I N F S Y S

R E S E A R C H

R E P O R T

INSTITUT FÜR INFORMATIONSSYSTEME

ARBEITSBEREICH WISSENSBASIERTE SYSTEME

ANSWERING REGULAR PATH QUERIES IN EXPRESSIVE DESCRIPTION LOGICS VIA ALTERNATING TREE-AUTOMATA

Diego Calvanese Thomas Eiter Magdalena Ortiz

INFSYS RESEARCH REPORT 1843-09-04

DECEMBER 2009

Institut für Informationssysteme
AB Wissensbasierte Systeme
Technische Universität Wien
Favoritenstrassße 9-11
A-1040 Wien, Austria
Tel: +43-1-58801-18405
Fax: +43-1-58801-18493
sek@kr.tuwien.ac.at
www.kr.tuwien.ac.at

TU WIEN

INFSYS Research Report

INFSYS Research Report 1843-09-04, December 2009

Answering Regular Path Queries in Expressive Description Logics via Alternating Tree-Automata

Diego Calvanese,[1] Thomas Eiter,[2] Magdalena Ortiz[3]

Abstract. Expressive Description Logics (DLs) have been advocated as formalisms for modeling the domain of interest in various application areas, including the Semantic Web, data and information integration, peer-to-peer data management, and ontology-based data access. An important requirement there 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 2-way regular path queries (P2RPQs)* over knowledge bases in the expressive DL $\mathcal{ZIQ}$. P2RPQs are more general than conjunctive queries, union of conjunctive queries, and regular path queries from the literature. They allow regular expressions over roles and data joins that require inverse paths. The DL $\mathcal{ZIQ}$ extends the core DL $\mathcal{ALC}$ with qualified number restrictions, inverse roles, safe Boolean role expressions, regular expressions over roles, and concepts of the form $\exists P.\mathsf{Self}$ in the style of the DL $\mathcal{SRIQ}$. Using techniques based on two-way tree-automata, we first provide as a stepping stone an elegant characterization of TBox and ABox satisfiability testing which gives us a tight ExpTime bound for this problem. We then establish a double exponential upper bound for answering P2RPQs over $\mathcal{ZIQ}$ knowledge bases; this bound is tight. Our result significantly pushes the frontier of 2ExpTime decidability of query answering in expressive DLs, both with respect to the query language and the considered DL. Furthermore, by reducing the well known DL $\mathcal{SRIQ}$ to $\mathcal{ZIQ}$ (with an exponential blowup in the size of the knowledge base), we also provide a tight 2ExpTime upper bound for knowledge base satisfiability in $\mathcal{SRIQ}$ and establish the decidability of query answering for this significant fragment of the new OWL 2 standard.

[1]KRDB Research Centre, Free University of Bozen-Bolzano, Italy. E-mail: calvanese@inf.unibz.it
[2]Institute of Information Systems, Vienna University of Technology, Austria. E-mail: eiter@kr.tuwien.ac.at
[3]Institute of Information Systems, Vienna University of Technology, Austria. E-mail: ortiz@kr.tuwien.ac.at

Acknowledgements: We are very grateful to Oliver Carton, Orna Kupferman, and Moshe Vardi for their kind and valuable advise on automata-theoretic questions. We owe particular thanks to Yoad Lustig and Nir Piterman for providing a proof sketch connected to Lemma 5.16.
This work has been partially supported by the Austrian Science Fund (FWF) grant P20840, the Mexican National Council for Science and Technology (CONACYT) grant 187697, and the EU project OntoRule (IST-2009-231875).

Some results of this paper have appeared, in preliminary form, in a conference paper in *Proc. AAAI '07* [16].

Contents

1 Introduction

Description Logics (DLs) [4] are a well-established branch of logics for knowledge representation and reasoning, and today the premier logic-based formalisms 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, many of the standard Web ontologies from the OWL family are based on DLs: the new OWL 2 standard [20] is based on a DL known as $\mathcal{SROIQ}$ [32], whose fragment $\mathcal{SRIQ}$ [31] extends the DL $\mathcal{SHIQ}$ underlying OWL-Lite [5].

In DLs, reasoning tasks like classification and instance checking, which deal with taxonomic issues, had been traditionally studied. However, the widening range of applications in which DLs are used has motivated an increasing interest in query languages whose expressive power goes beyond that of DL concept and role expressions. The aim of such languages is to allow one to join pieces of information in finding the query answer, thus overcoming one of the most significant drawbacks of DLs as languages for data management. Since the initial work of Calvanese et al. [8], many further works have addressed the problem of evaluating complex queries over DL knowledge bases. Special attention has been devoted to *conjunctive queries* (CQs) [2], which are the formal counterpart of the most widely used fragment of SQL (or relational algebra) queries, namely select-project-join queries. CQs over over DL knowledge bases have been studied for many DLs, ranging from weak ones that allow for efficient algorithms, like those of the $\mathcal{EL}$ [40, 49, 39] and DL-Lite families [18], to the very expressive ones of the $\mathcal{ALCH}$ and $\mathcal{SH}$ families, cf. [28, 29, 37, 48].

Another important language for querying knowledge bases is that of *regular path queries* (RPQs) [7, 1, 12], which allow one to ask for pairs of objects that are connected by a path conforming to a regular expression. Due to their ability to express complex navigations in graphs, RPQs are the fundamental mechanism for querying semi-structured data. RPQs are particularly useful when *inverse roles* are allowed to occur in the regular expression, since they can express complex conditions that require to navigate the data, without being constrained by the direction initially chosen by the designer to represent relations between data items. We consider the yet more expressive class of *positive (existential) two-way regular path queries* (in short, P2RPQs), which are inductively built using conjunction and disjunction from atoms that are regular expressions over direct and inverse roles and allow for testing the objects encountered during navigation for membership in concepts. P2RPQs, which subsume CQs and unions thereof, are also a natural generalization of several extensions of RPQs that have been studied by different authors, e.g. [15, 30, 14, 21, 13, 3, 26]. They are, to our knowledge, the most expressive query language considered so far over DL knowledge bases [17, 16].

In this paper, we describe a technique, first presented in [16], for deciding the entailment problem for P2RPQs expressed over $\mathcal{ZIQ}$ knowledge bases. In query entailment, we are given a knowledge base and a Boolean query, i.e., a query that in a given interpretation evaluates either to true or to false, expressed over that knowledge base, and we are asked to determine whether the query evaluates to true in every model of the knowledge base. The DL $\mathcal{ZIQ}$, also known as $\mathcal{ALCQIb}^{\mathsf{Self}}_{reg}$, extends the well known DL $\mathcal{ALCQIb}$ (to which reasoning in $\mathcal{SHIQ}$ can be reduced [55]) with regular role expressions [10], Boolean role inclusion axioms, and concepts of the form $\exists S.\mathsf{Self}$ [31]. By means of a translation that reduces the query entailment problem over $\mathcal{SRIQ}$ KBs to $\mathcal{ZIQ}$ KBs, we also obtain an algorithm for entailment of P2RPQs over $\mathcal{SRIQ}$ knowledge bases. This is the first algorithm for query entailment (and hence for query answering) that allows both for regular expressions and for conjunctions of atoms in the query, while considering, on the DL side, a logic that extends $\mathcal{ALC}$ with inverses and counting and, notably, also supports the kind of complex role inclusions that have been advocated in the new OWL standards [20].

Previously, algorithms for query answering over expressive DLs had used a variety of techniques, ranging from query rewriting [8, 28, 35], over modified tableaux techniques [48], to resolution [36]. We obtain our results by exploiting techniques based on *automata on infinite trees* [53], which have been developed initially in the context of modal logics and program logics [56, 58, 57, 41, 6]. While the application of automata techniques in DLs is not novel, cf. [10, 9, 54], previous work was concerned with deciding satisfiability of a knowledge base consisting of a taxonomy part (TBox) only. Here we address the much more involved task of query answering over a knowledge base, which also has a data part (an ABox). Specifically, we extend previous automata-based algorithms for TBox satisfiability [10, 9] and incorporate the ABox part. Then, to decide query entailment over DL knowledge bases, we build on the ideas of Calvanese et. al. [13], which had been developed in the context of automata on finite words, and extend them to automata over infinite trees. For deciding query entailment, we implement automata operations that rely on transformations between different kinds of automata, which, from a technical point of view, are more challenging in our case than in the case of finite words. The technique we present here has been recently extended to some DLs that support nominals [17].

In this paper, we make the following contributions:

- As a stepping stone to our main results, we first present an automata-based algorithm for checking the satisfiability of an $\mathcal{ZIQ}$ knowledge base that comprises both a TBox and an ABox, and that runs in ExpTime, which is worst-case optimal.

- We then show that answering P2RPQs over $\mathcal{ZIQ}$ knowledge bases is feasible in 2ExpTime. From recent results for answering CQs over $\mathcal{ALCI}$ [44] and $\mathcal{SH}$ [23] KBs, it follows that this is worst case optimal. By the aforementioned reduction, the same bound holds for $\mathcal{SHIQ}$. This shows that, once either inverse roles or role hierarchies and transitivity are allowed, one can significantly extend both the query language and the DL considered without further increasing the worst case complexity of the query entailment problem.

- We provide a rewriting that, with an unavoidable exponential blow-up, translates a $\mathcal{SRIQ}$ KB into an $\mathcal{ZIQ}$ one. In this way we obtain a relevant result: a new tight 2ExpTime upper bound for knowledge base satisfiability in $\mathcal{SRIQ}$, the nominal free-fragment of OWL 2.

- Furthermore, we show that entailment for P2RPQs is decidable over $\mathcal{SRIQ}$ knowledge bases (in fact, the problem is in 3ExpTime); this is the first decidability result for query entailment in an expressive DL with complex role hierarchies, and identifies the first expressive fragment of OWL 2 for which decidability of query entailment has been established. For full OWL 2 (i.e., the DL $\mathcal{SRIOQ}$), decidability of query entailment remains open.

The rest of the paper is organized as follows. We first give some technical preliminaries in Section 2. Then, in Section 3, we discuss in detail some properties of $\mathcal{ZIQ}$ KBs and present some transformations on them that lie at the core of our automata algorithms. In Section 4, we describe the automata-based technique for satisfiability of $\mathcal{ZIQ}$ KBs, and in Section 5 its extension to query entailment. In Section 6, we present the rewriting from $\mathcal{SRIQ}$ to $\mathcal{ZIQ}$, obtaining algorithms for KB satisfiability and query entailment in this logic. In Section 7, we draw final conclusions. In order to increase readability, technical details of some proofs have been moved to an appendix.

2 Preliminaries

In this section, we define the main Description Logic (DL) and the query answering problem considered in this work. We also provide some general preliminaries on automata on infinite trees. Throughout the paper, we use $|X|$ to denote the cardinality of a set X, and $\|X\|$ to denote the length of some string encoding X. For a word w, $|w|$ denotes the length of w, i.e., the number of its symbols.

2.1 The Description Logic $\mathcal{ZIQ}$

$\mathcal{ZIQ}$ is the short name for the DL $\mathcal{ALCQIb}^{\mathsf{Self}}_{reg}$, which extends the well known DL $\mathcal{ALCQIb}$ [55] with regular role expressions, Boolean role inclusion axioms, and concepts of the form $\exists S.\mathsf{Self}$ in the style of $\mathcal{SRIQ}$ [31]. In turn, $\mathcal{ALCQIb}$ extends the basic DL $\mathcal{ALC}$ with qualified number restrictions and inverses, which are available in $\mathcal{SHIQ}$, $\mathcal{SRIQ}$ and other well known DLs, and supports safe Boolean expressions over simple roles. $\mathcal{ZIQ}$ is a slight extension of $\mathcal{ALCQIb}_{reg}$ considered in [10, 16].

Definition 2.1 [Concepts and roles] We consider fixed, countably infinite alphabets $\mathbf{C}$ of *concept names* (also called *atomic concepts*), $\mathbf{R}$ of *role names*, and $\mathbf{I}$ of individual names. We assume that $\mathbf{C}$ contains the special concepts $\top$ *(top)* and $\bot$ *(bottom)*, while $\mathbf{R}$ contains the *top (universal) role* T and the *bottom (empty) role* B.

Concepts C, C', *atomic roles* P, *simple roles* S, S', and *roles* R, R', are formed according to the following syntax, where $A \in \mathbf{C}$, $p \in \mathbf{R}$ and $p \neq \mathsf{T}$.

$$\begin{array}{rcl} C, C' & \longrightarrow & A \mid \neg C \mid C \sqcap C' \mid C \sqcup C' \mid \forall R.C \mid \exists R.C \mid \geqslant n\, S.C \mid \leqslant n\, S.C \mid \exists S.\mathsf{Self} \\ P & \longrightarrow & p \mid p^- \\ S, S' & \longrightarrow & P \mid S \cap S' \mid S \cup S' \mid S \setminus S' \\ R, R' & \longrightarrow & \mathsf{T} \mid S \mid R \cup R' \mid R \circ R' \mid R^* \mid id(C) \end{array}$$

An $\mathcal{ZIQ}$ *expression* is a concept or a role. The set of *subconcepts* (*subroles*) of a given concept (resp., *role*) is defined in the natural way considering the syntactic structure of the concept (resp., *role*). ∎

Definition 2.2 [Knowledge base] A *concept inclusion axiom (CIA)* is of the form $C \sqsubseteq C'$, where C and C' are arbitrary concepts, while a *Boolean role inclusion axiom (BRIA)* is of the form $S \sqsubseteq S'$ where S and S' are simple roles. A *TBox* is a set of CIAs and BRIAs. An *assertion* is of the form $C(a)$, $S(a, b)$, or $a \not\approx b$, where C is a concept, S is a simple role and $a, b \in \mathbf{I}$. An *ABox* is a set of assertions.

A *knowledge base (KB)* is a pair $\mathcal{K} = \langle \mathcal{A}, \mathcal{T} \rangle$ where $\mathcal{T}$ is a TBox and $\mathcal{A}$ is a non-empty ABox.[1] We denote by $\mathbf{C}_\mathcal{K}$, $\mathbf{R}_\mathcal{K}$, and $\mathbf{I}_\mathcal{K}$ the sets of concept names, role names, and individuals occurring in $\mathcal{K}$, respectively. Furthermore, we let $\overline{\mathbf{R}}_\mathcal{K} = \mathbf{R}_\mathcal{K} \cup \{p^- \mid p \in \mathbf{R}_\mathcal{K}\}$. ∎

Definition 2.3 [Semantics] An *interpretation* $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ consists of a non-empty *domain* $\Delta^\mathcal{I}$ and a *valuation function* $\cdot^\mathcal{I}$ that maps each individual $a \in \mathbf{I}$ to an element $a^\mathcal{I} \in \Delta^\mathcal{I}$, each concept name $A \in \mathbf{C}$ to a set $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$, and each role name $p \in \mathbf{R}$ to a set $p^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$, in such a way that $\top^\mathcal{I} = \Delta^\mathcal{I}$, $\bot^\mathcal{I} = \emptyset$, $\mathsf{T}^\mathcal{I} = \Delta^\mathcal{I} \times \Delta^\mathcal{I}$, and $\mathsf{B}^\mathcal{I} = \emptyset$. The function $\cdot^\mathcal{I}$ is inductively extended to all concepts and roles as follows:

[1] If $\mathcal{A} = \emptyset$, we can always add $\top(a)$ to $\mathcal{A}$ for some fresh individual name a.

$$
\begin{array}{rlrl}
(\neg C)^{\mathcal{I}} &= \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}} & (p^{-})^{\mathcal{I}} &= \{(y,x) \mid (x,y) \in p^{\mathcal{I}}\} \\
(C \sqcap C')^{\mathcal{I}} &= C^{\mathcal{I}} \cap C'^{\mathcal{I}} & (S \cap S')^{\mathcal{I}} &= S^{\mathcal{I}} \cap S'^{\mathcal{I}} \\
(C \sqcup C')^{\mathcal{I}} &= C^{\mathcal{I}} \cup C'^{\mathcal{I}} & (R \cup R')^{\mathcal{I}} &= R^{\mathcal{I}} \cup R'^{\mathcal{I}} \\
(\forall R.C)^{\mathcal{I}} &= \{x \mid \forall y.(x,y) \in R^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\} & (S \setminus S')^{\mathcal{I}} &= S^{\mathcal{I}} \setminus S'^{\mathcal{I}} \\
(\exists R.C)^{\mathcal{I}} &= \{x \mid \exists y.(x,y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} & (R \circ R')^{\mathcal{I}} &= R^{\mathcal{I}} \circ R'^{\mathcal{I}} \\
(\geqslant n\, S.C)^{\mathcal{I}} &= \{x \mid |\{y \mid (x,y) \in S^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}| \geq n\} & (R^{*})^{\mathcal{I}} &= (R^{\mathcal{I}})^{*} \\
(\leqslant n\, S.C)^{\mathcal{I}} &= \{x \mid |\{y \mid (x,y) \in S^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}| \leq n\} & (id(C))^{\mathcal{I}} &= \{(x,x) \mid x \in C^{\mathcal{I}}\} \\
(\exists S.\mathsf{Self})^{\mathcal{I}} &= \{x \mid (x,x) \in S^{\mathcal{I}}\}
\end{array}
$$

where $\circ$ denotes composition and $\cdot^{*}$ the reflexive transitive closure of a binary relation. $\mathcal{I}$ *satisfies* (or, is a *model* of)

- a CIA or BRIA $E \sqsubseteq E'$, if $E^{\mathcal{I}} \subseteq E'^{\mathcal{I}}$;
- an assertion $C(a)$, if $a^{\mathcal{I}} \in C^{\mathcal{I}}$, an assertion $S(a,b)$, if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in S^{\mathcal{I}}$, and an assertion $a \not\approx b$, if $a^{\mathcal{I}} \neq b^{\mathcal{I}}$;
- an ABox $\mathcal{A}$, if it satisfies every assertion in $\mathcal{A}$;[2]
- a TBox $\mathcal{T}$, if it satisfies every CIA and BRIA in $\mathcal{T}$;
- a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, if it satisfies both $\mathcal{T}$ and $\mathcal{A}$.

Satisfaction of a CIA, BRIA, assertion, ABox, etc. η is denoted by $\mathcal{I} \models \eta$. *Knowledge base satisfiability* is the problem of deciding, given a KB $\mathcal{K}$, whether there exists an interpretation $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{K}$. ∎

2.2 Query Answering

We next introduce P2RPQs, which naturally generalize conjunctive regular path queries [13] and unions thereof.

Definition 2.4 [P2RPQs] A *positive 2-way regular path query* (P2RPQ) is a formula $\exists \vec{v}.\varphi(\vec{v})$, where $\vec{v}$ is a tuple of variables and $\varphi(\vec{v})$ is built using $\wedge$ and $\vee$ from atoms of the form $C(v)$ and $R(v, v')$, where v, v' are variables from $\vec{v}$ or individuals, C is a concept, and R is a role. If all atomic concepts and roles in φ occur in a KB $\mathcal{K}$, the query is *over* $\mathcal{K}$.

Let $q = \exists \vec{v}.\varphi(\vec{v})$ be a P2RPQ, and let $\mathbf{V}_q$ and $\mathbf{I}_q$ respectively denote the sets of variables and individuals in q. Given an interpretation $\mathcal{I}$, let $\pi : \mathbf{V}_q \cup \mathbf{I}_q \rightarrow \Delta^{\mathcal{I}}$ be a total function such that $\pi(a) = a^{\mathcal{I}}$ for each individual $a \in \mathbf{I}_q$. We write $\mathcal{I}, \pi \models C(v)$ if $\pi(v) \in C^{\mathcal{I}}$, and $\mathcal{I}, \pi \models R(v, v')$ if $(\pi(v), \pi(v')) \in R^{\mathcal{I}}$. Let γ be the Boolean expression obtained from φ by replacing each atom α in φ with **true**, if $\mathcal{I}, \pi \models \alpha$, and with **false** otherwise. We say that π is a *match for* $\mathcal{I}$ *and* q, denoted $\mathcal{I}, \pi \models q$, if γ evaluates to **true**. We say that $\mathcal{I}$ *satisfies* q, written $\mathcal{I} \models q$, if there is a match π 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 $\mathcal{K} \models q$. ∎

P2RPQs are a generalization of *Conjunctive Queries (CQs)*, a well known query language widely studied in databases [19, 2] and, more recently, in DLs [43, 11, 48, 28]. A CQ is a P2RPQ not containing $\vee$, and where no regular role expressions occur. The presence of regular expressions in the query atoms of P2RPQs

$$
\begin{aligned}
\mathit{mortal} &\sqsubseteq \neg \mathit{deity} \\
\top &\sqsubseteq \mathit{male} \sqcup \mathit{female} \\
\mathit{male} &\equiv \neg \mathit{female} \\
\top &\sqsubseteq \exists \mathit{HasFather}.\mathit{male} \sqcap \exists \mathit{HasMother}.\mathit{female} \\
\mathit{HasParent} &\equiv \mathit{HasMother} \cup \mathit{HasFather} \\
\forall \mathit{HasParent}.\mathit{mortal} &\sqsubseteq \mathit{mortal} \\
\mathit{deity} &\sqsubseteq \forall \mathit{HasParent}^*.\mathit{deity}
\end{aligned}
$$

HasParent (Heracles, Zeus)
HasParent (Heracles, Alcmene)
HasParent (Alcmene, Electryon)
HasParent (Electryon, Perseus)
HasParent (Perseus, Zeus)
male (Zeus)
female (Alcmene)
deity (Zeus)
mortal (Alcmene)

Figure 1: The genealogy KB used in Example 2.5

allow to express complex navigations in the models of the given KB, similar to (conjunctive) regular path queries [13].

Example 2.5 We consider a genealogy KB $\mathcal{K}_g = \langle \mathcal{T}, \mathcal{A} \rangle$, where $\mathcal{T}$ contains the CIAs and BRIAs in the left column of Figure 1, while $\mathcal{A}$ contains the assertions in the right column. We use $E \equiv E'$ as a shortcut for $E \sqsubseteq E'$ and $E' \sqsubseteq E$.

The following query q_g is a P2RPQ over $\mathcal{K}_g$:

$$
\begin{aligned}
q_g = \exists v_1, v_2, v_3.\ & \mathit{HasParent}^* \circ \mathit{HasParent}^{-*}(v_1, v_2) \wedge \mathit{HasParent}^{-}(v_1, v_3) \wedge \mathit{HasParent}^{-}(v_2, v_3) \wedge \\
& \mathit{male}(v_1) \wedge \mathit{female}(v_2) \wedge (\neg \mathit{deity}(v_1) \vee \neg \mathit{deity}(v_2))
\end{aligned}
$$

Informally, q_g asks whether there are two individuals (represented by v_1 and v_2) who are relatives (i.e., related by the expression $\mathit{HasParent}^* \circ \mathit{HasParent}^{-*}$) that are not both deities, and who have a common child v_3. Note that $\mathcal{K} \models q$, since $\pi(v_1) = \text{Zeus}^{\mathcal{I}}$, $\pi(v_2) = \text{Alcmene}^{\mathcal{I}}$ and $\pi(v_3) = \text{Heracles}^{\mathcal{I}}$ is a match for q in every model $\mathcal{I}$ of $\mathcal{K}$. ∎

Note that we have restricted out attention to queries that are formulas without free variables, i.e., so called *Boolean* queries. We can consider w.l.o.g. the entailment problem for Boolean queries, since query answering for non-Boolean queries is polynomially reducible to query entailment.[3] Note that the problem of deciding whether a given KB has a model can be trivially reduced to query non-entailment. Indeed, a KB $\mathcal{K}$ is satisfiable iff $\mathcal{K} \not\models \exists v.\bot(v)$.

2.3 Automata on Infinite Trees.

In the rest of this section, we recall the definitions of infinite labeled trees and of two way alternating automata over such trees [57].

Definition 2.6 An *(infinite) tree* is a prefix-closed set $T \subseteq \mathbf{N}^*$ of words over the natural numbers $\mathbf{N}$; if $T \subseteq \{1, \ldots, k\}^*$ for some $k \geq 0$, we call it a k-tree. The elements of T are called *nodes*, the empty word ε is

[2] We do not make the *unique name assumption*, which can be simulated using assertions of the form $a \not\approx b$.

[3] Here we refer to the associated decision problem, i.e., whether a given tuple is in the query answer.

its *root*. For every $x \in T$, the nodes $x{\cdot}c$ with $c \in \mathbf{N}$ are the *successors* of x, and x is the *predecessor* of each $x{\cdot}c$; the *ancestor* relation is the transitive closure of predecessor. 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, and T has *branching degree bounded by* b if $d(x) \leq b$ for each node x of T. If $T = \{1, \ldots, k\}^*$ (i.e., T is a k-tree and each node has exactly k successors), we say that T is k-ary. An *infinite path* π of T is a prefix-closed set $\pi \subseteq T$ where for every $i \geq 0$ there exists a unique node $x \in P$ with $|x| = i$. A *labeled tree* over an alphabet Σ (or simply a Σ-labeled tree) is a pair (T, L), where T is a tree and $L : T \to \Sigma$ maps each node of T to an element of Σ. ∎

Now we define *two-way alternating tree automata* (2ATAs) over infinite trees as introduced in [57], which generalize the standard non-deterministic (one-way) automata on infinite trees (1NTAs) in two ways. First, *alternation* is a generalization of non-determinism that allows for an elegant and compact encoding of decision problems in several logics [45]. Second, *two-way* automata are better suited for logics that have 'backward' operators, like inverse roles, since they may move up on the input tree or stay at the current position. In contrast, one-way automata navigate (infinite) trees in a strictly top-down manner, moving always to the successors of the current node.

Definition 2.7 Given a finite set I, let $\mathcal{B}(I)$ be the set of positive Boolean formulas built inductively using $\wedge$ and $\vee$ from **true**, **false** and atoms from I. A set $J \subseteq I$ *satisfies* a formula $\varphi \in \mathcal{B}(I)$, if assigning **true** to the atoms in J and **false** to those in $I \setminus J$ makes φ true. A *two-way alternating tree automaton* (2ATA) running over k-ary trees is a tuple $\mathbf{A} = \langle \Sigma, \mathrm{Q}, \delta, q_0, F \rangle$, where Σ is the input alphabet; Q is a finite set of states; $\delta : \mathrm{Q} \times \Sigma \to \mathcal{B}([k] \times \mathrm{Q})$, where $[k] = \{-1, 0, 1, \ldots, k\}$, is the transition function; $q_0 \in \mathrm{Q}$ is the initial state; and $F = (G_1, \ldots, G_n)$ with $G_1 \subseteq G_2 \subseteq \cdots \subseteq G_n = \mathrm{Q}$ is a *(parity) acceptance condition*, whose length n is called the *index* of $\mathbf{A}$ and denoted $\mathsf{ind}(\mathbf{A})$. We refer to each component Σ, Q, etc. of $\mathbf{A}$ by $\Sigma(\mathbf{A})$, $Q(\mathbf{A})$, etc., respectively. ∎

The transition function δ maps a state $q \in \mathrm{Q}$ and an input letter $\sigma \in \Sigma$ to a positive Boolean formula φ over the atoms in $[k] \times \mathrm{Q}$. Intuitively, if $\delta(q, \sigma) = \varphi$, then each atom (c, q') in φ corresponds to a new copy of the automaton going in the direction given by c and starting in state q'. For example, 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 σ, it proceeds by sending off either (i) two copies, in the states q_2 and q_3 respectively, to the first successor of x (i.e., $x{\cdot}1$), or (ii) 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., $x{\cdot}0$).

Standard non-deterministic (one-way) automata on infinite trees can be defined as particular 2ATAs, in which the transitions cannot use the directions 0 and -1, but instead the automaton always moves to the k successors of the current node and to a tuple of k states, one for each successor. Such a choice can be expressed as a formula in conjunctive form:

Definition 2.8 A *one-way non-deterministic automaton* (1NTA) running over k-ary trees is a 2ATA $\mathbf{A} = \langle \Sigma, \mathrm{Q}, \delta, q_0, F \rangle$ such that for every $q \in \mathrm{Q}$ and every $\sigma \in \Sigma$, $\delta(q, \sigma)$ is of the form $((1, q_1^1) \wedge \cdots \wedge (k, q_k^1)) \vee \cdots \vee ((1, q_1^j) \wedge \cdots \wedge (k, q_k^j))$, with $j \geq 0$, and $q_\ell^i \in \mathrm{Q}$ for each $1 \leq i \leq j$ and each $1 \leq \ell \leq k$. ∎

Acceptance of 2ATAs is defined in terms of *runs*. Informally, a run of a 2ATA $\mathbf{A}$ over a Σ labeled tree (T, L) is a labeled tree (T_r, r) in which each node n is labeled by an element $r(n) = (x, q) \in T \times \mathrm{Q}$ 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, we define:

Definition 2.9 Let $\mathbf{A} = \langle \Sigma, \mathrm{Q}, \delta, q_0, F \rangle$ be a 2ATA running over k-ary trees. A *run* (T_r, r) of $\mathbf{A}$ over a Σ-labeled k-ary tree (T, L) is a $(T \times \mathrm{Q})$-labeled tree satisfying:

1. $\varepsilon \in T_r$ and $r(\varepsilon) = (\varepsilon, q_0)$.
2. Each $y \in T_r$ *satisfies* δ, i.e., if $r(y) = (x, q)$ and $\delta(q, L(x)) = \varphi$, then there is a (possibly empty) set $W = \{(c_1, q_1), \ldots, (c_n, q_n)\} \subseteq [k] \times \mathrm{Q}$ such that:
   - W satisfies φ, and
   - for every $1 \leq i \leq n$, it holds that $y{\cdot}i \in T_r$, $x{\cdot}c_i$ is defined and $r(y{\cdot}i) = (x{\cdot}c_i, q_i)$.

An infinite path π of T_r *satisfies* the acceptance condition F of $\mathbf{A}$, if there is an *even* $i \geq 0$ such that $\mathsf{Inf}(\pi) \cap G_i \neq \emptyset$ and $\mathsf{Inf}(\pi) \cap G_{i-1} = \emptyset$, where $\mathsf{Inf}(\pi) = \{q \in \mathrm{Q} \mid r(n) = (x, q) \text{ for infinitely many } n \in \pi\}$. The run (T_r, r) is *accepting*, if all its infinite paths satisfy F.

A 2ATA $\mathbf{A}$ *accepts* a Σ-labeled tree $\mathbf{T}$, if there is an accepting run of $\mathbf{A}$ over $\mathbf{T}$; $\mathcal{L}(\mathbf{A})$ denotes the set of all trees that $\mathbf{A}$ accepts. The *nonemptiness problem* is to decide whether $\mathcal{L}(\mathbf{A}) \neq \emptyset$ for a given $\mathbf{A}$. ∎

The following result is well-known.

Theorem 2.10 ([57]) *Nonemptiness of a given 2ATA* $\mathbf{A}$ *running on k-ary trees is decidable in time single exponential in* $|Q(\mathbf{A})|$ *and polynomial in* $|\Sigma(\mathbf{A})|$. *Furthermore, it is possible to construct a 1NTA* $\mathbf{A}_1$ *with* $|Q(\mathbf{A}_1)| \leq 2^{O(|Q(\mathbf{A})|^c)}$ *for some constant c and* $\mathsf{ind}(\mathbf{A}_1) = O(\mathsf{ind}(\mathbf{A}))$ *such that* $\mathcal{L}(\mathbf{A}_1) = \mathcal{L}(\mathbf{A})$.

We will often take intersections of 2ATAs, relying on the fact that this operation is trivial.

Lemma 2.11 *Given any set of 2ATAs* $\mathbf{A}_1, \ldots, \mathbf{A}_n$, *it is possible to construct a 2ATA* $\mathbf{A}$ *with* $|Q(\mathbf{A})| = \sum_{i=1}^{n} |Q(\mathbf{A}_i)| + 1$ *and* $\mathsf{ind}(\mathbf{A}) = \max_{i=1}^{n} \mathsf{ind}(\mathbf{A}_i)$ *such that* $\mathcal{L}(\mathbf{A}) = \bigcap_{i=1}^{n} \mathcal{L}(\mathbf{A}_i)$.

Automata on infinite trees provide elegant solutions for decision problems in temporal and program logics [24], and have been widely exploited for the satisfiability problem of many variants of PDL, the μ-calculus, and similar logics [57, 58]. They have also been explored in DLs, but mostly for deciding *concept satisfiability* [54, 10], given that in many DLs, concepts have tree-shaped models.

3 Normal Form and Canonical Models

In this section we prove some properties of KBs and define key notions that allow us to develop then the automata algorithm for reasoning in $\mathcal{ZIQ}$.

3.1 Normalizing Knowledge Bases

First of all, we will prove a quite simple property of KBs that will be useful later: that they have connected models, in which every node can be reached from the interpretation of some ABox individual by a sequence of roles.

Let $\mathcal{K}$ be a KB. We say that an element $d \in \Delta^{\mathcal{I}}$ is *connected* to an element $d_0 \in \Delta^{\mathcal{I}}$ in an interpretation $\mathcal{I}$ of $\mathcal{K}$, if there is some sequence $d_0, \ldots, d_n$ such that $d = d_n$ and for each $0 \leq i < n$ we have $(d_i, d_{i+1}) \in P^{\mathcal{I}}$ for some $P \in \overline{\mathbf{R}}_{\mathcal{K}}$. An interpretation $\mathcal{I}$ is called *connected*, if each $d \in \Delta^{\mathcal{I}}$ is connected to $a^{\mathcal{I}}$ for some $a \in \mathbf{I}_{\mathcal{K}}$.

Lemma 3.1 *[Connected model property] Let $\mathcal{K}$ be an $\mathcal{ZIQ}$ KB. Then, for every P2RPQ q, $\mathcal{K} \not\models q$ implies that there is a connected model $\mathcal{I}$ of $\mathcal{K}$ with $\mathcal{I} \not\models q$.*

Proof. [Sketch] We simply take some model $\mathcal{I}$ of $\mathcal{K}$ with $\mathcal{I} \not\models q$ and restrict it to the elements d for which the individual a required by the definition above exists; the resulting interpretation $\mathcal{I}'$ is connected by construction. It is trivial to verify that $\mathcal{I}' \models \mathcal{K}$. Indeed, for each $d \in \Delta^{\mathcal{I}}$, removing elements not reachable from d does not alter the satisfaction of any concept at d, nor the participation of d in the extension $R^{\mathcal{I}}$ of any role R occurring in $\mathcal{K}$. Hence no CIA or BRIA is violated in $\mathcal{I}'$. The ABox also remains satisfied, since in $\mathcal{I}'$ all domain elements interpreting some ABox individual remain unchanged, and they participate in the same concepts and roles as in $\mathcal{I}$. Finally, since any query match in $\mathcal{I}'$ would also be a query match in $\mathcal{I}$, $\mathcal{I} \not\models q$ implies $\mathcal{I}' \not\models q$. □

Now we present some simple reductions to rewrite a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ into a *normal form* in which the TBox contains only CIAs, negation appears only at the atomic level, and $\top$, $\bot$, B, T do not appear.

1. **ABox reduction.** $\mathcal{A}$ is transformed into an *extensionally reduced ABox*, i.e., an ABox in which only concept and role names are used:
   - Each assertion $C(a)$, where C is not a concept name, is replaced by $A_C(a)$ for a fresh $A_C \in \mathbf{C}$, and $A_C \sqsubseteq C$ is added to $\mathcal{T}$.
   - Each assertion $S(a, b)$, where S is not a role name, is replaced by $p_S(a, b)$ for a fresh $p_S \in \mathbf{R}$, and $p_S \sqsubseteq S$ is added to $\mathcal{T}$.
2. **BRIA Elimination.** Each BRIA $S \sqsubseteq S'$ in $\mathcal{T}$ is replaced by $\exists(S \setminus S').\top \sqsubseteq \bot$ (cf. [50]).
3. **Elimination of $\top$, $\bot$ and B.** The empty role B is simulated by a fresh role name p_{B}, by adding $\top \sqsubseteq \forall p_{\mathsf{B}}.\bot$ to $\mathcal{T}$. The concepts $\top$ and $\bot$ are simulated via fresh concept names $A_\top$ and $A_\bot$, by adding to $\mathcal{T}$ $A \sqcup \neg A \sqsubseteq A_\top$ and $A_\top \sqsubseteq \neg A_\bot$, for an arbitrary concept name A.
4. **Elimination of T.** We add assertions $p_U(a, b)$ to $\mathcal{A}$, for all $a, b \in \mathbf{I}_{\mathcal{K}}$, where p_U is a fresh role name, and replace in $\mathcal{K}$ each occurrence of T by the role $U = (p_U \cup \{R \mid R \in \overline{\mathbf{R}}_{\mathcal{K}}\})^*$. The rewriting preserves query entailment, since whenever $\mathcal{K} \not\models q$, there is a connected model $\mathcal{I}$ of $\mathcal{K}$ such that $\mathcal{I} \not\models q$ (cf. Lemma 3.1); as $p_U \notin \mathbf{R}_{\mathcal{K}}$, we can assume w.l.o.g. that $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in p_U^{\mathcal{I}}$ for all $a, b \in \mathbf{I}_{\mathcal{K}}$. This and the connectedness of $\mathcal{I}$ imply that $U^{\mathcal{I}} = \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, hence $\mathcal{I}$ is a model of the rewritten KB such that $\mathcal{I} \not\models q$.
5. **Negation Normal Form.** Finally, it is well known that an $\mathcal{ZIQ}$ concept C can be efficiently transformed into *negation normal form* (NNF), i.e., one where $\neg$ is applied only to atomic concepts, and $\setminus$ only to atomic roles.

Definition 3.2 [Normal knowledge bases] An $\mathcal{ZIQ}$ knowledge base $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is *normal*, if $\mathcal{A}$ is extensionally reduced, $\mathcal{T}$ contains only CIAs, $\top$, $\bot$, T, and B do not occur in $\mathcal{K}$, and all concepts in $\mathcal{K}$ are in NNF. ∎

Each of the above transformations is linear and preserves all the properties enforced by the preceding transformations. Since they also preserve query entailment, we obtain:

Proposition 3.3 *Given an $\mathcal{ZIQ}$ KB $\mathcal{K}$, it is possible to construct in time linear in $\|K\|$ a normal knowledge base $\mathcal{K}'$ such that for every P2RPQ q, $\mathcal{K} \models q$ iff $\mathcal{K}' \models q$.*

if	$C \in Cl(D)$	then	$\sim C \in Cl(D)$	if	$\exists S.C \in Cl(D)$	then	$\geqslant 1\, S.C \in Cl(D)$
if	$C \boxdot C' \in Cl(D)$	then	$C,\, C' \in Cl(D)$	if	$\forall S.C \in Cl(D)$	then	$\leqslant 0\, S.\sim C \in Cl(D)$
if	$S \in Cl(D)$	then	$\sim S \in Cl(D)$	if	$\mathrm{Q}(R \cup R').C \in Cl(D)$	then	$\mathrm{Q}R.C,\ \mathrm{Q}R'.C \in Cl(D)$
if	$S \in Cl(D)$	then	$\mathsf{Inv}(S) \in Cl(D)$	if	$\mathrm{Q}(R \circ R').C \in Cl(D)$	then	$\mathrm{Q}R.\mathrm{Q}R'.C \in Cl(D)$
if	$S \bigcirc S' \in Cl(D)$	then	$S,\, S' \in Cl(D)$	if	$\mathrm{Q}R^*.C \in Cl(D)$	then	$\mathrm{Q}R.\mathrm{Q}R^*.C \in Cl(D)$
if	$\exists S.\mathsf{Self} \in Cl(D)$	then	$S \in Cl(D)$	if	$\mathrm{Q}id(C).C' \in Cl(D)$	then	$C,\, C' \in Cl(D)$
if	$\gtrless n\, S.C \in Cl(D)$	then	$S,\, C \in Cl(D)$				

Table 1: Syntactic closure ($\mathrm{Q} \in \{\forall, \exists\}$, $\boxdot \in \{\sqcup, \sqcap\}$, $\bigcirc \in \{\cap, \cup\}$)

3.2 Syntactic Closure

Now we introduce the notion of *syntactic closure* of a concept, which contains all concepts and simple roles that are relevant for deciding its satisfiability. It contains D, it is closed under subconcepts and simple subroles, as well as under negations in NNF, and it is also *Fischer-Ladner closed* in the style of a similarly defined closure for PDL [25].

We define here the closure for the DL $\mathcal{ALCQIB}^{\mathsf{Self}}_{reg}$, which is like $\mathcal{ZIQ}$ but instead of role difference $S \setminus S'$, it has negation $\neg S$ as a simple role constructor. Semantically, $\neg S^{\mathcal{I}} = (\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}) \setminus S^{\mathcal{I}}$, hence $S \setminus S'$ can be expressed as $S \cap \neg S'$. We call an $\mathcal{ALCQIB}^{\mathsf{Self}}_{reg}$ expression *safe*, if it is equivalent to one in $\mathcal{ZIQ}$; unsafe Boolean roles are convenient for a simple definition of syntactic closure.

In what follows, $\sim E$ denotes the NNF of $\neg E$, for E a concept or simple role. The symbol $\gtrless$ is generic for $\geqslant$ or $\leqslant$; Q for $\forall$ and $\exists$, $\boxdot$ for $\sqcap$ and $\sqcup$, and $\bigcirc$ for $\cap$ and $\cup$. For a role name $p \in \mathbf{R}$, the *inverse of* p is p^- and the inverse of p^- is p; the inverse of an atomic role P is denoted $\mathsf{Inv}(P)$. For any simple role S, $\mathsf{Inv}(S)$ denotes the role obtained by replacing each atomic role P occurring in S by its inverse $\mathsf{Inv}(P)$. As usual, C and C', S and S', and R and R' respectively stand for concepts, simple roles, and arbitrary roles.

Definition 3.4 The closure $Cl(D)$ of an $\mathcal{ALCQIB}^{\mathsf{Self}}_{reg}$ concept D is the smallest set $D' \supseteq \{D\}$ closed under the rules of Table 1. ∎

Note that $Cl(D)$ may contain unsafe expressions even if D is an $\mathcal{ZIQ}$ concept, and that $|Cl(D)|$ is linear in the length of D.

3.3 Canonical Model Property

Like many DLs, $\mathcal{ZIQ}$ enjoys some form of *forest model property*. In fact, every satisfiable concept C has a model that can be seen as a tree, possibly having loops at some nodes. This extends to TBoxes. For knowledge bases, we need to suitable extend tree-shaped to forest-shaped models.[4]

First, we observe that in $\mathcal{ZIQ}$ every TBox $\mathcal{T}$ can be *internalized* into an equivalent concept $C_{\mathcal{T}}$, such that the satisfiability of $\mathcal{T}$ can be established by obtaining a model of $C_{\mathcal{T}}$ [51].

Definition 3.5 [TBox internalization, $C_{\mathcal{T}}$] Given a normal $\mathcal{ZIQ}$ knowledge base $\mathcal{K} = \langle \mathcal{A}, \mathcal{T} \rangle$, let

$$C_{\mathcal{T}} = \forall (\textstyle\bigcup_{P \in \overline{\mathbf{R}}_{\mathcal{K}}} P)^*.\ \prod_{C_1 \sqsubseteq C_2 \in \mathcal{T}} (\neg C_1 \sqcup C_2).$$

[4] Unlike the results of the previous subsection, these results do not hold for $\mathcal{ALCQIB}^{\mathsf{Self}}_{reg}$; see [47] for discussion.

■

If $C_{\mathcal{T}}$ holds everywhere in an interpretation $\mathcal{I}$, then $\mathcal{I} \models \mathcal{T}$. Furthermore, if a domain element satisfies $C_{\mathcal{T}}$, then the same holds for every element that is connected to it in $\mathcal{I}$.

Proposition 3.6 *Let $\mathcal{I}$ be an interpretation. If each element $d \in \Delta^{\mathcal{I}}$ is connected to some $d_0 \in C_{\mathcal{T}}^{\mathcal{I}}$, then $\mathcal{I} \models \mathcal{T}$.*

Now we define a *canonical model* of a normal $\mathcal{ZIQ}$ KB $\mathcal{K} = \langle \mathcal{A}, \mathcal{T} \rangle$, in which each ABox individual is the root of a tree and satisfies $C_{\mathcal{T}}$.

Definition 3.7 For $k \geq 0$, an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ for an $\mathcal{ZIQ}$ KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is *k-canonical*, if there is a non-empty finite set $\mathit{Roots}(\mathcal{I}) \subseteq \Delta^{\mathcal{I}}$ such that:

(1) $\{\varepsilon\} \cup \Delta^{\mathcal{I}}$ is a tree. (Note that this implies $\Delta^{\mathcal{I}} \subseteq \mathbf{N}^*$.)

(2) $\mathit{Roots}(\mathcal{I}) = \{a^{\mathcal{I}} \mid a \in \mathbf{I}_{\mathcal{K}}\} \subseteq \mathbf{N}$.

(3) Each element of $\Delta^{\mathcal{I}}$ is of the form $i{\cdot}x$ with $i \in \mathit{Roots}(\mathcal{I})$ and $x \in \{1, \ldots, k\}^*$.

(4) For every pair $x, y \in \Delta^{\mathcal{I}}$ with y of the form $x \cdot i$, there is some atomic role P such that $(x, y) \in P^{\mathcal{I}}$.

(5) If $(x, y) \in p^{\mathcal{I}}$ for some role name p and some $x, y \in \Delta^{\mathcal{I}}$, then either (a) $x, y \in \mathit{Roots}(\mathcal{I})$, or (b) for some $i \in \mathit{Roots}(\mathcal{I})$, x is of form $i{\cdot}w$, y of form $i{\cdot}w'$, and either $w = w'$, or w' is a successor of w, or w' is the predecessor of w.

The elements of $\mathit{Roots}(\mathcal{I})$ are called the *roots* of $\mathcal{I}$. ■

Since every node in a canonical interpretation is connected to a root, by Proposition 3.6, satisfaction of $C_{\mathcal{T}}$ at the roots ensures satisfaction of $\mathcal{T}$.

Proposition 3.8 *Let $\mathcal{I}$ be a k-canonical interpretation for $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, for some $k \geq 0$. Then $\mathcal{I} \models \mathcal{K}$ iff $\mathcal{I} \models \mathcal{A}$ and $\mathit{Roots}(\mathcal{I}) \subseteq C_{\mathcal{T}}^{\mathcal{I}}$.*

Now we can establish the *canonical model property* of $\mathcal{ZIQ}$, by straightforward adaptation of the similar property of related logics [58, 54]. It states that, to decide query entailment, it suffices to consider the canonical models of the given KB $\mathcal{K}$. For a concept D, let $k_D = |\mathit{Cl}(D)| \cdot n_{\max}$, where $n_{\max} = \max(\{n \mid \geqslant n\, S.C \in \mathit{Cl}(D)\} \cup \{0\})$.

Theorem 3.9 (canonical model property) *Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be a normal $\mathcal{ZIQ}$ KB, and let q be a P2RPQ. If $\mathcal{K} \not\models q$, then there is a $k_{C_{\mathcal{T}}}$-canonical model $\mathcal{I}$ of $\mathcal{K}$ such that $\mathcal{I} \not\models q$.*

Proof. [Sketch] Following [6, 57], with minor adaptations to properly handle ABoxes, Boolean role constructs and Self, one can show that any model $\mathcal{I}$ of $\mathcal{K}$ that admits no match for a P2RPQ q can be unraveled into a canonical model $\mathcal{I}'$ that also admits no match. See the Appendix for details. □

By Theorem 3.9, and since KB satisfiability reduces to query non-entailment, $\mathcal{K}$ has a $k_{C_{\mathcal{T}}}$-canonical model whenever it is satisfiable. Hence we can restrict to canonical forest-shaped models for deciding KB satisfiability and query entailment. To solve these problems using tree automata, we represent canonical interpretations as infinite labeled trees, as described in the next sections.

4 Deciding KB satisfiability via automata

The lack of tree-shaped models complicates a straight use of tree automata for KB reasoning, and adaptations are needed to exploit the weaker canonical model property of Section 3.3. An example of such an adaptation is the *pre-completion technique* [54], in which after a reasoning step on the ABox, automata are used to verify the existence of a tree-shaped model of the TBox rooted at each ABox individual. We follow a different approach, introduced in [16], namely to represent forest-shaped canonical interpretations as trees, and to encode $\mathcal{K}$ into an automaton $\mathbf{A}_\mathcal{K}$ that accepts exactly the set of trees that represent canonical models of $\mathcal{K}$. To the best of our knowledge, this is the first approach handling ABox assertions and individuals directly in the automaton; importantly, the resulting automata-based algorithm can be extended to query answering, which we do in Section 5.

4.1 Representing Canonical Models as Trees

In the following, let $\mathcal{K} = \langle \mathcal{A}, \mathcal{T} \rangle$ be a normal $\mathcal{ZIQ}$ KB, and let $b_\mathcal{K} = max(k_{C_\mathcal{T}}, |\mathbf{I}_\mathcal{K}|)$. To represent interpretations for $\mathcal{K}$ we define *interpretation trees*, which are labelled $b_\mathcal{K}$-ary trees. Each node is labelled with a (possibly empty) set of atomic concepts and roles, and special symbols p_{ij} (used to indicate that the pair (i, j) of roots is in the extension of the role p) and p_{Self} (to indicate that a pair (x, x) is in the extension of p). The label of the root ε contains the special identifier r, and its children may contain individual names from $\mathbf{I}_\mathcal{K}$ in their labels; if the latter holds we call them *individual nodes*.

Definition 4.1 Given $\mathcal{K}$, let $PI_\mathcal{K} = \{p_{ij} \mid p \in \mathbf{R}_\mathcal{K} \text{ and } i, j \in \{1, \ldots, b_\mathcal{K}\}\}$, and $PS_\mathcal{K} = \{p_{\mathsf{Self}} \mid p \in \mathbf{R}_\mathcal{K}\}$. An *interpretation tree for* $\mathcal{K}$ is a labeled $b_\mathcal{K}$-ary tree $\mathbf{T} = (T, L)$ over the alphabet

$$\Sigma_\mathcal{K} = 2^{\mathbf{C}_\mathcal{K} \cup \overline{\mathbf{R}}_\mathcal{K} \cup \mathbf{I}_\mathcal{K} \cup \{r\} \cup PI_\mathcal{K} \cup PS_\mathcal{K}}$$

such that:

(t1) $r \in L(\varepsilon)$ and $r \notin L(x)$ for every node $x \in T$ with $x \neq \varepsilon$,

(t2) for every $a \in \mathbf{I}_\mathcal{K}$ there is exactly one node $x \in T$ with $1 \leq x \leq b_\mathcal{K}$ and $a \in L(x)$,

(t3) $\mathbf{I}_\mathcal{K} \cap L(x{\cdot}j) = \emptyset$ for every node $x \neq \varepsilon$ and $j > 0$ such that $x{\cdot}j \in T$. ∎

4.1.1 From Canonical Interpretations to Trees

For a canonical interpretation $\mathcal{I}$, we now define its *tree representation* $\mathbf{T}_\mathcal{I}$, which informally is built as follows. Since the domain of $\mathcal{I}$ is always contained in a $b_\mathcal{K}$-ary tree, we only need to add a root ε and enough 'dummy' nodes to ensure the correct branching.

The interpretations of individuals, concepts and roles are represented using node labels from the alphabet $\Sigma_\mathcal{K}$. Roughly, each element $x \in \Delta^\mathcal{I}$ is labeled with a set $L(x)$ that contains (*i*) the atomic concepts A such that $x \in A^\mathcal{I}$; (*ii*) the atomic roles P connecting the predecessor of x to x, and (*iii*) a special symbol p_{Self} for each p such that $(x, x) \in p^\mathcal{I}$. The label $L(i)$ of each root i of $\mathcal{I}$ contains the names of the individuals in $\mathbf{I}_\mathcal{K}$ it interprets and the atomic concepts to which i belongs, but it contains no basic roles. The relations p between individual nodes are stored in the root label $L(\varepsilon)$ via symbols p_{ij}. Formally,

Definition 4.2 Let $\mathcal{I}$ be a canonical interpretation for $\mathcal{K}$ with n roots. The *tree representation of* $\mathcal{I}$ is the interpretation tree $\mathbf{T}_\mathcal{I} = (T, L)$ where

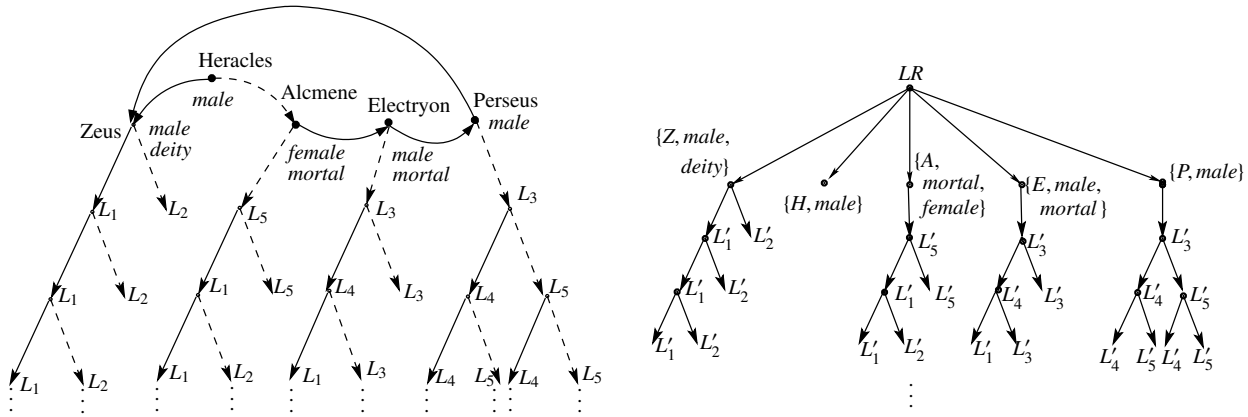

Figure 2: A canonical model and its tree representation

- $T = \{1, \ldots, b_{\mathcal{K}}\}^*$,
- $L(\varepsilon) = \{r\} \cup \{p_{ij} \mid i, j \in Roots(\mathcal{I}), p \in \mathbf{R}_{\mathcal{K}} \text{ and } (i, j) \in p^{\mathcal{I}}\}$,
- for each $1 \leq x \leq b_{\mathcal{K}}$, $L(x) = \{a \in \mathbf{I}_{\mathcal{K}} \mid a^{\mathcal{I}} = x\} \cup \{A \in \mathbf{C}_{\mathcal{K}} \mid x \in A^{\mathcal{I}}\}$,
- for all other nodes x of T (i.e., those with length $|x| > 1$), $L(x) = \{A \in \mathbf{C}_{\mathcal{K}} \mid x \in A^{\mathcal{I}}\} \cup \{p \in \overline{\mathbf{R}}_{\mathcal{K}} \mid (x \cdot -1, x) \in p^{\mathcal{I}}\} \cup \{p_{\mathsf{Self}} \mid p \in \mathbf{R}_{\mathcal{K}} \text{ and } (x, x) \in p^{\mathcal{I}}\}$. ∎

Note that $L(x) = \emptyset$ for every dummy node $x \in T \setminus (\Delta^{\mathcal{I}} \cup \{\varepsilon\})$ not representing a domain element.

Example 4.3 *The left part of Figure 2 depicts part of a canonical model $\mathcal{I}_g$ of $\mathcal{K}_g$. Its roots are* $1 = \text{Zeus}^{\mathcal{I}_g}$, $2 = \text{Heracles}^{\mathcal{I}_g}$, $3 = \text{Alcmene}^{\mathcal{I}_g}$, $4 = \text{Electryon}^{\mathcal{I}_g}$, *and* $5 = \text{Perseus}^{\mathcal{I}_g}$. *They are depicted as large dots, and each of them is labelled with the name of the individual it interprets as well as with the concept names from $C_{\mathcal{K}}$ to whose interpretation it belongs. Other domain elements are represented by smaller dots, and are also labelled with the concept names to whose interpretation they belong. For readability, we use the following label names:* $L_1 = \{male, deity\}$, $L_2 = \{female, deity\}$, $L_3 = \{female, mortal\}$, $L_4 = \{male\}$, *and* $L_5 = \{female\}$. *The interpretation represented here is infinite, but only some of its domain elements are depicted. Every non-root element has two successors, which are the fulfillers of the HasMother and HasFather relations, respectively. The HasFather relation is represented by solid arrows, while HasMother is represented by dashed arrows. HasParent is the union of HasMother and HasFather and is not depicted explicitely.*

The right part of the figure depicts the tree representation of $\mathcal{I}_g$. For readability, we use only the initial letter of each individual name in the labels. The label of the root is $LR = \{r, HasFather_{21}, HasFather_{34}, HasFather_{45}, HasFather_{51}, HasMother_{23}, HasParent_{21}, HasParent_{34}, HasParent_{45}, HasParent_{51}, HasParent_{23}\}$. *The labels of the level one nodes are given explicitely in the figure, while for the other nodes we use the labels* $L'_1 = L_1 \cup \{HasFather, HasParent\}$, $L'_2 = L_2 \cup \{HasMother, HasParent\}$, $L'_3 = L_3 \cup \{HasMother, HasParent\}$, $L'_4 = L_4 \cup \{HasFather, HasParent\}$, *and* $L'_5 = L_5 \cup \{HasMother, HasParent\}$. ∎

4.1.2 From Trees to Canonical Interpretations

With each interpretation tree $\mathbf{T}$, we can in turn associate a canonical interpretation $\mathcal{I}_\mathbf{T}$. Informally, its domain $\Delta^{\mathcal{I}_\mathbf{T}}$ is given by (*i*) the set $\mathbf{I}$ of all the nodes x in $\mathbf{T}$ having some individual a in their label $L(x)$, and (*ii*) the nodes in $\mathbf{T}$ reachable from any such x through the roles in $\mathcal{K}$. Note that each node with an empty label and all nodes below it are not included in the interpretation $\mathcal{I}_\mathbf{T}$.

The extensions of individuals, concepts and roles are determined by the node labels in $\mathbf{T}$. We build the extension of each role name p as the union of two sets of pairs $\mathcal{R}^1_p$ and $\mathcal{R}^2_p$, which respectively contain (*i*) the pairs (x, y) of p-neighbours x, y in the tree where at least one of x, y is not an individual node, and (*ii*) the pairs i, j of individual nodes related by p (which is represented by the special label p_{ij} at the root). Formally,

Definition 4.4 Let $\mathbf{T} = (T, L)$ be an interpretation tree. For each role name $p \in \mathbf{R}_\mathcal{K}$, we define:

$$\begin{aligned}\mathcal{R}^1_p &= \{(x, x{\cdot}i) \mid p \in L(x{\cdot}i)\} \cup \{(x{\cdot}i, x) \mid \mathsf{Inv}(p) \in L(x{\cdot}i)\} \cup \{(x, x) \mid p_{\mathsf{Self}} \in L(x)\}\\ \mathcal{R}^2_p &= \{(i, j) \mid p_{ij} \in L(\varepsilon)\}\end{aligned}$$

Then we let $\mathbf{I}_\mathbf{T} = \{i \in \{1, \dots, b_\mathcal{K}\} \mid a \in L(i) \text{ for some } a \in \mathbf{I}_\mathcal{K}\}$ and, for each $i \in \mathbf{I}_\mathbf{T}$,

$$\mathbf{D}_i = \{\, x' \mid (i, x') \in (\bigcup_{p \in \mathbf{R}_\mathcal{K}} (\mathcal{R}^1_p \cup (\mathcal{R}^1_p)^-))^* \,\},$$

where $(\mathcal{R}^1_p)^-$ denotes the inverse of relation $\mathcal{R}^1_p$.

The *interpretation $\mathcal{I}_\mathbf{T}$ represented by* $\mathbf{T}$ is:

$$\begin{aligned}\Delta^{\mathcal{I}_\mathbf{T}} &= \mathbf{I}_\mathbf{T} \cup \textstyle\bigcup_{i \in \mathbf{I}_\mathbf{T}} \mathbf{D}_i,\\ a^{\mathcal{I}_\mathbf{T}} &= x \in \mathbf{I}_\mathbf{T} \text{ such that } a \in L(x), && \text{for each } a \in \mathbf{I}_\mathcal{K},\\ A^{\mathcal{I}_\mathbf{T}} &= \Delta^{\mathcal{I}_\mathbf{T}} \cap \{\, x \mid A \in L(x) \,\}, && \text{for each concept name } A \in \mathbf{C}_\mathcal{K},\\ p^{\mathcal{I}_\mathbf{T}} &= (\Delta^{\mathcal{I}_\mathbf{T}} \times \Delta^{\mathcal{I}_\mathbf{T}}) \cap (\mathcal{R}^1_p \cup \mathcal{R}^2_p), && \text{for each role name } p \in \mathbf{R}_\mathcal{K}.\end{aligned}$$

■

Note that, by (t2), for each a there is exactly one x such that $a^{\mathcal{I}_\mathbf{T}} = x$. The set $\mathbf{I}_\mathbf{T}$ contains the roots of $\mathcal{I}_\mathbf{T}$, and $\{\varepsilon\} \cup \Delta^{\mathcal{I}_\mathbf{T}}$ is a $b_\mathcal{K}$-tree. It is not hard to verify that $\mathcal{I}_\mathbf{T}$ satisfies the conditions of Definition 3.7.

Lemma 4.5 *If $\mathbf{T}$ is an interpretation tree, then $\mathcal{I}_\mathbf{T}$ is a canonical interpretation, and $\mathbf{T}_{\mathcal{I}_\mathbf{T}} = \mathbf{T}$.*

4.2 Constructing the Automaton

In this section, we show how to construct from $\mathcal{K}$ an automaton $\mathbf{A}_\mathcal{K}$ that accepts the $\Sigma_\mathcal{K}$-labeled trees that are tree representations of canonical models of $\mathcal{K}$. We thus can decide the satisfiability of $\mathcal{K}$ by testing $\mathbf{A}_\mathcal{K}$ for emptiness.

To simplify the technical details, we construct $\mathbf{A}_\mathcal{K}$ in four steps. First, we construct from $\mathcal{K}$ a 2ATA $\mathbf{A}_\mathcal{I}$ that accepts a given tree $\mathbf{T}$ iff it is an interpretation tree for $\mathcal{K}$. In a second step, we construct another 2ATA $\mathbf{A}_\mathcal{A}$ that accepts $\mathbf{T}$ iff the represented interpretation satisfied all assertions in $\mathcal{A}$. We then construct a third 2ATA $\mathbf{A}_\mathcal{T}$ that verifies whether each individual in $\mathbf{I}_K$ satisfies $C_\mathcal{T}$. Finally, by intersecting these three 2ATAs, we obtain $\mathbf{A}_\mathcal{K}$. We note that all these automata run over $b_\mathcal{K}$-ary trees labeled with the alphabet $\Sigma_\mathcal{K}$, given in Definition 4.1.

4.2.1 Automaton $\mathbf{A}_\mathcal{I}$ verifying interpretation trees

We start with the automaton $\mathbf{A}_\mathcal{I}$ that verifies whether an input tree is an interpretation tree.

Definition 4.6 Let $\mathbf{A}_\mathcal{I} = \langle \Sigma_\mathcal{I}, \mathrm{Q}_\mathcal{I}, \delta_\mathcal{I}, q_0^\mathcal{I}, F_\mathcal{I} \rangle$, where $\Sigma_\mathcal{I} = \Sigma_\mathcal{K}$ and

- The set of states is $\mathrm{Q}_\mathcal{I} = \{q_0^\mathcal{I}, s_1\} \cup \{s, \neg s \mid s \in \mathbf{I}_\mathcal{K} \cup \{r\}\}$.
- The transition function $\delta_\mathcal{I} : \mathrm{Q}_\mathcal{I} \times \Sigma_\mathcal{K} \to \mathcal{B}([b_\mathcal{K}] \times \mathrm{Q}_\mathcal{I})$ contains:
  1. for each $\sigma \in \Sigma_\mathcal{K}$ with $r \in \sigma$, a transition
  $$\delta_\mathcal{I}(q_0^\mathcal{I}, \sigma) = B_1 \wedge B_2 \wedge B_3,$$
  where
  $$B_1 = \textstyle\bigwedge_{a \in \mathbf{I}_\mathcal{K}} \bigvee_{1 \le j \le b_\mathcal{K}} (j, a),$$
  $$B_2 = \textstyle\bigwedge_{1 \le i < j \le b_\mathcal{K}} \bigwedge_{a \in \mathbf{I}_\mathcal{K}} ((i, \neg a) \vee (j, \neg a)),$$
  $$B_3 = \textstyle\bigwedge_{1 \le i \le b_\mathcal{K}} ((i, \neg r) \wedge (i, s_1)),$$
  and for each $\sigma \in \Sigma_\mathcal{K}$, a transition
  $$\delta_\mathcal{I}(s_1, \sigma) = \textstyle\bigwedge_{1 \le i \le b_\mathcal{K}} ((i, \neg r) \wedge (\bigwedge_{a \in \mathbf{I}_\mathcal{K}} (i, \neg a)) \wedge (i, s_1));$$
  2. for each $\sigma \in \Sigma_\mathcal{K}$ and each $s \in \mathbf{I}_\mathcal{K} \cup \{r\}$, transitions
  $$\delta_\mathcal{I}(s, \sigma) = \begin{cases} \textbf{true}, & \text{if } s \in \sigma \\ \textbf{false}, & \text{if } s \notin \sigma \end{cases}, \qquad \delta_\mathcal{I}(\neg s, \sigma) = \begin{cases} \textbf{true}, & \text{if } s \notin \sigma \\ \textbf{false}, & \text{if } s \in \sigma \end{cases};$$
- the acceptance condition is $F_\mathcal{I} = (\emptyset, \mathrm{Q}_\mathcal{I})$. ∎

The transitions in 2 simply verify whether the node label σ contains the symbol s. Overall, the definition of $\delta_\mathcal{I}$ ensures that every tree accepted by $\mathbf{A}_\mathcal{I}$ satisfies conditions (t1)–(t3) in Definition 4.2:

- B_1 verifies that the label identifying each individual a occurs in some node of the first level.
- B_2 verifies that a label identifying an individual does not occur in two different first level nodes.
- B_3 checks that the labels of the nodes of the first level do no contain r, and switches from such states to the state s_1. From s_1 it further checks that r and all $a \in \mathbf{I}_\mathcal{K}$ do not appear anywhere below the first level in the whole tree.

Lemma 4.7 $\mathcal{L}(\mathbf{A}_\mathcal{I}) = \{\mathbf{T} \mid \mathbf{T}$ *is an interpretation tree for* $\mathcal{K}\}$.

4.2.2 Automaton $\mathbf{A}_\mathcal{A}$ verifying ABox satisfaction

Next, we define the automaton $\mathbf{A}_\mathcal{A}$ verifying whether the interpretation represented by a given input $\Sigma_\mathcal{K}$-labeled $b_\mathcal{K}$-ary tree $\mathbf{T}$ satisfies all assertions in $\mathcal{A}$, assuming that $\mathbf{T}$ is indeed an interpretation tree. In what follows, we use $\mathbf{C}_\mathcal{A}$ and $\mathbf{R}_\mathcal{A}$ to denote the sets of concept and role names occurring in $\mathcal{A}$, respectively.

Definition 4.8 Let $\mathbf{A}_\mathcal{A} = \langle \Sigma_\mathcal{A}, \mathrm{Q}_\mathcal{A}, \delta_\mathcal{A}, q_0^\mathcal{A}, F_\mathcal{A} \rangle$, where $\Sigma_\mathcal{A} = \Sigma_\mathcal{K}$ and

- The set of states is $\mathrm{Q}_{\mathcal{A}} = \{q_0^{\mathcal{A}}\} \cup \{s, \neg s \mid s \in \mathbf{I}_{\mathcal{K}}\} \cup \mathbf{C}_{\mathcal{A}} \cup \{p_{ij} \mid p \in \mathbf{R}_{\mathcal{A}} \text{ and } i, j \in \{1, \ldots, b_{\mathcal{K}}\}\}$.
- The transition function $\delta_{\mathcal{A}} : \mathrm{Q}_{\mathcal{A}} \times \Sigma_{\mathcal{K}} \to \mathcal{B}([b_{\mathcal{K}}] \times \mathrm{Q}_{\mathcal{A}})$ contains the following transitions:
  1. for each $\sigma \in \Sigma_{\mathcal{K}}$ with $r \in \sigma$, a transition
  $$\delta_{\mathcal{A}}(q_0^{\mathcal{A}}, \sigma) = B_4 \wedge B_5 \wedge B_6,$$
  where
  $$B_4 = \textstyle\bigwedge_{a \neq b \in \mathcal{A}} \bigwedge_{1 \leq k \leq b_{\mathcal{K}}} ((k, \neg a) \vee (k, \neg b)),$$
  $$B_5 = \textstyle\bigwedge_{A(a) \in \mathcal{A}} \bigvee_{1 \leq i \leq b_{\mathcal{K}}} ((i, a) \wedge (i, A)),$$
  $$B_6 = \textstyle\bigwedge_{p(a,b) \in \mathcal{A}} \bigvee_{1 \leq i \leq b_{\mathcal{K}}, 1 \leq j \leq b_{\mathcal{K}}} ((0, p_{ij}) \wedge (i, a) \wedge (j, b));$$
  2. for each $\sigma \in \Sigma_{\mathcal{K}}$, each $s \in \mathbf{I}_{\mathcal{K}}$ and each $t \in \mathbf{C}_{\mathcal{A}} \cup \{p_{ij} \mid p \in \mathbf{R}_{\mathcal{A}} \text{ and } i, j \in \{1, \ldots, b_{\mathcal{K}}\}\}$, transitions
  $$\delta_{\mathcal{A}}(s, \sigma) = \begin{cases} \mathbf{true}, & \text{if } s \in \sigma \\ \mathbf{false}, & \text{if } s \notin \sigma \end{cases}, \quad \delta_{\mathcal{A}}(\neg s, \sigma) = \begin{cases} \mathbf{true}, & \text{if } s \notin \sigma \\ \mathbf{false}, & \text{if } s \in \sigma \end{cases}, \quad \delta_{\mathcal{A}}(t, \sigma) = \begin{cases} \mathbf{true}, & \text{if } t \in \sigma \\ \mathbf{false}, & \text{if } t \notin \sigma \end{cases}.$$
- The acceptance condition is $F_{\mathcal{A}} = (\emptyset, \mathrm{Q}_{\mathcal{A}})$. ∎

Similarly as for $\mathbf{A}_{\mathcal{I}}$, the transitions in item 2 verify the presence of atomic symbols in the node labels. The rest of the transitions verify the following conditions, starting from the root of the input tree:

- B_4 checks, for each assertion $a \not\approx b$ in $\mathcal{A}$, that a and b do not occur both in the label of the same node.
- B_5 ensures that each assertion $A(a)$ in $\mathcal{A}$ is satisfied by verifying that the node labeled a is also labeled A.
- B_6 verifies that each assertion $p(a, b)$ is satisfied, by finding the individual nodes i and j that represent the individuals a and b, and checking p_{ij} at the root.

Hence, assuming that $\mathbf{T}$ is an interpretation tree, the transition function verifies that all the ABox assertions are satisfied in the corresponding interpretation.

Lemma 4.9 *If* $\mathbf{T}$ *is an interpretation tree for a KB* $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$*, then* $\mathbf{T} \in \mathcal{L}(\mathbf{A}_{\mathcal{A}})$ *iff* $\mathcal{I}_{\mathbf{T}} \models \mathcal{A}$.

4.2.3 Automaton $\mathbf{A}_{\mathcal{T}}$ verifying TBox satisfaction

Finally, we define the automaton $\mathbf{A}_{\mathcal{T}}$ that ensures the satisfaction of the TBox $\mathcal{T}$. Recall that $\mathcal{T}$ is satisfied by a canonical interpretation $\mathcal{T}$ iff $i \in C_{\mathcal{T}}^{\mathcal{I}}$ for each root i (see Proposition 3.6). This will be verified by the 2ATA $\mathbf{A}_{\mathcal{T}}$ for the interpretation $\mathcal{I}_{\mathbf{T}}$ represented by an input tree $\mathbf{T}$. The definition of $\mathbf{A}_{\mathcal{T}}$ is rather involved, given that $C_{\mathcal{T}}$ might be complex.

Definition 4.10 Let $\mathbf{A}_{\mathcal{T}} = \langle \Sigma_{\mathcal{T}}, \mathrm{Q}_{\mathcal{T}}, \delta_{\mathcal{T}}, q_0^{\mathcal{T}}, F_{\mathcal{T}} \rangle$, where $\Sigma_{\mathcal{T}} = \Sigma_{\mathcal{K}}$ and

$$
\begin{array}{rcl}
Cl_{ext} & = & Cl(C_{\mathcal{T}}) \cup \{s, \neg s \mid s \in \mathbf{I}_{\mathcal{K}}\} \\
\mathrm{Q}_{\mathsf{Self}} & = & \{S_{\,\mathsf{Self}} \mid S \text{ a simple role in } Cl(C_{\mathcal{T}})\} \\
\mathrm{Q}_{\mathcal{A}\_role} & = & \{S_{\,ij} \mid i, j \in \{1, \ldots, b_{\mathcal{K}}\} \text{ and } S \text{ a simple role in } Cl(C_{\mathcal{T}})\} \\
\mathrm{Q}_{num} & = & \{\langle \geqslant n\, S.C, i, j\rangle \mid \geqslant n\, S.C \in Cl(C_{\mathcal{T}}),\ 0 \leq i \leq b_{\mathcal{K}}{+}1,\ 0 \leq j \leq n\} \cup \\
 & & \{\langle \leqslant n\, S.C, i, j\rangle \mid \leqslant n\, S.C \in Cl(C_{\mathcal{T}}),\ 0 \leq i \leq b_{\mathcal{K}}{+}1,\ 0 \leq j \leq n{+}1\} \\
\mathrm{Q}_{\mathcal{A}\_num} & = & \{\langle a, \geqslant n\, R.C, i, j\rangle \mid a \in \mathbf{I}_{\mathcal{K}},\ \geqslant n\, R.C \in Cl(C_{\mathcal{T}}),\ 1 \leq i \leq b_{\mathcal{K}},\ 0 \leq j \leq n\} \cup \\
 & & \{\langle a, \leqslant n\, R.C, i, j\rangle \mid a \in \mathbf{I}_{\mathcal{K}},\ \leqslant n\, R.C \in Cl(C_{\mathcal{T}}),\ 1 \leq i \leq b_{\mathcal{K}},\ 0 \leq j \leq n{+}1\}
\end{array}
$$

Table 2: State set $\mathrm{Q}_{\mathcal{T}} = \{q_0^{\mathcal{T}}\} \cup Cl_{ext} \cup \mathrm{Q}_{\mathsf{Self}} \cup \mathrm{Q}_{\mathcal{A}\_role} \cup \mathrm{Q}_{num} \cup \mathrm{Q}_{\mathcal{A}\_num}$

- The set of states $\mathrm{Q}_{\mathcal{T}}$ is shown in Table 2. Intuitively, the 'basic' states of $\mathbf{A}_{\mathcal{T}}$ correspond to the concepts in the closure of $\mathcal{T}$, and the automaton moves to a state C and a node x in order to check whether x represents an instance of C. To this aim, C is recursively decomposed and the resulting formula tree navigated. For number restrictions, the automaton must navigate and count the neighbours of x, for which auxiliary states Q_{num} are used. Furthermore, due to the encoding of the ABox at the root, special states $\mathrm{Q}_{\mathcal{A}\_num}$ are used to navigate the neighbours of an individual node. When during the decomposition a simple role S is reached and the automation must verify whether S holds between x and its predecessor, it moves to the state S and S is recursively decomposed (note that S and all its subroles are in Cl_{ext} and thus states of $\mathbf{A}_{\mathcal{T}}$). If it must verify S between individuals nodes i, j, it proceeds similarly but uses the states in $\mathrm{Q}_{\mathcal{A}\_role}$; and if it must verify whether S connects a node to itself, it uses the states in $\mathrm{Q}_{\mathsf{Self}}$. Finally, atomic concepts and roles, the special symbols in $PI_{\mathcal{K}} \cup PS_{\mathcal{K}}$, and the individual names are checked locally at the node labels, using corresponding states. The role played by each state in $\mathrm{Q}_{\mathcal{T}}$ will be detailed in the description of the transitions.

- The transition function $\delta_{\mathcal{T}} : \mathrm{Q}_{\mathcal{T}} \times \Sigma_{\mathcal{K}} \to \mathcal{B}([b_{\mathcal{K}}] \times \mathrm{Q}_{\mathcal{T}})$ consists of five groups of transitions: 1. initialization, 2. concept checking, 3. role checking, 4. number restriction checking, and 5. atomic checks. Here, in 2 and 3, recursive decomposition happens, using states in $Cl(C_{\mathcal{T}})$ and in $Cl(C_{\mathcal{T}}) \cup \mathrm{Q}_{\mathcal{A}\_role} \cup \mathrm{Q}_{\mathsf{Self}}$, respectively, while in 4 the automaton uses auxiliary states in Q_{num} and $\mathrm{Q}_{\mathcal{A}\_num}$ for counting. In detail, the groups of transitions are as follows:

  1. For each $\sigma \in \Sigma_{\mathcal{K}}$ with $r \in \sigma$, there is a transition from the initial state:

     $$\delta_{\mathcal{T}}(q_0, \sigma) \;=\; \textstyle\bigwedge_{1 \leq i \leq b_{\mathcal{K}}}((\bigwedge_{a \in \mathbf{I}_{\mathcal{K}}}(i, \neg a)) \vee (i, nnf(C_{\mathcal{T}}))).$$

     This transition verifies that each individual node is the root of a tree representing a model of $C_{\mathcal{T}}$.

  2. The automaton moves to a node x and a state corresponding to a concept C in $Cl(C_{\mathcal{T}})$ if it wants to verify whether x represents an instance of C. This is achieved with transitions that recursively decompose C and its subconcepts, as well as the non-simple roles inside the existential and universal restrictions, shown on the left side of Table 3. Concepts of the form $\exists S.C$ and $\forall S.C$ with S a simple role are replaced by the equivalent $\geqslant 1\, S.C$ and $\leqslant 0\, S.{\sim}C$, while $\forall R^*.C$ and $\exists R^*.C$, are decomposed into the equivalent $C \sqcap \forall R.\forall R^*.C$ and $C \sqcup \exists R.\exists R^*.C$, respectively.

  3. The automaton moves to a node x and a state for a simple role S in $Cl(C_{\mathcal{T}})$ in order to verify S between the predecessor of x and x. For simple roles S between individual nodes i, j, we use

$$
\begin{aligned}
\delta_{\mathcal{T}}(C \sqcap C', \sigma) &= (0, C) \ \wedge \ (0, C') \\
\delta_{\mathcal{T}}(C \sqcup C', \sigma) &= (0, C) \ \vee \ (0, C') \\
\delta_{\mathcal{T}}(\exists P.\mathsf{Self}, \sigma) &= (0, P_{\mathsf{Self}}) \\
\delta_{\mathcal{T}}(\forall (R \cup R').C, \sigma) &= (0, \forall R.C) \ \wedge \ (0, \forall R'.C) \\
\delta_{\mathcal{T}}(\forall (R \circ R').C, \sigma) &= (0, \forall R.\forall R'.C) \\
\delta_{\mathcal{T}}(\forall R^*.C, \sigma) &= (0, C) \ \wedge \ (0, \forall R.\forall R^*.C) \\
\delta_{\mathcal{T}}(\forall id(C).C', \sigma) &= (0, \sim C) \ \vee \ (0, C') \\
\delta_{\mathcal{T}}(\exists (R \cup R').C, \sigma) &= (0, \exists R.C) \ \vee \ (0, \exists R'.C) \\
\delta_{\mathcal{T}}(\exists (R \circ R').C, \sigma) &= (0, \exists R.\exists R'.C) \\
\delta_{\mathcal{T}}(\exists R^*.C, \sigma) &= (0, C) \ \vee \ (0, \exists R.\exists R^*.C) \\
\delta_{\mathcal{T}}(\exists id(C).C', \sigma) &= (0, C) \ \wedge \ (0, C') \\
\delta_{\mathcal{T}}(\forall S.C, \sigma) &= (0, \leqslant 0\, S.\sim C) \\
\delta_{\mathcal{T}}(\exists S.C, \sigma) &= (0, \geqslant 1\, S.C)
\end{aligned}
$$

$$
\begin{aligned}
\delta_{\mathcal{T}}(S \cap S', \sigma) &= (0, S) \ \wedge \ (0, S') \\
\delta_{\mathcal{T}}(S \cup S', \sigma) &= (0, S) \ \vee \ (0, S') \\
\delta_{\mathcal{T}}(P \setminus P', \sigma) &= (0, P) \ \wedge \ (0, \neg P') \\
\delta_{\mathcal{T}}((S \cap S')_{\mathsf{Self}}, \sigma) &= (0, S_{\mathsf{Self}}) \wedge (0, S'_{\mathsf{Self}}) \\
\delta_{\mathcal{T}}((S \cup S')_{\mathsf{Self}}, \sigma) &= (0, S_{\mathsf{Self}}) \vee (0, S'_{\mathsf{Self}}) \\
\delta_{\mathcal{T}}((P \setminus P')_{\mathsf{Self}}, \sigma) &= (0, P_{\mathsf{Self}}) \wedge (0, \neg P'_{\mathsf{Self}}) \\
\delta_{\mathcal{T}}(p^-_{\mathsf{Self}}, \sigma) &= (0, p_{\mathsf{Self}}) \\
\delta_{\mathcal{T}}((\neg p^-)_{\mathsf{Self}}, \sigma) &= (0, (\neg p)_{\mathsf{Self}}) \\
\delta_{\mathcal{T}}((S \cap S')_{ij}, \sigma) &= (0, S_{ij}) \wedge (0, S'_{ij}) \\
\delta_{\mathcal{T}}((S \cup S')_{ij}, \sigma) &= (0, S_{ij}) \vee (0, S'_{ij}) \\
\delta_{\mathcal{T}}((P \setminus P')_{ij}, \sigma) &= (0, P_{ij}) \wedge (0, (\neg P')_{ij}) \\
\delta_{\mathcal{T}}(p^-_{ij}, \sigma) &= (0, p_{ji}) \\
\delta_{\mathcal{T}}((\neg p^-)_{ij}, \sigma) &= (0, (\neg p)_{ji})
\end{aligned}
$$

Table 3: Transitions in $\mathbf{A}_{\mathcal{T}}$ from groups 2 (left) and 3 (right), where $\sigma \in \Sigma_{\mathcal{K}}$ and $C \in Cl(C_{\mathcal{T}})$ is a non-atomic concept, while a, b are individual names, and S, P and p in $Cl(C_{\mathcal{T}})$ are a simple role, an atomic role, and a role name, respectively.

the state S_{ij} in $\mathrm{Q}_{\mathcal{A}\_role}$, and for reflexive S-loops the state S_{Self} in the set $\mathrm{Q}_{\mathsf{Self}}$; see right side of Table 3.

4. For verifying the satisfaction of a number restriction, we need to navigate all nodes to which the current node can be connected via some role. We say a node y is a *potential neighbor* of a node $x \neq \varepsilon$, if either (a) $y = x$, (b) y is a successor of x, (c) $y \neq \varepsilon$ is the predecessor of x, or (d) both x and y are individual nodes. We say that a node y is a (S, C)-*neighbor* of a node x, if (x, y) is in $S^{\mathcal{I}_{\mathbf{T}}}$ and y is in $C^{\mathcal{I}_{\mathbf{T}}}$.

   To verify that a node x satisfies a number restriction, the automaton traverses all potential neighbors of x, and this requires us to encode counters into the automaton. Intuitively, in a state $\langle \geqslant n\, S.C, i, j\rangle$ of Q_{num}, the number i stores how many potential neighbors have been navigated, and j how many of them are actually (S, C)-neighbors. More precisely, when the automaton is in a state $\geqslant n\, S.C$ and visits a node x, it changes to the state $\langle \geqslant n\, S.C, 0, 0\rangle$, and then navigates the potential neighbors of x, increasing the counters; it will be in state $\langle \geqslant n\, S.C, i, j\rangle$, if j among the first $i - 1$ potential neighbors of x are (S, C)-neighbors of x.

   For each concept $\geqslant n\, S.C$ in $Cl(C_{\mathcal{T}})$ and $\sigma \in \Sigma_{\mathcal{K}}$, we have:

$$\delta_{\mathcal{T}}(\geqslant n\, S.C, \sigma) \ = \ (0, \langle \geqslant n\, S.C, 0, 0\rangle)$$

   Once the counters are set to 0 by this transition, the automaton starts navigating the successors of the node, which are at most $k_{C_{\mathcal{T}}}$. This is done with a set of transitions of the form

$$
\begin{aligned}
\delta_{\mathcal{T}}(\langle \geqslant n\, S.C, i, j\rangle, \sigma) \ = \ & (((i{+}1, \sim S) \vee (i{+}1, \sim C)) \wedge (0, \langle \geqslant n\, S.C, i{+}1, j\rangle)) \vee \\
& ((i{+}1, S) \wedge (i{+}1, C) \wedge (0, \langle \geqslant n\, S.C, i{+}1, j{+}1\rangle))
\end{aligned}
$$

   for all states in Q_{num}, with the counters i and j ranging over the following values:

– $0 \leq i < b_{\mathcal{K}}$ stores how many successors have been counted, and checks the $(i+1)$-th;
– if $\gtrless$ is $\geqslant$, then $0 \leq j < n$: we stop counting if we reach n, as we already know the at-least restriction is satisfied;
– Otherwise, if $\gtrless$ is $\leqslant$, then $0 \leq j \leq n$: similarly, we can stop counting if we reach $n+1$, as we know that the at-least restrictions is not satisfied.

After navigating the successors, the following transitions check whether the current node is its own (S, C)-neighbor, where j is as above (i.e., $0 \leq j < n$ if $\gtrless$ is $\geqslant$, and $0 \leq j \leq n$ otherwise):

$$\begin{aligned}\delta_{\mathcal{T}}(\langle \gtrless n\, S.C, b_{\mathcal{K}}, j\rangle, \sigma) \;=\; & (((0, {\sim}S_{\mathsf{Self}}) \vee (0, {\sim}C)) \wedge (0, \langle \gtrless n\, S.C, b_{\mathcal{K}}{+}1, j\rangle)) \vee \\ & ((0, S_{\mathsf{Self}}) \wedge (0, C) \wedge (0, \langle \gtrless n\, S.C, b_{\mathcal{K}}{+}1, j{+}1\rangle))\end{aligned}$$

If we are at a node that is not an individual node, we finally only have to consider its predecessor. Thus, for each $\sigma \in \Sigma_{\mathcal{K}}$ with $\sigma \cap \mathbf{I}_{\mathcal{K}} = \emptyset$, we let

$$\begin{aligned}\delta_{\mathcal{T}}(\langle \gtrless n\, S.C, b_{\mathcal{K}}{+}1, j\rangle, \sigma) \;=\; & (((0, {\sim}\mathsf{Inv}(S)) \vee (-1, {\sim}C)) \wedge (0, \langle \gtrless n\, S.C, b_{\mathcal{K}}{+}2, j\rangle)) \vee \\ & ((0, \mathsf{Inv}(S)) \wedge (-1, C) \wedge (0, \langle \gtrless n\, S.C, b_{\mathcal{K}}{+}2, j{+}1\rangle))\end{aligned}$$

with j as above (i.e., $0 \leq j < n$ if $\gtrless$ is $\geqslant$, and otherwise $0 \leq j \leq n$).

Otherwise, if the current node is an individual node, we need to navigate all the individual nodes at the first level to count its (S, C)-neighbours. This is done using the states in $\mathrm{Q}_{\mathcal{A}\_num}$ of the form $\langle a, \gtrless n\, Q.C, i, j\rangle$. From an individual node ℓ, the automaton moves to some state $\langle a, \gtrless n\, Q.C, 0, j\rangle$ for some a represented by ℓ. From this state, it uses the counter i to navigate all the level one nodes looking for (S, C)-neighbours of ℓ, and stores in j how many it has found. Note that a can be any individual represented by ℓ, since we only use it to identify it. At each i the automaton verifies whether the root is labelled $S_{\ell i}$ and i is labeled C. It increases the value of i and j if this is the case, and only the value of i otherwise. Hence, for each $\sigma \in \Sigma_{\mathcal{K}}$ such that $\sigma \cap \mathbf{I}_{\mathcal{K}} \neq \emptyset$, we have a transition:

$$\delta_{\mathcal{T}}(\langle \gtrless n\, S.C, b_{\mathcal{K}}{+}1, j\rangle, \sigma) \;=\; \bigvee\nolimits_{a \in \sigma} (-1, \langle a, \gtrless n\, S.C, 0, j\rangle)$$

and for all states in $\mathrm{Q}_{\mathcal{A}\_num}$ and all $\sigma \in \Sigma_{\mathcal{K}}$ with $r \in \sigma$:

$$\begin{aligned}\delta_{\mathcal{T}}(\langle a, \gtrless n\, S.C, i, j\rangle, \sigma) \;=\; & \big(\textstyle\bigvee_{1 \leq \ell \leq b_{\mathcal{K}}} (0, S_{\ell i}) \wedge (\ell, a) \wedge (i, C) \wedge\ (0, \langle a, \gtrless n\, S.C, i{+}1, j{+}1\rangle)\big) \vee \\ & \big(\textstyle\bigwedge_{1 \leq \ell \leq b_{\mathcal{K}}} ((0, {\sim}S_{\ell i}) \vee (\ell, \neg a) \vee (i, \neg C)) \wedge\ (0, \langle a, \gtrless n\, S.C, i{+}1, j\rangle)\big)\end{aligned}$$

with $1 \leq i \leq b_{\mathcal{K}}$ and, similarly as above, $0 \leq j < n$ if $\gtrless$ is $\geqslant$ and $0 \leq j \leq n$ if $\gtrless$ is $\leqslant$.

Once all the necessary nodes have been navigated, the (un)satisfaction of the number restrictions can be established as in Table 4.

5. The above transitions decompose all concepts and roles until they reach states corresponding to possibly negated atomic expressions and symbols in $PI_{\mathcal{K}} \cup PS_{\mathcal{K}} \cup \mathbf{I}_{\mathcal{K}}$; it is then checked whether the expression is contained in the node label σ. Thus, for each $\sigma \in \Sigma_{\mathcal{K}}$ and each $s \in \mathbf{C}_{\mathcal{K}} \cup \overline{\mathbf{R}}_{\mathcal{K}} \cup \mathbf{I}_{\mathcal{K}} \cup PI_{\mathcal{K}} \cup PS_{\mathcal{K}}$, there are transitions:

$$\delta_{\mathcal{T}}(s, \sigma) \;=\; \begin{cases}\mathbf{true}, & \text{if } s \in \sigma \\ \mathbf{false}, & \text{if } s \notin \sigma\end{cases} \qquad \delta_{\mathcal{T}}(\neg s, \sigma) \;=\; \begin{cases}\mathbf{true}, & \text{if } s \notin \sigma \\ \mathbf{false}, & \text{if } s \in \sigma\end{cases}$$

$$
\begin{array}{rcll}
\delta_{\mathcal{T}}(\langle \geqslant n\, S.C, i, n\rangle, \sigma) & = & \textbf{true}, & \text{for } 0 \leq i \leq b_{\mathcal{K}}{+}2 \\
\delta_{\mathcal{T}}(\langle \geqslant n\, S.C, b_{\mathcal{K}}{+}2, j\rangle, \sigma) & = & \textbf{false}, & \text{for } 0 \leq j \leq n{-}1 \\
\delta_{\mathcal{T}}(\langle \leqslant n\, S.C, i, n{+}1\rangle, \sigma) & = & \textbf{false}, & \text{for } 0 \leq i \leq b_{\mathcal{K}}{+}2 \\
\delta_{\mathcal{T}}(\langle \leqslant n\, S.C, b_{\mathcal{K}}{+}2, j\rangle, \sigma) & = & \textbf{true}, & \text{for } 0 \leq j \leq n \\
\delta_{\mathcal{T}}(\langle a, \geqslant n\, S.C, i, n\rangle, \sigma) & = & \textbf{true}, & \text{for } 1 \leq i \leq b_{\mathcal{K}} \\
\delta_{\mathcal{T}}(\langle a, \geqslant n\, S.C, b_{\mathcal{K}}{+}1, j\rangle, \sigma) & = & \textbf{false}, & \text{for } 0 \leq j \leq n{-}1 \\
\delta_{\mathcal{T}}(\langle a, \leqslant n\, S.C, i, n{+}1\rangle, \sigma) & = & \textbf{false}, & \text{for } 1 \leq i \leq b_{\mathcal{K}}{+}1 \\
\delta_{\mathcal{T}}(\langle a, \leqslant n\, S.C, b_{\mathcal{K}}{+}1, j\rangle, \sigma) & = & \textbf{true}, & \text{for } 0 \leq j \leq n
\end{array}
$$

Table 4: Testing the unsatisfiability of number restrictions

- The acceptance condition is $F_{\mathcal{T}} = (\emptyset, \{\forall R^*.C \mid \forall R^*.C \in Cl(C_{\mathcal{T}})\}, \mathrm{Q}_{\mathcal{T}})$. It ensures that there are no paths where satisfaction of $\exists R^*.C$ is indefinitely postponed in accepting runs (cf. [10]). ∎

The transition of groups 2–4 enable $\mathbf{A}_{\mathcal{T}}$ to verify whether a node in the input tree satisfies the concepts in the closure of $C_{\mathcal{T}}$; more precisely, an accepting run of $\mathbf{A}_{\mathcal{T}}$ over an interpretation tree $\mathbf{T}$ can have a node labelled (x, C) iff x is an instance of C in the interpretation $\mathcal{I}_{\mathbf{T}}$.

Lemma 4.11 *For every interpretation tree* $\mathbf{T}$ *for* $\mathcal{K}$*, the following holds:*

1. *If* (T_r, r) *is an accepting run of* $\mathbf{A}_{\mathcal{T}}$ *over* $\mathbf{T}$ *and* $y \in T_r$ *with* $r(y) = (x, C)$ *for* $C \in Cl(C_{\mathcal{T}})$*, then* $x \in C^{\mathcal{I}_{\mathbf{T}}}$.
2. *If for some* $x \in \mathbf{T}$ *and some* $C \in Cl(C_{\mathcal{T}})$*, there does not exist an accepting run* (T_r, r) *of* $\mathbf{A}_{\mathcal{T}}$ *over* $\mathbf{T}$ *such that* $r(y) = (x, C)$ *for some* $y \in T_r$*, then* $x \notin C^{\mathcal{I}_{\mathbf{T}}}$.

Proof. (Sketch) Item 1 can be shown by structural induction on C; the proof is straightforward but tedious, as it must respect the encoding of assertions between individuals and self loops, as well as the correctness of counters for the number restrictions. The only interesting cases are when $C = \forall R^*.D$ or $C = \exists R^*.D$. In the former case, the transitions ensure that for $r(y) = (x, \forall R^*.D)$, the run continues with nodes labeled (x, D) and $(x, \forall R.\forall R^*.D)$, which ensures that $\forall R^*.D$ is propagated to each x' reachable from x via R. This clearly implies $x \in (\forall R^*.D)^{\mathcal{I}_{\mathbf{T}}}$. In the latter case, the transitions propagate $\exists R^*.D$ by either choosing (x, D) or $(x, \exists R.\exists R^*.D)$, which in turn propagates $\exists R^*.D$ to some x' reachable from x via R. To satisfy $\exists R^*.D$, the concept D must be eventually satisfied on some finite R-path starting at x. Towards a contradiction, suppose that D is unsatisfied along every infinite R-path starting at x. As $x \notin D^{\mathcal{I}_{\mathbf{T}}}$, every run compliant with the transition function $\delta_{\mathcal{T}}$ must have successive nodes y', y'' labeled $r(y') = (x, \exists R.\exists R^*.D)$ and $r(y'') = (x', \exists R^*.D)$ for some x' that is reachable from x via R. As $x' \notin D^{\mathcal{I}_{\mathbf{T}}}$, to satisfy $\exists R^*.D$ at x' the disjunct $\exists R.\exists R^*.D$ must be chosen, and the path must continue with nodes z and z' labeled $r(z) = (x', \exists R.\exists R^*.D)$ and $r(z') = (x'', \exists R.\exists R^*.D)$ for some x'' that is reachable from x via R, etc. Consequently, T_r has an infinite path π such that $\mathsf{Inf}(\pi) = \{\exists R^*.D, \exists R.\exists R^*.D\}$. Clearly, π does not satisfy $F_{\mathcal{T}}$; this contradicts that T_r is an accepting run over $\mathcal{T}$. Hence, $r(y) = (x, \exists R.\exists R^*.D)$ implies $x \in (\exists R^*.D)^{\mathcal{I}_{\mathbf{T}}}$, as desired.

Item 2 is also shown by structural induction on C. Roughly, one can show that if a node y in a partial run has $r(y) = (x, C)$ with $x \in C^{\mathcal{I}_{\mathbf{T}}}$, then it is possible to choose a suitable set W as required by Definition 2.9 to satisfy $\delta_{\mathcal{T}}$ at y. Again, the proof is straightforward but tedious because of the large number of constructors and the encodings of assertions between individuals at the root label and of self loops. If $C = \exists R^*.D$ or $C = \forall R^*.D$, the run can require repeatedly generating successors labeled with (x', C). For $C = \forall R^*.D$, this

may be infinitely repeated, but the resulting branch is accepting as $\mathsf{Inf}(\pi) = \{\forall R^*.D, \forall R.\forall R^*.D\}$, and $F_{\mathcal{T}}$ is satisfied. For $\exists R^*.D$, this will happen only finitely often: as $x \in (\exists R^*.D)^{\mathcal{I}_{\mathbf{T}}}$, D must be eventually reached on some finite path, and some $x' \in D^{\mathcal{I}_{\mathbf{T}}}$ will be encountered; we then can add (x', D) to the run. □

$\mathbf{A}_{\mathcal{T}}$ starts its run over an interpretation tree $\mathbf{T}$ at the root ε in state q_0. It then moves to each successor i and switches to the state $C_{\mathcal{T}}$ if i represents some ABox individual a (which is the case if i has some a in its label; otherwise, no further steps from i are made). By Lemma 4.11, it will succeed in completely decomposing $C_{\mathcal{T}}$ at i iff a is in the interpretation of $C_{\mathcal{T}}$ (more precisely, $a^{\mathcal{I}_{\mathbf{T}}} \in C_{\mathcal{T}}^{\mathcal{I}_{\mathbf{T}}}$). Thus using Proposition 3.6, we obtain:

Lemma 4.12 *If* $\mathbf{T}$ *is an interpretation tree for* $\mathcal{K}$*, then* $\mathbf{T} \in \mathcal{L}(\mathbf{A}_{\mathcal{T}})$ *iff* $\mathcal{I}_{\mathbf{T}} \models \mathcal{T}$.

4.2.4 Automaton $\mathbf{A}_{\mathcal{K}}$ verifying KB satisfaction

Finally, by intersecting the automata defined above, we obtain the desired automaton $\mathbf{A}_{\mathcal{K}}$.

Definition 4.13 Given $\mathcal{K}$, let $\mathbf{A}_{\mathcal{K}}$ be an automaton such that $\mathcal{L}(\mathbf{A}_{\mathcal{K}}) = \mathcal{L}(\mathbf{A}_{\mathcal{I}}) \cap \mathcal{L}(\mathbf{A}_{\mathcal{A}}) \cap \mathcal{L}(\mathbf{A}_{\mathcal{T}})$, as in Lemma 2.11. ∎

4.3 Soundness and Completeness

The following proposition states soundness and completeness of $\mathbf{A}_{\mathcal{K}}$ with respect to canonical models of $\mathcal{K}$.

Proposition 4.14 *For a given* $\mathcal{K}$, $\mathcal{L}(\mathbf{A}_{\mathcal{K}}) = \{\mathbf{T}_{\mathcal{I}} \mid \mathcal{I}$ *is a canonical model of* $\mathcal{K}\}$.

Proof. (⊆). By definition, $\mathbf{T} \in \mathcal{L}(\mathbf{A}_{\mathcal{K}})$ implies $\mathbf{T} \in \mathcal{L}(\mathbf{A}_{\mathcal{I}}) \cap \mathcal{L}(\mathbf{A}_{\mathcal{A}}) \cap \mathcal{L}(\mathbf{A}_{\mathcal{T}})$. As $\mathbf{T} \in \mathcal{L}(\mathbf{A}_{\mathcal{I}})$, by Lemma 4.7 $\mathcal{I}_{\mathbf{T}}$ is a canonical interpretation for $\mathcal{K}$. Hence, by Lemma 4.9, $\mathbf{T} \in \mathcal{L}(\mathbf{A}_{\mathcal{A}})$ implies $\mathcal{I}_{\mathbf{T}} \models \mathcal{A}$, and by Lemma 4.12, $\mathbf{T} \in \mathcal{L}(\mathbf{A}_{\mathcal{T}})$ implies $\mathcal{I}_{\mathbf{T}} \models \mathcal{T}$. Consequently, $\mathcal{I}_{\mathbf{T}}$ is a canonical model of $\mathcal{K}$; as by Lemma 4.5 $\mathbf{T} = \mathbf{T}_{\mathcal{I}_{\mathbf{T}}}$, the left-to-right inclusion holds.

(⊇). If $\mathcal{I}$ is a canonical model of $\mathcal{K}$, then $\mathbf{T}_{\mathcal{I}}$ is an interpretation tree. By Lemma 4.7, $\mathbf{A}_{\mathcal{I}}$ accepts $\mathbf{T}_{\mathcal{I}}$. As $\mathcal{I} \models \mathcal{A}$, Lemma 4.9 implies that $\mathbf{A}_{\mathcal{A}}$ accepts $\mathbf{T}_{\mathcal{I}}$. Finally, $\mathcal{I} \models \mathcal{T}$ and Lemma 4.12 imply that $\mathbf{A}_{\mathcal{T}}$ accepts $\mathbf{T}_{\mathcal{I}}$. Consequently, $\mathbf{T}_{\mathcal{I}} \in \mathcal{L}(\mathbf{A}_{\mathcal{K}})$ as desired. □

From Proposition 4.14 and the canonical model property of $\mathcal{ZIQ}$ in Theorem 3.9, we obtain:

Theorem 4.15 *An* $\mathcal{ZIQ}$ *KB* $\mathcal{K}$ *is satisfiable iff* $\mathcal{L}(\mathbf{A}_{\mathcal{K}}) \neq \emptyset$.

Thus, satisfiability of an $\mathcal{ZIQ}$ KB $\mathcal{K}$ reduces to testing the automaton $\mathbf{A}_{\mathcal{K}}$ for emptiness.

4.4 Complexity

Recall that $\mathbf{C}_{\mathcal{K}}$ and $\overline{\mathbf{R}}_{\mathcal{K}}$ denote the atomic concepts and roles occurring in $\mathcal{K}$, respectively, and $\mathbf{I}_{\mathcal{K}}$ the ABox individuals; $b_{\mathcal{K}}$ denotes $max(k_{C_{\mathcal{T}}}, |\mathbf{I}_{\mathcal{K}}|)$ where $k_{C_{\mathcal{T}}} = |Cl(C_{\mathcal{T}})| \cdot \max(\{n \mid \geqslant n\, S.C \in Cl(C_{\mathcal{T}})\} \cup \{0\})$. Furthermore, let $n'_{\max} = \max(\{n \mid \geqslant n\, S.C \in Cl(C_{\mathcal{T}})\} \cup \{0\})$. $\mathcal{K}$ is represented as a string of length $\|\mathcal{K}\|$, and $|\mathbf{C}_{\mathcal{K}}|$, $|\overline{\mathbf{R}}_{\mathcal{K}}|$, $|\mathbf{I}_{\mathcal{K}}|$ and $|Cl(C_{\mathcal{T}})|$ are linear in $\|\mathcal{K}\|$; under unary number coding in the number restrictions, this holds also for $b_{\mathcal{K}}$ and $n'_{\max}$. We thus obtain:

Lemma 4.16 *For* $\mathbf{A}_{\mathcal{K}}$*, we have* $|\Sigma(\mathbf{A}_{\mathcal{K}})| \leq 2^{O(\|\mathcal{K}\|^3)}$, $|Q(\mathbf{A}_{\mathcal{K}})| \leq O(\|\mathcal{K}\|^4)$*, and* $\mathsf{ind}(\mathbf{A}_{\mathcal{K}}) = 3$.

Proof. Recall that $\Sigma(\mathbf{A}_\mathcal{K}) = \Sigma_K$ and $Q(\mathbf{A}_\mathcal{K}) = Q_K$. The result is a consequence of the following simple estimates:

- $|\Sigma_\mathcal{K}| = 2^{M(\mathcal{K})}$, where $M(\mathcal{K}) = |\mathbf{C}_\mathcal{K}| + |\overline{\mathbf{R}}_\mathcal{K}| + |\mathbf{I}_\mathcal{K}| + |PI| + |PS| + 1$, and we have $|PI| = |\overline{\mathbf{R}}_\mathcal{K}| \cdot |b_\mathcal{K}|^2$ and $|PS| = |\overline{\mathbf{R}}_\mathcal{K}|$. Clearly, $M(\mathcal{K}) = O(\|\mathcal{K}\|^3)$.

- $|\mathrm{Q}_\mathcal{K}| \leq |\mathrm{Q}_\mathcal{T}| + |\mathrm{Q}_\mathcal{A}| + |\mathrm{Q}_\mathcal{I}| + 1$, where

$$\begin{aligned}
|\mathrm{Q}_\mathcal{A}| &\leq 1 + 2|\mathbf{I}_\mathcal{K}| + |\mathbf{C}_\mathcal{A}| + |\overline{\mathbf{R}}_\mathcal{A}| \cdot |\mathbf{I}_\mathcal{K}|^2,\\
|\mathrm{Q}_\mathcal{I}| &\leq 2 + 2(|\mathbf{I}_\mathcal{K}| + 1)\\
|\mathrm{Q}_\mathcal{T}| &\leq 1 + |Cl_{ext}| + |\mathrm{Q}_{\mathsf{Self}}| + |\mathrm{Q}_{\mathcal{A}_role}| + |\mathrm{Q}_{num}| + |\mathrm{Q}_{\mathcal{A}_num}|\\
|Cl_{ext}| &\leq |Cl(C_\mathcal{T})| + 2|\mathbf{I}_\mathcal{K}|,\\
|\mathrm{Q}_{\mathsf{Self}}| &\leq |Cl(C_\mathcal{T})|,\\
|\mathrm{Q}_{\mathcal{A}_role}| &\leq |Cl(C_\mathcal{T})| \cdot |b_\mathcal{K}|^2,\\
|\mathrm{Q}_{num}| &\leq |Cl(C_\mathcal{T})| \cdot (b_\mathcal{K} + 1) \cdot (n'_{\max} + 1),\\
|\mathrm{Q}_{\mathcal{A}_num}| &\leq |\mathbf{I}_\mathcal{K}| \cdot |Cl(C_\mathcal{T})| \cdot b_\mathcal{K} \cdot (n'_{\max} + 1).
\end{aligned}$$

  Hence, it is easy to see that $|\mathrm{Q}_\mathcal{K}| = O(\|K\|^4)$ (cf. $|\mathrm{Q}_{\mathcal{A}\_num}|$).

- $\mathsf{ind}(\mathbf{A}) = \max(\mathsf{ind}(\mathbf{A}_\mathcal{I}), \mathsf{ind}(\mathbf{A}_\mathcal{A}), \mathsf{ind}(\mathbf{A}_\mathcal{T})) = \max(2, 2, 3) = 3$.

□

Thus, by Theorems 2.10 and 4.15, we get an optimal upper bound for KB satisfiability.

Corollary 4.17 *Deciding whether a given KB in $\mathcal{ZIQ}$ is satisfiable is in* ExpTime.

This is worst-case optimal, since a matching hardness result holds already for much weaker DLs, e.g. $\mathcal{ALC}$ [4].

5 Query answering via automata

We now turn to entailment of P2RPQs in KBs. As follows from Theorem 3.9, to decide whether $\mathcal{K} \models q$ for a P2RPQ q and a KB $\mathcal{K}$ in this DL, it is sufficient to decide whether $\mathcal{K}$ has a canonical model in which q has no match. We show how to do this using tree automata. Specifically, we build an automaton $\mathbf{A}_{\mathcal{K} \not\models q}$ that accepts all trees that represent a canonical model of $\mathcal{K}$ in which q has no match; hence, deciding query entailment reduces to checking $\mathcal{L}(\mathbf{A}_{\mathcal{K} \not\models q}) = \emptyset$.

Roughly, $\mathbf{A}_{\mathcal{K} \not\models q}$ is obtained by intersecting two automata: $\mathbf{A}_\mathcal{K}$ from Section 4.2 (which accepts the trees representing a canonical model of $\mathcal{K}$), and $\mathbf{A}_{\neg q}$, which accepts the trees representing an interpretation that admits no match for q. We construct $\mathbf{A}_{\neg q}$ in this section. As a preliminary step, we construct an automaton $\mathbf{A}_q$ that accepts a tree if q has a match in the interpretation it represents; we then show how to obtain from $\mathbf{A}_q$ the desired $\mathbf{A}_{\neg q}$. Figure 3 gives a general overview of the query answering technique; each of the steps will be discussed in detail below.

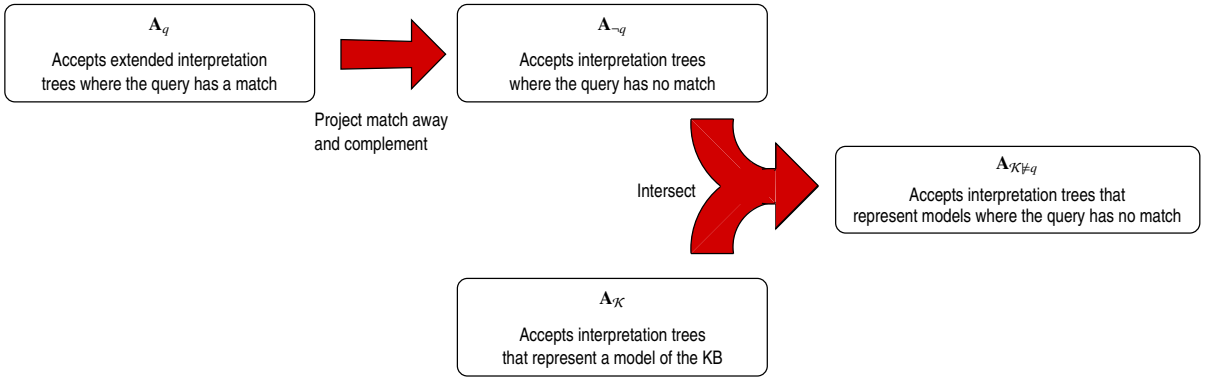

Figure 3: Overview of the automata algorithm for Query Entailment

5.1 Representing Query Matches

In what follows, let $q = \exists \vec{v}.\varphi(\vec{v})$. We assume w.l.o.g. that only role atoms of the form $R(v, v')$ occur q, since each atom $C(v)$ can be equivalently replaced with $id(C)(v, v)$. Let $At(q)$ be the set of all atoms occurring in q, and let $\mathbf{V}_q = \{v_1, \ldots, v_\ell\}$ be the variables in $\vec{v}$. We denote by $\mathbf{C}_q$, $\overline{\mathbf{R}}_q$, $\mathbf{I}_q$ the sets of atomic concepts, atomic roles, and individuals that occur in q, respectively.

Prior to defining $\mathbf{A}_q$, we introduce *extended interpretations*.

Definition 5.1 An extended interpretation is a pair $(\mathcal{I}, \pi)$ consisting of an interpretation $\mathcal{I}$ and a function $\pi : \mathbf{V}_q \cup \mathbf{I}_q \to \Delta^{\mathcal{I}}$ such that $\pi(a) = a^{\mathcal{I}}$ for each $a \in \mathbf{I}_q$. ∎

Intuitively, π is a possible match for the query. Extended interpretations are represented by *extended interpretation trees* labeled over the alphabet $\Sigma_{\mathcal{K},q}$, which enriches $\Sigma_{\mathcal{K}}$ with the variables in $\mathbf{V}_q$ and allows us to include the symbol v in the label of the node $\pi(v)$, for each $v \in \mathbf{V}_q$. We construct below an automaton $\mathbf{A}_q$ that accepts a $\Sigma_{\mathcal{K},q}$-labeled tree iff it represents an extended interpretation $(\mathcal{I}, \pi)$ such that $\mathcal{I}, \pi \models q$.

Definition 5.2 For $\Sigma = 2^{\Phi}$ and $\Sigma' = 2^{\Phi'}$ where $\Phi' \subseteq \Phi$, and for any Σ-labeled tree $\mathbf{T} = (T, L)$, the Σ'-*restriction of* $\mathbf{T}$ is the Σ'-labeled tree $\mathbf{T}' = (T, L')$, where L' is obtained by restricting L to Σ'.

We let $\Sigma_{\mathcal{K},q} = \{\sigma \cup \sigma' \mid \sigma \in \Sigma_{\mathcal{K}} \text{ and } \sigma' \in 2^{\mathbf{V}_q}\}$. An *extended interpretation tree* is a $\Sigma_{\mathcal{K},q}$-labeled $b_{\mathcal{K}}$-ary tree $\mathbf{T} = (T, L)$ such that

1. its $\Sigma_{\mathcal{K}}$-restriction is an interpretation tree, and
2. for each $v \in \mathbf{V}_q \cup \mathbf{I}_q$ there is exactly one $x \in T$ with $v \in L(x)$, called the *candidate match for* v; further, x is of the form $x = \varepsilon \cdot j \cdots$ with $L(\varepsilon \cdot j) \cap \mathbf{I}_{\mathcal{K}} \neq \emptyset$ (i.e., either x itself or its level-one ancestor is an individual node).

By $\pi_{\mathbf{T}}$ we denote the function $\mathbf{V}_q \cup \mathbf{I}_q \to T$ that maps each $v \in \mathbf{V}_q \cup \mathbf{I}_q$ to its candidate match. ∎

We associate extended interpretation trees with extended interpretations and vice versa, as follows.

Definition 5.3 The extended interpretation *represented by* an extended interpretation tree $\mathbf{T}$ is $\mathcal{J}_{\mathbf{T}} = (\mathcal{I}_{\mathbf{T}}, \pi_{\mathbf{T}})$. The *tree representation* $\mathbf{T}_{\mathcal{J}}$ of an extended interpretation $\mathcal{J} = (\mathcal{I}, \pi)$ is the extended interpretation tree (T, L) whose $\Sigma_{\mathcal{K}}$-restriction is $\mathbf{T}_{\mathcal{I}}$ and such that, for each $v \in \mathbf{V}_q$ and each $x \in T$, $v \in L(x)$ iff $x = \pi(v)$. ∎

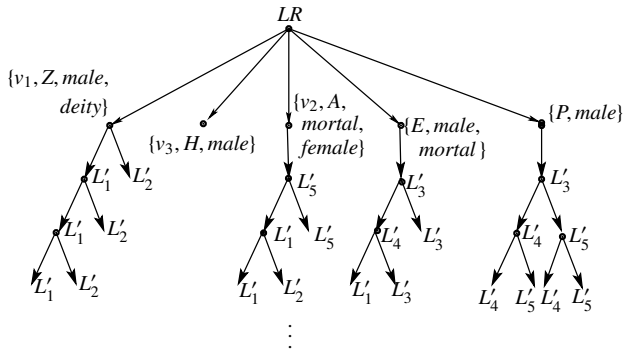

Figure 4: The tree representation of an extended interpretation

Analogously to Lemma 4.5, we have:

Lemma 5.4 *If* $\mathbf{T}$ *is an extended interpretation tree, then* $\mathcal{J}_{\mathbf{T}}$ *is an extended interpretation and* $\mathbf{T}_{\mathcal{J}_{\mathbf{T}}} = \mathbf{T}$.

Example 5.5 *Recall the interpretation* $\mathcal{I}_g$ *given in Example 4.3, and the match* $\pi(v_1) = \text{Zeus}^{\mathcal{I}_g}$, $\pi(v_2) = \text{Alcmene}^{\mathcal{I}_g}$ *and* $\pi(v_3) = \text{Heracles}^{\mathcal{I}_g}$ *for the query* q_g *in Example 2.5. The tree representation of the extended interpretation* $\mathcal{I}_g, \pi$ *is shown in Figure 4. It extends the tree representation of Figure 2 with the variable names* v_1, v_2 *and* v_3.

■

5.2 Constructing the Automaton

Now we construct the automaton $\mathbf{A}_q$ that accepts a $\Sigma_{\mathcal{K},q}$-labeled tree $\mathbf{T}$ iff (i) $\mathbf{T}$ is an extended interpretation tree and (ii) the map $\pi_{\mathbf{T}}$ represents a match for the query q in the associated interpretation $\mathcal{I}_{\mathbf{T}}$. We define $\mathbf{A}_q$ as the intersection of automata $\mathbf{A}_{\mathbf{T}}$ and $\mathbf{A}_\pi$ for (i) and (ii), respectively.

As for $\mathbf{A}_{\mathbf{T}}$, we can easily define a 2ATA $\mathbf{A}_V$ that verifies whether a given tree satisfies condition 2 of Definition 5.2 We obtain the desired $\mathbf{A}_{\mathbf{T}}$ by intersecting $\mathbf{A}_V$ with $\mathbf{A}_{\mathcal{I}}$ from Section 4.2, after adapting $\mathbf{A}_{\mathcal{I}}$ so that its alphabet is $\Sigma_{\mathcal{K},q}$ and it accepts all trees whose $\Sigma_{\mathcal{K}}$-restrictions are interpretation trees.

In detail, the 2ATA $\mathbf{A}_V = \langle \Sigma_V, \mathrm{Q}_V, \delta_V, q_0^V, F_V \rangle$ over Σ_V-labeled $b_{\mathcal{K}}$-ary trees is defined as follows:

- $\Sigma_V = \Sigma_{\mathcal{K},q}$;
- $\mathrm{Q}_V = \{q_0^V\} \cup \{v, \neg v, v^+, v^- \mid v \in \mathbf{V}_q\}$; v and $\neg v$ check the label of the current node, and respectively verify the presence or absence of v in it, while v^+ and v^- are used for checking whether the tree rooted at the current node contains the node labeled with v.
- $F_V = (\emptyset, \{v^- \mid v \in \mathbf{V}_q\}, \mathrm{Q}_V)$.
- The transition function $\delta_V : \mathrm{Q}_V \times \Sigma_{\mathcal{K},q} \to \mathcal{B}([b_{\mathcal{K}}] \times \mathrm{Q}_V)$ contains three groups of transitions:
  1. For each $\sigma \in \Sigma_{\mathcal{K},q}$ with $r \in \sigma$, a transition from the initial state:

$$\delta_V(q_0^V, \sigma) = \bigwedge_{v \in \mathbf{V}_q} \Big(\bigvee_{1 \le j \le b_{\mathcal{K}}} \big(\big(\bigvee_{a \in \mathbf{I}_{\mathcal{K}}} (j, a) \big) \wedge (j, v^+) \wedge \bigwedge_{1 \le j' \le b_{\mathcal{K}},\ j' \ne j} (j', v^-) \big) \Big).$$

2. For each $v \in \mathbf{V}_q$ and each $\sigma \in \Sigma_{\mathcal{K},q}$, transitions to the subtrees:

$$\begin{aligned}\delta_V(v^+, \sigma) &= ((0, v) \wedge \textstyle\bigwedge_{1 \le j \le b_\mathcal{K}}(j, v^-)) \vee ((0, \neg v) \wedge \bigvee_{1 \le j \le b_\mathcal{K}}((j, v^+) \wedge \bigwedge_{1 \le j' \le b_\mathcal{K},\ j' \neq j}(j', v^-))),\\ \delta_V(v^-, \sigma) &= (0, \neg v) \wedge \textstyle\bigwedge_{1 \le j \le b_\mathcal{K}}(j, v^-).\end{aligned}$$

3. For each $\sigma \in \Sigma_\mathcal{K}$ and each $v \in \mathbf{V}_q$, transitions that check the labeling with v:

$$\delta_V(v, \sigma) = \begin{cases}\textbf{true}, & \text{if } v \in \sigma\\ \textbf{false}, & \text{if } v \notin \sigma\end{cases}, \quad \delta_V(\neg v, \sigma) = \begin{cases}\textbf{true}, & \text{if } v \notin \sigma\\ \textbf{false}, & \text{if } v \in \sigma\end{cases}.$$

The automaton $\mathbf{A}_\mathcal{I}$ in Definition 4.6 is modified to $\mathbf{A}'_\mathcal{I} = \langle \Sigma_{\mathcal{K},q}, \mathrm{Q}_\mathcal{I}, \delta'_\mathcal{I}, q_0^\mathcal{I}, F_\mathcal{I}\rangle$, by changing $\Sigma_\mathcal{I}$ to $\Sigma_{\mathcal{K},q}$ and setting, for each $\sigma \in \Sigma_{\mathcal{K},q}$ and $q \in \mathrm{Q}_\mathcal{I}$, $\delta'_\mathcal{I}(\sigma, q) = \delta_\mathcal{I}(\sigma', q)$ whenever $\sigma \setminus \mathbf{V}_q = \sigma'$. Then, $\mathbf{A_T}$ is an automaton that accepts the intersection of $\mathbf{A}_V$ and $\mathbf{A}'_\mathcal{I}$, as in Lemma 2.11.

Lemma 5.6 $\mathbf{A_T}$ *accepts a* $\Sigma_{\mathcal{K},q}$*-labeled* $b_\mathcal{K}$*-ary tree* $\mathbf{T}$ *iff* $\mathbf{T}$ *is an extended interpretation tree for* $\mathcal{K}$*.*

Now we define the 2ATA $\mathbf{A}_\pi$. We use the individuals in $\mathbf{I}_q$ and the variables in $\mathbf{V}_q$ as atomic concepts, and use a *q-concept* C_α for each query atom $\alpha = R(v, v')$, such that C_α is satisfied at some root of an extended interpretation $(\mathcal{I}, \pi)$ iff $\mathcal{I}, \pi \models \alpha$.

Definition 5.7 A *q-concept* C is defined as a regular $\mathcal{ZIQ}$ concept, but allows also the elements of $\mathbf{V}_q \cup \mathbf{I}_q$ in place of atomic concepts. For each $\alpha = R(v, v')$ in q, we let $C_\alpha = \exists U.(v \sqcap \exists R.v')$, where $U = (\bigcup_{P \in \overline{\mathbf{R}}_\mathcal{K}} P)^*$.

The semantics of a q-concept C in a extended interpretation $\mathcal{J} = (\mathcal{I}, \pi)$ is as follows:

- if $C = A$ for some $A \in \mathbf{C}$, then $C^\mathcal{J} = A^\mathcal{I}$;
- if $C = v$ for some $v \in \mathbf{V}_q \cup \mathbf{I}_q$, then $C^\mathcal{J} = \{\pi(v)\}$.

This inductively extends to complex q-concepts as in Definition 2.3 (i.e., like for a regular interpretation). ∎

The q-concepts correctly capture the semantics of the query atoms:

Lemma 5.8 *For every extended interpretation* $\mathcal{J} = (\mathcal{I}, \pi)$ *and atom* $\alpha = R(v, v')$ *occurring in* q*, we have* $\mathcal{I}, \pi \models \alpha$ *iff there is some root* $i \in \Delta^\mathcal{I}$ *such that* $i \in C_\alpha^\mathcal{J}$*.*

By this Lemma, we only need to verify that $C_{\alpha_1}, \ldots, C_{\alpha_k}$ hold at some root for query atoms $\alpha_1, \ldots, \alpha_k$ that make the query q true. The satisfaction of the concepts C_{α_i} is verified by an automaton $\mathbf{A}_\pi$ that decomposes them via transitions analogous to those of $\mathbf{A}_\mathcal{T}$. Modulo the initial transition from the root node, $\mathbf{A}_\pi$ and $\mathbf{A}_\mathcal{T}$ are very similar.

Definition 5.9 Let $Cl_q = \bigcup_{\alpha \in At(q)} Cl(C_\alpha)$.We define the 2ATA $\mathbf{A}_\pi = (\Sigma_\pi, \mathrm{Q}_\pi, \delta_\pi, q_0^\pi, F_\pi)$ as follows.

- $\Sigma_\pi = \Sigma_{\mathcal{K},q}$;
- Q_π is like $\mathrm{Q}_\mathcal{K}$ in $\mathbf{A}_\mathcal{K}$, but defined using Cl_q instead of Cl (see Table 5).
- $F_\pi = (\emptyset, \{\forall R^*.C \mid \forall R^*.C \in Cl_q\}, \mathrm{Q}_\pi)$, analogously as in $\mathbf{A}_\mathcal{T}$.

$$
\begin{aligned}
Cl^q_{ext} &= Cl_q \cup \{s, \neg s \mid s \in \mathbf{I}_\mathcal{K} \cup \mathbf{V}_q\} \\
\mathrm{Q}_{q,\mathsf{Self}} &= \{S_{\mathsf{Self}} \mid S \text{ a simple role in } Cl_q\} \\
\mathrm{Q}_{q,\mathcal{A}\_role} &= \{S_{ij} \mid i, j \in \{1, \ldots, b_\mathcal{K}\} \text{ and } S \text{ a simple role in } Cl_q\} \\
\mathrm{Q}_{q,num} &= \{\langle \geqslant n\, S.C, i, j\rangle \mid \geqslant n\, S.C \in Cl_q,\ 0 \leq i \leq b_\mathcal{K}{+}1,\ 0 \leq j \leq n\} \cup \\
&\quad \{\langle \leqslant n\, S.C, i, j\rangle \mid \leqslant n\, S.C \in Cl_q,\ 0 \leq i \leq b_\mathcal{K}{+}1,\ 0 \leq j \leq n{+}1\} \\
\mathrm{Q}_{q,\mathcal{A}\_num} &= \{\langle a, \geqslant n\, S.C, i, j\rangle \mid a \in \mathbf{I}_\mathcal{K},\ \geqslant n\, S.C \in Cl_q,\ 1 \leq i \leq b_\mathcal{K},\ 0 \leq j \leq n\} \cup \\
&\quad \{\langle a, \leqslant n\, S.C, i, j\rangle \mid a \in \mathbf{I}_\mathcal{K},\ \leqslant n\, S.C \in Cl_q,\ 1 \leq i \leq b_\mathcal{K}; 0 \leq j \leq n{+}1\}
\end{aligned}
$$

Table 5: State set $\mathrm{Q}_\pi = \{q^\pi_0\} \cup Cl^q_{ext} \cup \mathrm{Q}_{q,\mathsf{Self}} \cup \mathrm{Q}_{q,\mathcal{A}\_role} \cup \mathrm{Q}_{q,num} \cup \mathrm{Q}_{q,\mathcal{A}\_num}$

- The transitions from the initial state are defined for each label σ containing r (identifying the root node) as

  $$\delta_\pi(q_0, \sigma) = B_\varphi,$$

  where B_φ results from $\varphi(\vec{v})$ by replacing each atom α with $(0, C_\alpha)$.

  When $\mathbf{A}_\pi$ is at the root node and in a state C_α for some $\alpha \in At(q)$, it checks that the concept C_α is satisfied at some individual node, via the following transitions, for each σ containing r:

  $$\delta_\pi(C_\alpha, \sigma) = \bigvee\nolimits_{1 \leq i \leq b_\mathcal{K}}((i, C_\alpha) \wedge \bigvee\nolimits_{a \in \mathbf{I}_\mathcal{K}}(i, a)).$$

  To further check that C_α is satisfied, $\mathbf{A}_\pi$ has transitions similar to those of $\mathbf{A}_\mathcal{T}$, viz. for each $\sigma \in \Sigma_\mathcal{K}$,

  1. transitions that recursively decompose complex concepts and non-simple roles in Cl_q, as in item 2 of $\delta_\mathcal{T}$;
  2. transitions that handle all simple roles in Cl_q and the states in $\mathrm{Q}_{q,\mathsf{Self}}$ and $\mathrm{Q}_{q,\mathcal{A}\_role}$, as in item 3 of $\delta_\mathcal{T}$;
  3. transitions $\delta_\pi(s, \sigma)$ for each $s \in Cl_q$ of the form $\geqslant n\, S.C$, and for each $s \in \mathrm{Q}_{q,num} \cup \mathrm{Q}_{q,\mathcal{A}\_num}$ to verify the satisfaction of the number restrictions, as in item 4 of $\delta_\mathcal{T}$;
  4. transitions $\delta_\pi(s, \sigma)$ for each $s \in \mathbf{C}_\mathcal{K} \cup \overline{\mathbf{R}}_\mathcal{K} \cup \mathbf{I}_\mathcal{K} \cup PI_\mathcal{K} \cup PS_\mathcal{K}$ as in item 5 of $\delta_\mathcal{T}$, and for each $s \in \mathbf{V}_q$, as in item 3 of $\delta_\mathcal{I}$, to check symbol occurrences in node labels. ∎

Given an extended interpretation tree, $\mathbf{A}_\pi$ correctly checks the satisfaction of complex concepts in Cl_q.

Lemma 5.10 *(Cf. Lemma 4.11) For every extended interpretation tree* $\mathbf{T}$*, the following holds.*

1. *If* (T_r, r) *is an accepting run of* $\mathbf{A}_\pi$ *over* $\mathbf{T}$*, then* $r(y) = (x, C)$ *for* $y \in T_r$ *and* $C \in Cl_q$ *implies* $x \in C^{\mathcal{I}_\mathbf{T}}$.
2. *If for some* $x \in \mathbf{T}$ *and some* $C \in Cl(C_\mathcal{T})$ *there does not exist an accepting run* (T_r, r) *of* $\mathbf{A}_\pi$ *over* $\mathbf{T}$ *such that* $r(y) = (x, C)$ *for some* $y \in T_r$*, then* $x \notin C^{\mathcal{I}_\mathbf{T}}$.

Hence a run of $\mathbf{A}_\pi$ on an extended interpretation tree that visits a state C_α correctly verifies the existence of a match for the atom α. By this and Lemma 5.8, the initial transition from the root is sufficient to verify whether this holds for a set of atoms that makes q true:

Lemma 5.11 *An extended interpretation tree* $\mathbf{T} = (T, L)$ *is accepted by* $\mathbf{A}_q$ *iff there exists some* $A \subseteq At(q)$ *such that*

- *for every* $\alpha \in A$*, there is some root* $i \in \Delta^{\mathcal{I}}$ *such that* $i \in C_\alpha^{\mathcal{J}}$*, and*
- *by assigning true to the atoms in A and false to all others,* φ *evaluates to* **true***.*

Finally, $\mathbf{A}_q$ is the intersection of $\mathbf{A}_\mathbf{T}$ and $\mathbf{A}_\pi$, which accepts the trees representing interpretations where q has a match:

Proposition 5.12 *For given* $\mathcal{K}$ *and* q*,* $\mathcal{L}(\mathbf{A}_q) = \{\mathbf{T}_\mathcal{J} \mid \mathcal{J} = (\mathcal{I}, \pi)$ *is a an extended interpretation for* $\mathcal{K}$ *and* $\mathcal{I}, \pi \models q\}$*.*

5.3 Deciding Query Entailment

Our algorithm for deciding $\mathcal{K} \models q$ roughly works as follows. The automaton $\mathbf{A}_q$ accepts a tree over the extended alphabet $\Sigma_{\mathcal{K},q}$, if it represents an extended interpretation for $\mathcal{K}$ in which π is a match for q. We project the query variables $\mathbf{V}_q$ from $\mathbf{A}_q$'s alphabet and obtain an automaton that accepts the same trees, but restricted to $\Sigma_\mathcal{K}$; they correspond to the interpretation trees for $\mathcal{K}$ in which q has a match, no matter where it is. The next step is to complement this automaton, such that it accepts an interpretation tree exactly when there is no match for q in it. Finally, we intersect this automaton with the automaton $\mathbf{A}_\mathcal{K}$ to obtain an automaton $\mathbf{A}_{\mathcal{K}\not\models q}$ that accepts the trees that represent a canonical model of $\mathcal{K}$ in which q has no match. By Theorem 3.9 $\mathcal{K} \not\models q$ iff such a model exists. Hence deciding $\mathcal{K} \models q$ reduces to testing the automaton $\mathbf{A}_{\mathcal{K}\not\models q}$ for emptiness.

We recall some results on 1NTA. The following bounds for automata complementation are given in [46].

Proposition 5.13 *For every 1NTA* $\mathbf{A}$ *running over k-ary trees,* $k \geq 1$*, it is possible to construct a 1NTA* $\overline{\mathbf{A}}$ *that accepts a k-ary* Σ*-labeled tree* (T, L) *iff* $(T, L) \notin \mathcal{L}(\mathbf{A})$*, and such that* $|Q(\overline{\mathbf{A}})| \leq 2^{O(f(\mathbf{A}))}$ *and* $\mathsf{ind}(\overline{\mathbf{A}}) = O(f(\mathbf{A}))$*, where* $f(\mathbf{A}) = \mathsf{ind}(\mathbf{A}) \cdot |Q(\mathbf{A})| \cdot \log |Q(\mathbf{A})|$*.*

For $\Sigma = 2^\Phi$ and $\Sigma' = 2^{\Phi'}$ where $\Phi' \subseteq \Phi$, let the Σ'*-projection* of a set $\mathcal{L}$ of Σ-labeled trees be the set containing the Σ'-restrictions of all trees in $\mathcal{L}$.

Lemma 5.14 *For* Σ *and* Σ' *as above, for every 1NTA* $\mathbf{A}$ *running over k-ary* Σ*-labeled trees,* $k \geq 1$*, it is possible to construct a 1NTA* $\mathbf{A}^{\Sigma'}$ *with* $|Q(\mathbf{A}^{\Sigma'})| \leq |Q(\mathbf{A})|$ *and* $\mathsf{ind}(\mathbf{A}^{\Sigma'}) \leq \mathsf{ind}(\mathbf{A})$ *that accepts the* Σ'*-projection of* $\mathcal{L}(\mathbf{A})$*.*

Proof. To obtain $\mathbf{A}^{\Sigma'}$ from $\mathbf{A}$, simply change Σ to Σ' and the transition function to δ' as follows. For each $\sigma \in \Sigma'$ and each $q \in Q(\mathbf{A})$, $\delta'(q, \sigma) = \bigvee_{\sigma' \in \Xi(\sigma)} \delta'(q, \sigma')$, where $\Xi(\sigma) = \{\sigma' \in \Sigma \mid \sigma' \cap \Phi' = \sigma\}$. □

Using this Lemma and Theorem 2.10, we can construct a 1NTA that accepts exactly the set of trees that either do not represent an interpretation, or represent an interpretation where q has no match. More precisely, we have:

Lemma 5.15 *Given q, it is possible to construct a 1NTA* $\mathbf{A}_{\neg q}$ *such that* $\mathcal{I}_\mathbf{T} \not\models q$ *for every interpretation tree* $\mathbf{T} \in \mathcal{L}(\mathbf{A}_{\neg q})$*, and with* $|Q(\mathbf{A}_{\neg q})| \leq 2^{2^{O(n_q^c)}}$ *and* $\mathsf{ind}(\mathbf{A}_{\neg q}) \leq 2^{O(n_q^c)}$ *for* $n_q = |Q(\mathbf{A}_q)|$ *and some constant c, i.e., with double exponentially many states and with index single exponential in the number of states of* $Q(\mathbf{A}_q)$*.*

Proof. Let $\mathbf{A}_0 = \mathbf{A}_q$ and $n_q = |Q(\mathbf{A}_q)|$. By Theorem 2.10, we can transform $\mathbf{A}_0$ into an 1NTA $\mathbf{A}_1$ with $\mathcal{L}(\mathbf{A}_1) = \mathcal{L}(\mathbf{A}_0)$ such that $|Q(\mathbf{A}_1)| \leq 2^{O(n_q^{c_0})}$ for some constant c_0 and $\mathsf{ind}(\mathbf{A}_1) = O(\mathsf{ind}(\mathbf{A}_0)) = O(1)$. By Lemma 5.14, we can transform $\mathbf{A}_1$ into a 1NTA $\mathbf{A}_2$ which accepts the $\Sigma_\mathcal{K}$-projection of $\mathcal{L}(\mathbf{A}_1)$ such that $|Q(\mathbf{A}_2)| \leq |Q(\mathbf{A}_1)| \leq 2^{O(|Q(\mathbf{A}_0)|^{c_0})}$ and $\mathsf{ind}(\mathbf{A}_2) \leq \mathsf{ind}(\mathbf{A}_1) = O(1)$. By Lemma 5.13, we can construct from $\mathbf{A}_2$ an automaton $\mathbf{A}_3 = \mathbf{A}_{\neg q}$ accepting the complement of $\mathcal{L}(\mathbf{A}_2)$ such that $|Q(\mathbf{A}_3)| \leq 2^{O(f(\mathbf{A}_2))}$ and $\mathsf{ind}(\mathbf{A}_3) = O(f(\mathbf{A}_2))$, where $f(\mathbf{A}_2) = \mathsf{ind}(\mathbf{A}_2) \cdot |Q(\mathbf{A}_2)| \cdot \log |Q(\mathbf{A}_2)| \leq 2^{O(|Q(\mathbf{A}_0)|^{c_2})} \cdot O(|Q(\mathbf{A}_0)|^{c_2})$, for some constant c_2. It follows that $|Q(\mathbf{A}_{\neg q})| \leq 2^{2^{O(n_q^c)}}$ and $\mathsf{ind}(\mathbf{A}_{\neg q}) \leq 2^{O(n_q^c)}$ for some constant c.

We know that $\mathcal{L}(\mathbf{A}_q) = \mathcal{L}(\mathbf{A}_1) = \{\mathbf{T}_\mathcal{J} \mid \mathcal{J} = (\mathcal{I}, \pi)$ is a an extended interpretation for $\mathcal{K}$ and $\mathcal{I}, \pi \models q\}$. By construction, $\mathbf{A}_2$ accepts the $\Sigma_\mathcal{K}$-projection of $\mathcal{L}(\mathbf{A}_1)$, namely $\mathcal{L}(\mathbf{A}_2) = \{\mathbf{T}_\mathcal{I} \mid$ for some $\pi, \mathcal{J} = (\mathcal{I}, \pi)$ is a an extended interpretation for $\mathcal{K}$ and $\mathcal{I}, \pi \models q\}$. Its complement is $\mathcal{L}(\mathbf{A}_3) = \{\mathbf{T} \mid$ there exist no $\mathcal{I}$ and no π such that $\mathbf{T} = \mathbf{T}_\mathcal{I}$, $\mathcal{J} = (\mathcal{I}, \pi)$ is a an extended interpretation for $\mathcal{K}$ and $\mathcal{I}, \pi \models q\}$, i.e., $\mathcal{L}(\mathbf{A}_3)$ accepts an interpretation tree $\mathbf{T}$ only if $\mathcal{I}_\mathbf{T} \not\models q$. Then $\mathbf{A}_3 = \mathbf{A}_{\neg q}$ is the desired automaton. □

To obtain the automaton $\mathbf{A}_{\mathcal{K} \not\models q}$, we intersect $\mathbf{A}_\mathcal{K}$, after transforming it into an 1NTA, with $\mathbf{A}_{\neg q}$. The following bounds for the intersection of two 1NTAs are known:[5]

Lemma 5.16 *Given 1NTAs $\mathbf{A}_1$ and $\mathbf{A}_2$, it is possible to construct a 1NTA $\mathbf{A}$ such that $\mathcal{L}(\mathbf{A}) = \mathcal{L}(\mathbf{A}_1) \cap \mathcal{L}(\mathbf{A}_2)$ with $\mathsf{ind}(\mathbf{A}) = O(f(\mathbf{A}_1, \mathbf{A}_2))$ and $|Q(\mathbf{A})| \leq 2^{O(f(\mathbf{A}_1, \mathbf{A}_2)^2)} \cdot f(\mathbf{A}_1, \mathbf{A}_2) \cdot |Q(\mathbf{A}_1)| \cdot |Q(\mathbf{A}_2)|$, where $f(\mathbf{A}_1, \mathbf{A}_2) = \mathsf{ind}(\mathbf{A}_1) + \mathsf{ind}(\mathbf{A}_2) + 1$.*

Finally, we can prove the existence of the automaton $\mathbf{A}_{\mathcal{K} \not\models q}$ that accepts exactly the canonical models of $\mathcal{K}$ where there is no match for q as desired:

Lemma 5.17 *Given $\mathcal{K}$ and q, it is possible to construct a 1NTA $\mathbf{A}_{\mathcal{K} \not\models q}$ with $|Q(\mathbf{A}_{\mathcal{K} \not\models q})| \leq 2^{2^{O((n_\mathcal{K} + n_q)^c)}}$ and $\mathsf{ind}(\mathbf{A}_{\mathcal{K} \not\models q}) \leq 2^{O(n_q^c)}$ for some constant c where $n_\mathcal{K} = |Q(\mathbf{A}_\mathcal{K})|$ and $n_q = |Q(\mathbf{A}_q)|$, i.e., with double exponentially many states in the number of states of $\mathbf{A}_\mathcal{K}$ and $\mathbf{A}_q$, and with index single exponential in the number of states of $\mathbf{A}_q$, such that $\mathcal{L}(\mathbf{A}_{\mathcal{K} \not\models q}) = \{\, \mathbf{T} \mid \mathcal{I}_\mathbf{T} \models \mathcal{K}$ and $\mathcal{I}_\mathbf{T} \not\models q \,\}$.*

Proof. By Theorem 2.10, $\mathbf{A}_\mathcal{K}$ is convertible into a 1NTA $\mathbf{A}_1$ with $|Q(\mathbf{A}_1)| \leq 2^{O(|Q(\mathbf{A}_\mathcal{K})|^{c_1})}$ for some constant c_1 and $\mathsf{ind}(\mathbf{A}_1) = O(\mathsf{ind}(\mathbf{A}_\mathcal{K})) = O(1)$ such that $\mathcal{L}(\mathbf{A}_\mathcal{K}) = \mathcal{L}(\mathbf{A}_1)$. We then construct a 1NTA $\mathbf{A}_3 = \mathbf{A}_{\mathcal{K} \not\models q}$ as the intersection of $\mathbf{A}_1$ and $\mathbf{A}_2 = \mathbf{A}_{\neg q}$, which by Lemma 5.16 has $\mathsf{ind}(\mathbf{A}_3) = O(f(\mathbf{A}_1, \mathbf{A}_2))$ and $|Q(\mathbf{A}_3)| \leq 2^{O(f(\mathbf{A}_1, \mathbf{A}_2)^2)} \cdot f(\mathbf{A}_1, \mathbf{A}_2) \cdot |Q(\mathbf{A}_1)| \cdot |Q(\mathbf{A}_2)|$, where $f(\mathbf{A}_1, \mathbf{A}_2) = \mathsf{ind}(\mathbf{A}_1) + \mathsf{ind}(\mathbf{A}_2) + 1$. As by Lemma 5.15, $|Q(\mathbf{A}_2)| \leq 2^{2^{O(n_q^{c_2})}}$ and $\mathsf{ind}(\mathbf{A}_2) \leq 2^{O(n_q^{c_2})}$ for some constant c_2, it follows that $\mathsf{ind}(\mathbf{A}_3) \leq 2^{O(n_q^{c_2})} + O(1)$, and $|Q(\mathbf{A}_3)| \leq 2^{O(f(\mathbf{A}_1, \mathbf{A}_2)^2) \cdot f(\mathbf{A}_1, \mathbf{A}_2) \cdot 2^{O(|Q(\mathbf{A}_\mathcal{K})|^{c_1})} \cdot 2^{2^{O(n_q^{c_2})}}}$, where $f(\mathbf{A}_1, \mathbf{A}_2) = \mathsf{ind}(\mathbf{A}_1) + \mathsf{ind}(\mathbf{A}_2) + 1 \leq 2^{O(n_q^{c_2})} + O(1) + 1$. It follows from this that $|Q(\mathbf{A}_{\mathcal{K} \not\models q})| \leq 2^{2^{O((n_\mathcal{K} + n_q)^c)}}$ and $\mathsf{ind}(\mathbf{A}_{\mathcal{K} \not\models q}) \leq 2^{O(n_q^c)}$ for some constant c. Suppose $\mathbf{T} \in \mathcal{L}(\mathbf{A}_{\mathcal{K} \not\models q})$. Then $\mathbf{T} \in \mathcal{L}(\mathbf{A}_\mathcal{K})$, which means that $\mathcal{I}_\mathbf{T}$ is a canonical model of $\mathcal{K}$, and $\mathbf{T} \in \mathcal{L}(\mathbf{A}_{\neg q})$, which means that q has no match in $\mathcal{I}_\mathbf{T}$. Conversely, if $\mathbf{T} \notin \mathcal{L}(\mathbf{A}_{\mathcal{K} \not\models q})$, then either $\mathbf{T} \notin \mathcal{L}(\mathbf{A}_\mathcal{K})$, which means $\mathcal{I}_\mathbf{T}$ is not a model of $\mathcal{K}$, or $\mathbf{T} \notin \mathcal{L}(\mathbf{A}_{\neg q})$, which means that q has a match in $\mathcal{I}_\mathbf{T}$. □

Therefore, $\mathbf{A}_{\mathcal{K} \not\models q}$ accepts some input tree if and only if $\mathcal{K}$ has some canonical model in which q has no match. Combined with Theorem 3.9, we thus obtain:

Theorem 5.18 *For every P2RPQ q over a knowledge base $\mathcal{K}$ in $\mathcal{ALCQI}b_{reg}$, it holds that $\mathcal{K} \models q$ iff $\mathcal{L}(\mathbf{A}_{\mathcal{K} \not\models q}) = \emptyset$.*

[5] To our knowledge, these bounds have not been published. A tighter bound of $\mathsf{ind}(\mathbf{A}) = \mathsf{ind}(\mathbf{A}_1) + \mathsf{ind}(\mathbf{A}_2)$ and $|Q(\mathbf{A})| = f'(\mathbf{A}_1, \mathbf{A}_2)! \cdot f'(\mathbf{A}_1, \mathbf{A}_2) \cdot |Q(\mathbf{A}_1)| \cdot |Q(\mathbf{A}_2)|$, where $f'(\mathbf{A}_1, \mathbf{A}_2) = (\mathsf{ind}(\mathbf{A}_1) + \mathsf{ind}(\mathbf{A}_2))/2 + 1$, was confirmed through personal communication with Yoad Lustig and Nir Piterman, to whom we are very grateful.

5.4 Complexity

We now show that the reduction of query entailment to automata emptiness as in Theorem 5.18 gives a tight upper complexity bound for the problem. Recall that $\|\mathcal{K}\|$ and $\|q\|$ respectively denote the size of some representation (as strings) of $\mathcal{K}$ and q, and let $\|\mathcal{K}, q\| = \|\mathcal{K}\| + \|q\|$ denote their combined size.

It is not difficult to show that $\mathbf{A}_q$ has polynomially many states in $\|\mathcal{K}, q\|$.

Lemma 5.19 *$|Q(\mathbf{A}_q)| = O(\|\mathcal{K}, q\|^c)$, for some constant c, and $\mathsf{ind}(\mathbf{A}_q) = 3$.*

Proof. Recall that $\mathsf{ind}(\mathbf{A}_q) = \max(\mathsf{ind}(\mathbf{A}_\pi), \mathsf{ind}(\mathbf{A}_V), \mathsf{ind}(\mathbf{A}_\mathcal{I})) = 3$. Let $n'_{\max} = \max(\{n \mid \geqslant n\, S.C \in Cl(C_\mathcal{T})\} \cup \{0\})$ (resp., $n^q_{\max} = \max(\{n \mid \geqslant n\, S.C \in Cl_q)\} \cup \{0\})$.

Recall that $|\mathbf{C}_\mathcal{K}|$, $|\overline{\mathbf{R}}_\mathcal{K}|$, $|\mathbf{I}_\mathcal{K}|$, $|Cl(C_\mathcal{T})|$, $n_{\max}$ and $b_\mathcal{K}$ are linear in $\|\mathcal{K}\|$ under the assumptions on the encoding of $\mathcal{K}$ (cf. Section 4.4). Under similar assumptions for q, we have that $|Cl_q|$, $|\mathbf{V}_q|$ and $n^q_{\max}$ are linear in $\|q\|$. Then $|Q(\mathbf{A}_q)| \leq |\mathrm{Q}_\pi| + |\mathrm{Q}_V| + |\mathrm{Q}_\mathcal{I}| + 2$, where

$$
\begin{aligned}
|\mathrm{Q}_\mathcal{I}| &\leq 2 + 2(|\mathbf{I}_\mathcal{K}| + 1),\\
|\mathrm{Q}_V| &\leq 1 + 4|\mathbf{V}_q|,\\
|\mathrm{Q}_\pi| &\leq 1 + |Cl^q_{ext}| + |\mathrm{Q}_{q,\mathsf{Self}}| + |\mathrm{Q}_{q,\mathcal{A}\_role}| + |\mathrm{Q}_{q,num}| + |\mathrm{Q}_{q,\mathcal{A}\_num}|,\\
|Cl^q_{ext}| &\leq |Cl_q| + 2(|\mathbf{I}_\mathcal{K}| + |\mathbf{V}_q|),\\
|\mathrm{Q}_{q,\mathsf{Self}}| &\leq |Cl_q|,\\
|\mathrm{Q}_{q,\mathcal{A}\_role}| &\leq |Cl_q| \cdot |b_\mathcal{K}|^2\\
|\mathrm{Q}_{q,num}| &\leq |Cl_q| \cdot (b_\mathcal{K} + 1) \cdot (n^q_{\max} + 1),\\
|\mathrm{Q}_{q,\mathcal{A}\_num}| &\leq |\mathbf{I}_\mathcal{K}| \cdot |Cl_q| \cdot b_\mathcal{K} \cdot n^q_{\max}.
\end{aligned}
$$

Hence, $|Q(\mathbf{A}_q)| = O(\|\mathcal{K}, q\|^4)$ (in fact, $|Q(\mathbf{A}_q)| = O(\|\mathcal{K}, q\|^3)$ for fixed $\mathcal{K}$ and $|Q(\mathbf{A}_q)| = O(\|\mathcal{K}, q\|^2)$ for fixed q). □

Lemma 5.20 *$|Q(\mathbf{A}_{\mathcal{K}\not\models q})| = 2^{2^{O(\|\mathcal{K},q\|^c)}}$ and $\mathsf{ind}(\mathbf{A}_{\mathcal{K}\not\models q}) = 2^{O(\|\mathcal{K},q\|^c)}$ for some constant c. Furthermore, $\mathbf{A}_{\mathcal{K}\not\models q}$ can be constructed in time double exponential in $\|\mathcal{K}, q\|$.*

Proof. The first part follows directly from Lemmas 4.16, 5.17, and 5.19. The second part is also straightforward, since in all the automata constructions given in Section 5.3 the time required to construct an automaton is polynomial in its size plus the input. □

Testing a 1NTA for emptiness is feasible within the following bounds.

Proposition 5.21 ([42]) *Given a 1NTA $\mathbf{A}$, the nonemptiness problem is decidable in time $O(|Q(\mathbf{A})|^{\mathsf{ind}(\mathbf{A})})$.*

We thus obtain the main result of this section.

Theorem 5.22 *Given a P2RPQ q over a KB $\mathcal{K}$ in $\mathcal{ALCQI}b_{reg}$, deciding whether $\mathcal{K} \models q$ is in* 2ExpTime.

This bound is worst case optimal. As shown in [44], answering conjunctive queries over KBs in $\mathcal{ALCI}$ (i.e., P2RPQs built only with $\wedge$ and where no regular role expressions, but only concepts and roles as in $\mathcal{ALCI}$, are allowed in the query) is 2ExpTime-hard. More recently, the same lower bound was established for $\mathcal{SH}$ [23].

Note that the P2RPQs we consider are more expressive than CQs in $\mathcal{ALCI}$ and $\mathcal{SH}$; however, we obtain the following result.

Theorem 5.23 *Let $\mathcal{L}_\mathcal{K} \subseteq \mathcal{ZIQ}$ be a DL, $\mathcal{L}_q \subseteq$ P2RPQs a query language, q in $\mathcal{L}_q$, and $\mathcal{K}$ in $\mathcal{L}_\mathcal{K}$. If either (a) $\mathcal{ALCI} \subseteq \mathcal{L}_\mathcal{K}$ and $\mathcal{L}_q$ contains conjunctive queries in $\mathcal{ALCI}$, or (b) $\mathcal{SH} \subseteq \mathcal{L}_\mathcal{K}$ and $\mathcal{L}_q$ contains conjunctive queries in $\mathcal{SH}$, then deciding $\mathcal{K} \models q$ is* 2ExpTime*-complete.*

5.4.1 Data complexity

A brief remark on the impact of $\mathcal{K}$ to the overall complexity of the algorithm is in order. Most of the existing query answering algorithms in expressive DLs are double exponential in $\|\mathcal{K}\| + \|q\|$, but just single exponential in $\|\mathcal{K}\|$. However, this is not the case for our algorithm. The definition of P2RPQs allows for arbitrary concepts in query atoms. It is common practice to restrict the queries and allow only atomic concepts instead. If we adopt this restriction (but allow for arbitrary roles), then $\mathbf{A}_q$ does not have to deal with number restrictions (as they do not occur in the concepts C_α representing the query atoms). Consequently, its set of states would be simpler ($Q(\mathbf{A}_q) = \{q_0\} \cup Cl^q_{ext} \cup \mathrm{Q}_{q,\mathcal{A}\_role} \cup \mathrm{Q}_{q,\mathcal{A}\_quant}$) and the construction can easily be modified so that it would depend only on $\|q\|$ and $|\mathbf{I}_\mathcal{K}|$. This means that if $\mathbf{I}_\mathcal{K}$ is bounded by a constant, $|Q(\mathbf{A}_{\mathcal{K}\not\models q})|$ and $\mathsf{ind}(\mathbf{A}_{\mathcal{K}\not\models q})$ are single exponential and constant in $\|\mathcal{K}\|$ respectively, which leads to a decision procedure that is single exponential in $\|\mathcal{K}\|$. In the general case, the cardinality of $Q(\mathbf{A}_q)$ and the size of $F(\mathbf{A}_q)$ are already double and single exponential in $|\mathbf{I}_\mathcal{K}|$, respectively. This implies that our decision procedure is double exponential in the size of the ABox in the worst case, i.e., it has doubly exponential data complexity. Although the data complexity of query answering in $\mathcal{ZIQ}$ has not been studied so far, this bound is not likely to be optimal. In fact, for the related description logic $\mathcal{SHIQ}$, a much lower coNP lower bound for data complexity is known [27]. It is unclear whether $\mathbf{A}_q$ could be designed in such a way that its size does not depend on $\|\mathcal{K}\|$, which would yield an exponentially better upper bound on data complexity.

6 Complex role inclusion axioms

The DL $\mathcal{SRIQ}$ was introduced in [31] as an extension of $\mathcal{RIQ}$ [33], which in turn extends the well known DL $\mathcal{SHIQ}$ [34] underlying OWL-Lite. $\mathcal{SRIQ}$ has gained considerable attention in the last years as the 'Lite' fragment of the DL $\mathcal{SROIQ}$ underlying the new OWL 2 standard. In this section, we show that our algorithm can also be utilized for query answering in $\mathcal{SRIQ}$ by means of a suitable reduction to the logic $\mathcal{ZIQ}$.

The most prominent feature of $\mathcal{SRIQ}$ are *role inclusion axioms* of the form $R_1 \circ \cdots \circ R_n \sqsubseteq R$ subject to some regularity restrictions. The latter, which are necessary to guarantee decidability of reasoning, make it also possible to simulate such axioms with regular expressions. $\mathcal{SRIQ}$ also allow one to explicitly state certain properties of roles, including (ir)reflexivity, symmetry, and disjointness, which can be simulated in $\mathcal{ZIQ}$ using BRIAs and CIAs. To recall $\mathcal{SRIQ}$ KBs, we follow [31] and [38].

Definition 6.1 [$\mathcal{SRIQ}$ knowledge bases] A $\mathcal{SRIQ}$ *role inclusion axiom (SRIA)* is an expression of the form $R_1 \circ \cdots \circ R_n \sqsubseteq R$, where $n > 1$, $\{R_1, \ldots, R_n, R\} \subseteq \overline{\mathbf{R}}$, and $\overline{\mathbf{R}} = \mathbf{R} \cup \{R^- \mid R \in \mathbf{R}\}$.

A set $\mathcal{R}$ of SRIAs is *regular*, if there exists a partial order $\prec$ on $\overline{\mathbf{R}}$ such that $\mathsf{Inv}(R) \prec R'$ iff $R \prec R'$ for every $R, R' \in \overline{\mathbf{R}}$, and such that every SRIA in $\mathcal{R}$ is of one of the forms (i) $R \circ R \sqsubseteq R$, (ii) $\mathsf{Inv}(R) \sqsubseteq R$, (iii) $w \sqsubseteq R$, (iv) $w \circ R \sqsubseteq R$, or (v) $R \circ w \sqsubseteq R$, where $w = R_1 \circ \cdots \circ R_n$ and $R_i \prec R$ for each $1 \leq i \leq n$.

For a given set of SRIAs $\mathcal{R}$, the relation $\sqsubseteq_\mathcal{R}$ is the smallest relation such that (*i*) $R \sqsubseteq_\mathcal{R} R$ for every $R \in \overline{\mathbf{R}}$ such that R or $\mathsf{Inv}(R)$ occurs in $\mathcal{R}$, and (*ii*) $R_1 \circ \cdots \circ R_n \sqsubseteq_\mathcal{R} R$ for each $R_1 \circ \cdots \circ R_{i-1} \circ R' \circ R_{j+1} \circ \cdots \circ R_n \sqsubseteq_\mathcal{R} R$ such that $R_i \circ \cdots \circ R_j \sqsubseteq R' \in \mathcal{R}$ or $\mathsf{Inv}(R_j) \circ \cdots \circ \mathsf{Inv}(R_i) \sqsubseteq R' \in \mathcal{R}$, for some $R' \in \overline{\mathbf{R}}$ and $1 \leq i \leq j \leq n$. A role is *simple* in $\mathcal{R}$ if there are no roles $R_1, \ldots, R_n$ with $n \geq 2$ such that $R_1 \circ \cdots \circ R_n \sqsubseteq_\mathcal{R} R$.

An *assertion about roles* is an expression of the form $\mathsf{Sym}(R)$, $\mathsf{Ref}(R)$ $\mathsf{Irr}(R)$, or $\mathsf{Dis}(R, R')$, for roles $R, R' \in \overline{\mathbf{R}}$. [6] An assertion about roles is *simple* w.r.t. to a set $\mathcal{R}$ of SRIAs, if all roles occurring in it are

[6] We use the term *assertion about roles* instead of *role assertions* used in [31], since the latter is often used to refer to ABox assertions of the form $R(a, b)$. In [31] also $\mathsf{Tra}(R)$, asserting that R is transitive, is allowed. We omit this as it is equivalently

simple in $\mathcal{R}$ or it is of the form $\mathsf{Sym}(R)$.

A $\mathcal{SRIQ}$ *RBox* is a finite set $\mathcal{R} = \mathcal{R}^i \cup \mathcal{R}^a$ of SRIAs $\mathcal{R}^i$ and assertions about roles $\mathcal{R}^a$ such that $\mathcal{R}^i$ is regular and each assertion in $\mathcal{R}^a$ is simple w.r.t. to $\mathcal{R}^i$.

To define $\mathcal{SRIQ}$ TBoxes and ABoxes, we assume a given $\mathcal{SRIQ}$ RBox $\mathcal{R}$ containing the set $\mathcal{R}^i$ of SRIAs. Then $\mathcal{SRIQ}$ *concepts* C, C' obey the following syntax:

$$C, C' \longrightarrow A \mid \neg C \mid C \sqcap C' \mid C \sqcup C' \mid \forall R.C \mid \exists R.C \mid \geqslant n\, S.C \mid \leqslant n\, S.C \mid \exists S.\mathsf{Self},$$

where $A \in \mathbf{C}$, $R, S \in \overline{\mathbf{R}}$ and S is simple in $\mathcal{R}^i$. A $\mathcal{SRIQ}$ *concept inclusion axiom (SCIA)* is an expression $C \sqsubseteq C'$ for arbitrary $\mathcal{SRIQ}$ concepts C and C'; a $\mathcal{SRIQ}$ *TBox* is a set of SCIAs. A $\mathcal{SRIQ}$ *assertion* is an expression $C(a)$, $R(a, b)$, $\neg S(a, b)$ or $a \not\approx b$, where C is a $\mathcal{SRIQ}$ concept, S, R are $\mathcal{SRIQ}$ roles, S is simple in $\mathcal{R}^i$, and $a, b \in \mathbf{I}$; a $\mathcal{SRIQ}$ *ABox* is a set of $\mathcal{SRIQ}$ assertions. Given a $\mathcal{SRIQ}$ RBox $\mathcal{R}$ as above, a $\mathcal{SRIQ}$ *knowledge base* is a triple $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ where $\mathcal{A}$ is a non-empty ABox and $\mathcal{T}$ is a TBox.[7]

The semantics of $\mathcal{SRIQ}$ TBoxes and ABoxes is defined as for $\mathcal{ZIQ}$. An interpretation $\mathcal{I}$ satisfies an assertion about roles $\mathsf{Sym}(R)$, $\mathsf{Ref}(R)$, or $\mathsf{Irr}(R)$, if $R^{\mathcal{I}}$ is symmetric, reflexive, or irreflexive, respectively; $\mathcal{I}$ satisfies $\mathsf{Dis}(R, R')$ if the relations $R^{\mathcal{I}}$ and $R'^{\mathcal{I}}$ are disjoint, i.e., $R^{\mathcal{I}} \cap R'^{\mathcal{I}} = \emptyset$; $\mathcal{I}$ satisfies a SRIA $R_1 \circ \cdots \circ R_n \sqsubseteq R$ if $R_1^{\mathcal{I}} \circ \cdots \circ R_n^{\mathcal{I}} \subseteq R^{\mathcal{I}}$, where again we override the symbol $\circ$ and use it to denote binary role composition. An interpretation $\mathcal{I}$ is a model of an RBox $\mathcal{R}$ if it satisfies all SRIAs and all assertions about roles in $\mathcal{R}$, written $\mathcal{I} \models \mathcal{R}$. Modelhood of a KB is restricted in the natural way to the models of the RBox, i.e., $\mathcal{I} \models \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ iff $\mathcal{I} \models \mathcal{A}$, $\mathcal{I} \models \mathcal{T}$, and $\mathcal{I} \models \mathcal{R}$.

6.1 Reducing $\mathcal{SRIQ}$ to $\mathcal{ZIQ}$

We describe a rewriting that transforms a $\mathcal{SRIQ}$ KB $\mathcal{K}$ into an $\mathcal{ZIQ}$ KB $\Psi(\mathcal{K})$, in a way that will allow us to exploit our automata based algorithms for reasoning in $\mathcal{SRIQ}$.

The rewriting builds on the following property. The restriction to regular sets of SRIAs, which is crucial for the decidability of $\mathcal{SRIQ}$, ensures that all implications between roles can be described by regular language. More precisely:

Lemma 6.2 ([33]) *If $\mathcal{R}$ is a regular set of SRIAs, then for each $R \in \overline{\mathbf{R}}$ occurring in $\mathcal{K}$, the set $L_{\mathcal{R}}(R) = \{R_1 \cdot \cdots \cdot R_n \mid R_1 \circ \cdots \circ R_n \sqsubseteq_{\mathcal{R}} R\}$ is a regular language.*

Further, the authors of [33] show how to construct a finite state automaton representing $L_{\mathcal{R}}(R)$. This automaton is equivalent to a regular expression $\rho^{\mathcal{R}}(R)$ over the language $\overline{\mathbf{R}}$, i.e. to an $\mathcal{ZIQ}$ role. In particular, if a role S is simple in $\mathcal{R}$, the resulting expression is the simple $\mathcal{ZIQ}$ role $\bigcup_{(S' \sqsubseteq_{\mathcal{R}} S) \in \mathcal{R}} S'$. Since $R \in L_{\mathcal{R}}(R)$ holds for every R, and since $w \sqsubseteq_{\mathcal{R}} R$ implies $w^{\mathcal{I}} \subseteq R^{\mathcal{I}}$ in each model of $\mathcal{R}$, we easily obtain the following corollary:

Corollary 6.3 *Given a regular set of SRIAs $\mathcal{R}$, we can construct, for each $R \in \overline{\mathbf{R}}$, an $\mathcal{ZIQ}$ role $\rho^{\mathcal{R}}(R)$ such that, for every interpretation $\mathcal{I}$, $R^{\mathcal{I}} \subseteq (\rho^{\mathcal{R}}(R))^{\mathcal{I}}$, and $\mathcal{I} \models \mathcal{R}$ implies $(\rho^{\mathcal{R}}(R))^{\mathcal{I}} \subseteq R^{\mathcal{I}}$. Moreover, $\rho^{\mathcal{R}}(R)$ is a simple role whenever R is simple in $\mathcal{R}$.*

The rewriting Ψ exploits this lemma. In what follows, we assume a fixed given regular set of SRIAs $\mathcal{R}$, and for each $R \in \overline{\mathbf{R}}$, $\rho^{\mathcal{R}}(R)$ denotes an arbitrary but fixed regular expression as described above.

---

expressed with a SRIA $R \circ R \sqsubseteq R$.

[7] As in Definition 2.2, we only consider w.l.o.g. non-empty ABoxes.

Reducing concepts and TBoxes Since SRIAs are not supported in $\mathcal{ZIQ}$, we need to ensure that the interpretation of concepts in $\Psi(\mathcal{K})$ respect the restrictions that arise from them. This can be achieved by replacing each role R by the regular expression $\rho^{\mathcal{R}}(R)$.

Definition 6.4 For any $\mathcal{SRIQ}$ concept C, we denote by $\Psi_{\mathcal{R}}(C)$ the $\mathcal{ZIQ}$ concept that results from replacing each role R in C with $\rho^{\mathcal{R}}(R)$. For a $\mathcal{SRIQ}$ TBox $\mathcal{T}$, we define $\Psi_{\mathcal{R}}(\mathcal{T}) = \{\Psi_{\mathcal{R}}(C) \sqsubseteq \Psi_{\mathcal{R}}(D) \mid C \sqsubseteq D \in \mathcal{T}\}$. ∎

In the models of $\mathcal{R}$, this transformation is equivalence preserving.

Lemma 6.5 *Let $\mathcal{I}$ be an interpretation such that $\mathcal{I} \models \mathcal{R}$. Then $C^{\mathcal{I}} = (\Psi_{\mathcal{R}}(C))^{\mathcal{I}}$ for each $\mathcal{SRIQ}$ concept C, and $\mathcal{I} \models \mathcal{T}$ iff $\mathcal{I} \models \Psi_{\mathcal{R}}(\mathcal{T})$ for each $\mathcal{SRIQ}$ TBox $\mathcal{T}$.*

Reducing ABoxes We replace each concept C in an assertion of the form $C(a)$ by the corresponding $\mathcal{ZIQ}$ concept $\Psi(C)$. We also need to remove the *negated role membership assertions* $\neg S(a, b)$, which are not allowed in $\mathcal{ZIQ}$. We simulate them using a fresh role name for each assertion, together with a BRIA that ensures that the fresh symbol is interpreted as the desired role negation.

Definition 6.6 Given a $\mathcal{SRIQ}$ ABox $\mathcal{A}$, we define the following:

- $\Psi_{\mathcal{R}}(\mathcal{A})$ is the $\mathcal{ZIQ}$ ABox obtained by replacing in $\mathcal{A}$ (i) each assertion $C(a)$ by $\Psi(C)(a)$, and (ii) each assertion $\neg S(a, b)$ by $P_{\neg S}(a, b)$ for a fresh role name $P_{\neg S}$.
- $\mathcal{T}^{\mathcal{A}}_{\mathcal{R}}$ is the TBox containing $P_{\neg S} \cap S \sqsubseteq \mathsf{B}$ for each $\neg S(a, b)$ in $\mathcal{A}$. ∎

Similarly as above, we obtain:

Lemma 6.7 *Let $\mathcal{I}$ be an interpretation such that $\mathcal{I} \models \mathcal{R}$. Then, for every $\mathcal{SRIQ}$ ABox $\mathcal{A}$, $\mathcal{I} \models \mathcal{A}$ iff $\mathcal{I} \models \Psi_{\mathcal{R}}(\mathcal{A})$ and $\mathcal{I} \models \mathcal{T}^{\mathcal{A}}_{\mathcal{R}}$.*

Reducing Assertions about Roles Finally, the assertions about roles in the RBox will be rewritten as part of the TBox, using BRIAs and CIAs.

Definition 6.8 For every set $\mathcal{R}^a$ of assertions about roles, $\Psi(\mathcal{R}^a)$ is the following $\mathcal{ZIQ}$ TBox:

$$\begin{aligned}\Psi(\mathcal{R}) \quad = \quad & \{\mathsf{Inv}(R) \sqsubseteq R \mid \mathsf{Sym}(R) \in \mathcal{R}\} \quad \cup \quad \{\top \sqsubseteq \exists R.\mathsf{Self} \mid \mathsf{Ref}(R) \in \mathcal{R}\} \quad \cup \\ & \{\exists R.\mathsf{Self} \sqsubseteq \bot \mid \mathsf{Irr}(R) \in \mathcal{R}\} \quad \cup \quad \{R \cap R' \sqsubseteq \mathsf{B} \mid \mathsf{Dis}(R, R') \in \mathcal{R}\}.\end{aligned}$$

∎

Lemma 6.9 *Let $\mathcal{R}^a$ be a set of assertions about roles. Then, for every interpretation $\mathcal{I}$, $\mathcal{I} \models \mathcal{R}^a$ iff $\mathcal{I} \models \Psi(\mathcal{R}^a)$.*

Reducing KBs Now we are ready to define the rewritten KB $\Psi(\mathcal{K})$.

Definition 6.10 For every $\mathcal{SRIQ}$ KB $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$, let $\Psi(\mathcal{K}) = \langle \Psi_{\mathcal{R}^i}(\mathcal{A}), \mathcal{T}' \rangle$, where $\mathcal{T}' = \Psi_{\mathcal{R}^i}(\mathcal{T}) \cup \mathcal{T}^{\mathcal{A}}_{\mathcal{R}^i} \cup \Psi(\mathcal{R}^a)$, and $\mathcal{R}^i$ and $\mathcal{R}^a$ respectively denote the set of SRIAs and the set of assertions about roles in $\mathcal{R}$. ∎

From Lemmas 6.5, 6.7, and 6.9, we easily get:

Lemma 6.11 *For every interpretation $\mathcal{I}$, $\mathcal{I} \models \mathcal{K}$ implies $\mathcal{I} \models \Psi(\mathcal{K})$.*

The converse holds only in a slightly weaker form, as the SRIAs of $\mathcal{K}$ need not be satisfied in every model $\mathcal{I}$ of $\Psi(\mathcal{K})$. However, each $\mathcal{I}$ can be transformed into a model of the SRIAs by adding all implied pairs of individuals to the extension of the roles. Using Lemmas 6.5, 6.7, and 6.9, we can then easily prove the following.

Lemma 6.12 *Let $\mathcal{I}$ be an interpretation such that $\mathcal{I} \models \Psi(\mathcal{K})$, and let $\mathcal{I}'$ be the interpretation that has $R^{\mathcal{I}'} = (\rho^{\mathcal{R}}(R))^{\mathcal{I}}$ for each role $R \in \mathbf{R}$ occurring in $\mathcal{K}$, and that is identical to $\mathcal{I}$ otherwise. Then $\mathcal{I}' \models \mathcal{K}$.*

Lemmas 6.11 and 6.12 provide a reduction from KB satisfiability in $\mathcal{SRIQ}$ to KB satisfiability in $\mathcal{ZIQ}$.

Proposition 6.13 *Let $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ be a $\mathcal{SRIQ}$ KB. Then $\mathcal{K}$ is satisfiable iff $\Psi(\mathcal{K})$ is satisfiable.*

An alternative translation from $\mathcal{SRIQ}$ to $\mathcal{ZIQ}$ can be defined, for example, by considering some normal form of KBs that only allows for universal concepts in GCIs of the form $C \sqsubseteq \forall R.A$ with A a concept name, and that does not allow complex concepts in ABox assertions (such normal forms are well known; see, e.g. [38]). A normal KB $\langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ is rewritten as $\langle \mathcal{A}', \mathcal{T}' \rangle$, where $\mathcal{A}'$ is obtained by removing the negated ABox assertions as in Definition 6.6 and adding the corresponding BRIAs to $\mathcal{T}$, while additionally replacing R with $\rho^{\mathcal{R}}(R)$ in each universal concept $\forall R.A$, and adding CIAs and BRIAs for the assertions about roles as in Definition 6.8 to obtain $\mathcal{T}'$.

6.2 Deciding KB satisfiability

Due to Proposition 6.13, the automata algorithm in Section 4 can be used to decide the satisfiability of $\mathcal{SRIQ}$ knowledge bases. Assuming that the number restrictions are coded in unary, the resulting algorithm is worst-case optimal. Let $\rho^*_{\mathcal{R}}$ be a longest regular expression ρ_R, $R \in \overline{\mathbf{R}}_{\mathcal{R}}$. All steps of the rewriting Ψ are clearly polynomial in the size of $\mathcal{A}$, $\mathcal{T}$, and $\rho^*_{\mathcal{R}}$. However, the size of $\rho^*_{\mathcal{R}}$ can be exponential in the size of $\mathcal{R}$ [33].

Theorem 6.14 *The satisfiability of a given $\mathcal{SRIQ}$ knowledge base $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ is decidable in time exponential in the combined size of $\mathcal{T}$, $\mathcal{A}$, and $\rho^*_{\mathcal{R}}$, and double exponential in the size of $\mathcal{K}$.*

As shown in [38], $\mathcal{SRIQ}$ is 2ExpTime-hard. Hence our bound is optimal.

Corollary 6.15 *Deciding whether a given KB in $\mathcal{SRIQ}$ is satisfiable is* 2ExpTime*-complete.*

Note that the blowup in complexity w.r.t. $\mathcal{ZIQ}$ is due to the size of $\rho^*_{\mathcal{R}}$, and that the algorithm is single exponential whenever $\rho^*_{\mathcal{R}}$ has size polynomial in $\mathcal{R}$, e.g., for the so-called *simple role hierarchies* defined in [33]. This compares well to the $\mathcal{SRIQ}$ algorithm given in [31] which, even for these restricted cases, may require time that is non-deterministic double exponential in $\mathcal{K}$.

6.3 Query Answering in $\mathcal{SRIQ}$

We also have an algorithm to decide query entailment in $\mathcal{SRIQ}$. To this end, we rewrite a P2RPQ q over $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$ into a query $\Psi_{\mathcal{R}}(q)$ over $\Psi(\mathcal{K})$, in such a way that query entailment is preserved.

Definition 6.16 For every P2RPQ q over a $\mathcal{SRIQ}$ knowledge base $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$, let $\Psi_{\mathcal{R}}(q)$ be the P2RPQ that results from substituting in q every occurrence of each role R by $\rho^{\mathcal{R}}(R)$. ∎

Note that $\Psi_{\mathcal{R}}(q)$ may contain regular expression while q does not, i.e., our technique reduces positive (resp. conjunctive) queries over $\mathcal{SRIQ}$ to positive (resp. conjunctive) regular path queries over $\mathcal{ZIQ}$.

Lemma 6.17 *Let q be a P2RPQ over a $\mathcal{SRIQ}$ knowledge base $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}, \mathcal{R} \rangle$. Then $\mathcal{K} \models q$ iff $\Psi(K) \models \Psi_{\mathcal{R}}(q)$.*

Proof. First observe that in every interpretation $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{R}$, a match for $\Psi(q)$ is a match for q. For the converse, let $\mathcal{I}'$ denote the modified version of the interpretation $\mathcal{I}$ in which each R is interpreted as $(\rho^{\mathcal{R}}(R))^{\mathcal{I}}$ (c.f. Lemma 6.12). Then every match for q in $\mathcal{I}'$ is a match for q in $\Psi_{\mathcal{R}}(q)$,.

Now suppose $\mathcal{K} \models q$, and consider an interpretation $\mathcal{I}$ such that $\mathcal{I} \models \Psi(\mathcal{K})$. Then $\mathcal{I}' \models \mathcal{K}$ by Lemma 6.12, thus $\mathcal{I}' \models q$ and $\mathcal{I} \models \Psi_{\mathcal{R}}(q)$; hence, $\Psi(\mathcal{K}) \models \Psi_{\mathcal{R}}(q)$. Conversely, suppose $\Psi(\mathcal{K}) \models \Psi_{\mathcal{R}}(q)$. Consider an arbitrary $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{K}$. By Lemma 6.11 we know that $\mathcal{I} \models \Psi(\mathcal{K})$. Therefore $\mathcal{I} \models \Psi_{\mathcal{R}}(q)$, and since $\mathcal{I} \models \mathcal{R}$, it follows $\mathcal{I} \models q$. □

Again, longest regular expression $\rho^*_{\mathcal{R}}$ may exponentially influence the overall complexity of the algorithm.

Theorem 6.18 *Query entailment $\mathcal{K} \models q$ for a given $\mathcal{SRIQ}$ KB $\mathcal{K}$ and a P2RPQ q over $\mathcal{K}$ is decidable (a) in double exponential time in the combined size of q, $C_{\mathcal{K}}$, $\mathbf{I}_{\mathcal{K}}$, and $\rho^*_{\mathcal{R}}$, and (b) in triple exponential time in the combined size of q and $\mathcal{K}$.*

7 Conclusion

In this paper, we have substantially pushed the frontier of decidable query answering over expressive Description Logics (DLs), which is an active area of research driven by the growing interest in deploying DLs to various application areas. Exploiting automata-theoretic results and methods, we have shown that query entailment for a very rich class of queries beyond the popular (union of) conjunctive queries, namely the positive (existential) two-way regular path queries (P2RPQs), is decidable over knowledge bases in the DL $\mathcal{ALCQIb}_{reg}$. Making use of this result, we also show decidability of query entailment over knowledge bases in the DL $\mathcal{SRIQ}$, which underlies the nominal-free fragment of the new OWL 2 ontology standard by the W3C.

More precisely, we have shown that for $\mathcal{ALCQIb}_{reg}$, the query entailment problem is 2ExpTime-complete, while for $\mathcal{SRIQ}$ it is in 3ExpTime. Given that conjunctive query entailment is 2ExpTime-hard already for the DLs $\mathcal{ALCI}$ [44] and $\mathcal{SH}$ [23], our results show that both on the query and the knowledge base side, one can increase the expressiveness substantially without a further increase in worst-case complexity. In particular, this applies to queries that allow one to navigate the models of a knowledge base in order to connect distant elements of the model, which is desired for instance in semistructured data models.

The automata-based technique we apply is, in a sense, more accessible than previous ones that are based on tableaux or resolution-based transformations to disjunctive datalog. It is computational in nature and

works directly on models of a knowledge base, processing them with flexible local operations; furthermore, subtasks can be modularly combined. Thanks to the technique, we are also able to obtain more general results, which seems more difficult using the other approaches. Indeed, this has been confirmed by [17], where along the lines and ideas of this paper, but using different automata models the decidability frontier for entailment of P2RPQs has been extended to $\mathcal{ALCQOI}b_{reg}$ and $\mathcal{ALCOI}b_{reg}$ as well as to $\mathcal{SROQ}$ and $\mathcal{SROI}$. Furthermore, also decidability results for query containment are given there, which are obtained by a reduction to query answering, extending well-known relationships between query containment and conjunctive query answering to the richer setting of P2RPQs.

Our results thus 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 the complexity of this problem. However, these kind of techniques have so far resisted implementation, even for simpler problems, such as KB satisfiability. Hence, we do not expect the results presented here to lead to practicable algorithms in the near future. It now becomes interesting to look for alternative techniques that are better suited for implementation. We are confident that the tight complexity bounds that we have established will provide a valuable guidance in this direction, and may provide interesting insights to exploit, for instance, tableaux as done in [48], or knots as in [22].

8 Appendix

In this section, we provide a proof of the canonical model property of $\mathcal{ZIQ}$ stated in Theorem 3.9.

In what follows, we denote by $\mathcal{K} = \langle \mathcal{A}, \mathcal{T} \rangle$ an arbitrary but fixed normal $\mathcal{ZIQ}$ KB. To show that $\mathcal{K}$ has a $k_{C_\mathcal{T}}$-canonical model, we will follow the lines of similar proofs for the μ-calculus in [6, 57] (which in turn, are adaptations of the original proof in [52]), and adapt them to the syntax of $\mathcal{ZIQ}$, while accommodating the ABox, Booleans over roles, and Self. We will show that if $\mathcal{K}$ has a model, then it has a *well-founded adorned pre-model*. Roughly, the latter is a model enhanced with additional information that allows us to 'trace' the satisfaction of the $\exists R^*.C$ concepts. Then we show that an adorned well-founded pre-model can be unraveled into an adorned well-founded pre-model that is $k_{C_\mathcal{T}}$-canonical, and that we can easily extract a $k_{C_\mathcal{T}}$-canonical model of $\mathcal{K}$ from it.

We start by defining *concept* and *role atoms*, which are consistent sets of concepts and roles from the syntactic closure of $C_\mathcal{T}$. In what follows, we denote by $Cl_{\mathsf{C}}(C_\mathcal{T})$ and $Cl_{\mathsf{R}}(C_\mathcal{T})$ the set of concepts and the set of roles in $Cl(C_\mathcal{T})$, respectively. A *concept atom* of $\mathcal{K}$ is a set $At \subseteq Cl_{\mathsf{C}}(C_\mathcal{T})$ of concepts closed under the rules of Table 6, while a *role atom* of $\mathcal{K}$ is a set $AtR \subseteq Cl_{\mathsf{R}}(C_\mathcal{T})$ of simple roles closed under the rules of Table 7. The set of all concept and the set of all role atoms of $\mathcal{K}$ are respectively denoted by $\mathsf{atC}(\mathcal{K})$ and $\mathsf{atR}(\mathcal{K})$.

A *pre-model* is an interpretation $\mathcal{I}$ in which each object is mapped to a concept atom and each pair of objects to a role atom. Formally, a *pre-model* of $\mathcal{K}$ is a pair $\langle \mathcal{I}, \theta \rangle$ where $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ is an interpretation for $\mathcal{K}$ and θ is a function that maps each $d \in \Delta^\mathcal{I}$ to a concept atom $\theta(d) \in \mathsf{atC}(\mathcal{K})$ and each $(d, d') \in \Delta^\mathcal{I} \times \Delta^\mathcal{I}$ to a role atom $\theta((d, d')) \in \mathsf{atR}(\mathcal{K})$ such that

(1) $C_\mathcal{T} \in \theta(a^\mathcal{I})$ for each $a \in \mathbf{I}_\mathcal{K}$,

(2) $A(a) \in \mathcal{A}$ implies $A \in \theta(a^\mathcal{I})$, and $p(a, b) \in \mathcal{A}$ implies $p \in \theta(a^\mathcal{I}, b^\mathcal{I})$,

(3) for each $d, d' \in \Delta^\mathcal{I}$ and $p \in Cl(C_\mathcal{T})$, $p \in \theta((d, d'))$ implies $(d, d') \in p^\mathcal{I}$, and $\neg p \in \theta((d, d'))$ implies $(d, d') \notin p^\mathcal{I}$,

if	A is a concept name in $Cl_{\mathsf{C}}(C_{\mathcal{T}})$,	then	$A \in At$	iff	$\neg A \notin At$
if	$C \sqcap C' \in Cl_{\mathsf{C}}(C_{\mathcal{T}})$,	then	$C \sqcap C' \in At$	iff	$\{C, C'\} \subseteq At$
if	$C \sqcup C' \in Cl_{\mathsf{C}}(C_{\mathcal{T}})$,	then	$C \sqcup C' \in At$	iff	$\{C, C'\} \cap At \neq \emptyset$
if	$\exists S.C \in Cl_{\mathsf{C}}(C_{\mathcal{T}})$,	then	$\exists S.C \in At$	iff	$\geqslant 1\, S.C \in At$
if	$\forall S.C \in Cl_{\mathsf{C}}(C_{\mathcal{T}})$,	then	$\forall S.C \in At$	iff	$\leqslant 0\, S.{\sim}C \in At$
if	$\exists(R \cup R').C \in Cl_{\mathsf{C}}(C_{\mathcal{T}})$,	then	$\exists(R \cup R').C \in At$	iff	$\{\exists R.C,\ \exists R'.C\} \cap At \neq \emptyset$
if	$\exists(R \circ R').C \in Cl_{\mathsf{C}}(C_{\mathcal{T}})$,	then	$\exists(R \circ R').C \in At$	iff	$\exists R.\exists R'.C \in At$
if	$\exists R^*.C \in Cl_{\mathsf{C}}(C_{\mathcal{T}})$,	then	$\exists R^*.C \in At$	iff	$\{C, \exists R.\exists R^*.C\} \cap At \neq \emptyset$
if	$\exists id(C).C' \in Cl_{\mathsf{C}}(C_{\mathcal{T}})$,	then	$\exists id(C).C' \in At$	iff	$\{C,\ C'\} \subseteq At$
if	$\forall(R \cup R').C \in Cl_{\mathsf{C}}(C_{\mathcal{T}})$,	then	$\forall(R \cup R').C \in At$	iff	$\{\forall R.C,\ \forall R'.C\} \subseteq At$
if	$\forall(R \circ R').C \in Cl_{\mathsf{C}}(C_{\mathcal{T}})$,	then	$\forall(R \circ R').C \in At$	iff	$\forall R.\forall R'.C \in At$
if	$\forall R^*.C \in Cl_{\mathsf{C}}(C_{\mathcal{T}})$,	then	$\forall R^*.C \in At$	iff	$\{C, \forall R.\forall R^*.C\} \subseteq At$
if	$\forall id(C).C' \in Cl_{\mathsf{C}}(C_{\mathcal{T}})$,	then	$\forall id(C).C' \in At$	iff	$\{{\sim}C,\ C'\} \cap At \neq \emptyset$

Table 6: Concept atom $At \subseteq Cl_{\mathsf{C}}(C_{\mathcal{T}})$

if	p is a role name in $Cl_{\mathsf{R}}(C_{\mathcal{T}})$,	then	$p \in AtR$	iff	$\neg p \notin AtR$
if	$S \cap S' \in Cl_{\mathsf{R}}(C_{\mathcal{T}})$,	then	$S \cap S' \in AtR$	iff	$\{S, S'\} \subseteq AtR$
if	$S \cup S' \in Cl_{\mathsf{R}}(C_{\mathcal{T}})$,	then	$S \cup S' \in AtR$	iff	$\{S, S'\} \cap AtR \neq \emptyset$

Table 7: Role atom $AtR \subseteq Cl_{\mathsf{R}}(C_{\mathcal{T}})$

(4) for each $d, d' \in \Delta^{\mathcal{I}}$ and $p \in Cl(C_{\mathcal{T}})$, $p \in \theta((d, d'))$ iff $p^- \in \theta((d', d))$, and

(5) for each $d \in \Delta^{\mathcal{I}}$

(a) $A \in \theta(d)$ implies $d \in A^{\mathcal{I}}$ and $\neg A \in \theta(d)$ implies $d \notin A^{\mathcal{I}}$, for each concept name $A \in Cl(C_{\mathcal{T}})$,

(b) $\exists S.\mathsf{Self} \in \theta(d)$ implies $S \in \theta((d, d))$,

(c) if $\geqslant n\, S.C \in \theta(d)$, then there is some $V \subseteq \mathsf{neigh}_{\mathcal{I},\theta}(S, d)$ such that $|V| \geq n$ and $C \in \theta(d')$ for every $d' \in V$, and

(d) if $\leqslant n\, S.C \in \theta(d)$, then there is some $V \subseteq \mathsf{neigh}_{\mathcal{I},\theta}(S, d)$ such that $|V| \leq n$ and ${\sim}C \in \theta(d')$ for every $d' \in \mathsf{neigh}_{\mathcal{I},\theta}(S, d) \setminus V$,

where $\mathsf{neigh}_{\mathcal{I},\theta}(S, d) = \{d' \in \Delta^{\mathcal{I}} \mid S \in \theta((d, d'))\}$.

Intuitively, $\langle \mathcal{I}, \theta \rangle$ is almost a model of $\mathcal{K}$, except that it is not ensured that concepts of the form $\exists R^*.C$ are satisfied. Instead, if $\exists R^*.C$ must hold at some element d, we only require that some R neighbour of d satisfies $\exists R^*.C$, and the satisfaction of C may be infinitely postponed. To trace the evaluation of $\exists R^*.C$ concepts and to distinguish pre-models that represent models of $\mathcal{K}$, we introduce *adorned pre-models* $\langle \mathcal{I}, \theta, \mathsf{ch} \rangle$ that extend pre-models $\langle \mathcal{I}, \theta \rangle$ with a *choice function*.

A *choice function* for a pre-model $\langle \mathcal{I}, \theta \rangle$ of $\mathcal{K}$ is a partial function ch such that:

- for each pair $(d, C \sqcup C')$ with $d \in \Delta^{\mathcal{I}}$ and $C \sqcup C' \in \theta(d)$, $\mathsf{ch}(d, C \sqcup C')$ is a concept in $\{C, C'\} \cap \theta(d)$;
- for each pair $(d, \geqslant n\, S.C)$ with $d \in \Delta^{\mathcal{I}}$ and $\geqslant n\, S.C \in \theta(d)$, $\mathsf{ch}(d, \geqslant n\, S.C)$ is a subset V of $\mathsf{neigh}_{\mathcal{I},\theta}(S, d)$ such that $|V| \geq n$ and $C \in \theta(d')$ for every $d' \in V$; and

- for each pair $(d, \leqslant n\, S.C)$ with $d \in \Delta^{\mathcal{I}}$ and $\leqslant n\, S.C \in \theta(d)$, $\mathsf{ch}(d, \leqslant n\, S.C)$ is a subset V of $\mathsf{neigh}_{\mathcal{I},\theta}(S, d)$ such that $|V| \leq n$ and $\sim C \in \theta(d')$ for every $d' \in \mathsf{neigh}_{\mathcal{I},\theta}(S, d) \setminus V$.

For an adorned pre-model $\langle \mathcal{I}, \theta, \mathsf{ch}\rangle$ of $\mathcal{K}$, the *derivation relation* $\vdash \subseteq\ (\Delta^{\mathcal{I}} \times Cl(C_{\mathcal{T}})) \times (\Delta^{\mathcal{I}} \times Cl(C_{\mathcal{T}}))$ is the smallest relation such that for every $d \in \Delta^{\mathcal{I}}$:

- $C \sqcup C' \in \theta(d)$ implies $(d, C \sqcup C') \vdash (d, \mathsf{ch}(d, C \sqcup C'))$,
- $C \sqcap C' \in \theta(d)$ implies $(d, C \sqcap C') \vdash (d, C)$ and $(d, C \sqcap C') \vdash (d, C')$,
- $\geqslant n\, S.C \in \theta(d)$ implies $(d, \geqslant n\, S.C) \vdash (d', C)$ for every $d' \in \mathsf{ch}(d, \geqslant n\, S.C)$,
- $\leqslant n\, S.C \in \theta(d)$ implies $(d, \leqslant n\, S.C) \vdash (d', C)$ for every $d' \in \mathsf{neigh}_{\mathcal{I},\theta}(S, d) \setminus \mathsf{ch}(d, \leqslant n\, S.C)$,
- $\exists R^*.C \in \theta(d)$ implies $(d, \exists R^*.C) \vdash (d, C \sqcup \exists R.\exists R^*.C)$, and
- $\forall R^*.C \in \theta(d)$ implies $(d, \forall R^*.C) \vdash (d, C \sqcap \forall R.\forall R^*.C)$.

A concept $\exists R^*.C$ is *regenerated* from d to d' in $\langle \mathcal{I}, \theta, \mathsf{ch}\rangle$, if there is a sequence $(d_1, C_1), \ldots, (d_k, C_K)$ with $k > 1$ such that $d_1 = d$, $d_k = d'$, $C_1 = C_k = \exists R^*.C$, $\exists R^*.C$ is a subconcept of every C_i and $(d_i, C_i) \vdash (d_{i+1}, C_{i+1})$ for each $1 \leq i < k$. We say that $\langle \mathcal{I}, \theta, \mathsf{ch}\rangle$ is *well-founded*, if there is no $\exists R^*.C \in Cl(C_{\mathcal{T}})$ and infinite sequence $d_1, d_2, \ldots$ such that $\exists R^*.C$ is *regenerated* from d_i to d_{i+1} for every $i \geq 1$. Then one can show:

Lemma 8.1 *For every normal $\mathcal{ZIQ}$ KB $\mathcal{K}$, the following holds.*

1. *If $\mathcal{K} \not\models q$ for some q, then $\mathcal{K}$ has a well-founded adorned pre-model $\langle \mathcal{I}, \theta, \mathsf{ch}\rangle$ such that $\mathcal{I} \not\models q$.*
2. *If $\langle \mathcal{I}, \theta, \mathsf{ch}\rangle$ is a well-founded adorned pre-model of $\mathcal{K}$, then $\mathcal{I}$ is a model of $\mathcal{K}$.*

Proof. [Sketch] The proof is essentially and adaptation of the ones in [6, 57], which extend the original proof for the μ-calculus in [52]. The absence of alternating fixpoints makes our setting somehow simpler, and the presence of the ABox and the additional constructs are not hard to accommodate.

For the first item, if $\mathcal{K} \not\models q$ for some q, then there is some $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{K}$ and $\mathcal{I} \not\models q$. The existence of a θ such that $\langle \mathcal{I}, \theta\rangle$ is a pre-model is straightforward: we simply set $\theta(d) = \{C \in Cl_{\mathsf{C}}(C_T) \mid d \in C^{\mathcal{I}}\}$ and $\theta(d, d') = \{S \in Cl_{\mathsf{R}}(C_T) \mid (d, d') \in S^{\mathcal{I}}\}$ for every $d, d' \in \Delta^{\mathcal{I}}$. The existence of a choice function ch that makes $\langle \mathcal{I}, \theta, \mathsf{ch}\rangle$ a well-founded adorned pre-model is also proved in the standard way. Roughly, while a choice function trivially exists, to prove well foundedness one observes that for every formula $\exists R^*.C$ such that $d \in \exists R^*.C^{\mathcal{I}}$ there is some finite sequence of elements reachable via R that lead to some $d' \in C^{\mathcal{I}}$; selecting such a path for the choice function avoids infinite regeneration of $\exists R^*.C$.

For the second item, one can show that: (†) if $\langle \mathcal{I}, \theta, \mathsf{ch}\rangle$ is a well-founded adorned pre-model, then $d \in C^{\mathcal{I}}$ for every $C \in \theta(d)$ and $(d, d') \in S^{\mathcal{I}}$ for every $S \in \theta(d, d')$; this can be shown by structural induction. We remark that Boolean role expressions are handled analogously as concept ones, while items 4 and 5b ensure the correct interpretation of inverse roles and Self concepts, respectively. For concepts of the form $\exists R^*.C$, we rely on the well-foundedness, which ensures that $\exists R^*.C$ is not infinitely regenerated and that C is eventually satisfied. Once (†) has been shown, it is easy to see that $\mathcal{I} \models \mathcal{K}$: item 1 ensures the satisfaction of the TBox, and item 2 of the ABox. □

Now we are ready to prove Theorem 3.9, i.e., that for $\mathcal{K}$ a normal $\mathcal{ZIQ}$ KB and q a P2RPQ, if $\mathcal{K} \not\models q$, then there is a $k_{C_{\mathcal{T}}}$-canonical model of $\mathcal{K}$ that admits no pre-match for q.

Proof of Theorem 3.9. Assume $\mathcal{K} \not\models q$. By item 1 of Lemma 8.1, there is some well-founded adorned pre-model $\langle \mathcal{I}, \theta, \mathsf{ch} \rangle$ of $\mathcal{K}$ such that $\mathcal{I} \not\models q$. We unravel $\langle \mathcal{I}, \theta, \mathsf{ch} \rangle$ into an adorned pre-model $\langle \mathcal{I}', \theta', \mathsf{ch}' \rangle$ that is also well founded, such that $\mathcal{I}'$ is a $k_{C_\mathcal{T}}$-canonical interpretation, and such that $\mathcal{I}' \not\models q$. By item 2 of Lemma 8.1, this implies that $\mathcal{I}$ is a $k_{C_\mathcal{T}}$-canonical model of $\mathcal{K}$ with $\mathcal{I}' \not\models q$, which proves the result.

We will inductively build the domain $\Delta^{\mathcal{I}'}$ of $\mathcal{I}'$ as a tree, and define a mapping $\tau : \Delta^{\mathcal{I}'} \to \Delta^{\mathcal{I}}$ that keeps track of the correspondence between nodes of the tree and elements of $\mathcal{I}$, which we use for defining the interpretation of concepts and roles in $\mathcal{I}'$. The functions θ' and ch' are defined simultaneously.

Let $\mathsf{R}(\mathcal{I}) = \{a^{\mathcal{I}} \mid a \in \mathbf{I}_\mathcal{K}\}$. Intuitively, to build the tree $\Delta^{\mathcal{I}'}$, we start with the roots $1, \dots, |\mathsf{R}(\mathcal{I})|$, and then continue building trees rooted at these roots, by inductively adding new levels.

For the base case, we let $\Delta^{\mathcal{I}'} := \{1, \dots, |\mathsf{R}(\mathcal{I})|\}$ and let $\tau : \{1, \dots, |\mathsf{R}(\mathcal{I})|\} \to \mathsf{R}(\mathcal{I})$ be an arbitrary bijection. Then we set, for each $j, j' \in \Delta^{\mathcal{I}'}$:

- $\theta'(j) = \theta(\tau(j))$,
- $\theta'((j, j')) = \theta((\tau(j), \tau(j')))$, and
- for each $C \sqcup C' \in \theta'(j)$, $\mathsf{ch}'(j, C \sqcup C') = \mathsf{ch}(\tau(j), C \sqcup C')$.

Choices for $\geqslant n\, S.C$ and $\leqslant n\, S.C$ concepts are defined in the induction step.

For the induction step, consider an $x \in \Delta^{\mathcal{I}'}$ of maximal length, and let $(\geqslant n_1\, S_1.C_1, e_1), \dots, (\geqslant n_m\, S_m.C_m, e_m)$ be all pairs of a formula $\geqslant n_i\, S_i.C_i \in \theta'(x)$ and an $e_i \in \mathsf{ch}(\tau(x), \geqslant n_i\, S_i.C_i)$. For each $1 \leq i \leq m$, we define:

$$\phi(e_i) \quad = \quad \begin{cases} y, & \text{if } e_i = \tau(y) \text{ for } y = x \text{ or } y = x{\cdot}{-}1, \\ x{\cdot}i & \text{otherwise.} \end{cases}$$

Then we set $\Delta^{\mathcal{I}'} := \Delta^{\mathcal{I}'} \cup \{\phi(e_1), \dots, \phi(e_m)\}$ and $\tau(x{\cdot}i) = e_i$ for each $x{\cdot}i \in \Delta^{\mathcal{I}'}$. To extend θ', set for each $x{\cdot}i \in \Delta^{\mathcal{I}'}$

- $\theta'(x{\cdot}i) = \theta(\tau(x{\cdot}i))$,
- $\theta'(y, x{\cdot}i) = \theta(\tau(y), \tau(x{\cdot}i))$ if $y = x$ or $y = x{\cdot}i$,
- $\theta'(x{\cdot}i, x) = \theta(\tau(x{\cdot}i), \tau(x))$, and
- $\theta'(y, x{\cdot}i) = t_\emptyset$ for every other y, where $t_\emptyset$ is a type such that $\neg p \in t_\emptyset$ for every $p \in \mathbf{R}$.

Finally, we extend the choice to concepts of the form $C \sqcup C'$ in the new $\theta'(x{\cdot}i)$, and to concepts $\geqslant n\, S.C$ and $\leqslant n\, S.C$ in $\theta'(x)$:

- for each $x{\cdot}i \in \Delta^{\mathcal{I}'}$ and each $C \sqcup C' \in \theta'(x{\cdot}i)$, $\mathsf{ch}'(x{\cdot}i, C \sqcup C') = \mathsf{ch}(\tau(x{\cdot}i), C \sqcup C')$;
- for each $\geqslant n\, S.C \in \theta'(x)$, $\mathsf{ch}'(x, \geqslant n\, S.C) = \{\phi(e) \mid e \in \mathsf{ch}(\tau(x), \geqslant n\, S.C)\}$; and
- for each $\leqslant n\, S.C \in \theta'(x)$, $\mathsf{ch}'(x, \leqslant n\, S.C) = \{\phi(e) \mid e \in \mathsf{ch}(\tau(x), \leqslant n\, S.C) \cap \{e_1, \dots, e_n\}\}$.

The interpretation $\mathcal{I}'$ is defined using the mapping τ:

- for each $A \in \mathbf{C}_\mathcal{K}$, $A^{\mathcal{I}'} := \{x \in \Delta^{\mathcal{I}'} \mid \tau(x) \in A^{\mathcal{I}}\}$, and
- for each $p \in \mathbf{R}_\mathcal{K}$, $p^{\mathcal{I}'} := \{(x, y) \in \Delta^{\mathcal{I}'} \times \Delta^{\mathcal{I}'} \mid (\tau(x), \tau(y)) \in p^{\mathcal{I}}\}$.

Clearly, (1) $\{\epsilon\} \cup \Delta^{\mathcal{I}'}$ is a tree and (2) $\mathit{Roots}(\mathcal{I}) = \{a^{\mathcal{I}} \mid a \in \mathbf{I}_{\mathcal{K}}\} \subseteq \mathbf{N}$ of Definition 3.7 hold. Next, m is bounded by $k_{C_{\mathcal{T}}}$ for every sequence $\phi(e_1), \ldots, \phi(e_m)$ above. Hence by the induction hypothesis, each $y \in \Delta^{\mathcal{I}'}$ is of the form $i{\cdot}x$ with $i \in \mathit{Roots}(\mathcal{I}')$ and $x \in \{1, \ldots, k_{C_{\mathcal{T}}}\}^*$, and (3) holds. Also, for each $x \in \Delta^{\mathcal{I}'}$, each new element added to $\Delta^{\mathcal{I}'}$ as a child of x is of the form $x \cdot i = \phi(e_i)$ with $e_i \in \mathsf{ch}(\tau(x), \geqslant n_i\, S_i{.}C_i) \subseteq \mathsf{neigh}_{\mathcal{I},\theta}(S_i, x)$, hence $(x, x{\cdot}i) \in (S_i)^{\mathcal{I}'}$. Since S_i is safe, this implies the existence of some atomic P such that $(x, x{\cdot}i) \in P^{\mathcal{I}'}$ as required by (4). For (5), it only remains to observe that for every pair (x, y) such that neither (a) $x = y$, nor (b) y is a successor of x, nor (c) x is the predecessor of y, the construction above ensures that $\theta'(x, y) = t_\emptyset$ is such that, for every $p \in \mathbf{R}$, $\neg p \in \theta'(x, y)$ and hence $(x, y) \notin p^{\mathcal{I}'}$. Therefore, $\mathcal{I}'$ is a $k_{C_{\mathcal{T}}}$-canonical interpretation of $\mathcal{K}$. Furthermore, as $\langle \mathcal{I}, \theta, \mathsf{ch}\rangle$ is an adorned pre-model of $\mathcal{K}$, it is readily checked that $\langle \mathcal{I}', \theta', \mathsf{ch}'\rangle$ is also an adorned pre-model of $\mathcal{K}$. Finally, if a concept $\exists R^*{.}C$ is *regenerated* from x to y in $\langle \mathcal{I}', \theta', \mathsf{ch}'\rangle$, then $\exists R^*{.}C$ is also regenerated from $\tau(x)$ to $\tau(y)$ in $\langle \mathcal{I}, \theta, \mathsf{ch}\rangle$. As a consequence, well-foundedness of $\langle \mathcal{I}, \theta, \mathsf{ch}\rangle$ implies well-foundedness of $\langle \mathcal{I}', \theta', \mathsf{ch}'\rangle$.

Finally, it is easy to observe that if $\mathcal{I}', \pi \models q$ for some π, then the composition $\pi^* = \pi \circ \tau$ would be a match for q in $\mathcal{I}$, and $I, \pi^* \models q$ would contradict the assumption that $\mathcal{I} \not\models q$. Hence $\mathcal{I}' \not\models q$. □

References

[1] S. Abiteboul, P. Buneman, and D. Suciu. *Data on the Web: from Relations to Semistructured Data and XML*. Morgan Kaufmann, 2000.

[2] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison Wesley Publ. Co., 1995.

[3] S. Abiteboul and V. Vianu. Regular path queries with constraints. *J. of Computer and System Sciences*, 58(3):428–452, 1999.

[4] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2003.

[5] S. Bechhofer, F. van Harmelen, J. Hendler, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, and L. A. Stein. OWL Web Ontology Language reference – W3C recommendation. Technical report, World Wide Web Consortium, Feb. 2004. Available at `http://www.w3.org/TR/owl-ref/`.

[6] P. Bonatti, C. Lutz, A. Murano, and M. Y. Vardi. The complexity of enriched μ-calculi. *Logical Methods in Computer Science*, 4(3:11):1–27, 2008.

[7] P. Buneman. Semistructured data. In *Proc. of the 16th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS'97)*, pages 117–121, 1997.

[8] D. Calvanese, G. De Giacomo, and M. Lenzerini. On the decidability of query containment under constraints. In *Proc. of the 17th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS'98)*, pages 149–158, 1998.

[9] D. Calvanese, G. De Giacomo, and M. Lenzerini. Reasoning in expressive description logics with fixpoints based on automata on infinite trees. In *Proc. of the 16th Int. Joint Conf. on Artificial Intelligence (IJCAI'99)*, pages 84–89, 1999.

[10] D. Calvanese, G. De Giacomo, and M. Lenzerini. 2ATAs make DLs easy. In *Proc. of the 15th Int. Workshop on Description Logic (DL 2002)*, pages 107–118. CEUR Electronic Workshop Proceedings, `http://ceur-ws.org/Vol-53/`, 2002.

[11] D. Calvanese, G. De Giacomo, and M. Lenzerini. Conjunctive query containment and answering under description logics constraints. *ACM Trans. on Computational Logic*, 9(3):22.1–22.31, 2008.

[12] D. Calvanese, G. De Giacomo, M. Lenzerini, and M. Y. Vardi. Rewriting regular expressions in semi-structured data. In *Proc. of ICDT'99 Workshop on Query Processing for Semi-Structured Data and Non-Standard Data Formats*, 1999.

[13] D. Calvanese, G. De Giacomo, M. Lenzerini, and M. Y. Vardi. Containment of conjunctive regular path queries with inverse. In *Proc. of the 7th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2000)*, pages 176–185, 2000.

[14] D. Calvanese, G. De Giacomo, M. Lenzerini, and M. Y. Vardi. Rewriting of regular expressions and regular path queries. *J. of Computer and System Sciences*, 64(3):443–465, 2002.

[15] D. Calvanese, G. De Giacomo, M. Lenzerini, and M. Y. Vardi. Reasoning on regular path queries. *SIGMOD Record*, 32(4):83–92, 2003.

[16] 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.

[17] D. Calvanese, T. Eiter, and M. Ortiz. Regular path queries in expressive description logics with nominals. In C. Boutilier, editor, *Proc. of the 21st Int. Joint Conf. on Artificial Intelligence (IJCAI 2009)*, pages 714–720, 2009.

[18] D. Calvanese, G. D. Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Tractable reasoning and efficient query answering in description logics: The *l-lite* family. *J. of Automated Reasoning*, 39(3):385–429, 2007.

[19] A. K. Chandra and P. M. Merlin. Optimal implementation of conjunctive queries in relational data bases. In *Proc. of the 9th ACM Symp. on Theory of Computing (STOC'77)*, pages 77–90, 1977.

[20] B. Cuenca Grau, I. Horrocks, B. Motik, B. Parsia, , and P. P.-S. U. Sattler. OWL 2: The next step for OWL. *Journal of Web Semantics*, 6(4):309–322, 2008.

[21] A. Deutsch and V. Tannen. Optimization properties for classes of conjunctive regular path queries. In G. Ghelli and G. Grahne, editors, *Proc. of the 8th Int. Workshop on Database Programming Languages (DBPL 2001)*, volume 2397 of *Lecture Notes in Computer Science*, pages 21–39. Springer, 2001.

[22] T. Eiter, C. Lutz, M. Ortiz, and M. Simkus. Query answering in description logics: The knots approach. In H. Ono, M. Kanazawa, and R. J. G. B. de Queiroz, editors, *Logic, Language, Information and Computation, 16th International Workshop, WoLLIC 2009*, volume 5514 of *Lecture Notes in Computer Science*, pages 26–36. Springer, 2009.

[23] T. Eiter, C. Lutz, M. Ortiz, and M. Šimkus. Query answering in description logics with transitive roles. In C. Boutilier, editor, *Proc. of the 21st Int. Joint Conf. on Artificial Intelligence (IJCAI 2009)*, pages 759–764, 2009.

[24] E. A. Emerson and C. S. Jutla. Tree automata, mu-calculus and determinacy. In *Proc. of the 32nd Annual Symp. on the Foundations of Computer Science (FOCS'91)*, pages 368–377, 1991.

[25] M. J. Fischer and R. E. Ladner. Propositional dynamic logic of regular programs. *J. of Computer and System Sciences*, 18:194–211, 1979.

[26] D. Florescu, A. Levy, and D. Suciu. Query containment for conjunctive queries with regular expressions. In *Proc. of the 17th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS'98)*, pages 139–148, 1998.

[27] B. Glimm, I. Horrocks, C. Lutz, and U. Sattler. Conjunctive query answering for the description logic $\mathcal{SHIQ}$. In *Proc. of the 20th Int. Joint Conf. on Artificial Intelligence (IJCAI 2007)*, pages 399–404, 2007.

[28] B. Glimm, I. Horrocks, C. Lutz, and U. Sattler. Conjunctive query answering for the description logic $\mathcal{SHIQ}$. *J. of Artificial Intelligence Research*, 31:151–198, 2008.

[29] B. Glimm and S. Rudolph. Conjunctive query entailment: Decidable in spite of O, I, and Q. In *Proc. of the 22nd Int. Workshop on Description Logic (DL 2009)*, 2009.

[30] G. Grahne and A. Thomo. Query containment and rewriting using views for regular path queries under constraints. In *Proc. of the 22nd ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS 2003)*, pages 111–122, 2003.

[31] I. Horrocks, O. Kutz, and U. Sattler. The irresistible $\mathcal{SRIQ}$. In *Proc. of the 1st Int. Workshop on OWL: Experiences and Directions (OWLED 2005)*, 2005.

[32] I. Horrocks, O. Kutz, and U. Sattler. The even more irresistible $\mathcal{SROIQ}$. In P. Doherty, J. Mylopoulos, and C. A. Welty, editors, *Proc. of the 10th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2006)*, pages 57–67. AAAI Press, 2006.

[33] I. Horrocks and U. Sattler. Decidability of shiq with complex role inclusion axioms. *Artif. Intell.*, 160(1):79–104, 2004.

[34] I. Horrocks, U. Sattler, and S. Tobies. Reasoning with individuals for the description logic $\mathcal{SHIQ}$. In D. McAllester, editor, *Proc. of the 17th Int. Conf. on Automated Deduction (CADE 2000)*, volume 1831 of *Lecture Notes in Computer Science*, pages 482–496. Springer, 2000.

[35] I. Horrocks and S. Tessaris. A conjunctive query language for description logic ABoxes. In *Proc. of the 17th Nat. Conf. on Artificial Intelligence (AAAI 2000)*, pages 399–404, 2000.

[36] U. Hustadt, B. Motik, and U. Sattler. A decomposition rule for decision procedures by resolution-based calculi. In *Proc. of the 11th Int. Conf. on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR 2004)*, pages 21–35, 2004.

[37] U. Hustadt, B. Motik, and U. Sattler. Data complexity of reasoning in very expressive description logics. In *Proc. of the 19th Int. Joint Conf. on Artificial Intelligence (IJCAI 2005)*, pages 466–471, 2005.

[38] Y. Kazakov. $\mathcal{RIQ}$ and $\mathcal{SROIQ}$ are harder than $\mathcal{SHOIQ}$. In *Proc. of the 11th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2008)*, pages 274–284, 2008.

[39] A. Krisnadhi and C. Lutz. Data complexity in the $\mathcal{EL}$ family of description logics. In *Proc. of the 20th Int. Workshop on Description Logic (DL 2007)*, volume 250 of *CEUR Electronic Workshop Proceedings,* `http://ceur-ws.org/Vol-250/`, 2007.

[40] M. Krötzsch, S. Rudolph, and P. Hitzler. Conjunctive queries for a tractable fragment of OWL 1.1. In a. Karl Aberer et. editor, *The Semantic Web, 6th Int. Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007*, volume 4825 of *Lecture Notes in Computer Science*, pages 310–323. Springer, 2007.

[41] O. Kupferman, U. Sattler, and M. Y. Vardi. The complexity of the graded μ-calculus. In A. Voronkov, editor, *Proc. of the 18th Int. Conf. on Automated Deduction (CADE 2002)*, volume 2392 of *Lecture Notes in Computer Science*, pages 423–437. Springer, 2002.

[42] O. Kupferman and M. Y. Vardi. Weak alternating automata and tree automata emptiness. In *Proc. of the 30th ACM SIGACT Symp. on Theory of Computing (STOC'98)*, pages 224–233. ACM Press, 1998.

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

[44] C. Lutz. Inverse roles make conjunctive queries hard. In *Proc. of the 20th Int. Workshop on Description Logic (DL 2007)*, volume 250 of *CEUR Electronic Workshop Proceedings,* `http://ceur-ws.org/Vol-250/`, pages 100–111, 2007.

[45] D. E. Muller and P. E. Schupp. Alternating automata on infinite trees. *Theoretical Computer Science*, 54:267–276, 1987.

[46] D. E. Muller and P. E. Schupp. Simulating alternating tree automata by nondeterministic automata: new results and new proofs of the theorems of Rabin, McNaughton and Safra. *Theoretical Computer Science*, 141(1-2):69–107, 1995.

[47] M. Ortiz. An automata-based algorithm for description logics around $\mathcal{SRIQ}$. In *Proc. of LANMR 2008*, volume 408 of *CEUR Electronic Workshop Proceedings,* `http://ceur-ws.org/Vol-408/`, 2008.

[48] 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.

[49] R. Rosati. On conjunctive query answering in $\mathcal{EL}$. In *Proc. of the 20th Int. Workshop on Description Logic (DL 2007)*, volume 250 of *CEUR Electronic Workshop Proceedings,* `http://ceur-ws.org/Vol-250/`, 2007.

[50] S. Rudolph, M. Krötzsch, and P. Hitzler. Cheap Boolean role constructors for description logics. In *Proc. of the 11th Eur. Conference on Logics in Artificial Intelligence (JELIA 2008)*, volume 5293 of *Lecture Notes in Computer Science*, pages 362–374, 2008.

[51] K. Schild. A correspondence theory for terminological logics: Preliminary report. In *Proc. of the 12th Int. Joint Conf. on Artificial Intelligence (IJCAI'91)*, pages 466–471, 1991.

[52] R. S. Streett and E. A. Emerson. An automata theoretic decision procedure for the propositional μ-calculus. *Information and Computation*, 81:249–264, 1989.

[53] W. Thomas. Automata on infinite objects. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, chapter 4, pages 133–192. Elsevier Science Publishers, 1990.

[54] S. Tobies. *Complexity Results and Practical Algorithms for Logics in Knowledge Representation*. PhD thesis, LuFG Theoretical Computer Science, RWTH-Aachen, Germany, 2001.

[55] S. Tobies. PSPACE reasoning for graded modal logics. *J. of Logic and Computation*, 11(1):85–106, 2001.

[56] M. Y. Vardi. The taming of converse: Reasoning about two-way computations. In R. Parikh, editor, *Proc. of the 4th Workshop on Logics of Programs*, volume 193 of *Lecture Notes in Computer Science*, pages 413–424. Springer, 1985.

[57] M. Y. Vardi. Reasoning about the past with two-way automata. In *Proc. of the 25th Int. Coll. on Automata, Languages and Programming (ICALP'98)*, volume 1443 of *Lecture Notes in Computer Science*, pages 628–641. Springer, 1998.

[58] M. Y. Vardi and P. Wolper. Automata-theoretic techniques for modal logics of programs. *J. of Computer and System Sciences*, 32:183–221, 1986.