



Practical Reasoning for the Semantic Web

Stefan Schlobach, Heiner Stuckenschmidt

Knowledge Representation and Reasoning Group

Vrije Universiteit Amsterdam

schlobac@few.vu.nl

heiner@cs.vu.nl

<http://www.cs.vu.nl/~schlobac/>

<http://www.cs.vu.nl/~heiner/>

The Course

- **Goal:**
 - Discuss characteristics of web-based knowledge representation
 - Present new methods trying to cope with these characteristics
- **Scope:**
 - Classical representations and reasoning (logic-based)
 - Selected approaches to non-classical, i.e. practical reasoning, co-developed by the presenters

Practical SW reasoning @VU

- Some Examples of Projects at the Vrije Universiteit
- I-CATCHER (Intensive care)
- STITCH (Cultural heritage)
- BEST (Legal advice)

I-Catcher



I-Catcher

- Purpose: Evaluation of IC units in the NL
- DICE: a terminological system for diagnosis at admission time.
- DICE: represented as a Frame-based ontology
- DL reasoning is used for classification, consistency checking
- Approximate reasoning needed for online application.

Practical reasoning with DICE

- DICE is translated into DL
- There is lots of reasoning:
 - Classical: classify and check for consistency
 - Practical:
 - debugging of errors
 - explain classification
 - Detecting equivalence, redundancy etc.
 - Approximation
 - Online admission (anytime behaviour)

STITCH



STITCH

- SemanTic InTeroperability in Cultural Heritage world.
- Just started in July 2005.
- Goal: Access to distributed collections
- Problem: heterogeneous sources
- Fortunately, there is meta-data; in form of thesauri, catalogues, ontologies, ...

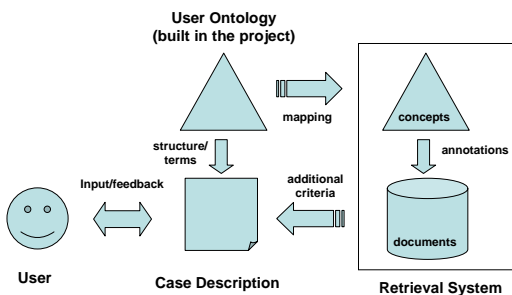
Reasoning in STITCH

- Standard reasoning: checking representations & classification
- Non-standard reasoning:
 - Modeling support (semanticising annotations)
 - Mapping meta-data & annotation schemas
 - Merging collections and thesauri
 - Distributed reasoning (e.g. an object can „mean“ two different things in two thesauri).

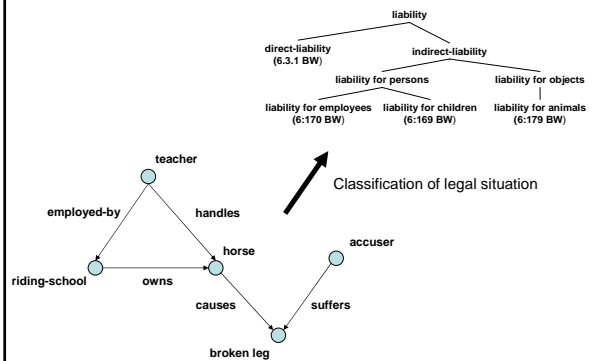
The BEST Project

- Build a system that supports the user in determining liability and possible compensation by providing advanced search facilities for legal documents
- Necessary Steps:
 - Domain analysis, resulting in light-weight legal ontology for damage disputes
 - Annotation of large corpus of case-law documents
 - Implementation of guided search and navigation interface

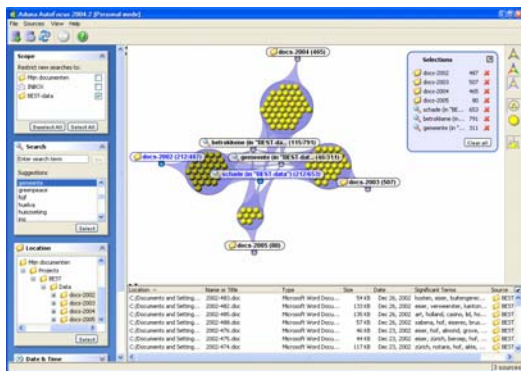
Conceptual Architecture



Reasoning in BEST



Case Retrieval



What makes the Semantic Web Special ?

- KBS used to be developed top down, the Semantic Web is developed bottom-up, this causes problems:
 - Scalability: Everybody having a web page is a potential Knowledge Engineer
 - Distribution: Pieces of knowledge can be all over the place, finding them can be hard
 - Heterogeneity: People use different ways of describing the same information
 - Quality: Knowledge will be incomplete and even inconsistent across different sources

Challenges for KR

- New methods are needed that
 - Scale to large amounts of data and knowledge
 - Are tolerant to errors, incompleteness and inconsistency between and within sources
 - Work on distributed representations and ideally are distributed themselves
- (...and are compatible with Semantic web standards)

Methods discussed in this course

1. Approximating classical logical reasoning to increase performance and to cope with inconsistencies
2. Diagnosing and repairing erroneous and inconsistent knowledge bases
3. Representing and reasoning with heterogeneous representations

Plan for this week

- **Monday:**
 - Overview, what is the semantic web, semantic web languages: RDF, RDF Schema, OWL, reasoning tasks
- **Tuesday:**
 - Description logics (DL) as a basis for KR&R on the semantic web, reasoning in DLs
- **Wednesday:**
 - Approximate reasoning in DLs: classification, instance retrieval, reasoning with inconsistencies
- **Thursday:**
 - Analysing ontologies, Diagnosing and repairing inconsistent ontologies
- **Friday:**
 - Extending OWL with mappings, Distributed Description Logics, Comparison of mapping approaches

The current Web



- WWW is an impressive success:
 - amount of available information (3.3 Giga-page)
 - number of web-servers (30 million)
 - number human users (500 million)

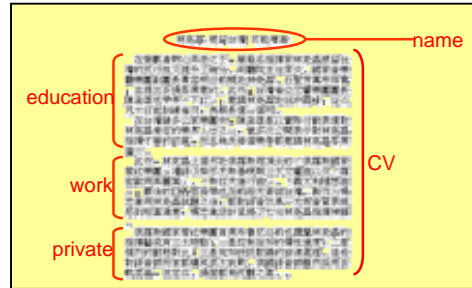


Semantic Web: the Vision

- We have seen two generations of the Web:
 - ① handwritten HTML
 - ② database generated pages
 } human readers
- The real power will come with the 3rd generation:
 - ③ machine accessible semantics

machine accessible meaning

(What it's like to be a machine)



Problem of the current web: Term Ambiguity



Problem of the current Web: Media Objects

- When looking for pictures about 'van Harmelen' you get:

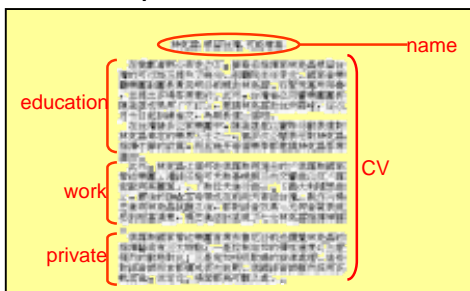
– Various things related to Frank van Harmelen



– And some things you really would not expect:



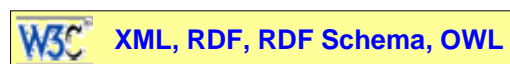
What is required to make this work?



Meta-data !

Required are:

- a **standard syntax**,
 - so meta-data can be recognized as such



- one or more **standard vocabularies**
 - so search engines, producers and consumers all speak the same language
- lots of resources with **meta-data attached**

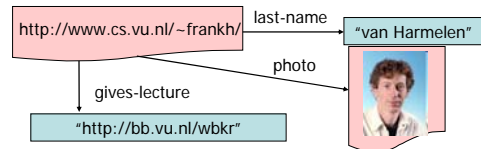
Short Break

After the Break: RDF and RDF Schema

See you back in 5 minutes

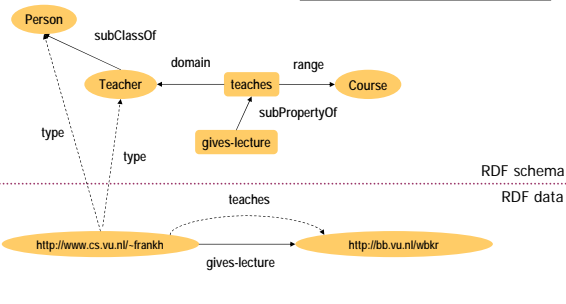
RDF: Resource Description Framework

- RDF is a **data model**
 - used to describe **meta-data** of a piece of data
 - not a language, like XML
 - although it has an XML **syntax** (but also other)
- Benefits:
 - Unique representations of content objects
 - Explicit relations between resources



RDF Schema

- $(X R Y), (R \text{ subPropertyOf } Q) \rightarrow (X Q Y)$
- $(X R Y), (R \text{ domain } C) \rightarrow (X \text{ type } C)$
- $(X \text{ type } C), (C \text{ subClassOf } D) \rightarrow (X \text{ type } D)$

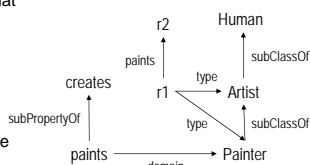


RDF Schema Reasoning

- Existential
 - $m(G) \leftarrow G$ for each map m
- SubPropertyOf
 - $(xxx \text{ subPropertyOf } xxx) \leftarrow (xxx \text{ type Property})$
 - $(xxx, ppp, yyy) \leftarrow (xxx, qqq, yyy), (qqq \text{ subPropertyOf } ppp)$
 - $(xxx \text{ subProperty } zzz) \leftarrow (xxx \text{ subPropertyOf } yyy), (yyy \text{ subPropertyOf } zzz)$
- SubClassOf
 - $(xxx \text{ subClassOf } xxx) \leftarrow (xxx \text{ type Class})$
 - $(xxx \text{ subClassOf } zzz) \leftarrow (xxx \text{ subClassOf } yyy), (yyy \text{ subClassOf } zzz)$
 - $(xxx \text{ type } zzz) \leftarrow (xxx \text{ type } yyy), (yyy \text{ subClassOf } zzz)$
- Type
 - $(xxx \text{ type } zzz) \leftarrow (xxx \text{ type } yyy), (yyy \text{ domain } zzz)$
 - $(xxx \text{ type } zzz) \leftarrow (zzz \text{ type } xxx), (yyy \text{ range } zzz)$
- For each rule $B \leftarrow A$ above define $G \rightarrow G \cup m(B)$ if there is a map $m: A \rightarrow G$
- Also define $G \rightarrow G'$ if G' is a subgraph of G
- $G_1 \Rightarrow G_2$ if and only if $G_1 \rightarrow G_2$

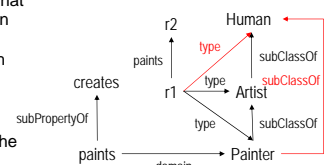
Closure and Reduction

- The **closure** of a graph is the graph defined by all triples that are inferred by the deduction system
- Using the **closure** of a graph for storing **minimizes query processing time**
- the **reduction** of a graph is the minimal subset of triples needed to compute its closure
- using the **reduction** of a graph for storing **minimizes the storage space needed**.



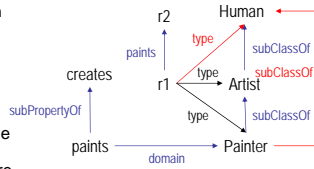
Closure and Reduction

- The **closure** of a graph is the graph defined by all triples that are inferred by the deduction system
- Using the **closure** of a graph for storing **minimizes query processing time**
- the **reduction** of a graph is the minimal subset of triples needed to compute its closure
- using the **reduction** of a graph for storing **minimizes the storage space needed**.



Closure and Reduction

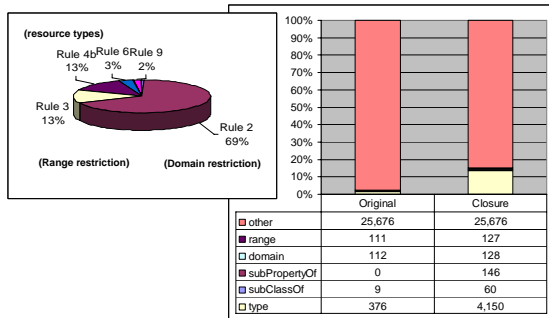
- The **closure** of a graph is the graph defined by all triples that are inferred by the deduction system
- Using the **closure** of a graph for storing **minimizes query processing time**
- the **reduction** of a graph is the minimal subset of triples needed to compute its closure
- using the **reduction** of a graph for storing **minimizes the storage space needed**.



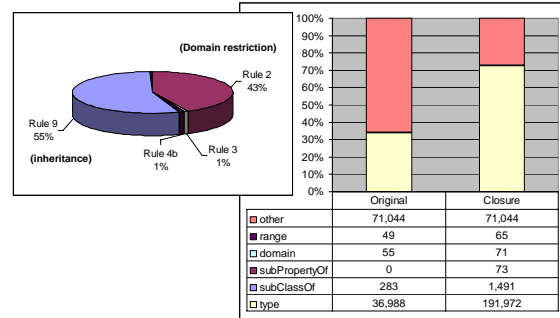
Empirical Analysis

- What kinds of statements are actually added to the closure ?
- Which inference rule contribute to the closure ?
- Is there a part of the closure that should be computed offline ?
- Criteria:
 - reduction of space requirement
 - Support for online reasoning process
- Look at behavior on realistic data
 - CIA World Fact Book (mostly instances, simple schema)
 - TAP KB (some hierarchies with many instances)
 - SUMO (rich schema structures, very few instances)
 - WordNet (huge hierarchy, no instances)

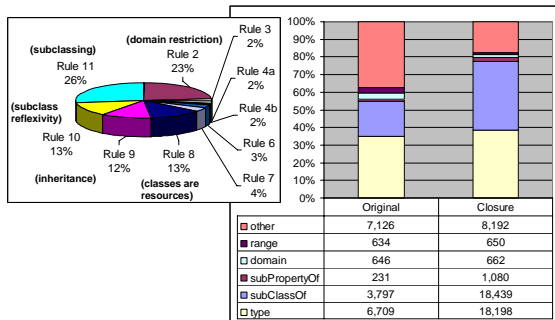
Results for CIA World Fact Book



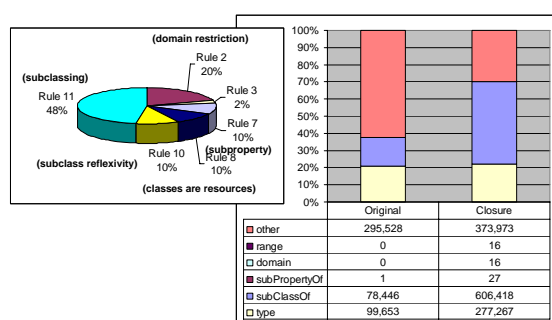
Results for TAP KB



Results for SUMO



Results for WordNet



Conclusions

- The greatest potential in saving closure space is in type and subclass statements
- Type relations show a significant increase in all models
- Once the transitive closure of the hierarchies are known, type statements can easily be computed online

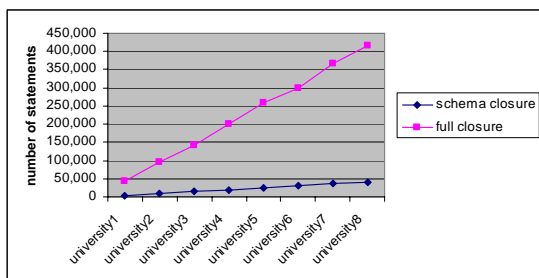
	type	subClassOf	overall
CIA Fact Book	11.03	6.67	1.15
TAP KB	5.19	5.27	2.24
Teknowlegde	2.71	4.86	2.47
Wordnet	2.78	7.73	2.66
Average	6.13	6.13	2.18

- Summary:**
 - We can reduce the size of the closure by leaving out type statements that follow from the model and computing them online.

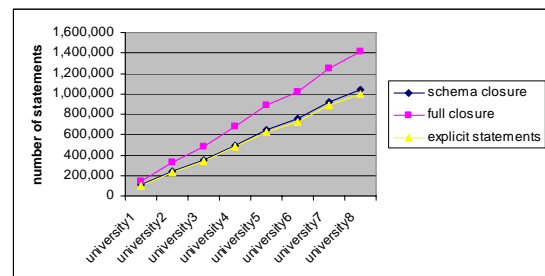
Experimental Evaluation

- Compare performance of the method on a benchmark dataset
 - What is the benefit of using the restricted closure ?
- Lehigh University Benchmark:
 - Fixed schema about universities
 - Data generator to produce an arbitrary amount of instances
- Test data:
 - Data for 10 universities, about 100.000 statements each

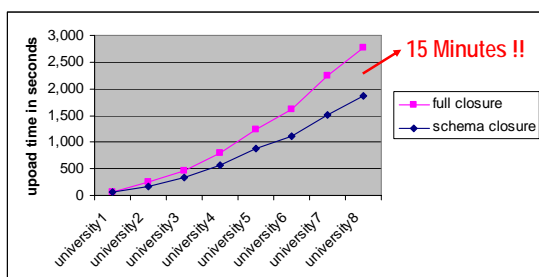
Number of Inferred Statements



Overall Model Size



Upload Time



Discussion on RDF

- RDF is a simple, but useful language to make relations between resources explicit
- It allows to define a simple form of schema which mostly consists of class and property hierarchies as well as relations with domain and range restriction
- There is a simple rule based reasoning mechanism for RDF schema that in most cases can be used offline because the increase is only factor 1.5 – this can further be reduced by only completing the schema offline
- Unlike most traditional KR languages RDF schema allows to freely combine modeling primitives which affects reasoning

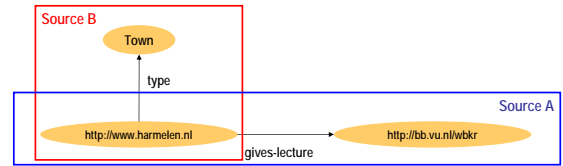
Short Break

After the break:
Richer Languages

See you back in 5 minutes

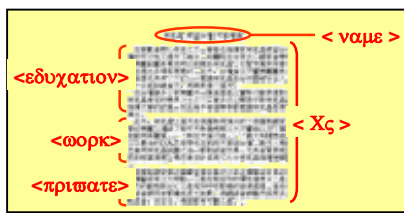
Problems with RDF

- No restriction to **meaningful** statements (the following is legal RDF Schema)



- We need a way to detect statements that do not make sense !

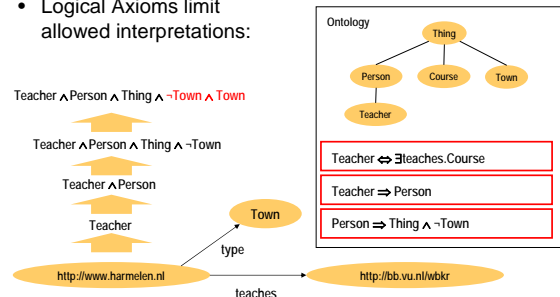
Richer meta-data



- $\langle \text{ναμε} \rangle$ is unique
- every $\langle \text{Χζ} \rangle$ has one $\langle \text{ωορκ} \rangle$,
- one $\langle \text{ωορκ} \rangle$ can belong to multiple $\langle \text{Χζ} \rangle$'s
- $\langle \text{επιπερατιν} \rangle$ is a part-of $\langle \text{ωορκ} \rangle$

Logical Reasoning about Resources

- Logical Axioms limit allowed interpretations:



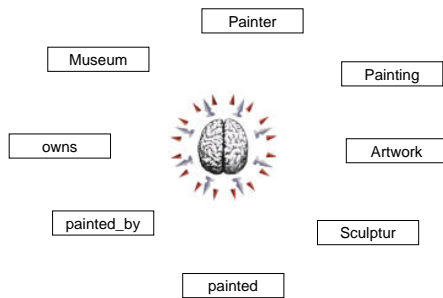
What are elements of a good language for ontologies in the Semantic Web?

We need to know the representation, before we can do reasoning...

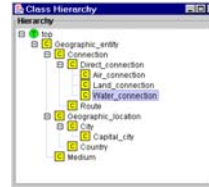
Ontology Engineering

- **Class (concept)**
 - a name and a set of properties that describe a certain set of individuals in the domain
- **Property (slot/role)**
 - assert general facts about the members of a class
- **Instances**
 - the members of the sets defined by classes

Important terms of the domain



Classes and their Hierarchies



C1 subclass of C2
=>

all members of C1 are also members of C2

C1 broader than C2
=>

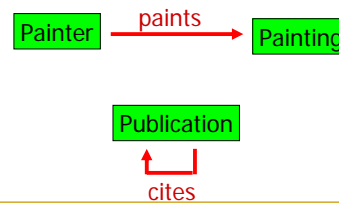
All documents about C1 are also about C2

Issues on class hierarchy

- are hierarchical relations isa?
 - $Inst(B) \subset Inst(A)$
- check transitivity:
 - C Subclass_of(A)
 - D Subclass_of(C) } \Rightarrow D Subclass_of(A)
- Are there cycles
 - B Subclass_of(A)
 - A Subclass_of(B) } \Rightarrow A=B

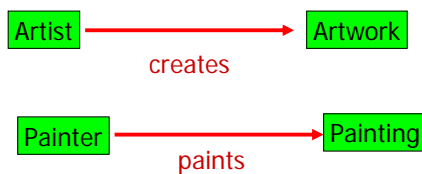
Properties of properties

- Domain: the class on which a property acts
- Range: the class of values of a property



Property hierarchy

- P sub-property of Q -> if P(x,y) then Q(x,y)



- paints is a subproperty of creates

Set operators on classes

- Intersection (logical conjunction)
 - $Painter \cap Writer = \{x \mid x \subset Painter \text{ AND } x \subset Writer\}$
- Union (logical disjunction)
 - $Painter \cup Writer = \{x \mid x \subset Painter \text{ OR } x \subset Writer\}$
- Complement
 - $complementOf(III, Healthy)$
- Disjoint
 - $disjointWith(Lung, Liver, Kidney)$

Cardinality restrictions on properties

- Restricting the number of values of a property



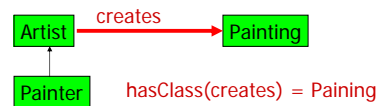
- Cardinality(has_location) = 1 :



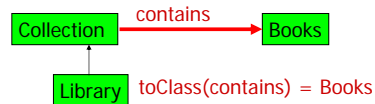
- Min_Cardinality(creates) = 1 :

Property quantification

- Existential quantifier:

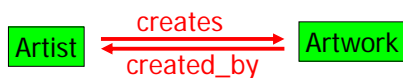


- Universal quantifier:



Property characteristics

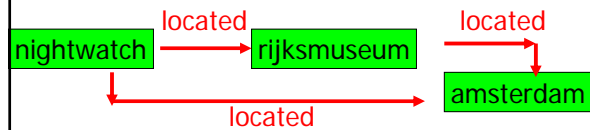
- Inverse properties: $P(x,y) \leftrightarrow Q(y,x)$



- Symmetric properties: $P(x,y) \rightarrow P(y,x)$
 - borders(Liver,Gall) \rightarrow borders(Gall,Liver)

Property characteristics

- Transitivity: $P(x,y) \ \& \ P(y,z) \rightarrow P(x,z)$



- Functionality: $P(x,y) \ \text{and} \ P(x,z) \rightarrow y=z$
 - same as Cardinality(P) = 1

Instances and relations on instances

- Instances

- Painter(rembrandt)
- Museum(rijksmuseum)
- Library(kb)

- Relations

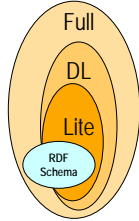
- painted(rembrandt,nightwatch)
- painted_by(nightwatch,rembrandt)

OWL: the W3C's ontology language

- Fact stating facilities from RDF,
- class property structuring from RDF Schema, and
- extends by logical operators.
 - Boolean operators
 - property hierarchies
 - properties can be defined transitive, functional, inverse...
 - individuals can be defined instances
 - equivalence and disjointness statements on classes
 - equivalence statements on properties
 - equality and inequality can be asserted between individuals

The Web Ontology Language OWL

- **OWL Lite:**
 - Classification hierarchy
 - Simple constraints
- **OWL DL:**
 - Maximal expressiveness
 - While maintaining tractability
 - Standard formalisation in DL
- **OWL Full:**
 - Very high expressiveness
 - Losing tractability
 - Non-standard formalisation
 - All syntactic freedom of RDF (self-modifying)



Syntactic layering
Semantic layering

■ OWL Light

- (sub)classes, individuals
- (sub)properties, domain, range
- conjunction
- (in)equality
- cardinality 0/1
- datatypes
- inverse, transitive, symmetric
- hasValue
- someValuesFrom
- allValuesFrom

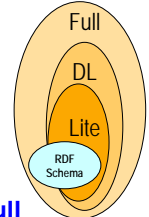
RDF Schema

■ OWL DL

- Negation
- Disjunction
- Full Cardinality
- Enumerated types

OWL Full

- Allow meta-classes etc



In RDFS you can say:

- declare classes like *Artist*, *Museum* or *Paintings*
- state that *Painter* is a subclass of *Artist*
- state that *rembrandt* is an instances of class *Painter*
- state that *hasPainted* is a property, with domain *Painter* and range *Painting*.
- state that *rembrand* is an instance of *Dutchman* with *deathdate* value 1669.

In OWL we can also

- state that *Country* and *Person* are disjoint classes;
- state that *the nl* and *england* are distinct individuals of the class *Country*
- declare *hasPainted* as inverse property of *paintedy*
- state that the class *stateless* is defined as those members of the class *Person* that have no values for the property *nationality*
- state that the class *Canadian* is defined as those members of the class *Person* that have *canada* as a value of the property *nationality*
- state that *age* is a functional property.

Where are we now ?

- **Monday:**
 - Overview, what is the SW ? , examples, languages
- **Tuesday:**
 - Description logics (DL) as a basis for KR&R on the SW
- **Wednesday:**
 - Approximate reasoning in DLs: classification, instance retrieval, reasoning with inconsistencies
- **Thursday:**
 - Analysing, diagnosing and repairing inconsistent ontologies
- **Friday:**
 - Extending OWL with mappings, Distributed Description Logics, Comparison of mapping approaches