Symbolic methods for RDF data interlinking

Manuel Atencia Jérôme David Jérôme Euzenat
Marie-Christine Rousset

&

Laboratoire d'Informatique de Grenoble
Montbonnot, France
{Manuel.Atencia,Jerome.David,Jerome.Euzenat}@inria.fr
http://moex.inria.fr
{Marie-Christine.Rousset}@imag.fr
http://slide.liglab.fr

January 23, 2017

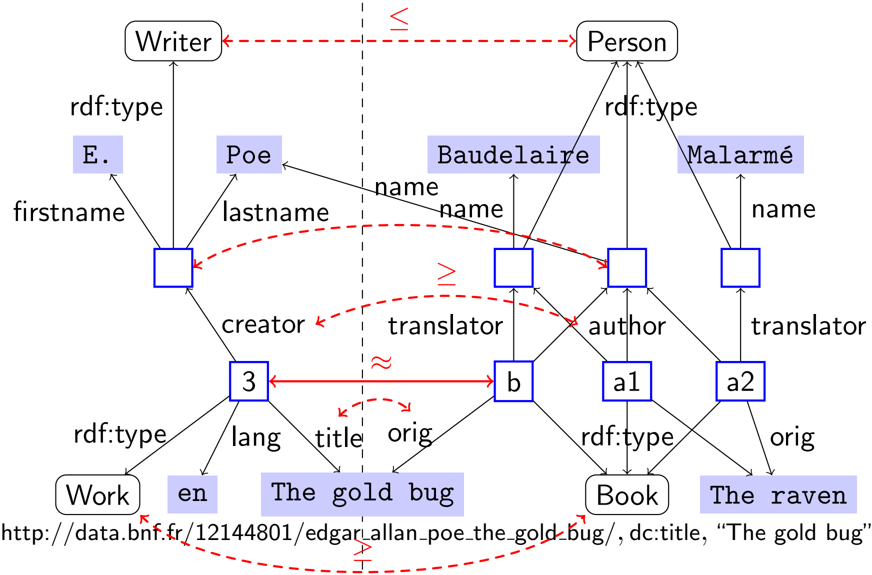
Outline

What is linked data?

- Structured data expressed with semantic web technologies (RDF, OWL, etc.)
- Published on the web (deferenceable URIs, online SPARQL endpoints), and
- **Linked**: same resources in different datasets have to be identified and related through `owl:sameAs` links

Many examples available: dbpedia, xlore, FAO data, Genebank, Open street map, many national libraries, etc.

The problem: RDF data interlinking



⟨http://data.bnf.fr/12144801/edgar_allan_poe_the_gold_bug/, dc:title, "The gold bug"

Data interlinking

Data interlinking is the task of finding the same entities within different datasets (RDF graphs).

There are two main approaches to data interlinking:

- similarity-based: resources are compared through a similarity measure and if they are similar enough, they are the same.
  - c.f. Vassilis presentation about entity reconciliation
- rule/key-based (symbolic): logical rules expressing sufficient conditions for two resources to be the same are used to deduce same entities

Both approaches can be (and have to be?) combined

Data interlinking process

Data interlinking process can be decomposed into two phases :

1. Specify how links will be generated
   - It consists in defining similarity-based linkage rules, link keys, logical rules, etc.
   - It can be done manually or (semi-)automatically
2. Generate links using specifications
   - single pass: all rules are applied in one single pass (via SPARQL query or link generation engine (SILK/Limes)
   - saturation/inference: all rules applied until no new links are generated (using some inference engine)

Symbolic approaches for (RDF) data interlinking

Why to use symbolic approaches ?

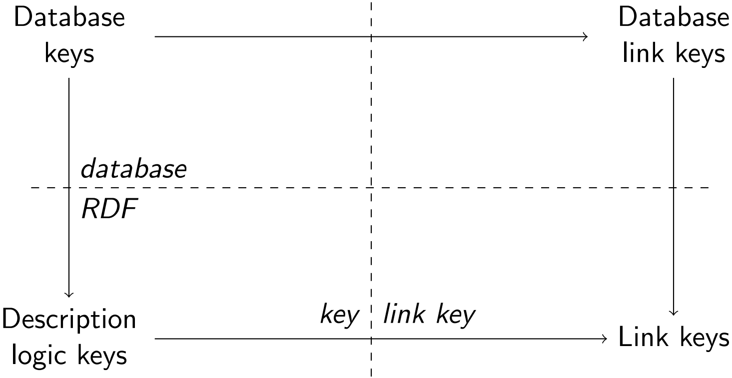
- They can be expressed as ontological constraints / rules that can be used for inferring new links
  - useful when data evolves continuously
  - can help to reduce redundancy
- They are meaningful for the user/domain expert
- They usually produce high quality links
  - precision is usually very high
  - but they are more sensitive to the quality of data (low recall)

Symbolic approaches for (RDF) data interlinking

Which symbolic approaches will be covered?

- Keys + ontology alignments [Atencia, David, Scharffe - EKAW 12]
- Link keys: there are generalization of keys + alignment [Atencia, David, Euzenat - ECAI 14]
- Rules [Al-Bakri, Atencia, David, Lalande, Rousset - ECAI 16]

(RDF) Data interlinking



Database keys

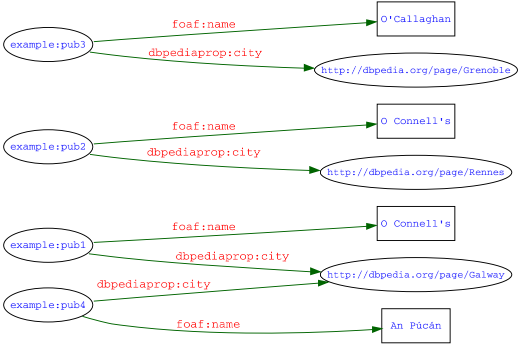
- A set of attributes which uniquely identifies elements of a relation
- e.g., Book: isbn, People: firstname, lastname, birthplace, birthdate
- usually given and used to check integrity

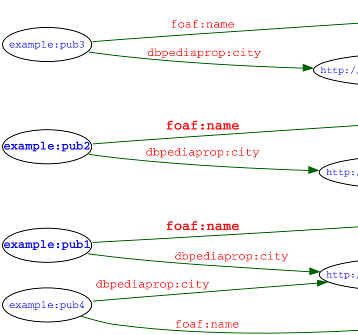
They may be used for identifying same entities across two databases.
But they require alignments.

RDF keys

- A key for a class C is a set of predicates P which allow to identify all instances of C
- A key P is minimal is there is no subset P' of P which is also a key

Example: What are the keys on this graph?





in-key and eq-key

We make the distinction between two kinds of keys

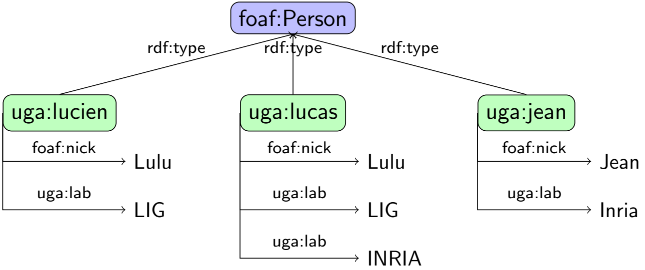
- An *in-key* ($\{p_1, \ldots, p_k\} \ \mathrm{key_{in}} \ C$) is satisfied by an interpretation $\mathcal{I}$ iff, for any $\delta, \delta' \in C^{\mathcal{I}}$,

$$p_1^{\mathcal{I}}(\delta) \cap p_1^{\mathcal{I}}(\delta') \neq \emptyset, \ldots, p_k^{\mathcal{I}}(\delta) \cap p_k^{\mathcal{I}}(\delta') \neq \emptyset \text{ implies } \delta = \delta'.$$

- An *eq-key* ($\{p_1, \ldots, p_k\} \ \mathrm{key_{eq}} \ C$) is satisfied by an interpretation $\mathcal{I}$ iff, for any $\delta, \delta' \in C^{\mathcal{I}}$,

$$p_1^{\mathcal{I}}(\delta) = p_1^{\mathcal{I}}(\delta') \neq \emptyset, \ldots, p_k^{\mathcal{I}}(\delta) = p_k^{\mathcal{I}}(\delta') \neq \emptyset \text{ implies } \delta = \delta'.$$

Example of an eq-key



{`foaf:nick, uga:lab`} is not an *in-key* because `uga:lucas` and `uga:lucien` share the pair of values `(Lulu, LIG)` for this mix of properties.
But {`foaf:name, uga:lab`} is an *in-key* because no instances share the same set of value for this mix of properties.

Key and pseudo-keys in RDF

Key axioms are not often given but they can be induced from data.

The problems are:

- Data quality
- The number of candidates is exponential ($2^{\#\text{ of prop. for C}}$)

Pseudo keys

To deal with imperfect data we need a relaxed notion of a key, named pseudo-key

- Which allows few exceptions
- Which is not defined on all instances

To that extent, a pseudo key is associated with two values:

- The **support** of a pseudo-key is the proportion of individuals having all the predicates of the key instanciated
- The **discriminability** of a pseudo-key is proportion of instances satisfying the key definition
- A pseudo key with both support and discriminability equals to 1 is a key.

Key discovery search space

The number of candidates is exponential ($2^{\#\text{ of prop. for C}}$)

Solution:
To take advantage of functional property axioms (Amstrong's axioms) in order to prune search space.

Approach used in relational databases:

- Breadth-first: Tane and variants
- Depth-first: Gordian and variants

We choose breath first strategy based on Tane

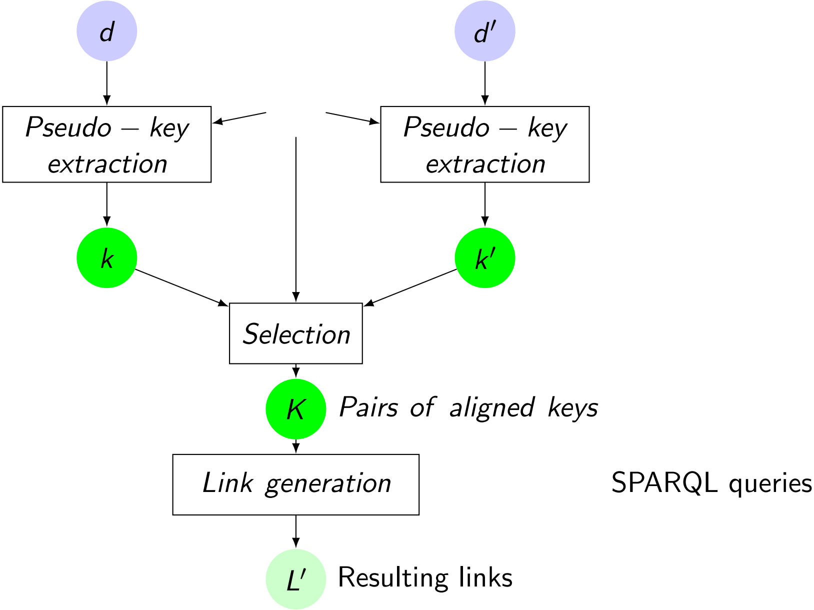
An algorithm for discovering pseudo keys

Difference with state of the art algorithms:

- RDF allows multivalued properties
- We want to use support threshold for pruning
- Transitivity of functional dependencies is not valid on RDF data (due to "missing" values)

A Java implementation available at
`https://gforge.inria.fr/projects/melinda`

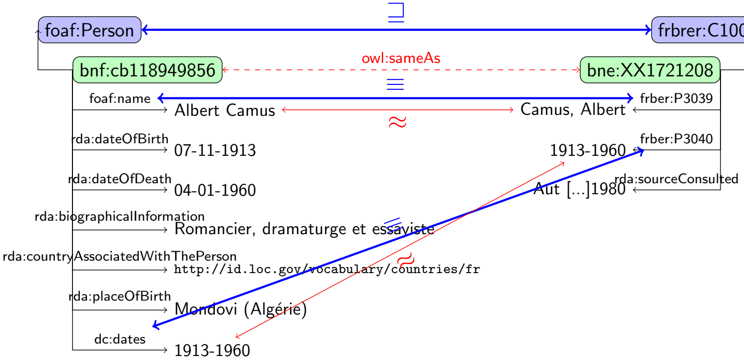
Key-based data interlinking process



Example of interlinking with keys and alignments

Are the resources bnf:cb118949856 and bne:XX1721208 the same?

- if BNF ontology states foaf:Person owl:hasKey {foaf:name, dc:dates}
- and we have the following alignment



Key-based interlinking methods

Keys allow for identifying entities: if they are aligned, this can be used for linking.

- Advantages
  - they are logically grounded
  - they allow to minimize the number of properties to compare (if we use minimal keys)
- Drawbacks
  - Require alignment between properties and classes
  - Very few key axioms are available, and they are not necessarily useful for interlinking

We overcome these drawbacks by introducing link keys

Link key

- Link keys are rules allowing to infer links;
- They are a generalisation of pairs of keys related by alignment;
- They are defined across a pair of (not disjoint) classes.

Link key definition

A *link key*

$$\langle \{ \langle p_1, q_1 \rangle, \ldots, \langle p_k, q_k \rangle \} \text{ linkkey } \langle c, d \rangle \rangle$$

holds iff
For all pairs of instances a and b belonging respectively to classes c and d of ontologies $\mathcal{O}$ and $\mathcal{O}'$,

if a and b share at least one value (object) for each pairs of properties p_i and q_i respectively,

then they are the same ($\langle a, \texttt{owl:sameAs}, b \rangle$).

Example:

$$\langle \{ \langle \text{foaf:name}, \text{frbr:P3039} \rangle, \langle \text{dc:dates}, \text{frbr:P3040} \rangle \} \text{ linkkey } \langle \text{foaf:Person}, \text{frbr:C1005} \rangle \rangle$$

Link key (the full definition)

A *link key*

$$\langle\{\langle p_1, q_1\rangle, \ldots, \langle p_k, q_k\rangle\}\ \{\langle p'_1, q'_1\rangle, \ldots, \langle p'_l, q'_l\rangle\}\ \text{linkkey}\ \langle c, d\rangle\rangle$$

holds iff
$\forall a; \mathcal{O} \models c(a)$, $\forall b; \mathcal{O}' \models d(b)$,

$$\left.\begin{array}{rl} \text{if} & \forall i \in 1, \ldots, k, p_i(a) \cap q_i(b) \neq \varnothing \\ \text{and} & \forall i \in 1, \ldots, l, p'_i(a) = q'_i(b) \neq \varnothing \end{array}\right\} \text{then } \langle a, \texttt{owl:sameAs}, b\rangle \text{ holds}$$

$$p(s) = \{o | O \models \langle s, p, o\rangle\}$$

Link key extraction

- Link keys are sufficient to generate links.

Problem: How to induce such link keys from data?

The number of set of pairs of properties is exponential

Our approach:

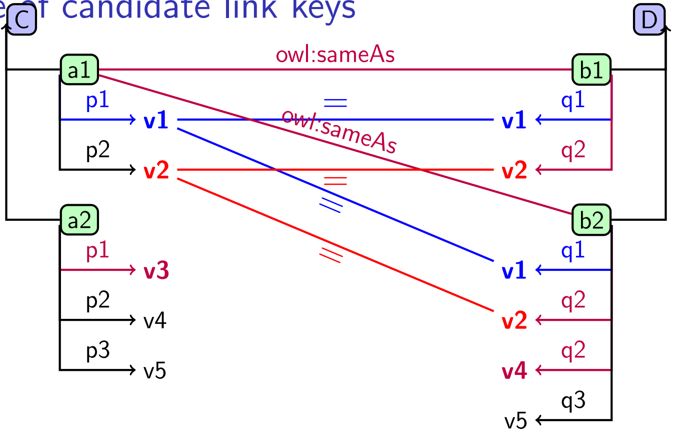
- discover only candidate link keys.
- evaluate them in order to select only the “good” ones

Candidate link key

A candidate link key is a set of property pairs $\{\langle p_1, q_1\rangle, \ldots, \langle p_k, q_k\rangle\}$ that

1. would generate at least one link if used as a link key
2. is maximal for at least one link, or is the intersection of several candidate link keys

Example of candidate link keys



- $\{\langle p_1, q_2\rangle\}$ a candidate? NO, it does not generate any link
- $\{\langle p_1, q_1\rangle\}$ a candidate? NO
  - it could generate links: $\langle a_1, b_1\rangle$ and $\langle a_1, b_2\rangle$
  - but it is not maximal: each link also shares $\{\langle p_2, q_2\rangle\}$
- Then $\{\langle p_1, q_1\rangle, \langle p_2, q_2\rangle\}$ is a candidate linkkey

Algorithm for candidate link key extraction

1. For each dataset, index each subject-property pair according to its values

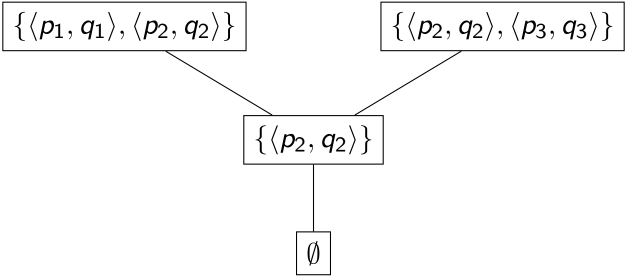
indexDataset(D)	indexDataset(D')
$v_1 : \{\langle a_1, p_1\rangle\}$	$v_1 : \{\langle b_1, q_1\rangle, \langle b_2, q_1\rangle\}$
$v_2 : \{\langle a_1, p_2\rangle\}$	$v_2 : \{\langle b_1, q_2\rangle, \langle b_2, q_2\rangle\}$
$v_3 : \{\langle a_2, p_1\rangle\}$	
$v_4 : \{\langle a_2, p_2\rangle\}$	$v_4 : \{\langle b_2, q_2\rangle\}$
$v_5 : \{\langle a_2, p_3\rangle\}$	$v_5 : \{\langle b_2, q_3\rangle\}$

2. Iterate on index and compute for each pair of subjects the maximal set of pair of property on which they agree

Candidate links		Candidate link keys
$\langle a_1, b_1\rangle$	$\rightarrow$	$\{\langle p_1, q_1\rangle, \langle p_2, q_2\rangle\}$
$\langle a_1, b_2\rangle$	$\rightarrow$	$\{\langle p_1, q_1\rangle, \langle p_2, q_2\rangle\}$
$\langle a_2, b_1\rangle$	$\rightarrow$	$\emptyset$
$\langle a_2, b_2\rangle$	$\rightarrow$	$\{\langle p_2, q_2\rangle, \langle p_3, q_3\rangle\}$

3. Close by intersection

Resulting candidate link keys



Results

- We have an algorithm for extracting them;
- But which candidate is the best?

Supervised selection measures

If a sample of reference links is available:

- Positive examples (L^+) : a set of owl:sameAs links
- Negative examples (L^-) : a set of owl:differentFrom links

Idea: Approximate precision and recall on that sample

Definition (Relative precision and recall)

$$\widehat{\text{precision}}(K, L^+, L^-) = \frac{|L^+ \cap L_{D,D'}(K)|}{|(L^+ \cup L^-) \cap L_{D,D'}(K)|}$$

$$\widehat{\text{recall}}(K, L^+) = \frac{|L^+ \cap L_{D,D'}(K)|}{|L^+|}$$

Unsupervised selection measures

When no reference link is available.

Idea: measuring how close the extracted links would be from one-to-one and total.

Definition (Discriminability)

$$\mathrm{disc}(K, D, D') = \frac{\min(|\{a : \langle a, b\rangle \in L_{D,D'}(K)\}|, |\{b : \langle a, b\rangle \in L_{D,D'}(K)\}|)}{|L_{D,D'}(K)|}$$

Definition (Coverage)

$$\mathrm{cov}(K, D, D') = \frac{|\{a : \langle a, b\rangle \in L_{D,D'}(K)\} \cup \{b : \langle a, b\rangle \in L_{D,D'}(K)\}|}{|\{a : c(a) \in D\} \cup \{b : d(b) \in D'\}|}$$

Data sets

Finding links between French municipalities described in two different public datasets:

- Insee dataset: 36700 instances of Communes;
- Geonames dataset: 36552 instances of French Features of NUTS level 4.

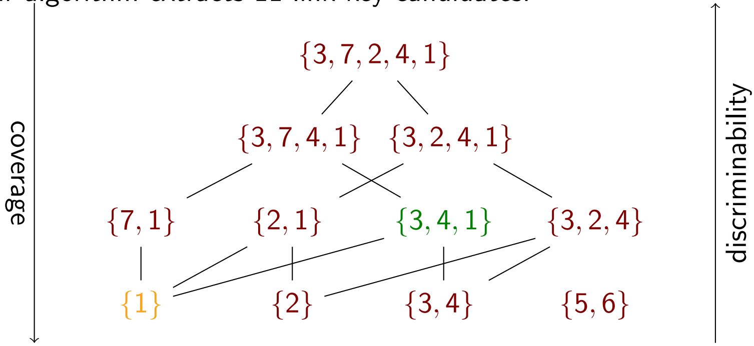
The reference link set is composed of:

- Positive links: 36552 owl:sameAs statements;
- owl:differentFrom links derived from owl:sameAs links (closed world assumption).

$2^{16\times 4} = 1.9 \times 10^{19}$ possible link keys

Evaluation

Our algorithm extracts 11 link key candidates:



5 = ⟨codeINSEE, population⟩

6 = ⟨codeCommune, population⟩

3 = ⟨subdivisionDe, parentFeature⟩

4 = ⟨subdivisionDe, parentADM3⟩

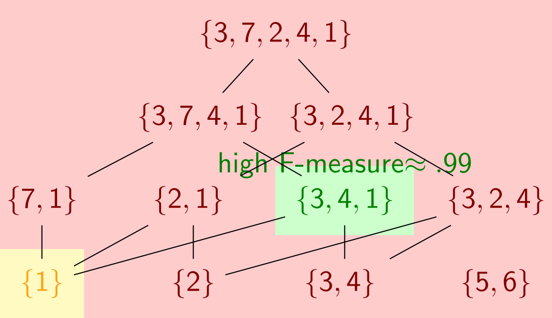
1 = ⟨nom, name⟩

2 = ⟨nom, alternateName⟩

7 = ⟨nom, officialName⟩

Evaluation

Harmonic means of discriminability and coverage and F-measure:



good F-measure≈ 0.89 bad F-measure≈ 0

h-mean(disc.,cov)≈ .99 h-mean(disc.,cov)≈ .89 h-mean(disc.,cov) ≈ 0

$5 = \langle \text{codeINSEE}, \text{population} \rangle$

$6 = \langle \text{codeCommune}, \text{population} \rangle$

$3 = \langle \text{subdivisionDe}, \text{parentFeature} \rangle$

$4 = \langle \text{subdivisionDe}, \text{parentADM3} \rangle$

$1 = \langle \text{nom}, \text{name} \rangle$

$2 = \langle \text{nom}, \text{alternateName} \rangle$

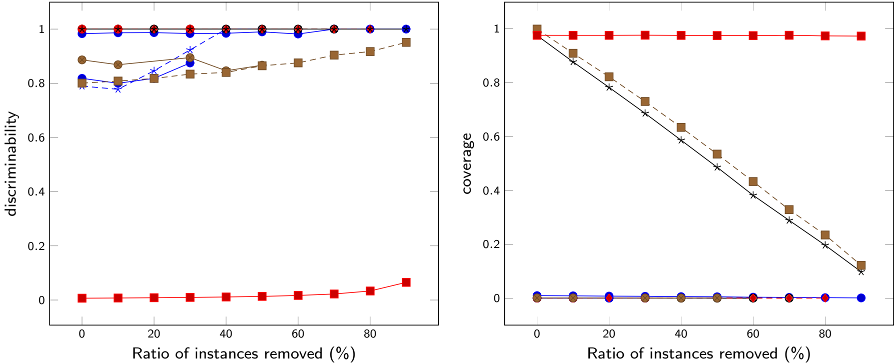
$7 = \langle \text{nom}, \text{officialName} \rangle$

Robustness of unsupervised measures

Are discriminability and coverage measures robust to alterations?

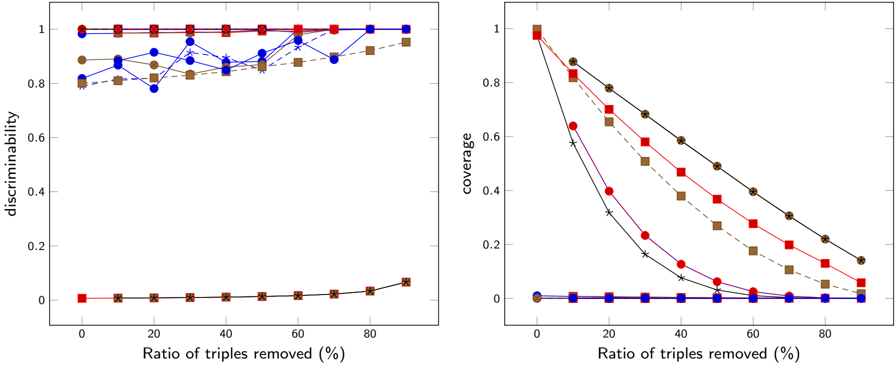
- instance removal: instances are randomly removed by suppressing all triples involving them;
- triples removal: we randomly suppress some triples;
- values scrambling: we randomly scramble the object of some triples.

Robustness to instance removal



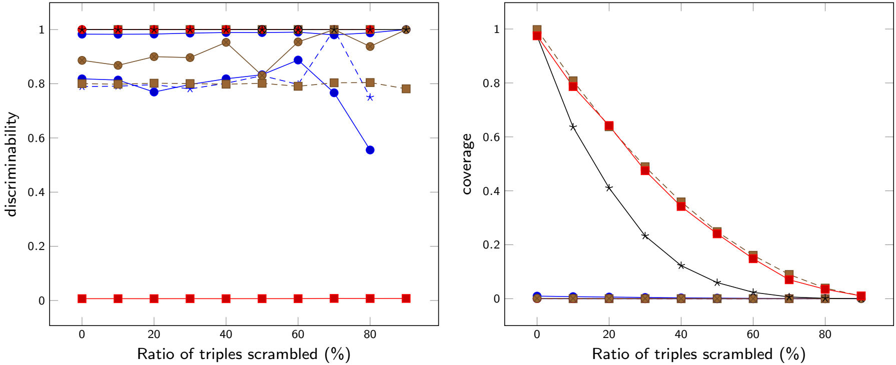
- Discriminability is slowly increasing
- Coverage is
  - linearly decreasing for the two good candidates
  - stable for the bad ones

Robustness to triple removal



- Triple removal introduces new candidates
- Discriminability is slowly increasing (idem than instances rem.)
- Coverage is more sensitive to triples removal than instances removal
  - the two good candidates (and derivatives) decreases faster than linearly
  - the other decrease linearly.

Robustness to triple scrambling



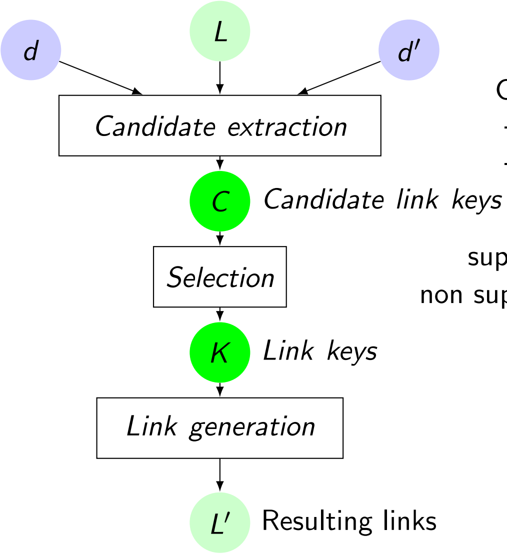
- Triple removal introduces even more new candidates
- Discriminability is almost stable
- Coverage is more sensitive to triples removal than instances removal
  - both good candidates and bad ones decrease faster than linearly

Conclusion on link key extraction

Experimental results show:

- both supervised and unsupervised measures are good estimators of precision and recall
- discriminability is robust to alterations
- coverage is (sub) linearly decreasing when alterations increase but candidate ranking is preserved
- we found 25 missing links in the reference (which were sent to INSEE).

Data interlinking process



Candidate link keys:
- generate links, and
- are maximal

supervised: precision/recall

non supervised: discriminability/covera

SPARQL queries

Contributions

- Definition of link keys;
- Algorithm for extracting all link key candidates [ECAI 2014];
- Measures to assess the quality of extracted link key candidates:
  - supervised measures: sample reference links are available;
  - unsupervised measures: any reference link at all;
- Further experiments showed robustness to perturbation;
- First characterisation as formal concept analysis [FCA4AI 2014];
- Supported by our Alignment API (SPARQL can be generated from link keys).

Future work

- Extend this link key extraction technique to:
  - Concepts referring to other concepts;
  - Interdependent concepts (recursion);
- Reasoning with ontologies+link keys;
- Studying disjunctive link keys;
- Extend the FCA characterisation to Relational concept analysis.

Rule-based data interlinking problem

We have a set of interlinking rules:

- given by an expert
- extracted with linkey extraction algorithm

How to generate links by taking into account:

- uncertainty in rules: they are maybe not perfect
- uncertainty in data

Uncertainty-sensitive rule-based data interlinking

Idea

Combine (datalog) rules and probabilistic weights to perform data-interlinking

Contributions

- A declarative framework based on **probabilistic Datalog** to model uncertain facts and rules
- **ProbFR**: an inference algorithm that computes the probability of inferred facts as well as the **uncertainty provenance** of this computation
- A series of experiments over real-world large RDF datasets showing the benefits and the scalability of our approach

Probabilistic Datalog[1]

A simple extension of Datalog in which rules and facts are associated with **symbolic probabilistic events**

Logical inference and probability computation are separated

1. **Step1 (ProbFR)**: compute **the provenance** of each inferred fact : the **boolean combination** of all the events associated with the facts and rules involved in its derivation.
   - exponential in the worst-case.
   - by-passed by a practical bound on the number of conjuncts in the provenances and a priority given to the most probable rules and facts
2. **Step2**: computation of the probabilities of the inferred facts from their provenances in which each event of input facts and rules is assigned a **probabilistic weight**
   - based on independence and disjointness assumptions to make it feasible

(1) N. Fuhr, Probabilistic models in information retrieval, The Computer Journal,

Illustrative Example

Rules: uncertain rules are in red, certain rules are in blue

$r_1 : (?x\ sameName\ ?y) \Rightarrow (?x\ sameAs\ ?y)$

$r_2 : (?x\ sameName\ ?y), (?x\ sameBirthDate\ ?y) \Rightarrow (?x\ sameAs\ ?y)$

$r_3 : (?x\ marriedTo\ ?z), (?y\ marriedTo\ ?z) \Rightarrow (?x\ sameAs\ ?y)$

$r_4 : (?x\ sameAs\ ?z), (?z\ sameAs\ ?y) \Rightarrow (?x\ sameAs\ ?y)$

Facts: uncertain facts are in red, certain facts are in blue

$f_1 : (i_1\ sameName\ i_2)$ $f_2 : (i_1\ sameBirthDate\ i_2)$ $f_3 : (i_2\ marriedTo\ i_3)$

$f_4 : (i_4\ marriedTo\ i_3)$ $f_5 : (i_2\ sameName\ i_4)$

Provenance of inferred facts

Inferred facts	Provenance	Uncertainty Provenance
$(i_2\ sameAs\ i_4)$	$(e(r_1) \wedge e(f_5)) \vee (e(r_3) \wedge e(f_3) \wedge e(f_4))$	$\top$
$(i_1\ sameAs\ i_2)$	$(e(r_1) \wedge e(f_1)) \vee (e(r_2) \wedge e(f_1) \wedge e(f_2))$	$e(r_2) \wedge e(f_1)$
$(i_1\ sameAs\ i_4)$	$e(r_4) \wedge Prov((i_1\ sameAs\ i_2))$ $\wedge Prov((i_2\ sameAs\ i_4))$	$e(r_2) \wedge e(f_1)$

Illustrative Example (cont.)

Rules: uncertain rules are in red, certain rules are in blue

$r_1 : (?x\ sameName\ ?y) \Rightarrow (?x\ sameAs\ ?y)$

$r_2 : (?x\ sameName\ ?y), (?x\ sameBirthDate\ ?y) \Rightarrow (?x\ sameAs\ ?y)$

$r_3 : (?x\ marriedTo\ ?z), (?y\ marriedTo\ ?z) \Rightarrow (?x\ sameAs\ ?y)$

$r_4 : (?x\ sameAs\ ?z), (?z\ sameAs\ ?y) \Rightarrow (?x\ sameAs\ ?y)$

Facts: uncertain facts are in red, certain facts are in blue

$f_1 : (i_1\ sameName\ i_2)$ $f_2 : (i_1\ sameBirthDate\ i_2)$ $f_3 : (i_2\ marriedTo\ i_3)$

$f_4 : (i_4\ marriedTo\ i_3)$ $f_5 : (i_2\ sameName\ i_4)$

Computation of the inferred facts probabilities

Inferred facts	Uncertainty Provenance	Probability
$(i_2\ sameAs\ i_4)$	$\top$	1
$(i_1\ sameAs\ i_2)$	$e(r_2) \wedge e(f_1)$	$Pr(e(r_2)) \times Pr(e(f_1))$
$(i_1\ sameAs\ i_4)$	$e(r_2) \wedge e(f_1)$	$Pr(e(r_2)) \times Pr(e(f_1))$

Illustrative Example (cont.)

Rules: uncertain rules are in red, certain rules are in blue

$r_1 : (?x\ sameName\ ?y) \Rightarrow (?x\ sameAs\ ?y)$

$r_2 : (?x\ sameName\ ?y), (?x\ sameBirthDate\ ?y) \Rightarrow (?x\ sameAs\ ?y)$

$r_3 : (?x\ marriedTo\ ?z), (?y\ marriedTo\ ?z) \Rightarrow (?x\ sameAs\ ?y)$

$r_4 : (?x\ sameAs\ ?z), (?z\ sameAs\ ?y) \Rightarrow (?x\ sameAs\ ?y)$

Facts: uncertain facts are in red, certain facts are in blue

$f_1 : (i_1\ sameName\ i_2)$ $f_2 : (i_1\ sameBirthDate\ i_2)$ $f_3 : (i_2\ marriedTo\ i_3)$

$f_4 : (i_4\ marriedTo\ i_3)$ $f_5 : (i_2\ sameName\ i_4)$

Computation of the inferred facts probabilities

Inferred facts	Uncertainty Provenance	Probability
$(i_2\ sameAs\ i_4)$	$\top$	1
$(i_1\ sameAs\ i_2)$	$e(r_2) \wedge e(f_1)$	0.8×0.9
$(i_1\ sameAs\ i_4)$	$e(r_2) \wedge e(f_1)$	0.8×0.9

Illustrative Example (cont.)

Rules: uncertain rules are in red, certain rules are in blue

$r_1 : (?x\ sameName\ ?y) \Rightarrow (?x\ sameAs\ ?y)$

$r_2 : (?x\ sameName\ ?y), (?x\ sameBirthDate\ ?y) \Rightarrow (?x\ sameAs\ ?y)$

$r_3 : (?x\ marriedTo\ ?z), (?y\ marriedTo\ ?z) \Rightarrow (?x\ sameAs\ ?y)$

$r_4 : (?x\ sameAs\ ?z), (?z\ sameAs\ ?y) \Rightarrow (?x\ sameAs\ ?y)$

Facts: uncertain facts are in red, certain facts are in blue

$f_1 : (i_1\ sameName\ i_2)$ $f_2 : (i_1\ sameBirthDate\ i_2)$ $f_3 : (i_2\ marriedTo\ i_3)$

$f_4 : (i_4\ marriedTo\ i_3)$ $f_5 : (i_2\ sameName\ i_4)$

Computation of the inferred facts probabilities

Inferred facts	Uncertainty Provenance	Probability
$(i_2\ sameAs\ i_4)$	$\top$	1
$(i_1\ sameAs\ i_2)$	$e(r_2) \wedge e(f_1)$	0.72
$(i_1\ sameAs\ i_4)$	$e(r_2) \wedge e(f_1)$	0.72

Experiments: interlinking DBpedia and MusicBrainz

Size and number of entities in the two datasets

Class	DBpedia	MusicBrainz
Person	1,445,773	385,662
Band	75,661	197,744
Song	52,565	448,835
Album	123,374	1,230,731
Number of RDF triples	73 millions	112 millions

86 rules from which 50 are certain and 36 are uncertain

ID	Rules
sameAsBirthDate	(?x :solrPSimilarName ?l), (?y skos:myLabel ?l), (?x dbo:birthDate ?$date$), (?y mb:beginDateC ?$date$) $\Rightarrow$ (?x :sameAsPerson ?y)
sameAsMemberOfBand	(?x :solrPSimilarName ?l), (?y skos:myLabel ?l), (?y mb:member_of_band ?$gr2$), (?$gr2$ skos:myLabel ?lg), (?$gr1$ dbp:members ?x), (?$gr1$:solrGrSimilarName ?lg) $\Rightarrow$ (?x :sameAsPerson ?y)

Table: Examples of uncertain rules for interlinking person entities in DBpedia and MusicBrainz.

Experimental results

Gain of rule chaining

43,923 links not discovered by Silk among the **144,467 sameAs links discovered by ProbFR** between DBpedia and MusicBrainz

Gain of using uncertain rules for improving recall without losing much in precision (precision and recall estimated on samples)

	DBpedia and MusicBrainz					
	Only certain rules			All rules		
	P	R	F	P	R	F
Person	1.00	0.08	0.15	1.00	0.80	0.89
Band	1.00	0.12	0.21	0.94	0.84	0.89
Song	NA	NA	NA	0.96	0.74	0.84
Album	NA	NA	NA	1.00	0.53	0.69

Gain of exploiting probabilities to filter out wrong sameAs links

	P	R	F
$\text{Band}_{\geqslant 0.90}$	1.00	0.80	0.89
$\text{Song}_{\geqslant 0.60}$	1.00	0.54	0.72

Conclusion on rule-based data interlinking

Probabilistic Datalog: a good trade-off for reasoning with uncertainty in Linked Data

Some restrictions compared to general probabilistic logical frameworks (e.g., Markov Logic)

- uncertain formulas restricted to Horn rules and ground facts
- probabilities computed for inferred facts only

Better scalability and more transparency

- explanations on probabilistic inference for end-users
- useful traces for experts to set-up the rules probabilities

Future work

- A method to set up automatically the threshold for filtering the probabilistic sameAs facts to be retained
- A backward-reasoning algorithm on probabilistic rules for importing on demand useful data from external sources

General conclusion

We have seen three symbolic methods to interlink RDF data:

- keys + property alignment
- linkeys
- Datalog rules + probabilistic weights

Links specified with keys and linkeys can be generated with:

- SPARQL queries
- some declarative similarity-based data-interlinking tool: SILK or Limes
- using ProbFR approach: combining forward reasoning and provenance

http://moex.inria.fr
http://slide.liglab.fr

Manuel . Atencia @ inria . fr
Jerome . David @ inria . fr
Jerome . Euzenat @ inria . fr
Marie-Christine . Rousset @ imag . fr