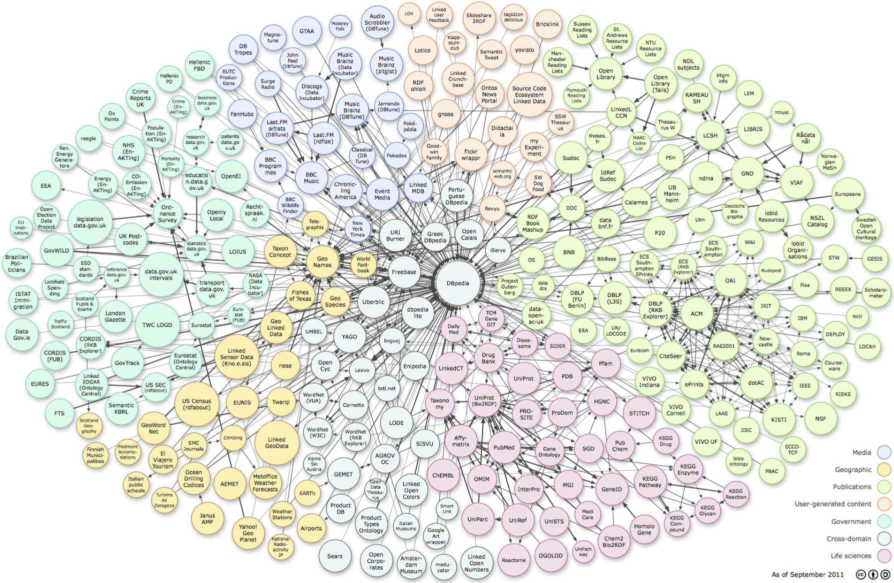
Datalog revisited for reasoning in Linked Data

Marie-Christine Rousset

Université de Grenoble-Alpes & Institut Universitaire de France

Joint work with M. Al Bakri, M. Atencia, J. David, F. Jouanot, S.Lalande, O.Palombi and F. Ulliana

Linked Data: the Semantic Web published in RDF



An RDF dataset : a set of triples (called an RDF graph)

```
@prefix rdf: ⟨ http://www.w3.org/1999/02/22-rdf-syntax-ns# ⟩ .
@prefix rdfs:   ⟨ http://www.w3.org/2000/01/rdf-schema#⟩ .
@prefix owl:   ⟨http://www.w3.org/2002/07/owl#⟩ .
@prefix wikipedia:   ⟨https://fr.wikipedia.org/wiki/⟩ .
```

```
wikipedia:Marie_Curie   hasName   "Marie Curie" .
wikipedia:Marie_Curie   rdf:type   Chemist .
wikipedia:Marie_Curie   hasWonPrize   NobelPrize.
wikipedia:Marie_Curie   bornIn   Europe.
wikipedia:Albert_Einstein   hasName   "Albert Einstein".
wikipedia:Albert_Einstein   rdf:type  Physicist.
wikipedia:Albert_Einstein   hasWonPrize   NobelPrize.
wikipedia:Albert_Einstein   birthPlace   Ulm .
Ulm   locatedIn   Germany.
Germany   partOf   Europe.
Chemist   rdfs:subClassOf  Scientist .
Physicist   rdfs:subClassOf  Scientist .
owl:ObjectPropertyChain (birthPlace Located partOf) rdfs:subPropertyOf bornIn.
```

An RDF dataset : a set of triples (called an RDF graph)

```
@prefix rdf: ⟨ http://www.w3.org/1999/02/22-rdf-syntax-ns# ⟩ .
@prefix rdfs:   ⟨ http://www.w3.org/2000/01/rdf-schema#⟩ .
@prefix owl:   ⟨http://www.w3.org/2002/07/owl#⟩ .
@prefix wikipedia:   ⟨https://fr.wikipedia.org/wiki/⟩ .
```

```
wikipedia:Marie_Curie   hasName   "Marie Curie" .
wikipedia:Marie_Curie   rdf:type   Chemist .
wikipedia:Marie_Curie   hasWonPrize   NobelPrize.
wikipedia:Marie_Curie   bornIn   Europe.
wikipedia:Albert_Einstein   hasName   "Albert Einstein".
wikipedia:Albert_Einstein   rdf:type  Physicist.
wikipedia:Albert_Einstein   hasWonPrize   NobelPrize.
wikipedia:Albert_Einstein   birthPlace   Ulm .
Ulm   locatedIn   Germany.
Germany   partOf   Europe.
Chemist   rdfs:subClassOf  Scientist .
Physicist   rdfs:subClassOf  Scientist .
owl:ObjectPropertyChain (birthPlace Located partOf) rdfs:subPropertyOf bornIn.
```

W3C SPARQL: the standard query language for RDF

SPARQL conjunctive queries

The core query language of SPARQL is: Basic Graph Pattern (BGP) queries, i.e. conjunctive or SELECT-PROJECT-JOIN queries.

Example of a SPARQL conjunctive query

Return the names of scientists born in Europe who received a Nobel Prize

- `SELECT ?n WHERE { ?p rdf:type Scientist . ?p hasWon NobelPrize . ?p bornIn Europe . ?p hasName ?n . }`
- `q(`**?n**`):- ?p rdf:type Scientist, ?p hasWon NobelPrize, ?p bornIn Europe, ?p hasName `**?n**`.`

A SPARQL query can search over the data and the schema

Return the properties having Europe as value

- `q(`**?prop**`):- ?s `**?prop**` Europe.`

SPARQL evaluation over an RDF graph (by example)

$\theta(?prop)$ is an answer for each substitution θ of the query variables by constants that maps every query conjunct to a fact.

RDF graph *G*

```
wikipedia:Marie_Curie   hasName   "Marie Curie" .
wikipedia:Marie_Curie   rdf:type   Chemist .
wikipedia:Marie_Curie   hasWonPrize   NobelPrize.
wikipedia:Marie_Curie   bornIn   Europe.
wikipedia:Albert_Einstein   hasName   "Albert Einstein".
wikipedia:Albert_Einstein   rdf:type  Physicist.
wikipedia:Albert_Einstein   hasWonPrize   NobelPrize.
wikipedia:Albert_Einstein   birthPlace   Ulm .
Ulm   locatedIn   Germany.         Germany   partOf  Europe.
Chemist rdfs:subClassOf Scientist .       Physicist rdfs:subClassOf Scientist .

owl:ObjectPropertyChain (birthPlace Located partOf) rdfs:subPropertyOf bornIn.
```

q(**?prop**):- ?s **?prop** Europe.

Result of SPARQL evaluation over G

q(G)= {**bornIn** , **partOf** }

SPARQL evaluation over an RDF graph (by example)

RDF graph G

```
wikipedia:Marie_Curie   hasName   "Marie Curie" .
wikipedia:Marie_Curie   rdf:type  Chemist.
wikipedia:Marie_Curie   hasWonPrize   NobelPrize.
wikipedia:Marie_Curie   bornIn   Europe.
wikipedia:Albert_Einstein   hasName   "Albert Einstein".
wikipedia:Albert_Einstein   rdf:type   Physicist.
wikipedia:Albert_Einstein   hasWonPrize   NobelPrize.
wikipedia:Albert_Einstein   birthPlace   Ulm .
Ulm   locatedIn  Germany.        Germany   partOf   Europe.
Chemist   rdfs:subClassOf   Scientist .   Physicist   rdfs:subClassOf   Scientist .
```

```
q(?n):?p rdf:type Scientist,?p hasWon NobelPrize,?p bornIn Europe,?p hasName ?n.
```

Result of standard SPARQL evaluation over G

q(G)= $\emptyset$

Query answering over RDF graphs requires reasoning

G^{∞}_{rdfs} : RDF facts + inferred facts by RDFS entailment

```
wikipedia:Marie_Curie  hasName  "Marie Curie" .
wikipedia:Marie_Curie  rdf:type Chemist.
wikipedia:Marie_Curie  hasWonPrize  NobelPrize.
wikipedia:Marie_Curie  bornIn  Europe.
wikipedia:Albert_Einstein  hasName  "Albert Einstein".
wikipedia:Albert_Einstein  rdf:type  Physicist.
wikipedia:Albert_Einstein  hasWonPrize  NobelPrize.
wikipedia:Albert_Einstein  birthPlace  Ulm .
Ulm  locatedIn Germany.        Germany  partOf  Europe.
Chemist  rdfs:subClassOf  Scientist .  Physicist  rdfs:subClassOf  Scientist .
wikipedia:Marie_Curie  rdf:type  Scientist.
wikipedia:Albert_Einstein  rdf:type  Scientist.

owl:ObjectPropertChain (birthPlace Located partOf)  rdfs:subPropertyOf  bornIn.
```

```
q(?n):?p rdf:type Scientist,?p hasWon NobelPrize,?p bornIn Europe,?p hasName ?n
```

Query answer

$q(G^{\infty}_{rdfs})= \{$"Marie Curie"$\}$

Complete query answering may require full reasoning

G^∞: RDF facts + inferred facts by RDFS entailment + owl rules

```
wikipedia:Marie_Curie   hasName   "Marie Curie" .
...
wikipedia:Albert_Einstein   hasName   "Albert Einstein".
wikipedia:Albert_Einstein   rdf:type   Physicist.
wikipedia:Albert_Einstein   hasWonPrize   NobelPrize.
wikipedia:Albert_Einstein   birthPlace   Ulm .
Ulm   locatedIn  Germany.          Germany   partOf   Europe.
Physicist   rdfs:subClassOf   Scientist .
wikipedia:Albert_Einstein   rdf:type   Scientist.
owl:ObjectPropertChain (birthPlace Located partOf)   rdfs:subPropertyOf   bornIn
wikipedia:Albert_Einstein   bornIn   Europe
```

```
q(?n):?p rdf:type Scientist,?p hasWon NobelPrize,?p bornIn Europe,?p hasName ?n
```

Query answer

q(G^∞)= {"Marie Curie", "Albert Einstein"}

Challenges raised by query answering in Linked Data

Scalability

- Linked Data cloud today: 9960 datasets, almost 150 billions triples (according to stats.lod2.eu)
- Almost no support for reasoning and thus very incomplete answers

⇒ Need for efficient query answering techniques involving some reasoning

Data quality

- Incomplete data (missing links, missing type information)
- Noisy data (some hub datasets like DBpedia or Yago are automatically generated)

⇒ Need for robust query answering and information discovery techniques

Remaining of the talk

A (partial) survey of recent works that have (partially) addressed some of these challenges using deductive RDF triplestores.

Deductive RDF triplestore: RDF dataset + a set of rules

Simple formalism for capturing several types of knowledge

- RDFS entailment

  (?i rdf:type ?s), (?s rdfs:subClassOf ?o) → (?i rdf:type ?o)

- (Most of) OWL constraints

  (?p birthPlace ?b), (?b Located ?c), (?c partOf ?d) → (?p bornIn ?d)

- Beyond FOL constraints

  (?p rdf:type owl:SymmetricProperty) , (?p rdfs:domain ?c) → (?p rdfs:range ?c)

- (Complex) mappings

  (?p1 ina:presenter ?v), (?v ina:title ?t) , (?p2 db:presenter ?t) → (?p1 owl:sameAs ?p2)

- Domain-specific rules (human embryo development)

  (?x mycf:absence_implies ?y), (?x mycf:depends_on ?z) → (?z mycf:absence_implies ?x)

A Datalog operational semantics to compute G^{∞}= SAT(D,R)

- Direct correspondence with a deductive DB using a single relation T

  (s p o) ↔ T(s p o)

Several instances of this generic framework

My Corporis Fabrica: an ontology-based suite of tools for combining complex anatomical models

- Rule-based interoperability between anatomical entities, human body functions and 3D graphic models
  ▷ with O. Palombi et al,
  - "My Corporis Fabrica: an ontology-based tool for reasoning and querying on complex anatomical models.", Journal of Biomedical Semantics 2014
  - "My Corporis Fabrica Embryo: An ontology-based 3D spatio-temporal modeling of human embryo development", Journal of Biomedical Semantics 2015

Module extraction from Semantic Web datasets

- Extraction of bounded-size RDF data modules enriched with rules
  ▷ with F. Ulliana, "Extracting Bounded-level Modules from Deductive RDF Triplestores.", AAAI 2015

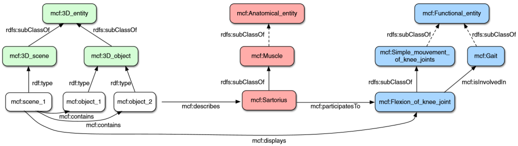
Rule-based Data Linkage

- Automatic discovery of same-As and DifferentFrom facts
  ▷ with M. Al Bakri, M. Atencia and Steffen Lalande, "Inferring same-as facts from Linked Data: an iterative import-by-query approach ", AAAI 2015, ECAI 2016

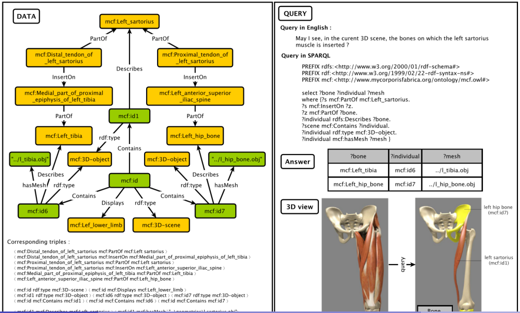
My Corporis Fabrica and MyCF Embryo

Rule-based interoperability between anatomical entities, human body functions and 3D graphic models

⇒ a declarative approach assisting interactive simulation and visualization



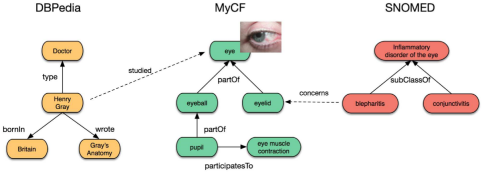
IF⟨ ?x rdf:type mcf:3D-object ⟩AND⟨ ?x mcf:Describes ?y ⟩
AND⟨ ?y rdfs:subClassOf mcf:Bone ⟩
THEN⟨ ?x mcf:hasColour 'yellow' ⟩ (R12)



Module extraction from Semantic Web datasets

Reuse of relevant extracts of big reference Web knowledge bases

⇒ a coherent and modular development of the Semantic Web



Existing works

- Well studied for Description Logics
  - not applicable to RDF datasets (e.g, DBpedia, Yago)
  - generally untractable, tractable approximations
  - may output large modules: the whole Tbox in the worst case
- Little work for RDF databases
  - RDF subgraph extraction, traversal views
  - reasoning not considered

Our contribution

A novel semantics of modules adapted to deductive RDF datasets

- Module signature $(p_1, \ldots, p_n)^k[a]$ involving properties, and individual and a bound k for property paths rooted in the specified individual.
- $\langle D_M, R_M \rangle$ is a bounded-level module of $\langle D, R \rangle$ iff D_M and R_M are conform to the signature, $\langle D, R \rangle \vdash \langle D_M, R_M \rangle$, and :

$$D, R^{\text{NonRec}} \vdash \pi_{(a,b)} \iff D_M, R_M \vdash \pi_{(a,b)} \quad (1)$$

$$D_M, R \vdash \pi_{(a,b)} \iff D_M, R_M \vdash \pi_{(a,b)} \quad (2)$$

for every path of atoms $\pi_{(a,b)}$ of bounded length in the signature.

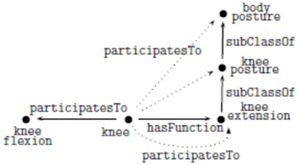
Non-recursive rules distinguished from recursive ones to avoid to waste k-parametricity.

Algorithms for module extraction

- Module data extraction expressed as a non-recursive Datalog program
- Construction of the R_M module rules by rule unfolding with a breadth-first strategy

Illustrative example

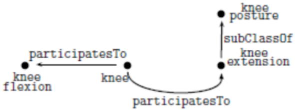
- Non recursive rules are needed to compute D_M
- Recursive rules must be delegated to R_M (if they are conform to the signature)

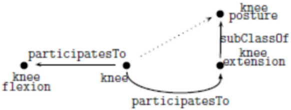


$(x, \mathtt{hasFunction}, y) \rightarrow (x, \mathtt{ParticipatesTo}, y)$

$(x, \mathtt{ParticipatesTo}, y), (y, \mathtt{subClassOf}, z)$
$\rightarrow (x, \mathtt{ParticipatesTo}, z)$

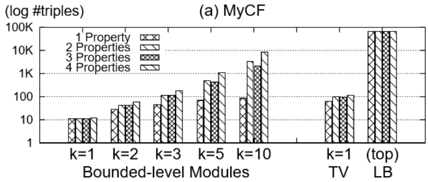
$[\mathtt{participatesTo}, \mathtt{subClassOf}]^{\mathbf{2}}(\mathtt{knee})$ with the *delegated* rule ↓

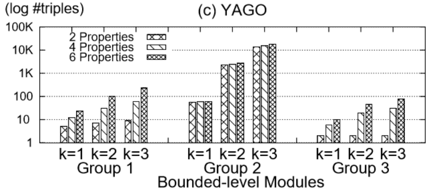




Module succinctness: experiments

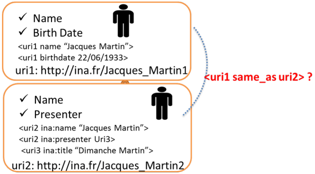
1. Comparison on MyCF with Traversal Views (applied to the saturared RDF dataset) and Locality-based extractor (applied to the corresponding DL ontology)
2. Impact of the properties in the signature: their number, their involvement and their interaction in (recursive) rules





Rule-based data linkage

Within a local dataset or accross different datasets



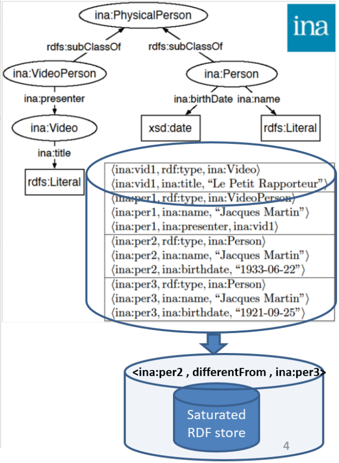
⇒ Our contributions:

- Import-by-Query, a backward-chaining algorithm combining local reasoning and external querying to bypass **local data incompleteness**
- ProbFR, a forward-chaining algorithm for reasoning with **uncertain data and rules**.

▷ joint work with M. Al Bakri, M. Atencia, J. David and Steffen Lalande (from INA)
▷ AAAI2015, ECAI2016

Reasoning with local data may not be enough

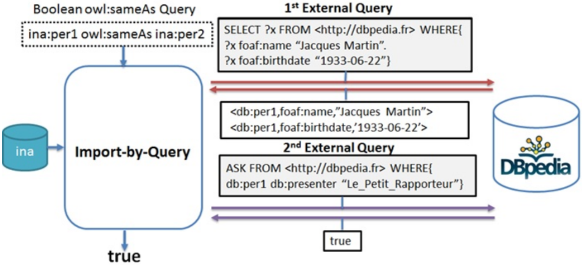
	IF	THEN
R1	?p1 name ?name ?p1 birthdate ?d ?p2 name ?name ?p2 birthdate ?d	?p1 same_as ?p2
	IF	**THEN**
R2	?p1 name ?name ?p1 ina:presenter ?v1, ?v1 title ?t ?p2 name ?name ?p2 db:presenter ?t	?p1 same_as ?p2
R3	?p1 birthdate ?d1 ?p2 birthdate ?d2 ?d1 <> ?d2	?p1 differentFrom?p2
R4	?x1 same_as ?x2 ?x2 same_as ?x3	?x1 same_as ?x3
R5	?x1 same_as ?x2 ?x2 differentFrom ?x3	?x1 differentFrom ?x3



BUT <ina:per1, same_as, ina:per2> ? STILL UNKNOWN

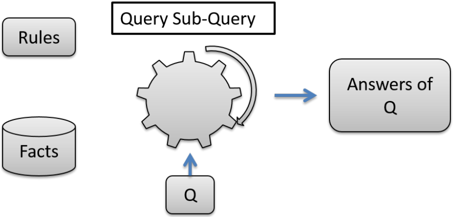
Import-by-Query

- Build **on demand** queries to some entry points of Linked Data
- The queries should be as instantiated as possible.
- Alternates steps of **query rewriting** and of **distant query evaluation**

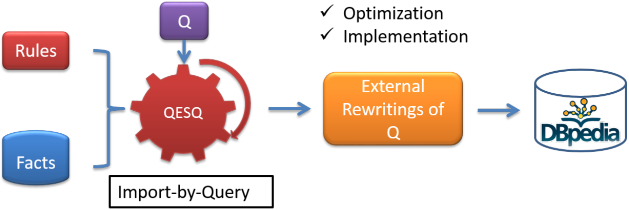


Query rewriting by adapting Query-SubQuery

A backward-chaining algorithm developed for answering queries in Datalog



- ✓ Correctness
- ✓ Completeness
- ✓ Optimization
- ✓ Implementation



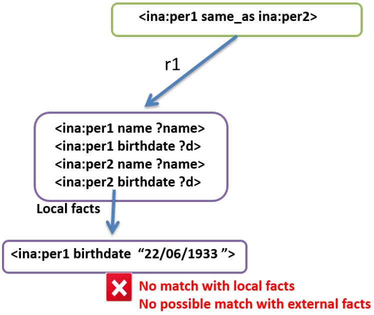
Query rewriting (by example)

<ina:per1 same_as ina:per2>

IF	THEN
r1: <?p1 name ?name>, <?p1 birthdate ?d>, <?p2 name ?name> <?p2 birthdate ?d>	<?p1 same_as ?p2>
r2: <?p1 name ?name>, <?p1 ina:presenter ?v1 >, <?v1 title ?t>, <?p2 name ?name> <?p2 db:presenter ?t>	<?p1 same_as ?p2>
r3: <?p3 same_as ?p2> , <?p3 same_as ?p1>	<?p1 same_as ?p2>

<ina:per1 name "Jacques Martin">
<ina:per1 presenter ina:v1 > <ina:v1 title "Le Petit Rapporteur"
<ina:per2 name "Jacques Martin">
<ina:per2 birthdate "22/06/1933 ">

Query rewriting (ctd)

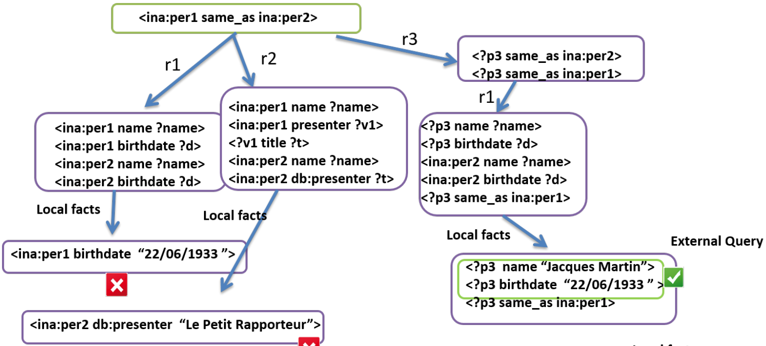


IF	THEN
r1: <?p1 name ?name>, <?p1 birthdate ?d>, <?p2 name ?name> <?p2 birthdate ?d>	<?p1 same_as ?p2>
r2: <?p1 name ?name>, <?p1 ina:presenter ?v1 >, <?v1 title ?t>, <?p2 name ?name> <?p2 db:presenter ?t>	<?p1 same_as ?p2>
r3: <?p3 same_as ?p2> , <?p3 same_as ?p1>	<?p1 same_as ?p2>

Local facts

<ina:per1 name "Jacques Martin">
<ina:per1 presenter ina:v1 > <ina:v1 title "Le Petit Rapporteur">
<ina:per2 name "Jacques Martin">
<ina:per2 birthdate "22/06/1933 ">

Query rewriting (ctd)



IF	THEN
r1: <?p1 name ?name>, <?p1 birthdate ?d>, <?p2 name ?name> <?p2 birthdate ?d>	<?p1 same_as ?p2>
r2: <?p1 name ?name>, <?p1 ina:presenter ?v1 >, <?v1 title ?t>, <?p2 name ?name> <?p2 db:presenter ?t>	<?p1 same_as ?p2>
r3: <?p3 same_as ?p2> , <?p3 same_as ?p1>	<?p1 same_as ?p2>

9/23/2015

Local facts

<ina:per1 name "Jacques Martin">
<ina:per1 presenter ina:v1 > <ina:v1 title "Le Petit Rapporteur"
<ina:per2 name "Jacques Martin">
<ina:per2 birthdate "22/06/1933 ">

17

Experiments

Conducted on a deductive RDF triplestore built with INA

- one million RDF facts (provided by INA) : RDF export and extraction of metadata from the INA catalog
- 35 rules (built with the help of INA experts)

	IF	THEN
r7	⟨?x1,foaf:name ,?name1⟩ , ⟨?x2, skos:altLabel ,?name2⟩, Similar(?name1, ?name2, 0.99)	⟨?x1, ina:sameNameDBp ,?x2⟩
r8	⟨?x1,foaf:name ,?name1⟩ , ⟨?x2, skos:prefLabel ,?name2⟩, Similar(?name1, ?name2, 0.99)	⟨?x1, ina:sameNameDBp ,?x2⟩
r9	⟨?x1,rdfs:label ,?name1⟩ , ⟨?x2, skos:prefLabel ,?name2⟩, Similar(?name1, ?name2, 0.99)	⟨?x1, ina:sameNameDBp ,?x2⟩
r10	⟨?x1,rdfs:label ,?name1⟩ , ⟨?x2, skos:altLabel ,?name2⟩, Similar(?name1, ?name2, 0.99)	⟨?x1, ina:sameNameDBp ,?x2⟩
r11	⟨?x1,prop-fr:nom ,?name1⟩ , ⟨?x2, skos:prefLabel ,?name2⟩, Similar(?name1, ?name2, 0.99)	⟨?x1, ina:sameNameDBp ,?x2⟩
r12	⟨?x1,prop-fr:nom ,?name1⟩ , ⟨?x2, skos:altLabel ,?name2⟩, Similar(?name1, ?name2, 0.99)	⟨?x1, ina:sameNameDBp ,?x2⟩

	IF	THEN
r13	⟨?x1, ina:sameNameDBp ,?x2⟩ , ⟨?x1, dbpedia:birthYear ,?Y1⟩ , ⟨?x2, ina:birthYear ,?Y1⟩ ⟨?x1, dbpedia:deathYear ,?Y2⟩ , ⟨?x2, ina:deathYear ,?Y2⟩	⟨?x1, ina:sameAs ,?x2⟩
r14	⟨?x1, ina:sameNameDBp ,?x2⟩ , ⟨?x1, dbpedia:birthYear ,?Y1⟩ , ⟨?x2, ina:birthYear ,?Y2⟩ notEqual(Y1, Y2)	⟨?x1, ina:differentFrom ,?x2⟩
r15	⟨?x1, ina:sameNameDBp ,?x2⟩ , ⟨?x1, dbpedia:deathYear ,?Y1⟩ , ⟨?x2, ina:deathYear ,?Y2⟩ notEqual(Y1, Y2)	⟨?x1, ina:differentFrom ,?x2⟩

Results

- External information in Linked Data is useful for disambiguation

	sameAs	DifferentFrom
35 rules, ina	2	10
35 rules, ina, Dbpedia	4884	9764

- Full reasoning on (recursive) rules is useful
  - Comparison between Silk and a forward reasoner applied to our rules
    Silk only discovered 3% of the sameAs links discovered by our approach
  - 100% precision by construction (if the rules and the facts are correct)
    - ★ checked in practice on a sample of 500 links
- Import-by-Query brings a drastic reduction of the imported facts

	Import By Query	Forward Reasoner
Number of Imported Facts for a sample of 500 Boolean queries	6,417 facts (13 per Boolean query)	500,000 facts

- Import-by-Query requires 3 iterations of rewritings on average

Time to answer a boolean query *after* fact propagation	7 seconds
Time to answer a boolean query *without* fact propagation	186 seconds
Time to propagate facts (done once for all queries)	191 seconds
Gain of doing fact propagation beforehand for answering the 500 reference queries using import-by-query	96%

ProbFR: Probabilistic Forward Reasoner

Unifying modeling of any kind of uncertainty as probabilities

- noisy data (e.g., due to automatic data extraction from WikiPedia)
- pseudo-keys, constraints with exceptions
- weighted mappings between vocabularies across datasets

Operational semantics of Probabilistic Datalog

- extension of probabilistic databases
- each input fact and rule is associated with a symbolic event
- an event expression is computed for each inferred fact, that encapsulates its provenance
- the probabilities are computed from the event expressions

ProbFR implemented on top of JENA RETE

Linkage between MusicBrainz and DBpedia using ProbFR

- MusicBrainz: 112 millions triples (12 GB)
- DBpedia (extract on songs, bands and persons): 73 millions triples
- 20 certain rules, 36 uncertain rules (probabilities from 0.3 to 0.9)
  - Runtime performance: less than 2 hours in total (including the loading time and the use of SOLR to compute some built-in predicates)
  - Impact of using uncertain information:
    - ★ Precision and recall based on certain rules only

	Precision	**Recall**
Person	100%	8%
Band	100%	12%

    - ★ Precision and recall based on all the rules

	Precision	**Recall**
Person	100%	80%
Band	94%	84%
Song	94%	74%

    - ★ Precision and recall after filtering the inferred facts with a probability over a threshold

	Precision	**Recall**
$\text{Band}_{\geqslant 0.9}$	100%	80%
$\text{Song}_{\geqslant 0.9}$	100%	44%

Conclusion

Semantic Web standards, data and applications are there

Linked Data is flourishing due to the simplicity and flexibility of the RDF data model.

However many challenges remain

- Efficient Semantic Web data and knowledge management is still challenging.
- Novel problems arise to handle at large scale the incomplete and uncertain nature of Web data

Our message:

(Extensions of) Datalog on top of RDF datasets is an interesting angle of attack for many of these challenges