Query Answering in Description Logics: the Knots Approach*

Thomas Eiter[1], Carsten Lutz[2], Magdalena Ortiz[1], and Mantas Šimkus[1]

[1] Institute of Information Systems, Vienna University of Technology
Favoritenstraße 9-11, A-1040 Vienna, Austria
(eiter|ortiz|simkus)@kr.tuwien.ac.at
[2] Fachbereich Mathematik und Informatik, University of Bremen
Bibliothekstraße 1, D-28359 Bremen, Germany
clu@informatik.uni-bremen.de

Abstract. In the recent years, query answering over Description Logic (DL) knowledge bases has been receiving increasing attention, and various methods and techniques have been presented for this problem. In this paper, we consider knots, which are an instance of the mosaic technique from Modal Logic. When annotated with suitable query information, knots are a flexible tool for query answering that allows for solving the problem in a simple and intuitive way. The knot approach yields optimal complexity bounds, as we illustrate on the DLs $\mathcal{ALCH}$ and $\mathcal{ALCHI}$, and can be easily extended to accommodate other constructs.

1 Introduction

The recent use of Description Logics (DLs) in a widening range of fields like the Semantic Web, data and information integration and ontology-based data access, has led to the study of new reasoning problems. In particular, accessing semantically enhanced data schemata expressed by means of DL ontologies via (extensions of) the popular *conjunctive queries (CQs)* has become an active area of research, cf. [3,9,8,11,18]. CQs are in general not expressible in the language of most DLs (at least not succinctly), and suitable methods for answering CQs are not always apparent.

The *mosaic technique* is a well-known method in Modal Logics [17,2], which has also been applied for reasoning in DLs [14,24]. Mosaics are small 'blocks' for building models, and possibly infinite models are represented by finite sets of these blocks. Local consistency conditions on mosaics and global coherency condition on sets of mosaics ensure correct model representation. As only finitely many mosaics must be considered and the global and local conditions are effectively verifiable, model existence can be decided by finding a suitable set of mosaics.

Here we discuss an instance of the mosaic technique called *knots* [6]. Knots are small tree-shaped mosaics easy to employ for solving the DL knowledge base

* This work has been partially supported by the Austrian Science Fund (FWF) grants P20840 and P20841, the EU Project Ontorule (FP7 231875), and the Mexican National Council for Science and Technology (CONACYT) grant 187697.

satisfiability problem. An attractive feature of this method is that it can be gracefully extended to CQ answering: we mark each knot with a *set of (sub)queries* that cannot be mapped locally into the model part the knot describes, and global conditions on sets of marked knots ensure that a full countermodel for the query can be constructed from them.

In this paper, we illustrate the approach on the DLs $\mathcal{ALCH}$ and $\mathcal{ALCHI}$, and obtain a worst-case optimal algorithm for CQ answering in both logics in a transparent way. For illustration of the core technique and to keep matters simple, we focus on a restricted case considering knowledge bases with a very simple data component (i.e., ABox), and give a general outline of how the technique extends to the general setting. As we will see, the marking of the knots is simple and intuitive, and flexible enough to easily extend to other DLs with different constructs. Furthermore, it allows for elegant refinements that yield optimal bounds even in the presence of very subtle sources of complexity.

The rest of the paper is organized as follows. After introducing the DLs we consider and their query answering problem in Section 2, we describe the knot-based algorithm in Section 3. This is done in three steps. First, we describe the generic knot-marking algorithm, and then we show how it can be used in the two considered DLs to obtain optimal complexity. The extensions to the case of unrestricted knowledge bases and to other DLs are briefly described next. Finally, related worked is addressed in Section 4.

2 Preliminaries

We introduce the DLs $\mathcal{ALCH}$ and $\mathcal{ALCHI}$ considered in this paper and discuss the basics of conjunctive query answering. We start with defining knowledge bases. Let $\mathbf{C}$, $\mathbf{R}$, and $\mathbf{I}$ be countably infinite sets of *concept names*, *role names*, and *individual names*. A *role* is either a role name $r \in \mathbf{R}$ or an expression r^- (called the *inverse role of* r). *Concepts* are defined inductively: (i) all concept names $A \in \mathbf{C}$ are concepts, and (ii) if C, D are concepts and r is a role, then $\neg C$, $C \sqcap D$, $C \sqcup D$, $\forall r.C$ and $\exists r.C$ are concepts. As usual, for an inverse role $r = s^-$, we write r^- to denote s.

An *$\mathcal{ALCHI}$ knowledge base (KB)* $\mathcal{K}$ is a finite set of statements of the following form: (i) *concept inclusions (CIs)* $C \sqsubseteq D$ with C and D concepts; (ii) *role inclusions (RIs)* $r \sqsubseteq s$ with r and s roles; (iii) *concept assertions* $A(a)$, with a an individual name and A a concept name; and (iv) *role assertions* $r(a, b)$, with a, b individual names and r a role. If $\mathcal{K}$ does not contain inverse roles, then it is an *$\mathcal{ALCH}$ knowledge base*. By $\mathbf{C}_\mathcal{K}$ and $\mathbf{R}_\mathcal{K}$, we denote the sets of all concept and role names, respectively, that occur in $\mathcal{K}$.

The semantics is given via first-order interpretations. An *interpretation* $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ consists of a non-empty set $\Delta^\mathcal{I}$ (the *domain*) and a *valuation function* $\cdot^\mathcal{I}$ that maps each concept name $A \in \mathbf{C}$ to subset $A^\mathcal{I}$ of $\Delta^\mathcal{I}$, each role name $r \in \mathbf{R}$ to a subset $r^\mathcal{I}$ of $\Delta^\mathcal{I} \times \Delta^\mathcal{I}$, and each individual name a to an element $a^\mathcal{I} \in \Delta^\mathcal{I}$. The function $\cdot^\mathcal{I}$ is extended to all concepts and roles as follows:

$$
\begin{aligned}
(p^-)^{\mathcal{I}} &= \{(y,x) \mid (x,y) \in r^{\mathcal{I}}\},\\
(\neg C)^{\mathcal{I}} &= \Delta^{\mathcal{I}} \backslash C^{\mathcal{I}},\\
(C \sqcap D)^{\mathcal{I}} &= C^{\mathcal{I}} \cap D^{\mathcal{I}}, \qquad (C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}},\\
(\exists r.C)^{\mathcal{I}} &= \{x \mid \exists y.(x,y) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\},\\
(\forall r.C)^{\mathcal{I}} &= \{x \mid \forall y.(x,y) \in r^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\}.
\end{aligned}
$$

An interpretation $\mathcal{I}$ *satisfies* a CI $C \sqsubseteq D$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, an RI $r \sqsubseteq s$ if $r^{\mathcal{I}} \subseteq s^{\mathcal{I}}$, a concept assertion $A(a)$ if $a^{\mathcal{I}} \in A^{\mathcal{I}}$, and a role assertion $r(a,b)$ if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$. Moreover, $\mathcal{I}$ is a *model of a KB* $\mathcal{K}$ *(in symbols,* $\mathcal{I} \models \mathcal{K}$*)* if it satisfies all inclusions and assertions in $\mathcal{K}$. A KB is *consistent* if it has a model.

We now recall conjunctive queries. To keep the presentation simple, we restrict ourselves to Boolean conjunctive queries, i.e., queries without answer variables. Technically, the general case is no more difficult than this restricted one and admits the same techniques and similar algorithms. We also consider queries with variables only, and constants can be simulated in the usual way. Let $\mathbf{V}$ be a countably infinite set of *variables*. A *(Boolean) conjunctive query (CQ)* is a finite set of atoms of the form $A(x)$ or $r(x,y)$, where A is a concept name, r is a role, and $x, y \in \mathbf{V}$. The variables occurring in the atoms of q are denoted $\mathbf{V}(q)$. When we work with $\mathcal{ALCH}$ KBs, we assume that CQs contain only role names, but no inverse roles.

A *match for* q *in an interpretation* $\mathcal{I}$ is a mapping $\pi : \mathbf{V}(q) \rightarrow \Delta^{\mathcal{I}}$ such that (i) $\pi(x) \in A^{\mathcal{I}}$ for each $A(x) \in q$, and (ii) $(\pi(x), \pi(y)) \in r^{\mathcal{I}}$ for each $r(x,y) \in q$. If such π exists, we write $\mathcal{I} \models q$. We write $\mathcal{K} \models q$ (and say "$\mathcal{K}$ entails q") if there is a match for q in every model of the KB $\mathcal{K}$. This defines the problem that we consider in this paper: given a KB $\mathcal{K}$ and a CQ q, decide whether $\mathcal{K}$ entails q. There is one additional simplifying assumption concerning the structure of the query. As usual and w.l.o.g., we assume throughout the paper that all queries are *connected*, i.e., there is only one connected component in the *query graph* G^q, which is the directed graph with nodes $\mathbf{V}(q)$ that has an edge from x to y iff $r(x,y) \in q$ for some role r. Non-connected queries can be answered by independently answering each connected component.

In most parts of this paper, we will concentrate on knowledge bases that contain only a single concept assertion $C_0(a_0)$ and no role assertions. We call such a KB *simple*. Just like the simplifying assumptions that we make on conjunctive queries, this serves the purpose of an easier exposition. In contrast to those assumptions, however, this case *is* technically simpler than the general case. We will discuss this in more detail in Section 3.4. When deciding query entailment over a simple $\mathcal{ALCHI}$ KB $\mathcal{K}$, it suffices to concentrate on models of $\mathcal{K}$ that are tree shaped. Formally, a model $\mathcal{I}$ of $\mathcal{K}$ is a *tree model* if:

1. $\Delta^{\mathcal{I}}$ is a prefixed-closed subset of $\mathbb{N}^*$ (i.e., of words over the natural numbers); the empty word is denoted by ε and called the *root* of $\Delta^{\mathcal{I}}$,
2. if $(d,e) \in (\exists r.C)^{\mathcal{I}}$ for some role r and some concept C, then, for some $j \in \mathbb{N}$, $(d, d{\cdot}j) \in r^{\mathcal{I}}$ and $d{\cdot}j \in C^{\mathcal{I}}$, and

3. if $(d, e) \in r^{\mathcal{I}}$ for some role name r, then, for some $j \in \mathbb{N}$, either (i) $e = d{\cdot}j$, or (ii) $d = e{\cdot}j$.

If $\mathcal{I}$ is a tree model of $\mathcal{K}$, and for each $(d, e) \in r^{\mathcal{I}}$ we have $e = d{\cdot}j$ for some $j \in \mathbb{N}$ (i.e., if (3ii) never applies), then $\mathcal{I}$ is a *1-way tree model*. In the case of $\mathcal{ALCH}$ KBs, it even suffices to concentrate on 1-way tree models. From now on, the *size* of a knowledge base $\mathcal{K}$ is denoted $|\mathcal{K}|$.

Proposition 1. *Every consistent simple $\mathcal{ALCHI}$ KB $\mathcal{K}$ has a tree model, and for each query q, if $\mathcal{K} \not\models q$, then there exists a tree model $\mathcal{I}$ of $\mathcal{K}$ such that $\mathcal{I} \not\models q$. The model $\mathcal{I}$ is such that $\Delta^{\mathcal{I}} \subseteq \{0, \ldots, |\mathcal{K}| - 1\}^*$ and it is 1-way if $\mathcal{K}$ is an $\mathcal{ALCH}$ KB.*

We note that a similar proposition can be established for non-simple KBs, but then tree models have to be replaced by forest-like ones; see Section 3.4.

3 Query Answering by Knot Elimination

We describe the knot technique, and show how it can be used to decide CQ entailment over simple KBs formulated in $\mathcal{ALCH}$ and $\mathcal{ALCHI}$, yielding tight upper complexity bounds. Then we discuss extensions to general KBs and more expressive DLs.

3.1 Knots

The aim of the *knot technique* is to obtain a finite representation of potentially infinite tree models by decomposing them into a collection of small pieces. Each such piece is described by a *knot*, a schematic labeled tree of depth ≤ 1 with bounded branching. A knot describes a node in the tree model together with all its successors, fixing the concepts that are satisfied at each node and the roles connecting the nodes. By restricting ourselves to only the *relevant* concepts and roles, we achieve that only finitely many distinct knots exist, and thus every tree model can be represented as a finite knot set. Conversely, knots can be viewed as the 'building blocks' for a potential tree model. To ensure that knots can indeed be assembled into such a model, two kinds of conditions are imposed. *Local* conditions apply to individual knots and deal with the internal consistency of the nodes in a knot. *Global* conditions ensure that instances of the knots in a set can be assembled together.

We assume from now on that concepts are in negation normal form (NNF). It is well know that every concept can be converted into an equivalent one in NNF in linear time. We use $\sim C$ to denote the result of converting $\neg C$ into NNF. For the rest of the section, fix a simple $\mathcal{ALCHI}$ knowledge base $\mathcal{K}$ with concept atom $C_0(a_0)$.

Definition 1 (Knot). *Let $\mathsf{cl}(\mathcal{K})$ denote the smallest set of concepts that contains every concept in $\mathcal{K}$ and is closed under subconcepts and NNF-negation "$\sim$". A* concept-type for $\mathcal{K}$ *is a set $\tau \subseteq \mathsf{cl}(\mathcal{K})$ such that for all $C, D \in \mathsf{cl}(\mathcal{K})$,*

1. $C \sqsubseteq D \in \mathcal{K}$ *implies* $\sim C \in \tau$ *or* $D \in \tau$*;*
2. $C \in \tau$ *implies* $\sim C \notin \tau$*,*
3. *if* $C \sqcap D \in \tau$*, then* $\{C, D\} \subseteq \tau$*, and*
4. *if* $C \sqcup D \in \tau$*, then* $C \in \tau$ *or* $D \in \tau$*.*

A role-type (for $\mathcal{K}$) *is a set* $\rho \subseteq \mathbf{R}_\mathcal{K} \cup \{p^- \mid p \in \mathbf{R}_\mathcal{K}\}$*. A* knot for $\mathcal{K}$ *is a pair* $\kappa = (\tau, S)$ *that consists of a concept-type* τ *for* $\mathcal{K}$ *(called* root type*) and a set* S *of pairs* (ρ, τ')*, where* ρ *and* τ' *are a role-type and a concept-type for* $\mathcal{K}$*.*

A knot $\kappa = (\tau, S)$ can be viewed as a tree of depth ≤ 1, whose nodes are labeled with subsets of $\mathsf{cl}(\mathcal{K})$, and whose edges are labeled with sets of roles. More specifically, τ is the label of the root node and each pair $(\rho, \tau') \in S$ describes a successor with edge label ρ and node label τ'. Next, we define local conditions for knots which ensure that there are no contradictions within the knot.

Definition 2 (Knot consistency). *A knot* $\kappa = (\tau, S)$ *is* $\mathcal{K}$-consistent *if:*

1. *if* $\exists r.C \in \tau$*, then* $r \in \rho$ *and* $C \in \tau'$ *for some* $(\rho, \tau') \in S$*;*
2. *if* $\forall r.C \in \tau$*, then* $C \in \tau'$ *for all* $(\rho, \tau') \in S$ *with* $R \in \rho$*;*
3. *if* $\forall r.C \in \tau'$ *for some* $(\rho, \tau') \in S$ *and* $r^- \in \rho$*, then* $C \in \tau$*; and*
4. *if* $r \sqsubseteq s \in \mathcal{K}$ *and* $(\rho, \tau') \in S$*, then* $r \in \rho$ *implies* $s \in \rho$*, and* $r^- \in \rho$ *implies* $s^- \in \rho$*.*
5. $|S| \leq |\mathsf{cl}(\mathcal{K})|$*.*

A knot being free of local contradictions does not yet guarantee that it can be part of a tree model as there could be existential restrictions at successor nodes that cannot be expanded into a full model. We therefore also need a global condition which guarantees that such an expansion is always possible

Definition 3 (Coherency of knot sets). *Given a knot set* $\mathbb{K}$*, a knot* $(\tau, S) \in \mathbb{K}$ *is* good in $\mathbb{K}$*, if for each* $(\rho, \tau') \in S$*, there is a knot* $(\tau_s, S_s) \in \mathbb{K}$ *with* $\tau' = \tau_s$*. Then* $\mathbb{K}$ *is* $\mathcal{K}$-coherent *if (i) each knot* $(\tau, S) \in \mathbb{K}$ *is* $\mathcal{K}$*-consistent and good in* $\mathbb{K}$*; and (ii) there is a knot* $(\tau, S) \in \mathbb{K}$ *with* $C_0 \in \tau$*.*

A tree-shaped model $\mathcal{I}$ of $\mathcal{K}$ can be decomposed into a $\mathcal{K}$-coherent knot set in a straightforward way. Conversely, if we have a $\mathcal{K}$-coherent set of knots $\mathbb{K}$, we can build a tree model of $\mathcal{K}$: start with the knot (τ, S) with $C_0 \in \tau$ as the 'root knot', then repeatedly append suitable successor knots to the leafs of the tree.

Theorem 1. *$\mathcal{K}$ is consistent iff there exists a $\mathcal{K}$-coherent knot set.*

3.2 Non-Entailment of a Set of Tree-shaped Queries

We now present a knot-based approach to decide query entailment in $\mathcal{ALCH}$ and $\mathcal{ALCHI}$, which yields a tight ExpTime upper bound in the $\mathcal{ALCH}$ case, and a tight 2-ExpTime upper bound for $\mathcal{ALCHI}$. We proceed in two steps. The first step is presented in the current section, where we give a knot-based algorithm for query entailment in $\mathcal{ALCHI}$ that presupposes tree-shaped queries and runs in ExpTime. In fact, the algorithm works with sets of queries and decides a special, non-standard version of entailment. The second step is presented in the

subsequent section, where we reduce standard query entailment to the special case treated in the current section.

We say that a query q is *tree-shaped* if the query graph G^q is a tree. We assume that the nodes in tree-shaped queries q have canonical names: the root of G^q is x_ϵ and if x_w is a node in G^q with n children, then these children are $x_{w\cdot 1}, \ldots, x_{w\cdot n}$. For a variable x_w, we denote by $\mathsf{subq}(q, x_w)$ the canonical query obtained by restricting q to the subtree rooted at x_w, renaming the nodes accordingly. We now introduce the special kind of entailment used in this section.

Definition 4 (Directed Entailment). *Let $\mathcal{K}$ be an $\mathcal{ALCHI}$ KB, $\mathcal{I}$ be a tree model of $\mathcal{K}$, Q a set of tree-shaped queries, and $q \in Q$. For $d \in \Delta^{\mathcal{I}}$, we write $\mathcal{I} \models^* q[d]$ if there exists a match π for q in $\mathcal{I}$ such that $\pi(x_\epsilon) = d$ and for every $r(x, y) \in q$, we have $\pi(y) = \pi(x) \cdot i$ for some $i \geq 0$. We write $\mathcal{I} \models^* q$ if $\mathcal{I} \models^* q[d]$ for some $d \in \Delta^{\mathcal{I}}$, $\mathcal{I} \models^* Q$ if $\mathcal{I} \models^* q$ for some $q \in Q$, and $\mathcal{K} \models^* Q$ if $\mathcal{I} \models^* Q$ for every tree model $\mathcal{I}$ of the knowledge base $\mathcal{K}$.*

Observe that the matches used in Definition 4 are directed in the sense that, even if the tree model is not 1-way, we map every atom $r(x, y) \in q$ only "downwards". If the tree model is 1-way *and* the query does not involve inverse roles as in the case of $\mathcal{ALCH}$, then these matches coincide with standard ones, i.e., we have $\mathcal{K} \models q$ iff $\mathcal{K} \models^* \{q\}$.

The algorithm devised in this section decides, given an $\mathcal{ALCH}$ or $\mathcal{ALCHI}$ KB $\mathcal{K}$ and a set of tree-shaped queries Q, whether $\mathcal{K} \models^* Q$. The following characterization of directed entailment is easy to establish.

Proposition 2. *Let $\mathcal{I}$ be a tree model of an $\mathcal{ALCHI}$ KB $\mathcal{K}$, $d \in \Delta^{\mathcal{I}}$, and q a tree-shaped query. Then $\mathcal{I} \not\models^* q[d]$ iff one of the following holds:*

(i) $\{A \mid A(x_\epsilon) \in q\} \not\subseteq \{A \mid d \in A^{\mathcal{I}}\}$ *or*
(ii) there exists a variable $x_{\epsilon\cdot i}$ of q such that for each child $d\cdot j \in \Delta^{\mathcal{I}}$ of d we have:
- $\{r \mid r(x_\epsilon, x_{\epsilon\cdot i}) \in q\} \not\subseteq \{r \mid (d, d\cdot j) \in r^{\mathcal{I}}\}$, *or*
- $\mathcal{I} \not\models^* \mathsf{subq}(q, x_{\epsilon\cdot i})[d\cdot j]$.

For the rest of the section, fix a simple $\mathcal{ALCH}$ or $\mathcal{ALCHI}$ KB $\mathcal{K}$ and a set of tree-shaped queries Q. The aim of our algorithm is to decide whether $\mathcal{K} \models^* Q$ by using knots to verify the existence of a tree model $\mathcal{I}$ of $\mathcal{K}$ with $\mathcal{I} \not\models^* Q$ (a *countermodel*). Proposition 2 suggests that we can do this by extending knots with auxiliary information that enables us to track the satisfaction of conditions (i) and (ii) for each query in Q at each node of the tree.

Definition 5 (Marked knots). *Let Q^* denote the smallest set such that $Q \subseteq Q^*$, and if $q \in Q^*$ and $x_{\epsilon\cdot i}$ is a variable of q, then $\mathsf{subq}(q, x_{\epsilon\cdot i}) \in Q^*$. A Q-*marked *knot is a tuple (τ, S, ν) where (τ, S) is a knot and $\nu : \{\epsilon\} \cup S \to 2^{Q^*}$.*

Intuitively, every node in a Q-marked knot is labeled with the set of those subqueries of queries in Q for which a (directed) match of the root should be *avoided* at that node. To capture this formally, we define additional local conditions.

algorithm CounterModel($\mathcal{K}, Q$)
Compute the set $\mathbb{K}_0$ of all Q-avoiding knots for $\mathcal{K}$
$i := 0$
repeat
  $i := i + 1$
  $\mathbb{K}_i := \mathbb{K}_{i-1} \setminus \{(\tau, S, \nu) \in \mathbb{K}_{i-1} \mid (\tau, S, \nu)$ is not good in $\mathbb{K}_{i-1}\}$
until $\mathbb{K}_i \neq \mathbb{K}_{i-1}$;
if there is $(\tau, S, \nu) \in \mathbb{K}_i$ with $C_0 \in \tau$ **then return** " a counter model exists"
  else return "a counter model does not exist"

Fig. 1. The knot elimination algorithm.

Definition 6 (Query avoiding knots). *A Q-marked knot (τ, S, ν) is* Q-avoiding*, if for each $q \in Q$, we have $q \in \nu(\epsilon)$ and that one of the following holds:*

(a) $\{A \mid A(x_\epsilon) \in q\} \not\subseteq \tau$*, or*
(b) there exists some variable $x_{\epsilon \cdot i}$ such that for every $(r, \tau') \in S$, it holds that $\{r \mid r(x_\epsilon, x_{\epsilon \cdot i}) \in q\} \not\subseteq \rho$ *or* $\mathsf{subq}(q, x_{\epsilon \cdot i}) \in \nu((\rho, \tau))$.

The above just mimics the conditions in Proposition 2. We now define global conditions to ensure that the marking is consistent between knots.

Definition 7 (Coherency of marked knot sets). *For a set $\mathbb{K}$ of Q-marked knots, we call $(\tau, S, \nu) \in \mathbb{K}$* good in $\mathbb{K}$ *if for each $(\rho, \tau') \in S$, there is some $(\tau_s, S_s, \nu_s) \in \mathbb{K}$ such that $\tau' = \tau_s$ and $\nu((\rho, \tau')) = \nu_s(\epsilon)$. Then $\mathbb{K}$ is* $\mathcal{K}$-coherent *if for each $(\tau, S, \nu) \in \mathbb{K}$, (τ, S) is $\mathcal{K}$-consistent and (τ, S, ν) is Q-avoiding and good in $\mathbb{K}$.*

To show the following, we can now argue as for Theorem 1, additionally using Proposition 2.

Proposition 3. *$\mathcal{K} \not\models^* Q$ iff there exists a $\mathcal{K}$-coherent set of Q-marked knots.*

To decide whether $\mathcal{K} \not\models^* Q$, it thus suffices to decide the existence of a knot set as in Proposition 3. This is done by knot elimination, inspired by the so-called type elimination technique due to Pratt; see Section 4. The details of the algorithm are presented in Figure 1, where we assume that $C_0(a_0)$ is the concept assertion in $\mathcal{K}$. The algorithm runs in exponential time since there are only exponentially many Q-marked knots for $\mathcal{K}$ and it can be checked in polynomial time whether a knot is Q-avoiding and good in a knot set.

Theorem 2. *Given $\mathcal{K}$ and Q, it can be decided in time exponential in the size of $\mathcal{K}$ and Q whether $\mathcal{K} \models^* Q$.*

3.3 From Unrestricted Queries to Tree-Shaped Ones

We now show how Theorem 2 can be used to derive tight complexity bounds for standard entailment of a conjunctive query q that is not necessarily tree-shaped by a simple $\mathcal{ALCHI}$ KB $\mathcal{K}$. By Proposition 1, q not being entailed by $\mathcal{K}$ implies that there is a tree model $\mathcal{I}$ of $\mathcal{K}$ that witnesses non-entailment. Further, any match π for q in model $\mathcal{I}$ gives rise to a rewriting q_π of q as follows:

– the variables of q_π are $\{x_d \mid \exists x \in \mathbf{V}(q) : \pi(x) = d\}$;
– the concept atoms of q_π are $\{A(x_d) \mid \exists\, A(x) \in q : \pi(x) = d\}$;
– the role atoms are $\{r(x_d, x_{d\cdot i}) \mid \exists\, r(x,y) \in q : \pi(x) = d \text{ and } \pi(y) = d \cdot i\} \cup \{r(x_d, x_{d\cdot i}) \mid \exists\, r^-(y,x) \in q : \pi(x) = d \text{ and } \pi(y) = d \cdot i\}$

Since q is connected and $\mathcal{I}$ is a tree model, q_π is obviously tree-shaped (and it is straightforward to assign canonical names to the variables). Moreover, the construction of q_π ensures that $\mathcal{I} \models^* q_\pi$, i.e., there is a *directed* match for q_π in $\mathcal{I}$. This observation suggests that entailment of q can be verified by replacing q with a set of tree-shaped rewritings and checking directed entailment.

Definition 8. *A tree-shaped query q' is a* tree rewriting *of q if there is a surjective map $\nu : \mathbf{V}(q) \to \mathbf{V}(q')$ such that*

(i) $A(x) \in q$ *iff* $A(\nu(x)) \in q'$, *and*
(ii) $r(x,y) \in q$ *iff* $r(\nu(x), \nu(y)) \in q'$ *or* $r^-(\nu(y), \nu(x)) \in q'$.

Let $\mathsf{TRew}(q)$ *denote all tree rewritings of q.*

Based on the observation above, the following is easy to prove.

Lemma 1. *For each simple $\mathcal{ALCHI}$ KB $\mathcal{K}$ and CQ q, we have $\mathcal{K} \not\models q$ iff $\mathcal{K} \not\models^* \mathsf{TRew}(q)$.*

For an $\mathcal{ALCHI}$ KB $\mathcal{K}$, we can thus decide whether $\mathcal{K} \models q$ by using the algorithm from the previous section with $Q = \mathsf{TRew}(q)$. Since the cardinality of $\mathsf{TRew}(q)$ is exponential in the size of q, we obtain a 2-ExpTime upper bound. This bound is tight: in [12], it was shown that CQ entailment over simple $\mathcal{ALCI}$ KBs (i.e., $\mathcal{ALCHI}$ KBs without role inclusions) is 2-ExpTime-hard.

In the case of $\mathcal{ALCH}$, we can replace $\mathsf{TRew}(q)$ with a single query! Recall that $\mathcal{ALCH}$ terminologies enjoy 1-way tree models and that we disallow inverse roles in the query when working with $\mathcal{ALCH}$. Together, this means that if we have $\mathcal{I} \models q$ with $\mathcal{I}$ a 1-way tree-model of the (simple) input KB $\mathcal{K}$, we can obtain a tree-shaped rewriting q' of q with $\mathcal{I} \models q'$ in a very easy way: simply eliminate all *forks* $r(x,y), r(x',y)$ in q by identifying the variables x and x'. Observe that, in contrast to the case of $\mathcal{ALCHI}$, this rewriting is independent of the concrete match π of q in $\mathcal{I}$. Thus, we obtain only a single rewriting q' (which can be obtained in polynomial time). As noted already in Section 3.2, we then have $\mathcal{I} \models q'$ iff $\mathcal{I} \models^* q'$ which enables the use of the algorithm in the previous section.

Definition 9 (Query rewriting). *[12] Let $\mathcal{K}$ be a simple $\mathcal{ALCH}$ KB, q a CQ without inverse roles, and let* $\mathsf{FE}(q)$ *denote the result of eliminating all forks in q. Then $\mathcal{K} \models q$ iff $\mathcal{K} \models^* \{\mathsf{FE}(q)\}$.*

We thus obtain an ExpTime upper bound by Theorem 2. A lower bound is easily obtained by a trivial reduction of satisfiability in $\mathcal{ALCH}$.

3.4 Extensions

The knot-based approach to query answering can be extended to the case of non-simple KBs and to more expressive DLs than $\mathcal{ALCH}$ and $\mathcal{ALCHI}$. We start with the former. As noted in Section 2, Proposition 1 can be adapted to the case of non-simple KBs by replacing tree models with forest-shaped ones. More precisely, such models consist of a core whose relational structure is unrestricted and a collection of possibly infinite trees whose roots are from the core. The core contains precisely those elements that are identified by some individual name in the knowledge base $\mathcal{K}$, and thus its size is bounded by that of $\mathcal{K}$. This also explains why the relational structure of the core cannot be restricted: it needs to mirror the role assertions $r(a, b)$ in $\mathcal{K}$.

In the knot approach, the whole core is represented by a single, large knot, cf. the *min-graphs* of [20]. To avoid matches when constructing a countermodel, we now have to deal with three types of matches: (i) matches located purely inside the core; (ii) matches located partially in the core and partially in one or more of the trees; and (iii) matches located purely inside a tree. This can be done by a careful extension of the local and global conditions for marked knots and is most subtle in the case of $\mathcal{ALCH}$. There, it is crucial to show that, although in matches of type (ii) there are exponentially many ways to split the query between the core and the trees, only polynomially many queries need to be taken into account when avoiding matches in the tree parts [20,13]. The assumption made in this paper that knowledge bases are simple allows us to concentrate on matches of type (iii). Though this may appear to be the easiest of the three cases, in $\mathcal{ALCH}$ and $\mathcal{ALCHI}$ it is actually the source of complexity: the 2-ExpTime lower bound for $\mathcal{ALCHI}$ in [12] applies to simple KBs, and the complexity does not increase for non-simple ones [9].

We now come to more expressive DLs than $\mathcal{ALCH}$ and $\mathcal{ALCHI}$. It is not hard to extend our approach to *number restrictions* by imposing counting constraints on the successors in knots, thus capturing the DLs $\mathcal{ALCHQ}$ and $\mathcal{ALCHIQ}$. The extended algorithm still yields ExpTime and 2-ExpTime upper bounds, respectively, though some care has to be taken when combining $\mathcal{Q}$ and $\mathcal{I}$. A particularly interesting extension is provided by *transitive roles*. In their presence, it is not possible to generate a set of tree-shaped queries of polynomial size even when we are working with 1-way models and simple KBs. The knot-based approach thus yields a 2-ExpTime upper bound [19], which is optimal if the considered DL also has role inclusions (i.e., if it contains the DL $\mathcal{SH}$) [5]. In [19], a class of queries is identified for which the complexity of CQ entailment in $\mathcal{SH}$ drops to ExpTime. When we disallow role inclusions, the 2-ExpTime upper bound is no longer tight. In fact, a somewhat intricate refinement of the knot approach can be used to show that, in $\mathcal{ALC}$ with transitive roles (also known as $\mathcal{S}$), CQ entailment w.r.t. simple KBs is still in ExpTime [5]. In the same paper, we show that for non-simple KBs, the complexity raises to co-NExpTime-hardness. In a nutshell, this is due to matches of type (ii) being more complicated than without transitive roles and giving rise to an exponential number of queries to be avoided in the tree parts of models. A tight complexity bound is still missing.

4 Related Work and Conclusion

There is a large number of other approaches to conjunctive query entailment in DLs, including reductions to satisfiability [9,8,12], automata and tableaux methods [3,18], and resolution [11]. We omit a detailed discussion due to lack of space and instead discuss techniques that are similar in spirit to knots. As already mentioned, knots are a special instance of the *mosaic* technique [17] that has been used to obtain decidability and complexity results in modal and description logic. The reader may refer to [16,2] and consult e.g. [14,24] as examples from the DL literature. With the exception of [4,23], we are not aware of papers in which other variations of the mosaic technique have been used for query answering. Both knots and mosaics are closely related to *type elimination*, which has been used extensively in description and modal logic, see e.g. [22,21,10,15]. Roughly, a type is a small mosaic with only one element and type elimination is the analogue of Figure 1 with knots replaced by types. We remark that the algorithm in Section 3.2 can also be formulated using annotated types instead of annotated knots. However, using knots allows for simpler local and global conditions, especially when extending the approach to more expressive DLs such as those involving transitive roles.

Summing up, in this paper we have illustrated how the knot technique can be applied for answering conjunctive queries in DLs. The method is conceptually simple yet powerful enough to handle different DLs with considerably different computational properties. To wit, we presented a worst-case optimal algorithm that directly scales from the DL $\mathcal{ALCH}$ to the exponentially harder DL $\mathcal{ALCHI}$. Given that knots are special mosaics tailored for DLs with tree-shaped models, investigating the mosaic technique for query answering in more expressive DLs which lack this property, like (fragments of) $\mathcal{SHOIQ}$ and $\mathcal{SROIQ}$, is an interesting topic for future research.

References

1. F. Baader, C. Lutz, and B. Motik, editors. *Proceedings of the 21st International Workshop on Description Logics (DL2008), Dresden, Germany, May 13-16, 2008*, volume 353 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.
2. P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic*, volume 53 of *Cambridge Tracts in Theoretical Computer Sc.* Cambridge University Press, Cambridge, 2001.
3. D. Calvanese, T. Eiter, and M. Ortiz. Answering regular path queries in expressive description logics: An automata-theoretic approach. In *Proc. of the 22nd Nat. Conf. on Artificial Intelligence (AAAI 2007)*, pages 391–396, 2007.
4. T. Eiter, G. Gottlob, M. Ortiz, and M. Šimkus. Query answering in the description logic Horn-SHIQ. In *Proceedings 9th European Conference on Logics in Artificial Intelligence (JELIA 2008)*, LNCS, pages 166–179. Springer, 2008.
5. T. Eiter, C. Lutz, M. Ortiz, and M. Šimkus. Query answering in description logics with transitive roles. In C. Boutilier, editor, *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI-09)*. AAAI Press/IJCAI, 2009.

6. T. Eiter, M. Ortiz, and M. Simkus. Reasoning using knots. In I. Cervesato, H. Veith, and A. Voronkov, editors, *LPAR*, volume 5330 of *Lecture Notes in Computer Science*, pages 377–390. Springer, 2008.
7. D. Fox and C. P. Gomes, editors. *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, AAAI 2008, Chicago, Illinois, USA, July 13-17, 2008*. AAAI Press, 2008.
8. B. Glimm, I. Horrocks, and U. Sattler. Conjunctive query entailment for $\mathcal{SHOQ}$. In *Proc. of the 2007 Description Logic Workshop (DL 2007)*, volume 250 of *CEUR Electronic Workshop Proceedings,* http://ceur-ws.org/Vol-250/, pages 65–75, 2007.
9. B. Glimm, C. Lutz, I. Horrocks, and U. Sattler. Answering conjunctive queries in the $\mathcal{SHIQ}$ description logic. *Journal of Artificial Intelligence Research*, 31:150–197, 2008.
10. J. Y. Halpern and Y. Moses. A guide to completeness and complexity for modal logics of knowledge and belief. *Artif. Intell.*, 54(3):319–379, 1992.
11. U. Hustadt, B. Motik, and U. Sattler. A decomposition rule for decision procedures by resolution-based calculi. In *Proc. 11th Int. Conf. on Logic for Programming, Artificial Intelligence and Reasoning (LPAR 2004)*, pages 21–35, 2004.
12. C. Lutz. The complexity of conjunctive query answering in expressive description logics. In A. Armando, P. Baumgartner, and G. Dowek, editors, *Proceedings of the 4th International Joint Conference on Automated Reasoning (IJCAR2008)*, number 5195 in LNAI, pages 179–193. Springer, 2008.
13. C. Lutz. Two upper bounds for conjunctive query answering in SHIQ. In Baader et al. [1].
14. C. Lutz, U. Sattler, and L. Tendera. The complexity of finite model reasoning in description logics. *Inf. Comput.*, 199(1-2):132–171, 2005.
15. C. Lutz, F. Wolter, and M. Zakharyaschev. Temporal description logics: A survey. In S. Demri and C. S. Jensen, editors, *Proc. 15th International Symposium on Temporal Representation and Reasoning (TIME 2008)*, pages 3–14. IEEE Computer Society, 2008.
16. Y. V. Maarten Marx. Local variations on a loose theme: Modal logic and decidability. In *Finite Model Theory and Its Applications*, chapter 7, pages 371–429. Springer, June 2007.
17. I. Németi. Free algebras and decidability in algebraic logic. DSc. thesis, Mathematical Institute of The Hungarian Academy of Sciences, Budapest, 1986.
18. M. Ortiz, D. Calvanese, and T. Eiter. Data complexity of query answering in expressive description logics via tableaux. *J. of Automated Reasoning*, 41(1):61–98, 2008.
19. M. Ortiz, M. Simkus, and T. Eiter. Conjunctive query answering in SH using knots. In Baader et al. [1].
20. M. Ortiz, M. Simkus, and T. Eiter. Worst-case optimal conjunctive query answering for an expressive description logic without inverses. In Fox and Gomes [7], pages 504–510.
21. G. Pan, U. Sattler, and M. Y. Vardi. BDD-based decision procedures for the modal logic k. *Journal of Applied Non-Classical Logics*, 16(1-2):169–208, 2006.
22. V. R. Pratt. Models of program logics. In *FOCS*, pages 115–122. IEEE, 1979.
23. I. Pratt-Hartmann. Data-complexity of the two-variable fragment with counting quantifiers. *Information and Computation*, 2008. Forthcoming. See CoRR http://arxiv.org/abs/0806.1636.
24. S. Rudolph, M. Krötzsch, and P. Hitzler. Terminological reasoning in $\mathcal{SHIQ}$ with ordered binary decision diagrams. In Fox and Gomes [7], pages 529–534.