From $\mathcal{EL}$ to Tractable Existential Rules with Complex Role Inclusions

Michaël Thomazo

University Montpellier 2

Abstract. Ontology-based data access consists in using ontologies while querying data. Due to the high complexity of this problem, considering lightweight description logics like $\mathcal{EL}$ is especially relevant. Another strand of research is based on existential rules. In this paper, we use this latter formalism in order to cover $\mathcal{EL}$ with the same complexity of reasoning while allowing any predicate arity and some cycles on variables. We then add complex role inclusions to enhance expressivity, while staying polynomial in data complexity and generalizing existing results. In particular, we consider transitivity and right/left identity rules, which do not behave well with respect to usual decidability paradigms.

1 Introduction

Ontology-based data access (OBDA) recently received a lot of attention both from knowledge representation and database communities. This problem can be stated as follows: given a set of facts and an ontology (the knowledge base), one wants to evaluate a conjunctive query against this knowledge base. The ontology can be represented in several ways. Traditional ontology languages are description logics (DLs,[4]). The original focus of DL research was the ontology in itself, with problems like satisfiability or subsumption between concepts. The conjunctive query answering problem has been considered more recently. Classical DLs appeared to be highly complex for that reasoning problem, and lightweight description logics (such as $\mathcal{EL}$ and DL-Lite) are thus very relevant.

A parallel approach relies on existential rules [5, 6], also known as TGDs [1] that form the basis of Datalog$^{\pm}$ [8]. Existential rules are logical formulas of the form $B \rightarrow H$, where B and H are conjunctions of atoms and where H might contain existentially quantified variables. In contrast to DLs, they allow for any predicate arity (which in particular eases the integration with database systems in which relations are naturally translated into n-ary predicates) and can express some cyclic dependencies on variables. On the other hand, neither disjunction nor negation is expressible with existential rules. Interestingly, the two main families of DLs that have been designed for conjunctive query answering can be translated into existential rules. The associated ontology-based conjunctive query answering problem (CQA) is formalized as follows: given a set of facts (in their logical form an existentially closed conjunction of atoms) F, a set of rules $\mathcal{R}$, and a conjunctive query Q, check whether $F, \mathcal{R} \models Q$ hold. This problem is undecidable, and numerous restrictions on $\mathcal{R}$ have been proposed recently in order to ensure decidability (see [12] for a survey), which is usually proven by means of one

of the two following mechanisms, or a combination of both. The first one is forward chaining: rules are iteratively applied to F, until either no new information is added, or the query is entailed. If a set of rules is such that for any fact, this process halts in finite time or generates a set of facts of bounded treewidth (which is defined on a graph naturally associated with the facts, see e.g. [6]), then the CQA problem is decidable ([7, 5]). The second mechanism is backward chaining: a query Q is rewritten into a set of conjunctive queries (which can be seen as a union of conjunctive queries), such that Q is entailed by F and $\mathcal{R}$ if and only if one its rewritings is entailed by F. The CQA problem is decidable if the set of rewritings is finite for any query.

$\mathcal{EL}$ [2, 11] is one of the description logics that have a reasonable complexity for CQA: NP-complete in combined complexity and PTIME-complete in data complexity. As pointed out before, any $\mathcal{EL}$ TBox can be translated into existential rules. However, the smallest known Datalog$^{\pm}$ decidable class covering such rules is a class for which CQA complexity is much higher than the original one (2-EXPTIME-complete in combined complexity). Finally, it is known that one can add to $\mathcal{EL}$ inclusions a special kind of complex role inclusions while keeping polynomial data complexity [10]. As far as we know, such results have no counter-part in the rule framework. Moreover, one of the most used complex role inclusion, namely transitivity, is out of the scope of known decidability criteria when combined with decidable classes of existential rules.

The contribution of this paper is two-fold:

- first, it presents a class of existential rules, namely orientable *fr1*, that covers $\mathcal{EL}$ ontologies while keeping the same (data or combined) complexity for CQA (Theorem 1). The proposed class allows for predicates of arbitrary arity and a form of cyclic dependencies between variables;
- second, it generalizes this class by adding complex role inclusions while staying polynomial in data complexity (Theorem 2); it allows for left and right identity rules, which have been proven useful for modeling purposes [3]. The syntactic regularity condition enforced is close from the one imposed in Horn-$\mathcal{SROIQ}$ ([9]).

In Section 2, basic definitions about $\mathcal{EL}$ and existential rules are recalled. In Section 3, *orientable fr1 rules* are introduced. Section 4 adapts the algorithm presented in [13] and designed for a more general existential rule class, yielding an easier and worst-case optimal algorithm for orientable *fr1* rules. This algorithm is further modified in Section 5 in order to take some complex role inclusions into account. Section 6 concludes the paper.

In this paper, we will avoid technical definitions and rely on examples, to provide an intuition about the main techniques.

2 Preliminaries

We briefly recall the preliminary definitions presented in [6]. An atom is of the form $p(t_1, \ldots, t_k)$ where p is a predicate with arity k, and the t_i are terms, i.e., variable or constants. A *fact* is the existential closure of a conjunction of atoms.[1] An *existential rule*

[1] Note that this notion generalizes the usual definition of fact by taking existential variables generated by rules into account.

(or simply a *rule* when not ambiguous) is a formula $R = \forall \mathbf{x} \forall \mathbf{y}(B[\mathbf{x}, \mathbf{y}] \rightarrow (\exists \mathbf{z} H[\mathbf{y}, \mathbf{z}]))$[2] where $B = body(R)$ and $H = head(R)$ are finite conjunctions of atoms, called the *body* and the *head* of R, respectively. The *frontier* of R, denoted by $fr(R)$, is the set of variables $vars(B) \cap vars(H) = \mathbf{y}$. A rule R is *applicable* to a fact F if there is a homomorphism π from $body(R)$ to F; the result of the *application of R on F w.r.t. π* is a fact $\alpha(F, R, \pi) = F \cup \pi^{\text{safe}}(head(R))$ where π^{safe} is a substitution of $head(R)$, that replaces each $x \in fr(R)$ with $\pi(x)$, and each other variable with a "fresh" variable, i.e., not introduced before. The direct saturation of F with $\mathcal{R}$ is defined as $\alpha(F, \mathcal{R}) = F \cup_{(R=(H,C),\pi) \in \Pi(F,\mathcal{R})} \pi^{\text{safe}}(C)$, where $\Pi(F, \mathcal{R}) = \{(R, \pi) \mid R = (B, H) \in \mathcal{R}$ and π is a homomorphism from H to $F\}$. The k saturation of F with $\mathcal{R}$ is denoted by $\alpha_k(F, \mathcal{R})$ and is such that: $\alpha_0(F, \mathcal{R}) = F$, and for $i > 0$, $\alpha_i(F, \mathcal{R}) = \alpha(\alpha_{i-1}(F, \mathcal{R}), \mathcal{R})$. The universal model of F and $\mathcal{R}$ is the union of $\alpha_k(F, \mathcal{R})$ for $k \in \mathbb{N}$. Q is entailed by F and $\mathcal{R}$ iff it is entailed by $\alpha_k(F, \mathcal{R})$ for some $k \in \mathbb{N}$ (i.e., by the universal model of F and $\mathcal{R}$).

An $\mathcal{EL}$ TBox contains concept inclusions $C_1 \sqsubseteq C_2$, where C_1 and C_2 are concepts built as follows:

$$C := \top \mid A \mid C_1 \sqcap C_2 \mid \exists R.C$$

Any $\mathcal{EL}$ ontology can be translated into a set of existential rules (which are called *$\mathcal{EL}$-rules*), where $C_1 \sqsubseteq C_2$ is translated into $\phi(C_1)(x) \rightarrow \phi(C_2)(x)$, with ϕ inductively built as explained Table 1. The smallest known decidable class covering rules needed for the translation of an arbitrary $\mathcal{EL}$ TBox is the set of frontier-1 (*fr1*) rules, i.e., the set of rules whose body and head share exactly one variable.

Table 1. Translation of an $\mathcal{EL}$ ontology

$\mathcal{EL}$ **concept**	Logical translation ϕ
A	$A(x)$
$C_1 \sqcap C_2$	$\phi(C_1)(x) \wedge \phi(C_2)(x)$
$\exists R.C$	$r(x, y) \wedge \phi(C)(y)$

3 Orientable *fr1* Rules

However, the combined complexity of reasoning with *fr1* rules and $\mathcal{EL}$ is quite different: 2-Exptime-complete in the former case, and NP-complete in the latter – though both are in Ptime for data complexity. In this section, we present a class of rules that covers $\mathcal{EL}$ while keeping the same combined and data complexities.

$\mathcal{EL}$-rules have the following idiosyncrasy: they add information "below" the frontier of the rule, as pictured in Figure 1. Any $\mathcal{EL}$-rule mapping its frontier to x_1 will map its body to dashed atoms, and will create atoms that are also below x_1. This is due to the fact that information about a term "above" x_1 is not even expressible, since no inverse role is possible. *Orientable fr1 rules* generalize this idea.

We define for every predicate r of arity k a strict total order on its positions $r^1, \ldots, r^k$. We denote by $<$ the union of all these relations. Given such a strict partial order, we can associate a directed graph with each set of atoms.

[2] We can now omit quantifiers since there is no ambiguity.

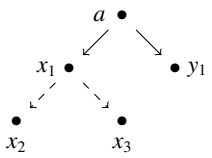

Fig. 1. An arc is directed from the first to the second argument of an atom.

Definition 1 (<-graph associated with a set of atoms). *Let A be a set of atoms. The* <-graph *associated with A is the directed graph defined as follows:*

- *to each term x appearing in A, we assign a vertex v_x,*
- *for any x, y such that $x \neq y$ and x and y appear in the same atom with position $p^x < p^y$, there is an arc from v_x to v_y (note that no loop can occur).*

Intuitively, a rule is oriented for an order $<$ if the atoms needed to trigger it as well as the atoms created by it are situated both in the same right direction according to $<$.

Definition 2 (<-oriented *fr1* rule). *Let R be a rule and $<$ a strict total order on the position of its predicates. R is <-oriented* fr1 *if it is* fr1 *and if the <-graph associated with $\text{body}(R) \cup \text{head}(R)$ is a directed acyclic graph such that there is a path from $\text{fr}(R)$ to any other node.*

Given this notion of rule orientation, we naturally define the notion of orientable *fr1* set of rules.

Definition 3 (Orientable *fr1* set of rules). *A set of rules $\mathcal{R}$ is orientable* fr1 *(or simply orientable when not ambiguous) if there exists an order $<$ such that every rule of $\mathcal{R}$ is <-oriented* fr1.

Example 1 (Orientable fr1 *set of rules).* Let us take $\mathcal{R} = \{r(x,y) \wedge p(y) \rightarrow q(x,z,t) \wedge s(z,t); q(x,y,z) \wedge s(y,z) \rightarrow r(x,t) \wedge r(t,t) \wedge p(t)\}$. By taking $r^1 < r^2, s^1 < s^2$ and $q^1 < q^2 < q^3$, we can easily check that this set of rules is <-oriented. Note that these rules are not translatable in $\mathcal{EL}$ because of the predicate of arity 3 and of cycles.

Property 1 (Recognizability problem) *The problem of deciding whether a set of rules $\mathcal{R}$ is orientable is* NP*-complete.*

The hardness result comes from a reduction of 3-SAT. NP-hardness of the recognizability problem might impede the practical applicability of following results. However, this complexity remains quite small compared to the combined complexity of CQA with a lot of known classes, and, more importantly, even strongly restricted sets of orientable rules are still of interest. Indeed, the next property shows that rules translating $\mathcal{EL}$ ontologies are very naturally oriented (with $R^1 < R^2$ for any role R).

Property 2 (Generalization of $\mathcal{EL}$) *Any set of $\mathcal{EL}$-rules is orientable.*

In the next examples, we assume for all predicate p that $p^i < p^j$ if and only if $i < j$.

4 An Algorithm for CQA with Orientable Rules

In this section, we present a forward chaining algorithm which is a simplified version of the algorithm introduced in [13] for a class of rules called greedy bounded treewidth set, which includes *fr1*. While performing forward chaining, a *greedy tree decomposition* (of bounded width) of the currently generated fact is maintained. We call *bags* the nodes of this tree, which is built as follows: the root of this tree contains all atoms in F, and each time a rule with frontier f is applied by means of a homomorphism π, we create a new bag that contains the newly generated atoms, and choose as its pBarent the root of the tree if $\pi(f)$ is a constant or a variable of F, otherwise the bag in which the variable $\pi(f)$ has been generated.

Example 2 (Greedy tree decomposition). Let $F = \{p(a), q(a)\}$ and $\mathcal{R} = \{R_1 : p(x) \rightarrow r(x, y) \wedge q(y); R_2 : q(x) \rightarrow t(x, y) \wedge p(y)\}$. We show the greedy tree decomposition of $\alpha_4(F, \mathcal{R})$ in Figure 2.

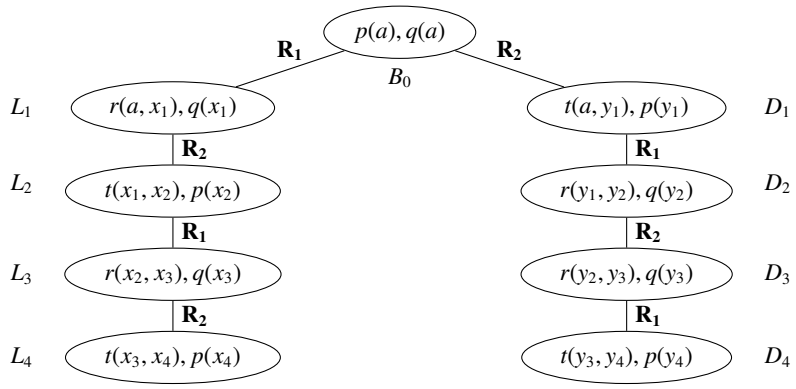

Fig. 2. The greedy tree decomposition of $\alpha_4(F, \mathcal{R})$ (Example 2)

Maintaining this tree decomposition is not sufficient by itself to ensure the termination of the algorithm, since the universal model can be infinite. The main notion used in [13] to ensure the finiteness of the built tree is the equivalence between bags: two bags B and B' are said equivalent when every fact that will eventually be mapped "under B" (i.e., using at least one term generated below B) can be mapped similarly (i.e., up to a bijection between terms of B and B')"under B'", and conversely. If two bags are equivalent, it is only necessary to apply rules below one of them, and the other one will be said "blocked". When no more rule is applicable on a non-blocked bag, we obtain the *full blocked tree*.

This equivalence as well as rule applications are computed in the original algorithm by means of *patterns*, that are attached to each bag. The complexity of the original algorithm is due to the high number of relevant patterns. In the remaining of this section,

we explain how to compute the equivalence relation as well as new rule applications without using patterns – and thus we do not present patterns here.

First, we can simplify equivalence between bags: with orientable *fr1* rules, two bags are equivalent if they have been created by the same rule, as stated by the following property.

Property 3 *Let $\mathcal{R}$ be a set of orientable* fr1 *rules, $R \in \mathcal{R}, R' \in \mathcal{R}$, and F be a fact. Let B_1 and B_2 be two bags of $\mathcal{T}^*$ (the tree decomposition of the universal model of F and $\mathcal{R}$) created by the same rule R. Let z be an existential variable of R, z_1 and z_2 be the corresponding fresh variables in B_1 and B_2. B_1 has a child created by a rule application of R' mapping $\mathrm{fr}(R')$ to z_1 if and only if B_2 has a child created by a rule application of R' mapping $\mathrm{fr}(R')$ to z_2.*

Example 2 (contd.). A full blocked tree of F and $\mathcal{R}$ is represented in Figure 3. L_2 is equivalent to D_1, D_2 to L_1; L_2 and D_2 are blocked and no new rule application can be done on B_0, L_1 or D_1 – this is checked by existence of a $*$-homomorphism, see below.

Second, we check rule applicability by means of $*$-*homomorphism*. This tool is introduced in [13] to evaluate a conjunctive query. Suppose that, at some step of the algorithm, we have generated a blocked tree $\mathcal{T}$. We want to check if there is a homomorphism from Q to the possibly infinite fact encoded by $\mathcal{T}$. This fact can be obtained from $\mathcal{T}$ by a possibly infinite sequence of *completions*, that iteratively copies under blocked bags the atoms found under their equivalent bag (up to variable renaming). Such a homomorphism induces a partition of Q (two atoms are in the same set if they are mapped to the same – initial or added – bag) and a tree structure for this partition (mimicking the tree structure of the image bags). Finally, [13] shows that if there exists such a homomorphism, there exists one requiring only a completion sequence of bounded length. To encode the homomorphism, we thus only need the tree decomposition of Q, homomorphisms from each subset of Q to a bag of $\mathcal{T}$, and the required bounded number of completions: this is the structure called a $*$-homomorphism.

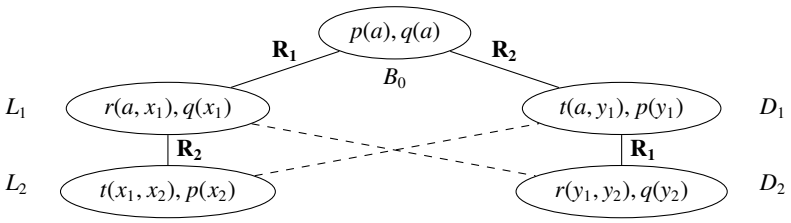

Fig. 3. The full blocked tree of F and $\mathcal{R}$ (Example 2)

Example 3 ($*$-homomorphism). Let $Q = \{r(z, z_1), t(z_1, z_2), r(z_2, z_3), t(z_3, z_4), t(z, z'_1)\}$. Let $\pi = \{(z, a), (z_1, x_1), (z_2, x_2), (z_3, x_3)(z_4, x_4), (z'_1, y_1)\}$ from Q to the fact associated with the tree drawn in Figure 2. The corresponding $*$-homomorphism is pictured in Figure 4.

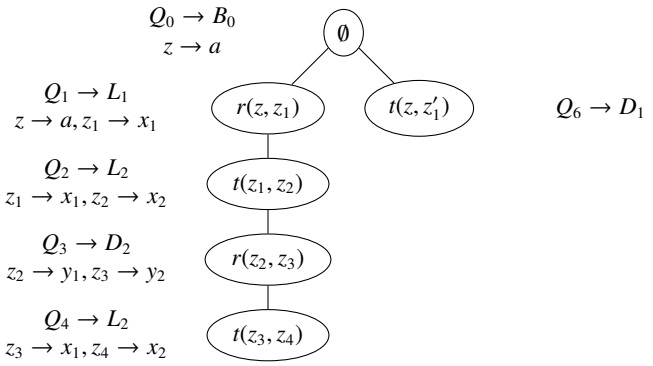

Fig. 4. A $*$-homomorphism from $Q = \{r(z, z_1), t(z_1, z_2), r(z_2, z_3), t(z_3, z_4), t(z, z_1')\}$ to the structure of Figure 3.

Given a tree decomposition of Q and homomorphisms from bags of this decomposition to corresponding bags of $\mathcal{T}$ (as in Figure 4), we check if it is actually a $*$-homomorphism. We check for any pair of adjacent nodes that the associated mappings are compatible, i.e., the image of any term is unique. In the given example, for Q_1, $\{(z, a), (z_1, x_1)\}$ is a homomorphism from $r(z, z_1)$ to $r(a, x_1), q(x_1)$, which is consistent with Q_0 since z is mapped to a in both cases. An interesting consistency check occurs for Q_2 and Q_3: since Q_2 has been mapped to a bag having no child, z_2 has image x_2 when considering Q_2, and y_1 when considering Q_3. However, this is consistent since when copying D_2 under L_2, we rename y_1 by x_2.

Theorem 1. *The conjunctive query answering problem with a set of orientable* fr1 *rules is* Ptime*-complete for data complexity and* NP*-complete for combined complexity.*

Proof. The Ptime membership comes from similar result for *fr1* rules in [13]. The NP-membership is due to the fact that the structure we build is of polynomial size in F and $\mathcal{R}$. The hardness holds because it generalizes $\mathcal{EL}$.

It is interesting to note that our algorithm, when further restricted to $\mathcal{EL}$ ontologies, becomes similar to the algorithm presented in [11]. The main difference is that, while our algorithm maintains equivalence classes, the algorithm in [11] *merges* equivalent bags. These merges result in the finiteness of the process, but also in the loss of homomorphism soundness with respect to first-order semantics. Soundness is regained by modifying homomorphism check, in a way both based on the orientation of $\mathcal{EL}$ and on its tree-model property.

5 Adding Complex Role Inclusions

Complex role inclusions are arguably useful in practice. However, this kind of rules does not behave well with respect to both forward chaining and backward chaining, as illustrated with Example 4.

Example 4. Let $\mathcal{R} = \{p(x) \rightarrow r(x,y) \wedge p(y); r(x,y) \wedge r(y,z) \rightarrow r(x,z)\}$. The query $Q = \{q(x), r(x,y), q(y)\}$ is not rewritable w.r.t. $\mathcal{R}$ into a finite union of conjunctive queries, and $\mathcal{R}$ does not generate facts of bounded treewidth either: a clique of arbitrary size can be built by performing forward chaining on the fact $p(a)$.

In order to stay as close as possible to the algorithm of previous section, we split any fact generated during the chase into two parts: atoms generated by *fr1* rules (the *backbone*), and those generated by role inclusions. The first part is of bounded treewidth, and can be managed in the same way as before. The second part will be dealt with in a different fashion: we restrict the set of allowed role inclusions in order to deal with them by means of automata, as it was already done in [10]. In the following, we define an abstract condition of regularity for so-called oriented join rules, and an easily checkable syntactic condition that generalizes existing results. We then illustrate the algorithm with Example 5.

Definition 4 (Join rule). *A* join rule *is a rule of the following shape:* $r(x,y) \wedge s(y,z) \rightarrow t(x,z)$. *Transitivity rules are the case where* $r = s = t$, *left-identity rules where* $r = t$.

Definition 5 (Backbone). *Let* F *be a fact,* $\mathcal{R} = \mathcal{R}_o \cup \mathcal{R}_j$ *where* $\mathcal{R}_o$ *is a set of* fr1 *rules and* $\mathcal{R}_j$ *is a set of join rules. The backbone associated with* F *and* $\mathcal{R}$ *is the subset of atoms of the universal model of* F *and* $\mathcal{R}$ *that have been created by a* fr1 *rule.*

The backbone is of bounded treewidth, and a tree decomposition can be built greedily. In order to have a counter-part of Property 3 on bag equivalence, we consider only orientable join rules.

Definition 6 (<-oriented join rules). *Let* R *be the following join rule:* $r(x,y) \wedge s(y,z) \rightarrow t(x,z)$. R *is a* $<$*-oriented join rule if:*

- *either* $r^1 < r^2, s^1 < s^2$ *and* $t^1 < t^2$,
- *or* $r^1 > r^2, s^1 > s^2$ *and* $t^1 > t^2$.

The following property allows us to keep a trivial equivalence relation on bags. Indeed, it ensures that the criterion used in the previous section to check equivalence between bags remains valid when adding <-oriented join rules to our framework.

Property 4 *Let* $\mathcal{R}_o$ *be a set of* $<$*-oriented* fr1 *rules,* $\mathcal{R}_j$ *be a set of* $<$*-oriented join rules (for the same order* $<$*),* $R, R' \in \mathcal{R}_o$, F *be a fact. Let* B_1 *and* B_2 *two bags of* $\mathcal{T}^*$ *(the greedy tree decomposition* of the backbone *associated with* F *and* $\mathcal{R}_o \cup \mathcal{R}_j$*) created by the same rule* $R \in \mathcal{R}_o$. *Let* z *be an existential variable of* R, z_1 *and* z_2 *be the corresponding fresh variables of* B_1 *and* B_2. B_1 *has a child created by a rule application of* R' *mapping* fr(R') *to* z_1 *if and only if* B_2 *has a child created by a rule application of* R' *mapping* fr(R') *to* z_2.

Having a finite representation of the backbone, we use *regularity* in order to manage join rules. Given a fact F and x and y two terms of F, we denote by $\mathcal{P}_F(x,y)$ the set of words that describe a finite elementary path from x to y. For instance, in the query Q of Figure 4, $\mathcal{P}_F(z, z_3) = \{rtr\}$.

Definition 7 (Regularity of a set of join rules). *Let P be a set of binary predicates, and $\mathcal{R}_c$ be a set of join rules over P. $\mathcal{R}_c$ is regular if for any predicate $p \in P$, there exists a regular language $\mathcal{L}_p$ such that, for any fact F , for any $x, y \in \text{terms}(F)$, the following holds:*

$$F, \mathcal{R}_c \models p(x, y) \Leftrightarrow \mathcal{P}_F(x, y) \cap \mathcal{L}_p \neq \emptyset$$

Similarly to a $*$-homomorphism, a $*_j$-homomorphism is a labeled tree structure, together with a mapping from its bags to the bags of the full blocked tree. In the $*$-homomorphism case, the consistency check consists only in checking that each term of the query has a well-defined image. In the $*_j$-homomorphism case, we also have to check that atoms generated by complex role inclusions can be effectively generated. Let us consider Example 5 and Figure 5. In order to have $w(a, x)$ entailed by the universal model, not only should x be mapped to t_1 in a bag B equivalent to B_3, but there should also be a path from a to the image of the frontier of the rule creating B, such that $\mathcal{A}_w$ (Figure 5) ends in s_3 when reading the word associated with that path. It is worth to note that this is not the case for B_3 and B_5, but it holds for B_7, even though B_3, B_5 and B_7 are equivalent. We thus add this information about states in the $*_j$-homomorphism.

Compared to the orientable *fr1* case, the size of a completion needed to map a query can be bigger up to a factor which is exponential in the query and in the automaton used to recognize regular predicates. This does not increase the data complexity of CQA with the union of a <-oriented *fr1* set of rules and a <-oriented and regular set of join rules, which remains PTIME-complete. However further work is required to determine the combined complexity of the problem.

Example 5. Let $\mathcal{R} = \mathcal{R}_o \cup \mathcal{R}_j$ with $\mathcal{R}_o = \{p(x) \rightarrow r(x, y) \wedge r(y, z) \wedge q(z) \wedge p(z), q(x) \rightarrow w(x, y)\}$ and $\mathcal{R}_j = \{r(x, y) \wedge r(y, z) \rightarrow s(x, z), s(x, y) \wedge r(y, z) \rightarrow t(x, z), t(x, y) \wedge w(y, z) \rightarrow w(x, z)\}$. We take a fact $F = \{p(a)\}$ and a query $Q = \{w(a, x)\}$. The regular expressions associated with r, s, t and w are r, $rr + s$, $rrr + sr + t$, and $(rrr + sr + t)^* w$.

There exists a homomorphism π from Q to the completion represented in Figure 6, mapping x to t_3. We represent this homomorphism by the structure represented Figure 7.

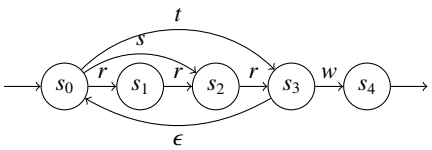

Fig. 5. $\mathcal{A}_w$, an automaton recognizing $\mathcal{L}_w$ (Example 5)

Last, we present a syntactic condition that ensures that a set of join rules is regular. In particular, this condition generalizes the condition proposed in [10].

Definition 8 (Stratified set of join rules). *A set of join rules is* stratified *if the directed graph G built as follows is acyclic. For every predicate p, there exists a vertex v_p in V. There is an arc from v_p to v_q where $p \neq q$ if there exists a rule $r(x, y) \wedge s(y, z) \rightarrow t(x, z)$, where $p = r$ or $p = s$ and $q = t$.*

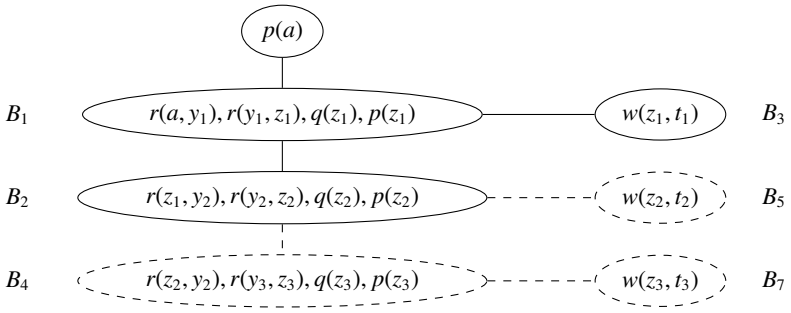

Fig. 6. In plain, the full blocked tree; in dashed, a completion (Example 5). By additionally using (only) role inclusions, $Q = w(a, x)$ can be mapped, mapping x to t_3.

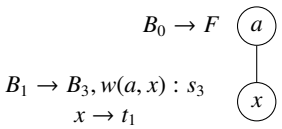

Fig. 7. A $*_j$-homomorphism of $Q = \{w(a, x)\}$ in the full blocked tree of Example 5

Property 5 (Regularity of a stratified set of join rules) *A stratified set of join rules is regular.*

Proof (sketch). By induction on the structure of G. A predicate p whose vertex does not have an incoming arc is associated with the regular expression p. For any q, if we have a regular expression for any p such that (v_p, v_q) is an arc of G, we can build one for q.

Theorem 2. *The CQA problem with rules being the union of of <-oriented* fr1 *rules and <-oriented regular join rules (for the same order <) is* Ptime *in data complexity.*

6 Conclusion

In this preliminary work, we identified a class of existential rules, namely *fr1* orientable rules, covering $\mathcal{EL}$ ontologies while keeping the same complexity for CQA. Rules allow to easily use predicates of any arity and some cycles between variables. We exploited the simplicity of orientable *fr1* rules to simplify the very recent algorithm from [13]. We then investigated how to add some expressive power that could be useful in practice, while staying polynomial in data complexity: we showed how to add transitivity axioms and left- and right-identity rules. Although adding these rules does not fulfill the usual decidability criteria, we adapted the algorithm for orientable rules by using finite automata. Some follow-up naturally come to mind: can we generalize our approach to cover Horn-$\mathcal{SROIQ}$? What is the combined complexity of CQA with the considered rules? What are interesting classes of rules with predicate of any arity that could be added to this basis, while staying polynomial in data complexity? How does the algorithm behave in practice? Moreover, compared to the algorithm presented in [11], the

representation of the universal model uses more space, since equivalent nodes are not merged. Can we adapt our algorithm in order to reduce the space requirements?

Aknowledgment

The author thanks the anonymous reviewers for their useful and constructive comments.

References

1. Abiteboul, S., Hull, R., Vianu, V.: Foundations of Databases. Addison Wesley (1994)
2. Baader, F.: Terminological cycles in a description logic with existential restrictions. In: IJCAI. pp. 325–330 (2003)
3. Baader, F., Brandt, S., Lutz, C.: Pushing the $\mathcal{EL}$ envelope. In: Kaelbling, L., Saffiotti, A. (eds.) Proc. 19th Int. Joint Conf. on Artificial Intelligence (IJCAI'05). pp. 364–369. Professional Book Center (2005)
4. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, second edn. (2007)
5. Baget, J.F., Leclère, M., Mugnier, M.L., Salvat, E.: Extending decidable cases for rules with existential variables. In: IJCAI (2009)
6. Baget, J.F., Leclère, M., Mugnier, M.L., Salvat, E.: On rules with existential variables: Walking the decidability line. Artif. Intell. 175(9-10), 1620–1654 (2011)
7. Calì, A., Gottlob, G., Kifer, M.: Taming the infinite chase: Query answering under expressive relational constraints. In: Proceedings of the Eleventh International Conference on the Principles of Knowledge Representation and Reasoning (KR 2008), Sydney, Australia. pp. 70–80 (2008)
8. Calì, A., Gottlob, G., Lukasiewicz, T.: A general datalog-based framework for tractable query answering over ontologies. In: Proceedings of the 28th symposium on Principles of database systems (PODS'09). pp. 77–86. ACM, New York, NY, USA (2009), http://doi.acm.org/10.1145/1559795.1559809
9. Horrocks, I., Kutz, O., Sattler, U.: The even more irresistible SROIQ. In: KR. pp. 57–67 (2006)
10. Krötzsch, M., Rudolph, S.: Conjunctive queries for $\mathcal{EL}$ with role composition. In: DL. vol. 250 (2007)
11. Lutz, C., Toman, D., Wolter, F.: Conjunctive query answering in the description logic $\mathcal{EL}$ using a relational database system. In: IJCAI (2009)
12. Mugnier, M.L.: Ontological query answering with existential rules. In: RR. pp. 2–23 (2011)
13. Thomazo, M., Baget, J.F., Mugnier, M.L., Rudolph, S.: A generic querying algorithm for greedy sets of existential rules. In: KR ((to appear) 2012)