

*ORE - A Tool for Repairing  
and Enriching Knowledge Bases*  
*<http://ore-tool.net>*

Jens Lehmann, Lorenz Bühmann

UNIVERSITÄT LEIPZIG

2010-11-10

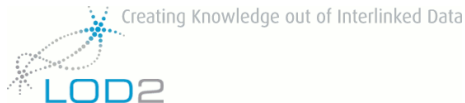
- 1 *Introduction*
- 2 *Basic Notions*
- 3 *Ontology Repair*
- 4 *Ontology Enrichment*
- 5 *Test Results / Live Demo*
- 6 *Conclusions and Future Work*

# Outline

- 1 *Introduction*
- 2 *Basic Notions*
- 3 *Ontology Repair*
- 4 *Ontology Enrichment*
- 5 *Test Results / Live Demo*
- 6 *Conclusions and Future Work*

# 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)



# Contribution

- 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



# Outline

- 1 *Introduction*
- 2 *Basic Notions*
- 3 *Ontology Repair*
- 4 *Ontology Enrichment*
- 5 *Test Results / Live Demo*
- 6 *Conclusions and Future Work*

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

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

(note: Manchester OWL Syntax used in this presentation)

- OWL **ontology**  $\mathcal{O}$  consists of a **set of axioms**  
capitalOf SubPropertyOf: locatedIn  
BEIJING capitalOf CHINA  
(note: Manchester OWL Syntax used in this presentation)
- semantics of OWL allow to draw **entailments**  
BEIJING locatedIn CHINA

- OWL **ontology**  $\mathcal{O}$  consists of a **set of axioms**  
capitalOf SubPropertyOf: locatedIn  
BEIJING capitalOf CHINA  
(note: Manchester OWL Syntax used in this presentation)
- 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 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 in  $\mathcal{O}$  is **unsatisfiable** if it cannot have an instance:

```
ISWCConference SubClassOf: SmallConference
```

```
SmallConference SubClassOf:
```

```
  not (hasEvent some (Tutorial or Workshop))
```

```
ISWConference SubClassOf:
```

```
  (hasEvent some Tutorial) and
```

```
  (hasEvent some Workshop)
```



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

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

# 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

# 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)

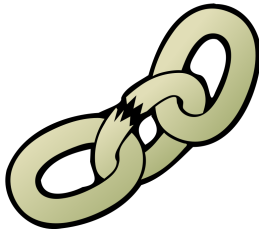


# Outline

- 1 *Introduction*
- 2 *Basic Notions*
- 3 *Ontology Repair***
- 4 *Ontology Enrichment*
- 5 *Test Results / Live Demo*
- 6 *Conclusions and Future Work*

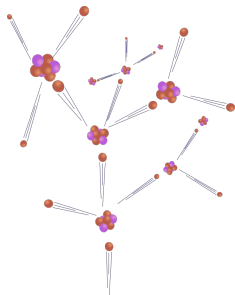
# Problems

- 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

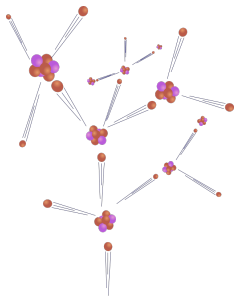
- idea: separate between **root and derived unsatisfiable classes**
- derived unsatisfiable class has justification, which contains a justification of another unsatisfiable class
- **fixing root problems may resolve further problems**



*"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  | $\Sigma$ |
|---|---|----|----------|
| Lumber <b>SubClassOf</b> ForestProduct                | 2 | 15 | 0.88     |
| Lumber <b>SubClassOf</b> ManufacturedProduct          | 1 | 25 | 0.54     |
| ForestProduct <b>DisjointWith</b> 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

- 1 *Introduction*
- 2 *Basic Notions*
- 3 *Ontology Repair*
- 4 *Ontology Enrichment***
- 5 *Test Results / Live Demo*
- 6 *Conclusions and Future Work*

- ORE supports enriching an ontology with super class axioms and definitions
- uses supervised CELOE machine learning algorithm implemented in DL-Learner framework
- 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

# 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”
  - but 3% of the cars do not have that information
- ORE shows those instances and allows to complete information



# Outline

- 1 *Introduction*
- 2 *Basic Notions*
- 3 *Ontology Repair*
- 4 *Ontology Enrichment*
- 5 *Test Results / Live Demo***
- 6 *Conclusions and Future Work*

## *Repair Step Evaluation*

| ontology           | inconsist. | resolved<br>unsat. | axioms<br>removed | axioms<br>retained | axioms<br>changed |
|--------------------|------------|--------------------|-------------------|--------------------|-------------------|
| camera.owl         | yes        | -                  | 0                 | 0                  | 2                 |
| koala.owl          | -          | 3                  | 3                 | 0                  | 0                 |
| Economy.owl        | -          | 51                 | 11                | 5                  | 0                 |
| UnsatCook.owl      | -          | 8                  | 1                 | 0                  | 0                 |
| pizza.owl          | -          | 2                  | 3                 | 0                  | 0                 |
| University.owl     | -          | 9                  | 3                 | 1                  | 2                 |
| Transportation.owl | -          | 62                 | 15                | 28                 | 0                 |

## *Enrichment Step Evaluation*

| ontology          | #axioms | #suggestions | accept in % | reject in % |
|-------------------|---------|--------------|-------------|-------------|
| SC.owl            | 20081   | 12           | 79          | 21          |
| Finance.owl       | 16057   | 50           | 52          | 48          |
| biopax-level2.owl | 12381   | 34           | 78          | 22          |
| GeoSkills.owl     | 8803    | 180          | 56          | 44          |
| Economy.owl       | 1625    | 22           | 74          | 26          |
| MDM0.73.owl       | 884     | 77           | 56          | 44          |
| eukariotic.owl    | 38      | 8            | 91          | 9           |

# *DEMO*

`http://web.ore-tool.net`

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'

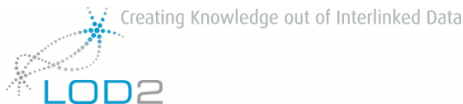


# Outline

- 1 *Introduction*
- 2 *Basic Notions*
- 3 *Ontology Repair*
- 4 *Ontology Enrichment*
- 5 *Test Results / Live Demo*
- 6 *Conclusions and Future Work*

## Limitations and Future Work

- ORE development will continue within the **LOD2 project**
- ORE 0.5 module based with several interfaces
- support for detection of **further modeling problems**
- further enrichment suggestions, e.g. disjointness axioms
- 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



# Conclusions

- ORE is **open source**
- uses **state-of-the-art ontology debugging and learning methods**
- combines advantages of different existing tools and methods
- can detect modeling problems in **very large knowledge bases**
- long term goal: build a **bridge between the current “Web of Data” and expressive OWL semantics**



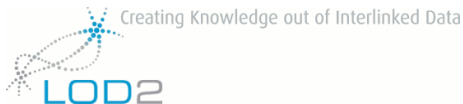


*The End*



`http://ore-tool.net`

AKSW/MOLE Group, University of Leipzig



- 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

- 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)