RDF Semantics
A graph-based approach

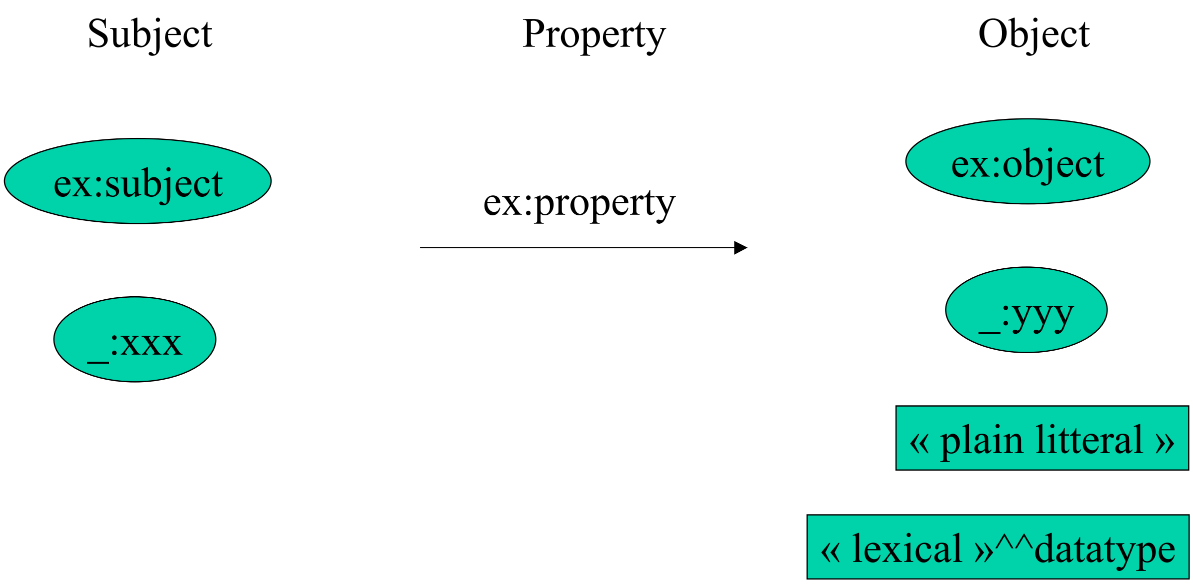
Jean-François Baget

INRIA Rhône-Alpes

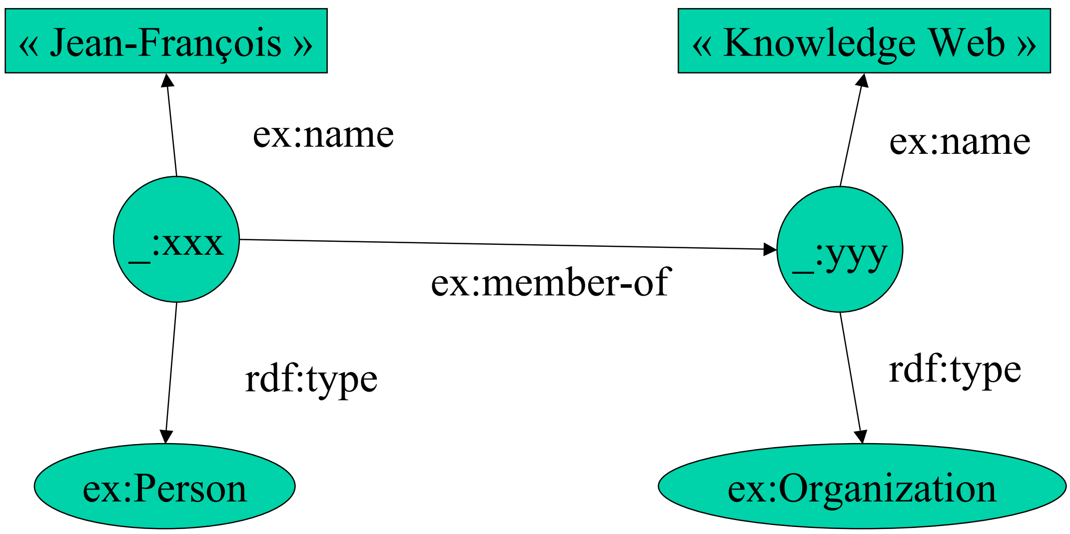
Jean-Francois.Baget@inrialpes.fr

Manchester Knowledge Web Meeting
Sept. 27 –29, 2004

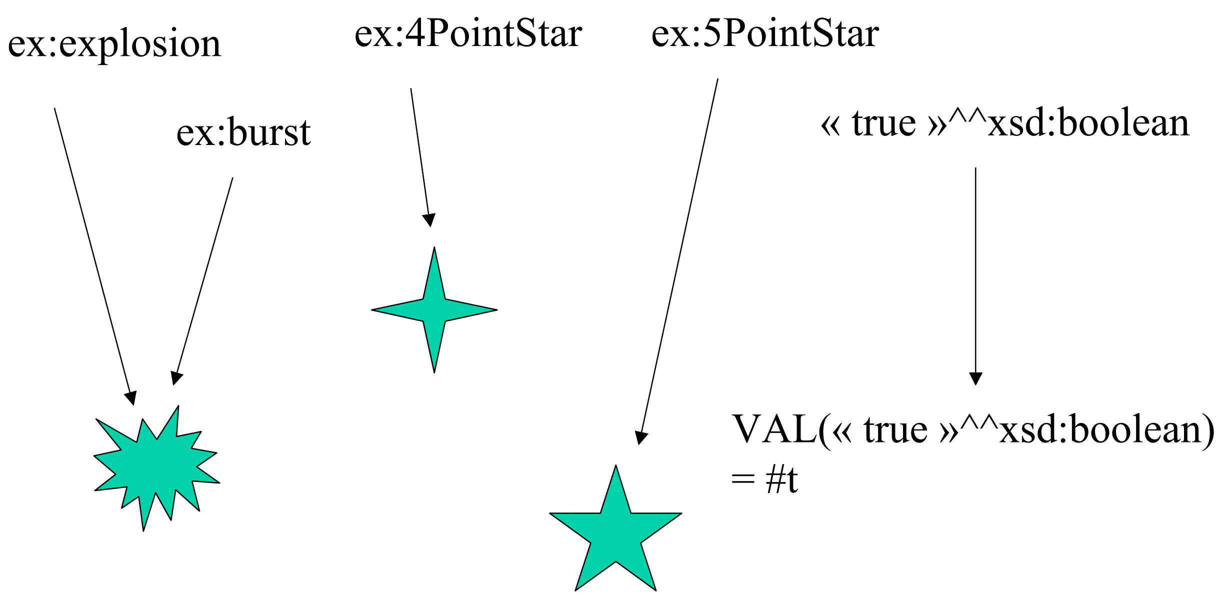
RDF Syntax (1)
Triples



RDF Syntax (2)
Graphs

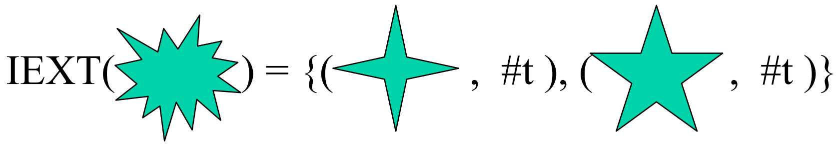


RDF/S Interpretations (1)
Mapping Uriefs & Litterals to Resources



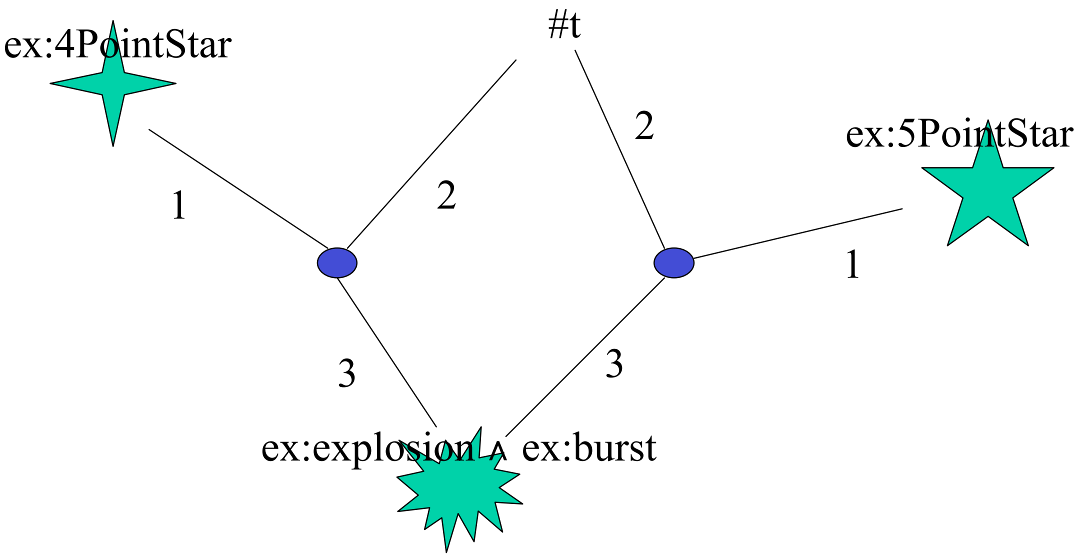
RDF/S Interpretations (2)

The IEXT Relation: Encoding Properties



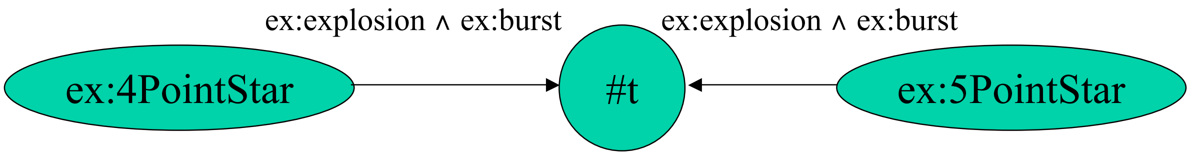
RDF/S Interpretations (3)

Encoding Interpretations with Labelled Graphs



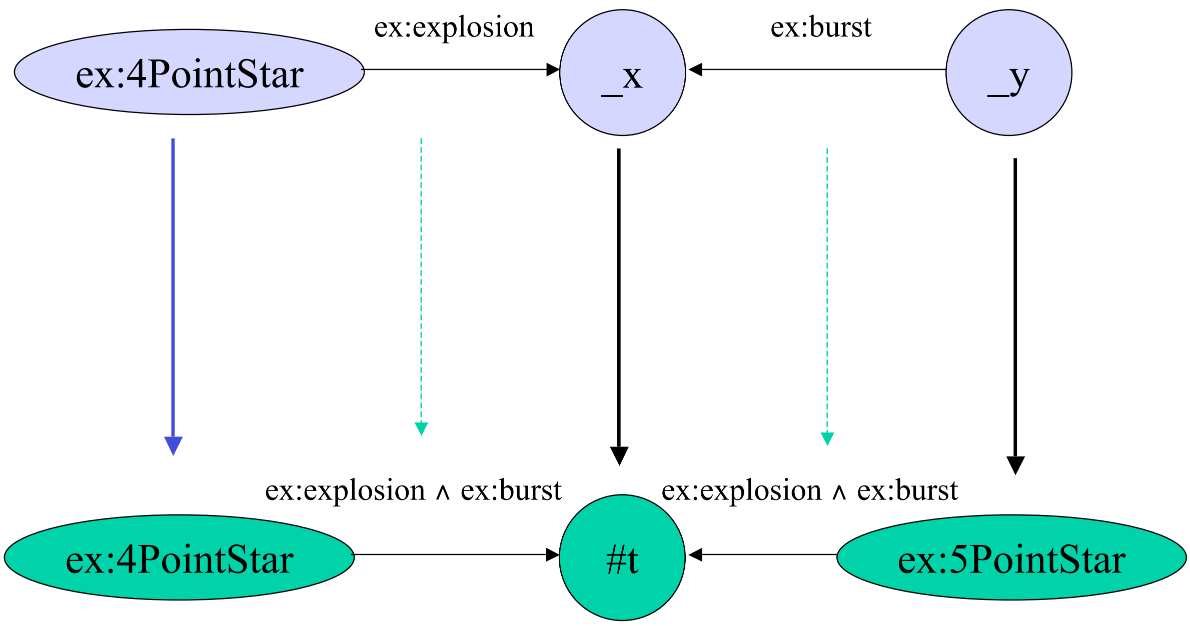
RDF/S Interpretations (4)

Fine Tuning: The Interpretation Graph



<u>Simple</u> RDF Models (1)

The Interpretation Lemma

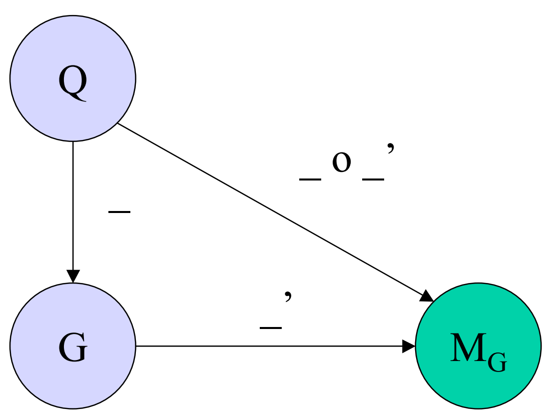


Characterization of Entailment (1)
Main Theorem

- <u>Definition:</u> An RDF graph G (simply) entails an RDF graph Q iff every (simple) interpretation that is a model of G is also a model of Q.

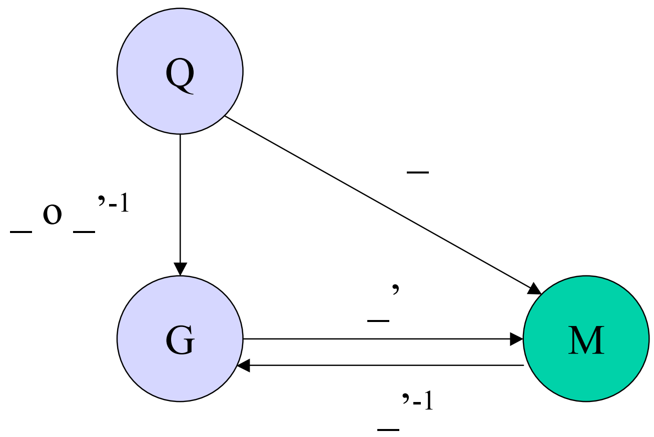
- <u>Theorem:</u> An RDF graph G simply entails an RDF graph Q iff there exists a projection from the normal form of Q into the normal form of G.

Characterization of Entailment (2)
Idea of Proof (⇐)



Characterization of Entailment (2)

Idea of Proof (⇒)



RDF Interpretations (1)

Extract from the book

RDF semantic conditions.

x is in IP if and only if <x, I(rdf:Property)> is in IEXT(I(rdf:type))

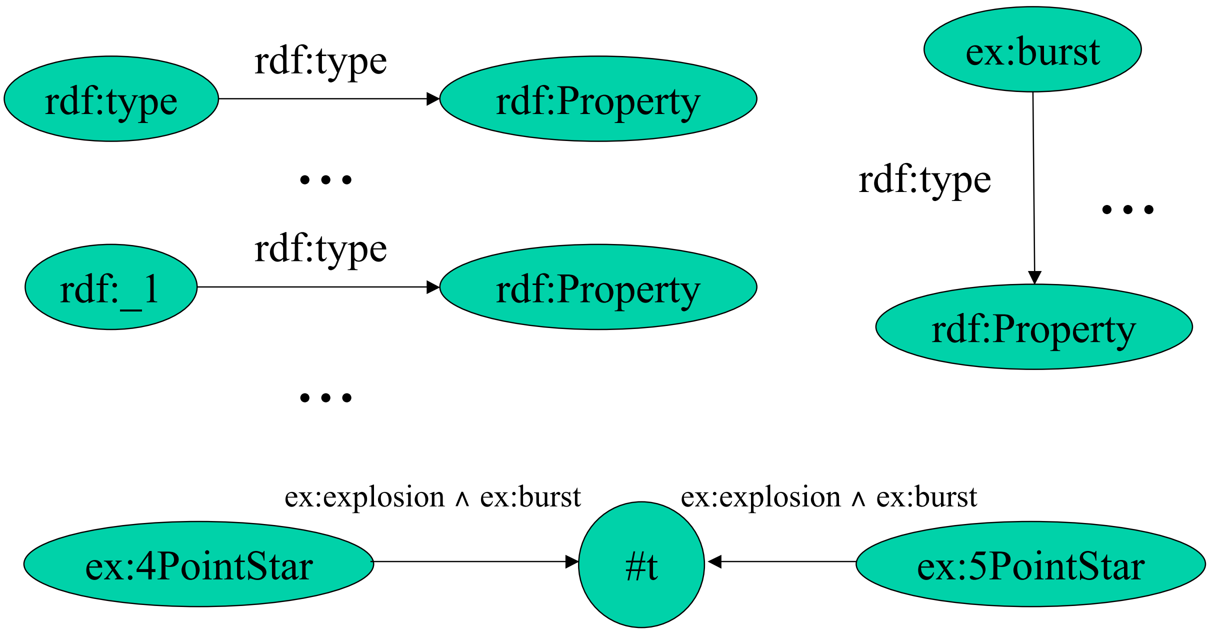
If "xxx"^^rdf:XMLLiteral is in V and xxx is a well-typed XML literal string, then
IL("xxx"^^rdf:XMLLiteral) is the XML value of xxx;
IL("xxx"^^rdf:XMLLiteral) is in LV;
IEXT(I(rdf:type)) contains <IL("xxx"^^rdf:XMLLiteral), I(rdf:XMLLiteral)>

If "xxx"^^rdf:XMLLiteral is in V and xxx is an ill-typed XML literal string, then
IL("xxx"^^rdf:XMLLiteral) is not in LV;
IEXT(I(rdf:type)) does not contain <IL("xxx"^^rdf:XMLLiteral), I(rdf:XMLLiteral)>.

RDF axiomatic triples.

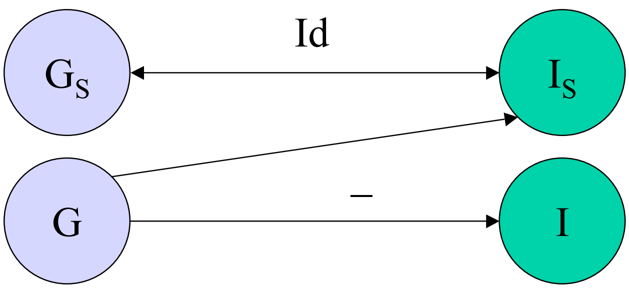
rdf:type rdf:type rdf:Property .
rdf:subject rdf:type rdf:Property .
rdf:predicate rdf:type rdf:Property .
rdf:object rdf:type rdf:Property .
rdf:first rdf:type rdf:Property .
rdf:rest rdf:type rdf:Property .
rdf:value rdf:type rdf:Property .
rdf:_1 rdf:type rdf:Property .
rdf:_2 rdf:type rdf:Property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

RDF Interpretations (2)
A graphical Explanation



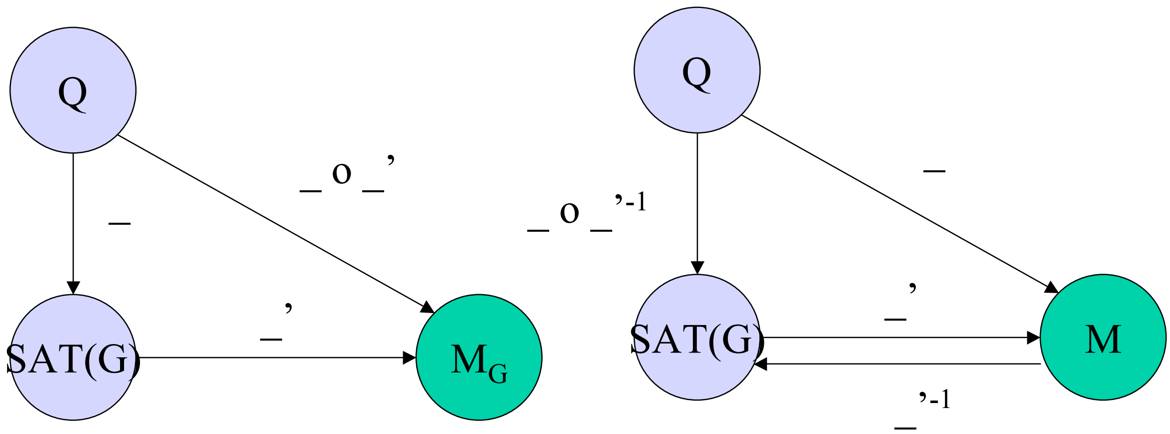
RDF Saturation of a graph
The RDF Interpretation Lemma

- <u>Lemma:</u> An RDF interpretation is a model for an RDF graph G iff there is a projection from $SAT_{RDF}(G)$ into the interpretation graph.



RDF Entailment

What’s changing in the main theorem?



But $SAT_{RDF}(G)$ is an infinite graph!

- Do not create « useless » axiomatic triples
- Do not create « useless » XMLLitteral triples

- Saturation is then linear in the size of G

- PROBLEM: We may lose an infinite number of projections/proofs. Are they intersting anyway?

RDFS Interpretations (1)
Axiomatic triples

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 .

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 .

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 .
...

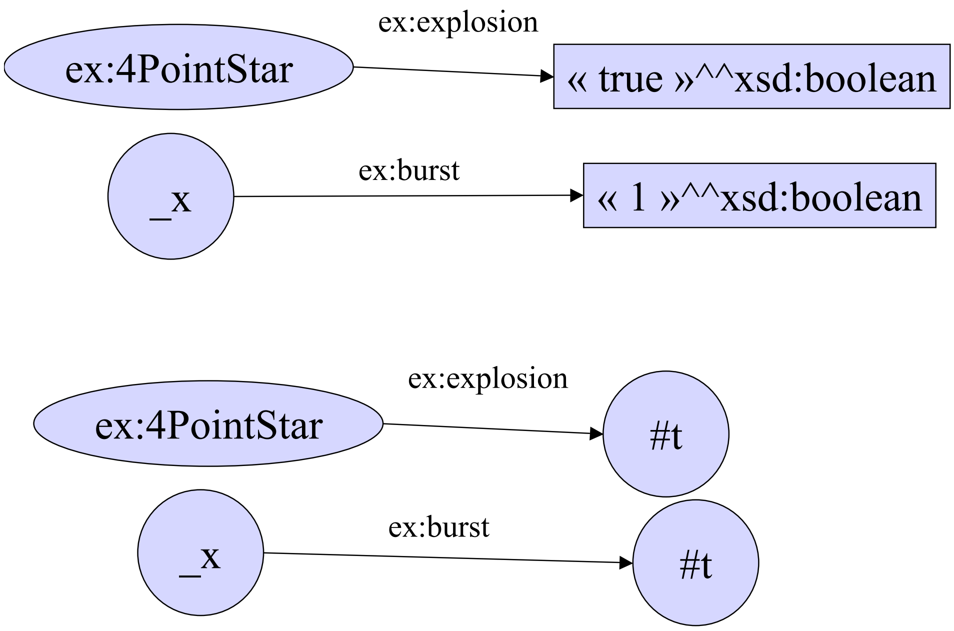
RDFS Interpretations (2)
Semantic conditions

x is in ICEXT(y) if and only if <x,y> is in IEXT(I(rdf:type)) IC = ICEXT(I(rdfs:Class)) IR = ICEXT(I(rdfs:Resource)) LV = ICEXT(I(rdfs:Literal))
If <x,y> is in IEXT(I(rdfs:domain)) and <u,v> is in IEXT(x) then u is in ICEXT(y)
If <x,y> is in IEXT(I(rdfs:range)) and <u,v> is in IEXT(x) then v is in ICEXT(y)
IEXT(I(rdfs:subPropertyOf)) is transitive and reflexive on IP
If <x,y> is in IEXT(I(rdfs:subPropertyOf)) then x and y are in IP and IEXT(x) is a subset of IEXT(y)
If x is in IC then <x, I(rdfs:Resource)> is in IEXT(I(rdfs:subClassOf))
If <x,y> is in IEXT(I(rdfs:subClassOf)) then x and y are in IC and ICEXT(x) is a subset of ICEXT(y)
IEXT(I(rdfs:subClassOf)) is transitive and reflexive on IC
If x is in ICEXT(I(rdfs:ContainerMembershipProperty)) then: < x, I(rdfs:member)> is in IEXT(I(rdfs:subPropertyOf))
If x is in ICEXT(I(rdfs:Datatype)) then <x, I(rdfs:Literal)> is in IEXT(I(rdfs:subClassOf))

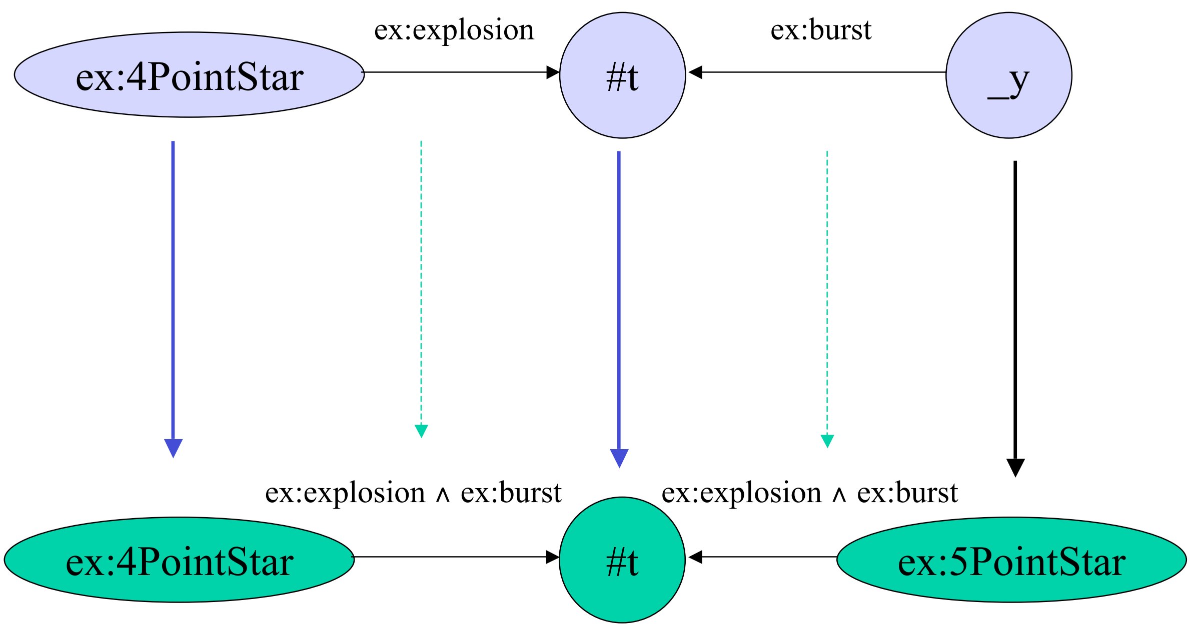
Datatype interpretations (1)
Semantic conditions

if <aaa,x> is in D then I(aaa) = x
if <aaa,x> is in D then ICEXT(x) is the value space of x and is a subset of LV
if <aaa,x> is in D then for any typed literal "sss"^^ddd in V with I(ddd) = x , if sss is in the lexical space of x then IL("sss"^^ddd) = L2V(x)(sss), otherwise IL("sss"^^ddd) is not in LV
if <aaa,x> is in D then I(aaa) is in ICEXT(I(rdfs:Datatype))

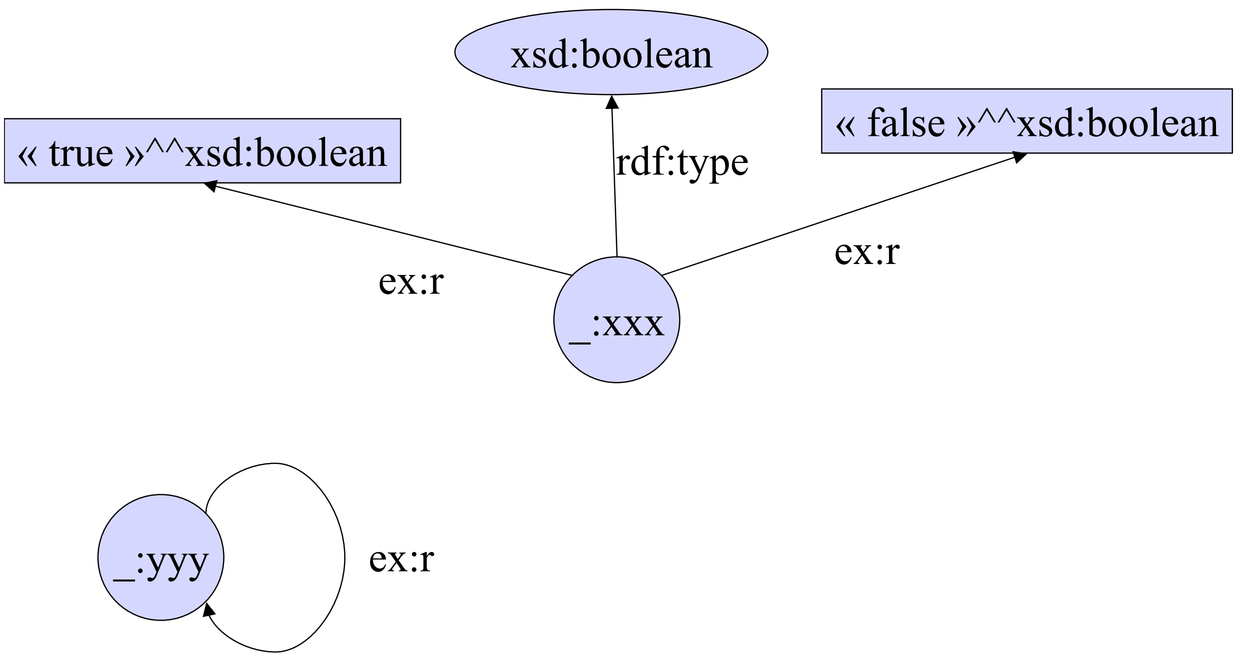
Datatype Interpretations (2)
Normal Form & Typed Litterals (a)



Datatype Interpretations (2)
Normal Form & Typed Litterals (b)



Datatype interpretations (3)
Graph matching is irrelevant!



Benefits from this reformulation

- Hopefully an easier presentation of RDF(S) semantics, and far shorter proofs!
- Reformulation as a graph homomorphism is a first step to develop efficient algorithms
- A good way to establish relationships with other KR formalisms (CG, CSP)

Use for WP 2.5.3 ?

- Computational complexity
  - NP complete
  - Polynomial when the query is a tree (or can be decomposed into a tree – Gottlob et al 99)
- Efficient algorithms for Simple Entailment
  - BT optimizations coming from the CSP community
  - Adaptation to CG done (Baget 02), the same for RDF
- Use RDF Rules for saturation
  - Have to test efficiency of backward chaining algorithm for that
  - Anyway, a nice graphical explanation of semantics