Logic and Reasoning in the Semantic Web (part I – RDF/RDFS)

Fulvio Corno, Laura Farinetti

Politecnico di Torino

Dipartimento di Automatica e Informatica

e-Lite Research Group – http://elite.polito.it

Outline

- Reasoning in Semantic Web Knowledge Bases
- RDF/RDFS Semantics and Entailments
- OWL Semantics
- OWL reasoning tools

Part I

Outline

- **Reasoning in Semantic Web Knowledge Bases**
- RDF/RDFS Semantics and Entailments
- OWL Semantics
- OWL reasoning tools

Reasoning

- Reasoning is required when a program must determine some information or some action that has not been explicitly told about
- It must figure out what it needs to know from what it already knows

Example

- Facts:
  - Robins are birds
  - All birds have wings
- Questions:
  - Are Robins birds? $\rightarrow$ Yes
  - Do all birds have wings? $\rightarrow$ Yes
  - Do robins have wings? $\rightarrow$?????

Semantic Web peculiarities

- Traditional KBS used to be developed top down
- The Semantic Web is developed bottom-up
- 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 of mapping approaches

Reasoning in the Semantic Web

- 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

What kind of reasoning can we expect?

- Depends on
  - The semantic level of the representation (RDFS, OWL dialect), that implies a given mathematical formalization for the knowledge base
  - The size of the involved knowledge base
  - The presence (or absence) of instances
  - The capabilities of the reasoning tool

Definitions

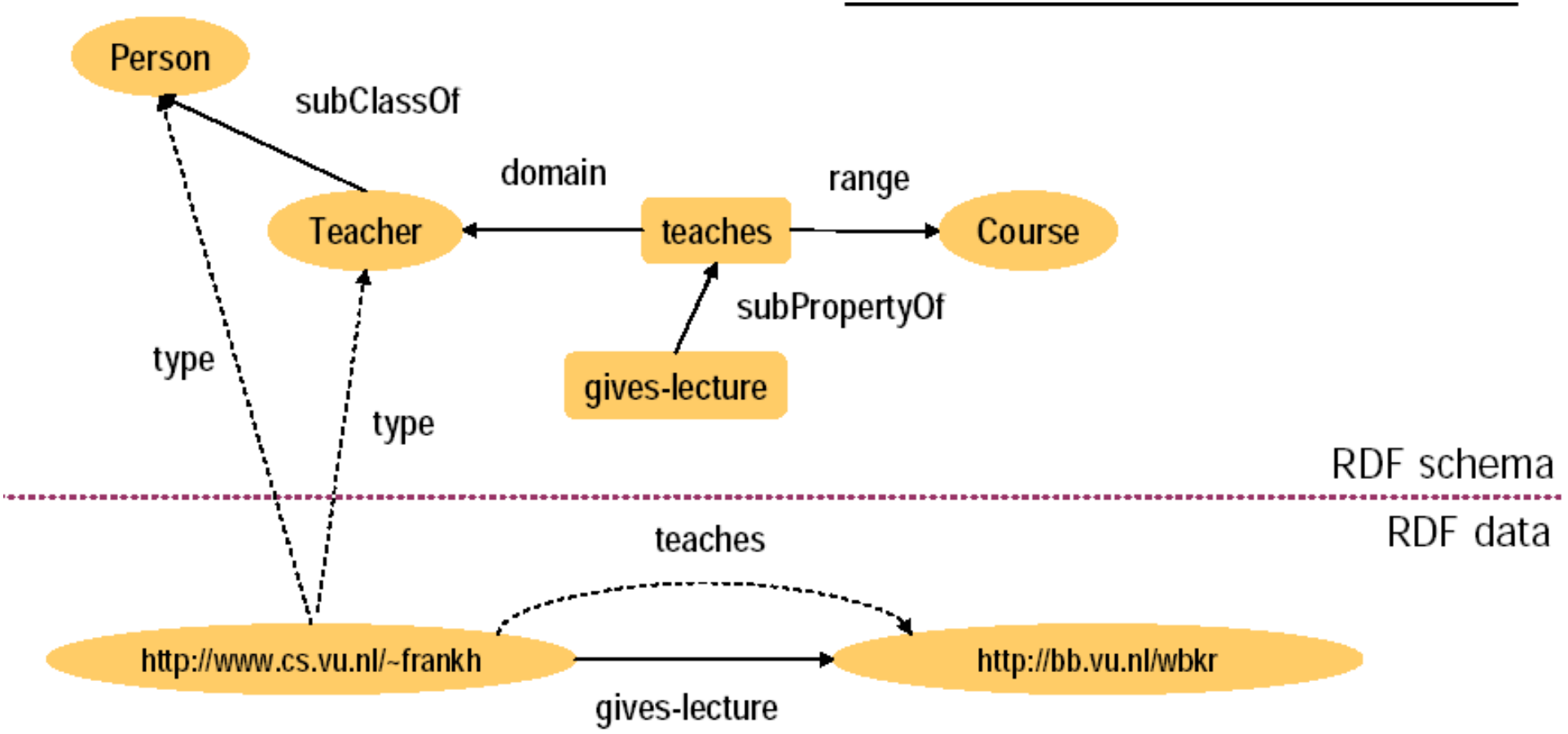
Inference (n.) An act or process of constructing new expressions from existing expressions, or the result of such an act or process. In RDF, inferences corresponding to entailments are described as *correct* or *valid*.

Inference rule, formal description of a type of inference; **inference system**, organized system of inference rules; also, software which generates inferences or checks inferences for validity.

Outline

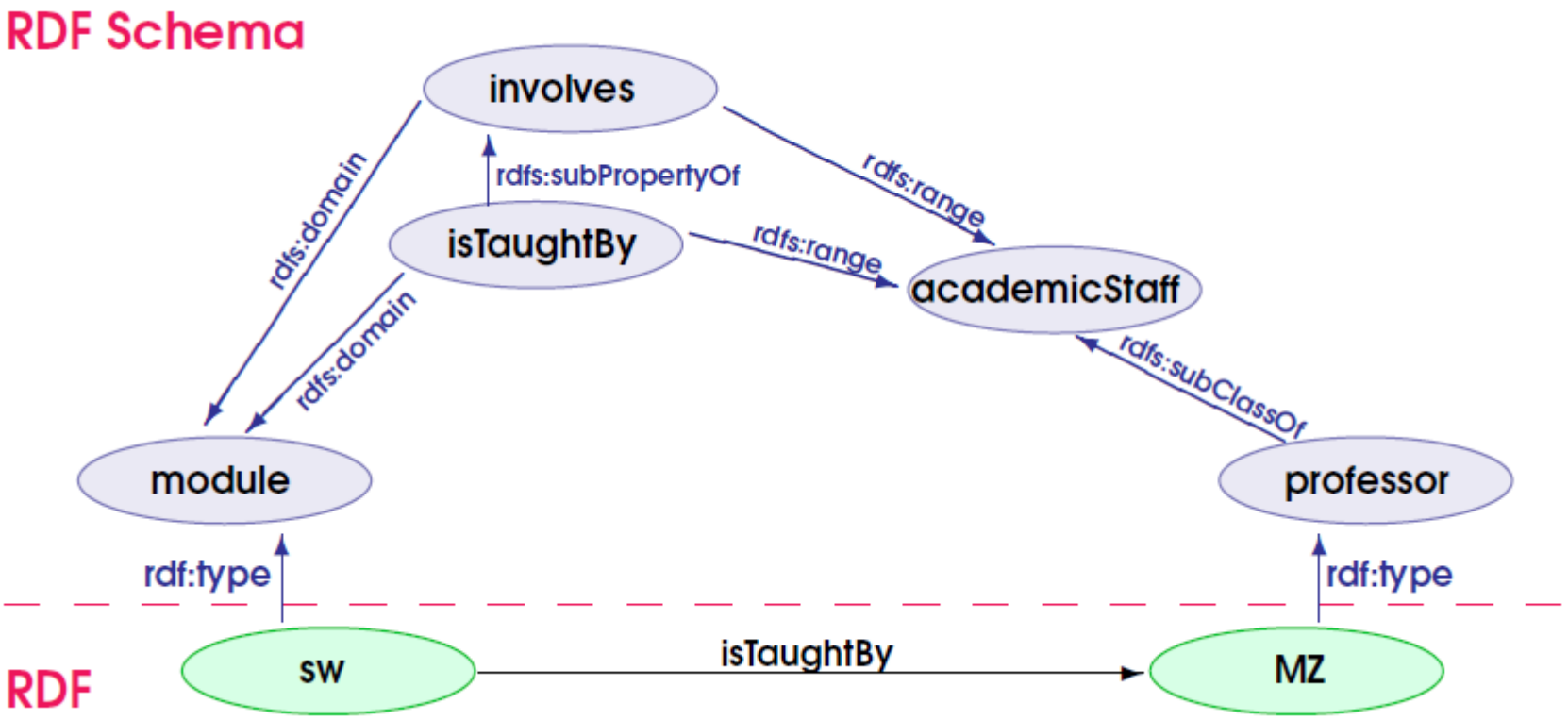
- Reasoning in Semantic Web Knowledge Bases
- **RDF/RDFS Semantics and Entailments**
- OWL Semantics
- OWL reasoning tools

Recall: RDF Schema



Source: Practical Reasoning for the Semantic Web (ESSLLI2005), S. Schlobach, H. Stuckenschmidt, http://www.few.vu.nl/~schlobac/esslli_day1.pdf

RDF vs RDFS



Source: http://www.dcs.bbk.ac.uk/~michael/sw/sw.html
Michael Zakharyaschev

Some RDFS inference rules

- (X R Y), (R subPropertyOf Q) => (X Q Y)
- (X R Y), (R domain C) => (X type C)
- (X type C), (C subClassOf D) => (X type D)
- …

RDF Semantics

- Semantics is not about the “meaning” of assertions
  - The ‘meaning’ of an assertion in RDF or RDFS may depend on many factors, including social conventions, comments in natural language or links to other content-bearing documents, … (non machine-processable information)
  - Semantics restricts itself to **a formal notion of meaning** which could be characterized as the part that is common to all other accounts of meaning, and can be captured in mechanical inference rules

Model Semantics

- The RDF Semantics W3C Recommendation uses model theory for specifying the semantics of the RDF language.
- Model theory assumes that the language refers to a '**world**', and describes the minimal conditions that a world must satisfy in order to assign an appropriate meaning for every expression in the language
- A particular world is called an **interpretation**
- The idea is to provide an abstract, mathematical account of the properties that any such interpretation must have, making as few assumptions as possible about its actual nature or intrinsic structure, thereby retaining as much generality as possible

Goals of the inference process

- To provide a **technical** way to determine when inference processes are valid, i.e., when they preserve truth
- Starting from a set of assertions that are *regarded as* true in an RDF model, derive whether a new RDF model contains *provably* true assertions
- We never known about the “real” truth of any assertion in the “real” world.

Formalization

- **Interpretations** (Normative)
  - Mapping of RDF assertions into an **abstract model**, based on set-theory
  - With an “interpretation operator” **I()**, maps a RDF graph into a highly abstract set of high-cardinality sets
  - Highly theoretical model, useful to prove mathematical properties
- **Entailments** (Informative)
  - **Transformation rules** to derive new assertions from existing ones
  - May be **proven complete and consistent** with the formal interpretation

Definitions

Interpretation (**of**) (n.) A minimal formal description of those aspects of a world which is just sufficient to establish the truth or falsity of any expression of a logic.

World (n.) (with *the:*) (i) The actual world. (with *a:*) (ii) A way that the actual world might be arranged. (iii) An interpretation (iv) A possible world.

Definition

Entail (v.), **entailment** (n.). A semantic relationship between expressions which holds whenever the truth of the first guarantees the truth of the second. Equivalently, whenever it is logically impossible for the first expression to be true and the second one false. Equivalently, when any interpretation which satisfies the first also satisfies the second. (Also used between a set of expressions and an expression.)

Interpretation: minimum notions

- Let V be a vocabulary containing all names (URIs and literals) occurring in RDF triples
- An RDF **interpretation** of V consists of:
  - IR, a non-empty set of **resources** (domain or universe)
  - $IP \subseteq IR$, a set of **properties** (each property is also a resource)
    (symbol $v \in V$ is a property if, and only if, $IS(v) \in IP$)
  - $IS : V \rightarrow IR$, an interpretation of resources
    (with each symbol from V a resource is associated)
  - $IEXT : IP \rightarrow 2^{IR \times IR}$, an interpretation of properties
    (each property is a binary relation, i.e., a subset of $IR \times IR$)

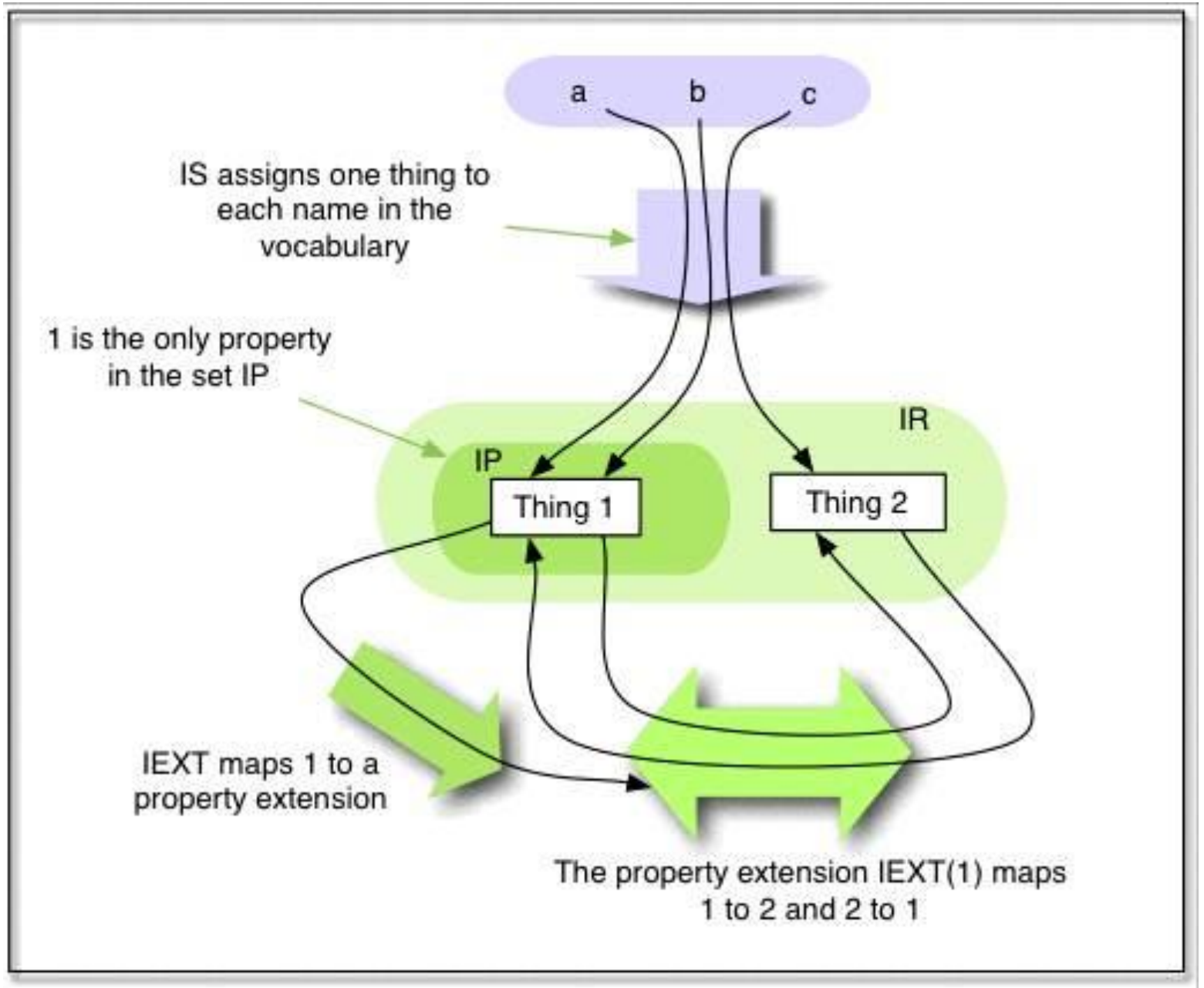
An RDF triple $\langle s, p, o \rangle$ is **true** in I if, and only if,
$s, p, o \in V$, $IS(p) \in IP$ and $(IS(s), IS(o)) \in IEXT(IS(p))$

An RDF triple is false in I if it is not true in I

An RDF graph is **true** in I if, and only if, every triple of it is true in I

In particular, it is false in I if some triple is not true in I

Conceptual framework

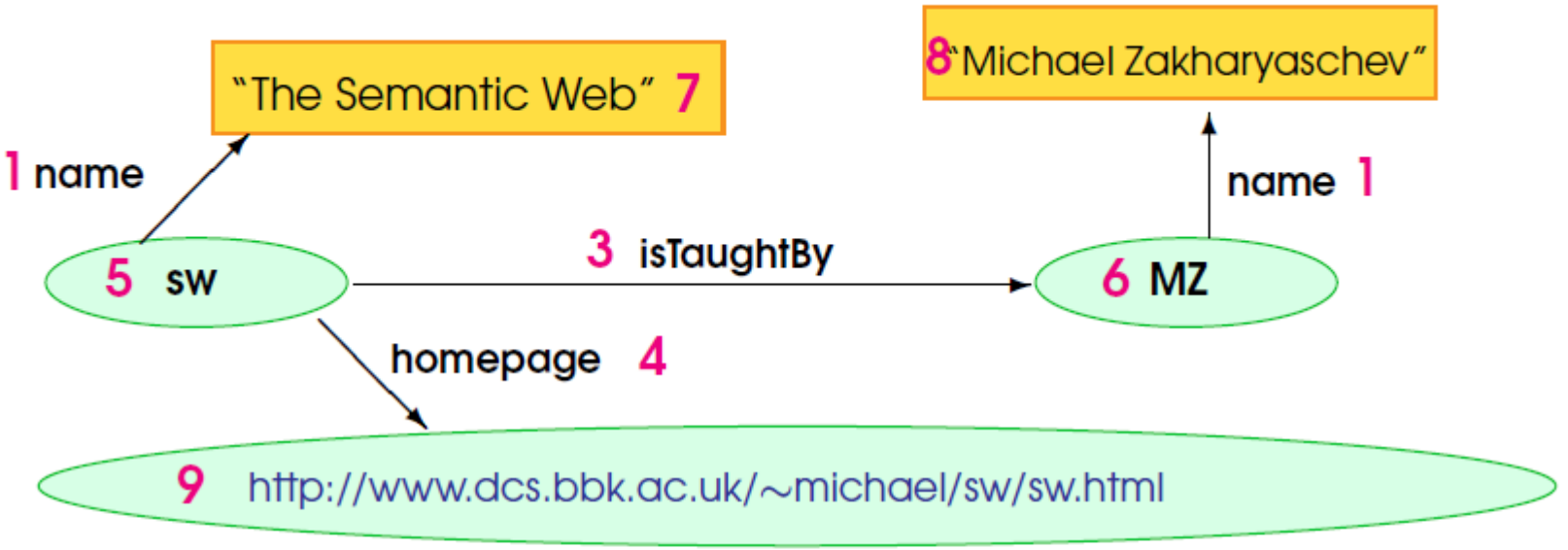


Interpretation: minimum notions

- I : interpretation operator
- E : fragment of RDF syntax
- If E is a graph:
  - I(E) = false if I(E') = false for some triple E' in E, otherwise I(E) =true
- If E is a triple <s, p, o>:
  - I(E) = true if s, p and o are in V, I(p) is in IP and <I(s),I(o)> is in IEXT(I(p))
  - otherwise I(E)= false
- IP : set of properties, IR : set of resources
- IEXT(I(p)) : mapping from IP into the powerset of IR x IR i.e. the set of sets of pairs <x,y> with x and y in IR

Example

Construct an interpretation in which the following RDF graph is true:

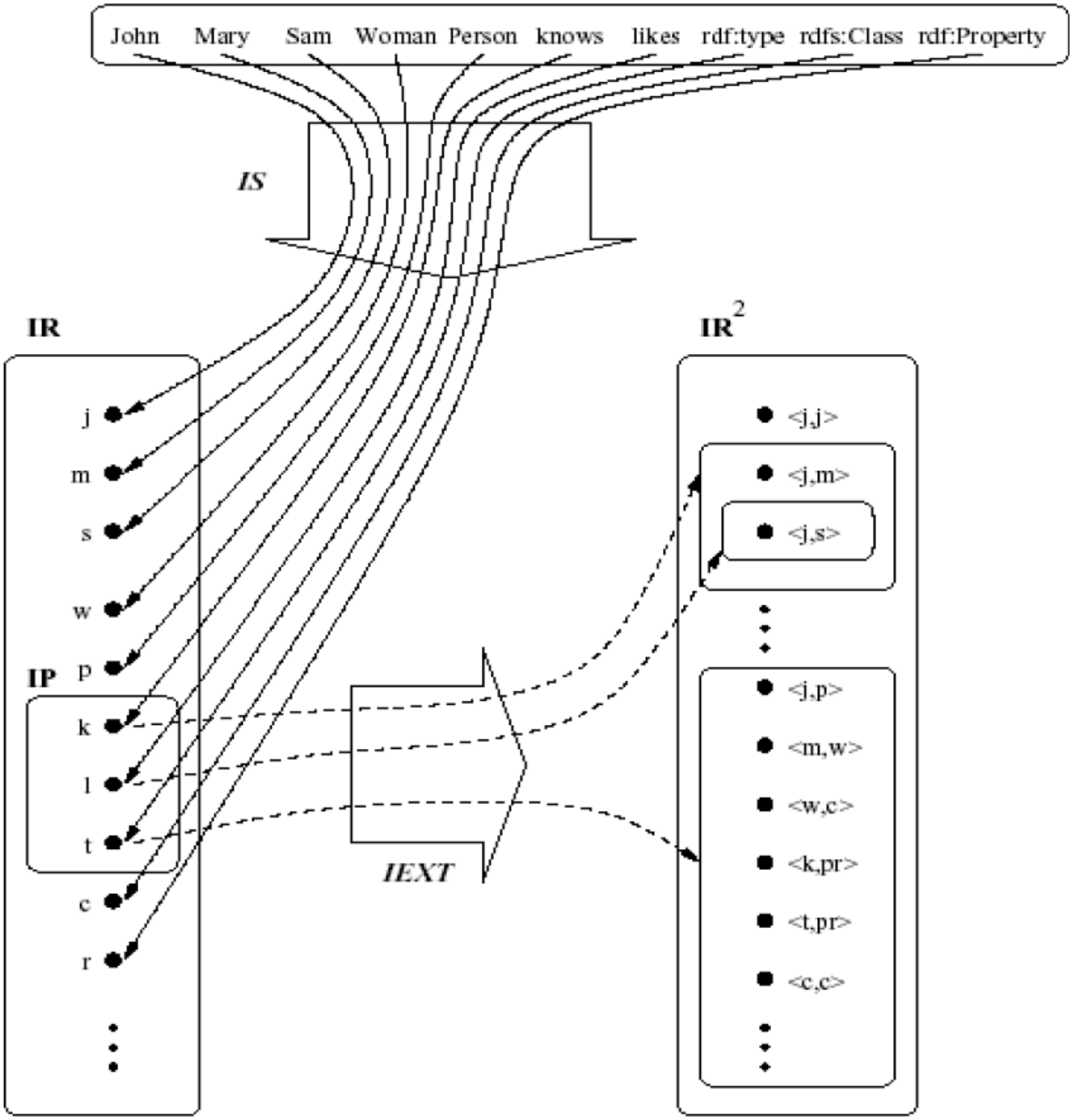


$IR = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ and $IP = \{1, 2, 3, 4\}$

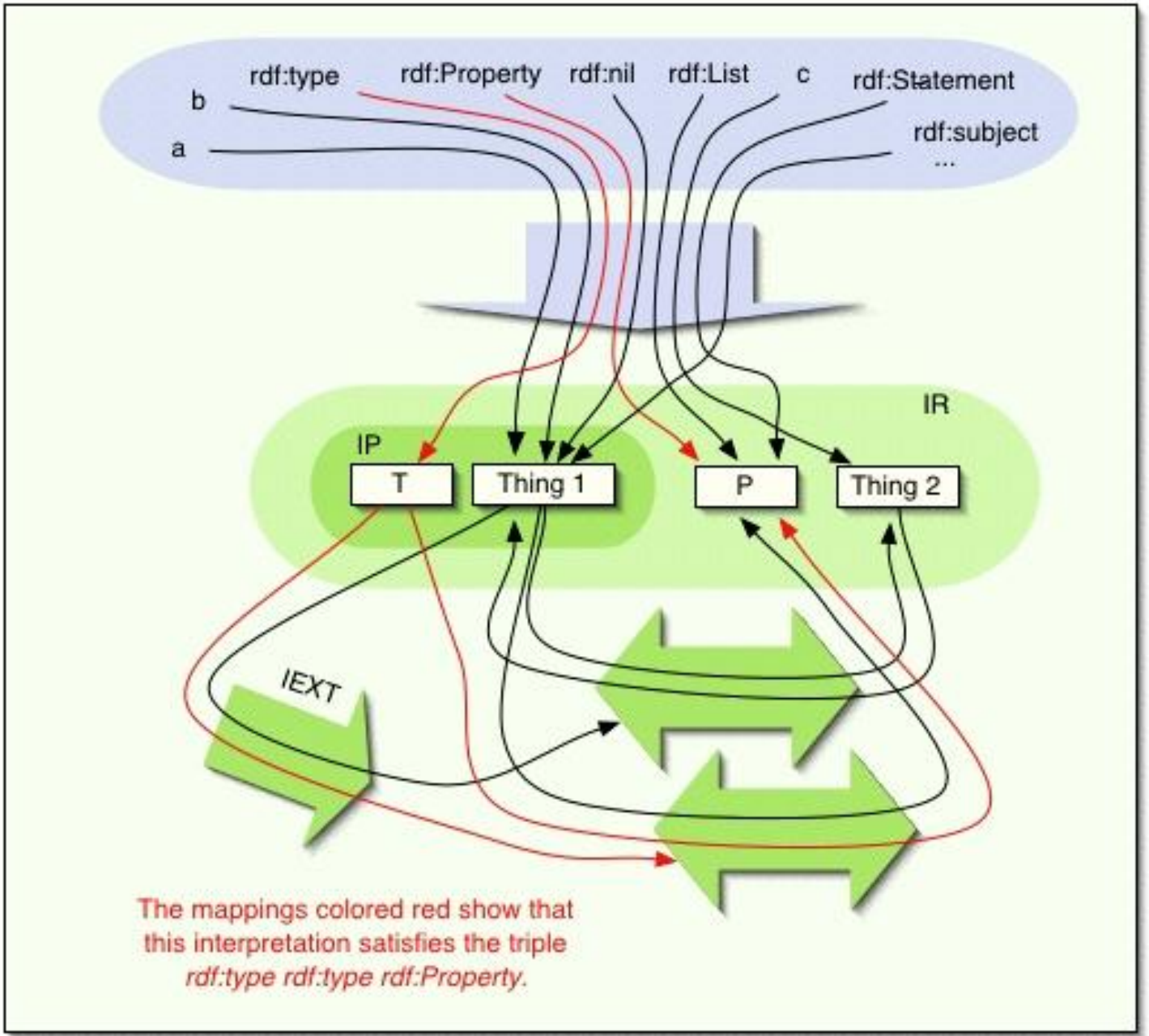
$IS(\textbf{name}) = 1$, $IS(\textbf{isTaughtBy}) = 3$, $IS(\textbf{homepage}) = 4$, $IS(\textbf{sw}) = 5$,
$IS(\textbf{MZ}) = 6$, $IS(\text{http://www.dcs.bbk.ac.uk/}\sim\text{michael/sw/sw.html}) = 9$,
$IS(\text{"The Semantic Web"}) = 7$, $IS(\text{"Michael Zakharyaschev"}) = 8$

$IEXT(1) = \{(5, 7), (6, 8)\}$, $IEXT(2) = \emptyset$, $IEXT(3) = \{(5, 6)\}$,
$IEXT(4) = \{(5, 9)\}$

Example


John
Mary
Sam
Woman
Person
knows
likes
rdf:type
rdfs:Class
rdf:Property
IS
IR
IR2
j
m
s
w
p
IP
k
l
t
c
r
IEXT
<j,j>
<j,m>
<j,s>
<j,p>
<m,w>
<w,c>
<k,pr>
<t,pr>
<c,c>

Example of interpretation



Entailment

- I *satisfies* E if I(E)=true
- A set S of RDF graphs **entails** a graph E if every interpretation which satisfies every member of S also satisfies E
- In human words:
  - assertion = a claim that the world is an interpretation which assigns the value true to the assertion
  - If A entails B, then
    - any interpretation that makes A true also makes B true
    - an assertion of A already contains the same "meaning" as an assertion of B
    - the meaning of B is somehow contained in, or subsumed by, that of A

Entailment regime

- RDF defines 4 entailment regimes:
  - Simple entailment : does **not** interpret any RDF or RDFS vocabulary (just structural matching, by re-naming blank nodes)
  - RDF entailment : interprets RDF vocabulary
  - RDFS entailment : interprets RDF and RDFS vocabularies
  - D entailment : additional support for datatypes

RDF Entailment rules (legend)

- `aaa`, `bbb`: any URI reference
- `uuu`, `vvv`: any URI reference or blank node identifier
- `xxx`, `yyy`: any URI reference, blank node identifier or literal
- `lll`: any literal
- `_:nnn`: blank node identifiers

Simple Entailment rules

Rule name	If E contains	then add
se1	`uuu aaa xxx .`	`uuu aaa _:nnn .` where _:nnn identifies a blank node allocated to xxx by rule se1 or se2.
se2	`uuu aaa xxx .`	`_:nnn aaa xxx .` where _:nnn identifies a blank node allocated to uuu by rule se1 or se2.
lg	`uuu aaa lll .`	`uuu aaa _:nnn .` where _:nnn identifies a blank node allocated to the literal lll by this rule.
gl	`uuu aaa _:nnn .` where _:nnn identifies a blank node allocated to the literal lll by rule lg.	`uuu aaa lll .`

Role of Entailment rules (1/3)

- Simple entailment satisfies the “Interpolation Lemma”:
  - S entails a graph E if and only if a subgraph of S is an instance of E
  - It completely characterizes simple RDF entailment in syntactic terms
- RDF entailment satisfies the “RDF entailment lemma”:
  - S rdf-entails E if and only if there is a graph which can be derived from S plus the RDF axiomatic triples by the application of rule lg and the RDF entailment rules and which simply entails E
  - This means that the entailment rules are “complete”

RDF Entailment rules

Rule name	If E contains	then add
rdf1	`uuu aaa yyy .`	`aaa rdf:type rdf:Property .`
rdf2	`uuu aaa lll .` where lll is a well-typed XML literal .	`_:nnn rdf:type rdf:XMLLiteral .` where _:nnn identifies a blank node allocated to lll by rule lg.

RDF Axiomatic triples

```
rdf:type          rdf:type          rdf:Property .
rdf:subject       rdf:type          rdf:Property .
rdf:predicate     rdf:type          rdf:Property .
rdf:object        rdf:type          rdf:Property .
rdf:first         rdf:type          rdf:Property .
rdf:rest          rdf:type          rdf:Property .
rdf:value         rdf:type          rdf:Property .
rdf:_1            rdf:type          rdf:Property .
rdf:_2            rdf:type          rdf:Property .
...
rdf:nil           rdf:type          rdf:List .
```

Role of Entailment rules (2/3)

- RDF entailment satisfies the “RDF entailment lemma”:
  - S rdf-entails E if and only if there is a graph which can be derived from S plus the RDF axiomatic triples by the application of rule lg and the RDF entailment rules and which simply entails E
  - This means that the entailment rules are “complete”

RDFS Entailment rules (1/2)

Rule name	If E contains	then add
rdfs1	`uuu aaa lll.` where lll is a plain literal (with or without a language tag)	`_:nnn rdf:type rdfs:Literal .` where _:nnn identifies a blank node allocated to lll by rule rule lg.
rdfs2	`aaa rdfs:domain xxx .` `uuu aaa yyy .`	`uuu rdf:type xxx .`
rdfs3	`aaa rdfs:range xxx .` `uuu aaa vvv .`	`vvv rdf:type xxx .`
rdfs4a	`uuu aaa xxx .`	`uuu rdf:type rdfs:Resource .`
rdfs4b	`uuu aaa vvv.`	`vvv rdf:type rdfs:Resource .`
rdfs5	`uuu rdfs:subPropertyOf vvv .` `vvv rdfs:subPropertyOf xxx .`	`uuu rdfs:subPropertyOf xxx .`

RDFS Entailment rules (2/2)

Rule name	If E contains	then add
rdfs6	`uuu rdf:type rdf:Property .`	`uuu rdfs:subPropertyOf uuu .`
rdfs7	`aaa rdfs:subPropertyOf bbb` `uuu aaa yyy .`	`uuu bbb yyy .`
rdfs8	`uuu rdf:type rdfs:Class .`	`uuu rdfs:subClassOf rdfs:Resource .`
rdfs9	`uuu rdfs:subClassOf xxx .` `vvv rdf:type uuu .`	`vvv rdf:type xxx .`
rdfs10	`uuu rdf:type rdfs:Class .`	`uuu rdfs:subClassOf uuu .`
rdfs11	`uuu rdfs:subClassOf vvv .` `vvv rdfs:subClassOf xxx .`	`uuu rdfs:subClassOf xxx .`
rdfs12	`uuu rdf:type rdfs:ContainerMembershipProperty .`	`uuu rdfs:subPropertyOf rdfs:member .`
rdfs13	`uuu rdf:type rdfs:Datatype`	`uuu rdfs:subClassOf rdfs:Literal .`

RDFS Axiomatic triples (1/3)

```
rdf:type                rdfs:domain     rdfs:Resource .
rdfs:domain             rdfs:domain     rdf:Property .
rdfs:range              rdfs:domain     rdf:Property .
rdfs:subPropertyOf      rdfs:domain     rdf:Property .
rdfs:subClassOf         rdfs:domain     rdfs:Class .
rdf:subject             rdfs:domain     rdf:Statement .
rdf:predicate           rdfs:domain     rdf:Statement .
rdf:object              rdfs:domain     rdf:Statement .
rdfs:member             rdfs:domain     rdfs:Resource .
rdf:first               rdfs:domain     rdf:List .
rdf:rest                rdfs:domain     rdf:List .
rdfs:seeAlso            rdfs:domain     rdfs:Resource .
rdfs:isDefinedBy        rdfs:domain     rdfs:Resource .
rdfs:comment            rdfs:domain     rdfs:Resource .
rdfs:label              rdfs:domain     rdfs:Resource .
rdf:value               rdfs:domain     rdfs:Resource .
```

RDFS Axiomatic triples (2/3)

```
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:subject             rdfs:range      rdfs:Resource .
rdf:predicate           rdfs:range      rdfs:Resource .
rdf:object              rdfs:range      rdfs:Resource .
rdfs:member             rdfs:range      rdfs:Resource .
rdf:first               rdfs:range      rdfs:Resource .
rdf:rest                rdfs:range      rdf:List .
rdfs:seeAlso            rdfs:range      rdfs:Resource .
rdfs:isDefinedBy        rdfs:range      rdfs:Resource .
rdfs:comment            rdfs:range      rdfs:Literal .
rdfs:label              rdfs:range      rdfs:Literal .
rdf:value               rdfs:range      rdfs:Resource .
```

RDFS Axiomatic triples (3/3)

```
rdf:Alt                    rdfs:subClassOf           rdfs:Container .
rdf:Bag                    rdfs:subClassOf           rdfs:Container .
rdf:Seq                    rdfs:subClassOf           rdfs:Container .
rdfs:ContainerMembershipProperty rdfs:subClassOf rdf:Property .

rdfs:isDefinedBy           rdfs:subPropertyOf        rdfs:seeAlso .

rdf:XMLLiteral             rdf:type                  rdfs:Datatype .
rdf:XMLLiteral             rdfs:subClassOf           rdfs:Literal .
rdfs:Datatype              rdfs:subClassOf           rdfs:Class .

rdf:_1                     rdf:type  rdfs:ContainerMembershipProperty .
rdf:_1                     rdfs:domain               rdfs:Resource .
rdf:_1                     rdfs:range                rdfs:Resource .
rdf:_2                     rdf:type  rdfs:ContainerMembershipProperty .
rdf:_2                     rdfs:domain               rdfs:Resource .
rdf:_2                     rdfs:range                rdfs:Resource .
...
```

Role of Entailment rules (3/3)

- RDFS entailment satisfies the “RDFS entailment lemma”:
  - S rdfs-entails E if and only if there is a graph which can be derived from S plus the RDF and RDFS axiomatic triples by the application of rule lg, rule gl and the RDF and RDFS entailment rules and which either simply entails E or contains an XML clash

Effect of Entailment rules (1/3)

- The outputs of these rules will often trigger others.

rule:	triggers rule: 1	2	3	4a	4b	5a	5b	6	7a	7b	8	9	10	11
1		*	*	*			*	*				*		
2		*	*				*	*	*	*		*	*	*
3		*	*				*	*	*	*		*	*	*
4a		*	*					*				*		
4b		*	*					*				*		
5a						*		*						
5b		*	*					*						
6		*	*			*	*	*	*	*	*	*	*	*
7a		*	*					*			*	*		
7b		*	*					*						
8								*			*	*		
9			*				*	*	*	*		*	*	*
10		*	*		*	*		*						
11		*	*					*			*	*		

Table 1: Dependencies between RDFS entailment rules

Source: Inferencing and Truth Maintenance in RDF Schema, Exploring a naive practical approach, J. Broekstra, A. Kampman, http://www.openrdf.org/doc/papers/inferencing.pdf

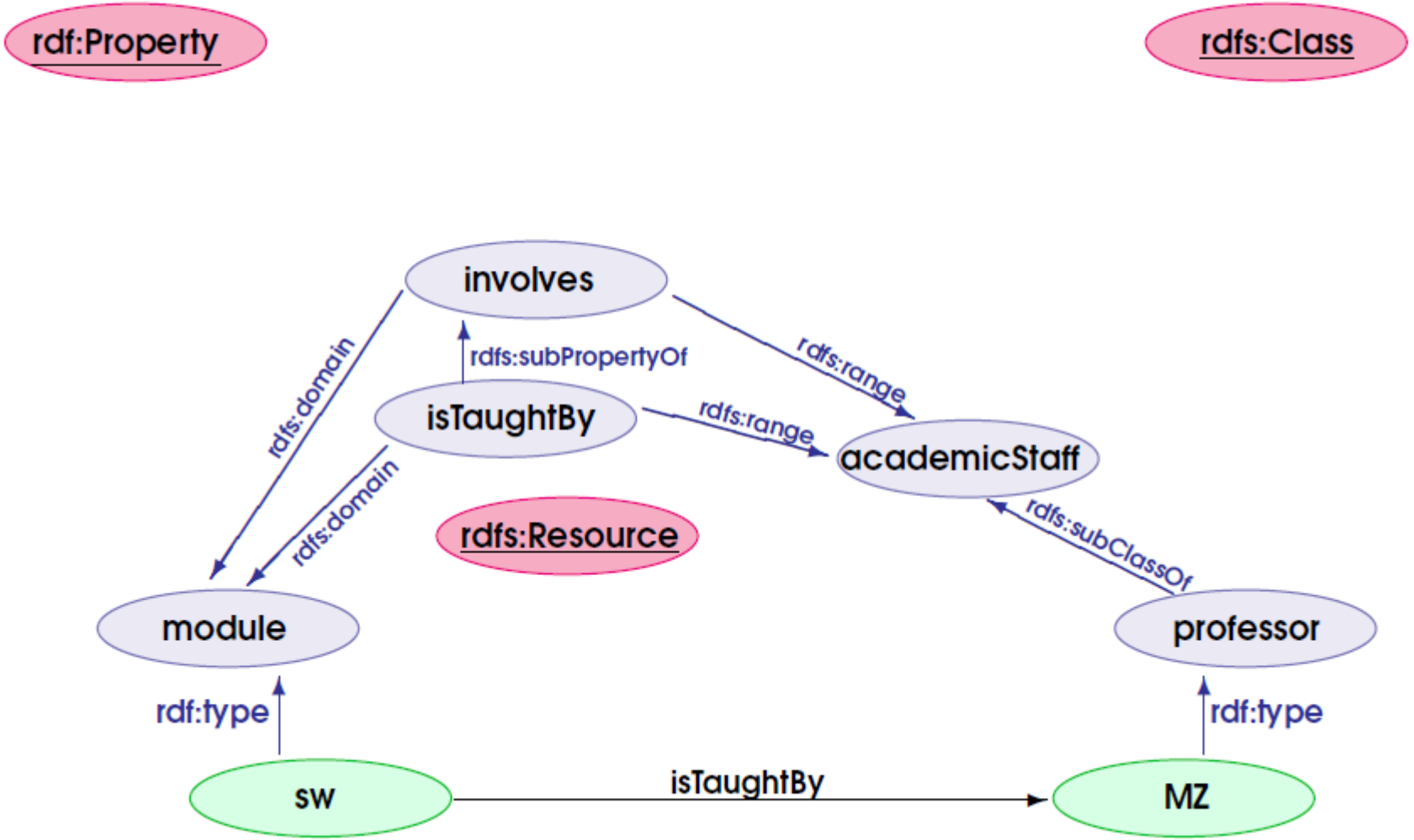
Effect of Entailment rules (2/3)

- Rules propagate all rdf:type assertions in the graph up the subproperty and subclass hierarchies
- rdf1 generates type assertions for all the property names used in the graph
  - `uuu aaa yyy → aaa rdf:type rdf:Property`
- rdfs3 together with the last RDFS axiomatic triple adds all type assertions for all the class names used
  - `→ rdf:_2 rdfs:range rdfs:Resource`
  - `aaa rdfs:range xxx ∧ uuu aaa vvv → vvv rdf:type xxx`

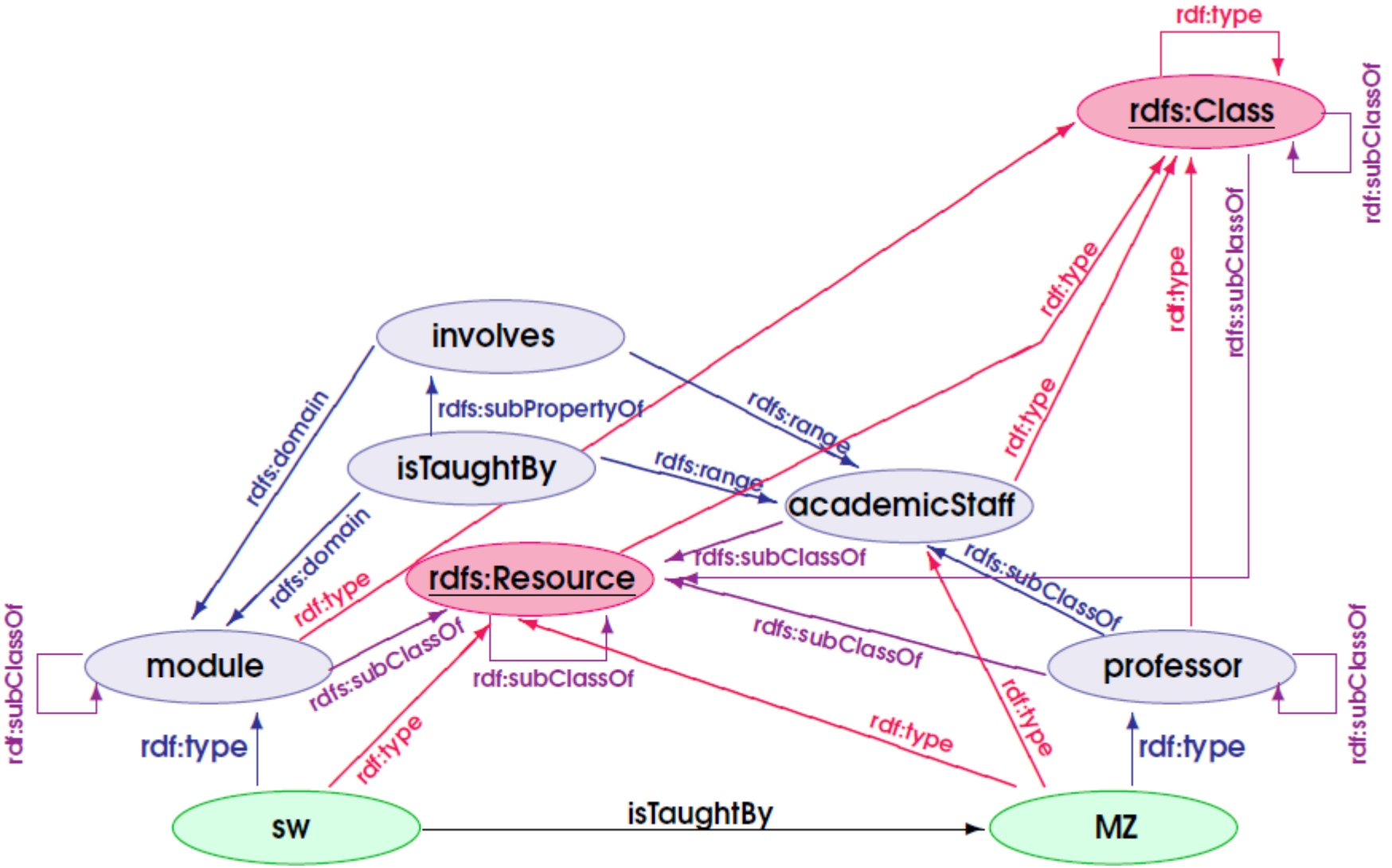
Effect of Entailment rules (3/3)

- Any subproperty or subclass assertion generates type assertions for its subject and object via rdfs2 and rdfs3 and the domain and range assertions in the RDFS axiomatic triple set
  - `aaa rdfs:domain xxx ∧ uuu aaa yyy → uuu rdf:type xxx`
  - `aaa rdfs:range xxx ∧ uuu aaa vvv → vvv rdf:type xxx`
- For every uuu in V, the rules generate all assertions
  - `uuu rdf:type rdfs:Resource`
- For every class name uuu, the rules generate
  - `uuu rdfs:subClassOf rdfs:Resource`

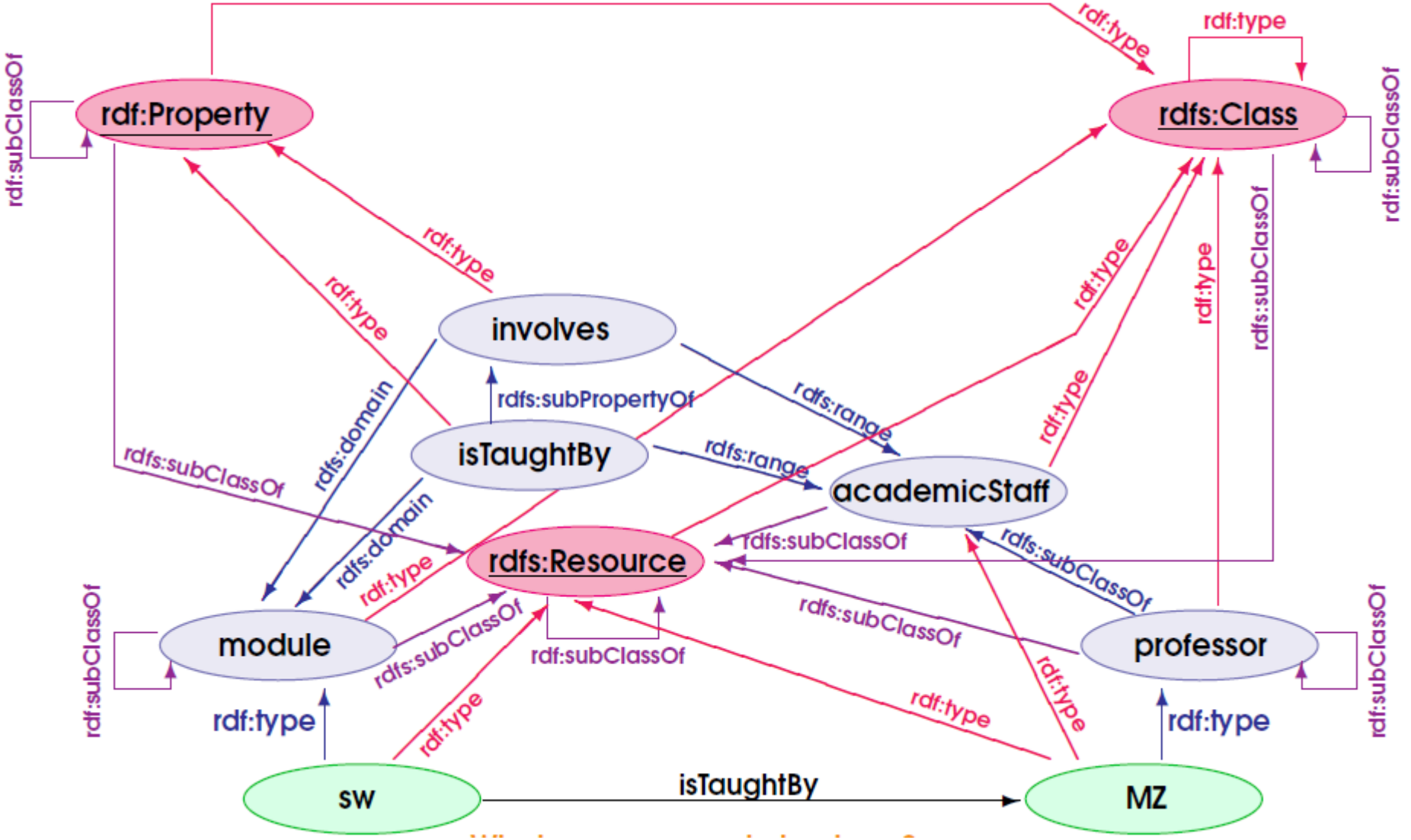
Example (axiomatic nodes)



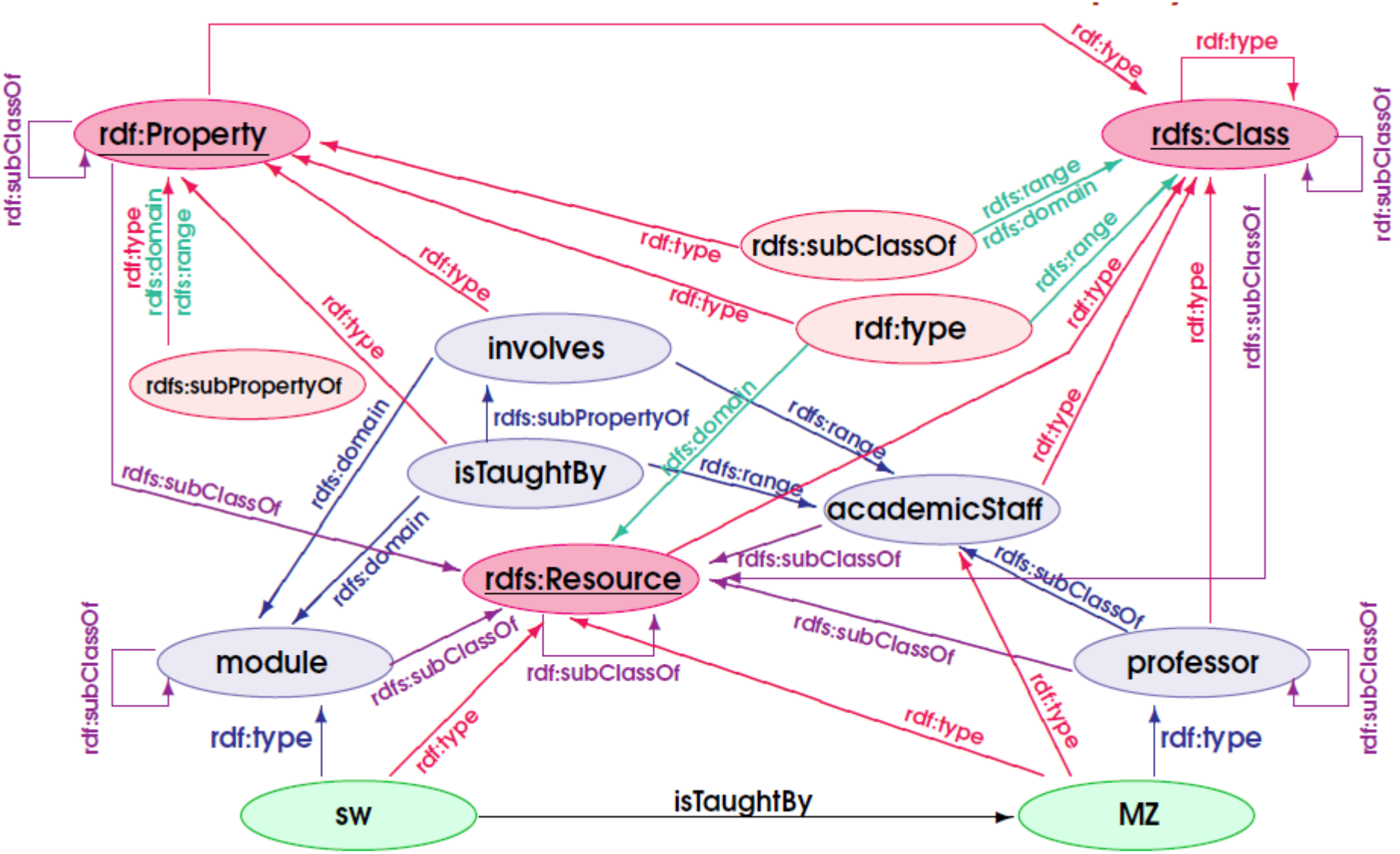
Example (rdf:type and rdfs:subclassOf)



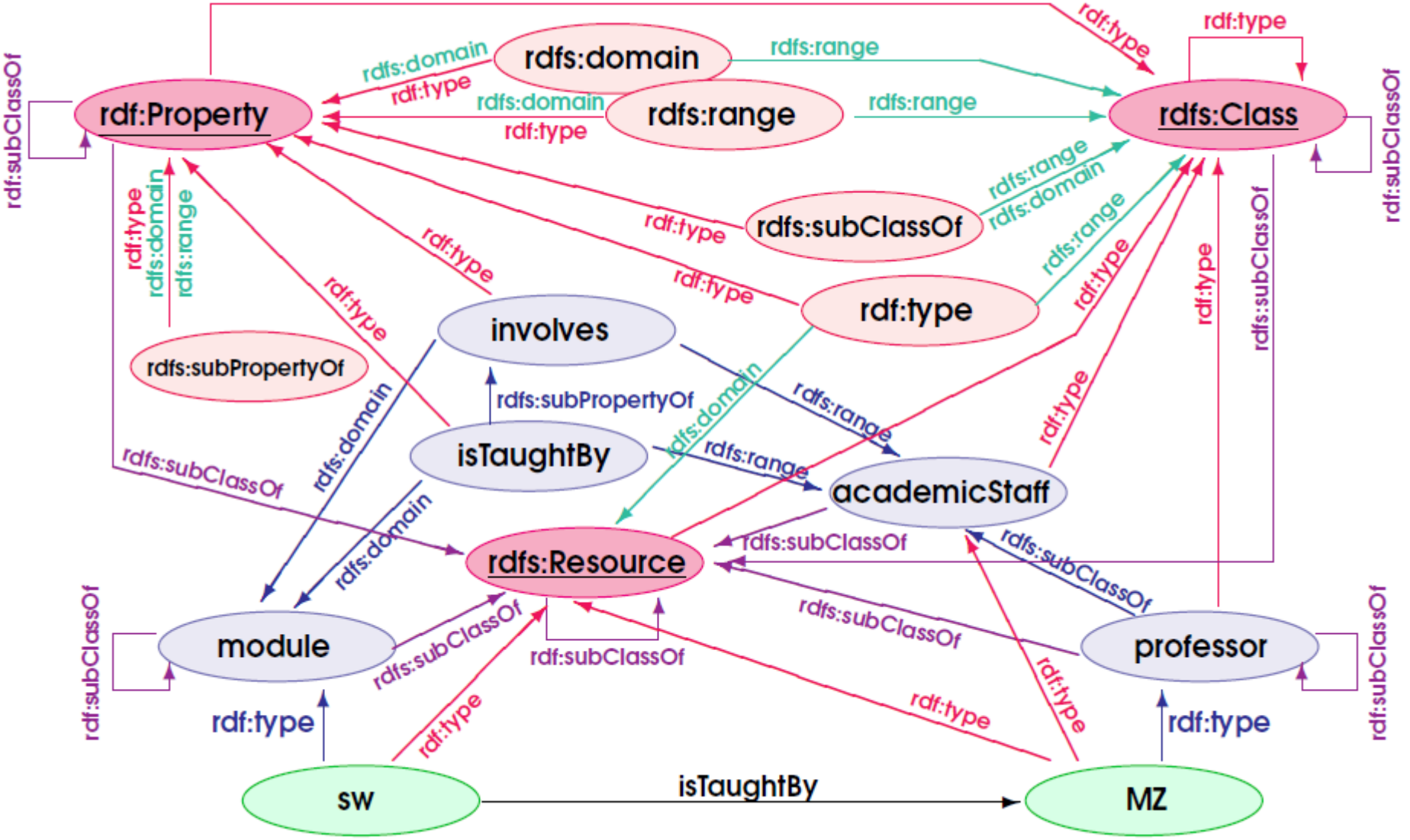
Example (properties, domains, ranges)



Example (reification of properties)



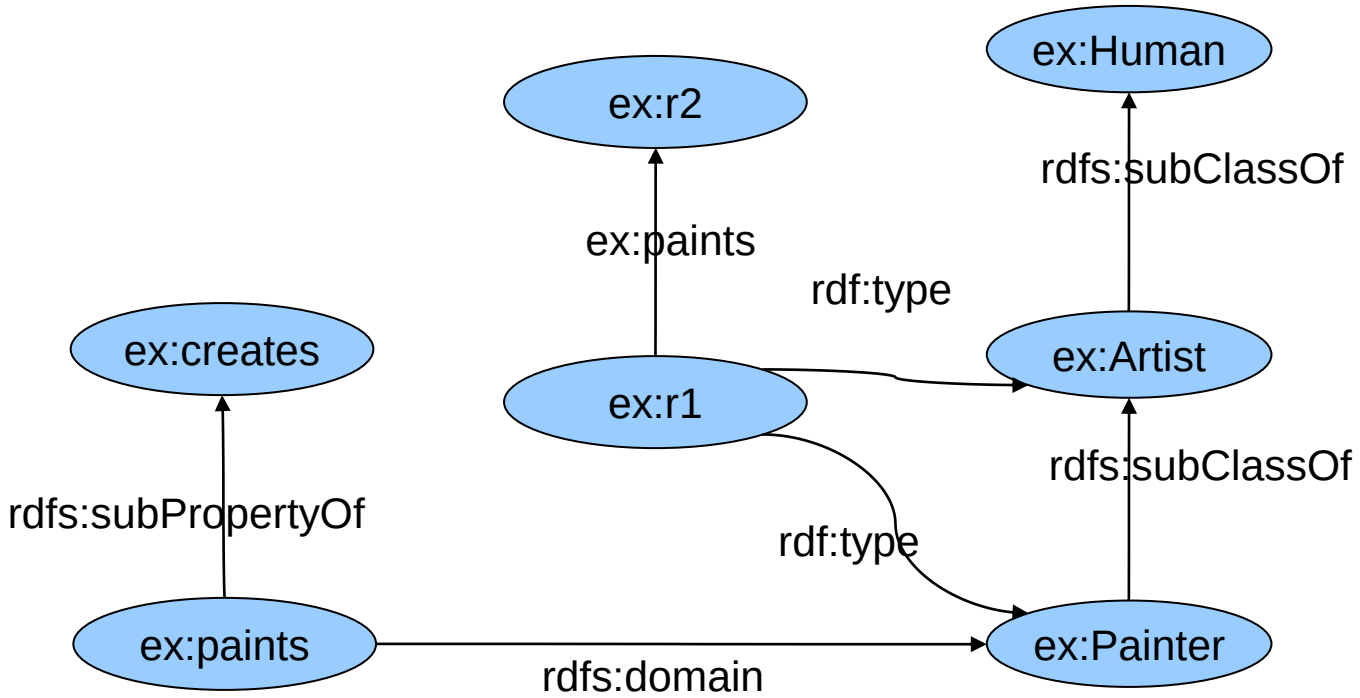
Example (reification of properties+axiomatic properties)



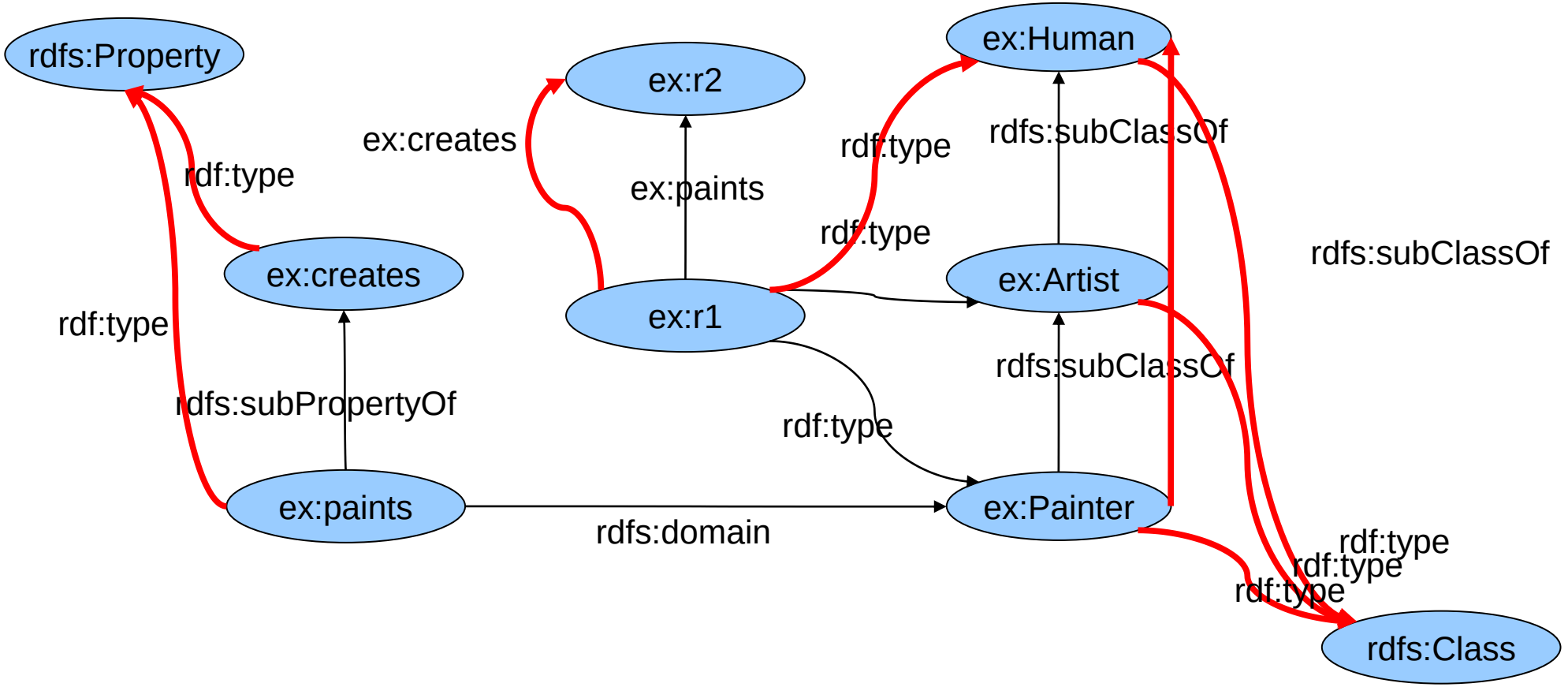
Closure and Reduction

- 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.

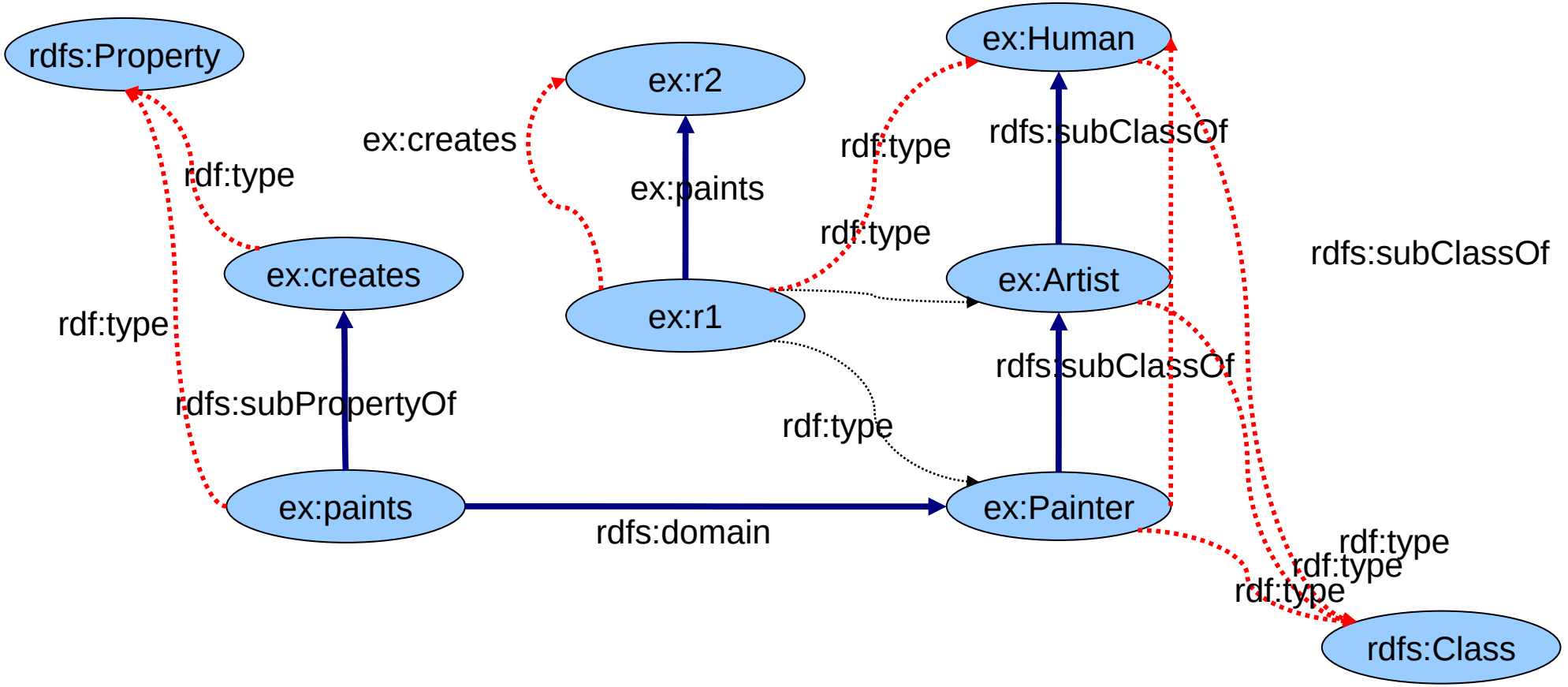
Example


ex:Human
ex:r2
rdfs:subClassOf
ex:paints
rdf:type
ex:creates
ex:Artist
ex:r1
rdfs:subClassOf
rdfs:subPropertyOf
rdf:type
ex:paints
ex:Painter
rdfs:domain

Example: closure



Example: reduction

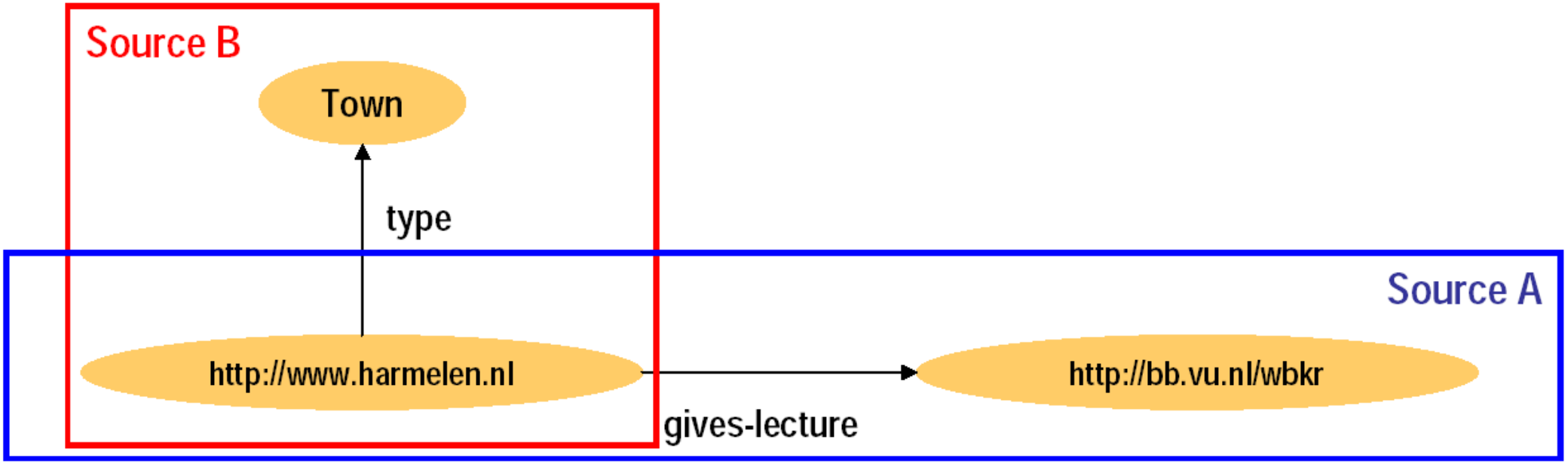


Discussion on RDF

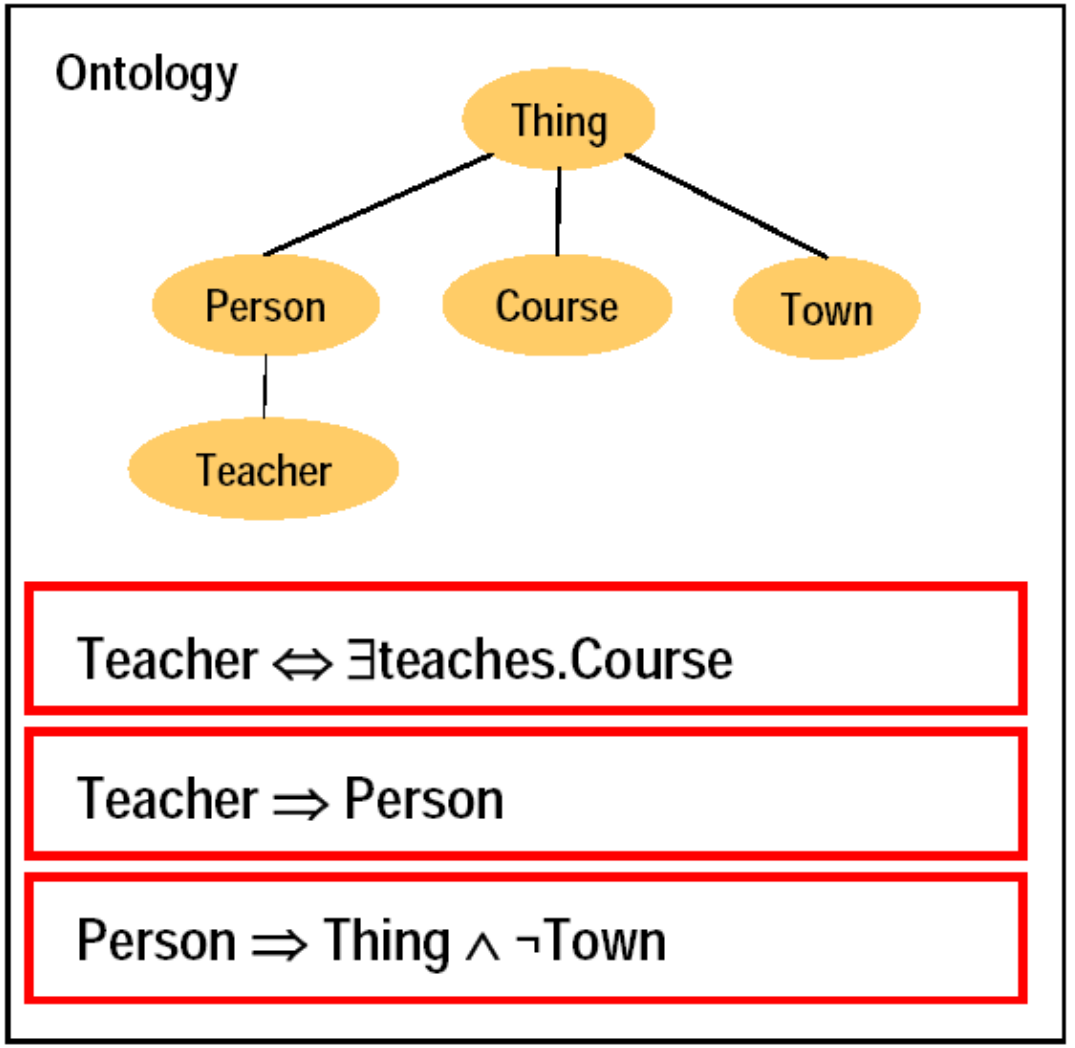
- RDF is a simple, but useful language to make relations between resources explicit
- It allows to define a simple form of schema which mostly consists of class and property hierarchies as well as relations with domain and range restriction
- There is a simple rule based reasoning mechanism for RDF schema 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

Problems with RDF

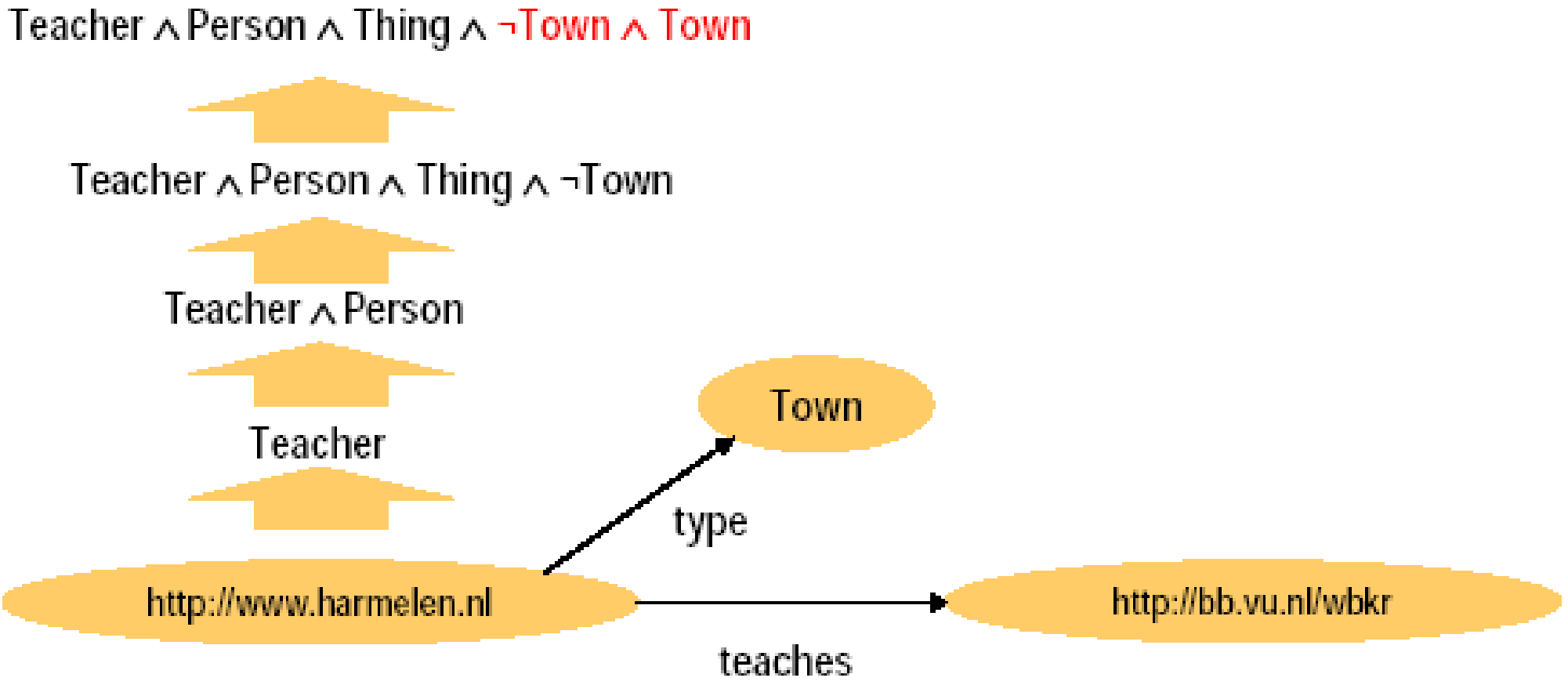
- Not always meaningful representation



Need logical axioms!

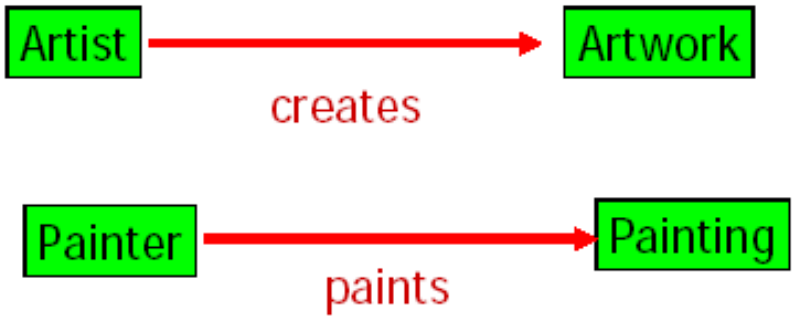


We can detect inconsistencies



What properties do we need?

- Subclass relations
- Sub-properties



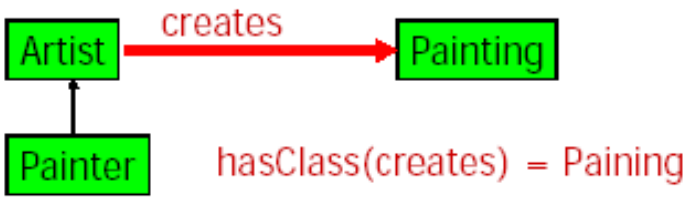
- Set operations
  - Intersection (logical conjunction)
    - Painter ∩ Writer = {x| x⊂Painter AND x⊂Writer}
  - Union (logical disjunction)
    - Painter ∪ Writer = {x| x⊂Painter OR x⊂Writer}
  - Complement
    - complementOf(Ill,Healthy)
  - Disjoint
    - disjointWith(Lung,Liver,Kidney)

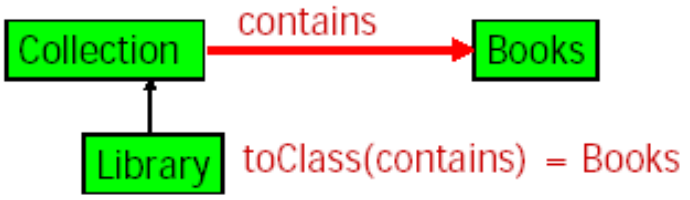
What properties do we need? /2

- Cardinality restriction
- Existential quantifier
- Universal quantifier

- Cardinality(has_location) = 1 :

- Min_Cardinality(creates) = 1 :





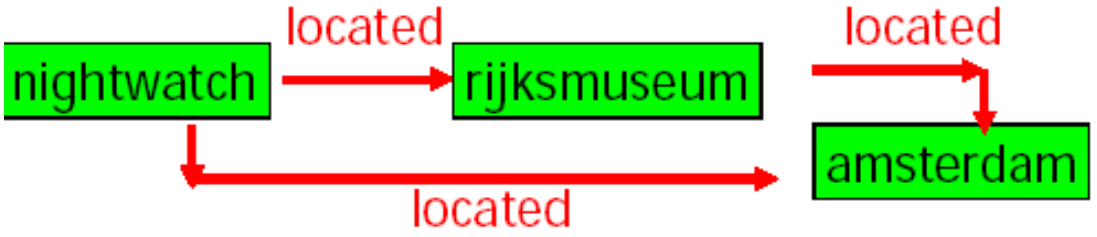
What properties do we need? /3

- Inverse properties

- Symmetric properties

- Transitivity

borders(Liver,Gall) -> borders(Gall,Liver)



References

- RDF Semantics - W3C Recommendation 10 February 2004
  - http://www.w3.org/TR/rdf-mt/
- Course material for “Practical Reasoning for the Semantic Web” course at the 17th European Summer School in Logic, Language and Information (ESSLLI)
  - http://www.few.vu.nl/~schlobac/

References

- http://www.aaai.org/AITopics/html/reason.html
- http://media.cwi.nl/survey/
- http://www.mkbergman.com
- http://www.dcs.bbk.ac.uk/~michael/sw/sw.html
- http://www.inf.unibz.it/~debruijn/teaching/swt/

License