# Answering Regular Path Queries in Expressive Description Logics: An Automata-Theoretic Approach *

**Diego Calvanese**
Faculty of Computer Science
Free University of Bozen-Bolzano
Piazza Domenicani 3, Bolzano, Italy
calvanese@inf.unibz.it

**Thomas Eiter** and **Magdalena Ortiz**
Institute of Information Systems
Vienna University of Technology
Favoritenstraße 9-11, Vienna, Austria
eiter|ortiz@kr.tuwien.ac.at

**Abstract**

Expressive Description Logics (DLs) have been advocated as formalisms for modeling the domain of interest in various application areas. An important requirement is the ability to answer complex queries beyond instance retrieval, taking into account constraints expressed in a knowledge base. We consider this task for positive existential path queries (which generalize conjunctive queries and unions thereof), whose atoms are regular expressions over the roles (and concepts) of a knowledge base in the expressive DL $\mathcal{ALCQIb}_{reg}$. Using techniques based on two-way tree-automata, we first provide an elegant characterization of TBox and ABox reasoning, which gives us also a tight EXPTIME bound. We then prove decidability (more precisely, a 2EXPTIME upper bound) of query answering, thus significantly pushing the decidability frontier, both with respect to the query language and the considered DL. We also show that query answering is EXPSPACE-hard already in rather restricted settings.

## Introduction

Description Logics (DLs) (Baader *et al.* 2003) are a well-established branch of logics for knowledge representation and reasoning, and the premier logic-based formalism for modeling concepts (i.e., classes of objects) and roles (i.e., binary relationships between classes). They have gained increasing attention in different areas including the Semantic Web, data and information integration, peer-to-peer data management, and ontology-based data access. In particular, some of the standard Web ontologies from the OWL family are based on DLs (Heflin & Hendler 2001).

In DLs, traditionally reasoning tasks had been studied that deal with taxonomic issues like classification and instance checking. Recently, however, the widening range of applications has led to extensive studies of answering queries over DL knowledge bases (KBs) that require, beyond simple instance retrieval, to join pieces of information in finding the answer. Specifically, *conjunctive queries* have been studied in several papers, cf. (Calvanese *et al*., 1998; 2006; Glimm *et al.*, 2007; Hufstadt *et al.*, 2004; 2005; Ortiz *et al*., 2006). As shown therein, answering (classes of) conjunctive queries is decidable for several DLs, including expressive ones. Glimm *et al.* (2007) proved this for arbitrary conjunctive queries over $\mathcal{SHIQ}$ KBs, while Hustadt *et al.* (2004; 2005) showed this for conjunctive queries without transitive roles and Ortiz *et al.* (2006) for unions of such queries.[1]

At present, (unions of) conjunctive queries over $\mathcal{SHIQ}$ KBs is among the most expressive decidable settings. In this paper, we push the frontier and establish decidability of query answering for the yet more expressive class of *positive (existential) two-way regular path queries* (in short, P2RPQs) over the expressive DL $\mathcal{ALCQIb}_{reg}$, which is close to $\mathcal{SHIQ}$. P2RPQs are queries inductively built, using conjunction and disjunction, from atoms that are regular expressions over direct and inverse roles (and allow for testing of concepts). They not only subsume conjunctive queries and unions of conjunctive queries, but also unions of conjunctive regular path queries (Calvanese *et al.* 2000).

More specifically, we make the following contributions.

- Different from previous works, which rely on resolution-based transformations to disjunctive datalog or on tableaux-based algorithms, we use automata techniques for query answering in expressive DLs. While the application of automata techniques in DLs is not novel, cf. (Calvanese *et al.*, 2002; Tobies 2001), previous work was concerned with deciding satisfiability of a KB consisting of a TBox only. Here we address the much more involved task of query answering over a KB, which has data in an ABox; incorporating the query is non-obvious.
- The technique we apply is more accessible than the existing ones based on tableaux and resolution. Indeed, it is computational in nature, and directly works on the models of the KB. In this way, we are also able to obtain more general results, which seems more difficult using the other approaches.
- As a first result, we present an automata-based algorithm for checking the satisfiability of a KB (consisting of TBox and ABox) in EXPTIME. This is worst-case optimal.
- Our main result then shows that answering positive existential queries over $\mathcal{ALCQIb}_{reg}$ KBs is feasible in 2EXPTIME. By a reduction of $\mathcal{SHIQ}$ to $\mathcal{ALCQIb}_{reg}$, a similar result follows for $\mathcal{SHIQ}$. This compares well to the N3EXPTIME bound for union of conjunctive queries by Or-

*Work partially supported by the Austrian Science Funds (FWF) project P17212 and by the European Commission under projects REWERSE (IST-2003-506779) and TONES (FP6-7603).


[1] Note that the technique in (Calvanese *et al.*, 1998) for unions of conjunctive regular path queries is actually incomplete.

tiz *et al.* (2006), and the 2EXPTIME bounds for (classes of) conjunctive queries that emerge from (Glimm *et al.*, 2007; Hustadt *et al.*, 2004). On the other hand, we establish an EXPSPACE lower bound for positive existential queries.

Our results indicate that automata-techniques have high potential for advancing the decidability frontier of query answering over expressive DLs, and are a useful tool for analyzing its complexity.

## Preliminaries

**Description Logics.** Concepts and roles in $\mathcal{ALCQIb}_{reg}$ obey the following syntax:

$$\begin{array}{rcl} C, C' & \longrightarrow & A \mid \neg C \mid C \sqcap C' \mid C \sqcup C' \mid \forall R.C \mid \\ & & \exists R.C \mid \geqslant n\, Q.C \mid \leqslant n\, Q.C \\ Q, Q' & \longrightarrow & P \mid P^- \mid Q \cap Q' \mid Q \cup Q' \mid Q \setminus Q' \\ R, R' & \longrightarrow & Q \mid R \cup R' \mid R \circ R' \mid R^* \mid id(C) \end{array}$$

where $A$ denotes an *atomic concept*, $P$ an *atomic role*, $C$ an arbitrary *concept*, and $R$ an arbitrary *role*. We use $Q$ to denote *basic roles*, which are those roles which may occur in number restrictions. W.l.o.g., we assume that "$\setminus$" is applied only to atomic roles and their inverses.

An $\mathcal{ALCQIb}_{reg}$ *knowledge base* (KB) is a pair $\mathcal{K} = \langle \mathcal{A}, \mathcal{T} \rangle$ where $\mathcal{A}$ (the *ABox*) is a set of *assertions* of the form $A(a)$, $P(a, b)$, and $a \neq b$, with $A$ an atomic concept, $P$ an atomic role, and $a$, $b$ *individuals*; and $\mathcal{T}$ (the *TBox*) is a set of *concept inclusion axioms* $C \sqsubseteq C'$ for arbitrary concepts $C$ and $C'$. W.l.o.g. all concepts occurring in $\mathcal{A}$ occur in $\mathcal{T}$. We denote by $\mathcal{C}_\mathcal{K}$ the set of atomic concepts occurring in $\mathcal{K}$, by $\mathcal{R}_\mathcal{K}$ the set of atomic roles occurring in $\mathcal{K}$ and their inverses, and by $\mathcal{J}_\mathcal{K}$ the individuals in $\mathcal{K}$.

The semantics is the standard one (Baader *et al.* 2003). We note that we do *not* adopt the *unique name assumption*. *KB satisfiability* consists in determining whether some interpretation $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ satisfies all assertions in $\mathcal{A}$ and all concept inclusion axioms in $\mathcal{T}$. By internalization (Schild 1991), this is reducible to finding an interpretation $\mathcal{I}$ satisfying $\mathcal{A}$ and such that each individual in $\mathcal{A}$ is in the extension of a concept $C_\mathcal{T}$ representing $\mathcal{T}$.

**Definition 1 (P2RPQs)** *A positive 2-way regular path query (P2RPQ) over a KB $\mathcal{K}$ is a formula $\exists \vec{x}.\varphi(\vec{x})$, where $\varphi(\vec{x})$ is built using $\wedge$ and $\vee$ from atoms of the form $C(z)$ and $R(z, z')$, with $z$, $z'$ variables from $\vec{x}$ or individuals, $C$ is an arbitrary concept, $R$ is an arbitrary role, and where all atomic concepts and roles in $\varphi$ occur in $\mathcal{K}$.*

Note that positive (regular path) queries naturally generalize unions of conjunctive (regular path) queries (Calvanese *et al.* 2000) by allowing for an unrestricted interaction of conjunction and disjunction[2], thus being in general also exponentially more compact.

**Example 2** Consider the query $q$ over a genealogy KB $\mathcal{K}$:

$$\exists x, y, z.\ parent^*{\cdot}parent^{-*}(x, y) \wedge parent^-(x, z) \wedge parent^-(y, z) \wedge male(x) \wedge \neg male(y) \wedge (\neg deity(x) \vee \neg deity(y))$$

Informally, $q$ is true if there are relatives $x$ and $y$ that have a common child, $z$, and if not both of them are deities.

[2]Instead, no negation is allowed, whence the name.

Let $q$ be a P2RPQ, and let $\mathsf{varind}(q)$ denote the set of variables and individuals in $q$. Given an interpretation $\mathcal{I}$, let $\pi : \mathsf{varind}(q) \to \Delta^\mathcal{I}$ be a total function such that $\pi(a) = a^\mathcal{I}$ for each individual $a \in \mathsf{varind}(q)$. We write $\mathcal{I}, \pi \models C(z)$ if $\pi(z) \in C^\mathcal{I}$, and $\mathcal{I}, \pi \models R(z, z')$ if $(\pi(z), \pi(z')) \in R^\mathcal{I}$. Let $\gamma$ be the Boolean expression obtained from $\varphi$ by replacing each atom $\alpha$ in $\varphi$ with **true**, if $\mathcal{I}, \pi \models \alpha$, and with **false** otherwise. We say that $\pi$ is a *match for $\mathcal{I}$ and $q$*, denoted $\mathcal{I}, \pi \models q$, if $\gamma$ evaluates to **true**. We say that $\mathcal{I}$ *satisfies* $q$, written $\mathcal{I} \models q$, if there is a match $\pi$ for $\mathcal{I}$ and $q$. A KB $\mathcal{K}$ *entails* $q$, denoted $\mathcal{K} \models q$, if $\mathcal{I} \models q$ for each model $\mathcal{I}$ of $\mathcal{K}$.

*Query entailment* consists in verifying, given a KB $\mathcal{K}$ and a P2RPQ $q$, whether $\mathcal{K} \models q$. Note that, w.l.o.g., we consider here query entailment for Boolean queries, i.e., queries without free variables, since query answering for non-Boolean queries is polynomially reducible to query entailment.

**Automata on Infinite Trees.** Infinite trees are represented as prefix-closed (infinite) sets of words over $\mathbb{N}$ (the set of positive integers). Formally, an *infinite tree* is a set of words $T \subseteq \mathbb{N}^*$, such that if $x{\cdot}c \in T$, where $x \in \mathbb{N}^*$ and $c \in \mathbb{N}$, then also $x \in T$. The elements of $T$ are called *nodes*, the empty word $\varepsilon$ is its *root*. For every $x \in T$, the nodes $x{\cdot}c$, with $c \in \mathbb{N}$, are the *successors* of $x$. By convention, $x{\cdot}0 = x$, and $x{\cdot}i{\cdot}{-1} = x$. The *branching degree* $d(x)$ of a node $x$ is the number of its successors. If $d(x) \leq k$ for each node $x$ of $T$, then $T$ has *branching degree* $k$. An *infinite path* $P$ of $T$ is a prefix-closed set $P \subseteq T$ where for every $i \geq 0$ there exists a unique node $x \in P$ with $|x| = i$. A *labeled tree* over an alphabet $\Sigma$ is a pair $(T, V)$, where $T$ is a tree and $V : T \to \Sigma$ maps each node of $T$ to an element of $\Sigma$.

Let $\mathcal{B}(I)$ be the set of positive Boolean formulas built inductively from **true**, **false**, and atoms from a set $I$ applying $\wedge$ and $\vee$. A set $J \subseteq I$ *satisfies* $\varphi \in \mathcal{B}(I)$, if assigning **true** to the atoms in $J$ and **false** to those in $I \setminus J$ makes $\varphi$ true.

A *two-way alternating tree automaton* (2ATA) running over infinite trees with branching degree $k$, is a tuple $\mathbf{A} = \langle \Sigma, Q, \delta, q_0, F \rangle$, where $\Sigma$ is the input alphabet; $Q$ is a finite set of states; $\delta : Q \times \Sigma \to \mathcal{B}([k] \times Q)$, where $[k] = \{-1, 0, 1, \ldots, k\}$, is the transition function; $q_0 \in Q$ is the initial state; and $F$ specifies the acceptance condition.

The transition function $\delta$ maps a state $q \in Q$ and an input letter $\sigma \in \Sigma$ to a positive Boolean formula $\varphi$. Intuitively, each atom $(c, q')$ in $\varphi$ corresponds to a new copy of the automaton going in the direction given by $c$ and starting in state $q'$. E.g., let $k = 2$ and $\delta(q_1, \sigma) = (1, q_2) \wedge (1, q_3) \vee (-1, q_1) \wedge (0, q_3)$. If $\mathbf{A}$ is in the state $q_1$ and reads the node $x$ labeled with $\sigma$, it proceeds by sending off either two copies, in the states $q_2$ and $q_3$ respectively, to the first successor of $x$ (i.e., $x{\cdot}1$), or one copy in the state $q_1$ to the predecessor of $x$ (i.e., $x{\cdot}{-1}$) and one copy in the state $q_3$ to $x$ itself (i.e., $x{\cdot}0$).

Informally, a run of a 2ATA $\mathbf{A}$ over a labeled tree $(T, V)$ is a labeled tree $(T_r, r)$ in which each node $n$ is labeled by an element $r(n) = (x, q) \in T \times Q$ and describes a copy of $\mathbf{A}$ that is in the state $q$ and reads the node $x$ of $T$; the labels of adjacent nodes must satisfy the transition function of $\mathbf{A}$. Formally, a *run* $(T_r, r)$ is a $T{\times}Q$-labeled tree satisfying:

1. $\varepsilon \in T_r$ and $r(\varepsilon) = (\varepsilon, q_0)$.
2. Let $y \in T_r$, with $r(y) = (x, q)$ and $\delta(q, V(x)) = \varphi$. Then there is a set $S = \{(c_1, q_1), \ldots, (c_h, q_h)\} \subseteq [k] \times Q$ s.t.

- $S$ satisfies $\varphi$ and
- for all $1 \leq i \leq h$, we have that $y{\cdot}i \in T_r$, $x{\cdot}c_i$ is defined, and $r(y{\cdot}i) = (x{\cdot}c_i, q_i)$.

A run $(T_r, r)$ is *accepting*, if it satisfies the *parity* condition that for every infinite path $\pi$, there is an *even* $i$ such that $\mathsf{Inf}(\pi) \cap G_i \neq \emptyset$ and $\mathsf{Inf}(\pi) \cap G_{i-1} = \emptyset$, where $F = (G_1, \ldots, G_m)$ is a finite sequence of sets of states with $G_1 \subseteq \cdots \subseteq G_m = Q$, and $\mathsf{Inf}(\pi) \subseteq Q$ denotes the states that occur infinitely often in $\pi$ (as second components of node labels). The *nonemptiness problem* for 2ATAs is deciding whether the set $\mathcal{L}(\mathbf{A})$ of trees accepted by a given 2ATA $\mathbf{A}$ is nonempty. We make use of the following result.

**Theorem 3 (Vardi 1998)** *For any 2ATA $\mathbf{A}$ with $n$ states, parity condition of length $m$, and input alphabet with $\ell$ elements, nonemptiness of $\mathbf{A}$ is decidable in time exponential in $n$ and polynomial in $m$ and $\ell$. There is a one-way nondeterministic tree automaton (1NTA) $\mathbf{A}_1$ with $2^{O(n)}$ states and parity condition of length $O(m)$ such that $\mathcal{L}(\mathbf{A}) = \mathcal{L}(\mathbf{A}_1)$.*

## Deciding KB satisfiability via automata

For many DLs including $\mathcal{ALCQIb}_{reg}$, the standard reasoning tasks are naturally solvable by tree-automata, thanks to their *tree model property*: each satisfiable concept $C$ has a tree-shaped model. This is similar in the presence of a TBox. For an ABox $\mathcal{A}$ this fails, since the assertions in $\mathcal{A}$ may arbitrarily connect individuals. While a satisfiable $\mathcal{ALCQIb}_{reg}$ KB $\mathcal{K} = \langle \mathcal{A}, \mathcal{T} \rangle$ may lack a tree-shaped model, it always has a forest-shaped *canonical model*, in which each individual is the root of a tree-shaped model of $\mathcal{T}$. This property is usually sufficient to adapt algorithms for concept satisfiability to decide KB satisfiability. In particular, automata-based algorithms have been adapted using the *precompletion* technique (Tobies 2001), in which after a reasoning step on the ABox, automata are used to verify the existence of a tree-shaped model rooted at each ABox individual.

Our approach is different. We represent forest-shaped interpretations as trees $\mathbf{T}$, and encode $\mathcal{K}$ into an automaton $\mathbf{A}_\mathcal{K}$ that accepts $\mathbf{T}$ iff $\mathbf{T}$ corresponds to a canonical model of $\mathcal{K}$. To the best of our knowledge, this is the first algorithm that deals with ABox assertions and individuals directly in the automaton. This enables us to extend the automata-based algorithm also to query answering.

We denote by $CL(C_\mathcal{T})$ the *(syntactic) closure* of $C_\mathcal{T}$ as defined in (Calvanese *et al.*, 2002). Intuitively, it contains all the concepts and roles that may occur when $C_\mathcal{T}$ is decomposed during a run of an automaton on a tree representing a model of $\mathcal{K}$. It contains $C_\mathcal{T}$ and it is closed under subconcepts and their negations. It also contains some basic roles (with their corresponding subroles and negations), and some concepts that may occur when decomposing a subconcept of $C_\mathcal{T}$ in which complex concepts occur (e.g., if $\exists(R \circ R').C \in CL(C_\mathcal{T})$ then $\exists R.\exists R'.C \in CL(C_\mathcal{T})$). We assume that $CL(C_\mathcal{T})$ also contains $a_i$ and $\neg a_i$ for each ABox individual $a_i$, plus $d$ and $\neg d$, where $d$ is a new *dummy* symbol. Note that $|CL(C_\mathcal{T})|$ is linear in the length of $\mathcal{K}$. Sometimes we consider expressions $E$ in *negation normal form*, denoted $nnf(E)$, in which negations are pushed inside as much as possible. We let $CL^{nnf}(C_\mathcal{T}) = \{nnf(E) \mid E \in CL(C_\mathcal{T})\}$.

Every satisfiable $\mathcal{ALCQIb}_{reg}$ concept $C_\mathcal{T}$ has a tree-model with branching degree $k_{C_\mathcal{T}} = O(|CL(C_\mathcal{T})|)$ (Calvanese *et al.*, 2002). Satisfiable $\mathcal{ALCQIb}_{reg}$ KBs have a weaker property:

**Theorem 4** *Every satisfiable $\mathcal{ALCQIb}_{reg}$ KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ has a canonical model $\mathcal{I}$ that comprises a set of tree-shaped models of $C_\mathcal{T}$ with branching degree $k_{C_\mathcal{T}}$, whose roots are the individuals in $\mathcal{A}$ (which might be interconnected).*

We represent such a canonical model $\mathcal{I}$ as a tree $\mathbf{T}_\mathcal{I}$. Let $\mathcal{J}_\mathcal{K} = \{a_1, \ldots, a_m\}$, and let $S = \{t_1, \ldots, t_n\}$ be the set of tree-shaped models of $C_\mathcal{T}$ in $\mathcal{I}$ (each with branching degree $k_{C_\mathcal{T}}$). As in (Calvanese *et al.*, 2002), we represent each such $t_j$ as a labeled tree. Each node $x$ is labeled with a set $\sigma$ that contains the atomic concepts that are true in $x$, and the basic roles that connect the predecessor of $x$ to $x$. The label of the root of $t_j$ also contains the names of the individuals in $\mathcal{J}_\mathcal{K}$ which it interprets, but no basic roles. The root of $\mathbf{T}_\mathcal{I}$ is a new node whose children are the roots of all trees $t_j$. Its label is $\{r\} \cup \{Pij \mid \langle a_i^\mathcal{I}, a_j^\mathcal{I} \rangle \in P^\mathcal{I}\}$. Since each $t_j$ is rooted at some $a_i$, we have $n \leq m$. If $n < m$, the root has $m - n$ dummy children labeled $d$. Note that the branching degree is $|\mathcal{J}_\mathcal{K}|$ at the root and $k_{C_\mathcal{T}}$ at all other levels.

We now construct from $\mathcal{K}$ a 2ATA $\mathbf{A}_\mathcal{K}$ that accepts a given tree $\mathbf{T}$ iff $\mathbf{T} = \mathbf{T}_\mathcal{I}$ for some canonical model $\mathcal{I}$ of $\mathcal{K}$. Calvanese *et al.* (2002) presented an automaton $\mathbf{A}_\mathcal{K} = (\Sigma_\mathcal{K}, S, \delta, s_0, F)$ for deciding concept satisfiability in $\mathcal{ALCQIb}_{reg}$. We adapt and expand $\mathbf{A}_\mathcal{K}$ to handle the ABox. The alphabet is $\Sigma_\mathcal{K} = 2^{\mathcal{C}_\mathcal{K} \cup \mathcal{R}_\mathcal{K} \cup \mathcal{J}_\mathcal{K} \cup \{r\} \cup \{d\} \cup PI}$, where $PI = \{Pij \mid a_i, a_j \in \mathcal{J}_\mathcal{K} \text{ and } P \in \mathcal{R}_\mathcal{K}\}$; the set of states is $S = \{s_0\} \cup CL^{nnf}(C_\mathcal{T}) \cup S_\mathcal{A} \cup S_Q$ where $s_0$ is the initial state. The acceptance condition is $F = (\emptyset, \{\forall R^*.C \in CL^{nnf}(C_\mathcal{T})\})$ (concepts $\exists R^*.C$ are not included in the acceptance condition, and are satisfied in all accepting runs, see (Calvanese *et al.*, 2002)).[3]

Intuitively, when $\mathbf{A}_\mathcal{K}$ is in a state $s \in CL^{nnf}(C_\mathcal{T})$ and visits a node $x$ of the tree, it must check that $s$ holds in $x$. The set $S_\mathcal{A}$ contains states of the form $Qij$ to verify whether ABox individuals $a_i$ and $a_j$ are related by a role $Q$, and states of the form $\langle j, \exists Q.C \rangle$ and $\langle j, \forall Q.C \rangle$ to check whether $a_j$ satisfies a concept of the form $\exists Q.C$ and $\forall Q.C$. The set $S_Q$ contains states of the form $\langle \geqslant n\, Q.C, i, j \rangle$ and $\langle k, \geqslant nQC, i, j \rangle$ for $\geqslant \in \{\geqslant, \leqslant\}$, which check the number restrictions. Intuitively, $i$ stores how many successors of the current node have been navigated, and $j$ how many of them are reached through $Q$ and labeled with $C$. Similarly, the states $\langle k, \geqslant nQC, i, j \rangle$ are used to verify that an individual $a_k$ satisfies the concept $\geqslant nQC$.

The transition function $\delta$ is as follows. First, for each $\sigma \in \Sigma_\mathcal{K}$ with $r \in \sigma$ we define $\delta(s_0, \sigma) = F_1 \wedge \cdots \wedge F_7$ from the initial state $s_0$, which verifies that the root contains $r$; that the level one nodes properly represent the individuals in the ABox ($F_1$–$F_3$); that all ABox assertions are satisfied ($F_4$–$F_6$); and that every non-dummy node at level one is the root of a tree representing a model of $C_\mathcal{T}$ ($F_7$):

[3] We could also use a *Büchi condition* $\{\forall R^*.C \in CL^{nnf}(C_\mathcal{T})\}$.

$$
\begin{aligned}
F_1 &= \textstyle\bigwedge_{1\le i\le|\mathcal{J}_\mathcal{K}|}((\bigvee_{1\le j\le|\mathcal{J}_\mathcal{K}|}(i,a_j)\wedge(i,\neg d))\vee(i,d))\\
F_2 &= \textstyle\bigwedge_{1\le i\le|\mathcal{J}_\mathcal{K}|}\bigvee_{1\le j\le|\mathcal{J}_\mathcal{K}|}(j,a_i)\\
F_3 &= \textstyle\bigwedge_{1\le i<j\le|\mathcal{J}_\mathcal{K}|}(\bigwedge_{1\le k\le|\mathcal{J}_\mathcal{K}|}(i,\neg a_k)\vee(j,\neg a_k))\\
F_4 &= \textstyle\bigwedge_{a_i\neq a_j\in\mathcal{A}}(\bigwedge_{1\le k\le|\mathcal{J}_\mathcal{K}|}(k,\neg a_i)\vee(k,\neg a_j))\\
F_5 &= \textstyle\bigwedge_{A(a_j)\in\mathcal{A}}(\bigvee_{1\le i\le|\mathcal{J}_\mathcal{K}|}(i,a_j)\wedge(i,A))\\
F_6 &= \textstyle\bigwedge_{P(a_i,a_j)\in\mathcal{A}}(0,Pij)\\
F_7 &= \textstyle\bigwedge_{1\le i\le|\mathcal{J}_\mathcal{K}|}((i,nnf(C_\mathcal{T}))\vee(i,d))
\end{aligned}
$$

Additional transitions ensure that $r$ and each $a_i$ do not occur anywhere else in the tree. Then, for each concept in $CL^{nnf}(C_\mathcal{T})$ and each $\sigma\in\Sigma_\mathcal{K}$, there are transitions that recursively decompose concepts and roles, and move to appropriate states of the automaton and nodes of the tree. Concepts $\forall R^*.C$ and $\exists R^*.C$ are propagated using the equivalent concepts $C\sqcap\forall R.\forall R^*.C$ and $C\sqcup\exists R.\exists R^*.C$, respectively. Most of these transitions are as in (Calvanese *et al.*, 2002). To verify that a concept of the form $\forall Q.C$, $\exists Q.C$, $\geqslant n\,Q.C$ or $\leqslant n\,Q.C$ is satisfied by a node $x$, all the nodes that reach or are reachable from $x$ must be navigated. We need different transitions for a node $x$ (*i*) at level one and (*ii*) at all other levels. In case (*ii*), the predecessor and the successors of $x$ are navigated as usual. In case (*i*), the transitions must consider the other individual nodes that are connected to $x$ via some role, which are stored in the root label. Therefore, the transitions must send suitable copies of the automaton to navigate the successors, and send a copy of the automaton up to the root. As an example, we provide the transitions for the quantifiers; the number restrictions are handled similarly. If $\sigma\cap(\mathcal{J}_\mathcal{K}\cup\{d\})\neq\emptyset$, we have transitions:

$$\delta(\exists Q.C,\sigma)=\textstyle\bigvee_{a_j\in\sigma}(-1,\langle j,\exists Q.C\rangle)\vee\bigvee_{1\le i\le k_{C_\mathcal{T}}}((i,Q)\wedge(i,C))$$

$$\begin{aligned}\delta(\forall Q.C,\sigma)=&\textstyle\bigwedge_{a_j\in\sigma}(-1,\langle j,\forall Q.C\rangle)\ \wedge\\ &\textstyle\bigwedge_{1\le i\le k_{C_\mathcal{T}}}((i,nnf(Q))\vee(i,C))\end{aligned}$$

Further, for each $\sigma\in\Sigma_\mathcal{K}$ and $\langle j,\exists Q.C\rangle$, $\langle j,\forall Q.C\rangle$ in $S_\mathcal{A}$,

$$\delta(\langle j,\exists Q.C\rangle,\sigma)=\bigvee_{0\le i\le|\mathcal{J}_\mathcal{K}|}\ (\bigvee_{0\le k\le|\mathcal{J}_\mathcal{K}|}((0,Qjk)\wedge(i,a_k)\wedge(i,C)))$$

$$\delta(\langle j,\forall Q.C\rangle,\sigma)=\bigwedge_{0\le i\le|\mathcal{J}_\mathcal{K}|}\ (\bigwedge_{0\le k\le|\mathcal{J}_\mathcal{K}|}((0,nnf(Q)jk)\vee(i,\neg a_k)\vee(i,C)))$$

Concepts and roles are recursively decomposed. When reaching the atomic level, it is checked whether the node label $\sigma$ contains the corresponding atomic symbol. Thus, for each $s\in\mathcal{C}_\mathcal{K}\cup\mathcal{R}_\mathcal{K}\cup\mathcal{J}_\mathcal{K}\cup d$:

$$\delta(s,\sigma)=\begin{cases}\textbf{true}, & \text{if } s\in\sigma\\ \textbf{false}, & \text{if } s\notin\sigma\end{cases}\qquad\delta(\neg s,\sigma)=\begin{cases}\textbf{true}, & \text{if } s\notin\sigma\\ \textbf{false}, & \text{if } s\in\sigma\end{cases}$$

Further transitions verify whether ABox individuals are connected via some atomic role by checking the label of the root. For each $\sigma\in\Sigma_\mathcal{K}$ and $Pij\in S_\mathcal{A}$ with $P\in\mathcal{R}_\mathcal{K}$:

$$\delta(Pij,\sigma)=\begin{cases}\textbf{true}, & \text{if } (Pij\in\sigma) \text{ or } (P^-ji\in\sigma)\\ \textbf{false}, & \text{otherwise}\end{cases}$$

A run of $\mathbf{A}_\mathcal{K}$ on an infinite tree $\mathbf{T}$ starts at the root, and moves to each individual node to check that $C_\mathcal{T}$ holds there. To this end, $nnf(C_\mathcal{T})$ is recursively decomposed while appropriately navigating the tree, until $\mathbf{A}_\mathcal{K}$ arrives at atomic elements, which are checked locally.

Given a labeled tree $\mathbf{T}=(T,V)$ accepted by $\mathbf{A}_\mathcal{K}$, we define an interpretation $\mathcal{I}_\mathbf{T}$ for $\mathcal{K}$. The domain $\Delta^{\mathcal{I}_\mathbf{T}}$ is given by the nodes $x$ in $\mathbf{T}$ with $a_i\in V(x)$ for some individual $a_i$, and the nodes in $\mathbf{T}$ that are reachable from any such $x$ through the roles. The extensions of concepts and roles are determined by the labels of the nodes in $\mathbf{T}$.

**Lemma 5** *Let $\mathbf{T}$ be a labeled tree accepted by $\mathbf{A}_\mathcal{K}$. Then $\mathcal{I}_\mathbf{T}$ is a model of $\mathcal{K}$.*

Conversely, given a canonical model $\mathcal{I}$ of $\mathcal{K}$, we can construct from it a labeled tree $\mathbf{T}_\mathcal{I}$ that is accepted by $\mathbf{A}_\mathcal{K}$.

**Lemma 6** *$\mathbf{A}_\mathcal{K}$ accepts $\mathbf{T}_\mathcal{I}$ for each canonical model $\mathcal{I}$ of $\mathcal{K}$.*

From Lemmas 5 and 6 and Theorem 4, we get:

**Theorem 7** *An $\mathcal{ALCQIb}_{reg}$ KB $\mathcal{K}$ is satisfiable iff the set of trees accepted by $\mathbf{A}_\mathcal{K}$ is nonempty.*

Under unary encoding of numbers in restrictions, the number of states of $\mathbf{A}_\mathcal{K}$ is polynomial in the size of $\mathcal{K}$. Since $\Sigma_\mathcal{K}$ is single exponential in the size of $\mathcal{K}$, by Theorems 3 and 7 we get an optimal upper bound for KB satisfiability (a matching lower bound holds already for much weaker DLs (Baader *et al.* 2003)).

**Corollary 8** *For $\mathcal{ALCQIb}_{reg}$, KB satisfiability is* EXPTIME*-complete.*

## Query answering via automata

We address now the problem of entailment of P2RPQs in $\mathcal{ALCQIb}_{reg}$ KBs. Consider a (Boolean) P2RPQ $q$ over a KB $\mathcal{K}$. We first show that, in order to check whether $\mathcal{K}\models q$, it is sufficient to restrict attention to canonical models. This is a consequence of the possibility to unravel an arbitrary counterexample model for entailment into a canonical model (cf. Theorem 4), and the fact that since the query does not contain negative information, there will still be no match for the unraveled model and the query.

**Lemma 9** *$\mathcal{K}\not\models q$ if and only if there is a canonical model $\mathcal{I}$ of $\mathcal{K}$ such that $\mathcal{I}\not\models q$.*

This result allows us to exploit tree-automata based techniques also for query entailment. Specifically, we consider trees representing canonical models over an alphabet extended with additional atomic concepts, one for each variable in $q$, each of which is satisfied in a single node of the tree. The intuition behind the use of such trees is that, since the existentially quantified "variables" appear explicitly in the tree, a 2ATA $\mathbf{A}_q$ can easily check the existence of a match for (the interpretation corresponding to) the tree and $q$. We show now how to construct such a 2ATA.

Let $q=\exists\vec{x}.\varphi(\vec{x})$ be a P2RPQ over $\mathcal{K}$, and let $atoms(q)$ be the set of atoms appearing in $q$. We consider $\mathcal{C}_\mathcal{K}$ to be enriched by the set $\mathcal{X}=\{x_1,\ldots,x_n\}$ of variables in $\vec{x}$, and additionally may make use of the ABox individuals in $\mathcal{J}_\mathcal{K}$ in place of atomic concepts.[4] Let then $U=(\bigcup_{P\in\mathcal{R}}(P\cup P^-))^*$. For each $\alpha\in atoms(q)$, we define

$$C_\alpha=\begin{cases}\exists U.(C\sqcap z), & \text{if } \alpha=C(z)\\ \exists U.(z_1\sqcap\exists R.z_2), & \text{if } \alpha=R(z_1,z_2)\end{cases}$$

[4] We do not need to enrich the alphabet of atomic concepts by the ABox individuals though, since they are already in it.

where $z, z_1, z_2 \in \mathcal{J}_\mathcal{K} \cup \mathcal{X}$.

We define the 2ATA $\mathbf{A}_q = (\Sigma_q, S_q, \delta, s_0, F)$ as follows

- $\Sigma_q = \Sigma_\mathcal{K} \cup \mathcal{X}$;
- $S_q$ is defined similarly as for $\mathbf{A}_\mathcal{K}$, except that we use $\bigcup_{\alpha \in atoms(q)} CL^{nnf}(C_\alpha)$ instead of $CL^{nnf}(C_\mathcal{T})$.
- The transitions from the initial state are defined for all labels $\sigma$ containing the symbol $r$ (identifying the root node) as $\delta(s_0, \sigma) = F_1 \wedge F_2 \wedge F_3 \wedge F_q \wedge F_v$, where:
  - $F_1$, $F_2$, and $F_3$ are as for $\mathbf{A}_\mathcal{K}$;
  - $F_q$ is obtained from $\varphi(\vec{x})$ by replacing each atom $\alpha$ with $(0, C_\alpha)$ (and by considering $\wedge$ and $\vee$ as the analogous connectives in a 2ATA transition);
  - $F_v$ checks that each atomic concept $x \in \mathcal{X}$ appears exactly once in the tree (this requires new states in $\mathbf{A}_q$);
- $F$ is defined as for $\mathbf{A}_\mathcal{K}$.

When $\mathbf{A}_q$ is in a state $C_\alpha$ in the root node (the only node labeled $r$), it does not "decompose" $C_\alpha$ as usual. Instead, it checks that the concept $C_\alpha$ is satisfied starting from a node at level one representing ABox individuals. This is done by the following transitions, for each $\alpha \in atoms(q)$ and $\sigma$ containing $\alpha$:

$$\delta(C_\alpha, \sigma) = \bigvee_{1 \leq i \leq |\mathcal{J}_\mathcal{K}|} (i, C_\alpha)$$

Then, $\mathbf{A}_q$ contains transitions analogous to those of $\mathbf{A}_\mathcal{K}$ to check that the various concepts $C_\alpha$ are satisfied. Exploiting that the concepts representing variables are enforced to be satisfied in a single node of the tree, and that under this assumption the concepts $C_\alpha$ correctly represent the atoms of $q$, we can show the following result.

**Lemma 10** *Let $\mathcal{I}_\mathbf{T}$ be the canonical interpretation defined from a tree $\mathbf{T}$ as above. Then $\mathbf{A}_q$ accepts $\mathbf{T}$ iff there is a match for $\mathcal{I}_\mathbf{T}$ and $q$ in which each variable $x$ of $q$ is mapped to the (only) object that is an instance of concept $x$.*

We then convert $\mathbf{A}_q$ into an equivalent 1NTA $\mathbf{A}_q^1$. By Theorem 3, the number of states (resp., parity condition) of $\mathbf{A}_q^1$ is exponential (resp., polynomial) in the number of states of $\mathbf{A}_q$, i.e., in the sum of the size of $q$ and $\mathcal{K}$.

We project out variables from $\mathbf{A}_q^1$ obtaining a 1NTA $\mathbf{A}_q^2$ of size not larger than that of $\mathbf{A}_q^1$. By construction, since $\mathbf{A}_q^2$ is a one-way automaton and it has been obtained from $\mathbf{A}_q^1$ by projecting away the variable symbols $\mathcal{X}$, we have that a tree $\mathbf{T}$ is accepted by $\mathbf{A}_q^2$ iff there is a match for $\mathcal{I}_\mathbf{T}$ and $q$.

We complement $\mathbf{A}_q^2$, obtaining a 1NTA $\mathbf{A}_{\neg q}$. The number of states (resp., acceptance condition) of $\mathbf{A}_{\neg q}$ is exponential (resp., polynomial) in the number of states of $\mathbf{A}_q^2$ (Klarlund 1994), i.e., double exponential (resp., polynomial) in the size of $q$ and $\mathcal{K}$. We have that a tree $\mathbf{T}$ is accepted by $\mathbf{A}_{\neg q}$ iff there is no match for $\mathcal{I}_\mathbf{T}$ and $q$.

We construct a 1NTA $\mathbf{A}_{\mathcal{K} \not\models q}$ that accepts the intersection of the languages accepted by $\mathbf{A}_\mathcal{K}$ and $\mathbf{A}_{\neg q}$. This can be done by first converting $\mathbf{A}_\mathcal{K}$ to a 1NTA whose number of states (resp., acceptance condition) is exponential (resp., linear) in the size of $\mathcal{K}$, and then constructing the product automaton with $\mathbf{A}_{\neg q}$. The number of states (resp., acceptance condition) of $\mathbf{A}_{\mathcal{K} \not\models q}$ is still double exponential (resp., polynomial) in the size of $q$ and $\mathcal{K}$.[5]

Since a tree $\mathbf{T}$ is accepted by $\mathbf{A}_\mathcal{K}$ iff $\mathcal{I}_\mathbf{T}$ is a canonical model of $\mathcal{K}$, while it is accepted by $\mathbf{A}_{\neg q}$ iff there is no match for $\mathcal{I}_\mathbf{T}$ and $q$, every tree accepted by $\mathbf{A}_{\mathcal{K} \not\models q}$ represents a counterexample to $\mathcal{K} \models q$. On the other hand, if a tree $\mathbf{T}$ is not accepted by $\mathbf{A}_{\mathcal{K} \not\models q}$, then either it is not accepted by $\mathbf{A}_\mathcal{K}$, in which case $\mathcal{I}_\mathbf{T}$ is not a model of $\mathcal{K}$, or it is not accepted by $\mathbf{A}_{\neg q}$, in which case it is accepted by $\mathbf{A}_q^2$, i.e., there is a match for $\mathcal{I}_\mathbf{T}$ and $q$. Hence the tree does not represent a counterexample to $\mathcal{K} \models q$. As a consequence, we get:

**Lemma 11** *There exists a canonical counterexample to $\mathcal{K} \models q$ iff the set of trees accepted by $\mathbf{A}_{\mathcal{K} \not\models q}$ is not empty.*

By Lemma 9, and the fact that non-emptiness of 1NTAs can be decided in time linear in the number of states of the automaton and exponential in the acceptance condition, see (Vardi 1998), we get the following result.

**Theorem 12** *For every $\mathcal{ALCQIb}_{reg}$ knowledge base $\mathcal{K}$ and P2RPQ query $q$, we have that $\mathcal{K} \models q$ iff the set of trees accepted by $\mathbf{A}_{\mathcal{K} \not\models q}$ is not empty. Moreover, $\mathcal{K} \models q$ is decidable in double exponential time in the size of $q$ and the number of atomic concepts, roles, and individuals in $\mathcal{K}$ and single exponential in the size of $\mathcal{K}$.*

Our results apply also to $\mathcal{SHIQ}$. Given a $\mathcal{SHIQ}$ KB $\mathcal{K}$, it can be rewritten as an $\mathcal{ALCQIb}_{reg}$ KB $\mathcal{K}'$ expressing the role hierarchy with role conjunction (complex roles are not allowed in the ABox, thus it must be closed w.r.t. the hierarchy) and propagating value restrictions over transitive roles by means of TBox axioms (Tobies 2001). Although this reduction does not preserve query entailment, the models of $\mathcal{K}$ and $\mathcal{K}'$ differ only in the interpretation of transitive roles. For a query $q$, deciding $\mathcal{K} \models q$ can be reduced to deciding $\mathcal{K}' \models q'$, where $q'$ is obtained from $q$ by replacing every transitive role $R$ in $q$ with $R \circ R^*$.

## EXPSPACE-Hardness of Query Answering

In this section, we provide a lower bound on answering PRPQs (i.e., P2RPQs without inverses) over $\mathcal{ALC}$ KBs.

**Theorem 13** *Given a PRPQ $q$ and a $\mathcal{ALC}$ knowledge base $\mathcal{K}$, deciding whether $\mathcal{K} \models q$ is* EXPSPACE*-hard.*

The proof is by a reduction from tiling problems, inspired by a similar reduction to query containment over semi-structured data (Calvanese *et al.* 2000).

A tiling problem consists of a finite set $\Delta$ of tile types, two binary relations $H$ and $V$ over $\Delta$, representing horizontal and vertical adjacency relations, respectively, and two distinguished tile types $t_S, t_F \in \Delta$. Deciding whether for a given a number $n$ in unary, a region of the integer plane of size $2^n \times k$, for some $k$, can be tiled consistently with $H$ and $V$, such that the left bottom tile of the region has type $t_S$ and the right upper tile has type $t_F$, can be shown to be EXPSPACE-complete (van Emde Boas 1997).

We construct an $\mathcal{ALC}$ KB $\mathcal{K}$ and a query $q$ such that $\mathcal{K} \models q$ iff there is no correct tiling, as follows. A tiling is spanned

[5] Note that, if only atomic concepts and (regular expressions over) atomic roles are used in $q$, then the number of states of $\mathbf{A}_{\mathcal{K} \not\models q}$ is single exponential in the size of $\mathcal{K}$.

row by row by a sequence of objects. Each object represents one tile and is connected by a specific role to the next tile. For the connections, we use the following two roles:

- $N$ connecting tiles within the same row;
- $L$ connecting the last tile of row $i$ to the first of row $i+1$.

The properties (i.e., the atomic concepts) attached to an object are the $n$ bits $B_1, \ldots, B_n$ of a counter for its address within the row, and its type. For that, we use pairwise disjoint concepts $D_1, \ldots, D_k$, where $\Delta = \{t_1, \ldots, t_k\}$.

We encode in $\mathcal{K}$ the following two conditions:

1. The first ensures that the counters progress correctly. It consists of $O(n)$ standard axioms involving $B_1, \ldots, B_n$ and $N$, which encode a counter bit by bit. Further axioms ensure that, if at least one bit is 0, there is an $N$ successor but no $L$ successor, and reset the counter otherwise.

$$\begin{aligned} \neg B_1 \sqcup \cdots \sqcup \neg B_n &\sqsubseteq \exists N.\top \sqcap \forall L.\bot \\ B_1 \sqcap \cdots \sqcap B_n &\sqsubseteq \exists L.(\neg B_1 \sqcap \cdots \sqcap \neg B_n) \sqcap \forall N.\bot \end{aligned}$$

2. The second ensures that there are no errors w.r.t. the horizontal adjacency relation $H$. For each tile type $D_i$,

$$D_i \sqsubseteq \bigsqcup_{(D_i, D_j) \in H} (\forall N.D_j \sqcap \forall L.D_j).$$

The query $q$ checks the failure of the vertical adjacency $V$ on the candidate tilings given by the models of $\mathcal{K}$. It asks whether two objects exist at distance $2^n$ (i.e., representing vertically adjacent tiles) with an error according to $V$. That the objects are exactly $2^n$ steps apart is achieved by ensuring that they have the same $n$ bits and are connected by a (possibly void) sequence of $N$-steps, followed by one $L$-step, and by a (possibly void) sequence of $N$-steps. We have

$$q = \exists x, y.\mathit{Vert} \wedge \mathit{Err} \wedge G_1 \wedge \cdots \wedge G_n, \quad \text{where}$$

$$\begin{aligned} \mathit{Vert} &= (N^* \circ L \circ N^*)(x, y), \\ \mathit{Err} &= \bigvee_{(D_i, D_j) \notin V} (D_i(x) \wedge D_j(y)), \\ G_i &= (B_i(x) \wedge B_i(y)) \vee (\neg B_i(x) \wedge \neg B_i(y)), \text{ for } 1 \leq i \leq n. \end{aligned}$$

The complete KB $\mathcal{K}$ entails $q$ iff there is no correct tiling. Note that only $\mathit{Vert}$ uses a regular expression. If we have transitive roles and role hierarchies, we can replace it in $q$ by

$$\begin{aligned} \mathit{Vert}' = &(N_t(x, z_1) \wedge L(z_1, z_2) \wedge N_t(z_2, y)) \vee \\ &(N_t(x, z_1) \wedge L(z_1, y)) \vee (L(x, z_2) \wedge N_t(z_2, y)) \end{aligned}$$

where $N_t$ is a transitive super-role of $N$, and $z_1$ and $z_2$ are existentially quantified variables. This shows that answering positive (existential) queries without regular expressions over KBs in $\mathcal{ALC}$ plus transitive roles and role hierarchies, and hence in $\mathcal{SH}$, is EXPSPACE-hard.

Finally, using an encoding closer to (Calvanese *et al.* 2000) where each tile is a block of $n+1$ objects, and the bits and tile types are encoded by roles, one can show that answering conjunctive regular path queries over KBs which only use existential roles and disjunction is EXPSPACE-hard.

## Conclusion

In this paper, we have substantially pushed the frontier of decidable query answering over expressive DLs, which is an active area of research driven by the growing interest to deploy DLs to various application areas related to AI. As we have shown, the rich class of positive two-way regular path queries (P2RPQs) is decidable for $\mathcal{ALCQI}b_{reg}$ KBs by means of automata-techniques; on the other hand, query answering has an EXPSPACE-lower bound already in settings where one of $\mathcal{K}$ and $Q$ is rather plain.

Recent results show that the 2EXPTIME upper bound we provide in this paper is indeed tight (Lutz 2007). However, such a hardness result makes essential use of inverse roles, and the precise complexity of PRPQs remains open. Finally, it would be interesting to see how far automata-based techniques similar to the ones in this paper can be utilized to push the decidability frontier of query answering in expressive DLs, both on the side of the query and the KB.

## References


Baader, F.; Calvanese, D.; McGuinness, D.; Nardi, D.; and Patel-Schneider, P. F., eds. 2003. *The Description Logic Handbook: Theory, Implementation and Applications*.

Calvanese, D.; De Giacomo, G.; Lenzerini, M.; and Vardi, M. Y. 2000. Containment of conjunctive regular path queries with inverse. In *Proc. of KR 2000*, 176–185.

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

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

Calvanese, D.; De Giacomo, G.; and Lenzerini, M. 2002. 2ATAs make DLs easy. In *Proc. of DL 2002*, 107–118.

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

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. 2004. A decomposition rule for decision procedures by resolution-based calculi. In *Proc. of LPAR 2004*, 21–35.

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

Klarlund, N. 1994. Progress measures, immediate determinacy, and a subset construction for tree automata. *Annals of Pure and Applied Logics* 69(2–3):243–268.

Lutz, C. 2007. Inverse roles make conjunctive queries hard. In *Proc. of DL 2007*.

Ortiz, M. M.; Calvanese, D.; and Eiter, T. 2006. Data complexity of answering unions of conjunctive queries in $\mathcal{SHIQ}$. In *Proc. of DL 2006*.

Schild, K. 1991. A correspondence theory for terminological logics: Preliminary report. In *Proc. of IJCAI'91*.

Tobies, S. 2001. *Complexity Results and Practical Algorithms for Logics in Knowledge Representation*. Ph.D. Dissertation, LuFG Theoretical Computer Science, RWTH-Aachen, Germany.

van Emde Boas, P. 1997. The convenience of tilings. In *Complexity, Logic, and Recursion Theory*, volume 187 of *Lecture Notes in Pure and Applied Mathematics*. 331–363.

Vardi, M. Y. 1998. Reasoning about the past with two-way automata. In *Proc. of ICALP'98*, volume 1443 of *LNCS*, 628–641. Springer.