Conjunctive Queries for $\mathcal{EL}$ with Role Composition

Markus Krötzsch and Sebastian Rudolph

Institute AIFB, Universität Karlsruhe, Germany
{mak|sru}@aifb.uni-karlsruhe.de

Abstract. $\mathcal{EL}^{++}$ is a rather expressive description logic (DL) that still admits polynomial time inferencing for many reasoning tasks. Conjunctive queries are an important means for expressive querying of DL knowledge bases. We address the problem of computing conjunctive query entailment for $\mathcal{EL}^{++}$ knowledge bases. As it turns out, querying unrestricted $\mathcal{EL}^{++}$ is actually undecidable, but we identify restrictions under which query answering becomes decidable and even tractable. We give precise characterisations of schema, query, and combined complexity. To the best of our knowledge, the presented algorithm is the first to answer conjunctive queries in a DL that admits complex role inclusion axioms.

1 Introduction

Conjunctive queries originated from research in relational databases [1], and, more recently, have been considered for expressive description logics (DLs) as well [2–6]. Algorithms for answering (extensions of) conjunctive queries in the expressive DL $\mathcal{SHIQ}$ have been discussed in [3, 4], but the first algorithm that supports queries for transitive roles was presented only very recently [6].

Modern DLs, however, allow for complex role inclusion axioms that encompass role composition and further generalise transitivity. To the best of our knowledge, no algorithms for answering conjunctive queries in those cases have been proposed yet. A relevant logic of this kind is $\mathcal{SROIQ}$ [7], the basic DL considered for OWL 1.1.[1] Another interesting DL that admits complex role inclusions is $\mathcal{EL}^{++}$ [8, 9], which has been proposed as a rather expressive logic for which many inference tasks can be computed in polynomial time. In this paper, we present a novel algorithm for answering conjunctive queries in $\mathcal{EL}^{++}$, which is based on an automata-theoretic formulation of complex role inclusion axioms that was also found useful in reasoning with $\mathcal{SROIQ}$ [10, 7].

Our algorithm in particular allows us to derive a number of complexity results related to conjunctive query answering in $\mathcal{EL}^{++}$. We first show that conjunctive queries in $\mathcal{EL}^{++}$ are undecidable in general, and identify the $\mathcal{EL}^{++}$-fragment of $\mathcal{SROIQ}$ as an appropriate decidable sub-DL. Under some related restrictions of role inclusion axioms, we show that conjunctive query answering in general is PSpace-complete. Query answering for fixed knowledge bases (query complexity) is shown to be NP-complete, whereas for fixed queries (schema complexity) it is merely P-complete.

After introducing some preliminaries in Section 2, we present a general undecidability result for conjunctive queries $\mathcal{EL}^{++}$ in Section 3. Thereafter, we present a modified, automata-based inferencing procedure for $\mathcal{EL}^{++}$ in Section 4. This will be the basis for

[1] http://owl1_1.cs.manchester.ac.uk/

the algorithm for checking entailment of conjunctive queries as presented in Section 5, which operates on a fragment of $\mathcal{EL}^{++}$ for which this problem is decidable. Finally, we derive a number of complexity results related to conjunctive queries in $\mathcal{EL}^{++}$. Proofs are usually omitted in the extended abstract for reasons of space. They are found in the accompanying technical report [11].

2 Preliminaries

We assume the reader to be familiar with the basic notions of description logics (DLs). The DLs that we will encounter in this paper are $\mathcal{EL}^{++}$ [8] and, marginally, $\mathcal{SROIQ}$ [7]. A *signature* of DL consists of a finite set of *role names* $\mathbf{R}$, a finite set of *individual names* $\mathbf{I}$, and a finite set of *concept names* $\mathbf{C}$, and we will use this notation throughout the paper. $\mathcal{EL}^{++}$ supports *nominals*, which we conveniently represent as follows: for any $a \in \mathbf{I}$, there is a concept $\{a\} \in \mathbf{C}$ such that $\{a\}^{\mathcal{I}} = \{a^{\mathcal{I}}\}$ (for any interpretation $\mathcal{I}$). As shown in [8], any $\mathcal{EL}^{++}$ knowledge base is equivalent to one in *normal form*, only containing the following axioms:

TBox:	$A \sqsubseteq C$	$A \sqcap B \sqsubseteq C$	$A \sqsubseteq \exists R.C$	$\exists R.A \sqsubseteq C$
RBox:	$R \sqsubseteq T$	$R \circ S \sqsubseteq T$		

where $A, B \in \mathbf{C} \cup \{\top\}$, $C \in \mathbf{C} \cup \{\bot\}$, and $R, S, T \in \mathbf{R}$. Note that ABox statements of the forms $C(a)$ and $R(a, b)$ are internalised into the TBox. The standard model theoretic semantics of $\mathcal{EL}^{++}$ can be found in [8]. Unless otherwise specified, the letters C, D, E in the remainder of this work always denote (arbitrary) concept names, and the letters R, S denote (arbitrary) role names. We do not consider concrete domains in this paper, but are confident that our results can be extended accordingly.

For *conjunctive queries*, we largely adopt the notation of [6] but directly allow for individuals in queries. Let $\mathbf{V}$ be a countable set of *variable names*. Given elements x, $y \in \mathbf{V} \cup \mathbf{I}$, a *concept atom* (*role atom*) is an expression $C(x)$ with $C \in \mathbf{C}$ ($R(x, y)$ with $R \in \mathbf{R}$). A *conjunctive query* q is a set of concept and role atoms, read as a conjunction of its elements. By $\mathsf{Var}(q)$ we denote the set of variables occurring in q. Consider an interpretation $\mathcal{I}$ with domain $\Delta^{\mathcal{I}}$, and a function $\pi : \mathsf{Var}(q) \cup \mathbf{I} \to \Delta^{\mathcal{I}}$ such that $\pi(a) = a^{\mathcal{I}}$ for all $a \in \mathbf{I}$. We define

$$\mathcal{I}, \pi \models C(x) \text{ if } \pi(x) \in C^{\mathcal{I}}, \quad \text{and} \quad \mathcal{I}, \pi \models R(x, y) \text{ if } (\pi(x), \pi(y)) \in R^{\mathcal{I}}.$$

If there is some π such that $\mathcal{I}, \pi \models A$ for all atoms $A \in q$, we write $\mathcal{I} \models q$ and say that $\mathcal{I}$ *models* q. A knowledge base KB entails q if all models of KB entail q.

3 Conjunctive Queries in $\mathcal{EL}^{++}$

We first investigate the complexity of conjunctive queries in general $\mathcal{EL}^{++}$ as defined in [8]. The following result might be mildly surprising, but is in fact closely related to similar results for logics with complex role expressions (see, e.g., [12]).

Theorem 1. *For an $\mathcal{EL}^{++}$ knowledge base KB and a conjunctive query q, the entailment problem $KB \models q$ is undecidable. Likewise, checking class subsumptions in $\mathcal{EL}^{++}$ extended with inverse roles or role conjunctions is undecidable, even if those operators occur only in the concepts whose subsumption is checked.*

Proof. Intuitively, the result holds since RBoxes can encode context-free languages, the intersection of which can then be checked with conjunctive queries/inverse roles/role conjunctions. This problem is undecidable. The proof in [11] uses an even simpler reduction of the undecidable Post correspondence problem. □

Clearly, arbitrary role compositions are overly expressive when aiming for a decidable (or even tractable) logic that admits conjunctive queries. We thus restrict our attention to the fragment of $\mathcal{EL}^{++}$ that is in the (decidable) description logic $\mathcal{SROIQ}$ [7], and investigate its complexity with respect to conjunctive query answering.

Definition 1. *An $\mathcal{EL}^{++}$ RBox in normal form is* regular *if there is a strict partial order $\prec$ on $\mathbf{R}$ such that, for all role inclusion axioms $R_1 \sqsubseteq S$ and $R_1 \circ R_2 \sqsubseteq S$, we find $R_i \prec S$ or $R_i = S$ $(i = 1, 2)$. An $\mathcal{EL}^{++}$ knowledge base is regular if it has a regular RBox.*

The existence of $\prec$ ensures that the role hierarchy does not contain cyclic dependencies other than through direct recursion of a single role.

4 Reasoning Automata for $\mathcal{EL}^{++}$

In this section, we describe the construction of an automaton that encodes certain concept subsumptions entailed by an $\mathcal{EL}^{++}$ knowledge base. The automaton itself is closely related to the reasoning algorithm given in [8], but the representation of entailments via nondeterministic finite automata (NFA) will be essential for the query answering algorithm in the following section. We describe an NFA $\mathcal{A}$ as a tuple $(Q_\mathcal{A}, \Sigma_\mathcal{A}, \delta_\mathcal{A}, i_\mathcal{A}, F_\mathcal{A})$, where $Q_\mathcal{A}$ is a finite set of states, $\Sigma_\mathcal{A}$ is a finite alphabet, $\delta_\mathcal{A} : Q_\mathcal{A} \times Q_\mathcal{A} \to 2^{\Sigma_\mathcal{A}}$ is a transition function that maps pairs of states to sets of alphabet symbols,[2] $i_\mathcal{A}$ is the initial state, and $F_\mathcal{A}$ is a set of final states.

Consider an $\mathcal{EL}^{++}$ knowledge base KB. Given a concept name $A \in \mathbf{C}$, we construct an NFA $\mathcal{A}_{KB}(A) = (Q, \Sigma, \delta, i, F)$ that computes superconcepts of A, where we omit the subscript if KB is clear from the context. Set $Q = F = \mathbf{C} \cup \{\top\}$, $\Sigma = \mathbf{C} \cup \mathbf{R} \cup \{\top, \bot\}$, and $i = A$. The transition function δ is initially defined as $\delta(C, C) := \{C, \top\}$ (for all $C \in Q$), and extended iteratively by applying the rules in Table 1. The rules correspond to completion rules in [8, Table 2], though the conditions for (CR6) are slightly relaxed, fixing a minor glitch in the original algorithm.

It is easy to see that the rules of Table 1 can be applied at most a polynomial number of times. The words accepted by $\mathcal{A}(A)$ are strings of concept and role names. For each such word w we inductively define a concept expression C_w as follows:

- if w is empty, then $C_w = \top$,
- if $w = Rv$ for some $R \in \mathbf{R}$ and word v, then $C_w = \exists R.(C_v)$,
- if $w = Cv$ for some $C \in \mathbf{C}$ and word v, then $C_w = C \sqcap C_v$.

For instance, the word $CRDES$ translates into $C_{CRDES} = C \sqcap \exists R.(D \sqcap E \sqcap \exists S.\top)$. Based on the close correspondence of the above rules to the derivation rules in [8], we can now establish the main correctness result for the automaton $\mathcal{A}(A)$.

[2] A possibly more common definition is to map pairs of states and symbols to sets of states, but the above is more convenient for our purposes.

Table 1. Completion rules for constructing an NFA from an $\mathcal{EL}^{++}$ knowledge base KB.

(CR1) If $C' \in \delta(C,C)$, $C' \sqsubseteq D \in KB$, and $D \notin \delta(C,C)$ then $\delta(C,C) := \delta(C,C) \cup \{D\}$.
(CR2) If $C_1, C_2 \in \delta(C,C)$, $C_1 \sqcap C_2 \sqsubseteq D \in KB$, and $D \notin \delta(C,C)$ then $\delta(C,C) := \delta(C,C) \cup \{D\}$.
(CR3) If $C' \in \delta(C,C)$, $C' \sqsubseteq \exists R.D \in KB$, and $R \notin \delta(C,D)$ then $\delta(C,D) := \delta(C,D) \cup \{R\}$.
(CR4) If $R \in \delta(C,D)$, $D' \in \delta(D,D)$, $\exists R.D' \sqsubseteq E \in KB$, and $E \notin \delta(C,C)$ then $\delta(C,C) := \delta(C,C) \cup \{E\}$.
(CR5) If $R \in \delta(C,D)$, $\bot \in \delta(D,D)$, and $\bot \notin \delta(C,C)$ then $\delta(C,C) := \delta(C,C) \cup \{\bot\}$.
(CR6) If $\{a\} \in \delta(C,C) \cap \delta(D,D)$, and there are states $C_1, \ldots, C_n$ such that
- $C_1 \in \{C, \top, A\} \cup \{\{b\} \mid b \in \mathbf{I}\}$,
- $\delta(C_j, C_{j+1}) \neq \emptyset$ for all $j = 1, \ldots, n-1$,

as well as $C_n = D$, and $\delta(D,D) \not\subseteq \delta(C,C)$ then $\delta(C,C) := \delta(C,C) \cup \delta(D,D)$.
(CR7) If $R \in \delta(C,D)$, $R \sqsubseteq S$, and $S \notin \delta(C,D)$ then $\delta(C,D) := \delta(C,D) \cup \{S\}$.
(CR8) If $R_1 \in \delta(C,D)$, $R_2 \in \delta(D,E)$, $R_1 \circ R_2 \sqsubseteq S$, and $S \notin \delta(C,E)$ then $\delta(C,E) := \delta(C,E) \cup \{S\}$.

Theorem 2. *Consider a knowledge base KB, concept A, and NFA $\mathcal{A}(A)$ as above, and let w be some word over the associated alphabet. Then $KB \models A \sqsubseteq C_w$ iff one of the following holds:*

- *$\mathcal{A}(A)$ accepts the word w, or*
- *there is a transition $\bot \in \delta(C,C)$ where $C = \top$, $C = A$, or $C = \{a\}$ for some individual a.*

In particular, $\mathcal{A}(A)$ can be used to check all subsumptions between A and some atomic concept B.

The second item of the theorem addresses the cases where A is inferred to be empty (i.e. inconsistent) or where the whole knowledge base is empty. While the above yields an alternative formulation of the $\mathcal{EL}^{++}$ reasoning algorithm presented in [8], it has the advantage that it also encodes all *paths* within the inferred models. This will be essential for our results in the next section where we will use the following convenient definition.

Definition 2. *Consider a knowledge base KB, concepts $A, B \in \mathbf{C}$, and the NFA $\mathcal{A}(A) = (Q, \Sigma, \delta, i, F)$. The automaton $\mathcal{A}_{KB}(A,B)$ (or just $\mathcal{A}(A,B)$) is defined as $(Q, \mathbf{R}, \delta, i, F')$ where $F' = \emptyset$ whenever $\bot \in \delta(A,A)$, and $F' = \{B\}$ otherwise.*

$\mathcal{A}(A,B)$ obviously accepts all words $R_1, \ldots, R_n$ such that $A \sqsubseteq \exists R_1(\ldots \exists R_n.B \ldots)$ is a consequence of KB, with the border case where $n = 0$ and $KB \models A \sqsubseteq B$. Moreover, the language accepted by the NFA is empty whenever $A \sqsubseteq \bot$ has been inferred.

5 Deciding Conjunctive Queries for $\mathcal{EL}$

In this section, we present a nondeterministic algorithm that decides the entailment of a query q with respect to some knowledge base KB. This is done by constructing a so-called *proof graph* which establishes, for any interpretation $\mathcal{I}$ of KB, the existence of a suitable function π that shows query entailment.

Formally, a proof graph is a tuple (N, L, E) consisting of a set of nodes N, a labelling function $L : N \to \mathbf{C} \cup \{\top\}$, and a partial transition function $E : N \times N \to \mathbf{A}$, where $\mathbf{A}$is the set of all NFA over the alphabet $\mathbf{C} \cup \{\top, \bot\} \cup \mathbf{R}$. The nodes of the proof graph

are abstract representations of elements in the domain of some model of KB. The labels assign a concept to each node, and our algorithm ensures that the represented element is necessarily contained in the interpretation of this concept. Intuitively, the label encodes all relevant concept information about one node. This is only possible since (1) KB is in normal form and thus supplies concept names for all composite concept expressions such as conjunctions, and (2) $\mathcal{EL}^{++}$ does not allow inverse roles or number restrictions that could be used to infer further information based on the relationship of an element to elements in the model. Finally, the transition function encodes paths in each model, which provide the basis for inferencing about role relationships between elements. It would be possible to adopt a more concrete representation for role paths (e.g. by guessing a single path), but our formulation reduces nondeterminism and eventually simplifies our investigation of algorithmic complexity.

The automaton of Definition 2 encodes concept subsumptions based on TBox and RBox. For deciding query entailment we also require automata that represent the content of the RBox.

Proposition 1. *Given a regular $\mathcal{EL}^{++}$ RBox, and some role $R \in \mathbf{R}$, there is an NFA $\mathcal{A}(R)$ over the alphabet $\mathbf{R}$ which accepts a word $R_1 \dots R_n$ iff $R_1 \circ \dots \circ R_n \sqsubseteq R$ is a consequence of every $\mathcal{EL}^{++}$ knowledge base with the given RBox.*

Proof sketch. One possible construction for the required automaton is discussed in [7]. Intuitively, the RBox can be understood as a grammar for a regular language, for which an automaton can be constructed in a canonical way. □

The above construction might be exponential for some RBoxes. In [10], restrictions have been discussed that prevent this blow-up, leading to NFA of only polynomial size w.r.t. the RBox. Accordingly, an RBox is *simple* whenever, for all axioms of the form $R_1 \circ S \sqsubseteq S$, $S \circ R_2 \sqsubseteq S$, the RBox does not contain a common subrole R of R_1 and R_2 for which there is an axiom of the form $R \circ S' \sqsubseteq R'$ or $S' \circ R \sqsubseteq R'$. We will usually consider only such simple RBoxes whenever the size of the constructed automata matters.

We are now ready to present the algorithm. It proceeds in various consecutive steps:

Query factorisation. The algorithm nondeterministically selects a variable $x \in \mathsf{Var}(q)$ and some element $e \in \mathsf{Var}(q) \cup \mathbf{I}$, and replaces all occurrences of x in q with e. This step can be executed an arbitrary number of times (including zero).

Proof graph initialisation. The proof graph (N, L, E) is initialised by setting $N := \{\top\} \cup \mathbf{I} \cup \mathsf{Var}(q)$. L is initialised by $L(\top) := \top$, $L(a) := \{a\}$ for each $a \in \mathbf{I}$. For each $x \in \mathsf{Var}(q)$, the algorithm selects a label $L(x) \in \mathbf{C} \cup \{\top\}$. Finally, E is initialised by setting $E(n, a) := \mathcal{A}(L(n), L(a))$ for each $n \in N$, $a \in \mathbf{I}$. A node $m \in N$ is *reachable* if there is some node $n \in N$ such that $E(n, m)$ is defined, and *unreachable* otherwise. Clearly, all nominal nodes are reachable by the initialisation of E. Now as long as there is some unreachable node $x \in \mathsf{Var}(q)$, the algorithm nondeterministically selects one such x and some node $n \in N$ that is either reachable or $\top$, and sets $E(n, x) := \mathcal{A}(L(n), L(x))$. After this procedure, the graph (N, L, E) is such that all nodes other than $\top$ are reachable. Finally, the algorithm checks whether any of the automata $E(n, m)$ ($n, m \in N$) accepts the empty language, and aborts with failure if this is the case.

Checking concept entailment. For all concept atoms $C(n) \in q$, the algorithm checks whether $L(n) \models C$ with respect to KB.

For the remaining steps of the algorithm, some preliminary definitions and observations are needed. The automata $E(n, m)$ of the proof graph represent chains of existential role restrictions that exist within any model. If $m \in \mathsf{Var}(q)$, then the automaton encodes (all possible) ways of constructing an element that belongs to the interpretation of $L(m)$ in each model. The role automata $\mathcal{A}(R)$ in turn encode possible chains of roles that suffice to establish role R along some such path. To show that an atom $R(n, m)$ is entailed, one thus merely has to check whether the automata $E(n, m)$ and $\mathcal{A}(R)$ have a non-empty intersection. Two issues must be taken into account. Firstly, not every pair of nodes is linked via some edge $E(n, m)$, so one might have to look for a longer path and check non-emptiness of its intersection with $\mathcal{A}(R)$. Secondly, there might be many role atoms that affect the path between n and m, and all of them must be satisfied concurrently. Hence, one either needs to check intersections of many automata concurrently, or one needs to retain the restrictions imposed by one (processed) role atom before treating further atoms. The following is easy to prove.

Proposition 2. *For every pair of nodes n, $m \in N$, there either is a unique* shortest *connecting path $n_0 = n, n_1, \ldots, n_k = m$ with $n_i \in N$ and $E(n_i, n_{i+1})$ defined, or there is no connecting path at all. If it exists, this path can be computed by a deterministic algorithm in polynomial time.*

Now any role atom in the query should span over some existing path, and we need to check whether this path suffices to establish the required role. To do this, we nondeterministically split the role automaton into parts that are distributed along the path.

Definition 3. *Consider an NFA $\mathcal{A} = (Q, \Sigma, \delta, i, \{f\})$. A* split *of $\mathcal{A}$ into k parts is given by NFA $\mathcal{A}_1, \ldots, \mathcal{A}_k$ that are constructed as follows. For every $j = 0, \ldots, k$, there is some state $q_j \in Q$ such that $q_0 = i$ and $q_k = f$. The NFA $\mathcal{A}_j$ has the form $(Q, \Sigma, \delta, q_{j-1}, \{q_j\})$.*

It is easy to see that, if each split automaton $\mathcal{A}_j$ accepts some word w_j, we find that $w_1 \ldots w_k$ is accepted by $\mathcal{A}$. Likewise, any word accepted by $\mathcal{A}$ is also accepted in this sense by split of $\mathcal{A}$. Since the combination of any split in general accepts less words than $\mathcal{A}$, splitting an NFA usually involves some don't-know nondeterminism. We can now proceed with the steps of the algorithm.

Splitting of role automata. For each role atom $R(n, m)$ within the query, the algorithm computes the shortest path $n = n_0, \ldots, n_k = m$ from n to m, or aborts with failure if no such path exists. Next, it splits the NFA $\mathcal{A}(R)$ into k automata $\mathcal{A}(R(n, m), n_0, n_1)$,..., $\mathcal{A}(R(n, m), n_{k-1}, n_k)$, and aborts with failure if any of the split automata is empty.

Check role entailment. Finally, for each n, $m \in N$ with $E(n, m)$ defined, the algorithm executes the following checks:

- If $m \in \mathbf{I}$, it checks whether the intersection of the edge automaton $E(n, m)$ with any single split automaton of the form $\mathcal{A}(F, n, m)$ is empty.
- If $m \in \mathsf{Var}(q)$, it checks whether the simultaneous intersection of the edge automaton $E(n, m)$ with all split automata of the form $\mathcal{A}(F, n, m)$ is empty.

If all those intersections have been shown to be non-empty, the algorithm confirms the entailment of the query (we say that the algorithm *accepts* the query). Otherwise it terminates with failure.

Formal proofs of soundness and completeness of the algorithm are given in [11]. Soundness is established by showing that acceptance implies the existence of a match for the query w.r.t. any model of KB. Indeed, a suitable section of the model can be found by retracting the algorithms construction of the proof graph to find suitable domain element, and by noting that the properties that the algorithm has inferred ensure that all conditions imposed by the query are satisfied for this match. For completeness, a canonical model is constructed and this model is used to guide the choices of the algorithm to successful acceptance. Similar to the constructed proof graph, the canonical model exposes a certain local "tree-likeness": while the presence of nominals prevents the model from being a tree, all cycles in the model must involve a named constant (and thus a nominal). This fact is exploited by the algorithm in its construction of shortest paths and allows us to focus on only one unique such path for showing the entailment of all role atoms in the query.

6 Complexity of Query Answering for $\mathcal{EL}^{++}$

Finally, we harvest a number of complexity results from the algorithm of Section 5.

Theorem 3. *The complexities of conjunctive query entailment for regular $\mathcal{EL}^{++}$ knowledge bases – estimated w.r.t. the size of the variable input – are shown in the following table. Whenever the RBox is variable, we assume that it is simple.*

	Variable parts:				
	Query	*RBox*	*TBox*	*ABox*	*Complexity*
Combined complexity	×	×	×	×	PSPACE-*complete*
Query complexity	×				NP-*complete*
Schema complexity		×	×	×	P-*complete*
Data complexity				×	P-*complete*

Proof. The hardness proofs detailed in [11] apply known hardness results for the data-complexity of instance checking in fragments of $\mathcal{EL}$ [13], evaluation of single Datalog clauses (NP-complete, [14]), and emptiness of the intersection of finite automata (PSPACE-complete, [15]). For containment in the respective complexity classes, one carefully estimates complexity boundaries for the algorithm of Section 5. □

We remark that the above results are quite generic, and can be established for many other DLs. Especially, NP-hardness w.r.t. knowledge base size can be shown for any logic that admits an ABox, whereas PSPACE hardness of the combined problem follows whenever the DL additionally admits role composition and existential role restrictions.

7 Conclusion

We have proposed a novel algorithm for answering conjunctive queries in $\mathcal{EL}^{++}$ KBs, which is worst-case optimal under various assumptions. Apparently, this also constitutes the first inference procedure for conjunctive queries in a DL that supports complex

role inclusions (including composition). Showing undecidability of conjunctive queries for unrestricted $\mathcal{EL}^{++}$, we illustrated that combining role atoms in queries and complex role inclusion axioms can make reasoning significantly more difficult.

A compact automata-based representation of role chains *and* (parts of) models allowed us to establish polynomial bounds for inferencing in various cases, thus identifying querying scenarios that are still tractable for $\mathcal{EL}^{++}$. Conjunctive queries inherently introduce some nondeterminism, but automata can conveniently represent sets of possible solutions instead of considering each of them separately. We therefore believe that central methods from the presented algorithm can be a basis for actual implementation that introduces additional heuristics to ameliorate nondeterminism.

References

1. Chandra, A.K., Merlin, P.M.: Optimal implementation of conjunctive queries in relational data bases. In Hopcroft, J.E., Friedman, E.P., Harrison, M.A., eds.: Proc. STOC'77, ACM Press (1977) 77–90
2. Horrocks, I., Sattler, U., Tessaris, S., Tobies, S.: How to decide query containment under constraints using a description logic. In Parigot, M., Voronkov, A., eds.: Proc. LPAR 2000. Volume 1955 of LNAI., Springer (2000) 326–343
3. Hustadt, U., Motik, B., Sattler, U.: A decomposition rule for decision procedures by resolution-based calculi. In: Proc. LPAR 2004. (2005) 21–35
4. Ortiz, M.M., Calvanese, D., Eiter, T.: Data complexity of answering unions of conjunctive queries in $\mathcal{SHIQ}$. In: Proc. DL 2006, CEUR Electronic Workshop Proceedings (2006)
5. Ortiz, M.M., Calvanese, D., Eiter, T.: Characterizing data complexity for conjunctive query answering in expressive description logics. In: Proc. AAAI'06. (2006)
6. Glimm, B., Horrocks, I., Lutz, C., Sattler, U.: Conjunctive query answering for the description logic $\mathcal{SHIQ}$. In: Proc. IJCAI-07, Hyderabad, India (2007)
7. Horrocks, I., Kutz, O., Sattler, U.: The even more irresistible $\mathcal{SROIQ}$. In: Proc. KR2006, AAAI Press (2006) 57–67
8. Baader, F., Brandt, S., Lutz, C.: Pushing the $\mathcal{EL}$ envelope. In: Proc. IJCAI-05, Edinburgh, UK, Morgan-Kaufmann Publishers (2005)
9. Krisnadhi, A., Lutz, C.: Data complexity in the $\mathcal{EL}$ family of DLs. In: Proc. DL 2007, CEUR Electronic Workshop Proceedings (2007)
10. Horrocks, I., Sattler, U.: Decidability of $\mathcal{SHIQ}$ with complex role inclusion axioms. In: Proc. IJCAI-03, Acapulco, Mexico, Morgan-Kaufmann Publishers (2003) 343–348
11. Krötzsch, M., Rudolph, S.: Conjunctive queries for $\mathcal{EL}$ with role composition. Technical report, Universität Karlsruhe (TH), Germany (2007) Available at `http://www.aifb.uni-karlsruhe.de/Publikationen/showPublikation?publ_id=1463`.
12. Wessel, M.: Obstacles on the way to qualitative spatial reasoning with description logics: Some undecidability results. In: Proc. DL 2001. (2001)
13. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Data complexity of query answering in description logics. In: Proc. KR 2006. (2006) 260–270
14. Dantsin, E., Eiter, T., Gottlob, G., Voronkov, A.: Complexity and expressive power of logic programming. ACM Computing Surveys **33** (2001) 374–425
15. Kozen, D.: Lower bounds for natural proof systems. In: Proc. 18th Symp. on the Foundations of Computer Science. (1977) 254–266