Worst-case Optimal Conjunctive Query Answering for an Expressive Description Logic without Inverses

Magdalena Ortiz and **Mantas Šimkus** and **Thomas Eiter**
Institut für Informationssysteme, Technische Universität Wien
Favoritenstraße 9-11, A-1040 Vienna, Austria
(ortiz|simkus|eiter)@kr.tuwien.ac.at

Abstract

Answering conjunctive queries (CQs) has been recognized as a key task for the usage of Description Logics (DLs) in a number of applications, and has thus been studied by many authors. In this paper, we present an algorithm for this problem in the DL $\mathcal{ALCH}$ which works in exponential time. It improves over previous algorithms which require double exponential time and is worst-case optimal, as already satisfiability testing in $\mathcal{ALC}$ is EXPTIME-complete. Furthermore, it shows that inverse roles cause an exponential jump in complexity; as recently shown, the problem is 2EXPTIME-complete for $\mathcal{ALCI}$. The algorithm is based on a technique that compiles knowledge bases into sets of trees of depth 1. It is in CONP under data complexity (i.e., if the taxonomy part and the query are fixed), thus worst-case optimal. An extension from $\mathcal{ALCH}$ to DLs with further constructs is possible.

Introduction

In recent years, Description Logics (DLs) have received increasing importance as formalisms to represent richer domain models in various contexts, including the Semantic Web, data and information integration, peer-to-peer data management, and ontology-based data access. For example, some of the standard Web ontologies from the OWL family are based on DLs (Heflin & Hendler 2001).

The wider use of DLs also requires to provide more reasoning services beyond traditional satisfiability, subsumption and instance checking. In particular, answering conjunctive queries (CQs) over knowledge bases has been recognized as a key task in this respect and studied in many papers, including (Hustadt *et al.* 2005; Glimm *et al.* 2007a,2007b; Lutz 2007; Calvanese *et al.* 1998; 2006; 2007; Krötzsch *et al.* 2007; Ortiz *et al.* 2008; Rosati 2007).

As the problem subsumes testing satisfiability of a knowledge base, well-known results on the complexity of the latter imply that it is at least EXPTIME-hard for any DL including $\mathcal{ALC}$, which is the very core of many DLs like $\mathcal{SHIQ}$, $\mathcal{SRIQ}$ and $\mathcal{DLR}$. As recently shown in (Lutz 2007), answering CQs is 2EXPTIME-hard for all DLs containing $\mathcal{ALCI}$; thus, for the aforementioned DLs, corresponding upper bounds from (Calvanese *et al.* 1998; 2007; Hustadt *et al.* 2005; Glimm *et al.* 2007a) are tight.

Various methods have been used in this context, ranging from adapted tableaux procedures (Levy & Rousset 1998; Ortiz *et al.* 2008) over query incorporation into the knowledge base (Calvanese *et al.* 1998; Tessaris 2001; Glimm *et al.* 2007a,2007b) and resolution techniques (Hustadt *et al.* 2005) to automata-based algorithms (Calvanese *et al.* 2007).

However, up to now, the complexity of answering CQs in $\mathcal{ALC}$ was not precisely known, and it was in particular not clear how much the complexity increases compared to satisfiability testing. We answer this question here. Our main contribution is an algorithm for answering CQs over $\mathcal{ALCH}$ knowledge bases in exponential time, which shows that the problem is not more expensive than satisfiability testing. The algorithm is worst case optimal and improves over previous ones that require double exponential time, confirming Lutz's finding (2007; 2008) that CQ answering in $\mathcal{ALC}$ is feasible in exponential time.

Our algorithm is based on *knots*, borrowed from (Simkus *et al.* 2007), which are schematic trees of depth ≤ 1 that occur in the forest-shaped models of a knowledge base, and has the following features:

- It is CONP under data complexity, i.e., for a fixed TBox and query, it can be nondeterministically run in polynomial time. This is also worst-case optimal, as the data complexity of CQ answering is known to be CONP-complete for a wide range of DLs including $\mathcal{ALC}$; see e.g. (Ortiz *et al.* 2008).
- It provides a modular *knowledge compilation* of the TBox and the query, such that further queries can reuse the TBox compilation. In particular, queries of bounded size can be incorporated into the compilation in polynomial time (w.r.t. the size of the latter), which is specially useful when many queries have to be answered over the same knowledge base.
- It works for non-ground query answering. After the compilation, query answering can be reduced to evaluating a datalog program over a set of facts. This may make the algorithm more amenable for efficient implementation in practice than automata- or tableaux-based algorithms.

While we focus here on $\mathcal{ALCH}$, the method is extendible to richer DLs including $\mathcal{SH}$, and shows that answering large classes of CQs in such logics is not more expensive than satisfiability testing. The knot technique opens an interesting perspective that might be fruitfully exploited for other purposes as well.

Preliminaries

In this section, we introduce $\mathcal{ALCH}$ knowledge bases and define their conjunctive query answering problem.

Syntax. We assume countably infinite sets $\mathbf{C}$, $\mathbf{R}$ and $\mathbf{I}$ of *concept names, roles* names, and *individuals* respectively. $\mathbf{C}$ contains $\top$ and $\bot$. $\mathcal{ALCH}$ *concepts* (or *concepts*) are inductively defined as follows: (a) every concept name $A \in \mathbf{C}$ is a concept, and (b) if C, D are concepts and $R \in \mathbf{R}$ is a role, then $C \sqcap D$, $C \sqcup D$, $\neg C$, $\forall R.C$, $\exists R.C$ are concepts.

Let C, D be concepts, R, S be roles, a, b be individuals, and A be a concept name. An expression $C \sqsubseteq D$ is a *general concept inclusion axiom (GCI)*, an expression $R \sqsubseteq S$ is a *role inclusion axiom (RI)*, and expressions $a{:}A$ and $\langle a, b\rangle{:}R$ are *assertions*. An $\mathcal{ALCH}$ *knowledge base* (KB) is a tuple $\mathcal{K}=\langle \mathcal{T}, \mathcal{A}\rangle$, where the *TBox* $\mathcal{T}$ is a finite set of GCIs and RIs; and the *ABox* $\mathcal{A}$ is a finite nonempty set of assertions. W.l.o.g. we assume that all concept and role names occurring in $\mathcal{A}$, as well as $\top$ and $\bot$, occur in $\mathcal{T}$. Let $\mathbf{C}(\mathcal{K})$, $\mathbf{R}(\mathcal{K})$ and $\mathbf{I}(\mathcal{K})$ respectively denote the sets of concept names, roles, and individuals occurring in $\mathcal{K}$.

Semantics. An *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ for a KB $\mathcal{K}$ consists of a *domain* $\Delta^{\mathcal{I}}$ and a *valuation function* $\cdot^{\mathcal{I}}$ that maps each individual $c \in \mathbf{I}(\mathcal{K})$ to an element $c^{\mathcal{I}} \in \Delta^{\mathcal{I}}$, every concept name $C \in \mathbf{C}(\mathcal{K})$ to a subset $C^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$, and every role $R \in \mathbf{R}(\mathcal{K})$ to a subset $R^{\mathcal{I}}$ of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. The function $\cdot^{\mathcal{I}}$ is extended to all concepts in such a way that it satisfies:

$$\top^{\mathcal{I}} = \Delta^{\mathcal{I}}, \quad \bot^{\mathcal{I}} = \emptyset, \quad (\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}},$$
$$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}, \quad (C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}},$$
$$(\exists R.C)^{\mathcal{I}} = \{x \mid \exists y.\langle x, y\rangle \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\},$$
$$(\forall R.C)^{\mathcal{I}} = \{x \mid \forall y.\langle x, y\rangle \in R^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\}.$$

An interpretation $\mathcal{I}$ is a *model* of $\mathcal{K} = \langle \mathcal{T}, \mathcal{A}\rangle$, if (i) for each GCI $C \sqsubseteq D$ in $\mathcal{T}$, $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$; (ii) for each RI $R \sqsubseteq S$ in $\mathcal{T}$, $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$; (iii) for each assertion $a{:}A$ in $\mathcal{A}$, $a^{\mathcal{I}} \in A^{\mathcal{I}}$; and (iv) for each assertion $\langle a, b\rangle{:}R$ in $\mathcal{A}$, $\langle a^{\mathcal{I}}, b^{\mathcal{I}}\rangle \in R^{\mathcal{I}}$.

Conjunctive Query Answering. Let $\mathbf{V}$ be a countably infinite set of variables. A *conjunctive query* (CQ, or *query*) over a KB $\mathcal{K}$ is a finite set of atoms of the form $A(x)$ or $R(x, y)$, where $A \in \mathbf{C}(\mathcal{K})$, $R \in \mathbf{R}(\mathcal{K})$ and $x, y \in \mathbf{V}$.[1] By $\mathbf{V}(q)$ we denote the variables occurring in the atoms of q. A query q is associated with a unique (possibly empty) tuple $\vec{x}=\langle x_1, \ldots, x_n\rangle$ of *answer* variables from $\mathbf{V}(q)$

A *match for q in an interpretation $\mathcal{I}$ for $\mathcal{K}$* is a mapping θ from $\mathbf{V}(q)$ to $\Delta^{\mathcal{I}}$ such that (i) $\theta(x) \in A^{\mathcal{I}}$ for each $A(x) \in q$, and (ii) $\langle \theta(x), \theta(y)\rangle \in R^{\mathcal{I}}$ for each $R(x, y) \in q$. A tuple $\langle c_1, \ldots, c_n\rangle$ of individuals from $\mathbf{I}(\mathcal{K})$ *is an answer of q over $\mathcal{I}$*, if $\langle {c_1}^{\mathcal{I}}, \ldots, {c_n}^{\mathcal{I}}\rangle = \langle \theta(x_1), \ldots, \theta(x_n)\rangle$ for some match θ for q in $\mathcal{I}$; $\mathsf{ans}(q, \mathcal{I})$ denotes all answers of q over $\mathcal{I}$. The *answer* of q over $\mathcal{K}$, denoted $\mathsf{ans}(q, \mathcal{K})$, is the set of all tuples $\vec{c}$ such that $\vec{c} \in \mathsf{ans}(q, \mathcal{I})$ for every model $\mathcal{I}$ of $\mathcal{K}$.

Normal Knowledge Bases

We focus in this paper on normalized KBs and a restricted class of their models, which correspond to the minimal Herbrand models of the theory obtained by the usual translation of $\mathcal{ALCH}$ into first-order logic and skolemization.

Definition 1. A KB $\mathcal{K}$ is *normal*, if all its GCIs are of the form (D) $A_0 \sqcap \ldots \sqcap A_n \sqsubseteq B_0 \sqcup \ldots \sqcup B_m$, (E) $A_0 \sqsubseteq \exists R.B_0$, or (U) $A_0 \sqsubseteq \forall R.B_0$, where each $A_i, B_j \in \mathbf{C}$, and $n, m > 0$.

For a normal KB $\mathcal{K}$, its *Herbrand universe* $\mathcal{U}_{\mathcal{K}}$ is the set of all *terms* inductively defined as follows: (i) each $c \in \mathbf{I}(\mathcal{K})$ is a term, and (ii) if t is a term and α is a GCI of type (E) occurring in $\mathcal{K}$, then $f_\alpha(t)$ is a term. Let $\mathcal{B}_{\mathcal{K}}$ be the set of all expressions $C(s)$ and $R(s, t)$ with $C \in \mathbf{C}(\mathcal{K})$, $R \in \mathbf{R}(\mathcal{K})$, and $s, t \in \mathcal{U}_{\mathcal{K}}$, which we call *atoms*.

An *Herbrand interpretation* of $\mathcal{K}$ is any set $I \subseteq \mathcal{B}_{\mathcal{K}}$; it represents the interpretation $\mathcal{I}$ with $\Delta^{\mathcal{I}} = \mathcal{U}_{\mathcal{K}}$, $C^{\mathcal{I}} = \{d \mid C(d) \in I\}$, $R^{\mathcal{I}} = \{\langle c, d\rangle \mid R(c, d) \in I\}$ and $c^{\mathcal{I}} = c$ for each $c \in \mathbf{I}(\mathcal{K})$. Such an I is an *S-Herbrand model* of $\mathcal{K}$, if it is a model of $\mathcal{K}$, and for each $\alpha = A \sqsubseteq \exists R.B$ in $\mathcal{K}$, $A(t) \in I$ implies $R(t, f_\alpha(t)) \in I$ and $B(f_\alpha(t)) \in I$. Moreover, I is a *minimal S-Herbrand* model of $\mathcal{K}$, if no $J \subset I$ is an S-Herbrand model of $\mathcal{K}$. We denote by $\mathcal{M}(\mathcal{K})$ the set of all minimal S-Herbrand models of $\mathcal{K}$.

Using well-known structural transformations, every KB $\mathcal{K}$ can be transformed in linear time into a normal KB $\mathcal{K}'$ in a way that preserves query answers. Moreover, one can show via standard first-order logic that, to answer a query over $\mathcal{K}$, it is sufficient to consider its minimal S-Herbrand models.

Theorem 1. *For any $\mathcal{ALCH}$ KB $\mathcal{K}$ and a CQ q, we can obtain in linear time a normal $\mathcal{ALCH}$ KB $\mathcal{K}'$ s.t.* $\mathsf{ans}(q, \mathcal{K}) = \mathsf{ans}(q, \mathcal{K}')$. *Moreover,* $\mathsf{ans}(q, \mathcal{K}') = \bigcap_{I \in \mathcal{M}(\mathcal{K}')} \mathsf{ans}(q, I)$.

In the following, unless stated otherwise, by 'interpretation' we mean Herbrand interpretation, and by '(minimal) model' we mean (minimal) S-Herbrand model.

Example 1. As running example, we consider the normal KBs $\mathcal{K}_1=\langle \mathcal{A}, \mathcal{T}_1\rangle$ and $\mathcal{K}_2=\langle \mathcal{A}, \mathcal{T}_2\rangle$, where $\mathcal{A}=\{a{:}D\}$, $\mathcal{T}_1=\{\alpha_0, \alpha_1, \alpha_2, \alpha_3, \alpha_4\}$, $\mathcal{T}_2=\{\alpha'_0, \alpha_1, \alpha_2, \alpha_3, \alpha_4\}$ and:

$$\alpha_0 = D \sqsubseteq A \sqcup B \quad \alpha_1 = B \sqsubseteq \exists P.A \quad \alpha_3 = A \sqsubseteq \exists Q.\top$$
$$\alpha'_0 = D \sqsubseteq B \quad \alpha_2 = B \sqsubseteq \exists P.C \quad \alpha_4 = C \sqsubseteq \exists P.D$$

We also consider the following CQ:

$q_1(x, x')=\{B(x), P(x, y), A(y), P(x', y), Q(y, z)\}$.

Note that $\mathsf{ans}(q_1, \mathcal{K}_1) = \emptyset$ and $\mathsf{ans}(q_1, \mathcal{K}_2) = \{\langle a, a\rangle\}$.

Knots

In this section, we exploit the model-theoretic properties of normal KBs to provide a method for finitely representing their possibly infinite minimal models. Of particular importance is the *forest-shaped model property*, on the basis of which, each minimal model of a normal KB can be viewed as a graph and a set of trees rooted at nodes of the graph. In what follows, let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A}\rangle$ be an arbitrary normal KB.

Proposition 1. *An interpretation of $\mathcal{K}$ is* forest-shaped, *if its binary atoms are of the form $R(a, b)$ or $R(t, f(t))$ for a term t and individuals a, b. Each $I \in \mathcal{M}(\mathcal{K})$ is forest-shaped.*

The key intuition underlying the finite representation is that minimal models of $\mathcal{K}$ can be composed out of trees of depth ≤ 1 that we call *knots*. For convenience of presentation, we write $t \hat{\in} I$ if a set of atoms I contains an atom with the term t as argument, and denote by $\mathcal{U}(I)$ the set of all terms $t \hat{\in} I$.

[1] Note that no individuals occur in q. This is no limitation, as for any constant a we can use a new concept name C_a, replace a in q by a new variable y, and add $C_a(y)$ to q and $a : C_a$ to $\mathcal{A}$.

Definition 2. (Knots) A *knot with root (term)* t is a set of atoms K such that each atom in K is of the form $A(t)$, $R(t, f(t))$, or $A(f(t))$ where A, R, and f are arbitrary. Let $\mathsf{succ}(K)$ denote the set of terms of the form $f(t)\hat{\in}K$.

A knot with root term t can be viewed as a labeled tree of depth at most 1, where the nodes and edges are labeled with concept names and roles respectively. In the following we only consider the knots K where each concept name and role occurring in K also occurs in $\mathcal{K}$ (t need not be from $\mathcal{U}_\mathcal{K}$). For a term t, let $\mathcal{B}_t$ denote the set of all atoms that can be built from concept names and roles of $\mathcal{K}$ using t and terms of the form $f(t)$ as arguments. Note that for a forest-shaped interpretation I for $\mathcal{K}$ and $t\hat{\in}I$, the set $I \cap \mathcal{B}_t$ is a knot.

We introduce *min-consistent* knots, which are self-contained model building blocks for minimal models of $\mathcal{K}$.

Definition 3. Given a knot K with root t, we say K is *consistent* (w.r.t. $\mathcal{K}$), if the following are satisfied:

(a) $\top(u)\in K$ and $\bot(u)\notin K$ for all $u\in\{t\}\cup\{f(t)|f(t)\in\mathcal{U}_\mathcal{K}\}$;
(b) if $A \sqsubseteq \forall R.B \in \mathcal{T}$, $A(t) \in K$ and $R(t, f(t)) \in K$, then $B(f(t)) \in K$;
(c) if $\alpha = A \sqsubseteq \exists R.B$, $\alpha\in\mathcal{T}$ and $A(t) \in K$, then $R(t, f_\alpha(t)) \in K$ and $B(f_\alpha(t)) \in K$;
(d) if $A_0 \sqcap \ldots \sqcap A_n \sqsubseteq B_0 \sqcup \ldots \sqcup B_m \in \mathcal{T}$, $s \in \mathsf{succ}(K)$ and $\{A_0(s), \ldots, A_n(s)\} \subseteq K$, then $B_i(s)\in K$ for some B_i;
(e) if $R\sqsubseteq S \in \mathcal{T}$ and $R(t, f(t))\in K$, then $S(t, f(t))\in K$.

K is *min-consistent* if each $K'\subset K$ obtained from K by removing atoms where an $s\in\mathsf{succ}(K)$ occurs is inconsistent.

Intuitively, given a term t and a set of concepts it satisfies, a min-consistent knot with root t encodes a possible combination of immediate successors for t in a model of $\mathcal{K}$.

Example 2. To simplify the notation, for $1\leq i\leq 4$, we write f_i instead of f_{α_i}, and we omit the atom $\top(u)$ for each u. The knots K_a=$\{C(t), P(t, f_4(t)), A(f_4(t)), D(f_4(t))\}$, K_b=$\{C(t), P(t, f_4(t)), B(f_4(t)), D(f_4(t))\}$ are both min-consistent w.r.t. $\mathcal{K}_1$; and only K_b is min-consistent w.r.t. $\mathcal{K}_2$.

After the necessary notions for dealing with the tree-part of forest-shaped interpretations, we deal with the graph part.

Definition 4. The KB $\mathcal{K}^g$ is obtained from $\mathcal{K}$ by deleting all axioms of type (E). Each $G\in\mathcal{M}(\mathcal{K}^g)$ is a *min-graph* of $\mathcal{K}$.

Example 3. G_1=$\{D(a), B(a)\}$ and G_2=$\{D(a), A(a)\}$ are the min-graphs of $\mathcal{K}_1$, while G_1 is the only min-graph of $\mathcal{K}_2$.

The next theorem characterizes the minimal models of $\mathcal{K}$.

Theorem 2. *Let I be an interpretation for $\mathcal{K}$, and let I^g be the set of all atoms in I of the form $A(a)$, $R(a, b)$ for any individuals a, b. Then $I\in\mathcal{M}(\mathcal{K})$ iff I is forest-shaped, I^g is a min-graph of $\mathcal{K}$, and for each term $t\hat{\in}I$, the knot $I \cap \mathcal{B}_t$ is min-consistent w.r.t. $\mathcal{K}$.*

The characterization above allows to view minimal models as being constructed of knots, each of which is min-consistent, and to provide a finite representation of these models. Roughly, the representation relies on the observation that although infinitely many knots might occur in some minimal model of a KB, only finitely many of them are non-isomorphic modulo the root term.

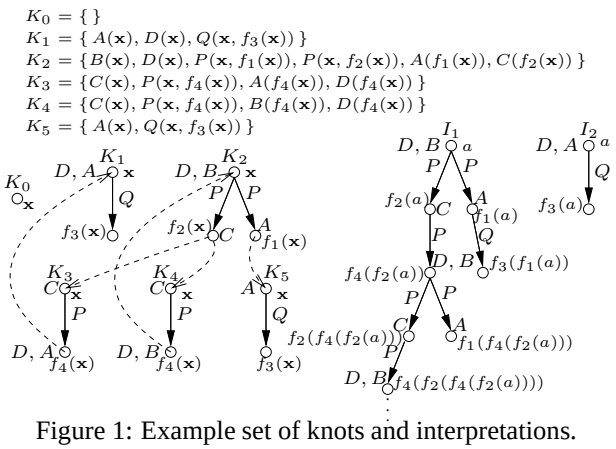

Figure 1: Example set of knots and interpretations.

Definition 5. Let $\mathbf{x}$ be an individual not occurring in any $\mathcal{ALCH}$ KB. A knot K with root t is *abstract*, if $t = \mathbf{x}$. A knot K' with root u is an *instance* of K, if K' can be obtained from K by replacing each occurrence of $\mathbf{x}$ with u.

Note that min-consistency is invariant under such substitutions, i.e., if K' is an instance of K, then K' is min-consistent iff K is min-consistent. We show next that a set of abstract knots can be used to represent all the minimal models. Given two sets of atoms I and J, we write $I_t \approx J_u$ if $\{A \mid A(t) \in I\} = \{A \mid A(u) \in J\}$.

Definition 6. Let L be a set of abstract knots. Given $K\in L$ and $s\in\mathsf{succ}(K)$, we say $K'\in L$ *is an s-successor of* K, if $K_s\approx K'_\mathbf{x}$. Let $L[K, s]$ be the set of s-successors of K in L. Then L is *$\mathcal{K}$-founded*, if for each $K\in L$, K is minimal-consistent w.r.t. $\mathcal{K}$, and $L[K, s]\neq\emptyset$ for each $s \in \mathsf{succ}(K)$.

Example 4. Figure 1 shows the abstract knots $K_0, \ldots, K_5$. Note that K_a and K_b are instances of K_3 and K_4 respectively. The set L_1=$\{K_0, \ldots, K_5\}$ is $\mathcal{K}_1$-founded, while L_2=$\{K_0, K_2, K_4, K_5\}$ is $\mathcal{K}_2$-founded. In the graphic, the dotted arcs connect a node s in K to $\mathbf{x}$ in K' whenever $K_s \approx K'_\mathbf{x}$; the links to $\mathbf{x}$ in K_0 are omitted.

In what follows, we provide a construction of minimal models out of knots in a founded set. Moreover, we show that for a given KB there exists a founded set of knots that captures all the minimal models.

Definition 7. We say *an interpretation I is generated* by a $\mathcal{K}$-founded knot set L if I is a $\subseteq$-minimal interpretation containing some min-graph G of $\mathcal{K}$ and for each term $t\hat{\in}I$, $I \cap \mathcal{B}_t$ is an instance of some $K \in L$. By $\mathcal{F}_\mathcal{K}(L)$ we denote the set of interpretations generated by L.

Intuitively, $\mathcal{F}_\mathcal{K}(L)$ represents all the forest-shaped interpretations that can be built from from some min-graph by instantiating the knots in L. Importantly, such interpretations are actually minimal models. Indeed, due to Theorem 2, if L is $\mathcal{K}$-founded and $I\in\mathcal{F}_\mathcal{K}(L)$, then $I\in\mathcal{M}(\mathcal{K})$.

Definition 8. Let $\mathbb{K}_\mathcal{K}$ denote the smallest set of abstract knots which contains each $\mathcal{K}$-founded set of knots.

The crucial property of $\mathbb{K}_\mathcal{K}$ is that it captures the tree-structures of the minimal models of $\mathcal{K}$, and together with the min-graphs, it captures all the minimal models of $\mathcal{K}$.

Theorem 3. *$\mathbb{K}_\mathcal{K}$ is $\mathcal{K}$-founded, and $\mathcal{F}_\mathcal{K}(\mathbb{K}_\mathcal{K}) = \mathcal{M}(\mathcal{K})$.*

Example 5. Clearly $L_1 \subseteq \mathbb{K}_{\mathcal{K}_1}$ and $L_2 \subseteq \mathbb{K}_{\mathcal{K}_2}$. For I_1, I_2 as given in the figure, $I_1, I_2 \in \mathcal{F}_{\mathcal{K}_1}(\mathbb{K}_{\mathcal{K}_1}) = \mathcal{M}(\mathcal{K}_1)$. Also $I_1 \in \mathcal{F}_{\mathcal{K}_2}(\mathbb{K}_{\mathcal{K}_2}) = \mathcal{M}(\mathcal{K}_2)$, which is not the case for I_2. In fact, each $I \in \mathcal{F}_{\mathcal{K}_2}(\mathbb{K}_{\mathcal{K}_2})$ starts with an instance of K_2 and contains infinitely many instances of each knot in L_2.

We can derive $\mathbb{K}_\mathcal{K}$ in two steps. First, the set $A_\mathcal{K}$ of all knots that are min-consistent w.r.t. $\mathcal{K}$ is constructed. This can be done, for instance, by exhaustively traversing each atom set $H = \{A_1(\mathbf{x}), \ldots, A_n(\mathbf{x})\}$, where each A_i is a concept name from $\mathcal{K}$, and extending H to become consistent w.r.t. $\mathcal{K}$ by adding necessary atoms to satisfy the conditions in Def. 3. Min-consistency of the resulting knot can be verified by considering its proper subsets. In the second step, $A_\mathcal{K}$ is cleaned to ensure foundedness according Def. 6, by exhaustively rewriting the set $L := A_\mathcal{K}$ with the following rule: if $L[K, s] = \emptyset$ for some $K \in L$ and $s \in \mathsf{succ}(K)$, then remove K from L. The resulting set L is $\mathbb{K}_\mathcal{K}$.

Query Answering with Knots

In what follows, we assume a fixed $\mathcal{K}$-founded set of knots L and a fixed CQ q, and describe a method for computing the answers of q in all the interpretations in $\mathcal{F}_\mathcal{K}(L)$. Using the results in the preceding section and setting $L = \mathbb{K}_\mathcal{K}$, this allows us to compute $\mathsf{ans}(q, \mathcal{K})$.

The idea is to decompose the problem. We first define the entailment of *subqueries at a knot* K, which informally means that there is a match for some parts of the query in each *tree* that is generated from L and starts with K, and provide a decision procedure for it. The method is based on a fixpoint computation that derives in each iteration new pairs of knots and subqueries for which the entailment relation holds, based on previously computed pairs. For a given knot K, the decision on whether it entails a subquery is made by looking at the subqueries that the possible successors knots of K entail. Hence, the algorithm "back-propagates" the information via the possible successor relation. The entailment of subqueries at the knots in L is the key to answering the whole query q over $\mathcal{K}$.

In a second stage, we consider the min-graphs of $\mathcal{K}$ and verify whether for each min-graph G, the query can be mapped in each forest-shaped minimal model of $\mathcal{K}$ that is built from G and the knots in $\mathbb{K}_\mathcal{K}$. To this end, we verify whether, for any possible way of constructing a model out of G, a mapping for the full query can be composed from some partial mapping of q into G and some mappings that exist in the trees rooted at the individuals. The existence of the latter mappings will be witnessed by the precomputed set of all pairs of knots and subqueries for which the entailment relation holds.

Rooted Subquery Entailment

Observe that, since the minimal models of $\mathcal{K}$ are forest-shaped, for any query match π and any tree-shaped part I of a model, the the image under π of the subquery of q that is mapped inside I is a subtree of I. This implies, for example, that if there are atoms of the form $R(x, y)$, $R'(x', y)$ in q, then x and x' must be mapped to the same node in the tree. Whenever a query match maps a variable x to a node in a tree, x induces a set of variables V_x which have to be mapped inside the tree defined by the image of the query graph under the match. This set contains x, the successors of x, and all variables that must be mapped to the same node as one of them because they have a common successor.

Definition 9. Assume a variable $x \in \mathbf{V}(q)$ and a variable set $X \subseteq \mathbf{V}(q)$. Let $\mathsf{R}_0(x) = \{x\}$ and

$$\mathsf{R}_{n+1}(x) = \{y \in \mathbf{V}(q) \mid R(x', y) \in q \text{ and } x' \in \mathsf{R}_n(x)\},$$

for every $n \geq 0$. Then by V_x we denote the set of all $y \in \mathbf{V}(q)$ such that $\mathsf{R}_n(x) \cap \mathsf{R}_m(y) \neq \emptyset$ for some $m \leq n$.

We also denote $\mathsf{R}_1(x)$ by *next*(x), and define *prev*$(x) = \{y \in \mathbf{V}(q) \mid x \in$ *next*$(y)\}$. Moreover, we define *next*$(X) = \bigcup_{x \in X}$ *next*(n) and *prev*$(X) = \bigcup_{x \in X}$ *prev*(n).

We are ready to formally define the subqueries and their entailment in trees.

Definition 10. For a knot $K \in L$, *I is a tree generated by L (starting with K)*, if I is a $\subseteq$-minimal set of atoms such that $K \subseteq I$ and, for each term $t \hat{\in} I$, $I \cap \mathcal{B}_t$ is an instance of some $K' \in L$. We denote by $\mathcal{T}(L, K)$ the set of all such trees.

Definition 11. A *root set of* q is any set $\rho_q \subseteq \mathbf{V}(q)$. By $\mathbb{R}_q$ we denote the set of all root sets of q.

For a tree I generated by L, a *rooted match for $x \in \rho_q$ in* I is a function π from V_x to $\mathcal{U}(I)$ s.t. for each $y, z \in \mathsf{V}_x$:

(RM1) if $A(y) \in q$ then $A(\pi(y)) \in I$;
(RM2) if $R(y, z) \in q$ then $R(\pi(y), \pi(z)) \in I$; and
(RM3) if $y \in \mathsf{V}_x$ and *prev*$(y) \not\subseteq \mathsf{V}_x$, then $\pi(y) = \mathbf{x}$.

We write $I \models \rho_q$ if for some $x \in \rho_q$ there exists a rooted match in I. Further, $I \models^d \rho_q$ holds if for some $x \in \rho_q$ there exists a rooted π match in I s.t. for each $y \in \mathsf{V}_x$, the depth of the term $\pi(y)$ is less or equal to d. We write $K \models_L \rho_q$ (resp., $K \models^d_L \rho_q$) if for each $I \in \mathcal{T}(L, K)$ we have $I \models \rho_q$ (resp., $I \models^d \rho_q$). We omit the subscripts if clear from the context.

Intuitively, a rooted match for x in a tree I is a homomorphic embedding into the tree I of the subquery of q obtained by restricting it to atoms whose variables are in V_x. Additionally, every variable y in V_x that has some predecessor variable not in V_x must be mapped to the root of I (RM3). A rooted match for x may be part of a full query mapping in some model containing I, provided that the predecessors of y have a match above the root of I.

Example 6. $\mathsf{V}_z = \{z\}$, and $\pi(z) = \mathbf{x}$ is a rooted match for z in every $I \in \mathcal{T}(L, K)$ for every $K \in L^1_1$. Hence $K \models \rho$ for each ρ with $z \in \rho$. Since $\mathsf{V}_x = \mathsf{V}_{x'} = \{x, x', y, z\}$, $I \models \{x\}$ iff $I \models \{x'\}$ for any tree I.

In the following, we construct a set $\Gamma(L, q)$ of all pairs (K, ρ) such that $K \models_L \rho$. We first compute the pairs (K, ρ) with $K \models^0_L \rho$, and then continue via fixpoint iteration to obtain the pairs (K, ρ) with $K \models^d_L \rho$ for an arbitrary $d \in \omega$. Such pairs capture the $\models_L$ relation due to the following.

Proposition 2. *If $K \models_L \rho$, then there exists $d \in \omega$ such that $K \models^d_L \rho$.*

A key part of the algorithm is to establish, using the pairs (K, ρ) with $K \models^d_L \rho$, the sets of variables that have a match within depth d in the different trees starting at K. This will be crucial for obtaining the pairs (K', ρ') with $K' \models^{d+1}_L \rho'$.

Specifically, for each K, we characterize the minimal sets of variables l such that each tree starting at K models necessarily the variables in l. To this aim, we use *minimal hitting sets*. Informally, we can see each set l as a most general way of 'grouping' the trees starting at K by the exact set of variables for which they provide a match of bounded depth.

Definition 12. Assume a knot $K \in L$ and a set $S \subseteq L \times \mathbb{R}_q$. A set of variables $l \subseteq \mathbf{V}(q)$ is a *minimal hitting set of S w.r.t. K* if it is a $\subseteq$-minimal set s.t. $l \cap \rho \neq \emptyset$ for every $(K, \rho) \in S$.

Proposition 3. *Assume $K \in L$, $d \in \omega$ and let S be the set of all tuples (K, ρ) such that $K \models^d_L \rho$ for some ρ. If l is a minimal hitting set of S w.r.t. K, then there is some $I \in \mathcal{T}(L, K)$ such that, for every $x \in \mathbf{V}(q)$, $I \models^d \{x\}$ iff $x \in l$.*

We now sketch the procedure for computing $\models_L$. For each $d \in \omega$, let S^d denote the set of all pairs (K, ρ) such that $K \models^d_L \rho$. As easily seen, in case $d = 0$, the set S^0 can be computed by checking which root sets can be satisfied by direct mappings into the roots of the knots in L.

For the inductive case, suppose for some $d \in \omega$ we have computed the set S^d. Assume some ρ and an arbitrary knot $K \in L$. To verify whether $K \models^{d+1}_L \rho$, we consider *K-hits* which capture the possible ways of choosing for each $s \in \mathsf{succ}(K)$ a knot $K' \in L[K, s]$ and a minimal hitting set l of S^d w.r.t. K'. Intuitively, we conclude $K \models^{d+1}_L \rho$ if for each K-hit there is a variable $x \in \rho$ such that part of V_x can be mapped into K, while the rest of the variables are contained in the chosen minimal hitting sets; this partitioning of V_x will be captured by the notion of *K-mapping*.

Definition 13. A *successor choice* for $K \in L$ is a function that maps each $s \in \mathsf{succ}(K)$ to some $K' \in L[K, s]$. A *K-hit of $S \subseteq L \times \mathbb{R}_q$* is a pair (sc, hs), where *sc* is a successor choice for K, and *hs* is function that maps each $s \in \mathsf{succ}(K)$ to a minimal hitting set of S w.r.t. the knot $sc(s)$.

Example 7. If $S = \{(K_0, \{z\})\}$, then $\{z\}$ is the only minimal hitting set of S w.r.t. K_0. Also, (sc, hs) is a K_1-hit of S, where $sc(f(\mathbf{x})) = K_0$ and $hs(f(\mathbf{x})) = \{z\}$ for every f.

Now we introduce K-mappings which are composed of a set r of variables and a function $b(\cdot)$ that maps variables to leaves of K; the variables in r have a match at the *root* of K, while the variables captured by b have a match *below* the root of K. Intuitively, in order for a K-mapping to represent a rooted match in a tree starting with K, each x in the domain of b must have a match in the subtree with root $b(x)$. In particular, the latter holds whenever each such x is in the hitting set $hs(b(x))$ of some K-hit; if this is the case, we say that the K-hit *complies* with the K-mapping.

For a (partial) function g from A to B, its domain is denoted $\mathsf{dom}(g)$. As usual, $g^{-1}(b) = \{a \in A \mid g(a) = b\}$.

Definition 14. For a knot $K \in L$, a *K-mapping for q* is a tuple $m = \langle r, b \rangle$, where $r \subseteq \mathbf{V}(q)$, b is a partial function from $\mathbf{V}(q)$ to $\mathsf{succ}(K)$, $r \cap \mathsf{dom}(b) = \emptyset$, and the following hold:

- $x \in r$ and $A(x) \in q$ imply $A(\mathbf{x}) \in K$;
- $x \in r$ and $R(x, y) \in q$ imply $y \in \mathsf{dom}(b)$ and $R(\mathbf{x}, b(y)) \in K$;
- for each $s \in \mathsf{succ}(K)$, $prev(b^{-1}(s)) \subseteq r \cup b^{-1}(s)$; and
- for each $s \in \mathsf{succ}(K)$, $next(b^{-1}(s)) \subseteq b^{-1}(s)$.

We define $roots(m)$ as the set of all $x \in r \cup \mathsf{dom}(b)$ s.t. $x \in r$ or $prev(x) = \emptyset$. A K-hit (sc, hs) *complies with* m, if for each $s \in \mathsf{succ}(K)$, $b^{-1}(s) \subseteq \mathsf{V}_v$, where $\mathsf{V}_v = \bigcup_{x \in hs(s)} \mathsf{V}_x$.

Example 8. $m_0 = \langle \{z\}, \emptyset \rangle$ is a K-mapping for any $K \in L^1_1$, and it complies with any K-hit; $m_1 = \langle \{y\}, \{z \mapsto f_3(\mathbf{x})\} \rangle$ is a K_1-mapping and a K_5-mapping, it complies with $\{z\}$; $m_2 = \langle \{x, x'\}, \{y \mapsto f_1(\mathbf{x}), z \mapsto f_1(\mathbf{x})\} \rangle$ and $m_3 = \langle \{x, x'\}, \{y \mapsto f_2(\mathbf{x}), z \mapsto f_2(\mathbf{x})\} \rangle$ are K_2-mappings.

We are ready to define a relation $S \vdash_{L,q} (K, \rho)$ for obtaining new pairs (K, ρ) such that $K \models \rho$, which follow from a given set S of pairs (K', ρ') such that $K' \models \rho'$.

Definition 15. Assume $K \in L$, $\rho \in \mathbb{R}_q$ and a set $S \subseteq L \times \mathbb{R}_q$. The pair (K, ρ) *follows from* S, in symbols $S \vdash_{L,q} (K, \rho)$, if for every K-hit k of S there is a K-mapping m such that $\rho \cap roots(m) \neq \emptyset$ and k complies with m.

Next, using the consequence relation above we construct a set $\Gamma(L, q)$ to capture all pairs (K, ρ) such that $K \models_L \rho$.

Definition 16. We define the following sets:

$$\Gamma(L, q)^0 = \{(K, \rho) \in L \times \mathbb{R}_q \mid \emptyset \vdash_{L,q} (K, \rho)\}, \text{ and}$$

$$\Gamma(L, q)^{d+1} = \{(K, \rho) \in L \times \mathbb{R}_q \mid \Gamma(L, q)^d \vdash_{L,q} (K, \rho)\},$$

for each $d > 0$. Then $\Gamma(L, q) = \bigcup_{d \in \omega} \Gamma(L, q)^d$.

Note that $\Gamma(L, q)^0 \subseteq \Gamma(L, q)^1 \subseteq \cdots \subseteq \Gamma(L, q)^d$ for each $d \in \omega$. Since $L \times \mathbb{R}_q$ is finite, $\Gamma(L, q)$ is finite and unique.

Example 9. For simplicity, we use $\vdash_1$ instead of $\vdash_{L^1_1, q_1}$ and $\vdash_2$ instead of $\vdash_{L^2_1, q_1}$. The following hold:

$$\begin{array}{rr} \emptyset \vdash_1 (K_0, \{z\}) & \emptyset \vdash_2 (K_0, \{z\}) \\ \{(K_0, \{z\})\} \vdash_1 (K_1, \{y\}) & \{(K_0, \{z\})\} \vdash_2 (K_5, \{y\}) \\ \{(K_0, \{z\})\} \vdash_1 (K_5, \{y\}) & \{(K_5, \{y\})\} \vdash_2 (K_2, \{x\}) \\ \{(K_5, \{y\})\} \vdash_1 (K_2, \{x\}) & \{(K_2, \{x\})\} \vdash_2 (K_4, \{x\}) \\ \{(K_2, \{x\})\} \vdash_1 (K_4, \{x\}) & \end{array}$$

Thus we have $\{(K_0, \{z\}), (K_1, \{y\}), (K_5, \{y\}), (K_2, \{x\}), (K_4, \{x\})\} \subseteq \Gamma(L^1_1, q_1)$; $\{(K_0, \{z\}), (K_5, \{y\}), (K_2, \{x\}), (K_4, \{x\})\} \subseteq \Gamma(L^2_1, q_1)$.

As intended, $\Gamma(L, q)$ captures the $\models_L$ relation.

Theorem 4. *For each pair $(K, \rho) \in L \times \mathbb{R}_q$, $K \models_L \rho$ iff $(K, \rho) \in \Gamma(L, q)$.*

Proof. (Sketch) Due to Proposition 3 and the definition of the $\models_L$ relation, it suffices to show that for each $d \in \omega$, $K \models^d_L \rho$ iff $(K, \rho) \in \Gamma(L, q)^d$. The latter is proved by induction on $d \in \omega$. As easily verified, the claim holds in case $d = 0$. Suppose the claim holds for an arbitrary $d \in \omega$.

For the soundness, assume a pair $(K, \rho) \in \Gamma(L, q)^{d+1}$. Due to the definition of the $\vdash_{L,q}$ relation, it can be shown that for an arbitrary $I \in \mathcal{T}(L, K)$, there exists a variable $x \in \rho$ for which a rooted match can be composed from a K-mapping m with $x \in roots(m)$ and rooted matches for the variables in the hitting sets of a complying K-hit; such matches exist due to the induction hypothesis.

For the completeness, assume a pair (K, ρ) such that $K \models_L^{d+1} \rho$. To show that $(K, \rho) \in \Gamma(L, q)^{d+1}$, take an arbitrary K-hit (sc, hs) of $\Gamma(L, q)^d$. For each $s \in \mathsf{succ}(K)$, build a tree I_s starting at $sc(s)$ which has rooted matches only for the variables in $hs(s)$. Take a tree I starting at K such that each sub-tree rooted at each leaf s of K coincides with the constructed tree I_s (up to the renaming of terms). Since $K \models_L^{d+1} \rho$, there exists a (bounded) rooted match π for some variable in ρ. Due to the construction of I, π has to map some variables into K, while the rest of variables that it maps are captured by minimal hitting sets. This rooted match witnesses the existence of a K-mapping that complies with the arbitrarily assumed K-hit (sc, hs). □

Query Answering over the Full Knowledge Base

Now that we have the machinery to decide subquery entailment, we move to query answering over $\mathcal{K}$. In what follows, we assume the query q has answer variables $\vec{x}$, while $\vec{c}$ is a tuple of individuals with the same arity as $\vec{x}$.

To answer q, we use the min-graphs of $\mathcal{K}$ and the knot set $\mathbb{K}_\mathcal{K}$. By Theorems 1 and 3, the models constructed from the min-graphs of $\mathcal{K}$ and the knots in $\mathbb{K}_\mathcal{K}$ suffice to provide query answers. The previously introduced machinery deals with the parts of query matches that occur inside the trees generated from $\mathbb{K}_\mathcal{K}$. Next, we extend this machinery to deal with the graph part of the forest-shaped minimal models of $\mathcal{K}$.

Definition 17. An extended min-graph H of $\mathcal{K}$ is a $\subseteq$-minimal set of atoms containing a min-graph of $\mathcal{K}$ and s.t. for each individual a, $H \cap \mathcal{B}_a$ is an instance of a knot in $\mathbb{K}_\mathcal{K}$.

Consider a min-graph G. Intuitively, each extended min-graph containing G can be viewed as a "super" knot whose root is G, while its leaves are the leaves of the knots that extend G. Given this similarity, the full query q can be answered by adjusting the notions of K-hits and K-mappings to deal with extended min-graphs.

Definition 18. Let H be an extended min-graph. A *successor choice* for H is a function that maps each term $f(c)\hat{\in}H$ to some $K \in L$ such that $H_{f(c)} \approx K_\mathbf{x}$. Then an *$H$-hit* is a pair (sc, hs), where sc is a successor choice for H, and hs is a function that maps each $f(c)\hat{\in}H$ to a minimal hitting set of $\Gamma(\mathbb{K}_\mathcal{K}, q)$ w.r.t. the knot $sc(f(c))$.

Example 10. The following are some minimal hitting sets of $\Gamma(\mathbb{K}_{\mathcal{K}_1}, q)$: $\{z\}$ w.r.t. K_0, $\{x, x'\}$ w.r.t. K_2, $\{x, x'\}$ w.r.t. K_4, and $\{x, y\}$ w.r.t. K_5.

The sets G_1' and G_2' given below are extended min-graphs for $\mathcal{K}_1$, while G_1' is also an extended min-graph for $\mathcal{K}_2$.

$G_1' = \{D(a), B(a), P(a, f_1(a)), P(a, f_2(a)), A(f_1(a)), C(f_2(a))\}$
$G_2' = \{A(a), D(a), Q(a, f_3(a)), E(f_3(a))\}$

(sc_1^0, hs_1^0) and (sc_1^1, hs_1^1) are G_1'-hits and (sc_2, hs_2) is a G_2'-hit, where: $sc_1^0(f_1(a))=K_5$, $hs_1^0(f_1(a))=\{y\}$, $sc_1^0(f_2(a))=K_3$, $hs_1^0(f_2(a))=\emptyset$, $sc_1^1(f_1(a))=K_5$, $hs_1^1(f_1(a))=\{y\}$, $sc_1^1(f_2(a))=K_4$, $hs_1^1(f_2(a))=\{x, x'\}$, and for every other f, $sc_1^0(f(a))=sc_1^1(f(a))=sc_2(f(a))=K_0$ and $hs_1^0(f(a))=hs_1^1(f(a))=sc_2(f(a))=\{z\}$.

Next, we provide a way to decide the existence of matches for the query in all models starting with all possible extended graphs H. In a similar way as for knots, we consider all possible H-hits and check whether they comply with the different partial mappings inside the extended graph H.

Definition 19. Let $q_{\downarrow V}$ be the restriction of the query q to the atoms containing variables in V. We say an H-hit (sc, hs) *complies with* a constant tuple $\vec{c}$ if there exists a set $V \subseteq \mathbf{V}(q)$ and a homomorphism π from $q_{\downarrow V}$ to H such that:

- if $\pi(x) = f(a)$, then $x \in hs(f(a))$;
- if $y \in \mathbf{V}(q) \setminus V$, then for some $f(a)\hat{\in}H$ and some $x \in hs(f(a))$, we have $y \in \mathsf{V}_x$;
- for each answer variable x_i, $\pi(x_i) = c_i$.

Definition 20. Let $\mathbb{C}_\mathcal{K}$ be the set of all tuples (H, sc, hs) such that H is an extended min-graph of $\mathcal{K}$ and (sc, hs) is an H-hit. Then, for each such tuple $\lambda = (H, sc, hs)$ in $\mathbb{C}_\mathcal{K}$, we define $\mathsf{ans}(q, \lambda)$ to be the set of all tuples $\vec{c}$ that comply with the H-hit (sc, hs).

The next theorem, which characterizes CQ entailment over $\mathcal{ALCH}$ KBs, is proven in a similar way as Theorem 4.

Theorem 5. *For any CQ q over an $\mathcal{ALCH}$ KB $\mathcal{K}$, we have*

$$\mathsf{ans}(\mathcal{K}, q) = \bigcap_{\lambda \in \mathbb{C}_\mathcal{K}} \mathsf{ans}(q, \lambda).$$

Example 11. Consider $V = \{x, x', y\}$ and the homomorphism $h(x) = h(x') = a$, $h(y) = f_1(a)$ from $q_{\downarrow V}$ to G_1'; (sc_1^0, hs_1^0) and (sc_1^1, hs_1^1) comply with $\langle a, a\rangle$. Moreover, $\{y\}$ is the only minimal hitting set of $\Gamma(\mathbb{K}_{L_1^1}, q)$ w.r.t. K_5, so every G_1'-hit must map $f_1(a)$ to $\{y\}$. As G_1' is the only extended min-graph of $\mathcal{K}_2$, we have $\langle a, a\rangle \in \mathsf{ans}(q_1, \mathcal{K}_2)$. Note that no $\vec{c}$ complies with (sc_2, hs_2), thus $\mathsf{ans}(q_1, \mathcal{K}_1) = \emptyset$.

We remark here that for a given $\lambda = (H, sc, hs)$ in $\mathbb{C}_\mathcal{K}$, the set $\mathsf{ans}(q, \lambda)$ can be computed by posing a union of conjunctive queries over a set of atoms H_λ that is obtained by augmenting H with additional atoms to capture the H-hit (sc, hs). Furthermore, the sets H_λ for different $\lambda \in \mathbb{C}_\mathcal{K}$ can be generated in models of a datalog program (with unstratified negation); hence, $\mathsf{ans}(\mathcal{K}, q)$ is reducible to computing cautious consequence in datalog (with negation).

Computational Complexity

We now state the main complexity result of this paper.

Theorem 6. *Given an $\mathcal{ALCH}$ KB $\mathcal{K}$, a conjunctive query q, and tuple of individuals $\vec{c}$, deciding $\vec{c} \in \mathsf{ans}(\mathcal{K}, q)$ is in* ExpTime *w.r.t. the combined size of $\mathcal{K}$ and q.*

Proof. (Sketch) Let $cs := |\mathcal{K}| + |q|$. The result is derived based on the following observations:

- The number of distinct abstract knots over the signature of $\mathcal{K}$ is (single) exponential in cs, and $\mathbb{K}_\mathcal{K}$ can be constructed in exponential time by the procedure sketched in this paper.
- For a given $S \subseteq \mathbb{K}_\mathcal{K} \times \mathbb{R}_q$ and $(K, \rho) \in \mathbb{K}_\mathcal{K} \times \mathbb{R}_q$, we can verify $S \vdash_{\mathbb{K}_\mathcal{K}, q} (K, \rho)$ in time exponential in cs. As $|\mathbb{K}_\mathcal{K} \times \mathbb{R}_q|$ is exponential in cs, we can compute $\Gamma(L, q)$ in time exponential in cs; note that $\vdash_{\mathbb{K}_\mathcal{K}, q}$ is monotonic.

• For a given tuple $\lambda = (H, sc, hs)$ in $\mathbb{C}_\mathcal{K}$, the set $\mathsf{ans}(q, \lambda)$ can be computed in time exponential in cs. Indeed, for any tuple $\vec{c}$ of constants, the compliance of $\vec{c}$ with the H-hit (sc, hs) can be decided in exponential time in cs, and there are only exponentially many tuples of constants from $\mathcal{K}$ matching the arity of the answer variables of q.

• Finally, since $|\mathbb{C}_\mathcal{K}|$ is exponential in cs, verifying $\vec{c} \in \mathsf{ans}(\mathcal{K}, q)$ is feasible in time exponential in cs. □

EXPTIME-completeness of the problem follows from the well-known hardness result in (Schild 1991). We note that the above also implies that the set of all answers for q over $\mathcal{K}$ can be computed in time that is exponential in the input size.

Data Complexity. For a given $\mathcal{K}=\langle\mathcal{T}, \mathcal{A}\rangle$ and a CQ q, the algorithm can be easily adjusted to run in CONP in the size of $\mathcal{A}$, i.e., in data complexity.

To this end, we add for each concept name A in $\mathcal{K}$ the axioms $\top \sqsubseteq A \sqcup \bar{A}$ and $A \sqcap \bar{A} \sqsubseteq \bot$ to $\mathcal{T}$, where $\bar{A}$ is a fresh concept name. This transformation clearly preserves the answers to q, is independent of $\mathcal{A}$ and ensures that each model of $\mathcal{K}^g$ is also a min-graph of $\mathcal{K}$. We also assume that $\mathbb{K}_\mathcal{K}$ and $\Gamma(\mathbb{K}_\mathcal{K}, q)$ are precomputed since they do not depend on $\mathcal{A}$.

To check $\vec{c} \notin \mathsf{ans}(\mathcal{K}, q)$, a tuple $(H, sc, hs) \in \mathbb{C}_\mathcal{K}$ s.t. $\vec{c}$ does not comply with the H-hit (sc, hs) must be found. This can be done in non-deterministic polynomial time in $|\mathcal{A}|$: (1) guess a tuple (H, sc, hs), where H is set of atoms (i.e., a candidate extended min-graph) and sc and hs are two functions that map each term $f(c)\hat{\in}H$ to an abstract knot (over the signature of $\mathcal{K}$) and to a set of variables in q respectively; (2) verify in polynomial time whether (sc, hs) is an H-hit and whether it complies with $\vec{c}$. Indeed, given that $\mathbb{K}_\mathcal{K}$ is precomputed, checking whether H is an extended min-graph is feasible in polynomial time. We then use $\Gamma(\mathbb{K}_\mathcal{K}, q)$ to verify if conditions in Def. 18 are satisfied. As the query is fixed, only polynomially many potential homomorphisms must be traversed for deciding the compliance of $\vec{c}$ with (H, sc, hs).

It follows that $\vec{c} \in \mathsf{ans}(\mathcal{K}, q)$ can be decided in CONP in data complexity by the procedure presented. This is also worst-case optimal (Schaerf 1993).

Conclusion

In this paper, we have introduced *knots* as a novel technique for reasoning in expressive DLs, and developed a conjunctive query answering algorithm for $\mathcal{ALCH}$ that runs in single exponential time and is thus worst-case optimal.

Our algorithm is based on knots, which are novel in the context of DLs, and is different from previous query answering techniques. Perhaps most closely related is the resolution-based method by Hustadt et al. (2005). Similar as in our approach, it first "compiles" the knowledge base and the query into a special form, and then exploits the possibility to determine the query answer by means of a disjunctive datalog program. On the other hand, this is done on different grounds: the knot technique is model-theoretic in nature, while Hustadt et al.'s approach is proof-theoretic and relies on sophisticated resolution and superposition machinery.

The method that we presented for $\mathcal{ALCH}$ can be extended to richer DLs, and can be useful for query answering and other tasks in this context. In particular, transitive roles can be incorporated with some adjustment on the query answering part, while number restrictions can also be accommodated by suitably adapting the knot representation of KBs. This and other possible extensions, like more expressive queries, will be explored elsewhere.

Acknowledgments This work was partially supported by the Austrian Science Fund (FWF) grant P17212, the EC NoE REWERSE (IST-506779), and the Mexican National Council for Science and Technology (CONACYT) grant 187697.

References

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2006. Data complexity of query answering in description logics. In *Proc. KR'06*, 260–270.

Calvanese, D.; De Giacomo, G.; and Lenzerini, M. 1998. On the decidability of query containment under constraints. In *Proc. ACM PODS'98*, 149–158.

Calvanese, D.; Eiter, T.; and Ortiz, M. 2007. Answering regular path queries in expressive description logics: An automata-theoretic approach. In *Proc. AAAI'07*, 391–396.

Glimm, B.; Horrocks, I.; Lutz, C.; and Sattler, U. 2007. Conjunctive query answering for the description logic $\mathcal{SHIQ}$. In *Proc. IJCAI'07*, 399–404.

Glimm, B.; Horrocks, I.; Sattler, U.; 2007. Conjunctive query entailment for $\mathcal{SHOQ}$. In *Proc. DL'07*, 65–75.

Heflin, J., and Hendler, J. 2001. A portrait of the Semantic Web in action. *IEEE Intelligent Systems* 16(2):54–59.

Hustadt, U.; Motik, B.; and Sattler, U. 2005. Data complexity of reasoning in very expressive description logics. In *Proc. IJCAI'05*, 466–471.

Krötzsch, M.; Rudolph, S.; and Hitzler, P. 2007. Conjunctive queries for a tractable fragment of OWL 1.1. In *Proc. ISWC/ASWC 2007*, LNCS 4825, 310–323.

Levy, A. Y., and Rousset, M.-C. 1998. Combining Horn rules and description logics in CARIN. *Artificial Intelligence* 104(1–2):165–209.

Lutz, C. 2007. Inverse roles make conjunctive queries hard. In Calvanese, D., et al., eds., *Proc. DL'07*, 100–111.

Lutz, C. 2008. Two Upper Bounds for Conjunctive Query Answering in SHIQ. In *Proc. DL'08*, to appear.

Ortiz, M.; Calvanese, D.; and Eiter, T. 2008. Data Complexity of Query Answering in Expressive Description Logics via Tableaux. J. of Automated Reasoning, to appear.

Rosati, R. 2007. On conjunctive query answering in $\mathcal{EL}$. In Calvanese, D., et al., eds., *Proc. DL'07*, 451–458.

Schaerf, A. 1993. On the Complexity of the Instance Checking Problem in Concept Languages with Existential Quantification. In *J. Intell. Inf. Syst.*, 265-278.

Schild, K. 1991. A Correspondence Theory for Terminological Logics: Preliminary Report. In *Proc. IJCAI'91*.

Šimkus, M. and Eiter, T. 2007. $\mathbb{FDNC}$: Decidable non monotonic disjunctive logic programs with function symbols. In *Proc. LPAR'07*, LNCS 4790, 514–530.

Tessaris, S. 2001. *Questions and Answers: Reasoning and Querying in Description Logic*. Ph.D. Dissertation, Univ. Manchester, CS Dept.