Query Rewriting under Linear $\mathcal{EL}$ Knowledge Bases

Mirko M. Dimartino, Andrea Calì, Alexandra Poulovassilis, Peter T. Wood
{mirko, andrea, ap, ptw}@dcs.bbk.ac.uk

Abstract

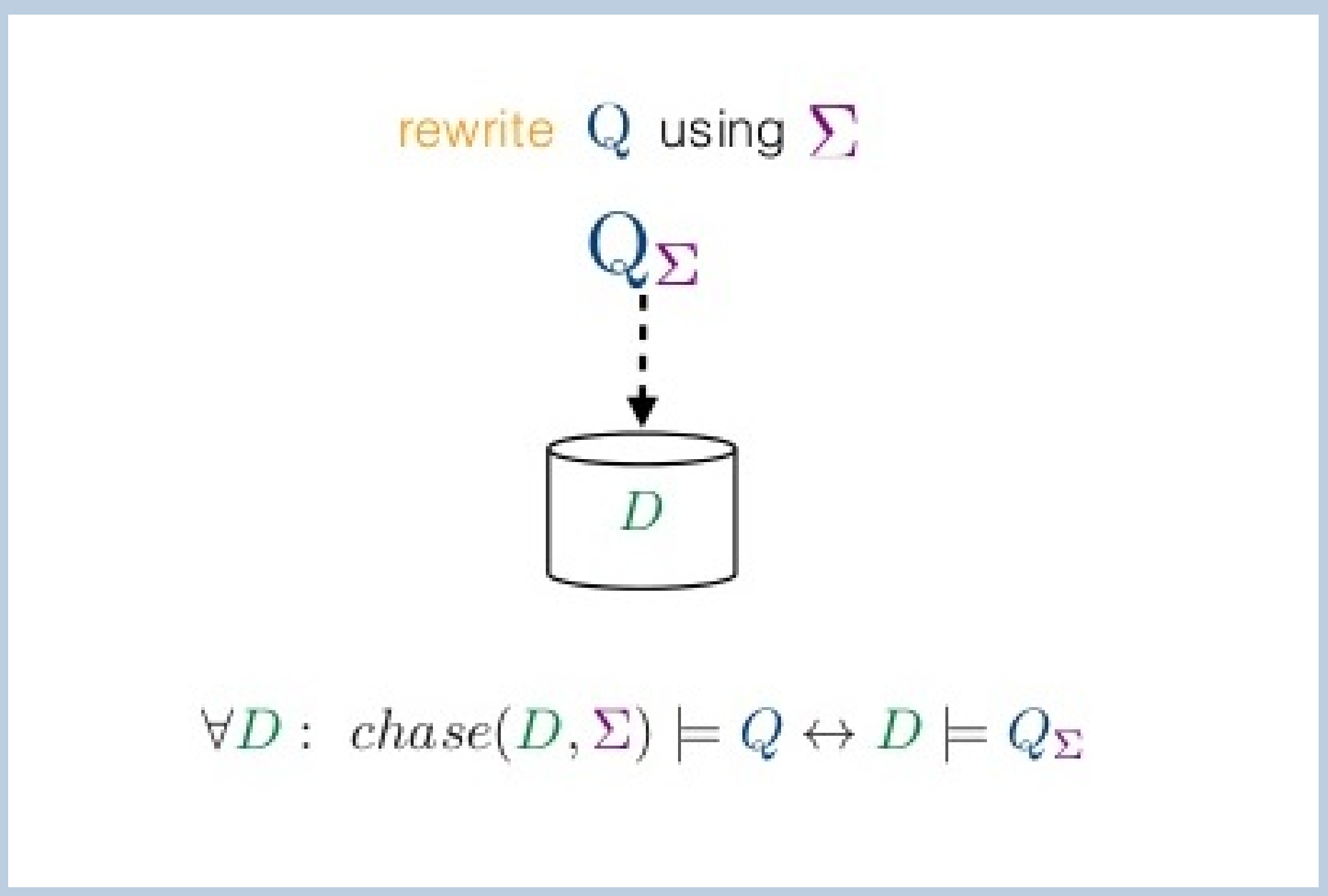
With the adoption of the recent SPARQL 1.1 standard, RDF databases are capable of directly answering more expressive queries than simple conjunctive queries. We exploit such capabilities to answer conjunctive queries (CQs) under ontologies expressed in the description logic called *linear* $\mathcal{EL}^{\ell in}$, a restricted form of $\mathcal{EL}$. In particular, we show a query answering algorithm that rewrites a given CQ into a conjunctive regular path query (CRPQ) which, evaluated on the given instance, returns the correct answer. Our technique is based on the representation of infinite unions of CQs by non-deterministic finite-state automata. Our results achieve optimal data complexity, as well as producing rewritings straightforwardly implementable in SPARQL 1.1.

Example

Answers to queries over knowledge bases can be computed, in certain cases, by a technique called *query rewriting*. In query rewriting, starting from a given query Q, a new query Q_Σ is computed according to a knowledge base

$$\mathcal{K} = (\Sigma, D)$$

where Σ is the ontology and D is the database, such that the answers to Q on $\mathcal{K}$ are obtained by evaluating Q_Σ on D only.



Consider the TBox

$$\mathcal{T} = \{\exists y\ R(x,y), A(y) \rightarrow A(x)\}$$

and the query $\boldsymbol{q}$ defined as $q(x) \leftarrow A(x)$. It is easy to see that answering $\boldsymbol{q}$ via query rewriting requires an infinite union of conjunctive queries for $\boldsymbol{q}'$:

- $q(x) \leftarrow A(x)$,
- $q(x) \leftarrow R(x,y), A(y)$,
- all conjunctive queries of the form

  $q(x) \leftarrow R(x,y_1), \ldots, R(y_k, y_{k+1}), A(y_{k+1})$

  , with $k \geqslant 1$.

Now, in order to capture this infinite rewriting, we can resort to the CRPQ rewriting $\boldsymbol{q}'$ defined as $q(x) \leftarrow R^*(x,y), A(y)$. To achieve this, we leverage the expressive power of a finite-state automata; intuitively, the automaton is able to encode infinite sequences of rewriting steps executed according to the resolution-based query rewriting algorithm.

The automaton

Consider the TBox $\mathcal{T}$ defined by the following Description Logic inclusion assertions:

- $\exists R.C \sqsubseteq \exists P.\top$,
- $\exists P.\top \sqsubseteq A$,
- $\exists P.\top \sqsubseteq B$,
- $\exists T.B \sqsubseteq C$ and
- $\exists S.A \sqsubseteq A$,

where P, R, S, T are role names and A, B, C are concept names. Consider now the query $\boldsymbol{q} = q(x) \leftarrow A(x,y)$. First, we transform $\mathcal{T}$ into normal form, say $\mathcal{T}'$, by adding a fresh concept name X and by replacing $\exists R.C \sqsubseteq \exists P.\top$ by $\exists R.C \sqsubseteq X$ and $X \sqsubseteq \exists P.\top$.
Let us consider the following automaton:

$$\begin{tikzcd}
\text{start} \arrow[r] & S_A \arrow[r, "P"] \arrow[d, "A"] \arrow[loop above, "S"] \arrow[rrrr, bend left, "\epsilon"] & S_\top & S_B \arrow[l, "P"] \arrow[d, "B"] \arrow[rr, bend left, "\epsilon"] & S_C \arrow[l, "T"] \arrow[d, "C"] & S_X \arrow[l, "R"] \arrow[d, "X"] \\
 & SF_A & & SF_B & SF_C & SF_X
\end{tikzcd}$$

The language accepted by the automaton can be described by the following regular expression:

$$(\exists S.)^*(A|X|(RT)^*(P|RC|RT(B|X)))$$

All the infinite rewritings of $\boldsymbol{q}$ with respect to $\mathcal{T}$ are of the form $q(x) \leftarrow \mathsf{NFA}(x,y)$, where NFA is the automaton illustrated above. For instance, some possible rewritings of $\boldsymbol{q}$ are:

$$q(x) \leftarrow S(x,z_1), S(z_1,z_2), P(z_2,y)$$

$$q(x) \leftarrow S(x,z_1), S(z_1,z_2), A(z_2,y)$$

$$q(x) \leftarrow R(x,z_1), T(z_1,z_2), R(z_2,z_3), C(z_3,y)$$

It is easy to verify that the language accepted by this automaton encodes the rewritings of $\boldsymbol{q}$.

Selected publications

- Dimartino, M.M., Calì, A., Poulovassilis, A.,Wood, P.T.: Query Rewriting under Linear EL Knowledge Bases. In: Proc. of RR (2016)
- Dimartino, M.M.: Peer-based Query Rewriting in SPARQL for Semantic Integration of Linked Data. In: Proc. of DC@ISWC (2015)
- Dimartino, M.M., Calì, A., Poulovassilis, A.,Wood, P.T.: Implementing peer-to-peer semantic integration of Linked Data. In: Proc. of BICOD (2015)
- Dimartino, M.M., Calì, A., Poulovassilis, A., Wood, P.T.: Peer-to-peer semantic integration of Linked Data. In: Proc. of EDBT/ICDT Workshops. pp. 213-220 (2015)

Our contributions

- We present a novel rewriting technique, based on non-deterministic finite-state automata, for *atomic* queries on $\mathcal{EL}^{\ell in}$ knowledge bases, with CRPQs as target language. Intuitively, the expressive power of CRPQs is able to finitely capture the infinite rewriting branches of resolution-based rewriting algorithm.

- Based on the rewriting technique for atomic queries, we present a technique for rewriting CQs into CRPQs. This is achieved by splitting the problem in two: first we deal with assertions that do not introduce labelled nulls; then, we show that the rest of the assertions are guaranteed to have only tree-like (or, more precisely, forest-like) models; this allows us to capture all paths (including the infinite ones) from roots in the forest by means of finite-state automata.

  The final rewriting is a CRPQ whose evaluation on the given ABox returns the correct answers to the initial CQ. Since CRPQs can be straightforwardly expressed in SPARQL 1.1, by the means of property paths, our approach is suitable for real-world settings.

- Our technique achieves optimal computational cost in *data complexity*, that is where the TBox and the query are fixed and the ABox alone is a variable input; in fact, our algorithm runs in NLOGSPACE in data complexity, which is the known (tight) bound for CQ answering under $\mathcal{EL}^{\ell in}$ knowledge bases. Notice also that, regarding *combined complexity* (where query, TBox and ABox all constitute the variable input), our rewriting is expressed in the language of CRPQs, which can be evaluated in NP.

  Moreover, in the case of "simple" queries such as atomic queries, our rewriting is expressed as a regular path query, which can be evaluated in NLOGSPACE.

Current and future work

We are extending our work in several directions. The most immediate ones are the following:

1. we plan to consider CRPQs as the language for queries, and devise a suitable rewriting algotithm;
2. we plan to consider inverse roles, and identify syntactic properties that would still guarantee rewritability of CQs into CRPQs; note that the ontology language so defined subsumes $\mathcal{QL}$;
3. we intend to include complex role chains and unions in the language.

We already have results on *(1)* and *(3)*.