ontologies
- RaDON
  - plugin for the NeOn toolkit
  - offers a number of techniques for working with inconsistent or incoherent ontologies
  - allows to reason with inconsistent ontologies and can handle sets of ontologies (ontology networks)
  - no fine-grained justifications, no repair impact analysis
- Pellint
  - searches for common patterns which lead to potential reasoning performance problems
  - integration in ORE planned

Related Tools II

- PION and DION
  - developed in the SEKT project to deal with inconsistencies
  - PION is an inconsistency tolerant reasoner (four-valued paraconsistent logic)
  - DION offers the possibility to compute justifications, but no repair
- Explanation Workbench
  - Protégé plugin for reasoner requests like class unsatisfiability or inferred subsumption relations
  - can compute regular and laconic justifications
  - motivated the ORE debugging interface
  - current version of Explanation Workbench does not allow to remove axioms in laconic justifications
- RepairTab
  - supports the user in finding and detecting errors in ontologies
  - RepairTab uses a modified tableau algorithm
  - shows inferences which can no longer be drawn after removing an axiom (inspired ORE)

Outline

Limitations and Future Work

- ORE and DL-Learner development continues within the LOD2 project
- DL-Learner with enrichment support released two weeks ago (alpha)
- ORE 0.2 release is a desktop application (mostly for local OWL files)
- next release will include a web version (live prototype on web.ore-tool-net)
- support for detection of further modeling problems
- improving Linked Data / SPARQL component - debugging LOD knowledge bases will be a major use case
- constant evolution/extension of underlying methods, e.g. automatic lemma generation, proofs, textual justifications

Creating Knowledge out of Interlinked Data

Conclusions

- OWL based on model theoretic semantics
- ORE and DL-Learner are open source tools for ontology repair and enrichment
- support re-active ontology engineering (see ISSLOD talk by Vojtech Svatek)
- ORE uses state-of-the-art ontology debugging and learning methods
- combines advantages of different existing tools and methods
- will be able to detect modeling problems in very large knowledge bases
- long term goal: build a bridge between the current "Web of Data" and expressive OWL semantics

ORE

http://ore-tool.net

AKSW/MOLE Group, University of Leipzig

Creating Knowledge out of Interlinked Data