Provenance for SPARQL queries

Carlos Viegas Damásio (cd@fct.unl.pt)
CENTRIA, Dept. Informática, Univ. Nova de Lisboa, Portugal

Anastasia Analyti (analyti@ics.forth.gr)
Institute of Computer Science, FORTH-ICS, Crete, Greece

Grigoris Antoniou (antoniou@ics.forth.gr)
FORTH-ICS, and Dept. of Computer Science, University of Crete, Crete, Greece

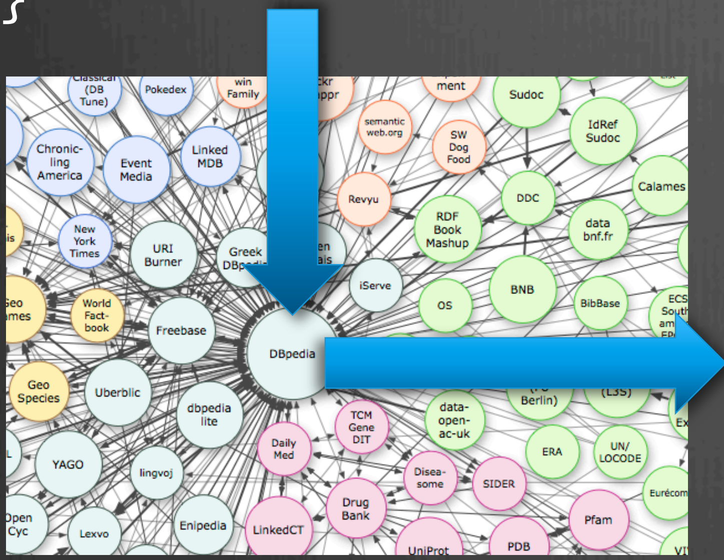
Overview

- Motivation
- Approach
- Representation using K-relations
- Translation of SPARQL graph patterns
- Related work
- Discussion and Conclusions

The problem

```
PREFIX dbpowl: <http://dbpedia.org/ontology/>

SELECT DISTINCT ?country WHERE
{?place ?label "Lisbon"@en .
 ?place a dbpowl:PopulatedPlace .
 ?place dbpowl:country ?country .
}
```



country
http://dbpedia.org/resource/Portugal
http://dbpedia.org/resource/United_States

The solution with duplicates

```
PREFIX dbpowl: <http://dbpedia.org/ontology/>

SELECT ?country WHERE
{?place ?label "Lisbon"@en .
 ?place a dbpowl:PopulatedPlace .
 ?place dbpowl:country ?country .
}
```

country
http://dbpedia.org/resource/Portugal (14 times)
http://dbpedia.org/resource/United_States (5 times)

Lineage of solution dbpedia:United States

- The set of all triples that contribute to the solution:

```
:Columbiana_County,_Ohio       dbpprop:seatWl     "Lisbon"@en .
:Columbiana_County,_Ohio       rdf:type   dbpowl:PopulatedPlace .
:Lisbon,_Illinois              foaf:name  "Lisbon"@en .
:Ransom_County,_North_Dakota   rdf:type   dbpowl:PopulatedPlace .
:Lisbon,_Illinois              dbpprop:name   "Lisbon"@en .
:Ransom_County,_North_Dakota   dbpprop:largestCityWl  "Lisbon"@en .
:Ransom_County,_North_Dakota   dbpprop:seatWl "Lisbon"@en .
:Ransom_County,_North_Dakota   dbpowl:country :United_States .
:Columbiana_County,_Ohio       dbpowl:country :United_States .
:Lisbon,_Illinois              dbpowl:country :United_States .
:Lisbon,_Illinois              rdf:type   dbpowl:PopulatedPlace .
```

Why-provenance dbpedia:United States

- Which sets of triples support the solution:

```
:Columbiana_County,_Ohio       dbpprop:seatWl     "Lisbon"@en .
:Columbiana_County,_Ohio       rdf:type    dbpowl:PopulatedPlace .
:Columbiana_County,_Ohio       dbpowl:country :United_States .
```

```
:Lisbon,_Illinois              foaf:name      "Lisbon"@en .
:Lisbon,_Illinois              dbpowl:country :United_States .
:Lisbon,_Illinois              rdf:type    dbpowl:PopulatedPlace .
```

```
:Lisbon,_Illinois              dbpprop:name   "Lisbon"@en .
:Lisbon,_Illinois              dbpowl:country :United_States .
:Lisbon,_Illinois              rdf:type    dbpowl:PopulatedPlace  .
```

```
:Ransom_County,_North_Dakota   dbpowl:country:United_States .
:Ransom_County,_North_Dakota   dbpprop:seatWl "Lisbon"@en .
:Ransom_County,_North_Dakota   rdf:type    dbpowl:PopulatedPlace .
```

```
:Ransom_County,_North_Dakota   dbpprop:largestCityWl "Lisbon"@en .
:Ransom_County,_North_Dakota   rdf:type    dbpowl:PopulatedPlace .
:Ransom_County,_North_Dakota   dbpowl:country:United_States .
```

How-provenance dbpedia:United States

- How is a solution constructed:

```
  (:Columbiana_County,_Ohio   dbpprop:seatWl     "Lisbon"@en     )
× (:Columbiana_County,_Ohio   rdf:type   dbpowl:PopulatedPlace )
× (:Columbiana_County,_Ohio   dbpowl:country :United_States     )
+ (:Lisbon,_Illinois          foaf:name   "Lisbon"@en           )
× (:Lisbon,_Illinois          rdf:type   dbpowl:PopulatedPlace )
× (:Lisbon,_Illinois          dbpowl:country :United_States     )
+ (:Lisbon,_Illinois          dbpprop:name   "Lisbon"@en        )
× (:Lisbon,_Illinois          rdf:type   dbpowl:PopulatedPlace )
× (:Lisbon,_Illinois          dbpowl:country :United_States     )
+ (:Ransom_County,_North_Dakota   dbpprop:seatWl  "Lisbon"@en    )
× (:Ransom_County,_North_Dakota   rdf:type   dbpowl:PopulatedPlace )
× (:Ransom_County,_North_Dakota dbpowl:country:United_States     )
+ (:Ransom_County,_North_Dakota dbpprop:largestCityWl "Lisbon"@en   )
× (:Ransom_County,_North_Dakota   rdf:type   dbpowl:PopulatedPlace )
× (:Ransom_County,_North_Dakota dbpowl:country:United_States     )
```

Graph source is important

SPARQL endpoint	?place
DBpedia	• dbpedia:Columbiana_County,_Ohio • dbpedia:Lisbon,_Illinois • dbpedia:Ransom_County,_North_Dakota
FactForge beta	• dbpedia:Columbiana_County,_Ohio • dbpedia:Lisbon,_Connecticut • dbpedia:Lisbon,_Florida • dbpedia:Lisbon,_Illinois • dbpedia:Lisbon,_Iowa • dbpedia:Lisbon,_Juneau_County,_Wisconsin • dbpedia:Lisbon,_Maine • dbpedia:Lisbon,_New_Hampshire • dbpedia:Lisbon,_New_York • dbpedia:Lisbon,_North_Dakota • dbpedia:Lisbon,_Ohio • dbpedia:Lisbon,_Waukesha_County,_Wisconsin • dbpedia:Ransom_County,_North_Dakota • yago:Lisbon,_Illinois • fb:m.0s9cd • w-flick:Lisbon,_Illinois

How-provenance for SPARQL

- Previous approaches do not handle how-provenance, particularly do not respect cardinality of solutions
- Even for why-provenance the existing proposals are somewhat limited in the treatment of the OPTIONAL construct, and ignore MINUS, EXISTS and NOT EXISTS
- We take care of the following SPARQL graph patterns:
  - Empty graph patterns
  - Triple patterns
  - AND, UNION, MINUS, OPTIONAL, FILTER and GRAPH
- We do not address aggregations, and property paths

Our approach

- Provenance for Relational Algebra is well-understood and has fundamental results and techniques that can be employed
- We map SPARQL queries into Relational Algebra queries over annotated relations ($\mathcal{K}$-relations)
- Tuples of annotated relation are mappings of ordinary tuples into a commutative semiring $\mathcal{K}$
- In order to be able to handle OPTIONAL and MINUS it is required to use the (universal) commutative ring $\mathcal{K}_{\text{dprov}}$ supporting difference and duplicate elimination

This was claimed to be impossible !

The universal m-semiring $\mathcal{K}_{dprov}$

- The commutative semiring $\mathcal{K}_{dprov}(X)$ is formed by elements constructed inductively:
  - The constants 0, and 1
  - The set of identifiers X (graph names and quad identifiers)
  - The terms (s + t), (s × t), (s – t), $\delta_{ki}(t)$
- Annotations of $\mathcal{K}_{dprov}$ are elements of the quotient structure of the free terms above with respect to the congruence relation induced by the axiomatization of monus-semirings
- This structure obeys to the factorization property, i.e. any query in a monus-semiring can be evaluated in $\mathcal{K}_{dprov}$

Query language $RA^+(-, \delta)$

- $\varnothing(t) = 0$
- $(R_1 \cup R_2)(t) = R_1(t) + R_2(t)$
- $\prod_V (R)(t) = \sum_{\{ t[V] = t'[V] \}} R(t')$
- $\sigma_P(R)(t) = R(t)$ if $P(t)$ is true, $\sigma_P(R)(t) = 0$ otherwise
- $(R_1 \bowtie R_2)(t) = R_1(t|U_1) \times R_2(t \mid U_2)$
- $(\rho_\beta(R)(t))$ is an annotated relation obtained by renaming the columns of R according to bijection β
- $(R_1 - R_2)(t) = R_1(t) - R_2(t)$
- $\delta_{ki}(R)(t) = k_i$ if $R(t) <> 0$, $\delta_{ki}(R)(t) = 0$ otherwise

Mapping of RDF graphs

Graphs:

gid	IRI	
0		*g0*
1	http://dbpedia.org	*g1*
2	http://factforge.net	*g2*

Quads:

gid	sub	pred	obj	
0	_:b1	rdfs:label	"Lisbon"@en	*a0*
0	_:b1	rdfs:label	"Lisboa"@pt	*a1*
1	dbpedia:Lisbon,_Illinois	foaf:name	"Lisbon"@en	*b1*
1	dbpedia:Lisbon,_Illinois	rdf:type	dbpedia:PopulatedPlace	*b2*
1	dbpedia:Lisbon,_Illinois	dbpowl:country	dbpedia:United_States	*b3*
⋮	⋮	⋮	⋮	⋮
2	dbpedia:Lisbon	rdf:type	dbpedia:PopulatedPlace	*c1*
⋮	⋮	⋮	⋮	⋮

SPARQL solutions

- A solution corresponds to an annotated tuple whose columns are the query free variables and a special given graph column

```
PREFIX dbpowl: <http://dbpedia.org/ontology/>
SELECT ?place ?country WHERE
{   [ rdfs:label ?search ]
    GRAPH <http://dbpedia.org>
    { ?place ?label ?search.
      ?place a dbpowl:PopulatedPlace .
     ?place dbpowl:country ?country .
    }
}
```

G	?place	?country	
0	dbpedia:Lisbon,_Illinois	dbpedia:United_States	$g0 \times a0 \times$ $g0 \times g1 \times b1 \times b2 \times b3$
⋮	⋮	⋮	

Empty and triple patterns

Empty graph pattern:

- $[()]^G$ selects all rows of `Graphs` table renaming `gid` by G, and discarding column IRI:

$[()]^G$

G	
0	*g0*
1	*g1*
2	*g2*

Triple graph pattern

- $[t]^G$ performs a select on the `Quads` table

[(?x,rdf:type, dbpedia:PopulatedPlace)]^G

G	?x	
1	dbpedia:Lisbon,_Illinois	*b2*
⋮	⋮	⋮

A glimpse of the translation

Translation of the graph pattern (P1 AND P2) where $\{v_1, \ldots, v_n\}$ are the shared variables of patterns P1 and P2.

$$\Pi_{\substack{G, \\ var(P_1) - var(P_2), \\ var(P_2) - var(P_1), \\ v_1 \leftarrow first(v'_1, v''_1), \ldots, \\ v_n \leftarrow first(v'_n, v''_n)}} \left[\sigma_{comp} \left(\rho_{\substack{v'_1 \leftarrow v_1 \\ \vdots \\ v'_n \leftarrow v_n}} \left([P_1]^G_{\mathcal{R}} \right) \bowtie \rho_{\substack{v''_1 \leftarrow v_1 \\ \vdots \\ v''_n \leftarrow v_n}} \left([P_2]^G_{\mathcal{R}} \right) \right) \right]$$

AND graph patterns

- $[(P1\ AND\ P2)]^G$ multiplies together compatible solutions obtained with $[P1]^G$ and $[P2]^G$ in the same graph G

$[P1]^G$

G	?x	?y	?w	
0	p	q	-	*e1*
0	p	r	s	*e2*
1	-	-	t	*e3*
1	-	-	u	*e4*

$[P2]^G$

G	?x	?y	?z	
0	p	q	-	*f1*
0	p	-	v	*f2*
0	p	q	m	*f3*
1	-	-	n	*f4*

$[(P1\ AND\ P2)]^G$

G	?x	?y	?w	?z	
0	p	q	-	-	$e1 \times f1$
0	p	q	-	v	$e1 \times f2$
0	p	q	-	m	$e1 \times f3$
0	p	r	s	v	$e2 \times f2$
1	-	-	t	n	$e3 \times f4$
1	-	-	u	n	$e4 \times f4$

UNION graph patterns

- $[(P1\ UNION\ P2)]^G$ sums together the solutions obtained with $[P1]^G$ and $[P2]^G$ in the same graph G.

$[P1]^G$

G	?x	?y	?w	
0	p	q	-	*e1*
0	p	r	s	*e2*
1	-	-	t	*e3*
1	-	-	u	*e4*

$[P2]^G$

G	?x	?y	?z	
0	p	q	-	*f1*
0	p	-	v	*f2*
0	p	q	m	*f3*
1	-	-	n	*f4*

$[(P1\ UNION\ P2)]^G$

G	?x	?y	?w	?z	
0	p	q	-	-	*e1* + *f1*
0	p	r	*s*	-	*e2*
0	p	-	-	v	*f2*
0	p	q	-	m	*f3*
1	-	-	t	-	*e3*
1	-	-	u	-	*e4*
1	-	-	-	*n*	*f4*

MINUS graph patterns

- [(P1 MINUS P2)]^G resorts to the difference operator:

[P1]^G

G	?x	?y	
0	p	q	*e1*
0	p	r	*e2*
0	r	*s*	*e3*
0	t	-	*e4*

[P2]^G

G	?y	?z	
0	q	r	*f1*
0	r	t	*f2*
0	r	-	*f3*
0	t	u	*f4*

[(P1 MINUS P2)]^G

G	?x	?y	
0	p	q	$e1 \times (1 - (e1 \times f1))$
0	p	r	$e2 \times (1 - (e2 \times f2 + e2 \times f3))$
0	r	s	*e3*
0	t	-	*e4*

OPTIONAL graph patterns

- $[(P1\ OPTIONAL\ P2)]^G$ constructs more complex annotations also requiring the difference operator:

$[P1]^G$

G	?x	?y	
0	p	q	*e1*
0	p	r	*e2*
0	r	s	*e3*

$[P2]^G$

G	?y	?z	
0	q	r	*f1*
0	r	t	*f2*
0	r	-	*f3*
0	t	u	*f4*

$[(P1\ OPTIONAL\ P2)]^G$

G	?x	?y	?z	
0	p	q	r	$e1 \times f1$
0	p	q	-	$e1 \times (1- (e1 \times f1))$
0	p	r	t	$e2 \times f2$
0	p	r	-	$e2 \times f3$ + $e2 \times (1 -(e2 \times f2 + e2 \times f3))$
0	r	s	-	$e3$

FILTER graph patterns

- If R does not contain [NOT] EXISTS subexpressions then $[(P \text{ FILTER } R)]^G$ keeps the solutions which satisfy the boolean condition R, getting the annotation from $[(P)]^G$
- Otherwise, every solution obtained gets the annotation *a* in $[(P)]^G$ multiplied by:
  - $(1 - (1 - e_i))$ for each EXISTS(P_i) in R, where e_i is the annotation for P_i
  - $(1 - e_j)$ for each NOT EXISTS(P_j) in R, where e_j is the annotation for P_j
- (P1 OPTIONAL (P2 FILTER R)) is translated as usual

GRAPH patterns

- If a specific graph IRI is provided then the annotation returned by executing the query (GRAPH iri_j P) in graph with annotation *g* has the shape:

$$g \times g_j \times [(P)]^{gj}$$

- When a variable graph identifier is used, then the graph pattern is evaluated in each named graph resulting in annotations of the form:

$$g \times (g_1 \times [(P)]^{g1} + \ldots + g_n \times [(P)]^{gn})$$

- If desired, the graph identifiers can be removed by mapping them into 1.

Related Work

- Support RDFS reasoning but do not support SPARQL:
  - [Flouris *et al.*, ISWC 2009] Coloring RDF triples to capture provenance. Support RDFS reasoning but no SPARQL.
  - [Buneman *et al.*, SWPM 2010] Annotation Algebras for RDFS provide an algebraic framework for RDFS reasoning. No treatment of SPARQL.
- Support SPARQL:
  - [Dividino *et al.*, J. of Web Semantics 2009] Querying for provenance, trust, uncertainty and other meta knowledge in RDF. Assume a set-based semantics (no duplicates) – only why-provenance.
  - [Zimmermann *et al.*, J. of Web Semantics 2012] Define a SPARQL based query language for annotated RDFS reasoning, where annotations can be used in the queries. Sum operator is idempotent, UNION is not evaluated in the annotation algebra and OPTIONAL occasionally looses some information. Allow aggregates
  - [Geerts *et al.*, unpublished 2012] Propose a novel algebraic structure called seba (semirings with an embedded boolean) and a full treatment of SPARQL.

Future work

- Capture remaining constructs of SPARQL:
  - Aggregation
  - Property paths
- Relate with explicit provenance models
- Explore relationships to the recent work of [Geerts *et al*, 2012]
- Test in practice the approach with real data and real queries
- Complexity of generated annotations

Conclusions and future work

- A first proposal for extracting how-provenance for a very significant fragment of the SPARQL recommendation, respecting the cardinality of solutions
- Uses established provenance models from the database community
- Annotations can be complex, requiring extra research on practical ways to deal with them