Complex Correspondences for Query Patterns Rewriting

Pascal Gillet   Cássia Trojahn   Ollivier Haemmerlé   Camille Pradel

IRIT & Université de Toulouse 2, Toulouse, France
pascalgillet@ymail.com,{cassia.trojahn,ollivier.haemmerle,camille.pradel}@irit.fr

8th Ontology Matching Workshop at ISWC 2013

Outline

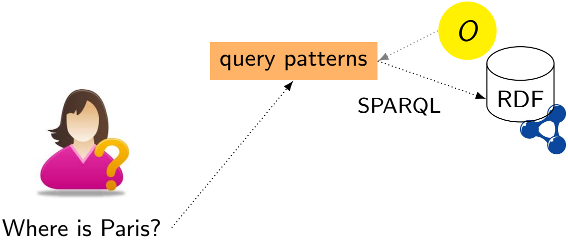
1. Context
2. Foundations
3. Rewriting approach
4. Experiments and discussion
5. Conclusions and perspectives

Outline

1. Context
2. Foundations
3. Rewriting approach
4. Experiments and discussion
5. Conclusions and perspectives

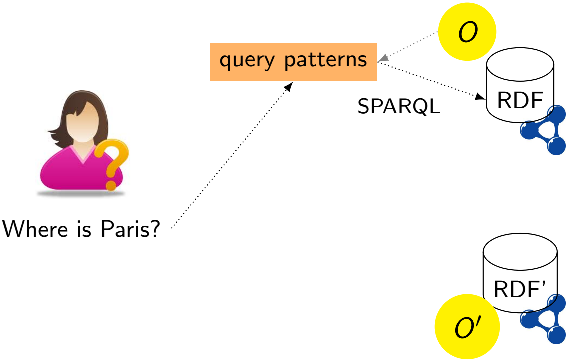
Context

- Hot topic in the Semantic Web community
  - translation of natural language queries into SPARQL
- Swip system [Pradel et al., 2012]
  - query pattern as a family of queries (RDF graphs)
  - pre-written patterns instantiated with respect of a syntactic analysis of the initial query



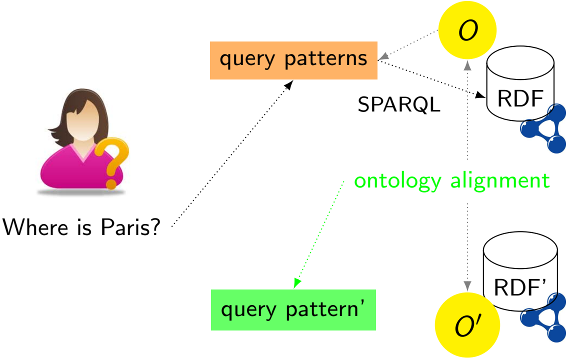
Limitation

- Query patterns are manually built
- Reuse of patterns across different data sets is very limited



Objective

- Use of ontology alignments for rewriting query patterns (applicative context)
- Rewriting patterns requires exploiting more expressive links between ontology entities



Outline

1. Context
2. Foundations
3. Rewriting approach
4. Experiments and discussion
5. Conclusions and perspectives

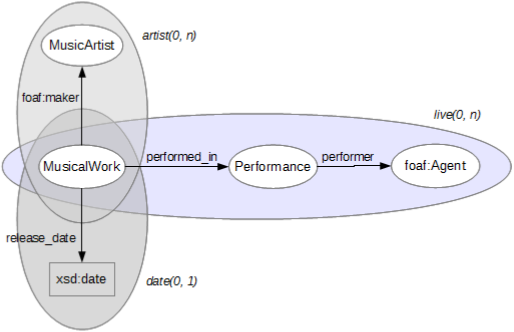
Complex correspondences

- An alignment $A_{O \rightarrow O'}$ is a set of correspondences $\{c_1, c_2, ..., c_n\}$
    - c_i is a 4-tuple $\langle e_O, e_{O'}, r, n \rangle$
    - c_i is **simple** : $Film_O \sqsubseteq Work_{O'}$
    - c_i is **complex** (FOL or DL fragments)
        - $\forall x, Short\_Film(x) \equiv Film(x) \wedge duration(x, y) \wedge y \leq 59$
        - $Short\_Film \equiv Film \sqcap \exists duration. \leq 59$

        - $\forall x, Biopic(x) \equiv Film(x) \wedge Celebrity(y) \wedge topic(x, y)$
        - $Biopic \equiv Film \sqcap \exists topic.Celebrity$

Query patterns

- *RDF graph* representing the prototype of a relevant family of queries
- A pattern p with respect to O is a set of sub-patterns sp_i
  - $p^O = \{sp_1, sp_2, ..., sp_n\}$



Outline

1 Context

2 Foundations

3 Rewriting approach

4 Experiments and discussion

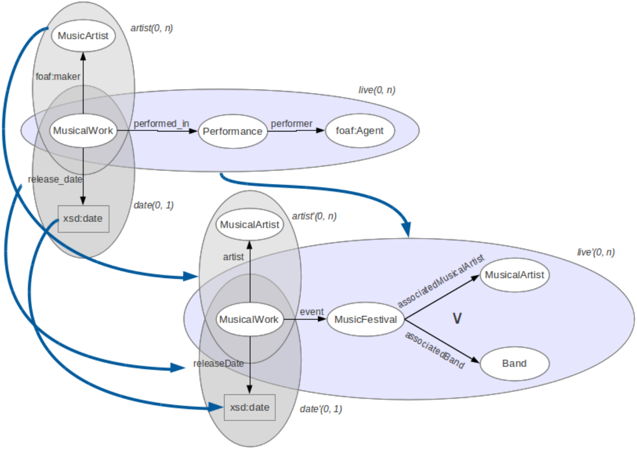
5 Conclusions and perspectives

Rewriting approach

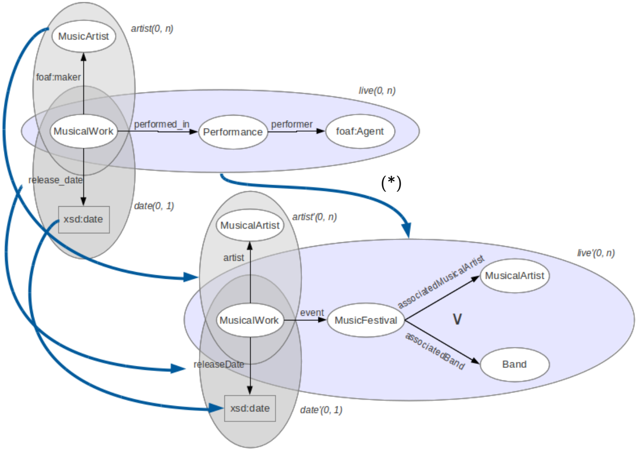
```
Input: $P^O = \{p_1^O, p_2^O, ..., p_n^O\}$,
       $A_{O \rightarrow O'}$
Output: $P^{O'} = \{p_1^{O'}, ..., p_n^{O'}\}$
FRecursRewrite($sg^O, A_{O \rightarrow O'}$)
foreach $e^O \in sg^O$ do
    if $\exists \langle e_O, e_{O'}, r, n \rangle \in A_{O \rightarrow O'}$
    then
        $e_O \leftarrow e_{O'}$;
    else if $e_O$ is class or property
    then
        Discard($sg^O$) ;
        /* cascading rollback
        */
    else
        FRecursRewrite($e_O, A_{O \rightarrow O'}$);
    end
end
return $sg^O$;
```

- *Depth-First Search* algorithm (DFS) for traversing and searching graph data structures in input query patterns:
  - Subpattern $\succ$ RDF triple $\succ$ class or property
  - At each step, we search a correspondence in $A_{O \rightarrow O'}$ for the considered subgraph
- *sp* is an indivisible expression rewritten by chunks (if it is not fully rewritten, it is discarded)
- Conservation of semantics of P_O depends on the completeness of $A_{O \rightarrow O'}$
- Some loss of (semantic) information is acceptable (it could be overcame using other techniques i.e. user interaction)

Rewriting approach



Rewriting approach



(*) $e_i^O = MusicalWork \sqcap \exists performed\_in(Performance \sqcap \exists performer.foaf:Agent)$

$e_j^{O'} = MusicalWork \sqcap \exists event(MusicFestival \sqcap (\exists associatedMusicalArtist.MusicalArtist \sqcup \exists associatedBand.Band))$

Outline

1. Context
2. Foundations
3. Rewriting approach
4. Experiments and discussion
5. Conclusions and perspectives

Query patterns and ontologies

- MusicBrainz patterns
  - Targeting MusicBrainz collection
  - Music Ontology[1] (249 $\mathcal{T}$Box entities)
  - 5 query patterns and 19 sub-patterns
- Cinema patterns
  - $\mathcal{A}$Box of Cinema ontology[2] (300 $\mathcal{T}$Box entities)
  - 6 query patterns 27 sub-patterns
- Rewrite query patterns targeting MusicBrainz/Cinema data sets into patterns targeting DBpedia
  - DBpedia 3.8[3] ontology (2213 $\mathcal{T}$Box entities)

[1] http://musicontology.com/

[2] http://ontologies.alwaysdata.net/cinema

[3] http://wiki.dbpedia.org/Ontology?v=181z

Preliminary experiments : MusicBrainz to DBpedia

- Simple correspondences for rewriting patterns
- Alignments (merge) from a sub-set of OAEI 2012 matching systems
- 67% of Music ontology entities were covered in the alignment
- 25 out of 60 entities in the query patterns replaced by a target entity (coverage of 41%)
- Only 2 sub-patterns out of the 19 sub-patterns could be fully rewritten
- Complex correspondences are needed instead

Complex correspondences : MusicBrainz to DBpedia

- Very few systems able to generate complex correspondences
  - Tools described in [Ritze et al., 2009, Ritze et al., 2010]
  - Set of pre-defined complex correspondence patterns
  - Few complex correspondences were identified for the pair Music-DBpedia
- Manually created set of 28 complex correspondences
  - process guided by the query sub-patterns for Music
  - take into account a set of 11 simple correspondences
  - do not cover all possible correspondences
- 52 multilingual complex correspondences for Cinema-Music (not fully evaluated)

Complex correspondences : MusicBrainz to DBpedia

- Correspondence pattern identified for each generated correspondence
- Patterns : CAT, CAT-1, CAV, PC, IP [Ritze et al., 2009] and AVR (CAV), OR, AND [Scharffe and Fensel, 2008]
- Correspondences as compositions of patterns

#1	**CAV** (Class by Attribute Value) $\mathsf{MusicalManifestation} \sqcap \exists \mathsf{release\_type.album} \equiv \mathsf{Album}$
#3	**CAV** $\sqsubseteq$ **CAT** (CAT : Class by Attribute Type) $\mathsf{MusicalManifestation} \sqcap \exists \mathsf{release\_type.live} \sqsubseteq$ $\mathsf{MusicalWork} \sqcap \exists \mathsf{recordedIn.PopulatedPlace}$
#4	**CAV + CAT** $\sqsupseteq$ **CAT** $\mathsf{MusicalManifestation} \sqcap \exists \mathsf{release\_type.soundtrack} \sqcap \exists \mathsf{composer.foaf{:}Agent} \sqsupseteq$ $\mathsf{Film} \sqcap \exists \mathsf{musicComposer.MusicalArtist}$

Rewriting SPARQL queries : MusicBrainz to DBpedia

- 28 complex correspondences (+11 simple) used for SPARQL rewriting
- SPARQL queries from the benchmark training data in QALD 2013[4]
- 25 (out of 100) SPARQL queries from QALD 2013 were rewritten
  - 18 out of 25 queries are correct and *consistent* : they do not necessarily give the same results, but they do answer the same question
    - 3 of these 18 results give the same number of solutions with exactly the same literals
  - 5 out of the 7 remaining results give no solution at all (no instance)
  - 2 last results are not fully correct since the complex correspondences ahead are not correct themselves

[4] Open challenge on Multilingual Question Answering over Linked Data

Rewriting SPARQL queries : MusicBrainz to DBpedia

- “Are there *members of the Ramones who are not named Ramone* ?” (question #25) over MusicBrainz

```
ASK
WHERE {
?band foaf:name ‘Ramones’ .
?artist foaf:name ?artistname .
?artist mo:member_of ?band .


FILTER (NOT regex(?artistname, “Ramone”))
}
```

```
ASK
WHERE {
?band foaf:name ‘Ramones’@en .
?artist foaf:name ?artistname .
{?band dbo:bandMember ?artist}
UNION
{?band dbo:formerBandMember ?artist} .
FILTER (NOT regex(?artistname, “Ramone”))
}
```

Rewriting query patterns

- Music query patterns rewritten in terms of the DBpedia vocabulary
- Rewriting percentage of 90% of the Music patterns
  - 17 (out of 19) sub-patterns were rewriting
  - 45 (out of 51) sub-patterns from the Cinema patterns
  - Rewritten patterns were injected in the Swip system along the DBpedia data set
  - 5 queries from QALD and originally intended to MusicBrainz were run
  - Generated SPARQL queries are (semantically) correct as long as
    1. correspondences do not apply any disjunction of terms (not currently supported in Swip)
    2. source and target in the correspondences involved have the same information level (basically, equivalence)

Outline

1. Context
2. Foundations
3. Rewriting approach
4. Experiments and discussion
5. Conclusions and perspectives

Conclusions and perspectives

- Reuse of query patterns via ontology alignment
- Rewritten patterns not fully validated (non-support of disjunctions by Swip)
- Approach validated on manually generated complex correspondences
- In the future :
  - propose an approach for complex correspondence generation (nowadays, few systems able to do that)
  - evolve the structure of query patterns in Swip
  - formalise the composition of complex correspondence patterns
  - use EDOAL for representing complex correspondences

References

Pradel, C., Haemmerlé, O., and Hernandez, N. (2012).
A Semantic Web Interface Using Patterns: The SWIP System.
In *Graph Structures for Knowledge Representation and Reasoning*, LNCS, pages 172–187. Springer Berlin Heidelberg.

Ritze, D., Meilicke, C., Sváb-Zamazal, O., and Stuckenschmidt, H. (2009).
A pattern-based ontology matching approach for detecting complex correspondences.
In *4th Workshop on Ontology Matching*.

Ritze, D., Völker, J., Meilicke, C., and Sváb-Zamazal, O. (2010).
Linguistic analysis for complex ontology matching.
In *5th Workshop on Ontology Matching*.

Scharffe, F. and Fensel, D. (2008).
Correspondence patterns for ontology alignment.
In *Knowledge Engineering: Practice and Patterns*, pages 83–92. Springer.