The Logic of the Semantic Web

Enrico Franconi

Free University of Bozen-Bolzano, Italy

What is this talk about

What is this talk about

- A sort of tutorial of RDF, the core semantic web knowledge representation language (data model)
- Emphasises the logical aspect...
- ...necessary to introduce in this context constraints, e.g., (conceptual) schemas, ontologies, etc
- Most material taken from my PODS-06 invited talk

Technical Content

- RDF as a novel KR language
  - with meta-modelling capabilities
  - with challenging theoretical problems
- SPARQL as the RDF query language
  - a query language for incomplete information
  - with interesting capabilities
- I'll present an abstract logic-based framework
- DISCLAIMER:
  - the field is very young
  - the current main effort is about clarifying issues

Summary

Summary

- RDF as a KR language

Summary

- RDF as a KR language
- The semantics of RDF

Summary

- RDF as a KR language
- The semantics of RDF
- RDF as a representation system for incomplete information: naive tables and conjunctive queries

Summary

- RDF as a KR language
- The semantics of RDF
- RDF as a representation system for incomplete information: naive tables and conjunctive queries
- RDF with constraints: RDFS

Summary

- RDF as a KR language
- The semantics of RDF
- RDF as a representation system for incomplete information: naive tables and conjunctive queries
- RDF with constraints: RDFS
- Querying RDF incomplete databases

Summary

- RDF as a KR language
- The semantics of RDF
- RDF as a representation system for incomplete information: naive tables and conjunctive queries
- RDF with constraints: RDFS
- Querying RDF incomplete databases
- SPARQL as the query language for RDF

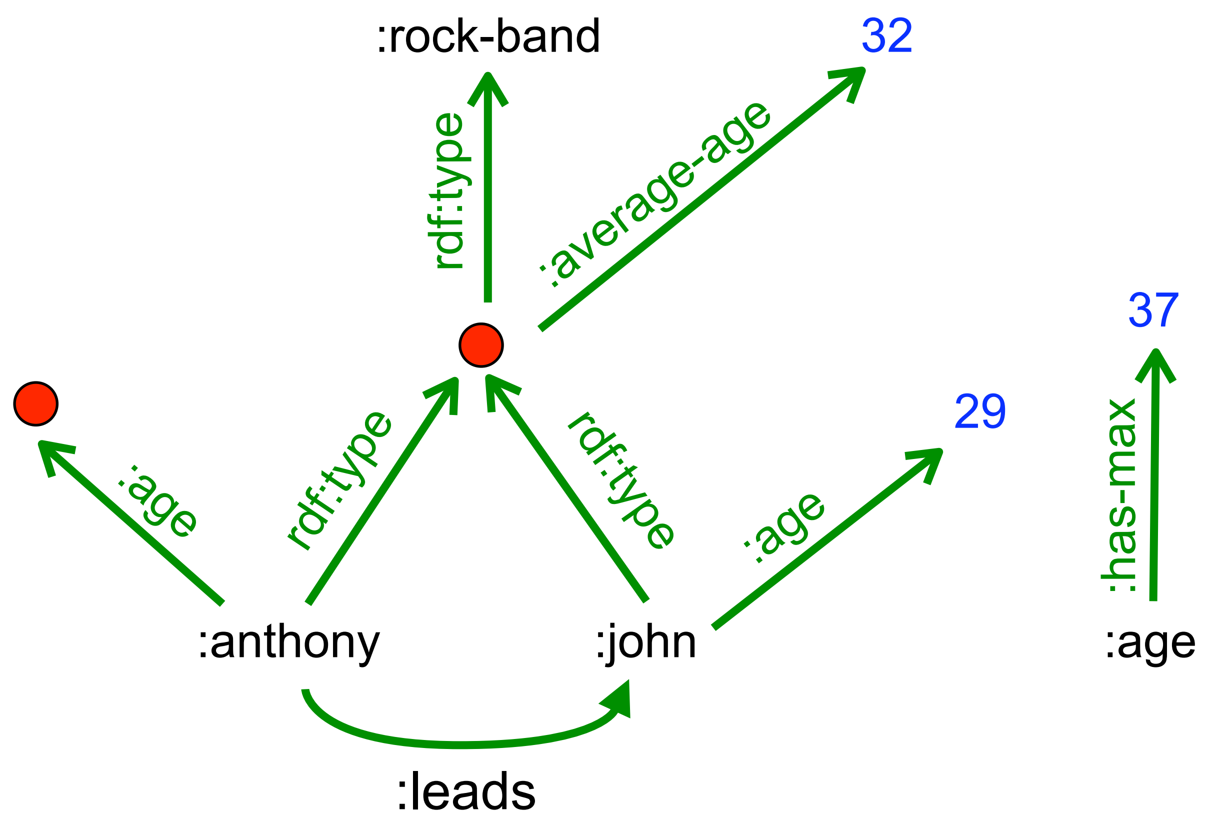
RDF in the real world

- RDF and SPARQL are W3C standards
- Widespread use for metadata representation, e.g.
  - Apple (MCF)
  - Adobe (XMP)
  - Mozilla/Firefox
- Oracle supports RDF, and provides an extension of SQL to query RDF data
- HP has a big lab (in Bristol) developing specialised data stores for RDF (Jena)
- ...but: research is beyond practice

RDF

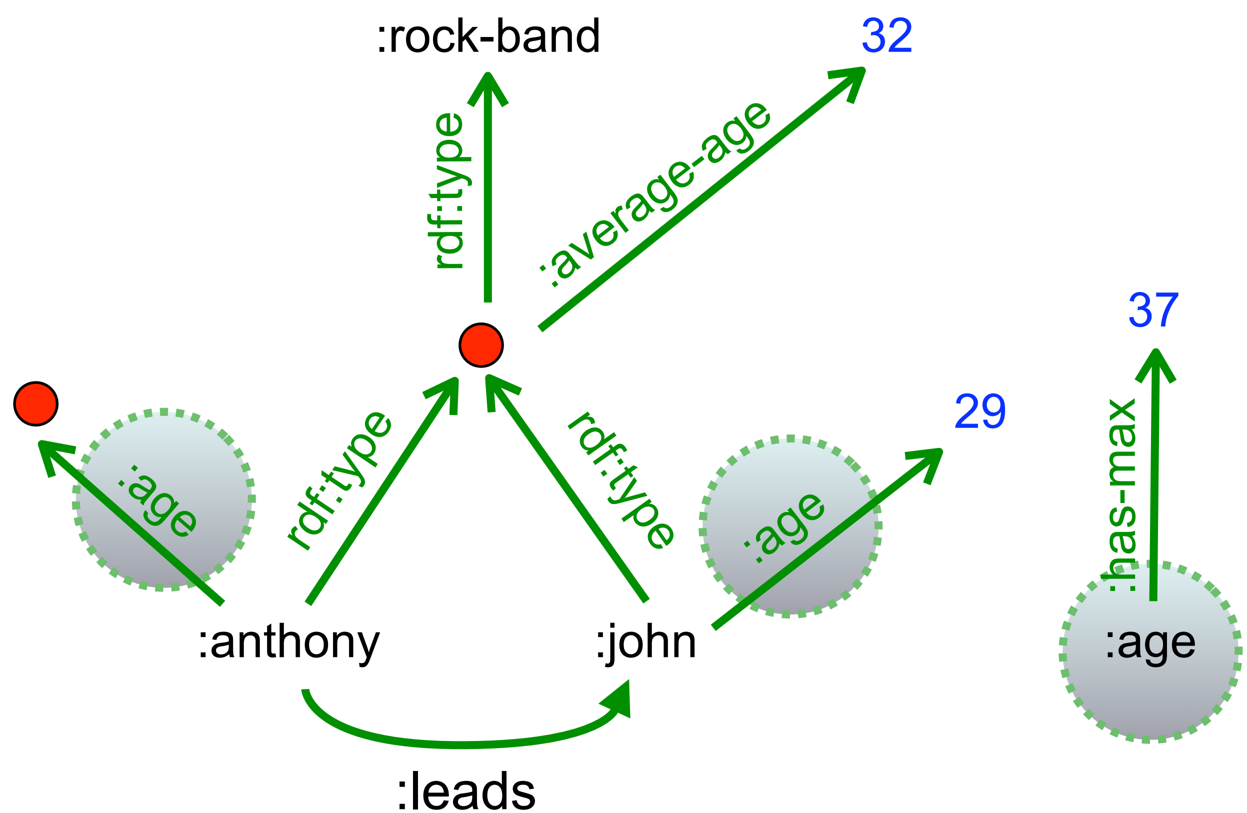
- A node- and edge-labelled directed graph
  - edges are called properties
  - the left node in a directed edge is called subject
  - the right node in a directed edge is called object
- Typical notations are:
  - **p(s,o)**
  - **Triple(s, p, o)**
  - **s p o.**
- Labels are URIs, literals, or bnodes

Example



:antony :leads :john.
:antony :age _:a.
:antony rdf:type _:b.
_:b rdf:type :rock-band.
:age :has-max 37.
...

Example



:antony :leads :john.
:antony :age _:a.
:antony rdf:type _:b.
_:b rdf:type :rock-band.
:age :has-max 37.
...

RDF peculiarities

- Some labels are anonymous: bnodes
- The alphabets of labels for nodes and for properties are not disjoint: a coreference is possible between nodes and properties
- There is a special pre-defined non well-founded "**rdf:type**" property, with the intended meaning of "is-element-of"

Meaning of RDF graphs

- We want to provide a model-theoretic semantics to RDF graphs, in order to properly define entailment and query answering
- We consider here a simplified RDF language:
  - no restrictions on literals
    - in normative RDF literals are not allowed in subject position
  - no restrictions on properties
    - in normative RDF bnodes are not allowed in property position
  - no “axiomatic” knowledge

RDF semantics (atoms)

$$\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, \cdot^{\mathcal{I}_p} \rangle$$

$$\begin{aligned}
\cdot^{\mathcal{I}} &: \quad \mathbb{U} \cup \mathbb{L} \mapsto \Delta^{\mathcal{I}} \\
\cdot^{\mathcal{I}_p} &: \quad \Delta^{\mathcal{I}} \mapsto 2^{\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}} \\
\alpha &: \quad \mathbb{B} \mapsto \Delta^{\mathcal{I}}
\end{aligned}$$

RDF semantics (atoms)

$$\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, \cdot^{\mathcal{I}_p} \rangle$$

$$\begin{aligned}
\cdot^{\mathcal{I}} &: \quad \mathbb{U} \cup \mathbb{L} \mapsto \Delta^{\mathcal{I}} \\
\cdot^{\mathcal{I}_p} &: \quad \Delta^{\mathcal{I}} \mapsto 2^{\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}} \\
\alpha &: \quad \mathbb{B} \mapsto \Delta^{\mathcal{I}}
\end{aligned}$$

$$\begin{aligned}
u^{\mathcal{I},\alpha} &= u^{\mathcal{I}} \\
l^{\mathcal{I},\alpha} &= l^{\mathcal{I}} \\
b^{\mathcal{I},\alpha} &= \alpha(b)
\end{aligned}$$

RDF semantics (atoms)

$$\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, \cdot^{\mathcal{I}_p} \rangle$$

$$\begin{array}{rcl} \cdot^{\mathcal{I}} & : & \mathbb{U} \cup \mathbb{L} \mapsto \Delta^{\mathcal{I}} \\ \cdot^{\mathcal{I}_p} & : & \Delta^{\mathcal{I}} \mapsto 2^{\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}} \\ \alpha & : & \mathbb{B} \mapsto \Delta^{\mathcal{I}} \end{array} \qquad \begin{array}{rcl} u^{\mathcal{I},\alpha} & = & u^{\mathcal{I}} \\ l^{\mathcal{I},\alpha} & = & l^{\mathcal{I}} \\ b^{\mathcal{I},\alpha} & = & \alpha(b) \end{array}$$

$$\boxed{\mathcal{I}, \alpha \models p(s, o) \quad \text{iff} \quad \langle s^{\mathcal{I},\alpha}, o^{\mathcal{I},\alpha} \rangle \in (p^{\mathcal{I},\alpha})^{\mathcal{I}_p}}$$

RDF semantics (atoms)

$$\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, \cdot^{\mathcal{I}_p} \rangle$$

$$\begin{array}{lcl} \cdot^{\mathcal{I}} & : & \mathbb{U} \cup \mathbb{L} \mapsto \Delta^{\mathcal{I}} \\ \cdot^{\mathcal{I}_p} & : & \Delta^{\mathcal{I}} \mapsto 2^{\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}} \\ \alpha & : & \mathbb{B} \mapsto \Delta^{\mathcal{I}} \end{array} \qquad \begin{array}{rcl} u^{\mathcal{I},\alpha} & = & u^{\mathcal{I}} \\ l^{\mathcal{I},\alpha} & = & l^{\mathcal{I}} \\ b^{\mathcal{I},\alpha} & = & \alpha(b) \end{array}$$

$$\boxed{\mathcal{I}, \alpha \models p(s,o) \quad \text{iff} \quad \langle s^{\mathcal{I},\alpha}, o^{\mathcal{I},\alpha} \rangle \in (p^{\mathcal{I},\alpha})^{\mathcal{I}_p}}$$

$$\mathcal{I}, \alpha \models p(s,o) \quad \text{iff} \quad \langle s^{\mathcal{I},\alpha}, o^{\mathcal{I},\alpha} \rangle \in p^{\mathcal{I}_p,\alpha} \quad \text{(FOL)}$$

RDF semantics (atoms)

$$\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, \cdot^{\mathcal{I}_p} \rangle$$

$$\begin{array}{rcl} \cdot^{\mathcal{I}} & : & \mathbb{U} \cup \mathbb{L} \mapsto \Delta^{\mathcal{I}} \\ \cdot^{\mathcal{I}_p} & : & \Delta^{\mathcal{I}} \mapsto 2^{\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}} \\ \alpha & : & \mathbb{B} \mapsto \Delta^{\mathcal{I}} \end{array} \qquad \begin{array}{rcl} u^{\mathcal{I},\alpha} & = & u^{\mathcal{I}} \\ l^{\mathcal{I},\alpha} & = & l^{\mathcal{I}} \\ b^{\mathcal{I},\alpha} & = & \alpha(b) \end{array}$$

$$\boxed{\mathcal{I}, \alpha \models p(s,o) \quad \text{iff} \quad \langle s^{\mathcal{I},\alpha}, o^{\mathcal{I},\alpha} \rangle \in (p^{\mathcal{I},\alpha})^{\mathcal{I}_p}}$$

$$\mathcal{I}, \alpha \models p(s,o) \quad \text{iff} \quad \langle s^{\mathcal{I},\alpha}, p^{\mathcal{I},\alpha}, o^{\mathcal{I},\alpha} \rangle \in T^{\mathcal{I}}$$

RDF semantics and entailment

- Non-atomic formulas:

$$\mathcal{I}, \alpha \models \{p_1(s_1, o_1), p_2(s_2, o_2), \cdots\} \quad \underline{\text{iff}}$$

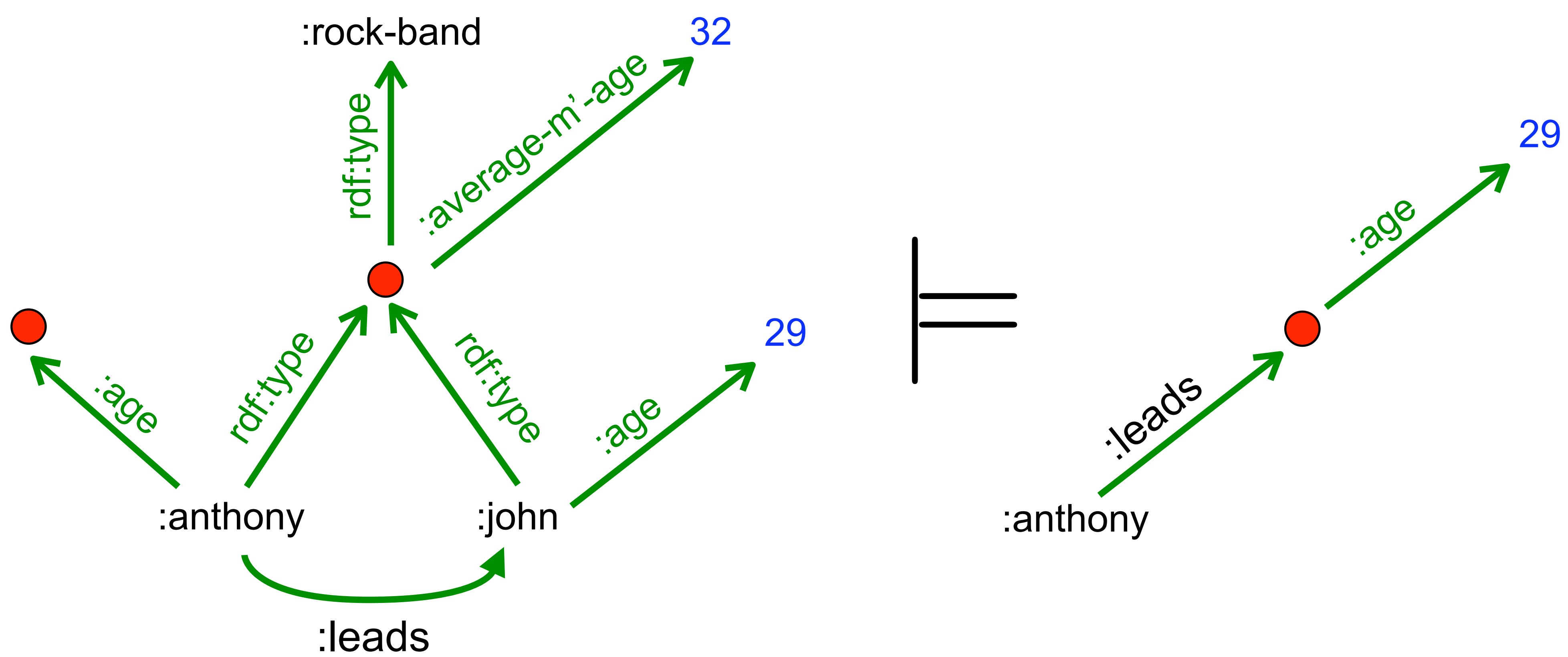
$$\mathcal{I}, \alpha \models p_1(s_1, o_1) \quad \text{and}$$

$$\mathcal{I}, \alpha \models \{p_2(s_2, o_2), \cdots\}$$

- $\mathcal{I}$ is a model of an RDF graph G, written $\mathcal{I} \models G$, if there exists an α such that $\mathcal{I}, \alpha \models G$

- An RDF graph G entails an RDF graph H ($G \models H$) iff for any $\mathcal{I}$ such that $\mathcal{I} \models G$ then $\mathcal{I} \models H$

Example



:rock-band
32
rdf:type
:average-m’-age
:age
rdf:type
rdf:type
29
:age
:anthony
:john
:leads
29
:age
:leads
:anthony

RDF and FOL

[_, Tessaris, 2004] The models of an RDF graph

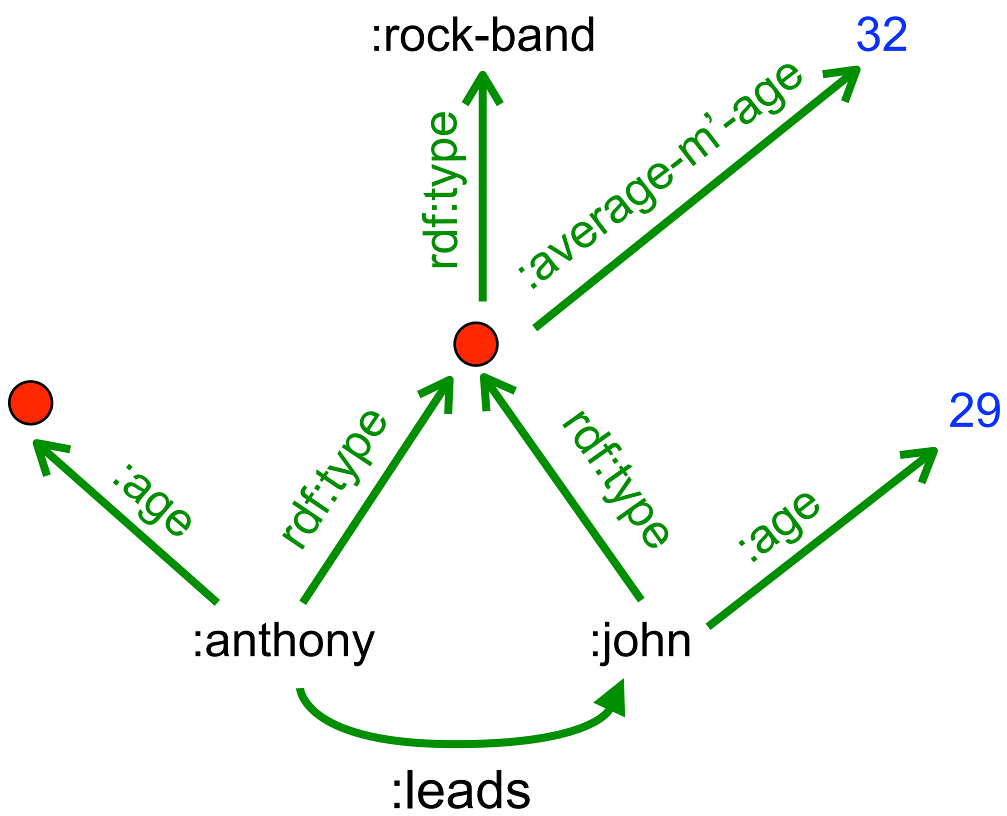
$$\mathcal{I} \models \{p_1(s_1, o_1), p_2(s_2, o_2), \cdots\}$$

are the same as the models of the FOL formula

$$\mathcal{I} \models_{\text{FOL}} \exists \bar{b}.\ T(s_1, p_1, o_1) \wedge T(s_2, p_2, o_2) \wedge \cdots$$

where $\bar{b}$ is the set of bnode names appearing in the graph

Example



$\exists$x,y. T(:antony, :leads, :john) $\wedge$ T(:antony, :age, x) $\wedge$ T(:antony, rdf:type, y) $\wedge$

T(y, rdf:type :rock-band) $\wedge$ ...

Complexity of RDF entailment

$$G \models H$$

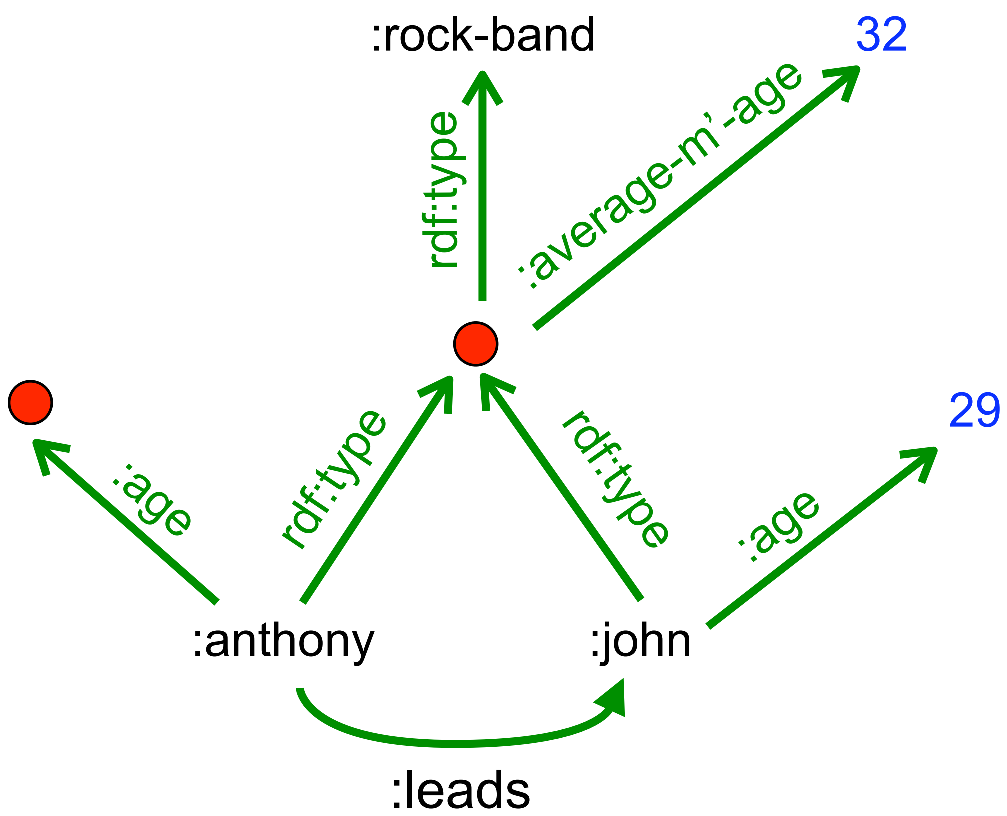
- NP-complete in the size of the graphs
- Polynomial in the size of the entailing graph G
- Algorithm: reduction to conjunctive query containment
  - Typically implemention: graph homomorphism

Naive Tables/Conjunctive Queries

- An RDF graph can be seen as an incomplete database represented in the form of a naive table
- An RDF graph is represented by a unique table T, with values being constants or named existential variables

- An RDF graph can be seen as a boolean conjunctive query over the unique relation T

Example



T

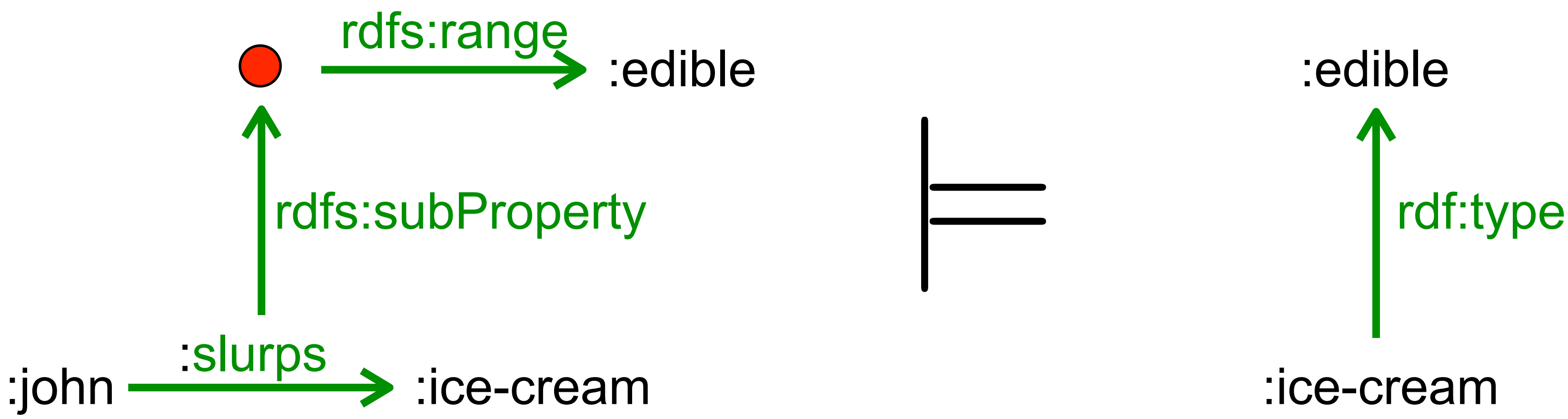
:antony	:leads	:john
:antony	:age	X
:antony	rdf:type	Y
Y	rdf:type	:rock-band
…		

∃x,y. T(:antony, :leads, :john) ∧ T(:antony, :age, x) ∧ T(:antony, rdf:type, y) ∧
T(y, rdf:type :rock-band) ∧ ...

RDFS

- RDFS adds to the signature properties with a fixed semantics
  - **rdf:type** (= is-element-of)
  - **rdfs:subclass**
  - **rdfs:subproperty**
  - **rdfs:domain**
  - **rdfs:range**
- Note that the (above) properties are also elements of the domain

Example


rdfs:range
:edible
rdfs:subProperty
:john
:slurps
:ice-cream
:edible
rdf:type
:ice-cream

Normative semantics of RDFS

$$\text{rdfs:subclass}^{\mathcal{I}_p,\alpha} \subseteq$$
$$\{\langle u,v\rangle \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid$$
$$\forall x.\ (x,u) \in \text{rdf:type}^{\mathcal{I}_p,\alpha} \rightarrow (x,v) \in \text{rdf:type}^{\mathcal{I}_p,\alpha}\}$$

$$\text{rdfs:domain}^{\mathcal{I}_p,\alpha} \subseteq$$
$$\{\langle u,v\rangle \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid$$
$$\forall x,y.\ (x,y) \in u^{\mathcal{I}_p,\alpha} \rightarrow (x,v) \in \text{rdf:type}^{\mathcal{I}_p,\alpha}\}$$

Normative semantics of RDFS (in FOL)

$$\forall u, v.$$
$$T(u, \text{rdfs:subclass}, v) \rightarrow$$
$$\forall x.\ T(x, \text{rdf:type}, u) \rightarrow T(x, \text{rdf:type}, v)$$

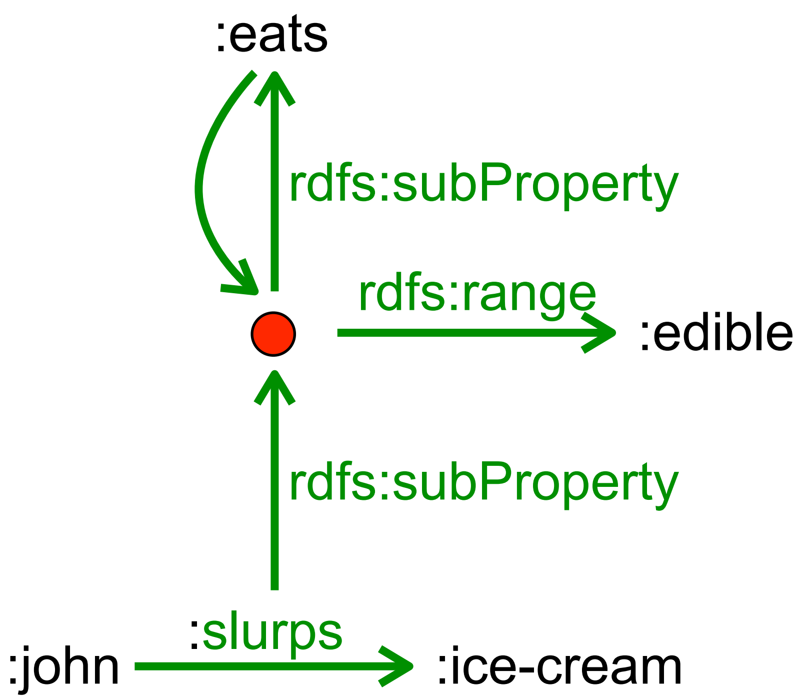
$$\forall u, v.$$
$$T(u, \text{rdfs:domain}, v) \rightarrow$$
$$\forall x, y.\ T(x, u, y) \rightarrow T(x, \text{rdf:type}, v)$$

Entailment in normative RDFS

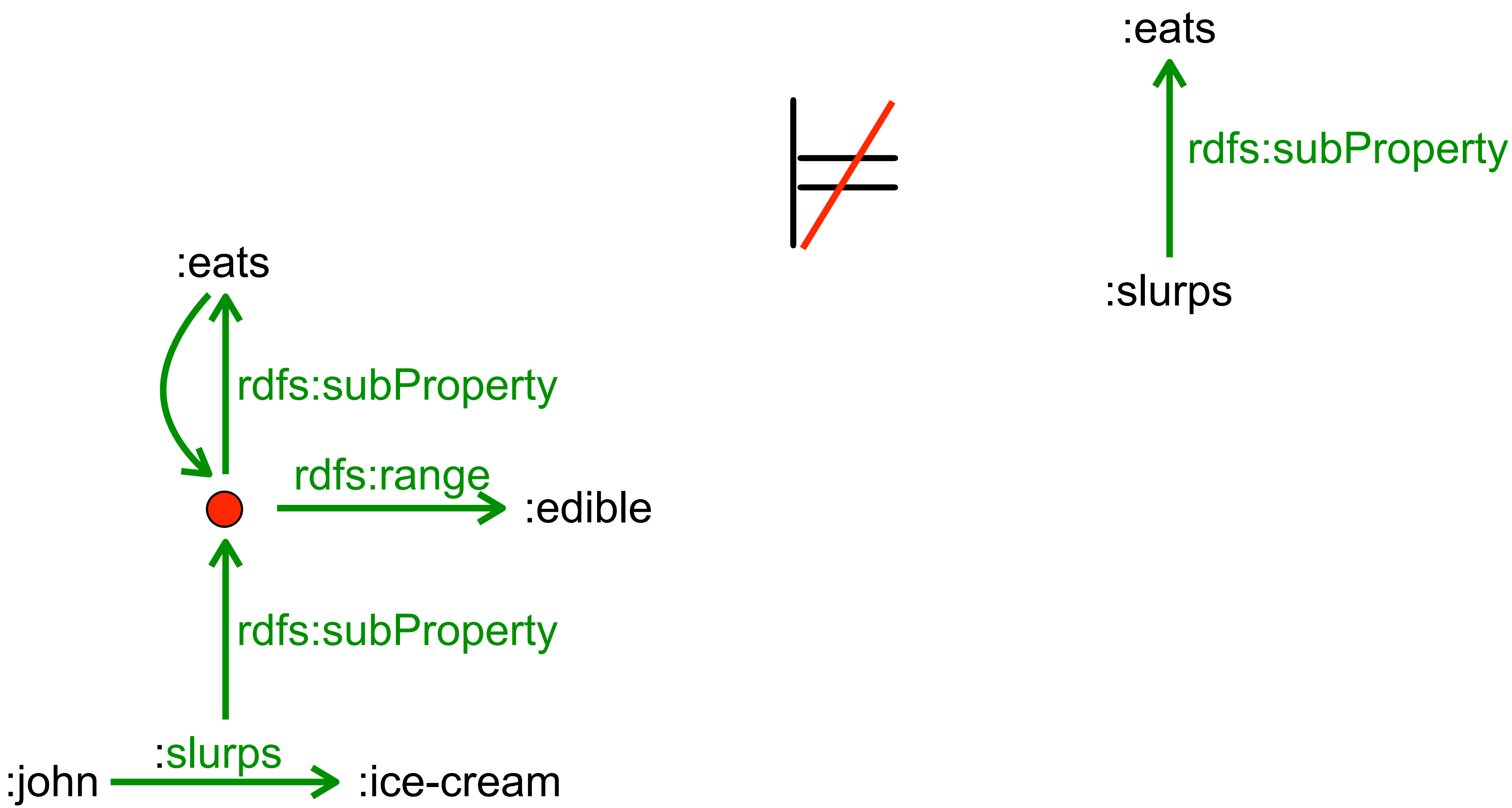
$$G \models_{\mathrm{RDFS}} H$$

- Entailment under constraints
- [ter Horst, 2005] NP-complete in the size of the graphs
  - Polynomial if H does not contain bnodes
- Algorithm: reduction to RDF entailment through a completion of graph G
  - Warning: W3C standard algorithm (by P. Hayes) is incomplete [ter Horst, 2005; Gutiérrez et al, 2004] (e.g., the previous example does not work)

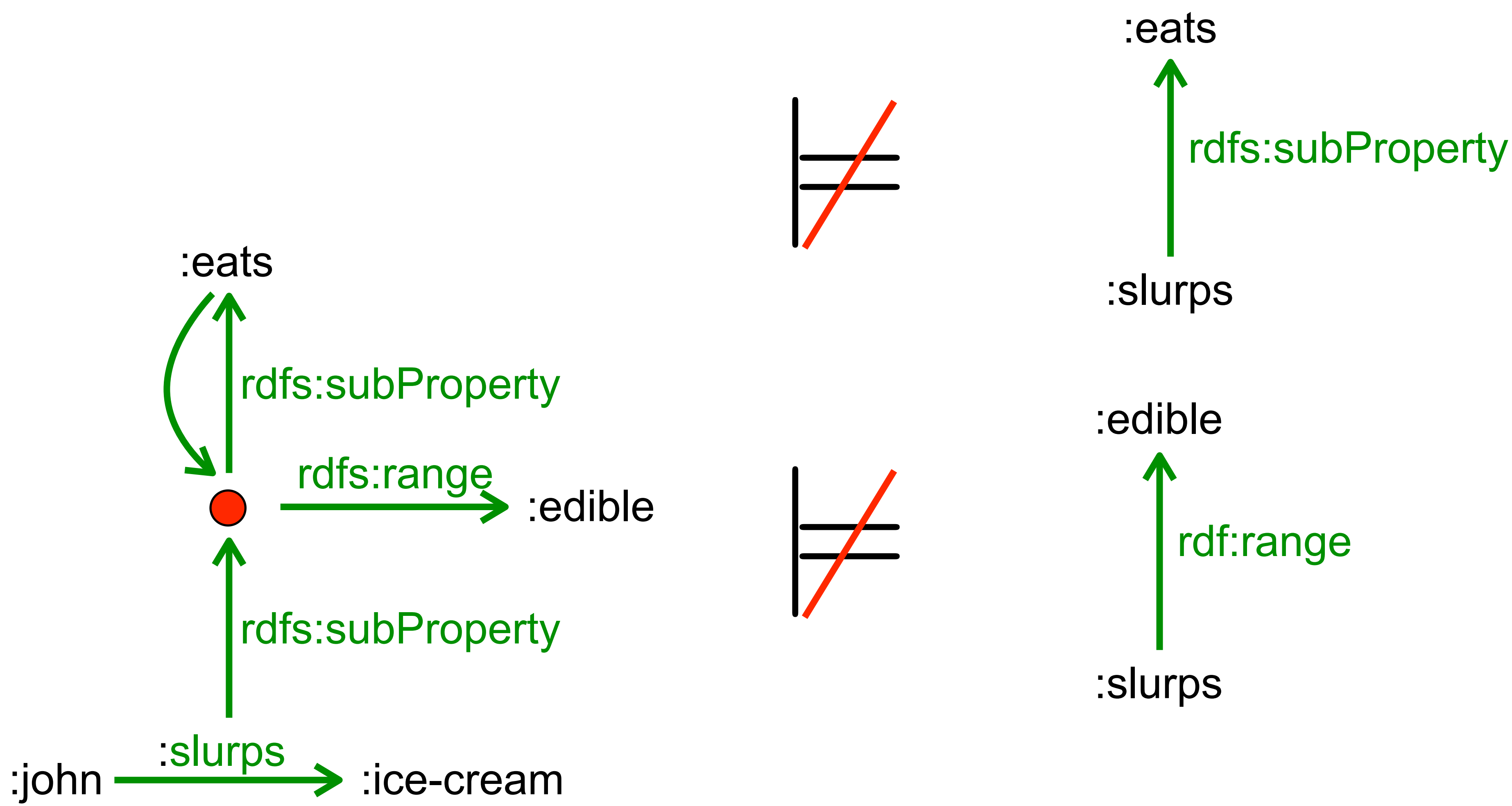
Example revisited ☹



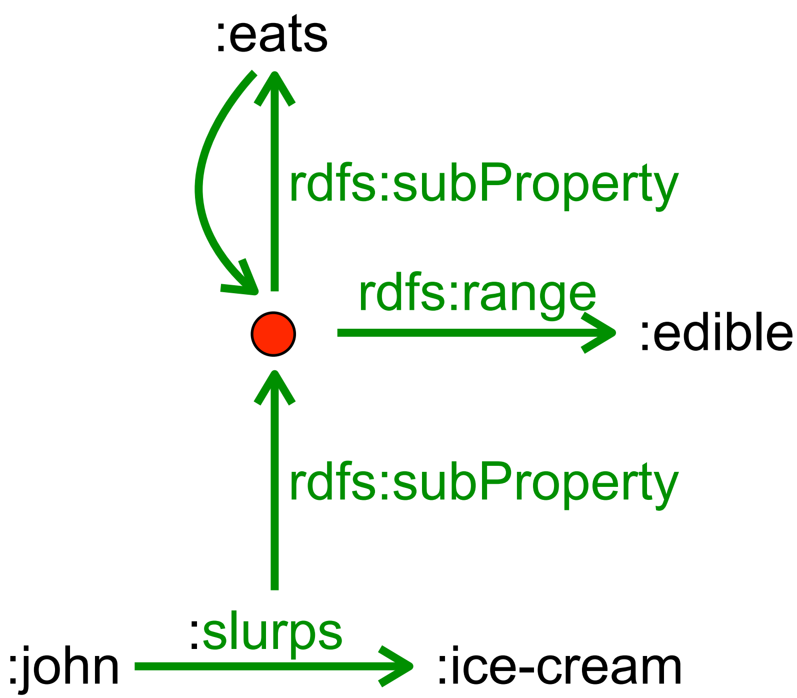
Example revisited ☹

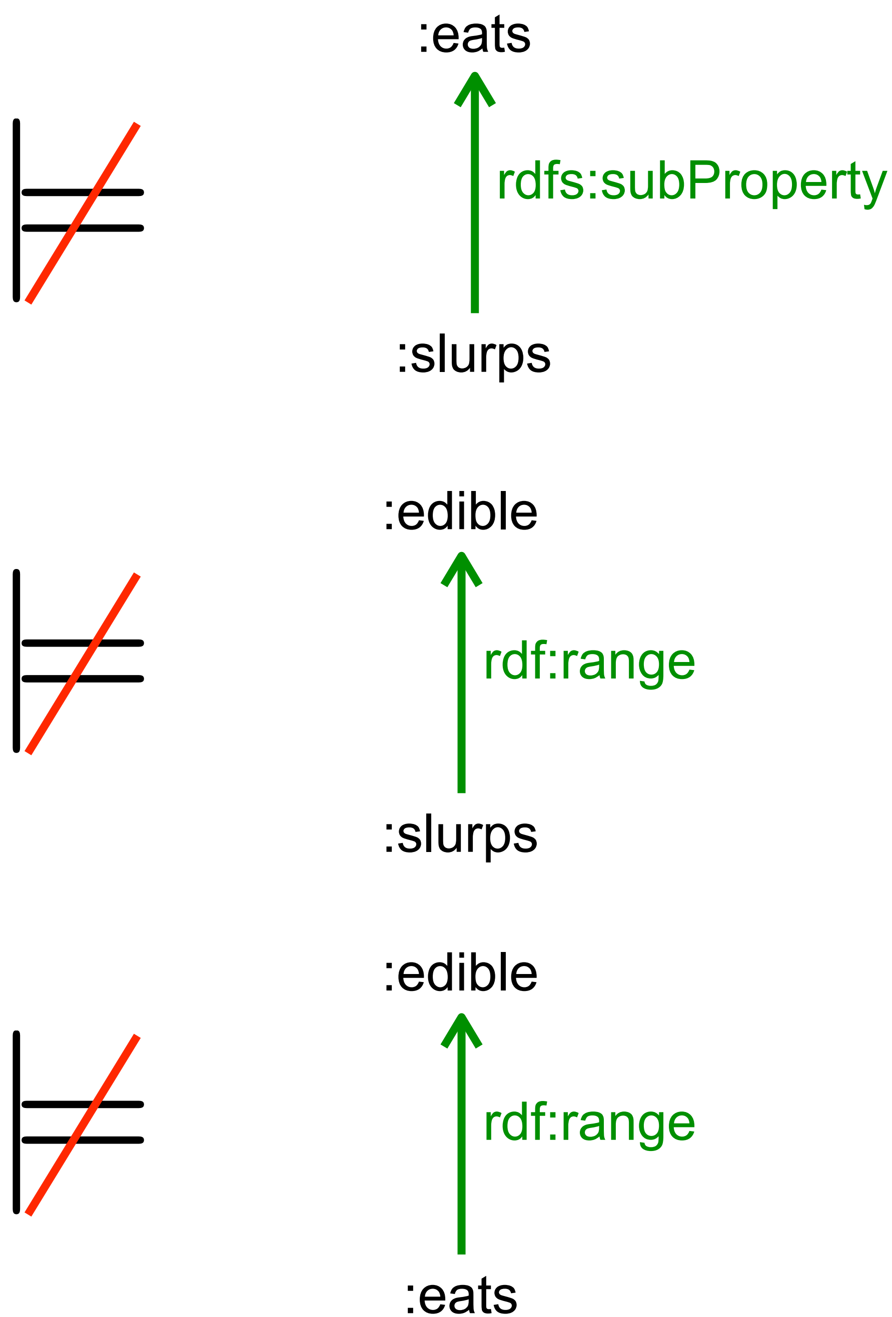


Example revisited ☹



Example revisited ☹





Extensional semantics of RDFS

$$\forall u, v.$$
$$T(u, \text{rdfs:subclass}, v) \leftrightarrow$$
$$(\forall x.\ T(x, \text{rdf:type}, u) \rightarrow T(x, \text{rdf:type}, v))$$

$$\forall u, v.$$
$$T(u, \text{rdfs:domain}, v) \leftrightarrow$$
$$(\forall x, y.\ T(x, u, y) \rightarrow T(x, \text{rdf:type}, v))$$

Entailment in extensional RDFS

$$G \models_{\mathrm{RDFS}^e} H$$

- Entailment under constraints
- General algorithm not known
- Complexity known only if H does not contain bnodes:
  - Theorem [_,Rosati, 2006 - unpublished]: polynomial in the size of the graphs

RDF & KR

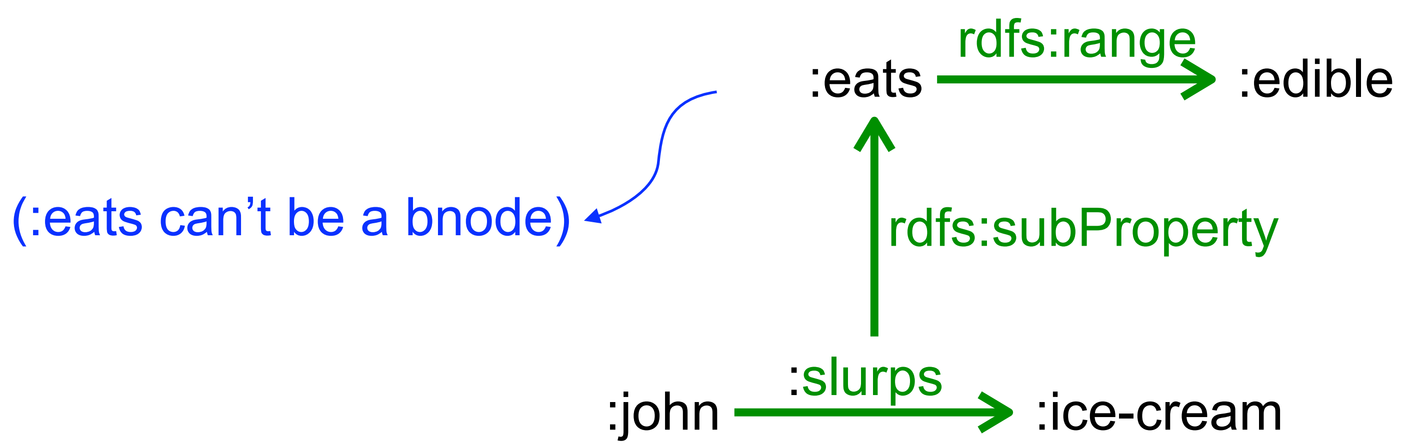
- The representation does not correspond to the expected representation in logic-based KR:
  - can we rewrite **T(i, rdf:type, c)** as **c(i)**?
  - can we rewrite **T(s, p, o)** as **p(s,o)**?
- Theorem [_, Tessaris, 2005]
  - if there are no bnodes in class/property position
  - then there is a 1-to-1 correspondence between the models of the RDFS graph, and the models of a FOL theory containing the RDFS graph without the RDFS vocabulary and with the constraints instantiated to the existing RDFS properties in the RDF graph
  - Decidability and complexity known (from DLs)

RDF & KR

- The representation does not correspond to the expected representation in logic-based KR:
  - can we rewrite **T(i, rdf:type, c)** as **c(i)**?
  - can we rewrite **T(s, p, o)** as **p(s,o)**?
- Theorem [_, Tessaris, 2005]

  HiLog
  - ~~if there are no bnodes in class/property position~~
  - then there is a 1-to-1 correspondence between the models of the RDFS graph, and the models of a ~~FOL~~ theory containing the RDFS graph without the RDFS vocabulary and with the constraints instantiated to the existing properties in the RDF graph
  - Decidability and complexity known (from DLs)

Example



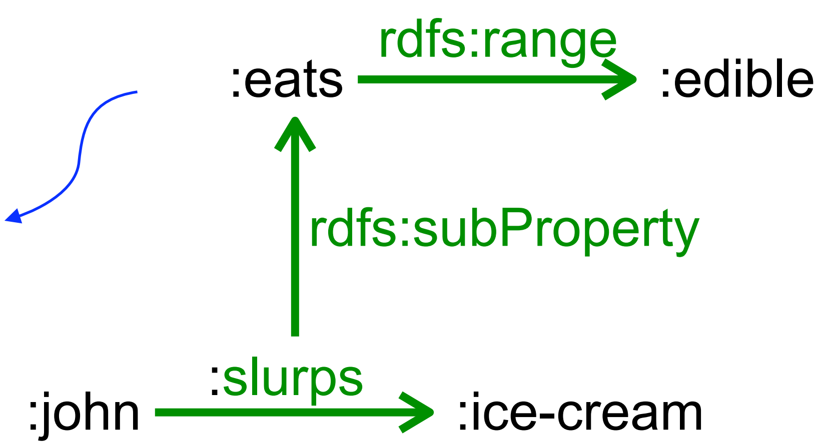
T(:john, :slurps, :ice-cream)

T(:slurps, :rdfs:subproperty, :eats)

T(:eats, :rdfs:range, :edible)

Example

(:eats can’t be a bnode)



T(:john, :slurps, :ice-cream)

T(:slurps, :rdfs:subproperty, :eats)

T(:eats, :rdfs:range, :edible)

:slurps(:john, :ice-cream)

$\forall$x,y. :slurps(x, y) $\rightarrow$:eats(x, y)

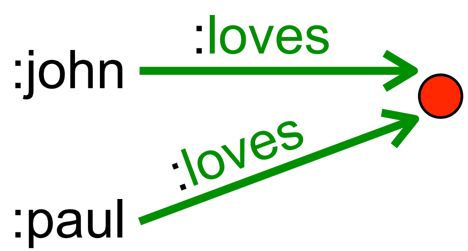
$\forall$x,y. :eats(x, y) $\rightarrow$:edible(y)

Queries

- Conjunctive queries (CQ) over the relation T as a query language
- This is elegantly just an RDF graph with possibly distinguished variables as node labels
- We want an algebra for CQ: the answer of a query should be a conjunctive query itself
- We assume the certain answer semantics
  - Query answering based on entailment
    - (unlike the elegant graph-theoretic approaches of [Gutiérrez, Mendelzon et al, 2004; Pérez, Arenas, Gutiérrez, 2006])
  - Note that this forms a weak representation system

Example

Data:



Query:

T(X, :loves, Y)

Answer:

X	Y
:john	_:b
:paul	_:b

Formalising queries

- An n-ary query is an RDF graph with n nodes labelled by distinct variable symbols
- An n-ary query can be written in FOL:

$$\exists \bar{b}.\ T(s_1, p_1, o_1) \wedge T(s_2, p_2, o_2) \wedge \cdots$$

- i.e., it is a conjunction of atoms, whose terms are URI or literals or bnodes or (free) variables, and $\bar{b}$ are the of bnodes in the query
- We write a query in short form as $\exists \bar{b}.Q(\bar{b}, \bar{x})$, where $\bar{x}$ are the variables in the query

Classical semantics of queries

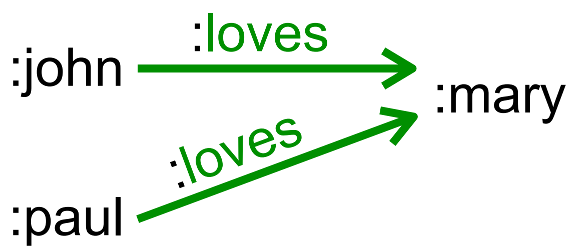
- The simple case: no bnodes in the answer set

$$\mathsf{ans}(Q, G) \equiv \{\bar{r} \in (\mathbb{U} \cup \mathbb{L})^n \mid G \models \exists \bar{b}.\, Q(\bar{b}, \bar{x}/\bar{r})\}$$

- This corresponds to the classical notion of query answer in databases (where G represent a single model) and knowledge representation (where G represents a possibly complex theory)

Example

Data:



G

Query:

T(X, :loves, Y)

Answer:

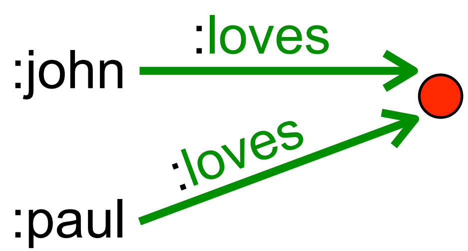
X	Y
:john	:mary
:paul	:mary

$G \models T(:john, :loves, :mary)$

$G \models T(:paul, :loves, :mary)$

Example & classical semantics

Data:



Query:

T(X, :loves, Y)

Answer:

X	Y

Empty!

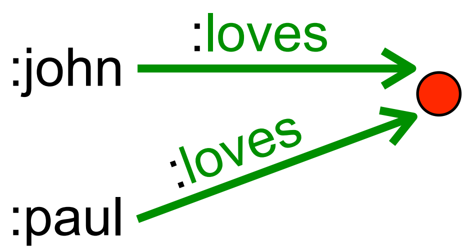
Semantics of queries

- bnodes are in the answer set
- the set of all answers is considered in a unique existential formula
- minimisation wrt entailment is required to avoid incorrect non minimal answers

$$
\begin{aligned}
&\mathsf{ans}(Q, G) \equiv \\
&\quad \mathsf{min}_{\models}\{\, R \subseteq (\mathbb{U} \cup \mathbb{L} \cup \mathbb{B})^n \mid \\
&\qquad\qquad G \models \exists \mathsf{bnodes}(R).\ \bigwedge_{\bar{r} \in R} \exists \bar{b}.\ Q(\bar{b}, \bar{x}/\bar{r})\}
\end{aligned}
$$

Example, with correct semantics

Data:



G

Query:

T(X, :loves, Y)

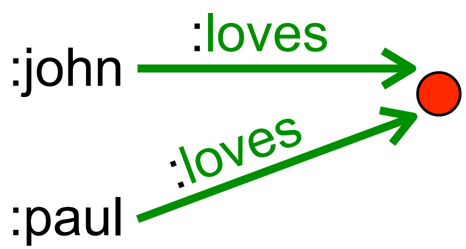
Answer:

X	Y
:john	_:b
:paul	_:b

$$G \models \exists \_{:}b.\ T(:john, :loves, \_{:}b) \wedge T(:paul, :loves, \_{:}b)$$

Not an answer

Data:



G

Query:

T(X, :loves, Y)

Not an answer, since not minimal wrt entailment:

X	Y
:john	_:b
:paul	_:c

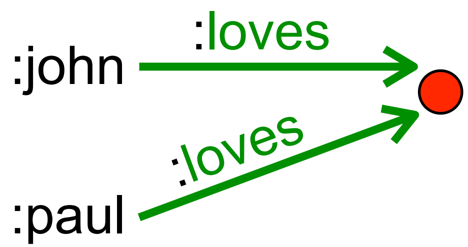
$$G \models \exists \_{:}b.\ T(:john, :loves, \_{:}b) \wedge T(:paul, :loves, \_{:}c)$$

Problem: non unique answer

- The certain answer of a query based on entailment as defined previously forms an incomplete database which is not uniquely representable, since it may contain arbitrary redundant information
- The number of logically equivalent representations of the certain answer depends on the number of available (distinguished) variable symbols

Example of redundant answer

Data:



Query:

$$X \xrightarrow{:loves} Y$$

An answer, even if redundant:

X	Y
:john	_:b
:paul	_:b
:paul	_:c

Uniqueness of answers

- In order to enforce the uniqueness of the answer, we may introduce a very general and apparently reasonable requirement for the query language: "the answer to the same query against two equivalent graphs should be the same"
- For example, this would be achieved by computing the core among the answer sets
- Theorem [_, Gutiérrez, 2006]: the evaluation of a query in a query language satisfying the above principle is necessarily NP-hard in data complexity, even for plain RDF.

How to enforce unique answers

Let S be an assignment of variables to RDF terms.
Given an entailment $\models_E$, a query Q, an RDF graph G, then Q E-matches graph G with answer S if:

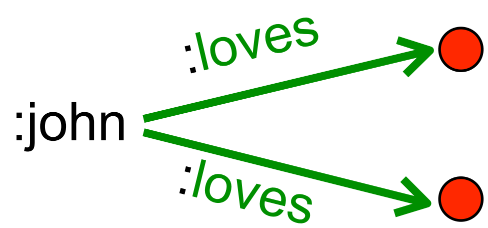
- there is a Q' isomorphic (modulo bnodes) to Q, such that Q' does not share any bnode with G
- S(Q') is a well-formed RDF graph for $\models_E$
- there is a injective assignment α of common bnodes between G and S(Q') to URIs not appearing in G or S(Q'), i.e., "skolem"
- $\boxed{\alpha(G) \models_E \alpha(S(Q'))}$
- the RDF terms in S all occur in G

Core Semantics

- the above semantic definition guarantees the uniqueness of answers, but not the requirement of getting the same answer from equivalent graphs.
- In the case of RDF entailment, the above definition is equivalent to compute the graph homomorphism, computable in polynomial time in data complexity.

Example

Data:



Query:

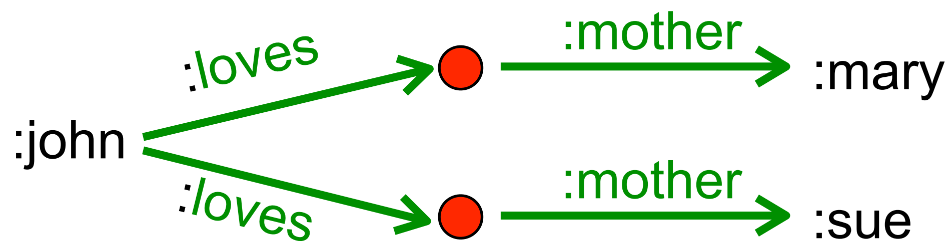
Unique answer:

X
_:b
_:c

X
_:b

Example

Data:



Query:

Unique answer:

X
_:b
_:c

SPARQL

- A W3C standardisation effort similar to the XQuery query language for XML data, within the Data Access Working Group (DAWG)
- As a query language, it is suitable for remote use by means of a remote access protocol.
- A basic graph pattern (BGP) query is an RDF graph with possibly variables as labels
- SPARQL defines on top of a BGP additional operators (AND, FILTER, UNION, OPTIONAL)

Additional results on SPARQL

- [_, Tessaris, 2006] SPARQL can be consistently used also for extensions of RDF (such as RDFS, OWL-DL and OWL-FULL)
- [Pérez, Arenas, Gutiérrez, 2006] Query answering in full SPARQL is PSPACE-complete in combined complexity, and in LOGSPACE in data complexity

Conclusions

- RDF(S) as a standard is a mess, but it has interesting theoretical properties and it poses challenging practical problems
- SPARQL as a query language is still at its infancy
- Many interesting open problems for the KR & DB research communities:
  - meta-data representation
  - incomplete information with constraints

Made on a Mac
LaTeX