Information and Computation 237 (2014) 12–55

Contents lists available at ScienceDirect

Information and Computation

www.elsevier.com/locate/yinco

Answering regular path queries in expressive Description Logics via alternating tree-automata[☆]

Diego Calvanese [a], Thomas Eiter [b], Magdalena Ortiz [b,*]

[a] KRDB Research Centre, Free University of Bozen-Bolzano, Via della Mostra 4, I-39100 Bolzano, Italy
[b] Institute of Information Systems, Vienna University of Technology, Favoritenstraße 9-11, A-1040 Vienna, Austria

ARTICLE INFO

Article history:
Received 23 December 2009
Received in revised form 3 April 2013
Available online 18 April 2014

Keywords:
Expressive Description Logics
Query answering
Computational complexity
Automata on infinite trees

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 $\exists S.\mathsf{Self}$ 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 (under unary number encoding). 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 blow-up 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.

© 2014 Elsevier Inc. All rights reserved.

1. Introduction

Description Logics (DLs) [2] is 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). It has 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 [3] is based on a DL known as $\mathcal{SROIQ}$ [4], whose fragment $\mathcal{SRIQ}$ [5] extends the DL $\mathcal{SHIQ}$ underlying OWL-Lite [6].

☆ Some results of this paper have appeared, in preliminary form, in a conference paper in the *Proc. of AAAI 2007* [1].
* Corresponding author.
E-mail addresses: calvanese@inf.unibz.it (D. Calvanese), eiter@kr.tuwien.ac.at (T. Eiter), ortiz@kr.tuwien.ac.at (M. Ortiz).

http://dx.doi.org/10.1016/j.ic.2014.04.002
0890-5401/© 2014 Elsevier Inc. All rights reserved.

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. [7], many further works have addressed the problem of evaluating complex queries over DL knowledge bases. Special attention has been devoted to (*unions of*) *conjunctive queries* (CQs and UCQs) [8], which are the formal counterpart of the most widely used fragments of SQL (or relational algebra) queries, namely (unions of) select-project-join queries. (U)CQs over DL knowledge bases have been studied for many DLs, ranging from weak ones that allow for efficient algorithms, like those of the $\mathcal{EL}$ [9–11] and DL-Lite families [12], to the very expressive ones of the $\mathcal{ALCH}$ and $\mathcal{SH}$ families, cf. [13–16].

Another important language for querying knowledge bases is that of *regular path queries* (RPQs) [17–19], which allow one to ask for pairs of objects that are connected by a path conforming to a regular expression. Due to their capability of expressing complex navigations in graphs, RPQs are the fundamental mechanism for querying semi-structured and graph-structured data. Indeed, as a query language, RPQs go beyond first-order logic, since they allow one to express a *controlled* form of recursion. This turns out to be essential for querying graph-like structures such as those encountered in several domains that are gaining increasing importance, notably social networks [20] and linked open data [21]. Notice that, in a setting of incomplete information as the one encoded by means of a knowledge base, the use of unrestricted recursion would quickly lead to undecidability, not only of intensional inference tasks such as query containment or equivalence [22], but also of query answering [23,24]. Instead, the restricted form of recursion provided by RPQs and their extensions considered here, provides a good trade-off between the ability to flexibly traverse data whose precise structure is not defined a priori (e.g., in terms of a relational schema), and the decidability of query answering also in the presence of complex domain knowledge encoded in DLs. The complex paths allowed in the query allow one to find in the data complex relations between items, without being constrained by the relations explicitly stated in the data or pre-defined in the ontology, and without having to modify the ontology solely for query answering. Moreover, when also *inverse roles* are allowed to occur in the regular expression, the complex relations expressed by the resulting *two-way RPQs* (2RPQs) are not restricted to the direction initially chosen by the designer to represent relations between data items. 2RPQs are for example present in the property paths in SPARQL 1.1 [25], the new standard RDF query language, and in the XML query language XPath [26]. We consider here 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 of CQs, are also a natural generalization of several extensions of RPQs that have been studied by different authors, e.g., [27,28,19,29–32]. They are, to our knowledge, the most expressive query language considered so far over DL knowledge bases [33,1].

In this paper, we describe a technique, first presented in [1], 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 [34]) with regular role expressions [35], Boolean role inclusion axioms, and concepts of the form $\exists S.\mathsf{Self}$ [5]. 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 [3].

Previously, algorithms for query answering over expressive DLs had used a variety of techniques, ranging from query rewriting [7,13,36], over modified tableaux techniques [16], to resolution [37]. We obtain our results by exploiting techniques based on *automata on infinite trees* [38], which have been developed initially in the context of modal logics and program logics [39–43]. While the application of automata techniques in DLs is not novel, cf. [35,44,45], 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 [35,44] and incorporate the ABox part. Then, to decide query entailment over DL knowledge bases, we build on the ideas of Calvanese et al. [30], 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 [33].

In this paper, we make the following contributions (all complexity results hold under unary number encoding):

- As a stepping stone to our main results, we first present an automata-based algorithm for checking the satisfiability of a $\mathcal{ZIQ}$ 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. By the aforementioned reduction [34], the same bound holds for $\mathcal{SHIQ}$. From known hardness results for answering CQs over $\mathcal{ALCI}$ [46] and $\mathcal{SH}$ [47] KBs, it follows that this is worst case optimal. In fact, a simple adaptation of the proof in [47] shows that the matching lower bounds hold even for answering *positive queries* (which do not allow regular expressions over roles in atoms) and *conjunctive RPQs* (i.e., P2RPQs that use only conjunction and no inverse roles) over plain $\mathcal{ALC}$ KBs. This shows that, once either inverse roles or role hierarchies and transitivity are allowed in the KB, or alternatively, regular expressions or disjunctions are allowed in the query, 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 $\mathcal{SRIQ}$ knowledge base into a $\mathcal{ZIQ}$ knowledge base. 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 $\mathcal{SROIQ}$), decidability of query entailment remains open.

The rest of this article 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 Appendix A.

2. Preliminaries

In this section, we define the main Description Logic (DL) and the query answering problem considered in this article. 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 any 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 $\mathcal{ALCQIb}$ [34] with regular role expressions, Boolean role inclusion axioms, and concepts of the form $\exists S.\mathsf{Self}$ in the style of $\mathcal{SRIQ}$ [5]. 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 [35,1].

Definition 2.1 *(Concepts and roles).* We consider fixed, countably infinite and pairwise disjoint 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* (or *universal*) *role* T and the *bottom* (or *empty*) *role* B. *Concepts* C, C' and *roles* P, S, S', R, R', are formed according to the following syntax, where $A \in \mathbf{C}$ and $p \in \mathbf{R} \setminus \{\mathsf{T}\}$[1]:

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

$$P \longrightarrow p \mid p^-$$

$$S, S' \longrightarrow P \mid S \cap S' \mid S \cup S' \mid S \setminus S'$$

$$R, R' \longrightarrow \mathsf{T} \mid S \mid R \cup R' \mid R \circ R' \mid R^* \mid id(C)$$

We call roles P *atomic*, and roles S, S' *simple*. Note that, as $p \neq \mathsf{T}$, the top role T may occur in arbitrary roles R, R' but not in simple roles S, S'. A $\mathcal{ZIQ}$ *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.

[1] We omit parentheses in expressions following the usual conventions.

A *knowledge base* (*KB*) is a pair $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, where $\mathcal{T}$ is a TBox and $\mathcal{A}$ is a non-empty ABox.[2] 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 $\bar{\mathbf{R}}_\mathcal{K} = \mathbf{R}_\mathcal{K} \cup \{p^- \mid p \in \mathbf{R}_\mathcal{K}\}$. □

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 $p^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$, such that:

$$\top^\mathcal{I} = \Delta^\mathcal{I}, \qquad \mathsf{T}^\mathcal{I} = \Delta^\mathcal{I} \times \Delta^\mathcal{I}$$
$$\bot^\mathcal{I} = \emptyset, \qquad \mathsf{B}^\mathcal{I} = \emptyset.$$

The function $\cdot^\mathcal{I}$ is inductively extended to all concepts and roles as follows:

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

where $\circ$ denotes composition of binary relations 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 $\mathcal{A}$;
- 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 $\mathcal{I} \models \mathcal{K}$. □

We remark that we do not make the *unique name assumption*, which can be simulated using assertions of the form $a \not\approx b$. Note also that we do not include in the language equality assertions $a \approx b$, since their addition would not provide additional expressiveness. Indeed, they could be easily compiled away by replacing all individuals in the same equality equivalence class with one representative.

2.2. Query answering

We next introduce P2RPQs, which generalize conjunctive regular path queries [32,30] 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* $\mathcal{K}$. We denote by $At(q)$ the set of all atoms occurring in a P2RPQ q.

Let $q = \exists \vec{v}.\varphi(\vec{v})$ be a P2RPQ, and let $\mathbf{V}_q$ and $\mathbf{I}_q$ denote the sets of variables and individuals in q, respectively. Given an interpretation $\mathcal{I}$, let $\pi : \mathbf{V}_q \cup \mathbf{I}_q \to \Delta^\mathcal{I}$ 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 exists some 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 $\mathcal{K} \models q$. □

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

			HasParent	(Heracles, Zeus)
mortal	$\sqsubseteq$	$\neg$*deity*	*HasParent*	(Heracles, Alcmene)
$\top$	$\sqsubseteq$	*male* $\sqcup$ *female*	*HasParent*	(Alcmene, Electryon)
male	$\equiv$	$\neg$*female*	*HasParent*	(Electryon, Perseus)
$\top$	$\sqsubseteq$	$\exists$*HasFather*.*male* $\sqcap$ $\exists$*HasMother*.*female*	*HasParent*	(Perseus, Zeus)
HasParent	$\equiv$	*HasMother* $\cup$ *HasFather*	*male*	(Zeus)
$\forall$*HasParent*.*mortal*	$\sqsubseteq$	*mortal*	*female*	(Alcmene)
deity	$\sqsubseteq$	$\forall$*HasParent**.*deity*	*deity*	(Zeus)
			mortal	(Alcmene)

Fig. 1. The genealogy KB $\mathcal{K}_g$ used in Example 2.5.

P2RPQs are a generalization of *conjunctive queries* (*CQs*), a well known query language widely studied in databases [48,8] and, more recently, in DLs [24,49,16,13]. A CQ is a P2RPQ in which neither disjunction ($\vee$) nor regular role expressions occur. We observe that the possibility of using regular role expressions in the query atoms of P2RPQs significantly increases the expressive power of the query language, since it allows one to express complex navigations in the models of the given KB, similar to those possible with (conjunctive) regular path queries studied in graph databases [32,30,19,27].

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 Fig. 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 $\mathcal{K}_g$:

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

Informally, q_g asks whether there are a male and female individual (represented by v_1 and v_2, resp.) who are relatives (i.e., related by the expression *HasParent** $\circ$ *HasParent*$^{-*}$), are not both deities, and have a common child v_3. Note that $\mathcal{K}_g \models q_g$, as $\pi(v_1) = \mathrm{Zeus}^{\mathcal{I}}$, $\pi(v_2) = \mathrm{Alcmene}^{\mathcal{I}}$ and $\pi(v_3) = \mathrm{Heracles}^{\mathcal{I}}$ is a match for q_g in every model $\mathcal{I}$ of $\mathcal{K}_g$. $\square$

Note that we have restricted our 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 [41].

Definition 2.6. An (*infinite*) *tree* is a prefix-closed set $T \subseteq \mathbb{N}^*$ of words over the natural numbers $\mathbb{N}$. If $T \subseteq \{1, \ldots, k\}^*$ for some $k \geqslant 0$, we call it a *k-tree*. The elements of T are called *nodes*, the empty word ε is its *root*. For every $x \in T$, the nodes $x{\cdot}c$ with $c \in \mathbb{N}$ 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 $(x{\cdot}c){\cdot}{-1} = x$. We call T *k-ary* if it is a k-tree and each node in it has k successors (i.e., $T = \{1, \ldots, k\}^*$). 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. $\square$

2.3.1. Two-way alternating tree automata (2ATAs)

Now we define *two-way alternating tree automata* (2ATAs) over infinite trees as introduced in [41], 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 [50]. 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 of propositional atoms, 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 $\mathbf{A} = \langle k, \Sigma, Q, \delta, s_0, F \rangle$, where:

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

- 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 $[k] = \{-1, 0, 1, \dots, k\}$, is the transition function;
- $s_0 \in \mathrm{Q}$ is the initial state; and
- $F = (G_1, \dots, G_n)$ with $G_1 \subseteq G_2 \subseteq \dots \subseteq G_n = \mathrm{Q}$ is a (*parity*) *acceptance condition*, whose length n, denoted by $\mathsf{ind}(\mathbf{A})$, is called the *index* of $\mathbf{A}$.

We refer to each component Σ, Q, etc. of $\mathbf{A}$ by $\Sigma(\mathbf{A})$, $Q(\mathbf{A})$, etc., respectively. $\square$

The transition function δ maps a state $s \in \mathrm{Q}$ and an input letter $\sigma \in \Sigma$ to a positive Boolean formula $\delta(s, \sigma) = \varphi$ over the atoms in $[k] \times \mathrm{Q}$. Intuitively, each atom (c, s') in φ corresponds to a new copy of the automaton going in the direction given by c and starting in state s'. For example, let $k = 2$ and $\delta(s_1, \sigma) = (1, s_2) \wedge (1, s_3) \vee (-1, s_1) \wedge (0, s_3)$. If $\mathbf{A}$ is in the state s_1 and reads a node x labeled with σ, it proceeds by sending off either (i) two copies, in the states s_2 and s_3 respectively, to the first successor of x (i.e., $x{\cdot}1$), or (ii) one copy in the state s_1 to the predecessor of x (i.e., $x{\cdot}{-1}$) and one copy in the state s_3 to x itself (i.e., $x{\cdot}0$). For convenience, we may specify the transition function δ only partially, and assume that $\delta(s, \sigma) = \mathsf{false}$ if not specified otherwise.

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 $r(n) = (x, s) \in T \times \mathrm{Q}$ and describes a copy of $\mathbf{A}$ that is in the state s and reads the node x of T; the labels of adjacent nodes must satisfy the transition function of $\mathbf{A}$. Formally, we define the following generalized notion of a run, called (x, s)-run:

Definition 2.8. Let $\mathbf{A} = \langle k, \Sigma, \mathrm{Q}, \delta, s_0, F\rangle$ be a 2ATA and let (T, L) be a Σ-labeled k-ary tree. Moreover, let $x \in T$ and $s \in \mathrm{Q}$. An (x, s)-*run* of $\mathbf{A}$ over (T, L) is a $(T \times \mathrm{Q})$-labeled tree (T_r, r) satisfying:

(R1) $\varepsilon \in T_r$ and $r(\varepsilon) = (x, s)$,
(R2) Each $w \in T_r$ *satisfies* δ, i.e., if $r(w) = (y, s')$ and $\delta(s', L(y)) = \varphi$, then there is a (possibly empty) set $W = \{(c_1, s_1), \dots, (c_n, s_n)\} \subseteq [k] \times \mathrm{Q}$ such that:
- W satisfies φ, and
- for every $1 \le i \le n$, it holds that $w{\cdot}i \in T_r$, $y{\cdot}c_i$ is defined, and $r(w{\cdot}i) = (y{\cdot}c_i, s_i)$.

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

We call an (ε, s_0)-run a (*full*) *run*. A 2ATA $\mathbf{A}$ *accepts* a Σ-labeled tree $\mathbf{T}$, if there exists a (full) 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 $\mathbf{A}$. $\square$

The following result is well-known.

Theorem 2.9. *(See [41].) Nonemptiness of a given 2ATA $\mathbf{A}$ is decidable in time single exponential in $|\mathrm{Q}(\mathbf{A})|$ and polynomial in $|\Sigma(\mathbf{A})|$.*

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

Proposition 2.10. *Given 2ATAs $\mathbf{A}_1, \dots, \mathbf{A}_n$, it is possible to construct a 2ATA $\mathbf{A}$ with $|\mathrm{Q}(\mathbf{A})| = \sum_{i=1}^{n} |\mathrm{Q}(\mathbf{A}_i)| + 1$ and $\mathsf{ind}(\mathbf{A}) = \max_{i=1}^{n} \mathsf{ind}(\mathbf{A}_i)$ such that $\mathcal{L}(\mathbf{A}) = \bigcap_{i=1}^{n} \mathcal{L}(\mathbf{A}_i)$.*

Proof (Sketch). Assuming the state sets $Q(\mathbf{A}_i)$ are pairwise disjoint, introduce a new state s_0^* and let $Q(\mathbf{A})$ be the union of all $Q(\mathbf{A}_i)$ plus s_0^*, which is the initial state of $\mathbf{A}$. The transition function $\delta(\mathbf{A})$ consists of the union of all $\delta(\mathbf{A}_i)$ and has $\delta(s_0^*, \sigma) = \bigwedge_{i=1}^{n} \delta(s_0(\mathbf{A}_i), \sigma)$ for all $\sigma \in \Sigma$. For each $\mathbf{A}_i$, let $F(\mathbf{A}_i) = (G_1^i, \dots, G_{m_i}^i)$ and let $m = \max_{i=1}^{n} m_i = \max_{i=1}^{n} \mathsf{ind}(\mathbf{A}_i)$. To obtain $F(\mathbf{A})$, we first 'fill up' all $F(\mathbf{A}_i)$ to $F'(\mathbf{A}_i)$ with index m, by setting $G_{m_i+1}^i = G_{m_i+2}^i = \dots = G_m^i = Q(\mathbf{A}_i)$. Then $F(\mathbf{A})$ consists of the component-wise union of all the $F'(\mathbf{A}_i)$. $\square$

Automata on infinite trees provide elegant solutions for decision problems in temporal and program logics [51], which has been widely exploited for providing optimal complexity bounds for the satisfiability problem of many variants of PDL, the μ-calculus, and similar logics [41,40]. They have also been explored in DLs, but mostly for deciding *concept satisfiability* [45,35], given that in many DLs, concepts have tree-shaped models.

2.3.2. (One-way) non-deterministic tree automata (1NTAs)

Standard non-deterministic (one-way) automata on infinite trees can be defined as particular 2ATAs that always move to the k successors of the current node and switch to states that are given by a tuple of k states, one for each successor. This can be expressed as a formula in disjunctive form:

Definition 2.11. A *one-way non-deterministic automaton* (1NTA) is a 2ATA $\mathbf{A} = \langle k, \Sigma, Q, \delta, s_0, F\rangle$ such that for every $s \in Q$ and every $\sigma \in \Sigma$, $\delta(s, \sigma)$ is of the form $((1, s_1^1) \wedge \cdots \wedge (k, s_k^1)) \vee \cdots \vee ((1, s_1^\ell) \wedge \cdots \wedge (k, s_k^\ell))$, with $\ell \geqslant 0$, and $s_j^i \in Q$ for each $1 \leqslant i \leqslant \ell$ and each $1 \leqslant \ell \leqslant k$. □

We recall some results on 1NTAs.

Proposition 2.12. *(See [41].) Given a 2ATA* $\mathbf{A}$*, it is possible to construct a 1NTA* $\mathbf{A}_1$*, with* $|Q(\mathbf{A}_1)| \leqslant 2^{O(|Q(\mathbf{A})|^c)}$ *for some constant c, and* $\mathsf{ind}(\mathbf{A}_1) = O(\mathsf{ind}(\mathbf{A}))$*, such that* $\mathcal{L}(\mathbf{A}_1) = \mathcal{L}(\mathbf{A})$*.*

The following bounds for automata complementation are given in [52].

Proposition 2.13 *(Complementation). For every 1NTA* $\mathbf{A}$ *running over k-ary trees, it is possible to construct a 1NTA* $\mathbf{A}'$ *that accepts a k-ary* Σ*-labeled tree* (T, L) *iff* $(T, L) \notin \mathcal{L}(\mathbf{A})$*, and such that* $|Q(\mathbf{A}')| \leqslant 2^{O(f(\mathbf{A}))}$ *and* $\mathsf{ind}(\mathbf{A}') = O(f(\mathbf{A}))$*, where* $f(\mathbf{A}) = \mathsf{ind}(\mathbf{A})\cdot|Q(\mathbf{A})|\cdot\log|Q(\mathbf{A})|$*.*

Another automata theoretic operation we exploit is *projection*, which restricts the trees in the language of an automaton to a smaller alphabet. For $\Sigma = 2^{\Phi}$ and $\Sigma' = 2^{\Phi'}$ where $\Phi' \subseteq \Phi$, and for a 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'. The *Σ'-projection* of a set $\mathcal{L}$ of Σ-labeled trees is the set consisting of the Σ'-restrictions of all trees in $\mathcal{L}$.

Proposition 2.14 *(Projection). For* Σ *and* Σ' *as above, for every 1NTA* $\mathbf{A}$ *running over* Σ*-labeled trees, it is possible to construct a 1NTA* $\mathbf{A}^{\Sigma'}$ *with* $|Q(\mathbf{A}^{\Sigma'})| \leqslant |Q(\mathbf{A})|$ *and* $\mathsf{ind}(\mathbf{A}^{\Sigma'}) \leqslant \mathsf{ind}(\mathbf{A})$ *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 $\Xi(\sigma) = \{\sigma' \in \Sigma \mid \sigma' \cap \Phi' = \sigma\}$. □

The following bounds for the intersection of two 1NTAs are known[4]:

Proposition 2.15 *(Intersection). 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* $|Q(\mathbf{A})| \leqslant 2^{O(f(\mathbf{A}_1,\mathbf{A}_2)^2)} \cdot f(\mathbf{A}_1, \mathbf{A}_2) \cdot |Q(\mathbf{A}_1)| \cdot |Q(\mathbf{A}_2)|$*, where* $f(\mathbf{A}_1, \mathbf{A}_2) = \mathsf{ind}(\mathbf{A}_1) + \mathsf{ind}(\mathbf{A}_2) + 1$*.*

Finally, testing a 1NTA for emptiness is feasible within the following bounds.

Proposition 2.16. *(See [53].) Given a 1NTA* $\mathbf{A}$*, the nonemptiness problem is decidable in time* $O(|Q(\mathbf{A})|^{\mathsf{ind}(\mathbf{A})})$*.*

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: 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 *$\mathcal{K}$-connected* to an element $d_0 \in \Delta^{\mathcal{I}}$ in an interpretation $\mathcal{I}$, if there is some sequence $d_0, \ldots, d_n$ such that $d = d_n$ and for each $0 \leqslant i < n$ 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 *$\mathcal{K}$-connected*, if each $d \in \Delta^{\mathcal{I}}$ is $\mathcal{K}$-connected to $a^{\mathcal{I}}$ for some $a \in \mathbf{I}_{\mathcal{K}}$.

Lemma 3.1 *(Connected model property). Let* $\mathcal{K}$ *be a* $\mathcal{ZIQ}$ *KB. Then, for every P2RPQ q,* $\mathcal{K} \not\models q$ *implies that there is a* $\mathcal{K}$*-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 that are $\mathcal{K}$-connected to $a^{\mathcal{I}}$ for some $a \in \mathbf{I}_{\mathcal{K}}$; the resulting interpretation $\mathcal{I}'$ is $\mathcal{K}$-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

[4] To our knowledge, these bounds are unpublished. A tighter bound of $|Q(\mathbf{A})| = f'(\mathbf{A}_1, \mathbf{A}_2)! \cdot f'(\mathbf{A}_1, \mathbf{A}_2) \cdot |Q(\mathbf{A}_1)| \cdot |Q(\mathbf{A}_2)|$, where $f'(\mathbf{A}_1, \mathbf{A}_2) = (\mathsf{ind}(\mathbf{A}_1) + \mathsf{ind}(\mathbf{A}_2))/2 + 1$, and $\mathsf{ind}(\mathbf{A}) = \mathsf{ind}(\mathbf{A}_1) + \mathsf{ind}(\mathbf{A}_2)$ was confirmed through personal communication with Yoad Lustig and Nir Piterman, to whom we are very grateful.

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 every 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 occurs only at the atomic level, and $\top$, $\bot$, B, T do not occur.

1. *ABox reduction*. We transform $\mathcal{A}$ into an *extensionally reduced ABox*, i.e., an ABox in which only concept and role names are used:
   - We replace each assertion $C(a)$, where C is not a concept name, by $A_C(a)$ for a fresh $A_C \in \mathbf{C}$, and we add to $\mathcal{T}$ the CIA $A_C \sqsubseteq C$.
   - We replace each assertion $S(a,b)$, where S is not a role name, by $p_S(a,b)$ for a fresh $p_S \in \mathbf{R}$, and we add to $\mathcal{T}$ the BRIA $p_S \sqsubseteq S$.
2. *Elimination of* T. Assume that $a_1, \dots, a_n$ are the individuals in $\mathbf{I}_{\mathcal{K}}$. Using a fresh role name p_U, we add to $\mathcal{A}$ assertions $p_U(a_i, a_{i+1})$, for all $1 \leqslant i < n$. We add to $\mathcal{T}$ the BRIA $\bigcup_{R \in \bar{\mathbf{R}}_{\mathcal{K}}} R \sqsubseteq p_U$, and we replace in $\mathcal{K}$ each occurrence of T by the role $p_U{}^*$.
3. *Elimination of* $\top$, $\bot$ *and* B. We replace in $\mathcal{K}$ the empty role B is by a fresh role name p_{B}, and we add to $\mathcal{T}$ the CIA $\top \sqsubseteq \forall p_{\mathsf{B}}.\bot$. We replace the concepts $\top$ and $\bot$ respectively by fresh concept names $A_\top$ and $A_\bot$, and we add to $\mathcal{T}$ the CIAs $A \sqcup \neg A \sqsubseteq A_\top$ and $A_\top \sqsubseteq \neg A_\bot$, for an arbitrary concept name A.
4. *BRIA elimination*. We replace each BRIA $S \sqsubseteq S'$ in $\mathcal{T}$ by the CIA $\exists(S \setminus S').\top \sqsubseteq \bot$ (cf. [54]).
5. *Negation normal form*. Finally, we transform each concept and role occurring in $\mathcal{K}$ into *negation normal form* (NNF), i.e., where negation is pushed inwards until $\neg$ is applied only to atomic concepts, and $\setminus$ only to atomic roles. For this, we first replace each concept of the form $\exists S.\mathsf{Self}$ by a fresh concept name $A_{\exists S.\mathsf{Self}}$, and add to $\mathcal{T}$ the CIAs $A_{\exists S.\mathsf{Self}} \sqsubseteq \exists S.\mathsf{Self}$ and $\exists S.\mathsf{Self} \sqsubseteq A_{\exists S.\mathsf{Self}}$. We also replace each $S \setminus S'$ by $S \cap \neg S'$. Then we move $\neg$ inwards over concepts and roles using the well-known equivalences:
$$\begin{array}{llll} \neg(C \sqcap C') = \neg C \sqcup \neg C', & \neg(\leqslant n S.C) = \geqslant (n+1) S.C, & \neg(\forall R.C) = \exists R.\neg C, & \neg(S \cup S') = \neg S \cap \neg S' \\ \neg(C \sqcup C') = \neg C \sqcap \neg C', & \neg(\geqslant n S.C) = \leqslant (n-1) S.C, & \neg(\exists R.C) = \forall R.\neg C, & \neg(S \cap S') = \neg S \cup \neg S' \end{array}$$
We apply these equivalences until $C \in \mathbf{C}$ for every concept of the form $\neg C$, and $R \in \mathbf{R} \cup \{p^- \mid p \in \mathbf{R}\}$ for every role of the form $\neg R$. Afterwards we convert $\neg$ back to $\setminus$, replacing where necessary $\neg S$ with $p_U \setminus S$, where p_U is the role introduced in item 2.

It will be convenient to assume in what follows that also queries are normalized in such a way that they do not contain $\top$, $\bot$, T, or B, and they are in NNF. Given a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, which has already been normalized, and a query q, we normalize q with respect to $\mathcal{T}$ as follows. We replace:

- each occurrence of $\top$ by the concept $A_\top$ (introduced in step 3 of the normalization of $\mathcal{K}$),
- each occurrence of $\bot$ by the concept $A_\bot$ (introduced in step 3 of the normalization of $\mathcal{K}$),
- each occurrence of T by the role $p_U{}^*$ (introduced in step 2 of the normalization of $\mathcal{K}$), and
- each occurrence of B by the role p_{B} (introduced in step 3 of the normalization of $\mathcal{K}$).

Finally, all concepts and roles in q are rewritten into NNF as explained above.

Definition 3.2 *(Normalized knowledge base, normalized query).* A $\mathcal{ZIQ}$ knowledge base $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is *normalized*, if (i) $\mathcal{A}$ is extensionally reduced, (ii) $\mathcal{T}$ contains only CIAs, (iii) $\top$, $\bot$, T, and B do not occur in $\mathcal{K}$, and (iv) all concepts and roles in $\mathcal{K}$ are in NNF. A P2RPQ q is *normalized*, if $\top$, $\bot$, T, and B do not occur in q and all concepts and roles in q are in NNF. □

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

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

Proof. We normalize $\mathcal{K}$ using steps 1–5 above, and then normalize q with respect to $\mathcal{K}$ (using the same role names to simulate $\top$, $\bot$, T and B). A simple inspection of the transformations 1 to 5 suffices to see that the properties enforced by the preceding transformations are always preserved. Hence it is easy to see that their application results in a normalized KB $\mathcal{K}'$. Similarly for the query q'.

The normalization also preserves satisfiability and query entailment. In fact, step 4 results in a logically equivalent KB, while steps 1, 3, and 5 result in a *model conservative extension* of the original KB. Indeed, they introduce fresh concept and

Table 1
Syntactic closure $Cl(D)$ of an $\mathcal{ALCQIB}^{\mathsf{Self}}_{reg}$ concept D ($\gtrless \in \{\geqslant, \leqslant\}$, $\mathsf{Q} \in \{\forall, \exists\}$, $\boxdot \in \{\sqcup, \sqcap\}$, $\bigcirc \in \{\cap, \cup\}$).

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

role names, but every interpretation $\mathcal{I}$ can be extended to an interpretation $\mathcal{I}'$ by interpreting these names as $A_C^{\mathcal{I}'} = \{a^{\mathcal{I}} \mid a \in \mathbf{I}_{\mathcal{K}}, a^{\mathcal{I}} \in C^{\mathcal{I}}\}$, $p_S^{\mathcal{I}'} = \{(a^{\mathcal{I}}, b^{\mathcal{I}}) \mid a, b \in \mathbf{I}_{\mathcal{K}}, (a^{\mathcal{I}}, b^{\mathcal{I}}) \in S^{\mathcal{I}}\}$, $p_{\mathsf{B}}^{\mathcal{I}'} = \emptyset$, $A_{\top}^{\mathcal{I}'} = \Delta^{\mathcal{I}}$, $A_{\bot}^{\mathcal{I}'} = \emptyset$, and $A_{\exists S.\mathsf{Self}}^{\mathcal{I}'} = (\exists S.\mathsf{Self})^{\mathcal{I}}$. In this way we ensure that $\mathcal{I}' \models \mathcal{K}'$ iff $\mathcal{I} \models \mathcal{K}$, and $\mathcal{I}$ and $\mathcal{I}'$ coincide on all matches for q. If we use the same $A_{\top}$, $A_{\bot}$ and p_{B} to replace in q respectively $\top$, $\bot$ and B, then $\mathcal{I}$ and $\mathcal{I}'$ also coincide on all matches for the query q' in which $\top$, $\bot$ and B have been eliminated. Step 2 results in an extension of $\mathcal{K}$ that preserves $\mathcal{K}$-connected models. By interpreting p_U as $p_U^{\mathcal{I}'} = \{(a_i^{\mathcal{I}}, a_{i+1}^{\mathcal{I}}) \mid a_i, a_{i+1}$ occur in $\mathcal{A}\} \cup \{d, d' \mid (d, d') \in P^{\mathcal{I}}$ for some $P \in \bar{\mathbf{R}}_{\mathcal{K}}\}$, we can obtain from each $\mathcal{K}$-connected interpretation $\mathcal{I}$ a new $\mathcal{I}'$ such that $(p_U{}^*)^{\mathcal{I}'} = \Delta^{\mathcal{I}'} \times \Delta^{\mathcal{I}'}$, hence $\mathcal{I}' \models \mathcal{K}'$ iff $\mathcal{I} \models \mathcal{K}$, and $\mathcal{I}$ and $\mathcal{I}'$ coincide on all matches for q. Similarly as above, they also coincide on matches for the query q' in which T has been replaced by $p_U{}^*$. Whenever $\mathcal{K} \not\models q$, there is a $\mathcal{K}$-connected model $\mathcal{I}$ of the original $\mathcal{K}$ such that $\mathcal{I} \not\models q$ (cf. Lemma 3.1), and hence also a model $\mathcal{I}'$ of the rewritten $\mathcal{K}'$ such that $\mathcal{I}' \not\models q$. This shows that the normalization preserves query entailment.

Finally, to show that the transformation is linear, it suffices to observe that each step can be executed in linear time. Steps 1 and 4 only replace a linear number of expressions, and they replace them by expressions of the same size plus some small constant, hence they can be applied in linear time. Steps 2 and 3 replace a linear number of expressions by a fixed concept or role name, add a constant number of CIAs, and step 2 adds a linear number of assertions to $\mathcal{A}$. Finally, in step 5, it is well known that pushing negation inside is feasible in linear time, and a linear number of expressions may be replaced by an expression of constant size. □

3.2. Syntactic closure

Now we introduce the notion of *syntactic closure* of a concept D, which contains all concepts and simple roles (in NNF) that are relevant for deciding its satisfiability. It contains D, is closed under subconcepts and simple subroles, negations, and is also *Fischer–Ladner closed* in the style of a similarly defined closure for PDL [55]. The notion extends naturally to sets of concepts.

We define here the closure for the DL $\mathcal{ALCQIB}^{\mathsf{Self}}_{reg}$, which is similar to $\mathcal{ZIQ}$, 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 an expression 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 every concept or simple role E. The symbol $\gtrless$ is generic for $\geqslant$ and $\leqslant$; 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 that of p^- is p. The inverse of an atomic role P is denoted by $\mathsf{Inv}(P)$. For a 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' stand for concepts, S and S' for simple roles, and R and R' for arbitrary roles.

Definition 3.4. The *closure* $Cl(D)$ of an $\mathcal{ALCQIB}^{\mathsf{Self}}_{reg}$ concept D is the smallest set that contains the NNF of D and is closed under the rules of Table 1. For a set $\mathcal{D}$ of concepts, $Cl(\mathcal{D}) = \bigcup_{D \in \mathcal{D}} Cl(D)$. □

It is easy to see that $|Cl(D)|$ (resp., $|Cl(\mathcal{D})|$) is linear in the length of D (resp., $\mathcal{D}$). Note that $Cl(D)$ may contain unsafe expressions even if D is a $\mathcal{ZIQ}$ concept. For example, the concept $D = \exists \mathit{HasFather}.\mathit{male}$ (which is in negation normal form) has $\neg \mathit{HasFather}$ ($\equiv \mathsf{T} \setminus \mathit{HasFather}$) in $Cl(D)$, which is not expressible in $\mathcal{ZIQ}$.

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 must suitably extend tree-shaped models to forest-shaped models.[5]

First, we observe that in $\mathcal{ZIQ}$ every TBox $\mathcal{T}$ can be *internalized* into a concept $C_{\mathcal{T}}$, such that the satisfiability of $\mathcal{T}$ can be established by obtaining a model of $C_{\mathcal{T}}$ [57].

[5] Unlike the results of the previous subsection, these results do not hold for $\mathcal{ALCQIB}^{\mathsf{Self}}_{reg}$; see [56] for discussion.

Definition 3.5 *(TBox internalization, $C_\mathcal{T}$).* Given a normalized $\mathcal{ZIQ}$ knowledge base $\mathcal{K} = \langle\mathcal{T}, \mathcal{A}\rangle$, we let $C_\mathcal{T}$ be the negation normal form of the concept

$$\forall p_U{}^*. \bigsqcap_{C_1 \sqsubseteq C_2 \in \mathcal{T}} (\neg C_1 \sqcup C_2),$$

where p_U is the role introduced in step 2 of the normalization of $\mathcal{K}$, or if this p_U is not present in $\mathcal{K}$, then $p_U = \bigcup_{P \in \bar{\mathbf{R}}_\mathcal{K}} P$. □

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 and let $\mathcal{K} = \langle\mathcal{T}, \mathcal{A}\rangle$ be a normalized $\mathcal{ZIQ}$ KB. If for each element $d \in \Delta^\mathcal{I}$ there is some $d_0 \in C_\mathcal{T}^\mathcal{I}$ such that d is $\mathcal{K}$-connected to d_0, then $\mathcal{I} \models \mathcal{T}$.*

Proof. Suppose that $d_0 \in C_\mathcal{T}^\mathcal{I}$. Then for every sequence $d_0, \ldots, d_n$, $n \geqslant 0$, such that for each $0 \leqslant i < n$ we have $(d_i, d_{i+1}) \in P^\mathcal{I}$ for some $P \in \bar{\mathbf{R}}_\mathcal{K}$, it holds that $d_n \in (\bigsqcap_{C_1 \sqsubseteq C_2 \in \mathcal{T}} (\neg C_1 \sqcup C_2))^\mathcal{I}$. By $\mathcal{K}$-connectedness, for every element $d \in \Delta^\mathcal{I}$ some sequence of this form exists such that $d = d_n$, so it follows that $(\bigsqcap_{C_1 \sqsubseteq C_2 \in \mathcal{T}} (\neg C_1 \sqcup C_2))^\mathcal{I} = \Delta^\mathcal{I}$; that is, $\mathcal{I} \models C_1 \sqsubseteq C_2$ for each $C_1 \sqsubseteq C_2 \in \mathcal{T}$, which means $\mathcal{I} \models \mathcal{T}$. □

Now we define a *canonical model* of a normalized $\mathcal{ZIQ}$ KB, which has a certain forest-shaped structure that allows us to recognize it using tree automata.

Definition 3.7. For $k \geqslant 0$, an interpretation $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ for a $\mathcal{ZIQ}$ KB $\mathcal{K} = \langle\mathcal{T}, \mathcal{A}\rangle$ is *k-canonical*, if there exists 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 $\Delta^\mathcal{I} \subseteq \mathbb{N}^*$.)
(2) $\mathit{Roots}(\mathcal{I}) = \{a^\mathcal{I} \mid a \in \mathbf{I}_\mathcal{K}\} \subseteq \mathbb{N}$.
(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 exists some atomic role $P \in \bar{\mathbf{R}}_\mathcal{K}$ 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 $\mathcal{K}$-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 a normalized $\mathcal{ZIQ}$ KB $\mathcal{K} = \langle\mathcal{T}, \mathcal{A}\rangle$, for some $k \geqslant 0$. Then $\mathcal{I} \models \mathcal{T}$ iff $\mathit{Roots}(\mathcal{I}) \subseteq C_\mathcal{T}^\mathcal{I}$.*

Proof. (*Only If*) Suppose $\mathcal{I} \models \mathcal{T}$. Then for every element $x \in \Delta^\mathcal{I}$ and every $C_1 \sqsubseteq C_2 \in \mathcal{T}$ we have $x \in (\neg C_1 \sqcup C_2)^\mathcal{I}$. It follows that $(\bigsqcap_{C_1 \sqsubseteq C_2 \in \mathcal{T}} (\neg C_1 \sqcup C_2))^\mathcal{I} = \Delta^\mathcal{I}$, which implies $C_\mathcal{T}^\mathcal{I} = \Delta^\mathcal{I}$; hence $\mathit{Roots}(\mathcal{I}) \subseteq C_\mathcal{T}^\mathcal{I}$.

(*If*) Suppose $\mathit{Roots}(\mathcal{I}) \subseteq C_\mathcal{T}^\mathcal{I}$. Observe that, as $\mathcal{I}$ is a k-canonical interpretation, it is $\mathcal{K}$-connected by definition. That is, for every $x \in \Delta^\mathcal{I}$, there is some $a \in \mathbf{I}_\mathcal{K}$ such that x is $\mathcal{K}$-connected to $a^\mathcal{I}$ (by conditions (2)–(4) in Definition 3.7). Moreover, $a^\mathcal{I} \in \mathit{Roots}(\mathcal{I})$ (by condition (2)), hence $a^\mathcal{I} \in C_\mathcal{T}^\mathcal{I}$. Then $\mathcal{I} \models \mathcal{T}$ follows from Proposition 3.6. □

Now we can establish the *canonical model property* of $\mathcal{ZIQ}$, by a straightforward adaptation of a similar property of related logics [40,45]. It states that, to decide query entailment, it suffices to consider the k-canonical models of the given KB $\mathcal{K}$. Here the branching factor k depends on the size of the closure of all concepts and roles in $\mathcal{T}$ and q, and on the number restrictions and existential concepts that occur therein.

Definition 3.9 *(Branching).* For a set $\mathcal{D}$ of concepts, let $br(\mathcal{D}) = |Cl(\mathcal{D})| \cdot n_{\max}$, where $n_{\max} = \max(\{n \mid \geqslant n S.C \in Cl(\mathcal{D})\} \cup \{0\})$.

Given a normalized $\mathcal{ZIQ}$ KB $\mathcal{K} = \langle\mathcal{T}, \mathcal{A}\rangle$ and a normalized P2RPQ q, we define $k_\mathcal{T} = br(\{C_\mathcal{T}\})$ and $k_{\mathcal{T},q} = br(\{C_\mathcal{T}\} \cup \mathcal{D}_q)$, where $\mathcal{D}_q = \{C \mid C(v) \in At(q)\} \cup \{\exists R.A \mid R(v, v') \in At(q)\}$ for an arbitrary concept name A.

Note that, by definition of $Cl(\mathcal{D})$, if there is some concept $\exists R.C \in Cl(\mathcal{D})$, then there is also some concept $\geqslant 1 S.C \in Cl(\mathcal{D})$. Hence $br(\mathcal{D}) = 0$ only if there are no existential and universal concepts, and no number restrictions in $Cl(\mathcal{D})$.

Theorem 3.10 *(Canonical model property). Let $\mathcal{K} = \langle\mathcal{T}, \mathcal{A}\rangle$ be a normalized $\mathcal{ZIQ}$ KB, and let q be a normalized P2RPQ. If $\mathcal{K} \not\models q$, then there exists some $k_{\mathcal{T},q}$-canonical model $\mathcal{I}$ of $\mathcal{K}$ such that $\mathcal{I} \not\models q$. Furthermore, if $\mathcal{K}$ is satisfiable, then it has a $k_{\mathcal{T}}$-canonical model.*

Proof (Sketch). Following [43,41], with minor adaptations to properly handle ABoxes, Boolean role constructs and Self, one can show that every model $\mathcal{I}$ of $\mathcal{K}$ that admits no match for q can be unraveled into a $k_{\mathcal{T},q}$-canonical model $\mathcal{I}'$ that also admits no match. We refer to Appendix A for details. The second part of the claim follows from the fact that KB satisfiability reduces to query non-entailment using a simple query q such that $br(\mathcal{D}_q) = 0$. □

We remark that the reason why the branching degree of the counterexample canonical model depends on q is that we allow for complex concepts in the query. In fact, given a KB $\mathcal{K} = \langle\mathcal{T}, \mathcal{A}\rangle$, then for every P2RPQ q where only concept names (and arbitrary roles) occur, if $\mathcal{K} \not\models q$, then there is some $k_{\mathcal{T}}$-canonical model $\mathcal{I}$ of $\mathcal{K}$ such that $\mathcal{I} \not\models q$.

By Theorem 3.10, we can restrict to $k_{\mathcal{T}}$-canonical (resp., $k_{\mathcal{T},q}$-canonical) forest-shaped models for deciding KB satisfiability (resp., 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* [45], 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 [1], 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 $\mathcal{K} = \langle\mathcal{T}, \mathcal{A}\rangle$ be a normalized $\mathcal{ZIQ}$ KB, and let $b_{\mathcal{K}} = \max(k_{\mathcal{T}}, |\mathbf{I}_{\mathcal{K}}|)$, where $k_{\mathcal{T}} = br(\{C_{\mathcal{T}}\})$ is as in Definition 3.9. To represent interpretations for $\mathcal{K}$, we define *interpretation trees*, which are labeled $b_{\mathcal{K}}$-ary trees. Each node is labeled 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} (used 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 $PI_{\mathcal{K}} = \{p_{ij} \mid p \in \mathbf{R}_{\mathcal{K}},\ i, j \in \{1, \ldots, b_{\mathcal{K}}\}\}$, 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

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

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 $1 \leqslant x \leqslant b_{\mathcal{K}}$ 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 speaking, 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) the special symbol p_{Self} for each role name 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 does not contain 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:

- $T = \{1, \ldots, b_{\mathcal{K}}\}^*$,

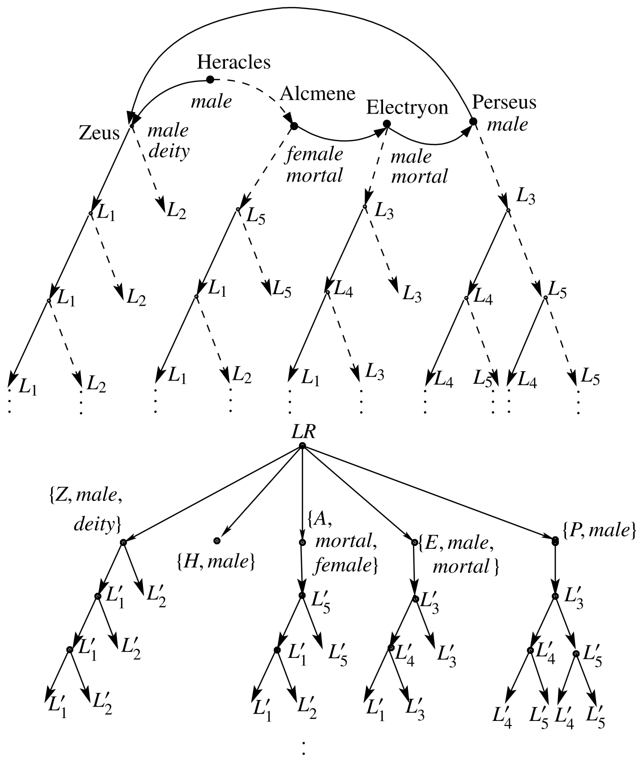

Fig. 2. The canonical model $\mathcal{I}_g$ of $\mathcal{K}_g$ (cf. Fig. 1) and its tree representation.

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

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

Example 4.3. The left part of Fig. 2 depicts a fragment of an infinite 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 labeled with the name of the individual it interprets, and with the concept names from $C_{\mathcal{K}}$ to whose interpretation it belongs. Other domain elements are represented by smaller dots, and are also labeled 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 $L_5 = \{female\}$. 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 explicitly.

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 1 nodes are given explicitly 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 its descendants 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 $\mathcal{R}^2_p$, where $\mathcal{R}^1_p$ contains the pairs (x, y) of p-neighbors x, y in the tree that are not both individual nodes, and $\mathcal{R}^2_p$ contains 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:

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

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

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

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 defined by:

$$\Delta^{\mathcal{I}_\mathbf{T}} = \mathbf{I_T} \cup \bigcup_{i \in \mathbf{I_T}} \mathbf{D}_i, \quad \text{and}$$
$$a^{\mathcal{I}_\mathbf{T}} = x \in \mathbf{I_T} \quad \text{such that } a \in L(x), \text{ for each } a \in \mathbf{I}_{\mathcal{K}}$$
$$A^{\mathcal{I}_\mathbf{T}} = \{x \in \Delta^{\mathcal{I}_\mathbf{T}} \mid A \in L(x)\} \Delta^{\mathcal{I}_\mathbf{T}} \cap \{x \mid A \in L(x)\} \quad \text{for each concept name } A \in \mathbf{C}_{\mathcal{K}},$$
$$p^{\mathcal{I}_\mathbf{T}} = (\Delta^{\mathcal{I}_\mathbf{T}} \times \Delta^{\mathcal{I}_\mathbf{T}}) \cap (\mathcal{R}^1_p \cup \mathcal{R}^2_p) \quad \text{for each role name } p \in \mathbf{R}_{\mathcal{K}} \qquad \square$$

Note that, by condition (t2) in Definition 4.1, for each $a \in \mathbf{I}_{\mathcal{K}}$ there is exactly one x such that $a^{\mathcal{I}_\mathbf{T}} = x$. The set $\mathbf{I_T}$ contains the roots of $\mathcal{I}_\mathbf{T}$, and $\{\varepsilon\} \cup \Delta^{\mathcal{I}_\mathbf{T}}$ is a $b_{\mathcal{K}}$-tree. Note also that i is a root of $\mathcal{I}_\mathbf{T}$ iff $\mathbf{I}_{\mathcal{K}} \cap L(i) \neq \emptyset$.

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

Proof. It is not hard to verify that $\mathcal{I}_\mathbf{T}$ satisfies the conditions (1)–(5) of Definition 3.7 (for $k = b_{\mathcal{K}}$):

(1) $\{\varepsilon\} \cup \Delta^{\mathcal{I}_\mathbf{T}}$ is a tree (note that ε is not in $\Delta^{\mathcal{I}_\mathbf{T}}$);
(2) $Roots(\mathcal{I}_\mathbf{T}) = \{a^{\mathcal{I}_\mathbf{T}} \mid a \in \mathbf{I}_{\mathcal{K}}\} \subseteq \{1, \dots, b_{\mathcal{K}}\} \subseteq \mathbb{N}$;
(3) Each element of $\Delta^{\mathcal{I}_\mathbf{T}}$ is of the form $i{\cdot}x$ with $i \in Roots(\mathcal{I}_\mathbf{T})$ and $x \in \{1, \dots, b_{\mathcal{K}}\}^*$;
(4) For every pair $x, y \in \Delta^{\mathcal{I}_\mathbf{T}}$ with y of the form $x{\cdot}i$, there exists some atomic role P such that $(x, y) \in P^{\mathcal{I}_\mathbf{T}}$.
(5) If $(x, y) \in p^{\mathcal{I}_\mathbf{T}}$ for some role name p and some $x, y \in \Delta^{\mathcal{I}_\mathbf{T}}$, then either (a) $x, y \in Roots(\mathcal{I}_\mathbf{T})$, or (b) for some $i \in Roots(\mathcal{I}_\mathbf{T})$, 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.

Thus $\mathcal{I}_\mathbf{T}$ is $b_{\mathcal{K}}$-canonical. As for the second part, the domain of $\mathcal{I}_{\mathbf{T}_\mathcal{I}}$ coincides with $\Delta^\mathcal{I}$: $\Delta^\mathcal{I}$ is connected and hence in the construction of $\mathcal{I}_{\mathbf{T}_\mathcal{I}}$ from $\mathbf{T}_\mathcal{I}$, the set $\mathbf{I}_{\mathbf{T}_\mathcal{I}}$ coincides with $Roots(\mathcal{I})$ and $\mathbf{I}_{\mathbf{T}_\mathcal{I}} \cup \bigcup_{i \in \mathbf{I}_{\mathbf{T}_\mathcal{I}}} \mathbf{D}_i$ contains all elements of $\Delta^\mathcal{I}$ (and no others). Furthermore, by construction $a^\mathcal{I} = a^{\mathcal{I}_{\mathbf{T}_\mathcal{I}}}$ for each $a \in \mathbf{I}_{\mathcal{K}}$ and for each $x \in \mathcal{I}_{\mathbf{T}_\mathcal{I}}$ and concept name A, we can verify that $x \in A^{\mathcal{I}_{\mathbf{T}_\mathcal{I}}}$ iff $x \in A^\mathcal{I}$ and for each $x, y \in \mathcal{I}_{\mathbf{T}_\mathcal{I}}$ and role name p, that $(x, y) \in p^{\mathcal{I}_{\mathbf{T}_\mathcal{I}}}$ iff $(x, y) \in p^\mathcal{I}$. Hence $\mathcal{I} = \mathcal{I}_{\mathbf{T}_\mathcal{I}}$. $\square$

4.2. Constructing the automaton to verify KB satisfaction

In this section, we show how to construct from a normalized $\mathcal{ZIQ}$ KB $\mathcal{K}$ 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: (1) 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 $\mathcal{K}$. (2) We construct another 2ATA $\mathbf{A}_\mathcal{A}$ that accepts $\mathbf{T}$ iff the represented interpretation satisfied all assertions in $\mathcal{A}$. (3) We construct a third 2ATA $\mathbf{A}_\mathcal{T}$ which verifies whether each individual in $\mathbf{I}_K$ satisfies $C_\mathcal{T}$. (4) Finally, by intersecting the three 2ATAs, we obtain $\mathbf{A}_{\mathcal{K}}$. We note that all these automata, which are summarized in Table 2, run over $b_{\mathcal{K}}$-ary trees labeled with the alphabet $\Sigma_{\mathcal{K}}$ from Definition 4.1. As we will see in the next section, their size is polynomial in $\mathcal{K}$.

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, that is, whether it correctly represents an interpretation that is canonical for $\mathcal{K}$.

Definition 4.6. Let $\mathbf{A}_\mathcal{I} = \langle b_{\mathcal{K}}, \Sigma_{\mathcal{K}}, Q_\mathcal{I}, \delta_\mathcal{I}, s^\mathcal{I}_0, F_\mathcal{I}\rangle$, where:

- The set of states is $Q_\mathcal{I} = \{s^\mathcal{I}_0, s_1, r, \neg r\} \cup \mathbf{I}_{\mathcal{K}} \cup \{\neg s \mid s \in \mathbf{I}_{\mathcal{K}}\}$.
- The transition function $\delta_\mathcal{I} : Q_\mathcal{I} \times \Sigma_{\mathcal{K}} \to \mathcal{B}([b_{\mathcal{K}}] \times Q_\mathcal{I})$ contains the following transitions:

Table 2
Automata for checking KB satisfiability.

Automaton	Construction	Language
$\mathbf{A}_\mathcal{I}$	Definition 4.6	Trees representing a canonical interpretation for $\mathcal{K}$
$\mathbf{A}_\mathcal{A}$	Definition 4.8	Trees such that if they represent a (canonical) interpretation, all assertions in $\mathcal{A}$ are satisfied
$\mathbf{A}_\mathcal{T}$	Definition 4.10	Trees such that if they represent a (canonical) interpretation, all elements satisfy $C_\mathcal{T}$
$\mathbf{A}_\mathcal{K}$	Definition 4.16, intersection of $\mathbf{A}_\mathcal{I}$, $\mathbf{A}_\mathcal{A}$ and $\mathbf{A}_\mathcal{T}$	Trees representing canonical models of $\mathcal{K}$

1. for each $\sigma \in \Sigma_\mathcal{K}$ with $r \in \sigma$, a transition $\delta_\mathcal{I}(s_0^\mathcal{I}, \sigma) = B_1 \wedge B_2 \wedge B_3$, where

$$B_1 \bigwedge_{a\in\mathbf{I}_\mathcal{K}} \bigvee_{1\leqslant i\leqslant b_\mathcal{K}} (i, a)$$

$$B_2 \bigwedge_{1\leqslant i<j\leqslant b_\mathcal{K}} \bigwedge_{a\in\mathbf{I}_\mathcal{K}} \big((i, \neg a) \vee (j, \neg a)\big)$$

$$B_3 \bigwedge_{1\leqslant i\leqslant b_\mathcal{K}} \big((i, \neg r) \wedge (i, s_1)\big),$$

   and for each $\sigma \in \Sigma_\mathcal{K}$, a transition $\delta_\mathcal{I}(s_1, \sigma) = \bigwedge_{1\leqslant i\leqslant b_\mathcal{K}}((i, \neg r) \wedge (\bigwedge_{a\in\mathbf{I}_\mathcal{K}}(i, \neg a)) \wedge (i, s_1))$;
2. for each $\sigma \in \Sigma_\mathcal{K}$ and each $s \in \mathbf{I}_\mathcal{K} \cup \{r\}$, transitions

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

- The acceptance condition is $F_\mathcal{I} = (\emptyset, Q_\mathcal{I})$. □

The transitions in item 2 simply verify whether the label of a node reached in state s 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 4.1:

- 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 level 1 nodes.
- B_3 checks that the labels of the nodes of level 1 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 occur anywhere below level 1 in the whole tree.

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

Proof. $\mathbf{A}_\mathcal{I}$ accepts an input $\Sigma_\mathcal{K}$-labeled $b_\mathcal{K}$-ary tree $\mathbf{T} = (T, L)$ iff there is an accepting run of $\mathbf{A}_\mathcal{I}$ over $\mathbf{T}$. If $\mathbf{T}$ is an interpretation tree, by condition (t2) in Definition 4.1, there is exactly one node $x \in T$ with $1 \leqslant x \leqslant b_\mathcal{K}$ and $a \in L(x)$, which we denote by x_a. Then we can build an accepting run (T_r, r) as follows. The root is labeled $r(\varepsilon) = (\varepsilon, s_0^\mathcal{I})$, and it has the following children:

1. For each $a \in \mathbf{I}_\mathcal{K}$, there is a child c_a labeled (x_a, a), which is a leaf of $\mathcal{T}_r$ (i.e., it has no children). These c_a ensure that B_1 is satisfied at the root, and since $a \in L(x_a)$ for each x_a and $\delta_\mathcal{I}(a, \sigma) = \text{true}$ whenever $a \in \sigma$, each c_a already satisfies (R2) in Definition 2.8.
2. For each $a \in \mathbf{I}_\mathcal{K}$ and each pair i, j with $1 \leqslant i < j \leqslant b_\mathcal{K}$, if $i \neq x_a$ then ε has a child c_{aij} labeled $(i, \neg a)$, and if $i = x_a$ then ε has a child c_{aij} labeled $(j, \neg a)$. In the former case, $i \neq x_a$ implies $a \notin L(i)$, hence $\delta_\mathcal{I}(\neg a, L(i)) = \text{true}$ and (R2) in Definition 2.8 holds for c_{aij}. In the latter case, $i = x_a$ implies $j \neq x_a$, so we also have $a \notin L(j)$ and $\delta_\mathcal{I}(\neg a, L(j)) = \text{true}$ as required. Moreover, the children c_{aij} ensure that ε satisfies B_2.
3. Finally, to satisfy B_3, ε has for each $1 \leqslant i \leqslant b_\mathcal{K}$ a child c_i labeled $(i, \neg r)$ (which satisfies (R2) in Definition 2.8 as $r \notin L(i)$ by condition (t1) and hence $\delta_\mathcal{I}(\neg r, L(i)) = \text{true}$), and a child c'_i labeled (i, s_1). To satisfy (R2) in Definition 2.8, each c'_w has children as follows:
   (a) a leaf c_{wj} labeled $(w{\cdot}j, \neg r)$ for each $1 \leqslant j \leqslant b_\mathcal{K}$. Since $r \notin L(w{\cdot}j)$, $\delta_\mathcal{I}(\neg r, L(w{\cdot}j)) = \text{true}$ as required.
   (b) a leaf c_{awj} labeled $(w{\cdot}j, \neg a)$ for each $1 \leqslant j \leqslant b_\mathcal{K}$ and $a \in \mathbf{I}_\mathcal{K}$. Since $a \notin L(w{\cdot}j)$, $\delta_\mathcal{I}(\neg s, L(w{\cdot}j)) = \text{true}$.
   (c) a node c'_{wj} $(w{\cdot}j, s_1)$ for each $1 \leqslant j \leqslant b_\mathcal{K}$, which in turn has children as described by items (a)–(c).

The only infinite paths in (T_r, r) are the sequences of nodes c'_{wj} labeled $(w{\cdot}j, s_1)$. Since such a path visits s_1 infinitely often the acceptance condition is satisfied and the run is accepting.

Conversely if a run (T_r, r) accepts $\mathbf{T}$, we must have $r \in L(\varepsilon)$, and to satisfy the formulas B_1, B_2 and B_3, ε must have children analogous to those described in items 1–3 above, which enforce conditions (t1)–(t3). Namely:

1. For each $a \in \mathbf{I}_\mathcal{K}$, a child c_a labeled (x, a) for some $1 \leqslant x \leqslant b_\mathcal{K}$ with $a \in L(x)$. This ensures part of (t2): that each a is in $L(x)$ for some $1 \leqslant x \leqslant b_\mathcal{K}$.
2. For each pair i, j with $1 \leqslant i < j \leqslant b_\mathcal{K}$ and each $a \in \mathbf{I}_\mathcal{K}$, a child c_{aij} labeled $(x, \neg a)$ for some $x \in \{i, j\}$ with $a \notin L(x)$. This ensures that the x with $a \in L(x)$ is unique, and together with item 1, it ensures (t2).
3. For $1 \leqslant i \leqslant b_\mathcal{K}$ a child c_i labeled $(i, \neg r)$, which implies that $r \notin L(i)$, and a child c'_i labeled (i, s_1). Since each node y in T_r with label of the form (w, s_1) must have a child c_{wj} with $r(w{\cdot}j, \neg r)$ for each $1 \leqslant j \leqslant b_\mathcal{K}$, and a child c_{awj} labeled $(w{\cdot}j, \neg a)$ for each $1 \leqslant j \leqslant b_\mathcal{K}$ and $a \in \mathbf{I}_\mathcal{K}$, it follows that $L(w{\cdot}j) \cap (\mathbf{I}_\mathcal{K} \cup \{r\}) = \emptyset$. Moreover, y must also have a child with label $r(w{\cdot}j, s_1)$ that in turn has analogous successors, thus $L(w') \cap (\mathbf{I}_\mathcal{K} \cup \{r\}) = \emptyset$ for all children w' of $w{\cdot}j$. Hence r and the symbols in $\mathbf{I}_\mathcal{K}$ can not occur in any label below the level 1 nodes. Thus (t1) and (t3) hold and $\mathbf{T}$ is an interpretation tree. □

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

Next, we define the automaton $\mathbf{A}_\mathcal{A}$ that verifies whether the interpretation represented by a given $\Sigma_\mathcal{K}$-labeled $b_\mathcal{K}$-ary tree $\mathbf{T}$ satisfies all assertions in $\mathcal{A}$, assuming that $\mathbf{T}$ is 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 $\mathbf{A}_\mathcal{A} = \langle b_\mathcal{K}, \Sigma_\mathcal{K}, Q_\mathcal{A}, \delta_\mathcal{A}, s_0^\mathcal{A}, F_\mathcal{A}\rangle$, where:

- The set of states is $Q_\mathcal{A} = \{s_0^\mathcal{A}\} \cup \mathbf{I}_\mathcal{K} \cup \{\neg s \mid s \in \mathbf{I}_\mathcal{K}\} \cup \mathbf{C}_\mathcal{A} \cup \{p_{ij} \mid p \in \mathbf{R}_\mathcal{A} \text{ and } i, j \in \{1, \ldots, b_\mathcal{K}\}\}$.
- The transition function $\delta_\mathcal{A} : Q_\mathcal{A} \times \Sigma_\mathcal{K} \to \mathcal{B}([b_\mathcal{K}] \times Q_\mathcal{A})$ contains the following transitions:
  1. for each $\sigma \in \Sigma_\mathcal{K}$ with $r \in \sigma$, a transition $\delta_\mathcal{A}(s_0^\mathcal{A}, \sigma) = B_4 \wedge B_5 \wedge B_6$, where

$$B_4 \quad \bigwedge_{a \not\approx b \in \mathcal{A}} \ \bigwedge_{1 \leqslant i \leqslant b_\mathcal{K}} \big((i, \neg a) \vee (i, \neg b)\big)$$

$$B_5 \quad \bigwedge_{A(a) \in \mathcal{A}} \ \bigvee_{1 \leqslant i \leqslant b_\mathcal{K}} \big((i, a) \wedge (i, A)\big)$$

$$B_6 \quad \bigwedge_{p(a,b) \in \mathcal{A}} \ \bigvee_{1 \leqslant i \leqslant b_\mathcal{K}, 1 \leqslant j \leqslant b_\mathcal{K}} \big((0, p_{ij}) \wedge (i, a) \wedge (j, b)\big);$$

  2. for each $\sigma \in \Sigma_\mathcal{K}$ and each $s \in \mathbf{I}_\mathcal{K}$, transitions

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

  and for each $\sigma \in \Sigma_\mathcal{K}$ 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

$$\delta_\mathcal{A}(t, \sigma) = \begin{cases} \text{true}, & \text{if } t \in \sigma \\ \text{false}, & \text{if } t \notin \sigma \end{cases}.$$

- The acceptance condition is $F_\mathcal{A} = (\emptyset, Q_\mathcal{A})$. □

Similarly to $\mathbf{A}_\mathcal{I}$, 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 with A.
- B_6 ensures that each assertion $p(a, b)$ is satisfied, by finding the individual nodes i and j that represent the individuals a and b, respectively, 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 $\mathcal{I}_\mathbf{T} \models \mathcal{A}$.*

Proof (Sketch). The argument is similar to the one in the proof of Lemma 4.7. Again, by design of the formulas B_4–B_6, for an interpretation tree $\mathbf{T}$ that encodes an interpretation $\mathcal{I}_\mathbf{T}$ where $\mathcal{A}$ is satisfied, we can find an accepting run (T_r, r) over $\mathbf{T}$ (in fact, a finite such run). On the other hand, every accepting run (T_r, r) over $\mathbf{T}$ must satisfy B_4–B_6 at the root ε, and hence by design of $\delta_\mathcal{I}$, each assertion in $\mathcal{A}$ must be satisfied by the interpretation $\mathcal{I}_\mathbf{T}$, i.e., $\mathcal{I}_\mathbf{T} \models \mathcal{A}$. □

Table 3
State set $\mathrm{Q}_{\mathcal{T}}$ of the automaton $\mathbf{A}_{\mathcal{T}}$.

$$
\begin{array}{l}
Cl_{ext} = Cl(C_{\mathcal{T}}) \cup \mathbf{I}_{\mathcal{K}} \cup \{\neg s \mid s \in \mathbf{I}_{\mathcal{K}}\} \\
\mathrm{Q}_{\mathsf{Self}} = \{S_{\mathsf{Self}} \mid S \text{ is a simple role in } Cl(C_{\mathcal{T}})\} \cup \{\langle a, p_{\mathsf{Self}}\rangle \mid a \in \mathbf{I}_{\mathcal{K}}, p \text{ is a role name in } Cl(C_{\mathcal{T}})\} \\
\mathrm{Q}_{\mathcal{A}\_role} = \{S_{ij} \mid 1 \leqslant i, j \leqslant b_{\mathcal{K}}, S \text{ is a simple role in } Cl(C_{\mathcal{T}})\} \\
\mathrm{Q}_{num} = \{\langle \geqslant nS.C, i, j\rangle \mid \geqslant nS.C \in Cl(C_{\mathcal{T}}),\ 0 \leqslant i \leqslant b_{\mathcal{K}}+1,\ 0 \leqslant j \leqslant n\} \cup \{\langle \leqslant nS.C, i, j\rangle \mid \leqslant nS.C \in Cl(C_{\mathcal{T}}),\ 0 \leqslant i \leqslant b_{\mathcal{K}}+1,\ 0 \leqslant j \leqslant n+1\} \\
\mathrm{Q}_{\mathcal{A}\_num} = \{\langle a, \geqslant nS.C, i, j\rangle \mid a \in \mathbf{I}_{\mathcal{K}},\ \geqslant nS.C \in Cl(C_{\mathcal{T}}),\ 1 \leqslant i \leqslant b_{\mathcal{K}},\ 0 \leqslant j \leqslant n\} \cup \{\langle a, \leqslant nS.C, i, j\rangle \mid a \in \mathbf{I}_{\mathcal{K}},\ \leqslant nS.C \in Cl(C_{\mathcal{T}}),\ 1 \leqslant i \leqslant b_{\mathcal{K}},\ 0 \leqslant j \leqslant n+1\} \\
\mathrm{Q}_{\mathcal{T}} = \{s_0^{\mathcal{T}}\} \cup Cl_{ext} \cup \mathrm{Q}_{\mathsf{Self}} \cup \mathrm{Q}_{\mathcal{A}\_role} \cup \mathrm{Q}_{num} \cup \mathrm{Q}_{\mathcal{A}\_num}
\end{array}
$$

Table 4
Transitions of the automaton $\mathbf{A}_{\mathcal{T}}$, part 1: initialization, concept and role checking, atomic checks.

(I) *Initialization.* For each $\sigma \in \Sigma_{\mathcal{K}}$ with $r \in \sigma$, we have: $\delta_{\mathcal{T}}(s_0, \sigma) = \bigwedge_{1 \leqslant i \leqslant b_{\mathcal{K}}}((\bigwedge_{a \in \mathbf{I}_{\mathcal{K}}}(i, \neg a)) \vee (i, C_{\mathcal{T}}))$.

(II) *Concept and role checking.* For each $\sigma \in \Sigma_{\mathcal{K}}$, each non-atomic concept $C \in Cl(C_{\mathcal{T}})$, each simple role$S \in Cl(C_{\mathcal{T}})$, each atomic role $P \in Cl(C_{\mathcal{T}})$, each role name $p \in Cl(C_{\mathcal{T}})$, and each pair $i, j \in \{1, \dots, b_{\mathcal{K}}\}$, we have:

$\delta_{\mathcal{T}}(C \sqcap C', \sigma) = (0, C) \wedge (0, C')$	$\delta_{\mathcal{T}}(S \cap S', \sigma) = (0, S) \wedge (0, S')$
$\delta_{\mathcal{T}}(C \sqcup C', \sigma) = (0, C) \vee (0, C')$	$\delta_{\mathcal{T}}(S \cup S', \sigma) = (0, S) \vee (0, S')$
	$\delta_{\mathcal{T}}(S \setminus P, \sigma) = (0, S) \wedge (0, \neg P)$
$\delta_{\mathcal{T}}(\exists S.\mathsf{Self}, \sigma) = (0, S_{\mathsf{Self}})$	$\delta_{\mathcal{T}}((S \cap S')_{ij}, \sigma) = (0, S_{ij}) \wedge (0, S'_{ij})$
$\delta_{\mathcal{T}}(\forall(R \cup R').C, \sigma) = (0, \forall R.C) \wedge (0, \forall R'.C)$	$\delta_{\mathcal{T}}((S \cup S')_{ij}, \sigma) = (0, S_{ij}) \vee (0, S'_{ij})$
$\delta_{\mathcal{T}}(\forall(R \circ R').C, \sigma) = (0, \forall R.\forall R'.C)$	$\delta_{\mathcal{T}}((S \setminus P)_{ij}, \sigma) = (0, S_{ij}) \wedge (0, (\neg P)_{ij})$
$\delta_{\mathcal{T}}(\forall R^*.C, \sigma) = (0, C) \wedge (0, \forall R.\forall R^*.C)$	$\delta_{\mathcal{T}}(p^-_{ij}, \sigma) = (0, p_{ji})$
$\delta_{\mathcal{T}}(\forall id(C).C', \sigma) = (0, \sim C) \vee (0, C')$	$\delta_{\mathcal{T}}((\neg p^-)_{ij}, \sigma) = (0, (\neg p)_{ji})$
$\delta_{\mathcal{T}}(\exists(R \cup R').C, \sigma) = (0, \exists R.C) \vee (0, \exists R'.C)$	
$\delta_{\mathcal{T}}(\exists(R \circ R').C, \sigma) = (0, \exists R.\exists R'.C)$	$\delta_{\mathcal{T}}((S \cap S')_{\mathsf{Self}}, \sigma) = (0, S_{\mathsf{Self}}) \wedge (0, S'_{\mathsf{Self}})$
$\delta_{\mathcal{T}}(\exists R^*.C, \sigma) = (0, C) \vee (0, \exists R.\exists R^*.C)$	$\delta_{\mathcal{T}}((S \cup S')_{\mathsf{Self}}, \sigma) = (0, S_{\mathsf{Self}}) \vee (0, S'_{\mathsf{Self}})$
$\delta_{\mathcal{T}}(\exists id(C).C', \sigma) = (0, C) \wedge (0, C')$	$\delta_{\mathcal{T}}((S \setminus P)_{\mathsf{Self}}, \sigma) = (0, S_{\mathsf{Self}}) \wedge (0, \neg P_{\mathsf{Self}})$
$\delta_{\mathcal{T}}(\forall S.C, \sigma) = (0, \leqslant 0S.\sim C)$	$\delta_{\mathcal{T}}(p^-{}_{\mathsf{Self}}, \sigma) = (0, p_{\mathsf{Self}})$
$\delta_{\mathcal{T}}(\exists S.C, \sigma) = (0, \geqslant 1S.C)$	$\delta_{\mathcal{T}}((\neg p^-)_{\mathsf{Self}}, \sigma) = (0, (\neg p)_{\mathsf{Self}})$

$$
\delta_{\mathcal{T}}(p_{\mathsf{Self}}, \sigma) = \begin{cases} \mathsf{true}, & \text{if } \sigma \cap \mathbf{I}_{\mathcal{K}} = \emptyset \text{ and } p \in \sigma \\ \mathsf{false}, & \text{if } \sigma \cap \mathbf{I}_{\mathcal{K}} = \emptyset \text{ and } p \notin \sigma \\ \bigvee_{a \in \sigma \cap \mathbf{I}_{\mathcal{K}}}((0, a) \wedge (-1, \langle a, p_{\mathsf{Self}}\rangle)), & \text{if } \sigma \cap \mathbf{I}_{\mathcal{K}} \neq \emptyset \end{cases}
$$

$$
\delta_{\mathcal{T}}(\langle a, p_{\mathsf{Self}}\rangle, \sigma) = \bigvee_{1 \leqslant i \leqslant b_{\mathcal{K}}}((i, a) \wedge (0, p_{ii})), \quad \text{if } r \in \sigma
$$

(III) *Atomic checks.* For each $\sigma \in \Sigma_{\mathcal{K}}$ and each $s \in \mathbf{C}_{\mathcal{K}} \cup \bar{\mathbf{R}}_{\mathcal{K}} \cup \mathbf{I}_{\mathcal{K}} \cup PI_{\mathcal{K}}$, we have:

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

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

Next 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 $\mathcal{I}$ 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 a complex concept that is formed using many of the different constructors available in $\mathcal{ZIQ}$.

Definition 4.10. Let $\mathbf{A}_{\mathcal{T}} = \langle b_{\mathcal{K}}, \Sigma_{\mathcal{K}}, \mathrm{Q}_{\mathcal{T}}, \delta_{\mathcal{T}}, s_0^{\mathcal{T}}, F_{\mathcal{T}}\rangle$, where:

- The set of states $\mathrm{Q}_{\mathcal{T}}$ is shown in Table 3.
- The transition function $\delta_{\mathcal{T}} : \mathrm{Q}_{\mathcal{T}} \times \Sigma_{\mathcal{K}} \to \mathcal{B}([b_{\mathcal{K}}] \times \mathrm{Q}_{\mathcal{T}})$ is given by the following groups of transitions:
  (I) initialization (Table 4),
  (II) concept and role checking (Table 4),
  (III) atomic checks (Table 4), and
  (IV) number restriction checking (Table 5).
- The acceptance condition is $F_{\mathcal{T}} = (\emptyset, \{\forall R^*.C \mid \forall R^*.C \in Cl(C_{\mathcal{T}})\}, \mathrm{Q}_{\mathcal{T}})$. □

Informally, the states in Cl_{ext} are used to check whether the node satisfies the corresponding concept, resp. it is (not) a particular ABox individual. The states in $\mathrm{Q}_{\mathsf{Self}}$ are used to check whether a role connects a non-ABox individual with itself; the states in $\mathrm{Q}_{\mathcal{A}\_role}$ are used to check whether a role connects two ABox individuals (i.e., level 1 nodes); and the states in Q_{num} (resp., $\mathrm{Q}_{\mathcal{A}\_num}$) serve for checking number restrictions of a non-ABox individual (resp., ABox individual) node.

To explain the state set in more detail, and the intuition behind the transition function of $\mathbf{A}_{\mathcal{T}}$, we describe informally a run on a given interpretation tree $\mathbf{T}$.

I. *Initialization.* $\mathbf{A}_{\mathcal{T}}$ starts a run over $\mathbf{T}$ at the root ε, in state s_0, and reading some $\sigma \in \Sigma_{\mathcal{K}}$ such that $r \in \sigma$. Then, by the *initialization* transitions in item I of Table 4, it 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, as the label of i is empty.

Table 5
Transitions of the automaton $\mathbf{A}_{\mathcal{T}}$, part 2: number restriction checking.

(IV) *Number restriction checking.*

1. For each $\gtrless nS.C$ in $Cl(C_{\mathcal{T}})$ and $\sigma \in \Sigma_{\mathcal{K}}$, we have: $\delta_{\mathcal{T}}(\gtrless nS.C, \sigma) = (0, \langle \gtrless nS.C, 0, 0\rangle)$
2. For each state $\langle \gtrless nS.C, i, j\rangle \in \mathsf{Q}_{num}$ and $\sigma \in \Sigma_{\mathcal{K}}$, we have:
   2.1. $\delta_{\mathcal{T}}(\langle \gtrless nS.C, i, j\rangle, \sigma) = (((i+1, \sim S) \vee (i+1, \sim C)) \wedge (0, \langle \gtrless nS.C, i+1, j\rangle)) \vee ((i+1, S) \wedge (i+1, C) \wedge (0, \langle \gtrless nS.C, i+1, j+1\rangle))$
   Additionally, if $\sigma \cap \mathbf{I}_{\mathcal{K}} = \emptyset$, we have:
   2.2. $\delta_{\mathcal{T}}(\langle \gtrless nS.C, b_{\mathcal{K}}, j\rangle, \sigma) = (((0, \sim S_{\mathsf{Self}}) \vee (0, \sim C)) \wedge (0, \langle \gtrless nS.C, b_{\mathcal{K}}+1, j\rangle)) \vee ((0, S_{\mathsf{Self}}) \wedge (0, C) \wedge (0, \langle \gtrless nS.C, b_{\mathcal{K}}+1, j+1\rangle))$
   2.3. $\delta_{\mathcal{T}}(\langle \gtrless nS.C, b_{\mathcal{K}}+1, j\rangle, \sigma) = (((0, \sim \mathsf{Inv}(S)) \vee (-1, \sim C)) \wedge (0, \langle \gtrless nS.C, b_{\mathcal{K}}+2, j\rangle)) \vee ((0, \mathsf{Inv}(S)) \wedge (-1, C) \wedge (0, \langle \gtrless nS.C, b_{\mathcal{K}}+2, j+1\rangle))$
   Otherwise, if $\sigma \cap \mathbf{I}_{\mathcal{K}} \neq \emptyset$, we have:
   2.4. $\delta_{\mathcal{T}}(\langle \gtrless nS.C, b_{\mathcal{K}}, j\rangle, \sigma) = \bigvee_{a \in \sigma \cap \mathbf{I}_{\mathcal{K}}} ((-1, \langle a, \gtrless nS.C, 0, j\rangle))$
   For all states in $\mathsf{Q}_{\mathcal{A}\_num}$ and all $\sigma \in \Sigma_{\mathcal{K}}$ with $r \in \sigma$, we have:
   2.5. $\delta_{\mathcal{T}}(\langle a, \gtrless nS.C, i, j\rangle, \sigma) = ((\bigwedge_{1 \leqslant \ell \leqslant b_{\mathcal{K}}} ((0, \sim S_{\ell(i+1)}) \vee (\ell, \neg a)) \vee (i+1, \neg C) \vee (\bigwedge_{b \in \mathbf{I}_{\mathcal{K}}} (i+1, \neg b))) \wedge (0, \langle a, \gtrless nS.C, i+1, j\rangle)) \vee$
   $(\bigvee_{1 \leqslant \ell \leqslant b_{\mathcal{K}}} ((0, S_{\ell(i+1)}) \wedge (\ell, a)) \wedge (i+1, C) \wedge (\bigvee_{b \in \mathbf{I}_{\mathcal{K}}} (i+1, b)) \wedge (0, \langle a, \gtrless nS.C, i+1, j+1\rangle))$
   where the counters i and j range over the following values:
   $$0 \leqslant i < b_{\mathcal{K}}, \quad \begin{cases} 0 \leqslant j < n, & \text{if } \gtrless \text{ is } \geqslant \\ 0 \leqslant j \leqslant n, & \text{if } \gtrless \text{ is } \leqslant \end{cases}$$
3. For each $\sigma \in \Sigma_{\mathcal{K}}$, we have:
   $\delta_{\mathcal{T}}(\langle \geqslant nS.C, i, n\rangle, \sigma) = \mathsf{true}$, for $0 \leqslant i \leqslant b_{\mathcal{K}}+2$
   $\delta_{\mathcal{T}}(\langle \geqslant nS.C, b_{\mathcal{K}}+2, j\rangle, \sigma) = \mathsf{false}$, for $0 \leqslant j \leqslant n-1$
   $\delta_{\mathcal{T}}(\langle \leqslant nS.C, i, n+1\rangle, \sigma) = \mathsf{false}$, for $0 \leqslant i \leqslant b_{\mathcal{K}}+2$
   $\delta_{\mathcal{T}}(\langle \leqslant nS.C, b_{\mathcal{K}}+2, j\rangle, \sigma) = \mathsf{true}$, for $0 \leqslant j \leqslant n$
   $\delta_{\mathcal{T}}(\langle a, \geqslant nS.C, i, n\rangle, \sigma) = \mathsf{true}$, for $1 \leqslant i \leqslant b_{\mathcal{K}}$
   $\delta_{\mathcal{T}}(\langle a, \geqslant nS.C, b_{\mathcal{K}}+1, j\rangle, \sigma) = \mathsf{false}$, for $0 \leqslant j \leqslant n-1$
   $\delta_{\mathcal{T}}(\langle a, \leqslant nS.C, i, n+1\rangle, \sigma) = \mathsf{false}$, for $1 \leqslant i \leqslant b_{\mathcal{K}}+1$
   $\delta_{\mathcal{T}}(\langle a, \leqslant nS.C, b_{\mathcal{K}}+1, j\rangle, \sigma) = \mathsf{true}$, for $0 \leqslant j \leqslant n$

II. *Concept and role checking.* Next, from state $C_{\mathcal{T}}$ and each node representing an individual, $\mathbf{A}_{\mathcal{T}}$ recursively decomposes $C_{\mathcal{T}}$ and navigates its formula tree, in order to establish its satisfaction. This recursive decomposition comprises the 'core' of the run. It uses the states in $Cl(C_{\mathcal{T}})$ and the transitions in the left column of item II in Table 4. The automaton moves to a state $C \in Cl(C_{\mathcal{T}})$ and a node x in order to check whether x represents an instance of C. Complex concepts and the non-simple roles occurring in them are decomposed, and $\mathbf{T}$ is navigated accordingly. Please note that $\mathbf{A}_{\mathcal{T}}$ decomposes non-simple roles inside universal and existential restrictions until it reaches expressions of the form $\exists S.C$ and $\forall S.C$ with S simple, then it verifies their satisfaction by moving to states $\geqslant 1S.C$ and $\leqslant 0S. \sim C$, respectively, and uses the transitions for number restrictions from Table 5 (which are explained below). Observe also that all transitions in item II of Table 4 move to states with strictly less complex expressions, except for the transitions that move from states of the form $\forall R^*.C$ and $\exists R^*.C$ to states of the form $\forall R.\forall R^*.C$ and $\exists R.\exists R^*.C$. As we will discuss below, these are the only transitions that may cause infinite runs.
The transitions for number restrictions, which will be explained next, may move to a state S corresponding to a simple role in $Cl(C_{\mathcal{T}})$, in order to verify whether S holds between a node x and its predecessor. To this aim, S is also recursively decomposed using the first group of transitions in the right column of item II in Table 4 (note that S and all its subroles are in Cl_{ext} and thus are states of $\mathbf{A}_{\mathcal{T}}$). If $\mathbf{A}_{\mathcal{T}}$ must verify S between individual nodes i and j, it proceeds similarly but uses the second group of transitions and the states in $\mathsf{Q}_{\mathcal{A}\_role}$. Finally, if the automaton must verify whether S connects a node to itself (this is relevant for the satisfaction of the number restrictions, and of the concepts of the form $\exists S.\mathsf{Self}$), it uses the last group of transitions and the states in $\mathsf{Q}_{\mathsf{Self}}$.

III. *Atomic checks.* Modulo the generation of possibly infinite sequences of states containing $\exists R^*.C$ and $\forall R^*.C$, the decomposition of concepts and roles (item (II)) always stops when $\mathbf{A}_{\mathcal{T}}$ reaches the 'atomic level' that comprises possibly negated atomic concepts and roles, special symbols in $PI_{\mathcal{K}}$, and individual names $\mathbf{I}_{\mathcal{K}}$. These are checked locally at the node labels, using corresponding states, by the respective transitions given in Table 4.

IV. *Number restriction checking.* To verify the satisfaction of a number restriction, $\mathbf{A}_{\mathcal{T}}$ needs to navigate all nodes to which the current node can be connected via some role. We say that a node y is a *potential neighbor* of a node $x \neq \varepsilon$, if either (i) $y = x$, (ii) y is a successor of x, (iii) $y \neq \varepsilon$ is the predecessor of x, or (iv) both x and y are level one nodes. Note that, by definition, the potential neighbors include all successors of a node, also the 'dummy' nodes in $\mathbf{T}$ that do not correspond to an object in $\Delta^{\mathcal{I}_{\mathbf{T}}}$. For a level 1 node, its potential neighbors also include all the level 1 nodes, both actual individual nodes and 'dummy' nodes. We order the potential neighbors of a node x as follows: the first $b_{\mathcal{K}}$ ones are its successors, in the order they occur in $\mathbf{T}$. If x is not a level 1 node, then the $(b_{\mathcal{K}}+1)$-th potential neighbor of x is x itself, and the $(b_{\mathcal{K}}+2)$-th is its predecessor. Otherwise, the $(b_{\mathcal{K}}+1)$-th to $(2b_{\mathcal{K}})$-th potential neighbors are the level 1 nodes, in the order they appear in $\mathbf{T}$.
Moreover, for a simple role S and a concept C, we say that a node y is an (S, C)-*neighbor* of a node x, if (x, y) is in $S^{\mathcal{I}_{\mathbf{T}}}$ and y is in $C^{\mathcal{I}_{\mathbf{T}}}$. Note that for every S and C, the (S, C)-*neighbors* of x are contained in the potential neighbors of x and, in contrast to the latter, the (S, C)-neighbors are necessarily elements of $\Delta^{\mathcal{I}_{\mathbf{T}}}$. Moreover, we order the potential neighbors of a node x as follows: the first $b_{\mathcal{K}}$ ones are its successors, in the order they occur in $\mathbf{T}$. If there is no $a \in \mathbf{I}_{\mathcal{K}}$ such that $a^{\mathcal{I}_{\mathbf{T}}} = x$, then the $(b_{\mathcal{K}}+1)$-th potential neighbor of x is x itself, and the $(b_{\mathcal{K}}+2)$-th is its predecessor. Otherwise, the $(b_{\mathcal{K}}+1)$-th to $(2b_{\mathcal{K}})$-th potential neighbors are the level 1 nodes, in the order they appear in $\mathbf{T}$.

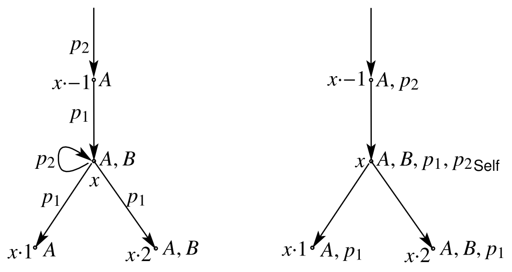

Fig. 3. A fragment of an interpretation and its tree representation.

To verify that a node x satisfies a number restriction, the automaton traverses all potential neighbors of x, and counts how many of them are actually its (S, C)-neighbors. This requires us to encode counters using the auxiliary states in Q_{num} and $Q_{\mathcal{A}\_num}$. Intuitively, in a state $\langle \gtrless nS.C, i, j\rangle$ of Q_{num}, the number i records 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 the state $\gtrless nS.C$ and visits a node x, it changes to the state $\langle \gtrless nS.C, 0, 0\rangle$ (item IV1 in Table 5) and then navigates the potential neighbors of x. The automaton navigates first the at most $k_{\mathcal{T}}$ many successors of the node, using the transitions in item IV(2)1. While doing this, it increases the counters accordingly. It will be in state $\langle \gtrless nS.C, i, j\rangle$, if exactly j among the first $i-1$ potential neighbors of x are (S, C)-neighbors of x. Each transition of this kind has two disjuncts. The first disjunct covers the case when the i-successor is *not* an (S, C)-neighbor of x, hence only the j-counter is increased. The second disjunct covers the case where the i-successor is an (S, C)-neighbor of x and increases both counters.

After all successors have been navigated, if x is a non-individual node, the automaton checks whether x is its own (S, C)-neighbor with the transitions in item IV(2)2, and finally it checks the predecessor of x using the transitions in item IV(2)3. Otherwise, the current node x is an individual node, and the automaton must navigate all individual nodes at level 1 (including x) to count the (S, C)-neighbors of the node. This is done by the transitions in items IV(2)4 and IV(2)5, which use the states in $Q_{\mathcal{A}\_num}$ of the form $\langle a, \gtrless nQ.C, i, j\rangle$. From an individual node ℓ, the automaton moves to some state $\langle a, \gtrless nQ.C, 0, j\rangle$ for some a represented by ℓ (item IV(2)4). From this state, it uses the counter i to navigate all the level 1 nodes looking for (S, C)-neighbors of ℓ, and stores in j how many of them it has found (item IV(2)5). 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 labeled $S_{\ell i}$, and i is labeled C and represents some individual b. It increases the value of i and j if this is the case, and only the value of i otherwise.

In all the transitions of item IV2, the counters i and j range over the following values:

- $0 \leqslant i < b_{\mathcal{K}}$: successors (resp., level 1 nodes) $1, 2, \ldots, i-1$ have been navigated;
- if $\gtrless$ is $\geqslant$, then $0 \leqslant j < n$: we stop counting if we reach n, as we already know that the at-least restriction is satisfied;
- otherwise, if $\gtrless$ is $\leqslant$, then $0 \leqslant j \leqslant n$: similarly, we can stop counting if we reach $n+1$, as we know that the at-most restrictions is not satisfied.

Once all the necessary nodes have been navigated, the (non-)satisfaction of the number restrictions is established with the transitions of item IV3 in Table 5.

Example 4.11. We give an example of how satisfaction of the concept $\leqslant 2p_2.B$ is verified at node x of some canonical interpretation. The relevant fragment of the interpretation is depicted on the left hand side of Fig. 3, and its tree representation on the right hand side. Here, we assume $b_{\mathcal{K}} = 2$. Note that $\leqslant 2p_2.B$ is satisfied at x, since its only (p_2, B)-neighbor is x itself.

To verify this, the automaton uses the following transitions. First, it sets the counters to zero with a transition as in item IV1 of Table 5:

$$\delta_{\mathcal{T}}\big(\leqslant 2p_2.B, \{A, B, p_1, p_{2\mathsf{Self}}\}\big) = \big(0, \langle \leqslant 2p_2.B, 0, 0\rangle\big)$$

Then, it checks whether $x\cdot1$ is a (p_2, B)-neighbor of x using a transition from item IV(2)1 in Table 5:

$$\begin{aligned}&\delta_{\mathcal{T}}\big(\langle \leqslant 2p_2.B, 0, 0\rangle, \{A, B, p_1, p_{2\mathsf{Self}}\}\big)\\ &\quad = \Big(\big((1, \neg p_2) \vee (1, \neg B)\big) \wedge \big(0, \langle \leqslant 2p_2.B, 1, 0\rangle\big)\Big) \vee \Big((1, p_2) \wedge (1, B) \wedge \big(0, \langle \leqslant 2p_2.B, 1, 1\rangle\big)\Big)\end{aligned}$$

Since this is not the case, the first disjunct, $((1, \neg p_2) \vee (1, \neg B)) \wedge (0, \langle \leqslant 2p_2.B, 1, 0\rangle)$, is chosen to satisfy the transition, and only the first counter i is increased, switching to state $\langle \leqslant 2p_2.B, 1, 0\rangle$. To satisfy the first conjunct $(1, \neg p_2) \vee (1, \neg B)$ within the chosen disjunct, the automaton can in turn choose either of its disjuncts, so it chooses the first, $(1, \neg p_2)$. That is, it moves to $x\cdot1$ at state $\neg p_2$, and checks the satisfaction of this atomic concept using a transition in item III of Table 4.

$$\delta_{\mathcal{T}}\big(\neg p_2, \{A, p_1\}\big) = \mathsf{true}$$

From state $\langle \leqslant 2p_2.B, 1, 0\rangle$ the automaton proceeds to the second successor, $x{\cdot}2$, and checks it similarly to $x{\cdot}1$:

$$\begin{aligned}&\delta_{\mathcal{T}}\big(\langle \leqslant 2p_2.B, 1, 0\rangle, \{A, B, p_1, p_{2\mathsf{Self}}\}\big)\\ &\quad = \big(\big((2, \neg p_2) \vee (2, \neg B)\big) \wedge \big(0, \langle \leqslant 2p_2.B, 2, 0\rangle\big)\big) \vee \big((2, p_2) \wedge (2, B) \wedge \big(0, \langle \leqslant 2p_2.B, 2, 1\rangle\big)\big)\end{aligned}$$

Again, since $x{\cdot}2$ is not a (p_2, B)-neighbor of x, the automaton chooses the first disjunct and switches to the state $\langle \leqslant 2p_2.B, 2, 0\rangle$. Now to satisfy the first disjunct it can choose as above state $\neg p_2$ and verify its satisfaction:

$$\delta_{\mathcal{T}}\big(\neg p_2, \{A, B, p_1\}\big) = \mathsf{true}$$

Now that the automaton has navigated all children, it proceeds with x itself. Since x is indeed its own (p_2, B)-neighbor, it chooses the second disjunct; both counters are increased, and the automaton switches to $\langle \leqslant 2p_2.B, 3, 1\rangle$. It also verifies, using the atomic transitions, that both $p_{2\mathsf{Self}}$ and B are in the label of x:

$$\begin{aligned}&\delta_{\mathcal{T}}\big(\langle \leqslant 2p_2.B, 2, 0\rangle, \{A, B, p_1, p_{2\mathsf{Self}}\}\big)\\ &\quad = \big(\big((0, \neg p_{2\mathsf{Self}}) \vee (0, \neg B)\big) \wedge \big(0, \langle \leqslant 2p_2.B, 3, 0\rangle\big)\big) \vee \big((0, p_{2\mathsf{Self}}) \wedge (0, B) \wedge \big(0, \langle \leqslant 2p_2.B, 3, 1\rangle\big)\big)\\ &\delta_{\mathcal{T}}\big(p_{2\mathsf{Self}}, \{A, B, p_1, p_{2\mathsf{Self}}\}\big) = \mathsf{true}\\ &\delta_{\mathcal{T}}\big(B, \{A, B, p_1, p_{2\mathsf{Self}}\}\big) = \mathsf{true}\end{aligned}$$

Finally, from $\langle \leqslant 2p_2.B, 3, 1\rangle$, the automaton uses a transition from item IV(2)3 in Table 5 to verify whether the parent of x is also its (p_2, B)-neighbor:

$$\begin{aligned}&\delta_{\mathcal{T}}\big(\langle \leqslant 2p_2.B, 3, 1\rangle, \{A, B, p_1, p_{2\mathsf{Self}}\}\big)\\ &\quad = \big(\big((0, \neg p_2^-) \vee (-1, \neg B)\big) \wedge \big(0, \langle \leqslant 2p_2.B, 4, 1\rangle\big)\big) \vee \big((0, p_2^-) \wedge (-1, B) \wedge \big(0, \langle a \leqslant 2p_2.B, 4, 2\rangle\big)\big)\end{aligned}$$

Since it is not, the automaton chooses the first disjunct and increases the first counter switching to $\langle \leqslant 2p_2.B, 4, 1\rangle$. It also verifies that x is not a p_2-successor of its parent by moving to the state $\neg p_2^-$; it then has a transition (as in item IV3 of Table 5) which ensures that x satisfies $\leqslant 2p_2.B$, if among the four potential neighbors of x we have only one (p_2, B)-neighbor:

$$\begin{aligned}&\delta_{\mathcal{T}}\big(\neg p_2^-, \{A, B, p_1, p_{2\mathsf{Self}}\}\big) = \mathsf{true}\\ &\delta_{\mathcal{T}}\big(\langle \leqslant 2p_2.B, 4, 1\rangle, \{A, B, p_1, p_{2\mathsf{Self}}\}\big) = \mathsf{true} \qquad \square\end{aligned}$$

Example 4.12 *(Continued).* For the same interpretation and tree representation as in the previous example, let us now verify that x satisfies the concept $\geqslant 3(p_1 \cup p_2).A$.

The automaton again starts by setting the counters to zero:

$$\delta_{\mathcal{T}}\big(\geqslant 3(p_1 \cup p_2).A, \{A, B, p_1, p_{2\mathsf{Self}}\}\big) = \big(0, \big\langle \geqslant 3(p_1 \cup p_2).A, 0, 0\big\rangle\big)$$

It moves to the first successor $x{\cdot}1$ and verifies that it is indeed a $(p_1 \cup p_2, A)$-neighbor of x, and increases both counters by one:

$$\begin{aligned}&\delta_{\mathcal{T}}\big(\big\langle \geqslant 3(p_1 \cup p_2).A, 0, 0\big\rangle, \{A, B, p_1, p_{2\mathsf{Self}}\}\big)\\ &\quad = \big(\big((1, \neg p_1 \cap \neg p_2) \vee (1, \neg A)\big) \wedge \big(0, \big\langle \geqslant 3(p_1 \cup p_2).A, 1, 0\big\rangle\big)\big)\\ &\qquad \vee \big(\big(1, (p_1 \cup p_2)\big) \wedge (1, A) \wedge \big(0, \big\langle \geqslant 3(p_1 \cup p_2).A, 1, 1\big\rangle\big)\big)\end{aligned}$$

Here, the right disjunct is chosen for satisfaction. To check that $x{\cdot}1$ is a $(p_1 \cup p_2, A)$-neighbor of x, the automaton needs to decompose the simple role $(p_1 \cup p_2)$ and choose one of its disjuncts:

$$\begin{aligned}&\delta_{\mathcal{T}}\big(p_1 \cup p_2, \{A, p_1\}\big) = (0, p_1) \vee (0, p_2)\\ &\delta_{\mathcal{T}}\big(p_1, \{A, p_1\}\big) = \mathsf{true}\\ &\delta_{\mathcal{T}}\big(A, \{A, p_1\}\big) = \mathsf{true}\end{aligned}$$

In this case, let it choose p_1. From $\langle \geqslant 3(p_1 \cup p_2).A, 1, 1\rangle$ the automaton proceeds similarly and verifies that $x{\cdot}2$ is also a $(p_1 \cup p_2, A)$-neighbor of x:

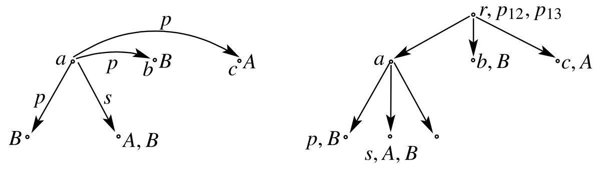

Fig. 4. Another fragment of an interpretation and its tree representation.

$$\begin{aligned}
&\delta_{\mathcal{T}}\big(\langle\geqslant 3(p_1\cup p_2).A,1,1\rangle,\{A,B,p_1,p_{2\mathsf{Self}}\}\big)\\
&\quad=\big(\big((2,\neg p_1\cap\neg p_2)\vee(2,\neg A)\big)\wedge\big(0,\langle\geqslant 3(p_1\cup p_2).A,2,1\rangle\big)\big)\\
&\qquad\vee\big(\big(2,(p_1\cup p_2)\big)\wedge(2,A)\wedge\big(0,\langle\geqslant 3(p_1\cup p_2).A,2,2\rangle\big)\big)\\
&\delta_{\mathcal{T}}\big(p_1\cup p_2,\{A,B,p_1\}\big)=(0,p_1)\vee(0,p_2)\\
&\delta_{\mathcal{T}}\big(p_1,\{A,B,p_1\}\big)=\mathsf{true}\\
&\delta_{\mathcal{T}}\big(A,\{A,B,p_1\}\big)=\mathsf{true}
\end{aligned}$$

As above, the right disjunct is chosen for satisfaction. From $\langle\geqslant 3(p_1\cup p_2).A,2,2\rangle$ the automaton verifies that x is its own $(p_1\cup p_2,A)$-neighbor using item IV(2)2 in Table 5:

$$\begin{aligned}
&\delta_{\mathcal{T}}\big(\langle\geqslant 3(p_1\cup p_2).A,2,2\rangle,\{A,B,p_1,p_{2\mathsf{Self}}\}\big)\\
&\quad=\big(\big(\big(0,\neg(p_1\cup p_2)_{\mathsf{Self}}\big)\vee(0,\neg A)\big)\wedge\big(0,\langle\geqslant 3(p_1\cup p_2).A,3,2\rangle\big)\big)\\
&\qquad\vee\big(\big(0,(p_1\cup p_2)_{\mathsf{Self}}\big)\wedge(0,A)\wedge\big(0,\langle\geqslant 3(p_1\cup p_2).A,3,3\rangle\big)\big)
\end{aligned}$$

That is, the right disjunct is chosen and the automaton verifies that x satisfies $(p_1\cup p_2)_{\mathsf{Self}}$ using role decomposition and atomic checks:

$$\begin{aligned}
&\delta_{\mathcal{T}}\big((p_1\cup p_2)_{\mathsf{Self}},\{A,B,p_1,p_{2\mathsf{Self}}\}\big)=(0,p_{1\mathsf{Self}})\vee(0,p_{2\mathsf{Self}})\\
&\delta_{\mathcal{T}}\big(p_{2\mathsf{Self}},\{A,B,p_1,p_{2\mathsf{Self}}\}\big)=\mathsf{true}\\
&\delta_{\mathcal{T}}\big(A,\{A,B,p_1,p_{2\mathsf{Self}}\}\big)=\mathsf{true}
\end{aligned}$$

Finally, it checks the number restriction $\langle\geqslant 3(p_1\cup p_2).A,3,3\rangle$ using a transition of item IV3 in Table 5:

$$\delta_{\mathcal{T}}\big(\langle\geqslant 3(p_1\cup p_2).A,3,3\rangle,\{A,B,p_1,p_{2\mathsf{Self}}\}\big)=\mathsf{true}$$

This suffices to establish that x satisfies $\geqslant 3(p_1\cup p_2).A$. Note that in this case, there is no need to navigate the parent of x (item IV(2)3 in Table 5). □

Example 4.13. Finally, to illustrate the way the automaton $\mathbf{A}_{\mathcal{T}}$ verifies the satisfaction of universal restrictions by ABox individuals, we consider the partial interpretation depicted on the left hand side of Fig. 4. Its tree representation, assuming $b_{\mathcal{K}}=3$ and $\mathbf{I}_{\mathcal{K}}=\{a,b,c\}$, is given on the right hand side.

To verify whether the concept $\forall p.\neg A$ is satisfied at the node 1 interpreting a (that is, when the automaton is in node 1 and state $\forall p.\neg A$), it moves to the state $\leqslant 0p.A$ using the corresponding transition in item II of Table 4. Then, it proceeds similarly to the examples above. First it sets the counters to zero:

$$\delta_{\mathcal{T}}\big(\leqslant 0p.A,\{a\}\big)=\big(0,\langle\leqslant 0p.A,0,0\rangle\big)$$

Then it successively checks that none of 1·1, 1·2, and 1·3 is a (p,A)-neighbor of 1. It always chooses the first disjunct of the corresponding transition and increases only the first counter:

$$\begin{aligned}
&\delta_{\mathcal{T}}\big(\langle\leqslant 0p.A,0,0\rangle,\{a\}\big)=\big(\big((1,\neg p)\vee(1,\neg A)\big)\wedge\big(0,\langle\leqslant 0p.A,1,0\rangle\big)\big)\vee\big((1,p)\wedge(1,A)\wedge\big(0,\langle\leqslant 0p.A,1,1\rangle\big)\big)\\
&\delta_{\mathcal{T}}\big(\neg A,\{p,B\}\big)=\mathsf{true}\\
&\delta_{\mathcal{T}}\big(\langle\leqslant 0p.A,1,0\rangle,\{a\}\big)=\big(\big((2,\neg p)\vee(2,\neg A)\big)\wedge\big(0,\langle\leqslant 0p.A,2,0\rangle\big)\big)\vee\big((2,p)\wedge(2,A)\wedge\big(0,\langle\leqslant 0p.A,2,1\rangle\big)\big)\\
&\delta_{\mathcal{T}}\big(\neg p,\{s,A,B\}\big)=\mathsf{true}\\
&\delta_{\mathcal{T}}\big(\langle\leqslant 0p.A,2,0\rangle,\{a\}\big)=\big(\big((3,\neg p)\vee(3,\neg A)\big)\wedge\big(0,\langle\leqslant 0p.A,3,0\rangle\big)\big)\vee\big((3,p)\wedge(3,A)\wedge\big(0,\langle\leqslant 0p.A,3,1\rangle\big)\big)\\
&\delta_{\mathcal{T}}\big(\neg p,\{\}\big)=\mathsf{true}
\end{aligned}$$

Since $L(1) \cap \mathbf{I}_{\mathcal{K}} = \{a\}$, from $\langle \leqslant 0p.A, 3, 0\rangle$, the automaton uses a transition as in item IV(2)4 of Table 5 to move to the root and to the state $\langle a, \leqslant 0p.A, 0, 0\rangle$, resetting the first counter to 0:

$$\delta_{\mathcal{T}}\big(\langle \leqslant 0p.A, 3, 0\rangle, \{a\}\big) = \big(-1, \langle a, \leqslant 0p.A, 0, 0\rangle\big)$$

From here, it verifies whether each of the nodes 1 to 3 is a (p, A)-neighbor of the node labeled a, using transitions as in item IV(2)5 of Table 5. For node 1, the transition is as follows:

$$\begin{aligned}&\delta_{\mathcal{T}}\big(\langle a, \leqslant 0p.A, 0, 0\rangle, \{r, p_{12}, p_{13}\}\big)\\ &\quad = \Big(\Big(\Big(\big((0, \sim p_{11}) \vee (1, \neg a)\big) \wedge \big((0, \sim p_{21}) \vee (2, \neg a)\big) \wedge \big((0, \sim p_{31}) \vee (3, \neg a)\big)\Big) \vee (1, \neg A) \vee \big((1, \neg a) \wedge (1, \neg b)\\ &\qquad \wedge (1, \neg c)\big)\Big) \wedge \big(0, \langle a, \leqslant 0p.A, 1, 0\rangle\big)\Big) \vee \Big(\Big(\big((0, p_{11}) \wedge (1, a)\big) \vee \big((0, p_{21}) \wedge (2, a)\big) \vee \big((0, p_{31}) \wedge (3, a)\big) \wedge (1, A)\\ &\qquad \wedge \big((1, a) \vee (1, b) \vee (1, c)\big) \wedge \big(0, \langle a, \leqslant 0p.A, 1, 1\rangle\big)\Big)\end{aligned}$$

A is not in the label of 1, and this is enough to ensure that 1 is not a (p, A)-neighbor of any node. Hence the automaton needs to choose the first big disjunct, and satisfy it by moving to 1 in state $\neg A$. It increases only the first counter to stay at the root in state $\langle a, \leqslant 0p.A, 1, 0\rangle$, from which it verifies node 2:

$$\begin{aligned}&\delta_{\mathcal{T}}\big(\neg A, \{a\}\big) = \text{true}\\ &\delta_{\mathcal{T}}\big(\langle a, \leqslant 0p.A, 1, 0\rangle, \{r, p_{12}, p_{13}\}\big)\\ &\quad = \Big(\Big(\Big(\big((0, \sim p_{12}) \vee (1, \neg a)\big) \wedge \big((0, \sim p_{22}) \vee (2, \neg a)\big) \wedge \big((0, \sim p_{32}) \vee (3, \neg a)\big)\Big) \vee (2, \neg A)\\ &\qquad \vee \big((2, \neg a) \wedge (2, \neg b) \wedge (2, \neg c)\big)\Big) \wedge \big(0, \langle a, \leqslant 0p.A, 2, 0\rangle\big)\Big) \vee \Big(\Big(\big((0, p_{12}) \wedge (1, a)\big) \vee \big((0, p_{22}) \wedge (2, a)\big)\\ &\qquad \vee \big((0, p_{32}) \wedge (3, a)\big)\Big) \wedge (2, A) \wedge \big((2, a) \vee (2, b) \vee (2, c)\big) \wedge \big(0, \langle a, \leqslant 0p.A, 2, 1\rangle\big)\Big)\end{aligned}$$

Again, as 2 is not a (p, A)-neighbor of any node, the automaton must choose to move to 2 in state $\neg A$ and to increase only the first counter:

$$\begin{aligned}&\delta_{\mathcal{T}}\big(\neg A, \{b, B\}\big) = \text{true}\\ &\delta_{\mathcal{T}}\big(\langle a, \leqslant 0p.A, 2, 0\rangle, \{r, p_{12}, p_{13}\}\big)\\ &\quad = \Big(\Big(\Big(\big((0, \sim p_{13}) \vee (1, \neg a)\big) \wedge \big((0, \sim p_{23}) \vee (2, \neg a)\big) \wedge \big((0, \sim p_{33}) \vee (3, \neg a)\big)\Big) \vee (3, \neg A)\\ &\qquad \vee \big((3, \neg a) \wedge (3, \neg b) \wedge (3, \neg c)\big)\Big) \wedge \big(0, \langle a, \leqslant 0p.A, 3, 0\rangle\big)\Big) \vee \Big(\Big(\big((0, p_{13}) \wedge (1, a)\big) \vee \big((0, p_{23}) \wedge (2, a)\big)\\ &\qquad \vee \big((0, p_{33}) \wedge (3, a)\big)\Big) \wedge (3, A) \wedge \big((3, a) \vee (3, b) \vee (3, c)\big) \wedge \big(0, \langle a, \leqslant 0p.A, 3, 1\rangle\big)\Big)\end{aligned}$$

Now the automaton can not choose the first disjunct: it cannot satisfy the first part as p_{13} is in the label of the root and a is in the label of node 1. It cannot move to node 3 in state $\neg A$ either, as its label contains A, and it cannot satisfy the third part as c is in the label of node 3. If it chooses the second disjunct, it needs to move to state p_{13}, to state a in node 1, and to states A and c in node 3. But then it would have to increase both counters; by item IV3 in Table 5, all transitions from state $\langle a, \leqslant 0p.A, 3, 1\rangle$ are false:

$$\begin{aligned}&\delta_{\mathcal{T}}\big(p_{13}, \{r, p_{12}, p_{13}\}\big) = \text{true}\\ &\delta_{\mathcal{T}}\big(a, \{a\}\big) = \text{true}\\ &\delta_{\mathcal{T}}\big(A, \{c, A\}\big) = \text{true}\\ &\delta_{\mathcal{T}}\big(\langle a, \leqslant 0p.A, 3, 1\rangle, \{r, p_{12}, p_{13}\}\big) = \text{false}\end{aligned}$$

This shows that the number restriction $\leqslant 0p.A$, and hence $\forall p.\neg A$, is not satisfied at node 1, as node 3 is indeed a (p, A)-neighbor of 1. □

As we have mentioned, infinite paths in the runs of $\mathbf{A}_{\mathcal{T}}$ only arise from the transitions that move from states of the form $\forall R^*.C$ and $\exists R^*.C$ to states of the form $\forall R.\forall R^*.C$ and $\exists R.\exists R^*.C$, and then move to all or to some successors of the current node in the preceding state, that is, to states $\forall R^*.C$ and $\exists R^*.C$, respectively. Hence, in every infinite path of a run, a state of either (i) the form $\forall R^*.C$ or (ii) the form $\exists R^*.C$ occurs infinitely often. Case (i) amounts to the existence of an infinite sequence of R-neighbors in the represented interpretation, over which the satisfaction of the concept $\forall R^*.C$ is correctly ensured. Paths of case (ii) are also caused by infinite sequences of R-neighbors in interpretations, but do not correctly ensure the satisfaction of the concept $\exists R^*.C$. To satisfy $\exists R^*.C$ at some node x, we need to ensure that on every infinite sequence of R-neighbors starting from x, an instance of C is eventually reached. Hence we disallow paths where $\exists R^*.C$ occurs infinitely often, requiring that the disjunct C is eventually chosen and the satisfaction of C is not indefinitely postponed; this is achieved by the acceptance condition (cf. [35]).

Summing up, the core property of the automaton $\mathbf{A}_{\mathcal{T}}$ is that it correctly verifies the satisfaction of the concepts in the closure of $C_{\mathcal{T}}$: an accepting run of $\mathbf{A}_{\mathcal{T}}$ over an interpretation tree $\mathbf{T}$ can have a node labeled (x, C) iff x is an instance of C in the interpretation $\mathcal{I}_{\mathbf{T}}$. To formalize this argument, recall that an (x, s)-run is just like a (full) run, but the root must be labeled with a given pair of an arbitrary node x of the input tree and a state s (see Definition 2.8). We now show:

Lemma 4.14. *Let $\mathbf{T} = (T, L)$ be an interpretation tree for $\mathcal{K}$ and let $x \in \Delta^{\mathcal{I}_{\mathbf{T}}}$. Furthermore, let $C \in \mathsf{Q}(\mathbf{A}_{\mathcal{T}})$ be a concept in $Cl(C_{\mathcal{T}})$. Then there is an accepting (x, C)-run of $\mathbf{A}_{\mathcal{T}}$ over $\mathbf{T}$ iff $x \in C^{\mathcal{I}_{\mathbf{T}}}$.*

Proof (Sketch). We give only a short sketch here; a more detailed proof can be found in Appendix A. We first show similar properties for simple roles and for the states in $\mathsf{Q}_{\mathsf{Self}}$ and $\mathsf{Q}_{\mathcal{A}\_role}$. In the following claims, $\mathbf{T} = (T, L)$ is an interpretation tree for $\mathcal{K}$, as above.

(C1) Let $x \cdot i \in \Delta^{\mathcal{I}_{\mathbf{T}}}$ and let S be a simple role in $Cl(C_{\mathcal{T}})$. Then there is an accepting $(x \cdot i, S)$-run of $\mathbf{A}_{\mathcal{T}}$ over $\mathbf{T}$ iff $(x, x \cdot i) \in S^{\mathcal{I}_{\mathbf{T}}}$.
(C2) Let $x \in \Delta^{\mathcal{I}_{\mathbf{T}}}$ and let $S_{\mathsf{Self}} \in \mathsf{Q}_{\mathsf{Self}}$. Then there is an accepting (x, S)-run of $\mathbf{A}_{\mathcal{T}}$ over $\mathbf{T}$ iff $(x, x) \in S^{\mathcal{I}_{\mathbf{T}}}$.
(C3) Let $i, j \in \Delta^{\mathcal{I}_{\mathbf{T}}}$ and $S_{ij} \in \mathsf{Q}_{\mathcal{A}\_role}$. Then there is an accepting (ε, S_{ij})-run of $\mathbf{A}_{\mathcal{T}}$ over $\mathbf{T}$ iff $(i, j) \in S^{\mathcal{I}_{\mathbf{T}}}$.

Each of these claims is shown by a straightforward structural induction on the role S. Then, using (C1)–(C3), one can show Lemma 4.14 by structural induction on C. The induction is rather tedious, due to the large number of constructors of $\mathcal{ZIQ}$. Most cases are straightforward, since the transitions in the left column of item II in Table 4 decompose complex concepts correctly, according to the semantics of the different constructors. The only interesting cases are (i) $C = \leqslant n S.D$, (ii) $C = \geqslant n S.D$, (iii) $C = \forall R^*.D$, and (iv) $C = \exists R^*.D$. For (i) and (ii), we rely on the following auxiliary claims, where we use the notions of *potential neighbors* and (S, D)-*neighbors* defined above:

(C4) Let $x \in \Delta^{\mathcal{I}_{\mathbf{T}}}$ and let $\langle \geqslant n S.D, i, j\rangle \in \mathsf{Q}_{num}$ (resp., $\langle \leqslant n S.D, i, j\rangle \in \mathsf{Q}_{num}$). Then there is an accepting $(x, \langle \geqslant n S.D, i, j\rangle)$-run (resp., $(x, \langle \leqslant n S.D, i, j\rangle)$-run) of $\mathbf{A}_{\mathcal{T}}$ over $\mathbf{T}$ iff there are at least (resp., at most) $n - j$ many (S, D)-neighbors of x among its potential neighbors beyond the i-th one.
(C5) Let $\langle a, \geqslant n S.D, i, j\rangle \in \mathsf{Q}_{\mathcal{A}\_num}$ (resp., $\langle a, \leqslant n S.D, i, j\rangle \in \mathsf{Q}_{\mathcal{A}\_num}$). Then there is an accepting $(\varepsilon, \langle a, \gtrless n S.D, i, j\rangle)$-run of $\mathbf{A}_{\mathcal{T}}$ over $\mathbf{T}$ iff there are at least (resp., at most) $n - j$ many (S, D)-neighbors of $a^{\mathcal{I}_{\mathbf{T}}}$ among its potential neighbors beyond the $(b_{\mathcal{K}} + i)$-th.

The interesting feature of (iii) and (iv) is that a run can repeatedly generate successors labeled with (x', C). As discussed above, this is handled in the usual way by the termination condition. If $C = \forall R^*.D$ is satisfied, the generation of nodes (x', C) 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. □

Now we can easily show that $\mathcal{L}(\mathbf{A}_{\mathcal{T}})$ accepts exactly the interpretation trees that represent a model of $\mathcal{T}$.

Lemma 4.15. *If $\mathbf{T}$ is an interpretation tree for $\mathcal{K}$, then $\mathbf{T} \in \mathcal{L}(\mathbf{A}_{\mathcal{T}})$ iff $\mathcal{I}_{\mathbf{T}} \models \mathcal{T}$.*

Proof (Sketch). Since $\mathbf{T}$ is an interpretation tree for $\mathcal{K}$, $\mathcal{I}_{\mathbf{T}}$ is a $b_{\mathcal{K}}$-canonical interpretation. By the transitions in item I of Table 4, $\mathbf{A}_{\mathcal{T}}$ first moves to each node $i \in Roots(\mathcal{I}_{\mathbf{T}})$ in state $C_{\mathcal{T}}$. Then, by Lemma 4.14, it will succeed in completely decomposing $C_{\mathcal{T}}$ at i iff $i \in C_{\mathcal{T}}^{\mathcal{I}_{\mathbf{T}}}$. Hence, $\mathbf{A}_{\mathcal{T}}$ has an accepting run on $\mathbf{T}$ (i.e., $\mathbf{T} \in \mathcal{L}(\mathbf{A}_{\mathcal{T}})$) iff $Roots(\mathcal{I}_{\mathbf{T}}) \subseteq C_{\mathcal{T}}^{\mathcal{I}_{\mathbf{T}}}$, and it follows from Proposition 3.8 that $\mathbf{T} \in \mathcal{L}(\mathbf{A}_{\mathcal{T}})$ 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 4.16. Given $\mathcal{K}$, let $\mathbf{A}_{\mathcal{K}}$ be a 2ATA 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 Proposition 2.10. □

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 4.17. *For a given normalized $\mathcal{ZIQ}$ KB $\mathcal{K}$, $\mathcal{L}(\mathbf{A}_{\mathcal{K}}) = \{\mathbf{T} \mid \mathbf{T}$ is an interpretation tree and $\mathcal{I}_{\mathbf{T}} \models \mathcal{K}\}$.*

Proof. ($\subseteq$). 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 Lemmas 4.5 and 4.7 $\mathbf{T}$ is an interpretation tree for $\mathcal{K}$ and $\mathcal{I}_{\mathbf{T}}$ is a canonical interpretation for $\mathcal{K}$. Furthermore, by Lemma 4.9, $\mathbf{T} \in \mathcal{L}(\mathbf{A}_{\mathcal{A}})$ implies $\mathcal{I}_{\mathbf{T}} \models \mathcal{A}$, and by Lemma 4.15, $\mathbf{T} \in \mathcal{L}(\mathbf{A}_{\mathcal{T}})$ implies $\mathcal{I}_{\mathbf{T}} \models \mathcal{T}$. Consequently, $\mathcal{I}_{\mathbf{T}}$ is a canonical model of $\mathcal{K}$.

($\supseteq$). Let $\mathbf{T}$ be an interpretation tree such that $\mathcal{I}_{\mathbf{T}}$ is a model of $\mathcal{K}$. By Lemma 4.7, $\mathbf{A}_{\mathcal{I}}$ accepts $\mathbf{T}$. As $\mathcal{I}_{\mathbf{T}} \models \mathcal{A}$, Lemma 4.9 implies that $\mathbf{A}_{\mathcal{A}}$ accepts $\mathbf{T}$, and as $\mathcal{I}_{\mathbf{T}} \models \mathcal{T}$, Lemma 4.15 implies that $\mathbf{A}_{\mathcal{T}}$ accepts $\mathbf{T}$. Consequently, $\mathbf{T} \in \mathcal{L}(\mathbf{A}_{\mathcal{I}}) \cap \mathcal{L}(\mathbf{A}_{\mathcal{A}}) \cap \mathcal{L}(\mathbf{A}_{\mathcal{T}}) = \mathcal{L}(\mathbf{A}_{\mathcal{K}})$. □

From Proposition 4.17 and the canonical model property of $\mathcal{ZIQ}$ in Theorem 3.10, we obtain:

Theorem 4.18. *A normalized $\mathcal{ZIQ}$ KB $\mathcal{K}$ is satisfiable iff $\mathcal{L}(\mathbf{A}_{\mathcal{K}}) \neq \emptyset$.*

Proof. If $\mathcal{L}(\mathbf{A}_{\mathcal{K}}) \neq \emptyset$, then by Proposition 4.17 $\mathcal{K}$ has some model, hence $\mathcal{K}$ is satisfiable. Conversely, if $\mathcal{K}$ is satisfiable, then by Theorem 3.10 it has some canonical model $\mathcal{I}$, and as $\mathcal{I} = \mathcal{I}_{\mathbf{T}_{\mathcal{I}}}$ (by Lemma 4.5), by Proposition 4.17 $\mathbf{T}_{\mathcal{I}} \in \mathcal{L}(\mathbf{A}_{\mathcal{K}})$, hence $\mathcal{L}(\mathbf{A}_{\mathcal{K}}) \neq \emptyset$. □

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

4.4. Complexity

By $\|\mathcal{K}\|$ we denote the size of a (string) representation of $\mathcal{K}$; we assume here *unary number encoding*, i.e., the numbers n in number restrictions are encoded by a string of length $\Theta(n)$. Recall that $\mathbf{C}_{\mathcal{K}}$ and $\bar{\mathbf{R}}_{\mathcal{K}}$ denote the atomic concepts and roles, respectively, that occur in $\mathcal{K}$, and $\mathbf{I}_{\mathcal{K}}$ denotes the ABox individuals; $b_{\mathcal{K}}$ denotes $\max(k_{\mathcal{T}}, |\mathbf{I}_{\mathcal{K}}|)$ where $k_{\mathcal{T}} = |Cl(C_{\mathcal{T}})| \cdot \max(\{n \mid \geqslant n S.C \in Cl(C_{\mathcal{T}})\} \cup \{0\})$. Furthermore, let $n'_{\max} = \max(\{n \mid \gtrless n S.C \in Cl(C_{\mathcal{T}})\} \cup \{0\})$. Note that $|\mathbf{C}_{\mathcal{K}}|$, $|\bar{\mathbf{R}}_{\mathcal{K}}|$, $|\mathbf{I}_{\mathcal{K}}|$, and $|Cl(C_{\mathcal{T}})|$ are linear in $\|\mathcal{K}\|$. Under unary number encoding, this holds also for $n'_{\max}$, and $b_{\mathcal{K}}$ is quadratic in $\|\mathcal{K}\|$. We thus obtain:

Lemma 4.19. *For $\mathbf{A}_{\mathcal{K}}$, we have $|\Sigma(\mathbf{A}_{\mathcal{K}})| \leqslant 2^{O(\|\mathcal{K}\|^5)}$, $|Q(\mathbf{A}_{\mathcal{K}})| \leqslant O(\|\mathcal{K}\|^5)$, 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 $M(\mathcal{K}) = |\mathbf{C}_{\mathcal{K}}| + |\bar{\mathbf{R}}_{\mathcal{K}}| + |\mathbf{I}_{\mathcal{K}}| + |PI| + |PS| + 1$, and we have $|PI| = |\bar{\mathbf{R}}_{\mathcal{K}}| \cdot |b_{\mathcal{K}}|^2$ and $|PS| = |\bar{\mathbf{R}}_{\mathcal{K}}|$. Clearly, $M(\mathcal{K}) = O(\|\mathcal{K}\|^5)$.
- $|Q_{\mathcal{K}}| \leqslant |Q_{\mathcal{T}}| + |Q_{\mathcal{A}}| + |Q_{\mathcal{I}}| + 1$, where the following bounds hold:

$$|Q_{\mathcal{A}}| \leqslant 1 + 2|\mathbf{I}_{\mathcal{K}}| + |\mathbf{C}_{\mathcal{A}}| + |\bar{\mathbf{R}}_{\mathcal{A}}| \cdot |\mathbf{I}_{\mathcal{K}}|^2$$
$$|Q_{\mathcal{I}}| \leqslant 2 + 2(|\mathbf{I}_{\mathcal{K}}| + 1)$$
$$|Q_{\mathcal{T}}| \leqslant 1 + |Cl_{ext}| + |Q_{\mathsf{Self}}| + |Q_{\mathcal{A}\_role}| + |Q_{num}| + |Q_{\mathcal{A}\_num}|$$
$$|Cl_{ext}| \leqslant |Cl(C_{\mathcal{T}})| + 2|\mathbf{I}_{\mathcal{K}}|$$
$$|Q_{\mathsf{Self}}| \leqslant |Cl(C_{\mathcal{T}})| + |\mathbf{I}_{\mathcal{K}}| \cdot |Cl(C_{\mathcal{T}})|$$
$$|Q_{\mathcal{A}\_role}| \leqslant |Cl(C_{\mathcal{T}})| \cdot |b_{\mathcal{K}}|^2$$
$$|Q_{num}| \leqslant |Cl(C_{\mathcal{T}})| \cdot (b_{\mathcal{K}} + 1) \cdot \left(n'_{\max} + 1\right)$$
$$|Q_{\mathcal{A}\_num}| \leqslant |\mathbf{I}_{\mathcal{K}}| \cdot |Cl(C_{\mathcal{T}})| \cdot b_{\mathcal{K}} \cdot \left(n'_{\max} + 1\right)$$

  Hence, it is easy to see that $|Q_{\mathcal{K}}| = O(\|K\|^5)$ (cf. $|Q_{\mathcal{A}\_num}|$).
- $\mathsf{ind}(\mathbf{A}) = \max(\mathsf{ind}(\mathbf{A}_{\mathcal{I}}), \mathsf{ind}(\mathbf{A}_{\mathcal{A}}), \mathsf{ind}(\mathbf{A}_{\mathcal{T}})) = \max(2, 2, 3) = 3$. □

Thus, by Theorems 2.9 and 4.18, we get an optimal upper bound for KB satisfiability.

Corollary 4.20. *Deciding whether a given KB in $\mathcal{ZIQ}$ is satisfiable is in* ExpTime *under unary number encoding.*

This is worst-case optimal, since a matching hardness result holds already for much weaker DLs, e.g., $\mathcal{ALC}$ [2].

5. Query answering via automata

We now turn to entailment of P2RPQs in KBs. As follows from Theorem 3.10, to decide whether $\mathcal{K} \models q$ for a normalized P2RPQ q and a normalized 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 speaking, $\mathbf{A}_{\mathcal{K} \not\models q}$ 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 is constructed in this section and accepts the trees representing

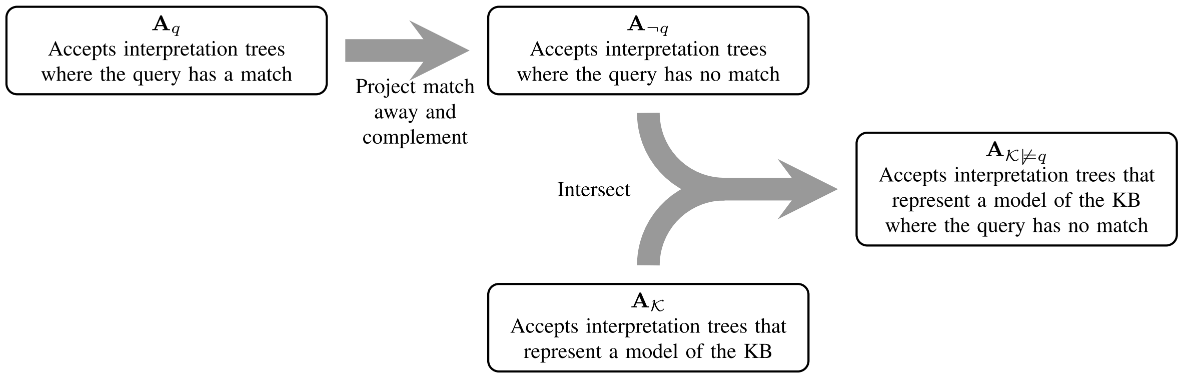
$\mathbf{A}_q$
Accepts interpretation trees where the query has a match
Project match away and complement
$\mathbf{A}_{\neg q}$
Accepts interpretation trees where the query has no match
Intersect
$\mathbf{A}_{\mathcal{K}\not\models q}$
Accepts interpretation trees that represent a model of the KB where the query has no match
$\mathbf{A}_{\mathcal{K}}$
Accepts interpretation trees that represent a model of the KB

Fig. 5. Overview of the automata algorithm for query entailment.

an interpretation that admits no match for q. We first construct an automaton $\mathbf{A}_q$ that accepts a tree $\mathcal{T}$ iff q has a match in the interpretation $\mathcal{I}_{\mathbf{T}}$; we then show how to obtain from $\mathbf{A}_q$ the desired $\mathbf{A}_{\neg q}$. Fig. 5 gives a general overview of the query answering technique; each of the steps will be discussed in detail below. More information about the construction of the automata in Fig. 5 is summarized in Tables 6 and 8.

All automata we have constructed in Section 4 run over $b_{\mathcal{K}}$-ary trees, where $b_{\mathcal{K}} = \max(k_{\mathcal{T}}, |\mathbf{I}_{\mathcal{K}}|)$ for $k_{\mathcal{T}} = br(C_{\mathcal{T}})$. By Theorem 3.10, we know that to decide query entailment it suffices to consider the $k_{\mathcal{T},q}$-canonical models of $\mathcal{K}$, for $k_{\mathcal{T},q} = br(\{C_{\mathcal{T}}\} \cup \mathcal{D}_q)$ as in Definition 3.9. To be able to use the same automata constructions for query answering, in this section we assume that $k_{\mathcal{T},q} \leqslant b_{\mathcal{K}}$. Note that this is trivially true for queries in which complex concepts are disallowed. It also holds if the numbers n in number restrictions $\geqslant nS.C$ and $\leqslant nS.C$ occurring in q are not greater than those in the query (and in the absence of number restrictions, if $\mathcal{K}$ contains some existential or universal concept whenever q does). For an arbitrary q and $\mathcal{K}$, we can ensure this condition by simply adding $C \sqsubseteq C$ to the TBox of $\mathcal{K}$ for every concept C that occurs in q and contains a number restriction, or a universal or existential concept.

5.1. Representing query matches

Prior to defining $\mathbf{A}_q$, we extend the tree representation of interpretations to include also query matches. In what follows, we assume a given normalized P2RPQ $q = \exists\vec{v}.\varphi(\vec{v})$. We assume w.l.o.g. that in q each atom $C(v)$ has been equivalently replaced with $id(C)(v, v)$, so that all atoms in $At(q)$ are of the form $R(v, v')$, where R is in NNF. Let $\mathbf{V}_q = \{v_1, \ldots, v_\ell\}$ be the variables in $\vec{v}$. We denote by $\mathbf{C}_q$, $\mathbf{R}_q$, and $\mathbf{I}_q$ the sets of atomic concepts, role names, and individuals, respectively, that occur in q, and define $\bar{\mathbf{R}}_q = \mathbf{R}_q \cup \{p^- \mid p \in \mathbf{R}_q\}$.

We need the following notion, combining an interpretation $\mathcal{I}$ with a *possible* match π for q.

Definition 5.1. An *extended* (*canonical*) *interpretation* is a pair $\mathcal{J} = (\mathcal{I}, \pi)$ consisting of a (canonical) interpretation $\mathcal{I}$ and a total function $\pi : \mathbf{V}_q \cup \mathbf{I}_q \to \Delta^{\mathcal{I}}$ such that $\pi(a) = a^{\mathcal{I}}$ for each $a \in \mathbf{I}_q$. If $\mathcal{I}, \pi \models q$, we write $\mathcal{J} \models q$. □

Example 5.2. Recall the interpretation $\mathcal{I}_g$ given in Example 4.3, and the mapping $\pi(v_1) = \text{Zeus}^{\mathcal{I}_g}$, $\pi(v_2) = \text{Alcmene}^{\mathcal{I}_g}$ and $\pi(v_3) = \text{Heracles}^{\mathcal{I}_g}$ for the query q_g in Example 2.5; the pair $(\mathcal{I}_g, \pi)$ is an extended interpretation. □

Extended canonical interpretations are represented by *extended interpretation trees* that are labeled using an alphabet $\Sigma_{\mathcal{K},q}$. The latter enriches $\Sigma_{\mathcal{K}}$ with the variables in $\mathbf{V}_q$ and for each $v \in \mathbf{V}_q$ allows us to include the symbol v in the label of the node $\pi(v)$. 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 π is actually a match for $\mathcal{I}$ and q, i.e., $\mathcal{I}, \pi \models q$.

Definition 5.3. 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:

(e1) Its $\Sigma_{\mathcal{K}}$-restriction $\mathbf{T}|_{\Sigma_{\mathcal{K}}} = (T, L|_{\Sigma_{\mathcal{K}}})$, where $L|_{\Sigma_{\mathcal{K}}}(x) = L(x) \cap \Sigma_{\mathcal{K}}$ for each $x \in T$, is an interpretation tree. Abusing notation, we use $\mathcal{I}_{\mathbf{T}}$ to denote $\mathcal{I}_{\mathbf{T}|_{\Sigma_{\mathcal{K}}}}$.
(e2) for each $v \in \mathbf{V}_q$ there is exactly one $x \in T$ with $v \in L(x)$, called the *candidate match for* v; furthermore, x is such that $x \in \Delta^{\mathcal{I}_{\mathbf{T}}}$ (i.e., it is connected to an individual node via the roles in $\bar{\mathbf{R}}_{\mathcal{K}}$).

We let $\pi_{\mathbf{T}} : \mathbf{V}_q \cup \mathbf{I}_q \to T$ be the function that maps each $v \in \mathbf{V}_q$ to its candidate match, and each $a \in \mathbf{I}_q$ to the (unique) node $i \in T$ with $a \in L(i)$. □

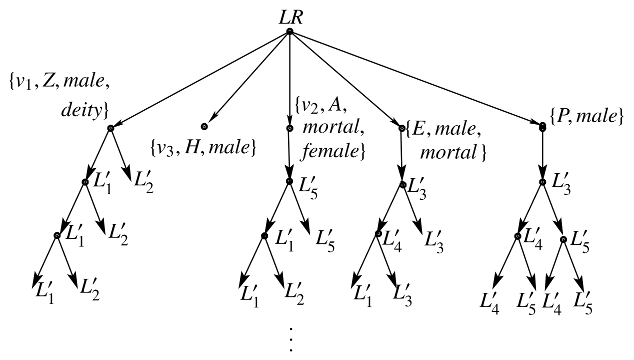

Fig. 6. The tree representation of the extended interpretation $(\mathcal{I}_g, \pi)$ (cf. Fig. 2).

Table 6
Automata for finding query matches.

Automaton	Construction	Language
$\mathbf{A}'_{\mathcal{I}}$	Definition 5.7 adaption of $\mathbf{A}_{\mathcal{I}}$	Trees whose $\Sigma_{\mathcal{K}}$ restriction represents an interpretation $\mathcal{I}$
$\mathbf{A}_V$	Definition 5.7	Trees in which each query variable has one candidate match
$\mathbf{A}_{\mathbf{T}}$	Definition 5.7 intersection of $\mathbf{A}_V$ and $\mathbf{A}'_{\mathcal{I}}$	Trees representing an extended interpretation $(\mathcal{I}, \pi)$
$\mathbf{A}_\pi$	Definition 5.12	Trees such that if they represent an extended interpretation $(\mathcal{I}, \pi)$, then π is a match for q in $\mathcal{I}$
$\mathbf{A}_q$	Definition 5.12, intersection of $\mathbf{A}_{\mathbf{T}}$ and $\mathbf{A}_\pi$	Trees representing an interpretation $\mathcal{I}$ together with a match π for q

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

Definition 5.4. 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 canonical interpretation $\mathcal{J} = (\mathcal{I}, \pi)$ is the extended interpretation tree $\mathbf{T}_{\mathcal{J}} = (T, L)$ 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 $x = \pi(v)$. □

Analogously to Lemma 4.5, we have:

Lemma 5.5. *If* $\mathbf{T}$ *is an extended interpretation tree, then* $\mathcal{J}_{\mathbf{T}}$ *is an extended canonical interpretation.*

Proof. If $\mathbf{T}$ is an extended interpretation tree, then by condition (e1) in Definition 5.3, $\mathbf{T}|_{\Sigma_{\mathcal{K}}}$ is an interpretation tree and by Lemma 4.5, $\mathcal{I}_{\mathbf{T}}$ is an extended interpretation. By condition (t2) in Definition 4.1, for each individual $a \in \mathbf{I}_{\mathcal{K}}$ there is exactly one $i \in \mathbf{T}|_{\Sigma_{\mathcal{K}}}$ such that $a \in L(i)$. Furthermore, $i \in \Delta^{\mathcal{I}_{\mathbf{T}}}$ and $a^{\mathcal{I}_{\mathbf{T}}} = i$. Similarly, for each $v \in \mathbf{V}_q$, by condition (e2) in Definition 5.3, there is exactly one $x \in \mathbf{T}|_{\Sigma_{\mathcal{K}}}$ such that $x \in \Delta^{\mathcal{I}_{\mathbf{T}}}$ and $v \in L(x)$. Therefore the mapping $\pi_{\mathbf{T}}$ is well-defined and it defines a total function from $\mathbf{V}_q \cup \mathbf{I}_q$ to $\Delta^{\mathcal{I}_{\mathbf{T}}}$, with $\pi(a) = a^{\mathcal{I}_{\mathbf{T}}}$. This proves that $\mathcal{J}_{\mathbf{T}}$ is an extended canonical interpretation. □

Example 5.6. The tree representation of the extended interpretation $(\mathcal{I}_g, \pi)$ (see Examples 5.2 and 4.3) is shown in Fig. 6. It extends the tree representation of Fig. 2 with the variables $\mathbf{V}_q = \{v_1, v_2, v_3\}$. □

5.2. Constructing the automaton that checks query matches

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; they are summarized in Table 6. All these automata are 2ATAs that run over $b_{\mathcal{K}}$-ary trees. As seen later, their size (in terms of states) is polynomial in $\|\mathcal{K}\|$.

As for $\mathbf{A}_{\mathbf{T}}$, we can easily define first a 2ATA $\mathbf{A}_V$ that verifies whether a given tree satisfies condition (e2) of Definition 5.3. We then obtain the desired $\mathbf{A}_{\mathbf{T}}$ by intersecting $\mathbf{A}_V$ with an adaptation $\mathbf{A}'_{\mathcal{I}}$ of $\mathbf{A}_{\mathcal{I}}$ from Section 4.2, such that its alphabet is $\Sigma_{\mathcal{K},q}$ and it accepts all trees whose $\Sigma_{\mathcal{K}}$-restrictions are interpretation trees.

Definition 5.7. The 2ATA $\mathbf{A}_V = \langle b_{\mathcal{K}}, \Sigma_{\mathcal{K},q}, Q_V, \delta_V, s_0^V, F_V \rangle$ is defined as follows:

- $Q_V = \{s_0^V\} \cup \mathbf{V}_q \cup \bigcup_{v \in \mathbf{V}_q} \{\neg v, v^+, v^-\} \cup \bar{\mathbf{R}}_{\mathcal{K}}$. Intuitively, the state v is used to check that the label of the current node contains v, and the state $\neg v$ that it doesn't. The state v^+ is used to check that the node labeled v is in the tree rooted at the current node, and the state v^- that it is not.

- $F_V = (\emptyset, \{v^- \mid v \in \mathbf{V}_q\}, \mathrm{Q}_V)$.
- 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:

$$\delta_V\left(s_0^V, \sigma\right) = \bigwedge_{v \in \mathbf{V}_q} \left((0, \neg v) \wedge \bigvee_{1 \leqslant j \leqslant b_\mathcal{K}} \left(\bigvee_{a \in \mathbf{I}_\mathcal{K}} (j, a) \wedge \left(j, v^+\right) \wedge \bigwedge_{1 \leqslant j' \leqslant b_\mathcal{K},\ j' \neq j}\right)\right)$$

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

$$\delta_V\left(v^+, \sigma\right) = \left((0, v) \wedge \bigwedge_{1 \leqslant i \leqslant b_\mathcal{K}} \left(i, v^-\right)\right) \vee \left((0, \neg v) \wedge \bigvee_{1 \leqslant i \leqslant b_\mathcal{K}} \left(\left(i, v^+\right) \wedge \bigvee_{P \in \mathbf{R}_\mathcal{K}} (i, P) \wedge \bigwedge_{1 \leqslant j \leqslant b_\mathcal{K},\ j \neq i} \left(j, v^-\right)\right)\right)$$

$$\delta_V\left(v^-, \sigma\right) = (0, \neg v) \wedge \bigwedge_{1 \leqslant i \leqslant b_\mathcal{K}} \left(i, v^-\right)$$

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

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

  4. For each $\sigma \in \Sigma_{\mathcal{K},q}$ and each $P \in \bar{\mathbf{R}}_\mathcal{K}$, transitions that check the labeling with P:

$$\delta_V(P, \sigma) = \begin{cases} \text{true}, & \text{if } P \in \sigma \\ \text{false}, & \text{if } P \notin \sigma \end{cases}$$

The automaton $\mathbf{A}_\mathcal{I}$ in Definition 4.6 is modified to $\mathbf{A}'_\mathcal{I} = \langle b_\mathcal{K}, \Sigma_{\mathcal{K},q}, \mathrm{Q}_\mathcal{I}, \delta'_\mathcal{I}, s_0^\mathcal{I}, F_\mathcal{I}\rangle$, by changing $\Sigma_\mathcal{K}$ to $\Sigma_{\mathcal{K},q}$ and setting, for each $\sigma \in \Sigma_{\mathcal{K},q}$ and each state $q' \in \mathrm{Q}_\mathcal{I}$, $\delta'_\mathcal{I}(\sigma, q') = \delta_\mathcal{I}(\sigma', q')$ whenever $\sigma' = \sigma \setminus \mathbf{V}_q$. Then, $\mathbf{A}_\mathbf{T}$ is an automaton that accepts the intersection of $\mathbf{A}_V$ and $\mathbf{A}'_\mathcal{I}$, as in Lemma 2.10. □

Lemma 5.8. $\mathcal{L}(\mathbf{A}_\mathbf{T}) = \{\mathbf{T} \mid \mathbf{T}$ *is an extended interpretation tree for* $\mathcal{K}\}$.

Proof. $\mathcal{L}(\mathbf{A}_\mathbf{T})$ is the intersection of $\mathcal{L}(\mathbf{A}'_\mathcal{I})$ and $\mathcal{L}(\mathbf{A}_V)$. By construction, for every $\Sigma_{\mathcal{K},q}$-labeled $b_\mathcal{K}$-ary tree $\mathbf{T} = (T, L)$, we have $\mathbf{T} \in \mathcal{L}(\mathbf{A}'_\mathcal{I})$ iff $\mathbf{T}|_{\Sigma_\mathcal{K}} \in \mathcal{L}(\mathbf{A}_\mathcal{I})$. Hence $\mathbf{T} \in \mathcal{L}(\mathbf{A}'_\mathcal{I})$ iff it satisfies (e1) in Definition 5.3. It is then enough to prove that there is an accepting run of $\mathbf{A}_V$ over $\mathbf{T}$ iff $\mathbf{T}$ satisfies (e2) in Definition 5.3. To this aim, we first observe that, by item 3 in the definition of δ_V, the following hold for every $x \in T$ and $v \in \mathbf{V}_q$:

- There is an accepting (x, v)-run of $\mathbf{A}_V$ over $\mathbf{T}$ iff $v \in L(x)$.
- There is an accepting $(x, \neg v)$-run of $\mathbf{A}_V$ over $\mathbf{T}$ iff $v \notin L(x)$.

Similarly, by item 4, for every $x \in T$ and $P \in \mathbf{R}_\mathcal{K}$:

- There is an accepting (x, P)-run of $\mathbf{A}_V$ over $\mathbf{T}$ iff $P \in L(x)$.

Then we can use the second item in the definition of δ_V to show the following:

- There is an accepting (x, v^-)-run of $\mathbf{A}_V$ over $\mathbf{T}$ iff $v \notin L(x \cdot w)$ for every $x \cdot w \in T$ with $w \in \mathbb{N}^*$. This is an easy consequence of the fact that the transition function ensures that there exists an accepting (x, v^-)-run iff there exists an accepting $(x, \neg v)$-run and, for each child $x \cdot i$ of x there exists an accepting $(x \cdot i, v^-)$-run.
- Assume $x \in \Delta^{\mathcal{I}_\mathbf{T}}$. There is an accepting (x, v^+)-run of $\mathbf{A}_V$ over $\mathbf{T}$ iff there is exactly one $x \cdot w \in T$ with $w \in \mathbb{N}^*$ such that $v \in L(x \cdot w)$, and for this $x \cdot w$, we have $x \cdot w \in \Delta^{\mathcal{I}_\mathbf{T}}$. This holds because the transition function ensures that there exists an accepting (x, v^+)-run iff either: (i) there is an accepting (x, v)-run, and for every child $x \cdot i$ of x, there exists an accepting $(x \cdot i, v^-)$-run; or (ii) there is an accepting $(x, \neg v)$-run, there is one child $x \cdot i$ of x such that there exists an accepting $(x \cdot i, v^+)$-run and an accepting $(x \cdot i, P)$-run for some $P \in \mathbf{R}_\mathcal{K}$, and for all remaining children $x \cdot j$, $j \neq i$, there exists an accepting $(x \cdot j, v^-)$-run. Note that $x \in \Delta^{\mathcal{I}_\mathbf{T}}$ and the existence of the $(x \cdot i, P)$-run for some $P \in \mathbf{R}_\mathcal{K}$ imply that $x \cdot i \in \Delta^{\mathcal{I}_\mathbf{T}}$.

Item 1 in the definition of δ_V ensures that, in every run starting at ε and s_0^V, for each $v \in \mathbf{V}_q$, (after checking that $v \notin L(\varepsilon)$) the automaton moves to one individual node in state v^+, and to all remaining level one nodes in state v^- ($\mathbf{A}_V$ does not check that the labels $a \in \mathbf{I}_\mathcal{K}$ correctly identify a unique individual node, but this is verified by $\mathbf{A}'_\mathcal{I}$ so we can assume it is the case). From this and the items shown above, it follows that there exists an accepting (ε, s_0^V)-run iff for each $v \in \mathbf{V}_q$, there exists some individual node i and exactly one $i \cdot w \in T$ with $w \in \mathbb{N}^*$ such that $v \in L(i \cdot w)$, and for this $i \cdot w$, we have $i \cdot w \in \Delta^{\mathcal{I}_\mathbf{T}}$. Moreover, $v \notin L(x)$ for every $x \in T$, $x \neq i \cdot x_v$. This shows that there exists an accepting (ε, s_0^V)-run iff (e2) holds. □

To define the 2ATA $\mathbf{A}_\pi$ we use *q-concepts*. They are like regular concepts, but may use the individuals and variables in q as atomic concepts.

Definition 5.9. 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. 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). For each atom $\alpha = R(v, v')$ in q, we define the q-concept $C_\alpha = \exists p_U{}^*.(v \sqcap \exists R.v')$, where p_U is the role introduced in step 2 of the normalization of $\mathcal{K}$, or if this p_U is not present in $\mathcal{K}$, then $p_U = \bigcup_{P \in \bar{\mathbf{R}}_\mathcal{K}} P$. □

Intuitively, C_α expresses that somewhere (reachable by any chain of roles) there is an object labeled v, which is related via R to an object labeled v'. Since the interpretations represented in extended interpretation trees are $\mathcal{K}$-connected, and every node is reachable from a root, we have that $\mathcal{I}, \pi \models \alpha$ iff C_α is satisfied at some root. Hence, we can check whether π is a match for q by checking the satisfaction of the q-concepts for its atoms at the roots.

Example 5.10. Recall the query q_g from Example 2.5, with atoms $\alpha_1 = \mathit{HasParent}^* \circ \mathit{HasParent}^{-*}(v_1, v_2)$, $\alpha_2 = \mathit{HasParent}^-(v_1, v_3)$, $\alpha_3 = \mathit{HasParent}^-(v_2, v_3)$, $\alpha_4 = \mathit{male}(v_1)$, $\alpha_5 = \mathit{female}(v_2)$, $\alpha_6 = \neg \mathit{deity}(v_1)$, and $\alpha_7 = \neg\mathit{deity}(v_2)$. In order to have only binary atoms, we consider $\alpha_4' = \mathit{id}(\mathit{male})(v_1, v_1)$, $\alpha_5' = \mathit{id}(\mathit{female})(v_2, v_2)$, $\alpha_6' = \mathit{id}(\neg\mathit{deity})(v_1, v_1)$, and $\alpha_7' = \mathit{id}(\neg\mathit{deity})(v_2, v_2)$. The respective q-concepts are:

$$\begin{aligned}
C_{\alpha_1} &= \exists p_U{}^*.\big(v_1 \sqcap \exists \mathit{HasParent}^* \circ \mathit{HasParent}^{-*}.v_2\big) & C_{\alpha_4'} &= \exists p_U{}^*.\big(v_1 \sqcap \exists \mathit{id}(\mathit{male}).v_1\big)\\
C_{\alpha_2} &= \exists p_U{}^*.\big(v_1 \sqcap \exists \mathit{HasParent}^-.v_3\big) & C_{\alpha_5'} &= \exists p_U{}^*.\big(v_2 \sqcap \exists \mathit{id}(\mathit{female}).v_2\big)\\
C_{\alpha_3} &= \exists p_U{}^*.\big(v_2 \sqcap \exists \mathit{HasParent}^-.v_3\big) & C_{\alpha_6'} &= \exists p_U{}^*.\big(v_1 \sqcap \exists \mathit{id}(\neg\mathit{deity}).v_1\big)\\
& & C_{\alpha_7'} &= \exists p_U{}^*.\big(v_2 \sqcap \exists \mathit{id}(\neg\mathit{deity}).v_2\big)
\end{aligned}$$

Consider the extended interpretation $\mathcal{J} = (\mathcal{I}_g, \pi)$ represented in Fig. 6, which has roots $\{1, \ldots 5\}$. Since the root label is $LR = \{r, \mathit{HasFather}_{21}, \mathit{HasFather}_{34}, \mathit{HasFather}_{45}, \mathit{HasFather}_{51}, \mathit{HasMother}_{23}, \mathit{HasParent}_{21}, \mathit{HasParent}_{34}, \mathit{HasParent}_{45}, \mathit{HasParent}_{51}, \mathit{HasParent}_{23}\}$, we have that:

$$\begin{aligned}
\big(v_1 \sqcap \exists \mathit{HasParent}^* \circ \mathit{HasParent}^{-*}.v_2\big)^\mathcal{J} &= \{1\} & \big(v_1 \sqcap \exists \mathit{id}(\mathit{male}).v_1\big)^\mathcal{J} &= \{1\}\\
\big(v_1 \sqcap \exists \mathit{HasParent}^-.v_3\big)^\mathcal{J} &= \{1\} & \big(v_2 \sqcap \exists \mathit{id}(\mathit{female}).v_2\big)^\mathcal{J} &= \{3\}\\
\big(v_2 \sqcap \exists \mathit{HasParent}^-.v_3\big)^\mathcal{J} &= \{3\} & \big(v_1 \sqcap \exists \mathit{id}(\neg\mathit{deity}).v_1\big)^\mathcal{J} &= \emptyset\\
& & \big(v_2 \sqcap \exists \mathit{id}(\neg\mathit{deity}).v_2\big)^\mathcal{J} &= \{3\}
\end{aligned}$$

Hence $C_{\alpha_1}^\mathcal{J} = C_{\alpha_2}^\mathcal{J} = C_{\alpha_3}^\mathcal{J} = C_{\alpha_4'}^\mathcal{J} = C_{\alpha_5'}^\mathcal{J} = C_{\alpha_7'}^\mathcal{J} = \Delta^\mathcal{J}$, and in particular we have $i \in =^\mathcal{J} C_{\alpha_2}^\mathcal{J} = C_{\alpha_3}^\mathcal{J} = C_{\alpha_4'}^\mathcal{J} = C_{\alpha_5'}^\mathcal{J} = C_{\alpha_7'}^\mathcal{J}$ for every root i of $\mathcal{I}_g$. In contrast, $C_{\alpha_6'}^\mathcal{J} = \emptyset$, so $i \notin C_{\alpha_6'}^\mathcal{J}$ for every root i. □

As discussed above, the following holds.

Lemma 5.11. *For every extended canonical 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 $i \in C_\alpha{}^\mathcal{J}$.*

Proof. (*Only If*) Assume $\mathcal{I}, \pi \models R(v, v')$. Then $(\pi(v), \pi(v')) \in R^\mathcal{I}$, and by the semantics of q-concepts, $\pi(v) \in (\exists R.v')^\mathcal{J}$. As $v^\mathcal{J} = \{\pi(v)\}$ by definition, we get $\pi(v) \in (v \sqcap \exists R.v')^\mathcal{J}$. Since $\mathcal{I}$ is a canonical interpretation, there is some $i \in \mathit{Roots}(\mathcal{I})$ such that $\pi(v)$ is $\mathcal{K}$-connected to i, that is, $(i, \pi(v)) \in (p_U{}^*)^\mathcal{I}$. By the semantics of q-concepts, this implies $i \in (\exists p_U{}^*.v)^\mathcal{J}$. This, together with $v^\mathcal{J} = \{\pi(v)\}$ and $\pi(v) \in (v \sqcap \exists R.v')^\mathcal{J}$ gives $i \in (\exists p_U{}^*.(v \sqcap \exists R.v'))^\mathcal{J}$ as desired.

(*If*) Assume $i \in (\exists p_U{}^*.(v \sqcap \exists R.v'))^\mathcal{J}$ for some i. Then $(v \sqcap \exists R.v')^\mathcal{J} \neq \emptyset$. As $(v \sqcap \exists R.v')^\mathcal{J} \subseteq v^\mathcal{J}$ and $v^\mathcal{J} = \{\pi(v)\}$ by definition, we have $\pi(v) \in (v \sqcap \exists R.v')^\mathcal{J}$. Hence $\pi(v) \in (\exists R.v')^\mathcal{J}$, which by $(v')^\mathcal{J} = \{\pi(v')\}$ gives $(\pi(v), \pi(v')) \in R^\mathcal{J}$, so $(\pi(v), \pi(v')) \in R^\mathcal{I}$ and $\mathcal{I}, \pi \models R(v, v')$. □

Table 7
State set Q_π of the automaton $\mathbf{A}_\pi$.

$Cl^q_{ext} = Cl_q \cup \mathbf{V}_q \cup \mathbf{I}_q \cup \{\neg s \mid s \in \mathbf{V}_q \cup \mathbf{I}_q\}$
$\mathrm{Q}_{q,\mathsf{Self}} = \{S_{\mathsf{Self}} \mid S \text{ is a simple role in } Cl_q\} \cup \{\langle a, p_{\mathsf{Self}}\rangle \mid a \in \mathbf{I}_q, p \text{ is a role name in } Cl_q\}$
$\mathrm{Q}_{q,\mathcal{A}\_role} = \{S_{ij} \mid i, j \in \{1, \ldots, b_\mathcal{K}\} \text{ and } S \text{ is a simple role in } Cl_q\}$
$\mathrm{Q}_{q,num} = \{\langle \geqslant nS.C, i, j\rangle \mid \geqslant nS.C \in Cl_q,\ 0 \leqslant i \leqslant b_\mathcal{K}+1,\ 0 \leqslant j \leqslant n\} \cup \{\langle \leqslant nS.C, i, j\rangle \mid \leqslant nS.C \in Cl_q,\ 0 \leqslant i \leqslant b_\mathcal{K}+1,\ 0 \leqslant j \leqslant n+1\}$
$\mathrm{Q}_{q,\mathcal{A}\_num} = \{\langle a, \geqslant nS.C, i, j\rangle \mid a \in \mathbf{I}_q,\ \geqslant nS.C \in Cl_q,\ 1 \leqslant i \leqslant b_\mathcal{K},\ 0 \leqslant j \leqslant n\} \cup \{\langle a, \leqslant nS.C, i, j\rangle \mid a \in \mathbf{I}_q,\ \leqslant nS.C \in Cl_q,\ 1 \leqslant i \leqslant b_\mathcal{K}; 0 \leqslant j \leqslant n+1\}$
$\mathrm{Q}_\pi = \{s^\pi_0\} \cup Cl^q_{ext} \cup \mathrm{Q}_{q,\mathsf{Self}} \cup \mathrm{Q}_{q,\mathcal{A}\_role} \cup \mathrm{Q}_{q,num} \cup \mathrm{Q}_{q,\mathcal{A}\_num}$

By this lemma, we only need to verify that each of $C_{\alpha_1}, \ldots, C_{\alpha_k}$ holds 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. We recall that $q = \exists\vec{v}.\varphi(\vec{v})$, and that all atoms in φ are of the form $R(v, v')$ with R a $\mathcal{ZIQ}$ role in NNF.

Definition 5.12. Let $Cl_q = \bigcup_{\alpha \in At(q)} Cl(C_\alpha)$. We define the 2ATA $\mathbf{A}_\pi = \langle b_\mathcal{K}, \Sigma_{\mathcal{K},q}, \mathrm{Q}_\pi, \delta_\pi, s^\pi_0, F_\pi\rangle$ as follows:

- Q_π is like $\mathrm{Q}_\mathcal{T}$ in $\mathbf{A}_\mathcal{T}$, but defined using Cl_q and $\mathbf{I}_q$ instead of $Cl(C_\mathcal{T})$ and $\mathbf{I}_\mathcal{K}$, and treating the symbols in $\mathbf{V}_q$ analogously to the concept names in $\mathbf{C}_q$, see Table 7.
- $F_\pi = (\emptyset, \{\forall R^*.C \mid \forall R^*.C \in Cl_q\}, \mathrm{Q}_\pi)$, analogously to $\mathbf{A}_\mathcal{T}$.
- The transitions from the initial state are defined for each label σ containing r (identifying the root node) as

  $$\delta_\pi(s_0, \sigma) = B_\varphi,$$

  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 $\alpha \in q$ and each σ containing r:

  $$\delta_\pi(C_\alpha, \sigma) = \bigvee_{1 \leqslant i \leqslant b_\mathcal{K}} \Big((i, C_\alpha) \wedge \bigvee_{a \in \mathbf{I}_\mathcal{K}} (i, a) \Big)$$

  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, and 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 group (II) of $\delta_\mathcal{T}$;
  2. transitions $\delta_\pi(s, \sigma)$ for each $s \in Cl_q$ of the form $\geqslant nS.C$, 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 group (IV) of $\delta_\mathcal{T}$;
  3. transitions $\delta_\pi(s, \sigma)$ for each $s \in \mathbf{C}_q \cup \overline{\mathbf{R}}_q \cup \mathbf{I}_q \cup \{p_{ij} \mid p \in \mathbf{R}_q,\ i, j \in \{1, \ldots, b_\mathcal{K}\}\}$, as in group (III) of $\delta_\mathcal{T}$, and for each $s \in \mathbf{V}_q \cup \mathbf{I}_q$, as in group 3 of δ_V, to check symbol occurrences in node labels.

Finally, we define $\mathbf{A}_q$ as a 2ATA accepting the intersection of $\mathbf{A}_\mathbf{T}$ and $\mathbf{A}_\pi$. □

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

Lemma 5.13. *(Cf. Lemma 4.14.) Let $\mathbf{T} = (T, L)$ be an extended interpretation tree and let $x \in \Delta^{\mathcal{I}_\mathbf{T}}$. Furthermore, let $C \in Cl_q$. Then there is an accepting (x, C)-run of $\mathbf{A}_\pi$ over $\mathbf{T}$ iff $x \in C^{\mathcal{J}_\mathbf{T}}$.*

Proof (Sketch). The proof is analogous to that of Lemma 4.14. The only difference is that in the base case and in the case of atomic negation, C may also be a possibly negated variable or individual $t \in \mathbf{I}_q \cup \mathbf{V}_q$. In this case the last group of atomic transitions ensures that an accepting (x, t)-run (resp., an accepting $(x, \neg t)$-run) exists iff $t \in L(x)$ (resp., $t \notin L(x)$). □

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 5.11, the initial transition from the root is sufficient to verify whether this holds for a set of atoms that makes q true:

Lemma 5.14. *An extended interpretation tree $\mathbf{T} = (T, L)$ is accepted by $\mathbf{A}_\pi$ iff there exists some $B \subseteq At(q)$ such that*

(a) *by assigning true to the atoms in B and false to those in $At(q) \setminus B$, φ evaluates to* true, *and*
(b) *for every $\alpha \in B$, there is some root $i \in \Delta^{\mathcal{I}_\mathbf{T}}$ such that $i \in C_\alpha{}^{\mathcal{J}_\mathbf{T}}$.*

Proof. The result follows straightforwardly from the definition of δ_π and Lemma 5.13. Indeed, suppose that there is an accepting run (T_r, r) of $\mathbf{A}_\pi$ over $\mathbf{T}$. Since $\delta_\pi(s_0, \sigma) = B_\varphi$ is the only given transition from state s_0, there is a set $B \subseteq At(q)$ such that B satisfies condition (a), and the root of T_r has a child labeled (ε, C_α) for each $\alpha \in B$. By the transitions of the

Table 8
Automata for deciding query entailment.

Automaton	Construction	Language of trees representing	Type, number of states
$\mathbf{A}_{\mathcal{K}}$	Definition 4.16	canonical models of $\mathcal{K}$	2ATA, polynomial in $\|\mathcal{K}\|$
$\mathbf{A}_q$	Definition 5.12 intersection of $\mathbf{A_T}$ and $\mathbf{A}_\pi$	canonical interpretations for $\mathcal{K}$ together with a match for q	2ATA, polynomial in $\|\mathcal{K}\| + \|q\|$
$\mathbf{A}_{\neg q}$	Lemma 5.17 complement of the projection of $\mathbf{A}_q$ to $\Sigma_{\mathcal{K}}$, after transforming to 1NTA.	canonical interpretations of $\mathcal{K}$ where q has no match	1NTA, double exponential in $\|\mathcal{K}\| + \|q\|$
$\mathbf{A}_{\mathcal{K}\not\models q}$	Lemma 5.18 intersection of $\mathbf{A}_{\neg q}$ with $\mathbf{A}_{\mathcal{K}}$, after transforming to 1NTA.	canonical models of $\mathcal{K}$ where q has no match	1NTA, double exponential in $\|\mathcal{K}\| + \|q\|$

form $\delta_\pi(C_\alpha, \sigma) = \bigvee_{1\leqslant i\leqslant b_{\mathcal{K}}}((i, C_\alpha) \wedge \bigvee_{a\in \mathbf{I}_{\mathcal{K}}}(i, a))$, each such child with label (ε, C_α) must in turn have a child (i, a) and a child (i, C_α). Since the run is accepting, it must be the case that $a \in L(i)$, which implies that i is a root of $\mathcal{I}_{\mathbf{T}}$, and that, by Lemma 5.13, $i \in C_\alpha^{\mathcal{J}_{\mathbf{T}}}$, so B satisfies condition (b) as well. For the converse, assume a set B as in the claim. Then we know that for each $\alpha \in B$ there is a root i_α of $\mathbf{T}_{\mathcal{I}}$ such that $i_\alpha \in C_\alpha^{\mathcal{J}_{\mathbf{T}}}$. As i_α is a root of $\mathbf{T}_{\mathcal{I}}$, there exists some $a \in L(i_\alpha)$ and an accepting (i_α, a)-run $\mathbf{T}_{a,\alpha}$ of $\mathbf{A}_\pi$ on $\mathbf{T}$. As $i_\alpha \in C_\alpha^{\mathcal{J}_{\mathbf{T}}}$, by Lemma 5.13 there is an accepting (i_α, C_α)-run $\mathbf{T}_{C,\alpha}$ of $\mathbf{A}_\pi$ on $\mathbf{T}$. We can obtain an accepting run of $\mathbf{A}_\pi$ on $\mathbf{T}$ by taking a root labeled (ε, s_0), creating a child y_α labeled (ε, C_α) for each $\alpha \in B$, and adding to each y_α the whole $\mathbf{T}_{a,\alpha}$ and $\mathbf{T}_{C,\alpha}$ as subruns (that is, adding descendants and labels as in the respective runs). This shows that an accepting run of $\mathbf{A}_\pi$ over $\mathbf{T}$ exists. □

The following is a simple corollary of Lemmas 5.11 and 5.14:

Corollary 5.15. *Let $\mathbf{T}$ be an extended interpretation tree. Then $\mathbf{T} \in \mathcal{L}(\mathbf{A}_\pi)$ iff $\mathcal{J}_{\mathbf{T}} \models q$.*

Now we can easily show that $\mathbf{A}_q$ accepts the trees representing interpretations where q has a match:

Proposition 5.16. *$\mathcal{L}(\mathbf{A}_q) = \{\mathbf{T} \mid \mathbf{T}$ is an extended interpretation tree and $\mathcal{J}_{\mathbf{T}} \models q\}$*

Proof (Sketch). ($\subseteq$) Assume $\mathbf{T}$ is a $\Sigma_{\mathcal{K},q}$ labeled $b_{\mathcal{K}}$-ary tree that $\mathbf{T} \in \mathcal{L}(\mathbf{A}_q)$. Then $\mathbf{T} \in \mathcal{L}(\mathbf{A_T})$. By Lemma 5.8, $\mathbf{T}$ is an extended interpretation tree. Furthermore, $\mathbf{T} \in \mathcal{L}(\mathbf{A}_\pi)$, so by Corollary 5.15 $\mathcal{J}_{\mathbf{T}} \models q$.

($\supseteq$) Assume $\mathbf{T}$ is an extended interpretation tree such that $\mathcal{J}_{\mathbf{T}} \models q$. Then $\mathbf{T} \in \mathcal{L}(\mathbf{A_T})$ by Lemma 5.8. As $\mathcal{J}_{\mathbf{T}} \models q$, by Corollary 5.15 we have $\mathbf{T} \in \mathcal{L}(\mathbf{A}_\pi)$. Hence $\mathbf{T} \in \mathcal{L}(\mathbf{A_T}) \cap \mathcal{L}(\mathbf{A}_\pi) = \mathcal{L}(\mathbf{A}_q)$. □

5.3. Deciding query entailment

Our algorithm for deciding $\mathcal{K} \models q$ roughly works as follows (cf. Fig. 5). The automaton $\mathbf{A}_q$ accepts a tree over the alphabet $\Sigma_{\mathcal{K},q}$, if it represents an extended canonical interpretation $(\mathcal{I}, \pi)$ 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 the resulting automaton $\mathbf{A}_{\neg q}$ 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 3.10, $\mathcal{K} \not\models q$ 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. For easier reference, we summarize in Table 8 the characteristics of the different automata which the query entailment algorithm comprises (recall Fig. 5).

We now show in detail how $\mathbf{A}_{\mathcal{K}\not\models q}$ can be obtained and analyze its size. We start by transforming $\mathbf{A}_q$ into a 1NTA whose language we can project to the alphabet $\Sigma_{\mathcal{K}}$. Note that the transformation to 1NTA causes an exponential blow-up in the number of states (Proposition 2.12), but after it the projection can be easily done (Proposition 2.14). Then we complement the resulting automaton in order to construct a 1NTA that accepts exactly the set of trees such that, if they represent an interpretation for $\mathcal{K}$, it is an interpretation where q has no match. The complementation also causes yet another exponential blow-up in the states, and it makes the index as large as the original state set (Proposition 2.13). More precisely, we have:

Lemma 5.17. *Given $\mathcal{K}$ and q, it is possible to construct a 1NTA $\mathbf{A}_{\neg q}$ such that*

1. *If $\mathbf{T}$ is an interpretation tree for $\mathcal{K}$, then $\mathbf{T} \in \mathcal{L}(\mathbf{A}_{\neg q})$ iff $\mathcal{I}_{\mathbf{T}} \not\models q$.*
2. *$|Q(\mathbf{A}_{\neg q})| \leqslant 2^{2^{O(n_q^c)}}$ and $\mathsf{ind}(\mathbf{A}_{\neg q}) \leqslant 2^{O(n_q^c)}$ for $n_q = |Q(\mathbf{A}_q)|$ and some constant c, i.e., $\mathbf{A}_{\neg q}$ has double exponentially many states and 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 Proposition 2.12, we can construct from $\mathbf{A}_0$ a 1NTA $\mathbf{A}_1$ with $\mathcal{L}(\mathbf{A}_1) = \mathcal{L}(\mathbf{A}_0)$ such that $|Q(\mathbf{A}_1)| \leqslant 2^{O(n_q^{c_0})}$ for some constant c_0 and $\mathsf{ind}(\mathbf{A}_1) = O(\mathsf{ind}(\mathbf{A}_0)) = O(1)$. By Lemma 2.14, we can transform $\mathbf{A}_1$ into a

1NTA $\mathbf{A}_2$ that accepts the $\Sigma_{\mathcal{K}}$-projection of $\mathcal{L}(\mathbf{A}_1)$ such that $|Q(\mathbf{A}_2)| \leqslant |Q(\mathbf{A}_1)| \leqslant 2^{O(|Q(\mathbf{A}_0)|^{c_0})}$ and $\mathsf{ind}(\mathbf{A}_2) \leqslant \mathsf{ind}(\mathbf{A}_1) = O(1)$. By Lemma 2.13, 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 $|Q(\mathbf{A}_3)| \leqslant 2^{O(f(\mathbf{A}_2))}$ and $\mathsf{ind}(\mathbf{A}_3) = O(f(\mathbf{A}_2))$, where $f(\mathbf{A}_2) = \mathsf{ind}(\mathbf{A}_2)\cdot|Q(\mathbf{A}_2)|\cdot\log|Q(\mathbf{A}_2)| \leqslant 2^{O(|Q(\mathbf{A}_0)|^{c_2})}\cdot O(|Q(\mathbf{A}_0)|^{c_2})$, for some constant c_2. It follows that $|Q(\mathbf{A}_{\neg q})| \leqslant 2^{2^{O(n_q^c)}}$ and $\mathsf{ind}(\mathbf{A}_{\neg q}) \leqslant 2^{O(n_q^c)}$ for some constant c. This shows the second item.

For the first item, we know from Proposition 5.16 that $\mathcal{L}(\mathbf{A}_q) = \mathcal{L}(\mathbf{A}_1) = \{\mathbf{T} \mid \mathbf{T}$ is an extended interpretation tree and $\mathcal{J}_{\mathbf{T}} \models q\}$. By construction, $\mathbf{A}_2$ accepts the $\Sigma_{\mathcal{K}}$-projection of $\mathcal{L}(\mathbf{A}_1)$, i.e., $\mathcal{L}(\mathbf{A}_2) = \{\mathbf{T}|_{\Sigma_{\mathcal{K}}} \mid \mathbf{T}$ is an extended interpretation tree and $\mathcal{J}_{\mathbf{T}} \models q\}$. Recall that, by definition, $\mathbf{T}|_{\Sigma_{\mathcal{K}}}$ is an interpretation tree. Moreover, for an arbitrary interpretation tree $\mathbf{T}$ such that there exists some $\pi : \mathbf{V}_q \cup \mathbf{I}_q \to \Delta^{\mathcal{I}_{\mathbf{T}}}$ with $\mathcal{I}_{\mathbf{T}}, \pi \models q$, we have that $(\mathcal{I}_{\mathbf{T}}, \pi)$ is an extended canonical interpretation and there is an extended interpretation tree $\mathbf{T}'$ such that $\mathcal{J}_{\mathbf{T}'} = (\mathcal{I}_{\mathbf{T}}, \pi)$ (obtained by adding in $\mathbf{T}$ the variable v to the label of the node $\pi(v)$, for all query variables v). This implies that $\mathbf{T}$ is accepted by $\mathbf{A}_2$. Hence, we can equivalently describe $\mathcal{L}(\mathbf{A}_2)$ as $\{\mathbf{T} \mid \mathbf{T}$ is an interpretation tree and there exists some $\pi : \mathbf{V}_q \cup \mathbf{I}_q \to \Delta^{\mathcal{I}_{\mathbf{T}}}$ with $\mathcal{I}_{\mathbf{T}}, \pi \models q\}$. Then the complement $\mathbf{A}_3 = \mathbf{A}_{\neg q}$ of $\mathbf{A}_2$ accepts an interpretation tree $\mathbf{T}$ iff there exists no $\pi : \mathbf{V}_q \cup \mathbf{I}_q \to \Delta^{\mathcal{I}_{\mathbf{T}}}$ with $\mathcal{I}_{\mathbf{T}}, \pi \models q$, i.e., iff $\mathcal{I}_{\mathbf{T}} \not\models q$. This shows the first item. □

We can now transform $\mathbf{A}_{\mathcal{K}}$ into a 1NTA and intersect it with $\mathbf{A}_{\neg q}$, to obtain the automaton $\mathbf{A}_{\mathcal{K}\not\models q}$. It accepts exactly the canonical models of $\mathcal{K}$ for which there is no match for q, as desired.

Lemma 5.18. *Given $\mathcal{K}$ and q, it is possible to construct via $\mathbf{A}_{\mathcal{K}}$ and $\mathbf{A}_{\neg q}$ a 1NTA $\mathbf{A}_{\mathcal{K}\not\models q}$ such that:*

1. *$\mathcal{L}(\mathbf{A}_{\mathcal{K}\not\models q}) = \{\mathbf{T} \mid \mathbf{T}$ is an interpretation tree such that $\mathcal{I}_{\mathbf{T}} \models \mathcal{K}$ and $\mathcal{I}_{\mathbf{T}} \not\models q\}$.*
2. *$|Q(\mathbf{A}_{\mathcal{K}\not\models q})| \leqslant 2^{2^{O((n_{\mathcal{K}}+n_q)^c)}}$ and $\mathsf{ind}(\mathbf{A}_{\mathcal{K}\not\models q}) \leqslant 2^{O(n_q^c)}$ for some constant c where $n_{\mathcal{K}} = |Q(\mathbf{A}_{\mathcal{K}})|$ and $n_q = |Q(\mathbf{A}_q)|$, i.e., $\mathbf{A}_{\mathcal{K}\not\models q}$ has double exponentially many states in the number of states of $\mathbf{A}_{\mathcal{K}}$ and $\mathbf{A}_q$, and index single exponential in the number of states of $\mathbf{A}_q$.*

Proof. By Proposition 2.12 we can construct from $\mathbf{A}_{\mathcal{K}}$ a 1NTA $\mathbf{A}_1$, with $|Q(\mathbf{A}_1)| \leqslant 2^{O(|Q(\mathbf{A}_{\mathcal{K}})|^{c_1})}$ 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 2.15 has

$$\mathsf{ind}(\mathbf{A}_3) = O\big(f(\mathbf{A}_1, \mathbf{A}_2)\big),$$
$$\big|Q(\mathbf{A}_3)\big| \leqslant 2^{O(f(\mathbf{A}_1,\mathbf{A}_2)^2)} \cdot f(\mathbf{A}_1, \mathbf{A}_2) \cdot \big|Q(\mathbf{A}_1)\big| \cdot \big|Q(\mathbf{A}_2)\big|,$$

where $f(\mathbf{A}_1, \mathbf{A}_2) = \mathsf{ind}(\mathbf{A}_1) + \mathsf{ind}(\mathbf{A}_2) + 1$. Since by Lemma 5.17, $|Q(\mathbf{A}_2)| \leqslant 2^{2^{O(n_q^{c_2})}}$ and $\mathsf{ind}(\mathbf{A}_2) \leqslant 2^{O(n_q^{c_2})}$ for some constant c_2, we have that

$$\mathsf{ind}(\mathbf{A}_3) \leqslant 2^{O(n_q^{c_2})} + O(1),$$
$$\big|Q(\mathbf{A}_3)\big| \leqslant 2^{O(f(\mathbf{A}_1,\mathbf{A}_2)^2)} \cdot f(\mathbf{A}_1, \mathbf{A}_2) \cdot 2^{O(|Q(\mathbf{A}_{\mathcal{K}})|^{c_1})} \cdot 2^{2^{O(n_q^{c_2})}},$$

where $f(\mathbf{A}_1, \mathbf{A}_2) = \mathsf{ind}(\mathbf{A}_1) + \mathsf{ind}(\mathbf{A}_2) + 1 \leqslant 2^{O(n_q^{c_2})} + O(1) + 1$. It follows from this that $|Q(\mathbf{A}_{\mathcal{K}\not\models q})| \leqslant 2^{2^{O((n_{\mathcal{K}}+n_q)^c)}}$ and $\mathsf{ind}(\mathbf{A}_{\mathcal{K}\not\models q}) \leqslant 2^{O(n_q^c)}$ for some constant c. This shows the second item. Now we show the first item, i.e., that $\mathcal{L}(\mathbf{A}_{\mathcal{K}\not\models q}) = \mathcal{L}(\mathbf{A}_{\mathcal{K}}) \cap \mathcal{L}(\mathbf{A}_{\neg q})$ is the set of trees representing models of $\mathcal{K}$ where q has no match. ($\subseteq$) Suppose $\mathbf{T} \in \mathcal{L}(\mathbf{A}_{\mathcal{K}\not\models q})$. Then $\mathbf{T} \in \mathcal{L}(\mathbf{A}_{\mathcal{K}})$, which by Proposition 4.17 means that $\mathbf{T}$ is an interpretation tree and $\mathcal{I}_{\mathbf{T}} \models \mathcal{K}$. We also have $\mathbf{T} \in \mathcal{L}(\mathbf{A}_{\neg q})$, which by Lemma 5.17 means that $\mathcal{I}_{\mathbf{T}} \not\models q$. ($\supseteq$) Conversely, if $\mathbf{T}$ is an interpretation tree such that $\mathcal{I}_{\mathbf{T}} \models \mathcal{K}$ and $\mathcal{I}_{\mathbf{T}} \not\models q$, then by Proposition 4.17 $\mathbf{T} \in \mathcal{L}(\mathbf{A}_{\mathcal{K}})$, and by Lemma 5.17 $\mathbf{T} \in \mathcal{L}(\mathbf{A}_{\neg q})$. This implies that $\mathbf{T} \in \mathcal{L}(\mathbf{A}_{\mathcal{K}}) \cap \mathcal{L}(\mathbf{A}_{\neg q}) = \mathcal{L}(\mathbf{A}_{\mathcal{K}\not\models q})$. □

Therefore we can decide whether $\mathcal{K} \models q$ by testing $\mathbf{A}_{\mathcal{K}\not\models q}$ for emptiness.

Theorem 5.19. *For every normalized P2RPQ q over a normalized KB $\mathcal{K}$ in $\mathcal{ALCQIb}_{reg}$, it holds that $\mathcal{K} \models q$ iff $\mathcal{L}(\mathbf{A}_{\mathcal{K}\not\models q}) = \emptyset$.*

Proof. It follows from Lemma 5.18 that if $\mathcal{L}(\mathbf{A}_{\mathcal{K}\not\models q}) \neq \emptyset$, there exists some interpretation $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{K}$ and $\mathcal{I} \not\models q$, so $\mathcal{K} \not\models q$. For the converse, if $\mathcal{K} \not\models q$, by Theorem 3.10 and our assumption that $b_{\mathcal{K}} \geqslant k_{\mathcal{T},q}$ (see the beginning of Section 5), there exists some $b_{\mathcal{K}}$-canonical interpretation $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{K}$ and $\mathcal{I} \not\models q$. Since by Lemma 4.5 $\mathcal{I} = \mathcal{I}_{\mathbf{T}_{\mathcal{I}}}$, we can apply Lemma 5.18 to conclude that $\mathbf{T}_{\mathcal{I}} \in \mathcal{L}(\mathbf{A}_{\mathcal{K}\not\models q})$ hence $\mathcal{L}(\mathbf{A}_{\mathcal{K}\not\models q}) \neq \emptyset$. □

5.4. Complexity

We now show that the reduction of query entailment to automata emptiness in Theorem 5.19 gives a tight upper complexity bound for the problem. By $\|q\|$ we denote the size of a (string) representation of a query q, and by $\|\mathcal{K}, q\| = \|\mathcal{K}\| + \|q\|$ the combined size of a KB $\mathcal{K}$ and q (assuming unary number encoding).

It is not difficult to show that $\mathbf{A}_q$ has polynomially many states in $\|\mathcal{K}, q\|$, and a short acceptance condition (i.e., its index is a small constant).

Lemma 5.20. $|Q(\mathbf{A}_q)| = O(\|\mathcal{K}, q\|^c)$, *for some constant c, and* $\mathsf{ind}(\mathbf{A}_q) = 3$.

Proof. Recall that under unary number encoding in $\mathcal{K}$ (cf. Section 4.4), $b_{\mathcal{K}}$ is quadratic in $\|\mathcal{K}\|$ (and hence in $\|\mathcal{K}, q\|$). Likewise for q we have that $|Cl_q|$, $|\mathbf{V}_q|$, and $n^q_{\max} = \max(\{n \mid \geqslant n S.C \in Cl_q\} \cup \{0\})$ are linear in $\|q\|$ (and in $\|\mathcal{K}, q\|$). As $\mathbf{A}_q = \mathbf{A}_\pi \cap \mathbf{A}_\mathbf{T} = \mathbf{A}_\pi \cap (\mathbf{A}_V \cap \mathbf{A}'_\mathcal{I})$, it follows from Proposition 2.10 that $|Q(\mathbf{A}_q)| \leqslant |Q_\pi| + |Q_V| + |Q_\mathcal{I}| + 2$, where:

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

Hence, $|Q(\mathbf{A}_q)| = O(\|\mathcal{K}, q\|^5)$. Moreover, by Proposition 2.10, $\mathsf{ind}(\mathbf{A}_q) = \max(\mathsf{ind}(\mathbf{A}_\pi), \mathsf{ind}(\mathbf{A}_V), \mathsf{ind}(\mathbf{A}_\mathcal{I})) = 3$. □

We next establish that the number of states of $\mathbf{A}_{\mathcal{K}\not\models q}$ is double exponential in $\|\mathcal{K}, q\|$.

Lemma 5.21. $|Q(\mathbf{A}_{\mathcal{K}\not\models q})| = 2^{2^{O(\|\mathcal{K},q\|^c)}}$ *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 from Lemmas 4.19, 5.18, and 5.20. The second part holds 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 size of the input. □

Using Proposition 2.16, we obtain the main result of this section.

Theorem 5.22. *Given a P2RPQ q over a KB* $\mathcal{K}$ *in* $\mathcal{ALCQI}b_{reg}$ *and a P2RPQ q over* $\mathcal{K}$, *deciding whether* $\mathcal{K} \models q$ *is in* 2EXPTIME *under unary number encoding.*

This bound is worst case optimal. Indeed, it was shown in [46] that answering CQs over KBs in $\mathcal{ALCI}$ (i.e., P2RPQs built only with conjunction $\wedge$, where no regular role expressions, but only concept and role names are allowed) is 2EXPTIME-hard, and the same lower bound was established in [47] for $\mathcal{SH}$, a DL with transitive roles but lacking inverse roles. It is not hard to see that the proof for $\mathcal{SH}$ in [47] can be adapted to show hardness already for $\mathcal{ALC}$, for two restricted classes of P2RPQs that generalize CQs: (i) *positive queries* (*PQs*), which allow conjunction and disjunction, but atoms contain only concept and role names and no regular role expressions, and (ii) *conjunctive RPQs* (*CRPQs*), which do not allow for either disjunctions between atoms or inverse roles in the regular expressions (this latter lower bound was recently observed in [58]). Hence 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* $\mathcal{L}_\mathcal{K}$. *Then deciding* $\mathcal{K} \models q$ *is* 2EXPTIME-*complete under unary number encoding if any of the following holds:*

(i) $\mathcal{L}_q \subseteq$ *CQs, and either* $\mathcal{ALCI} \subseteq \mathcal{L}_\mathcal{K}$ *or* $\mathcal{SH} \subseteq \mathcal{L}_\mathcal{K}$.
(ii) $\mathcal{ALC} \subseteq \mathcal{L}_\mathcal{K}$, *and either* $\mathcal{L}_q \subseteq$ *CRPQs or* $\mathcal{L}_q \subseteq$ *PQs.*

6. Complex role inclusion axioms

The DL $\mathcal{SRIQ}$ was introduced in [5] as an extension of $\mathcal{RIQ}$ [59], which in turn extends the well-known DL $\mathcal{SHIQ}$ [60] underlying OWL-Lite. $\mathcal{SRIQ}$ has gained considerable attention in the last years as the nominal-free fragment of the DL $\mathcal{SROIQ}$ underlying the new OWL 2 standard [3]. 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 $\mathcal{ZIQ}$.

$$\begin{array}{rcl} \mathit{mortal} & \sqsubseteq & \neg \mathit{deity} \\ \top & \sqsubseteq & \exists \mathit{HasFather}.\mathit{male} \sqcap \exists \mathit{HasMother}.\mathit{female} \\ \forall \mathit{HasParent}.\mathit{mortal} & \sqsubseteq & \mathit{mortal} \\ \mathit{deity} & \sqsubseteq & \forall \mathit{hasAncestor}.\mathit{deity} \end{array} \qquad \begin{array}{rcl} \mathit{HasMother} & \sqsubseteq & \mathit{HasParent} \\ \mathit{HasFather} & \sqsubseteq & \mathit{HasParent} \\ \mathit{HasParent} & \sqsubseteq & \mathit{hasAncestor} \\ \mathit{hasAncestor} \circ \mathit{hasAncestor} & \sqsubseteq & \mathit{hasAncestor} \end{array}$$

$$\begin{array}{l} \mathsf{Dis}(\mathit{HasMother}, \mathit{HasFather}) \\ \mathsf{Irr}(\mathit{HasParent}) \end{array}$$

Fig. 7. Some $\mathcal{SRIQ}$ axioms for a genealogy KB.

The most prominent feature of $\mathcal{SRIQ}$ are complex *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 allows 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 [5] and [61]. Let $\bar{\mathbf{R}} = \mathbf{R} \cup \{R^- \mid R \in \mathbf{R}\}$.

Definition 6.1 *($\mathcal{SRIQ}$ knowledge base).* A *$\mathcal{SRIQ}$ role inclusion axiom (SRIA)* is an expression of the form $R_1 \circ \cdots \circ R_n \sqsubseteq R$, $n \geqslant 1$, where R and all R_i, $1 \leqslant i \leqslant n$, are from $\bar{\mathbf{R}}$, and different from T if $n > 1$.

A set $\mathcal{R}$ of SRIAs is *regular*, if there exists a strict partial order $\prec$ on $\bar{\mathbf{R}}$ such that

(i) for every $R, R' \in \bar{\mathbf{R}}$, $\mathsf{Inv}(R) \prec R'$ iff $R \prec R'$, and
(ii) every SRIA in $\mathcal{R}$ is of one of the forms (a) $R \circ R \sqsubseteq R$, (b) $\mathsf{Inv}(R) \sqsubseteq R$, (c) $w \sqsubseteq R$, (d) $w \circ R \sqsubseteq R$, or (e) $R \circ w \sqsubseteq R$, where $w = R_1 \circ \cdots \circ R_n$ and $R_i \prec R$ for each $1 \leqslant i \leqslant n$.

For a given set $\mathcal{R}$ of SRIAs, the relation $\sqsubseteq_{\mathcal{R}}$ is the smallest relation such that

(i) $R \sqsubseteq_{\mathcal{R}} R$ for every $R \in \bar{\mathbf{R}}$ such that either R or $\mathsf{Inv}(R)$ occurs in $\mathcal{R}$, and
(ii) if $w_1 \circ R' \circ w_2 \sqsubseteq_{\mathcal{R}} R$, where $w_i = R^i_1 \circ \cdots \circ R^i_{n_2}$, $n_i \geqslant 0$ for $i = 1, 2$ and $\mathcal{R}$ contains either $w \sqsubseteq R'$ or $\mathsf{Inv}(w) \sqsubseteq R'$, where $w = R_1 \circ \cdots \circ R_n$, $n \geqslant 1$ and $\mathsf{Inv}(w) = \mathsf{Inv}(R_n) \circ \cdots \circ \mathsf{Inv}(R_1)$, then $w_1 \circ w \circ w_2 \sqsubseteq_{\mathcal{R}} R$ (i.e., $\sqsubseteq_{\mathcal{R}}$ is closed with respect to unfolding roles on the left hand side with SRIAs).

A role is *simple* in $\mathcal{R}$, if no roles $R_1, \ldots, R_n$, $n \geqslant 2$, exist 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 $R, R' \in \bar{\mathbf{R}}$.[6] It is *simple* w.r.t. to a set $\mathcal{R}$ of SRIAs, if all roles occurring in it are 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 $A \in \mathbf{C}$, $R, S \in \bar{\mathbf{R}}$, 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. Finally, a *$\mathcal{SRIQ}$ knowledge base* is a triple $\mathcal{K} = \langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle$ where $\mathcal{A}$ is a non-empty ABox, $\mathcal{T}$ is a TBox, and $\mathcal{R}$ is a $\mathcal{SRIQ}$ RBox as above.[7] $\square$

The semantics of $\mathcal{SRIQ}$ TBoxes and ABoxes is defined analogously to that of $\mathcal{ZIQ}$. 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 the symbol $\circ$ denotes composition of binary roles). An interpretation $\mathcal{I}$ *satisfies* (or *is a model of*) an RBox $\mathcal{R}$, if it satisfies all SRIAs and all assertions about roles in $\mathcal{R}$, written $\mathcal{I} \models \mathcal{R}$; it *satisfies* (or is a *model of*) a $\mathcal{SRIQ}$ KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle$, denoted $\mathcal{I} \models \mathcal{K}$, if $\mathcal{I} \models \mathcal{A}$, $\mathcal{I} \models \mathcal{T}$, and $\mathcal{I} \models \mathcal{R}$.

Example 6.2. In Fig. 7, we give part of a genealogy KB in $\mathcal{SRIQ}$ syntax. The first group in the left column is a TBox $\mathcal{T}'_g$ consisting of four SCIAs. They are almost identical to (some of) the CIAs of the KB $\mathcal{K}_g$ from Example 2.5, but in $\mathit{deity} \sqsubseteq \forall \mathit{HasParent}^*.\mathit{deity}$ the complex $\mathcal{ZIQ}$ role $\mathit{HasParent}^*$, which is not allowed in $\mathcal{SRIQ}$, has been replaced by the

[6] We use the term *assertion about roles* instead of *role assertions* as in [5], since the latter is often used to refer to ABox assertions of the form $R(a, b)$. In [5] also the assertion $\mathsf{Tra}(R)$, stating that R is transitive, is allowed. We omit this as it is expressible with the SRIA $R \circ R \sqsubseteq R$.

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

role *hasAncestor*. In the second group in the left column we give a set $\mathcal{R}^a_g$ of (two) assertions about roles, and the right column contains a set $\mathcal{R}^i_g$ of SRIAs.

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

We describe a rewriting that transforms a $\mathcal{SRIQ}$ KB $\mathcal{K}$ into a $\mathcal{ZIQ}$ KB $\Psi(\mathcal{K})$ in a way that will allow us to exploit our automata-based algorithms for reasoning in $\mathcal{SRIQ}$. It builds on the fact that the restriction to regular sets of SRIAs, which is crucial for the decidability of $\mathcal{SRIQ}$, ensures that the implications between roles define a regular language. Hence they can be simulated using the regular expressions over roles present in $\mathcal{ZIQ}$. More precisely:

Lemma 6.3. *(See [59].) If $\mathcal{R}$ is a regular set of SRIAs, then for each $R \in \bar{\mathbf{R}}$ occurring in $\mathcal{K}$, the set $L_{\mathcal{R}}(R) = \{R_1 \circ \cdots \circ R_n \mid R_1 \circ \cdots \circ R_n \sqsubseteq_{\mathcal{R}} R\}$ is a regular language.*

Furthermore, the authors of [59] show how to construct a finite state automaton representing $L_{\mathcal{R}}(R)$, which is equivalent to a regular expression $\rho^{\mathcal{R}}(R)$ over the alphabet $\bar{\mathbf{R}}$. That is, $L_{\mathcal{R}}(R)$ can be written as a $\mathcal{ZIQ}$ role $\rho^{\mathcal{R}}(R)$. In particular, if a role S is simple in $\mathcal{R}$, the resulting expression is $\rho^{\mathcal{R}}(S) = \bigcup_{S' \sqsubseteq_{\mathcal{R}} S} S'$, which is a simple $\mathcal{ZIQ}$ role. It is easy to see that $R^{\mathcal{I}} \subseteq (\rho^{\mathcal{R}}(R))^{\mathcal{I}}$ in every interpretation $\mathcal{I}$, since $R \sqsubseteq_{\mathcal{R}} R$ trivially holds. The converse, $(\rho^{\mathcal{R}}(R))^{\mathcal{I}} \subseteq R^{\mathcal{I}}$, holds in every interpretation in which $w \sqsubseteq_{\mathcal{R}} R$ implies $w^{\mathcal{I}} \subseteq R^{\mathcal{I}}$, that is, in every model of $\mathcal{R}$. Hence we obtain the following.

Corollary 6.4. *Given a regular set $\mathcal{R}$ of SRIAs and $R \in \bar{\mathbf{R}}$, we can construct a $\mathcal{ZIQ}$ role $\rho^{\mathcal{R}}(R)$ such that $(\rho^{\mathcal{R}}(R))^{\mathcal{I}} = \bigcup_{R_1 \circ \cdots \circ R_n \sqsubseteq_{\mathcal{R}} R}(R_1 \circ \cdots \circ R_n)^{\mathcal{I}}$ in every interpretation $\mathcal{I}$, and hence $(\rho^{\mathcal{R}}(R))^{\mathcal{I}} = R^{\mathcal{I}}$ whenever $\mathcal{I} \models \mathcal{R}$. Moreover, $\rho^{\mathcal{R}}(R)$ is simple whenever R is simple in $\mathcal{R}$.*

Example 6.5. Given the SRIAs $\mathcal{R}^i_g$ in Fig. 7, we can construct the following $\mathcal{ZIQ}$ roles for the role names in $\mathcal{R}^i_g$: $\rho^{\mathcal{R}}(\mathit{HasMother}) = \mathit{HasMother}$, $\rho^{\mathcal{R}}(\mathit{HasFather}) = \mathit{HasFather}$, $\rho^{\mathcal{R}}(\mathit{HasParent}) = \mathit{HasMother} \cup \mathit{HasFather} \cup \mathit{HasParent}$, and $\rho^{\mathcal{R}}(\mathit{hasAncestor}) = (\mathit{HasMother} \cup \mathit{HasFather} \cup \mathit{HasParent} \cup \mathit{hasAncestor})^{+}$. $\square$

In what follows, we assume a fixed regular set $\mathcal{R}$ of SRIAs, and for each $R \in \bar{\mathbf{R}}$, $\rho^{\mathcal{R}}(R)$ is an arbitrary but fixed regular expression as above. The rewriting Ψ of a $\mathcal{SRIQ}$ KB exploits Lemma 6.3 above, and has the following steps:

1. *TBox rewriting*. In every concept occurring in the TBox, we replace the role R by the regular expression $\rho^{\mathcal{R}}(R)$. We show below that this ensures that the interpretation of concepts in the rewritten TBox respects the restrictions that arise from the SRIAs.
2. *ABox rewriting*. Similarly as above, we replace in every concept assertion of the form $C(a)$ each role R occurring in C by $\rho^{\mathcal{R}}(R)$. We furthermore remove the *negated role membership assertions* $\neg S(a,b)$, which are not allowed in $\mathcal{ZIQ}$, and simulate them using a fresh role name $P_{\neg S}$ for each role S. We replace $\neg S(a,b)$ by $P_{\neg S}(a,b)$, and add BRIAs which ensure that $P_{\neg S}$ is interpreted as a role that is disjoint from S.
3. *RBox rewriting*. SRIAs are dropped, and assertions about roles are simulated using BRIAs and CIAs.

More formally, the rewriting $\Psi(\mathcal{K})$ is as follows.

Definition 6.6. Consider a $\mathcal{SRIQ}$ KB $\langle \mathcal{T}, \mathcal{R} \cup \mathcal{R}^a, \mathcal{A}\rangle$, where $\mathcal{R}^a$ is a set of assertions about roles.

(1) For a $\mathcal{SRIQ}$ concept C, we denote by $\Psi^{\mathcal{R}}(C)$ the $\mathcal{ZIQ}$ concept that results from replacing every occurrence of a role R by $\rho^{\mathcal{R}}(R)$. Then $\Psi^{\mathcal{R}}(\mathcal{T})$ is the $\mathcal{ZIQ}$ TBox $\{\Psi^{\mathcal{R}}(C) \sqsubseteq \Psi^{\mathcal{R}}(D) \mid C \sqsubseteq D \in \mathcal{T}\}$.

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

(3) $\Psi^{\mathcal{R}}(\mathcal{R}^a)$ is the $\mathcal{ZIQ}$ TBox

$$\Psi^{\mathcal{R}}\big(\mathcal{R}^a\big) = \big\{\mathsf{Inv}\big(\rho^{\mathcal{R}}(R)\big) \sqsubseteq R \;\big|\; \mathsf{Sym}(R) \in \mathcal{R}^a\big\} \cup \big\{\top \sqsubseteq \exists R.\mathsf{Self} \;\big|\; \mathsf{Ref}(R) \in \mathcal{R}^a\big\}$$
$$\cup \big\{\exists \rho^{\mathcal{R}}(R).\mathsf{Self} \sqsubseteq \bot \;\big|\; \mathsf{Irr}(R) \in \mathcal{R}^a\big\} \cup \big\{\rho^{\mathcal{R}}(R) \cap \rho^{\mathcal{R}}\big(R'\big) \sqsubseteq \mathsf{B} \;\big|\; \mathsf{Dis}\big(R, R'\big) \in \mathcal{R}^a\big\},$$

where the notation $\mathsf{Inv}(R)$ (defined for simple roles in Section 3.2) is extended to arbitrary $\mathcal{ZIQ}$ roles as follows: $\mathsf{Inv}(R \cup R') = \mathsf{Inv}(R) \cup \mathsf{Inv}(R')$, $\mathsf{Inv}(R \circ R') = \mathsf{Inv}(R') \circ \mathsf{Inv}(R)$, $\mathsf{Inv}(R^*) = \mathsf{Inv}(R)^*$, and $\mathsf{Inv}(id(C)) = id(C)$.

(4) Finally, $\Psi(\mathcal{K})$ is the $\mathcal{ZIQ}$ KB $\langle \Psi^{\mathcal{R}}(\mathcal{T}) \cup \mathcal{T}^{\mathcal{R}}_{\mathcal{A}} \cup \Psi^{\mathcal{R}}(\mathcal{R}^a), \Psi^{\mathcal{R}}(\mathcal{A})\rangle$. $\square$

The rewriting of ABoxes and TBoxes preserves the semantics in the models of $\mathcal{R}$. The final step, rewriting assertions about roles, has the stronger property of preserving equivalence in all models. Hence, by suitably interpreting the added role names $P_{\neg S}$, all models of $\Psi(\mathcal{K})$ become models of $\mathcal{K}$. 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 such $\mathcal{I}$ can be transformed into a model of the SRIAs by adding all implied pairs of individuals to the extension of the roles.

Proposition 6.7. *Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{R} \cup \mathcal{R}^a, \mathcal{A}\rangle$ be a $\mathcal{SRIQ}$ KB as above. Then:*

1. *For every interpretation $\mathcal{I}$, if $\mathcal{I} \models \mathcal{K}$, then there is an interpretation $\mathcal{I}'$ such that $\mathcal{I}' \models \Psi(\mathcal{K})$, and $\mathcal{I}'$ coincides with $\mathcal{I}$ on all concepts and roles over the signature of $\mathcal{K}$.*
2. *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 is identical to $\mathcal{I}$ otherwise. Then $\mathcal{I}' \models \mathcal{K}$.*

Proof. For the first item, we start by proving the following three claims, for an arbitrary interpretation $\mathcal{I}$:

(i) $\mathcal{I} \models \mathcal{R}$ implies $C^{\mathcal{I}} = (\Psi^{\mathcal{R}}(C))^{\mathcal{I}}$ for each $\mathcal{SRIQ}$ concept C; hence, if $\mathcal{I} \models \mathcal{R}$ and $\mathcal{I} \models \mathcal{T}$ then $\mathcal{I} \models \Psi^{\mathcal{R}}(\mathcal{T})$.
(ii) If $\mathcal{I} \models \mathcal{R}$ and $\mathcal{I} \models \mathcal{A}$, then there is an interpretation $\mathcal{I}'$ such that $\mathcal{I}' \models \Psi^{\mathcal{R}}(\mathcal{A})$, $\mathcal{I}' \models \mathcal{T}^{\mathcal{R}}_{\mathcal{A}}$, and $\mathcal{I}'$ coincides with $\mathcal{I}$ on all concepts and roles over the signature of $\mathcal{K}$.
(iii) If $\mathcal{I} \models \mathcal{R}$ and $\mathcal{I} \models \mathcal{R}^a$, then $\mathcal{I} \models \Psi^{\mathcal{R}}(\mathcal{R}^a)$.

Assume $\mathcal{I} \models \mathcal{R}$. To show item (i), we start by showing that $C^{\mathcal{I}} = (\Psi^{\mathcal{R}}(C))^{\mathcal{I}}$ by structural induction on C. The claim is trivial for atomic concepts and for concepts of the forms $\neg D$, $D \sqcap D'$, and $D \sqcup D'$, since $\Psi^{\mathcal{R}}(C)$ coincides with $\neg\Psi^{\mathcal{R}}(D)$, $\Psi^{\mathcal{R}}(D) \sqcap \Psi^{\mathcal{R}}(D')$, and $\Psi^{\mathcal{R}}(D) \sqcup \Psi^{\mathcal{R}}(D')$, respectively. For C of the form $\exists R.D$, we have $\Psi^{\mathcal{R}}(C) = \exists\rho^{\mathcal{R}}(R).\Psi^{\mathcal{R}}(D)$. Then $(\exists R.D)^{\mathcal{I}} \subseteq (\exists\rho^{\mathcal{R}}(R).\Psi^{\mathcal{R}}(D))^{\mathcal{I}}$ follows from the induction hypothesis and $R^{\mathcal{I}} \subseteq (\rho^{\mathcal{R}}(R))^{\mathcal{I}}$. For the converse, $(\exists\rho^{\mathcal{R}}(R).\Psi^{\mathcal{R}}(D))^{\mathcal{I}} \subseteq (\exists R.D)^{\mathcal{I}}$, we need to use the induction hypothesis and the assumption that $\mathcal{I} \models \mathcal{R}$ to infer $(\rho^{\mathcal{R}}(R))^{\mathcal{I}} \subseteq R^{\mathcal{I}}$. The cases for C of the forms $\geqslant nS.D$ and $\exists S.\mathsf{Self}$ are analogous. For C of the form $\forall R.D$ or $\leqslant nS.D$ the proof is similar, but now $\Psi^{\mathcal{R}}(C)^{\mathcal{I}} \subseteq C^{\mathcal{I}}$ follows directly from the induction hypothesis and $R^{\mathcal{I}} \subseteq (\rho^{\mathcal{R}}(R))^{\mathcal{I}}$, while showing $C^{\mathcal{I}} \subseteq \Psi^{\mathcal{R}}(C)^{\mathcal{I}}$ requires the assumption that $\mathcal{I} \models \mathcal{R}$ and hence $(\rho^{\mathcal{R}}(R))^{\mathcal{I}} \subseteq R^{\mathcal{I}}$. Once we have shown $C^{\mathcal{I}} = (\Psi^{\mathcal{R}}(C))^{\mathcal{I}}$, it follows that $\mathcal{I} \models \mathcal{T}$ implies $\mathcal{I} \models \Psi^{\mathcal{R}}(\mathcal{T})$, and item (i) holds.

For item (ii), assume $\mathcal{I} \models \mathcal{R}$ and $\mathcal{I} \models \mathcal{A}$. We let $\mathcal{I}'$ be the interpretation that has $(P_{\neg S})^{\mathcal{I}'} = (\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}) \setminus S^{\mathcal{I}}$ for each fresh role name $P_{\neg S}$ in $\Psi^{\mathcal{R}}(\mathcal{A})$, and is identical to $\mathcal{I}$ otherwise. As $\mathcal{I}$ and $\mathcal{I}'$ coincide on all symbols in $\mathcal{K}$, $\mathcal{I}' \models \mathcal{R}$ and $\mathcal{I}' \models \mathcal{A}$. To show $\mathcal{I}' \models \Psi^{\mathcal{R}}(\mathcal{A})$, we only need to show that $\mathcal{I}'$ satisfies each assertion from $\Psi^{\mathcal{R}}(\mathcal{A})$. We distinguish three cases:

- The assertion is of the form $\Psi^{\mathcal{R}}(C)(a)$, and it replaced some $C(a) \in \mathcal{A}$ in step (2i). As $\mathcal{I}' \models \mathcal{R}$, $C^{\mathcal{I}'} = \Psi^{\mathcal{R}}(C)^{\mathcal{I}'}$, and as $\mathcal{I}' \models \mathcal{A}$ and $C(a) \in \mathcal{A}$, $a^{\mathcal{I}'} \in C^{\mathcal{I}'}$. Hence $a^{\mathcal{I}'} \in \Psi^{\mathcal{R}}(C)^{\mathcal{I}'}$ and $\mathcal{I}'$ satisfies $\Psi^{\mathcal{R}}(C)(a)$.
- If the assertion is of the form $R(a,b)$, then $R(a,b) \in \mathcal{A}$ (i.e., it is an assertion from the original ABox). Then $\mathcal{I}'$ satisfies $R(a,b)$ because $\mathcal{I}' \models \mathcal{A}$.
- If the assertion is of the form $P_{\neg S}(a,b)$, then it replaced some $\neg S(a,b)$ in step (2ii). Since $\mathcal{I} \models \mathcal{A}$ and $\neg S(a,b) \in \mathcal{A}$, it follows that $(a^{\mathcal{I}}, b^{\mathcal{I}}) \notin S^{\mathcal{I}}$, and hence $(a^{\mathcal{I}'}, b^{\mathcal{I}'}) \in P_{\neg S}{}^{\mathcal{I}'}$ by construction of $\mathcal{I}'$. Hence $\mathcal{I}'$ satisfies $P_{\neg S}(a,b)$

This proves $\mathcal{I}' \models \Psi^{\mathcal{R}}(\mathcal{A})$. To show $\mathcal{I}' \models \mathcal{T}^{\mathcal{R}}_{\mathcal{A}}$, we consider an arbitrary $P_{\neg S} \sqcap \rho^{\mathcal{R}}(S) \sqsubseteq \mathsf{B}$ with $\neg S(a,b)$ in $\mathcal{A}$. Since $\mathcal{I} \models \mathcal{R}$, by Corollary 6.4 $(x,y) \in \rho^{\mathcal{R}}(S)^{\mathcal{I}}$ implies $(x,y) \in S^{\mathcal{I}}$, and by construction of $\mathcal{I}'$, $(x,y) \notin P_{\neg S}{}^{\mathcal{I}'}$. As $\rho^{\mathcal{R}}(S)^{\mathcal{I}} = \rho^{\mathcal{R}}(S)^{\mathcal{I}'}$, this shows that $(x,y) \in \rho^{\mathcal{R}}(S)^{\mathcal{I}'}$ implies $(x,y) \notin P_{\neg S}{}^{\mathcal{I}'}$ for every pair $(x,y) \in \Delta^{\mathcal{I}'} \times \Delta^{\mathcal{I}'}$, so $\mathcal{I}'$ satisfies $P_{\neg S} \sqcap \rho^{\mathcal{R}}(S) \sqsubseteq \mathsf{B}$ and $\mathcal{I}' \models \mathcal{T}^{\mathcal{R}}_{\mathcal{A}}$ as desired.

For item (iii), we assume $\mathcal{I} \models \mathcal{R}$ and $\mathcal{I} \models \mathcal{R}^a$, and show that $\mathcal{I}$ satisfies the different forms of inclusions in $\Psi^{\mathcal{R}}(\mathcal{R}^a)$:

- $\mathsf{Inv}(\rho^{\mathcal{R}}(R)) \sqsubseteq R$ with $\mathsf{Sym}(R) \in \mathcal{R}^a$. Consider an arbitrary $(x,y) \in \mathsf{Inv}(\rho^{\mathcal{R}}(R))^{\mathcal{I}}$. Then $(y,x) \in \rho^{\mathcal{R}}(R)^{\mathcal{I}}$. Since $\mathcal{I} \models \mathcal{R}$, from Corollary 6.4 we have $(y,x) \in R^{\mathcal{I}}$. From this and the fact that $\mathcal{I}$ satisfies $\mathsf{Sym}(R)$, it follows that $(x,y) \in R^{\mathcal{I}}$ and $\mathcal{I}$ satisfies the inclusion as desired.
- $\top \sqsubseteq \exists R.\mathsf{Self}$ with $\mathsf{Ref}(R) \in \mathcal{R}^a$. Since $\mathcal{I}$ satisfies $\mathsf{Ref}(R)$, $\exists R.\mathsf{Self}^{\mathcal{I}} = \Delta^{\mathcal{I}}$ and $\mathcal{I}$ satisfies $\top \sqsubseteq \exists R.\mathsf{Self}$ as desired.
- $\exists\rho^{\mathcal{R}}(R).\mathsf{Self} \sqsubseteq \bot$ with $\mathsf{Irr}(R) \in \mathcal{R}^a$. This case is similar to the previous one: since $\mathcal{I}$ satisfies $\mathsf{Irr}(R)$, $\exists R.\mathsf{Self}^{\mathcal{I}} = \emptyset$. Since $\mathcal{I} \models \mathcal{R}$, from Corollary 6.4 it follows that $\exists\rho^{\mathcal{R}}(R).\mathsf{Self}^{\mathcal{I}} = \emptyset$, and $\mathcal{I}$ satisfies $\exists\rho^{\mathcal{R}}(R).\mathsf{Self} \sqsubseteq \bot$ as desired.
- $\rho^{\mathcal{R}}(R) \sqcap \rho^{\mathcal{R}}(R') \sqsubseteq \mathsf{B}$ with $\mathsf{Dis}(R,R') \in \mathcal{R}^a$. We consider an arbitrary pair $(x,y) \in \rho^{\mathcal{R}}(R)^{\mathcal{I}}$. Since $\mathcal{I} \models \mathcal{R}$, from Corollary 6.4 it follows that $(x,y) \in R^{\mathcal{I}}$. Since $\mathcal{I}$ satisfies $\mathsf{Dis}(R,R')$, this means $(x,y) \notin R'^{\mathcal{I}}$. We can now use again Corollary 6.4 to infer $(x,y) \notin \rho^{\mathcal{R}}(R')^{\mathcal{I}}$. Hence $\mathcal{I}$ satisfies $\rho^{\mathcal{R}}(R) \sqcap \rho^{\mathcal{R}}(R') \sqsubseteq \mathsf{B}$.

This concludes the proof of item (iii). Item (1) in the claim now follows easily: consider an arbitrary $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{K}$, and let $\mathcal{I}'$ be as in the proof of item (ii), i.e., $(P_{\neg S})^{\mathcal{I}'} = (\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}) \setminus S^{\mathcal{I}}$ for each fresh role name $P_{\neg S}$ in $\Psi^{\mathcal{R}}(\mathcal{A})$ and $\mathcal{I}'$ is identical to $\mathcal{I}$ otherwise. We have shown in item (ii) that $\mathcal{I}' \models \Psi^{\mathcal{R}}(\mathcal{A})$ and $\mathcal{I}' \models \mathcal{T}^{\mathcal{R}}_{\mathcal{A}}$. Since $\mathcal{I}'$ coincides with $\mathcal{I}$ on

$$\begin{aligned}
mortal &\sqsubseteq \neg deity\\
\top &\sqsubseteq \exists HasFather.male \sqcap \exists HasMother.female\\
\forall(HasMother \cup HasFather \cup HasParent).mortal &\sqsubseteq mortal\\
deity &\sqsubseteq \forall(HasMother \cup HasFather \cup HasParent \cup hasAncestor)^{+}.deity\\
HasMother \cap HasFather &\sqsubseteq \mathsf{B}\\
\exists(HasMother \cup HasFather \cup HasParent).\mathsf{Self} &\sqsubseteq \bot
\end{aligned}$$

Fig. 8. Axioms for a genealogy KB translated to $\mathcal{ZIQ}$.

all concepts and roles over the signature of $\mathcal{K}$ and $\mathcal{I} \models \mathcal{K}$, it follows that $\mathcal{I}' \models \mathcal{R}$, $\mathcal{I}' \models \mathcal{T}$ and $\mathcal{I}' \models \mathcal{R}^a$, and by items (i) and (iii), we can conclude $\mathcal{I}' \models \Psi^{\mathcal{R}}(\mathcal{T})$ and $\mathcal{I}' \models \Psi^{\mathcal{R}}(\mathcal{R}^a)$. Hence $\mathcal{I}' \models \Psi^{\mathcal{R}}(\mathcal{K})$.

For the second item of Proposition 6.7, take an arbitrary $\mathcal{I}$ and let $\mathcal{I}'$ be as in the claim, i.e., $R^{\mathcal{I}'} = (\rho^{\mathcal{R}}(R))^{\mathcal{I}}$ for each role $R \in \mathbf{R}$ occurring in $\mathcal{K}$ and $\mathcal{I}'$ is identical to $\mathcal{I}$ otherwise. Observe that since $R^{\mathcal{I}'} = (\rho^{\mathcal{R}}(R))^{\mathcal{I}}$ by construction, $\mathcal{I}' \models \mathcal{R}$ holds. Now we assume $\mathcal{I} \models \Psi(\mathcal{K})$ and show that $\mathcal{I}' \models \mathcal{K}$. First, to show that $\mathcal{I}' \models \mathcal{T}$, we observe that since $\mathcal{I}' \models \mathcal{R}$, we have $R^{\mathcal{I}'} = (\rho^{\mathcal{R}}(R))^{\mathcal{I}'}$ for every R occurring in $\mathcal{K}$. From this and $R^{\mathcal{I}'} = (\rho^{\mathcal{R}}(R))^{\mathcal{I}}$, it follows that $(\rho^{\mathcal{R}}(R))^{\mathcal{I}'} = (\rho^{\mathcal{R}}(R))^{\mathcal{I}}$. It is then easy to show that for all concepts C, $\Psi^{\mathcal{R}}(C)^{\mathcal{I}} = \Psi^{\mathcal{R}}(C)^{\mathcal{I}'}$. Hence $\mathcal{I} \models \Psi^{\mathcal{R}}(\mathcal{T})$ implies $\mathcal{I}' \models \Psi^{\mathcal{R}}(\mathcal{T})$, and this together with $\mathcal{I}' \models \mathcal{R}$ and item (i) above implies $\mathcal{I}' \models \mathcal{T}$.

It is only left to show that $\mathcal{I}' \models \mathcal{A}$ and $\mathcal{I}' \models \mathcal{R}^a$. For the former, we again distinguish between the different forms of assertions in $\mathcal{A}$:

- For an assertion of the form $C(a) \in \mathcal{A}$, we again use the fact that $\Psi^{\mathcal{R}}(C)^{\mathcal{I}} = \Psi^{\mathcal{R}}(C)^{\mathcal{I}'}$, and the fact that $\mathcal{I}' \models \mathcal{R}$ implies $C^{\mathcal{I}'} = \Psi^{\mathcal{R}}(C)^{\mathcal{I}'}$ by item (i) above. It follows from $\mathcal{I} \models \Psi^{\mathcal{R}}(\mathcal{A})$ that $a^{\mathcal{I}} \in \Psi^{\mathcal{R}} C^{\mathcal{I}}$, so $a^{\mathcal{I}'} \in \Psi^{\mathcal{R}} C^{\mathcal{I}'}$, and $a^{\mathcal{I}'} \in C^{\mathcal{I}'}$. Hence $\mathcal{I}' \models C(a)$ as desired.
- For an assertion of the form $R(a,b) \in \mathcal{A}$, we have that $R(a,b) \in \Psi^{\mathcal{R}}(\mathcal{A})$ and $\mathcal{I} \models \Psi^{\mathcal{R}}(\mathcal{A})$, so $\mathcal{I} \models R(a,b)$ and $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$, which implies $(a^{\mathcal{I}'}, b^{\mathcal{I}'}) \in R^{\mathcal{I}'}$, hence $\mathcal{I}' \models R(a,b)$.
- For an assertion of the form $\neg S(a,b) \in \mathcal{A}$, we have that $P_{\neg S}(a,b) \in \Psi^{\mathcal{R}}(\mathcal{A})$ and $\mathcal{I} \models \Psi^{\mathcal{R}}(\mathcal{A})$, so $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in P_{\neg S}{}^{\mathcal{I}}$. Moreover, since $\mathcal{I} \models \mathcal{T}_{\mathcal{A}}^{\mathcal{R}}$ and $P_{\neg S} \cap \rho^{\mathcal{R}}(S) \sqsubseteq \mathsf{B} \in \mathcal{T}_{\mathcal{A}}^{\mathcal{R}}$, $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in P_{\neg S}{}^{\mathcal{I}}$ implies $(a^{\mathcal{I}}, b^{\mathcal{I}}) \notin \rho^{\mathcal{R}}(S)^{\mathcal{I}}$. Hence $(a^{\mathcal{I}'}, b^{\mathcal{I}'}) \notin S^{\mathcal{I}'}$, and $\mathcal{I}' \models \neg S(a,b)$.

This proves $\mathcal{I}' \models \mathcal{A}$. Finally, to show $\mathcal{I}' \models \mathcal{R}^a$, we consider the different forms of assertions about roles in $\mathcal{R}^a$:

- For $\mathsf{Sym}(R) \in \mathcal{R}^a$, we have $\mathsf{Inv}(\rho^{\mathcal{R}}(R)) \sqsubseteq R \in \Psi^{\mathcal{R}}(\mathcal{R}^a)$. Assume $(x,y) \in R^{\mathcal{I}'}$. Then $(x,y) \in \rho^{\mathcal{R}}(R)^{\mathcal{I}}$, so $(y,x) \in \mathsf{Inv}(\rho^{\mathcal{R}}(R))^{\mathcal{I}}$, and as $\mathcal{I}$ satisfies $\mathsf{Inv}(\rho^{\mathcal{R}}(R)) \sqsubseteq R$, we have $(y,x) \in R^{\mathcal{I}}$, and thus $(y,x) \in R^{\mathcal{I}'}$. This shows that $R^{\mathcal{I}'}$ is symmetric, so $\mathcal{I}'$ satisfies $\mathsf{Sym}(R)$.
- For $\mathsf{Ref}(R) \in \mathcal{R}^a$, we have $\top \sqsubseteq \exists R.\mathsf{Self} \in \Psi^{\mathcal{R}}(\mathcal{R}^a)$. Consider an arbitrary $x \in \Delta^{\mathcal{I}}$. Since $\mathcal{I}$ satisfies $\top \sqsubseteq \exists R.\mathsf{Self}$, $x \in \exists R.\mathsf{Self}^{\mathcal{I}}$, hence $(x,x) \in R^{\mathcal{I}}$ and $(x,x) \in R^{\mathcal{I}'}$. This shows that $R^{\mathcal{I}'}$ is reflexive and $\mathcal{I}'$ satisfies $\mathsf{Ref}(R)$.
- For $\mathsf{Irr}(R) \in \mathcal{R}^a$, we have $\exists\rho^{\mathcal{R}}(R).\mathsf{Self} \sqsubseteq \bot \in \Psi^{\mathcal{R}}(\mathcal{R}^a)$. Consider an arbitrary $x \in \Delta^{\mathcal{I}}$, and assume towards a contradiction that $(x,x) \in R^{\mathcal{I}'}$. Then $(x,x) \in \rho^{\mathcal{R}}(R)^{\mathcal{I}}$, which implies $x \in \exists\rho^{\mathcal{R}}(R).\mathsf{Self}^{\mathcal{I}}$. But since $\mathcal{I} \models \Psi^{\mathcal{R}}(\mathcal{R}^a)$ and $\exists\rho^{\mathcal{R}}(R).\mathsf{Self} \sqsubseteq \bot \in \Psi^{\mathcal{R}}(\mathcal{R}^a)$, we have $\exists\rho^{\mathcal{R}}(R).\mathsf{Self}^{\mathcal{I}} = \emptyset$, which contradicts $x \in \exists\rho^{\mathcal{R}}(R).\mathsf{Self}^{\mathcal{I}}$. This shows that $R^{\mathcal{I}'}$ is irreflexive and $\mathcal{I}'$ satisfies $\mathsf{Irr}(R)$.
- For $\mathsf{Dis}(R,R') \in \mathcal{R}^a$, we have $\rho^{\mathcal{R}}(R) \cap \rho^{\mathcal{R}}(R') \sqsubseteq \mathsf{B} \in \Psi^{\mathcal{R}}(\mathcal{R}^a)$. We consider an arbitrary pair $(x,y) \in R^{\mathcal{I}'}$. Then $(x,y) \in \rho^{\mathcal{R}}(R)^{\mathcal{I}}$, and since $\mathcal{I}$ satisfies $\rho^{\mathcal{R}}(R) \cap \rho^{\mathcal{R}}(R') \sqsubseteq \mathsf{B}$, $(x,y) \notin \rho^{\mathcal{R}}(R')^{\mathcal{I}}$, so $(x,y) \notin R'^{\mathcal{I}'}$. This shows that $R^{\mathcal{I}'}$ and $R'^{\mathcal{I}'}$ are disjoint, and $\mathcal{I}'$ satisfies $\mathsf{Dis}(R,R') \in \mathcal{R}^a$.

This proves $\mathcal{I}' \models \mathcal{R}^a$. We have shown that $\mathcal{I}' \models \mathcal{T}$, $\mathcal{I}' \models \mathcal{R}$, $\mathcal{I}' \models \mathcal{R}^a$, and $\mathcal{I}' \models \mathcal{A}$, so $\mathcal{I}' \models \mathcal{K}$. This concludes the proof of the claim. $\square$

As a consequence, we have a reduction from KB satisfiability in $\mathcal{SRIQ}$ to KB satisfiability in $\mathcal{ZIQ}$.

Corollary 6.8. *For every $\mathcal{SRIQ}$ KB $\mathcal{K}$, $\mathcal{K}$ is satisfiable iff $\Psi(\mathcal{K})$ is satisfiable.*

In the rewriting above, we have replaced R by $\rho^{\mathcal{R}}(R)$ for all roles R. It would also be possible to do this replacement only for non-simple roles, while leaving simple roles untouched and keeping in the resulting $\mathcal{ZIQ}$ TBox the corresponding inclusions (note that every SRIA with a simple role on the right hand side is syntactically a $\mathcal{ZIQ}$ BRIA). We have also replaced R by $\rho^{\mathcal{R}}(R)$ everywhere in $\mathcal{T}$ and $\mathcal{A}$. This is not strictly necessary, and an alternative translation from $\mathcal{SRIQ}$ to $\mathcal{ZIQ}$ can be defined by replacing R by $\rho^{\mathcal{R}}(R)$ only in concepts of the forms $\exists R.C$, $\exists R.\mathsf{Self}$, and $\geqslant nR.C$ on the left hand side of inclusions, and in concepts of the forms $\forall R.C$ and $\leqslant nR.C$ on the right hand side of inclusions and in ABox assertions. Such a translation would resemble more the one in [61] and similar ones. However, it would require us to rewrite the KB into NNF, and to eliminate complex concepts C that occur inside concepts of the forms $\exists R.C$, $\forall R.C \geqslant nR.C$, or $\leqslant nR.C$ (which can be easily done by introducing a fresh concept name A_C and adding CIAs $A_C \sqsubseteq C$, $C \sqsubseteq A_C$ to the TBox). In such a translation the negated role assertions in the ABox and the assertions about roles in the RBox can be handled exactly as in Definition 6.6.

Example 6.9. Fig. 8 shows the result of applying the transformation Ψ to the axioms in Fig. 7, using the regular expressions in Example 6.5. Note that the result is a $\mathcal{ZIQ}$ TBox containing one CIA and one BRIA. $\square$

6.2. Deciding KB satisfiability

By Corollary 6.8, the automata algorithm in Section 4 can be used to decide the satisfiability of $\mathcal{SRIQ}$ knowledge bases; under unary number encoding, the resulting algorithm is worst-case optimal. Let $\rho^{\mathcal{R}}_{max}$ be from the set $\{\rho^{\mathcal{R}}_R \mid R \in \bar{\mathbf{R}}_{\mathcal{R}}\}$ such that its length is maximal, i.e., $|\rho^{\mathcal{R}}_{max}| = \max_{R\in\bar{\mathbf{R}}_{\mathcal{R}}} |\rho^{\mathcal{R}}_R|$. Each step of the rewriting Ψ is clearly polynomial in the size of $\mathcal{A}$, $\mathcal{T}$, and $\rho^{\mathcal{R}}_{max}$; however, the size of $\rho^{\mathcal{R}}_{max}$ can be exponential in the size of $\mathcal{R}$ [59].

Theorem 6.10. *Under unary number encoding, the satisfiability of a given $\mathcal{SRIQ}$ knowledge base $\mathcal{K} = \langle\mathcal{T}, \mathcal{R}, \mathcal{A}\rangle$ is decidable in time exponential in the combined size of $\mathcal{T}$, $\mathcal{A}$, and $\rho^{\mathcal{R}}_{max}$, and in time double exponential in the size of $\mathcal{K}$.*

As shown in [61], KB satisfiability for $\mathcal{SRIQ}$ is 2ExpTime-hard. Hence our bound is optimal under the assumption of unary number encoding. Note that the blow-up in complexity w.r.t. $\mathcal{ZIQ}$ is due to the size of $\rho^{\mathcal{R}}_{max}$, and that the algorithm is single exponential whenever $\rho^{\mathcal{R}}_{max}$ has size polynomial in $\mathcal{R}$, e.g., for *simple role hierarchies* as defined in [59]. This compares well to the $\mathcal{SRIQ}$ algorithm given in [5], which, even for such hierarchies may require non-deterministic double exponential time in the size of $\mathcal{K}$.

6.3. Query answering in $\mathcal{SRIQ}$

We also obtain an algorithm to decide query entailment in $\mathcal{SRIQ}$. To this end, we rewrite a P2RPQ q over $\mathcal{K}$ into a new query over $\Psi(\mathcal{K})$ in a way such that query entailment is preserved.

Definition 6.11. For every P2RPQ q over a $\mathcal{SRIQ}$ knowledge base $\mathcal{K} = \langle\mathcal{T}, \mathcal{R}, \mathcal{A}\rangle$, $\Psi^{\mathcal{R}}(q)$ is the P2RPQ obtained from q by replacing each role R with $\rho^{\mathcal{R}}(R)$. $\square$

Note that $\Psi^{\mathcal{R}}(q)$ may contain regular expressions even when 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 6.12. *Let q be a P2RPQ over a $\mathcal{SRIQ}$ knowledge base $\mathcal{K} = \langle\mathcal{T}, \mathcal{R}, \mathcal{A}\rangle$. Then $\mathcal{K} \models q$ iff $\Psi(K) \models \Psi^{\mathcal{R}}(q)$.*

Proof. For the (*Only If*) direction, suppose $\mathcal{K} \models q$ and consider an interpretation $\mathcal{I}$ such that $\mathcal{I} \models \Psi(\mathcal{K})$. Let $\mathcal{I}'$ be the variant of $\mathcal{I}$ in which each R is interpreted as $(\rho^{\mathcal{R}}(R))^{\mathcal{I}}$. Then, by item (2) of Proposition 6.7, $\mathcal{I}' \models \mathcal{K}$, hence $\mathcal{I}' \models q$. As every match for q in $\mathcal{I}'$ is a match for $\Psi^{\mathcal{R}}(q)$ in $\mathcal{I}$, it follows $\mathcal{I} \models \Psi^{\mathcal{R}}(q)$; hence, $\Psi(\mathcal{K}) \models \Psi^{\mathcal{R}}(q)$.

For the (*If*) direction, suppose $\Psi(\mathcal{K}) \models \Psi^{\mathcal{R}}(q)$. Consider an interpretation $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{K}$. By item (1) of Proposition 6.7, we know that $\mathcal{I}' \models \Psi(\mathcal{K})$ for some $\mathcal{I}'$ that only differs from $\mathcal{I}$ in the interpretation of the fresh role names $P_{\neg S}$ that do not occur in $\mathcal{K}$ or q, which means $\mathcal{I}' \models \Psi^{\mathcal{R}}(q)$. Hence, it follows $\mathcal{I} \models \Psi^{\mathcal{R}}(q)$. Since $\mathcal{I} \models \mathcal{R}$, Corollary 6.4 implies that $R^{\mathcal{I}} = (\rho^{\mathcal{R}}(R))^{\mathcal{I}}$ for each $R \in \bar{\mathbf{R}}_{\mathcal{R}}$, and since $\mathcal{I} \models \Psi^{\mathcal{R}}(q)$ it follows $\mathcal{I} \models q$. This shows that $\mathcal{K} \models q$ as desired. $\square$

Again, the length of the longest regular expression $\rho^{\mathcal{R}}_{max}$ over all role names R in $\bar{\mathbf{R}}_{\mathcal{R}}$ affects the overall complexity of the algorithm.

Theorem 6.13. *Under unary number encoding, query entailment $\mathcal{K} \models q$ for a given $\mathcal{SRIQ}$ KB $\mathcal{K}$ and a P2RPQ q over $\mathcal{K}$ is decidable in double exponential time in the combined size of q, $\mathcal{C}_{\mathcal{K}}$, $\mathbf{I}_{\mathcal{K}}$, and $\rho^{\mathcal{R}}_{max}$, and 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 (union of) conjunctive queries, namely the positive (existential) two-way regular path queries (P2RPQs), is decidable over knowledge bases in the DL $\mathcal{ZIQ}$. 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 OWL 2 ontology standard by the W3C [3].

Our results also yield some novel complexity bounds, namely that the entailment problem of P2RPQs is 2ExpTime-complete for $\mathcal{ZIQ}$ and in 3ExpTime for $\mathcal{SRIQ}$. Given that conjunctive query entailment is 2ExpTime-hard already for the DLs $\mathcal{ALCI}$ [46] and $\mathcal{SH}$ [47], and that CRPQ/PQ entailment is equally hard already for $\mathcal{ALC}$, our results show that both on the query and on 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 for instance desired in applications of semi-structured data and graph databases [32,30,19,27,62].

The automata-based technique we apply is, in a sense, more accessible than other techniques that are based on tableaux [59,63] or resolution-based transformations to disjunctive datalog [64]. 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. This allows us to accommodate different DL constructors and obtain results for more expressive knowledge bases and queries than had been considered before, which seems to be more difficult using other approaches.

The viability of the automata approach has been confirmed by [33], where along the lines and ideas of this paper, the decidability frontier for entailment of P2RPQs has been orthogonally extended to $\mathcal{ZOQ}$ and $\mathcal{ZOI}$ 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. However, the results in [33] use richer, tailored automata models which directly support features such as counting and forest shaped structures. This makes the encoding simpler, but at the same time leaves out essential technical aspects in handling, for instance, number restrictions and ABoxes. The encoding in this paper is from first principles using classical automata on infinite trees; it is more illustrative on the technical issues that have to be resolved.

Unlike [35,44] we have considered qualified number restrictions. To incorporate them into the automata algorithm as simply as possible, we have assumed that numbers are encoded in unary and have used a natural encoding of counters into the states of the automaton. It is still somewhat cumbersome, as the simultaneous presence of inverses, concepts $\exists S.\mathsf{Self}$, and arbitrary ABoxes requires to navigate different parts of an interpretation to establish the satisfaction of a number restriction. We conjecture that one can obtain the same complexity bounds even if numbers are encoded in binary; however, this might require a significantly more involved encoding using binary counters.

Our results indicate that automata-techniques have high potential for advancing the decidability frontier of query answering over expressive DLs, and are a useful tool for analyzing the complexity of this problem. However, they seem to be of more limited use for assessing its *data complexity*, i.e., the complexity measured in terms of the ABox (data), assuming that the TBox and the query are fixed. Indeed, the set of states of the automaton $\mathbf{A}_q$ depends polynomially on the size of the ABox $\mathcal{A}$, hence the set of states of $\mathbf{A}_{\mathcal{K}\not\models q}$ can be double exponential in the size of $\mathcal{A}$ (more specifically, in the number of individuals occurring in $\mathcal{A}$). This means that the algorithm only gives a double exponential bound for data complexity. It is important to note, however, that this is due to the way the ABox is incorporated into interpretation trees, which uses states that check the relations for each pair of ABox individuals. The automata-theoretic operations used to build $\mathbf{A}_{\mathcal{K}\not\models q}$ and the emptiness test on it treat all automata states equally, which does not allow us to distinguish the complexity arising from a specific kind of states. Although this is indeed an intrinsic limitation of our approach, there is no reason to believe that the query entailment problem itself has such a high data complexity. Indeed, we would not be surprised if query entailment in $\mathcal{ZIQ}$ and $\mathcal{SRIQ}$ is coNP-complete like for other expressive DLs [13,16]. However, other approaches may be more promising in order to establish such a result. The automata approach may still be viable, but requires a different handling of ABoxes, decoupling them from the infinite model tree, similarly as in [65].

Another drawback of the approaches based on automata on infinite trees is that they have so far resisted implementation. Although for simpler problems, such as TBox satisfiability some initial proposals for automata-based implementations have been made [66]. 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 in [16] or knots as considered in [67].

Acknowledgments

This work has been partially supported by the FWF projects T515 *Recursive Queries over Semantically Enriched Data Repositories* and P20480 *Reasoning in Hybrid Knowledge Bases*, and the EU large-scale Integrating Project Optique *Scalable End-user Access to Big Data*, grant agreement No. FP7-318338.

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 2.15. Furthermore, we thank the reviewers for their helpful comments and suggestions to improve this paper.

Appendix A

In this appendix, we provide a proof of the canonical model property of $\mathcal{ZIQ}$ stated in Theorem 3.10, and of the correctness of the automata construction stated in Lemma 4.14.

Table 9
Concept atom $At \subseteq Cl_{\mathsf{C}}(\mathcal{T}, q)$.

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

Table 10
Role atom $AtR \subseteq Cl_{\mathsf{R}}(\mathcal{T}, q)$.

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

A.1. Proof of the canonical model property

In what follows, we assume a fixed KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ and a P2RPQ q. Recall that $k_{\mathcal{T},q} = br(C_{\mathcal{T}} \cup \mathcal{D}_q)$, where $\mathcal{D}_q = \{C \mid C(v) \in At(q)\} \cup \{\exists R.A \mid R(v, v') \in At(q)\}$ for an arbitrary concept name A, and $br(M) = |Cl(M)| \cdot n_{\max}$ where $n_{\max} = \max(\{n \mid\ \geqslant nS.C \in Cl(M)\} \cup \{0\})$. To simplify the notation, we use $Cl(\mathcal{T}, q)$ as a shorthand for $Cl(C_{\mathcal{T}} \cup \mathcal{D}_q)$. To show that $\mathcal{K}$ has a $k_{\mathcal{T},q}$-canonical model, we will follow the lines of similar proofs for the μ-calculus in [43,41] (which in turn, are adaptations of the original proof in [68]), 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 $k_{\mathcal{T},q}$-canonical, and that we can easily extract a $k_{\mathcal{T},q}$-canonical model of $\mathcal{K}$ from it. Moreover, since the unraveling preserves the satisfaction of all expressions in $Cl(\mathcal{T}, q)$ for every domain element, any match for q in the resulting model would already be present in the original one, hence the entailment of q is preserved.

We start by defining *concept* and *role atoms*, which are consistent sets of concepts and roles from $Cl(\mathcal{T}, q)$. In what follows, we denote by $Cl_{\mathsf{C}}(\mathcal{T}, q)$ and $Cl_{\mathsf{R}}(\mathcal{T}, q)$ the set of concepts and the set of roles in $Cl(\mathcal{T}, q)$, respectively. A *concept atom* is a set $At \subseteq Cl_{\mathsf{C}}(\mathcal{T}, q)$ of concepts closed under the rules of Table 9, while a *role atom* of $\mathcal{K}$ is a set $AtR \subseteq Cl_{\mathsf{R}}(\mathcal{T}, q)$ of simple roles closed under the rules of Table 10. The set of all concept and the set of all role atoms are respectively denoted by $\mathsf{atC}(\mathcal{T}, q)$ and $\mathsf{atR}(\mathcal{T}, q)$.

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 $\theta(d) \in \mathsf{atC}(\mathcal{T}, q)$ and each pair $(d, d') \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ to a role atom $\theta(d, d') \in \mathsf{atR}(\mathcal{T}, q)$, 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 $p \in Cl(\mathcal{T}, q)$, $p \in \theta(d, d')$ implies $(d, d') \in p^{\mathcal{I}}$, and $\neg p \in \theta(d, d')$ implies $(d, d') \notin p^{\mathcal{I}}$,
(4) for each $d, d' \in \Delta^{\mathcal{I}}$ and $p \in Cl(\mathcal{T}, q)$, $p \in \theta(d, d')$ 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 $A \in Cl(\mathcal{T}, q)$,
  (b) $\exists S.\mathsf{Self} \in \theta(d)$ implies $S \in \theta(d, d)$,
  (c) if $\geqslant nS.C \in \theta(d)$, then there is some $V \subseteq \mathsf{neigh}_{\mathcal{I},\theta}(S, d)$ such that $|V| \geqslant n$ and $C \in \theta(d')$ for every $d' \in V$, and
  (d) if $\leqslant nS.C \in \theta(d)$, then there is some $V \subseteq \mathsf{neigh}_{\mathcal{I},\theta}(S, d)$ such that $|V| \leqslant n$ 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 neighbor of d satisfies $\exists R^*.C$, and allow that 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 $(d, \geqslant nS.C)$ with $d \in \Delta^{\mathcal{I}}$ and $\geqslant nS.C \in \theta(d)$, $\mathsf{ch}(d, \geqslant nS.C)$ is a subset V of $\mathsf{neigh}_{\mathcal{I},\theta}(S, d)$ such that $|V| \geqslant n$ and $C \in \theta(d')$ for every $d' \in V$; and
- for each pair $(d, \leqslant nS.C)$ with $d \in \Delta^{\mathcal{I}}$ and $\leqslant nS.C \in \theta(d)$, $\mathsf{ch}(d, \leqslant nS.C)$ is a subset V of $\mathsf{neigh}_{\mathcal{I},\theta}(S, d)$ such that $|V| \leqslant n$ 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* $\vdash \subseteq (\Delta^{\mathcal{I}} \times Cl(\mathcal{T}, q)) \times (\Delta^{\mathcal{I}} \times Cl(\mathcal{T}, q))$ 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 $(d, C \sqcap C') \vdash (d, C')$,
- $\geqslant nS.C \in \theta(d)$ implies $(d, \geqslant nS.C) \vdash (d', C)$ for every $d' \in \mathsf{ch}(d, \geqslant nS.C)$,
- $\leqslant nS.C \in \theta(d)$ implies $(d, \leqslant nS.C) \vdash (d', \neg C)$ for every $d' \in \mathsf{neigh}_{\mathcal{I},\theta}(S, d) \setminus \mathsf{ch}(d, \leqslant nS.C)$,
- $\exists R^*.C \in \theta(d)$ implies $(d, \exists R^*.C) \vdash (d, C \sqcup \exists R.\exists R^*.C)$, and
- $\forall R^*.C \in \theta(d)$ implies $(d, \forall R^*.C) \vdash (d, C \sqcap \forall R.\forall R^*.C)$.

We say that 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 $1 \leqslant i < k$. We also say that $\langle \mathcal{I}, \theta, \mathsf{ch} \rangle$ is *well-founded*, if there is no $\exists R^*.C \in Cl(\mathcal{T}, q)$ and infinite sequence $d_1, d_2, \ldots$ such that $\exists R^*.C$ is regenerated from d_i to d_{i+1} for every $i \geqslant 1$. Then one can show:

Lemma A.1. *For every normalized $\mathcal{ZIQ}$ 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 an adaptation of similar proofs in [43,41], which extend the original proof for the μ-calculus in [68]. The absence of alternating fixpoints makes our setting simpler, and the presence of the ABox and the additional constructs are not hard to accommodate.

1: If $\mathcal{K} \not\models q$ for some q, then there exists 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: simply set $\theta(d) = \{C \in Cl_{\mathsf{C}}(\mathcal{T}, q) \mid d \in C^{\mathcal{I}}\}$ and $\theta(d, d') = \{S \in Cl_{\mathsf{C}}(\mathcal{T}, q) \mid (d, d') \in S^{\mathcal{I}}\}$ 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 speaking, 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 exists in $\mathcal{I}$ some finite sequence of elements reachable from d via R that leads to some $d' \in C^{\mathcal{I}}$; selecting such a path for the choice function avoids infinite regeneration of $\exists R^*.C$.

2: One can show by structural induction 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 $S \in \theta(d, d')$.

We remark that Boolean role expressions are handled analogously to concept expressions, while items (4) and (5b) of a pre-model 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 of $\langle \mathcal{I}, \theta, \mathsf{ch} \rangle$, 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) of a pre-model ensures the satisfaction of the TBox, and item (2) of the ABox. □

Now we are ready to prove Theorem 3.10, i.e., that for every normalized $\mathcal{ZIQ}$ KB $\mathcal{K}$ and P2RPQ q such that $\mathcal{K} \not\models q$, there exists a $k_{\mathcal{T},q}$-canonical model of $\mathcal{K}$ that admits no match for q.

Proof of Theorem 3.10. Assume $\mathcal{K} \not\models q$. By item 1 of Lemma A.1, there exists 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 $k_{\mathcal{T},q}$-canonical interpretation and $\mathcal{I}' \not\models q$. By item 2 of Lemma A.1, it follows that $\mathcal{I}'$ is a $k_{\mathcal{T},q}$-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 $1, \ldots, |\mathsf{R}(\mathcal{I})|$, and then continue building trees rooted at these roots, by inductively adding new levels.

For the base case, we let $\Delta^{\mathcal{I}'} = \{1, \ldots, |\mathsf{R}(\mathcal{I})|\}$ and let $\tau : \{1, \ldots, |\mathsf{R}(\mathcal{I})|\} \to \mathsf{R}(\mathcal{I})$ 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 concepts $\geqslant nS.C$ and $\leqslant nS.C$ are defined in the induction step.

For the induction step, consider an $x \in \Delta^{\mathcal{I}'}$ of maximal length, and let $(\geqslant n_1 S_1.C_1, e_1), \ldots, (\geqslant n_m S_m.C_m, e_m)$ 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 $1 \leqslant i \leqslant m$, we define:

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

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

We also extend the choice function to concepts of the form $C \sqcup C'$ in the new $\theta'(x\cdot i)$, and to concepts $\geqslant nS.C$ and $\leqslant nS.C$ 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 $\geqslant nS.C \in \theta'(x)$, $\mathsf{ch}'(x, \geqslant nS.C) = \{\phi(e) \mid e \in \mathsf{ch}(\tau(x), \geqslant nS.C)\}$; and
- for each $\leqslant nS.C \in \theta'(x)$, $\mathsf{ch}'(x, \leqslant nS.C) = \{\phi(e) \mid e \in \mathsf{ch}(\tau(x), \leqslant nS.C) \cap \{e_1, \ldots, e_n\}\}$.

Finally, 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 $p \in \mathbf{R}_{\mathcal{K}}$, $p^{\mathcal{I}'} = \{(x, y) \in \Delta^{\mathcal{I}'} \times \Delta^{\mathcal{I}'} \mid (\tau(x), \tau(y)) \in p^{\mathcal{I}}\}$.

We now verify that the conditions (1)–(5) of Definition 3.7 hold. Clearly, (1) $\{\epsilon\} \cup \Delta^{\mathcal{I}'}$ is a tree, and (2) $\mathit{Roots}(\mathcal{I}) = \{a^{\mathcal{I}} \mid a \in \mathbf{I}_{\mathcal{K}}\} \subseteq \mathbb{N}$. Next, m is bounded by $k_{\mathcal{T},q}$ 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 $x \in \{1, \ldots, k_{\mathcal{T},q}\}^*$, 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 role 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 $\neg p \in \theta'(x, y)$ for every $p \in \mathbf{R}$ and hence $(x, y) \notin p^{\mathcal{I}'}$. Therefore, $\mathcal{I}'$ is a $k_{\mathcal{T},q}$-canonical interpretation. 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, we show that $\mathcal{I}' \not\models q$. Assume towards a contradiction that $\mathcal{I}', \pi \models q$ for some π. Then the following claim implies that $\hat{\pi} = \pi \circ \tau$ is a match for q in $\mathcal{I}$, contradicting the assumption that $\mathcal{I} \not\models q$.

Claim. *Let $R(t, t')$ be an atom in q and let x, y be nodes in $\Delta^{\mathcal{I}'}$. If $(x, y) \in R^{\mathcal{I}'}$ then $(\tau(x), \tau(y)) \in R^{\mathcal{I}}$. Similarly, for an atom $C(t)$ of q and x in $\Delta^{\mathcal{I}'}$, $x \in C^{\mathcal{I}'}$ implies $\tau(x) \in C^{\mathcal{I}}$.*

It is only left to prove the claim. To this end we exploit the following two properties, which state that all concepts and roles in the query are preserved during the construction of $\mathcal{I}'$:

- Let $(x, y) \in \Delta^{\mathcal{I}'} \times \Delta^{\mathcal{I}'}$ and let S be a simple role in $\mathit{Cl}_{\mathsf{R}}(\mathcal{T}, q)$. Then $(x, y) \in S^{\mathcal{I}'}$ implies $(\tau(x), \tau(y)) \in S^{\mathcal{I}}$.
- Let $x \in \Delta^{\mathcal{I}'}$ and let C be a concept in $\mathit{Cl}_{\mathsf{C}}(\mathcal{T}, q)$. Then $x \in C^{\mathcal{I}'}$ implies $\tau(x) \in C^{\mathcal{I}}$.

The proof of both statements is easy, since the construction of $\mathcal{I}'$ ensures that each pair of objects x, y (resp., each object x) preserves all roles (resp., concepts) in the closure that are satisfied by $\tau(x), \tau(y)$ (resp., $\tau(x)$). Formally, they can be shown by a straightforward induction on the structure of S and C, respectively. To prove the claim, it then suffices to observe that by construction of $\mathcal{I}'$, if a sequence of objects in $\mathcal{I}'$ may participate in the satisfaction of some atom in q, the image of this sequence under τ would satisfy the same atom in $\mathcal{I}$. $\square$

A.2. Correctness of the main automata construction

Proof of Lemma 4.14. We first show similar properties for simple roles and for the states in $\mathsf{Q}_{\mathsf{Self}}$ and $\mathsf{Q}_{\mathcal{A}\_role}$. In the following claims, $\mathbf{T} = (T, L)$ is an interpretation tree for $\mathcal{K}$, as above.

(C1) Let $x \cdot i \in \Delta^{\mathcal{I}_\mathbf{T}}$ and let S be a simple role in $Cl(C_\mathcal{T})$. Then there is an accepting $(x \cdot i, S)$-run of $\mathbf{A}_\mathcal{T}$ over $\mathbf{T}$ iff $(x, x \cdot i) \in S^{\mathcal{I}_\mathbf{T}}$.
(C2) Let $x \in \Delta^{\mathcal{I}_\mathbf{T}}$ and let $S_{\mathsf{Self}} \in \mathsf{Q}_{\mathsf{Self}}$. Then there is an accepting (x, S)-run of $\mathbf{A}_\mathcal{T}$ over $\mathbf{T}$ iff $(x, x) \in S^{\mathcal{I}_\mathbf{T}}$.
(C3) Let $i, j \in \Delta^{\mathcal{I}_\mathbf{T}}$ and $S_{ij} \in \mathsf{Q}_{\mathcal{A}\_role}$. Then there is an accepting (ε, S_{ij})-run of $\mathbf{A}_\mathcal{T}$ over $\mathbf{T}$ iff $(i, j) \in S^{\mathcal{I}_\mathbf{T}}$.

Each of these properties is shown by a straightforward structural induction on the role S.

(C1) *Induction base*: Let $S = P \in \bar{\mathbf{R}}_\mathcal{K}$. (*If*) Assume $(x, x \cdot i) \in P^{\mathcal{I}_\mathbf{T}}$. By construction of $\mathcal{I}_\mathbf{T}$, $(x, x \cdot i) \in P^{\mathcal{I}_\mathbf{T}}$ implies that either $P = p \in \mathbf{R}$ and $(x, x \cdot i) \in \mathcal{R}^1_p$ or $P = p^-$ for some $p \in \mathbf{R}$ and $(x \cdot i, x) \in \mathcal{R}^1_p$. In the former case, $p = P \in L(x \cdot i)$, while in the latter $\mathsf{Inv}(p) = p^- = P \in L(x \cdot i)$. Since $P \in L(x \cdot i)$, $\delta_\mathcal{T}(P, L(x \cdot i)) = \mathsf{true}$ and the run with one single node ε labeled $r(\varepsilon) = (x \cdot i, P)$ is an accepting $(x \cdot i, S)$-run of $\mathbf{A}_\mathcal{T}$ over $\mathbf{T}$.
(*Only If*) Assume that there is an accepting $(x \cdot i, S)$-run of $\mathbf{A}_\mathcal{T}$ over $\mathbf{T}$. Then the root is labeled $r(\varepsilon) = (x \cdot i, P)$. By construction of $\mathbf{A}_\mathcal{T}$, $\delta_\mathcal{T}(P, L(x \cdot i)) = \mathsf{true}$ if $P \in L(x \cdot i)$ and false otherwise. Hence, since condition (R2) in Definition 2.8 holds, it follows that $P \in L(x \cdot i)$. If $P = p \in \mathbf{R}$, this implies $(x, x \cdot i) \in \mathcal{R}^1_p$, and if $P = p^-$ for some $p \in \mathbf{R}$, this implies $(x \cdot i, x) \in \mathcal{R}^1_p$. Since $x \cdot i \in \Delta^{\mathcal{I}_\mathbf{T}}$, in both cases this implies $(x, x \cdot i) \in P^{\mathcal{I}_\mathbf{T}}$.
(C1) *Inductive step*:
- First we consider the case of role difference. Recall that all roles in $Cl(C_\mathcal{T})$ are in NNF, hence we can assume that $S = S' \setminus P$ for an atomic role P.
(*If*) Assume $(x, x \cdot i) \in S^{\mathcal{I}_\mathbf{T}}$. Then $(x, x \cdot i) \in S'^{\mathcal{I}_\mathbf{T}}$, and $(x, x \cdot i) \notin P^{\mathcal{I}_\mathbf{T}}$. By definition of $\mathcal{I}_\mathbf{T}$, $(x, x \cdot i) \notin P^{\mathcal{I}_\mathbf{T}}$ implies $P \notin L(x \cdot i)$. By the inductive hypothesis, $(x, x \cdot i) \in S'^{\mathcal{I}_\mathbf{T}}$ implies that there is an accepting $(x \cdot i, S')$-run (T_r, r) of $\mathbf{A}_\mathcal{T}$ over $\mathbf{T}$. To extend it into an accepting $(x \cdot i, S)$-run, we first introduce the following notation: for $n \in \mathbb{N}$, let $n \cdot T_r = \{n \cdot w \mid w \in T_r\}$ (note that $n \cdot T_r$ is practically a tree, but its root is n rather than ε). Then we build a new run (T'_r, r') by taking $T'_r = \{\varepsilon, 1\} \cup (2 \cdot T_r)$, and setting $r'(\varepsilon) = (x \cdot i, S)$, $r'(1) = (x \cdot i, \neg P)$ and $r'(2 \cdot w) = r(w)$ for every $w \in T_r$. Condition (R2) in Definition 2.8 holds for all $2 \cdot w$ with $w \in T_r$ by construction. It also holds for $1 \in T'_r$ since $P \notin L(x \cdot i)$, as mentioned above, and hence $\delta_\mathcal{T}(P, L(x \cdot i)) = \mathsf{true}$ by item (III) in the definition of $\delta_\mathcal{T}$. Finally, condition (R2) also holds for ε as $\delta_\mathcal{T}(S, \sigma) = (0, S') \wedge (0, \neg P')$ (by item (II)). This shows that (T'_r, r') is an accepting $(x \cdot i, S)$-run.
(*Only If*) Assume that there is an accepting $(x \cdot i, S)$-run (T_r, r) of $\mathbf{A}_\mathcal{T}$ on $\mathbf{T}$. Then we have $r(\varepsilon) = (x \cdot i, S)$, and the only transition that can be used to satisfy condition (R2) is $\delta_\mathcal{T}(S, \sigma) = (0, S') \wedge (0, \neg P')$. This means that ε must have children j, ℓ such that $r(j) = (x \cdot i \cdot 0, S')$ and $r(\ell) = (x \cdot i \cdot 0, \neg P)$. It is not hard to see that if we restrict (T_r, r) to the subtrees rooted at j and ℓ, we obtain an accepting $(x \cdot i, S')$-run and an accepting $(x \cdot i, \neg P)$-run of $\mathbf{A}_\mathcal{T}$ on $\mathbf{T}$. By the induction hypothesis, this implies $(x, x \cdot i) \in S'^{\mathcal{I}_\mathbf{T}}$ and $(x, x \cdot i) \notin P^{\mathcal{I}_\mathbf{T}}$, hence $(x, x \cdot i) \in S^{\mathcal{I}_\mathbf{T}}$ as desired.
- The cases of $S = S_1 \cap S_2$ and $S = S_1 \cup S_2$ are analogous. We only sketch the former.
(*If*) Assume $(x, x \cdot i) \in S^{\mathcal{I}_\mathbf{T}}$. Then $(x, x \cdot i) \in S_1^{\mathcal{I}_\mathbf{T}}$, and $(x, x \cdot i) \in S_2^{\mathcal{I}_\mathbf{T}}$. Then by the inductive hypothesis, $(x, x \cdot i) \in S_1^{\mathcal{I}_\mathbf{T}}$ implies that there are an accepting $(x \cdot i, S_1)$-run (T^1_r, r_1) and an accepting $(x \cdot i, S_2)$-run (T^2_r, r_2) of $\mathbf{A}_\mathcal{T}$ over $\mathbf{T}$. We can then build an accepting $(x \cdot i, S)$-run (T'_r, r') by taking $T'_r = \{\varepsilon\} \cup (1 \cdot T^1_r) \cup (2 \cdot T^2_r)$, and setting $r'(\varepsilon) = (x \cdot i, S)$, $r'(j \cdot w) = r(w)$ for every $w \in T^j_r$, $j \in \{1, 2\}$.
(*Only If*) If there exists an accepting $(x \cdot i, S)$-run (T_r, r) of $\mathbf{A}_\mathcal{T}$ on $\mathbf{T}$, then $\delta_\mathcal{T}(S, \sigma) = (0, S_1) \wedge (0, S'_2)$ must be satisfied, which means that ε must have children j, ℓ such that $r(j) = (x \cdot i \cdot 0, S_1)$ and $r(\ell) = (x \cdot i \cdot 0, S_2)$. It is not hard to see that if we restrict (T_r, r) to the subtrees rooted at j and ℓ, we obtain an accepting $(x \cdot i, S_1)$-run and an accepting $(x \cdot i, S_2)$-run of $\mathbf{A}_\mathcal{T}$ on $\mathbf{T}$. By the induction hypothesis, this implies $(x, x \cdot i) \in S_1^{\mathcal{I}_\mathbf{T}}$ and $(x, x \cdot i) \notin P^{\mathcal{I}_\mathbf{T}}$, hence $(x, x \cdot i) \in S^{\mathcal{I}_\mathbf{T}}$ as desired.

(C2) *Induction base*: First we assume a role name $p \in \mathbf{R}$. If $\mathbf{I}_\mathcal{K} \cap L(x) \neq \emptyset$ then x is an individual node. Otherwise, if $x \in \Delta^{\mathcal{I}_\mathbf{T}}$ and $\mathbf{I}_\mathcal{K} \cap L(x) = \emptyset$, then it is not a level one node. Accordingly, we distinguish two cases:
- x is not a level one node. Then it suffices to observe that $p_{\mathsf{Self}} \in L(x)$ iff $(x, x) \in p^{\mathcal{I}_\mathbf{T}}$. Then it can be shown as above that the existence of an accepting (x, p_{Self})-run implies $p_{\mathsf{Self}} \in L(x)$ and conversely, if $p_{\mathsf{Self}} \in L(x)$, then there is an accepting (x, p_{Self})-run (comprising a single node ε with $r(\varepsilon) = (x, p_{\mathsf{Self}})$).
- $x \in \mathbb{N}$ is an individual node, i.e., there is some $a \in \mathbf{I}_\mathcal{K}$ such that $a \in L(x)$. Note that $a^{\mathcal{I}_\mathbf{T}} = x$, and $(x, x) \in p^{\mathcal{I}_\mathbf{T}}$ iff $p_{xx} \in L(\varepsilon)$. Hence, if $(x, x) \in p^{\mathcal{I}_\mathbf{T}}$, then we can build an accepting (x, p_{Self})-run by taking a root ε with $r(\varepsilon) = (x, p_{\mathsf{Self}})$, adding to it children 1 and 2 with $r(1) = (x, a)$ and $r(2) = (\varepsilon, \langle a, p_{\mathsf{Self}} \rangle)$, and then adding nodes $2 \cdot 1$ and $2 \cdot 2$, children of 2, which respectively have $r(2 \cdot 1) = (x, a)$ and $r(2 \cdot 2) = (\varepsilon, p_{xx})$. Since $a \in L(x)$ and $p_{xx} \in L(\varepsilon)$, the transition function is satisfied.
Conversely, if there is an accepting (x, p_{Self})-run, then its root must have children i and j with $r(i) = (x, b)$ and $r(j) = (\varepsilon, \langle b, p_{\mathsf{Self}} \rangle)$ for some $b \in \mathbf{I}_\mathcal{K}$. Since the run is accepting, this b must be such that $b \in L(x)$. The node j must have children with $r(j \cdot i') = (y, b)$ and $r(j \cdot j') = (\varepsilon, p_{yy})$. Again, since the run is accepting, we can conclude that $b \in L(y)$, which implies $y = x$ (since $\mathbf{T}$ is an interpretation tree, there is be exactly one node with b in its label). Since

the accepting run has a node labeled (ε, p_{xx}), it follows that $p_{xx} \in L(\varepsilon)$, which implies $(x,x) \in p^{\mathcal{I}_\mathbf{T}}$ by construction of $\mathcal{I}_\mathbf{T}$.

For S of the form p^- with $p \in \mathbf{R}$, we have that in an accepting $(x, p^-{}_{\mathsf{Self}})$-run the root has a child j such that $r(j) = (x, p_{\mathsf{Self}})$, which implies $(x,x) \in p^{\mathcal{I}_\mathbf{T}}$ and hence $(x,x) \in S^{\mathcal{I}_\mathbf{T}}$ and $x \in \exists S.\mathsf{Self}^{\mathcal{I}_\mathbf{T}}$. This shows the (*If*) direction. For the (*Only If*) direction, $x \in \exists S.\mathsf{Self}^{\mathcal{I}_\mathbf{T}}$ implies $(x,x) \in p^{\mathcal{I}_\mathbf{T}}$ and, as shown above, there is an accepting (x, p_{Self})-run, which can be easily extended to an $(x, p^-{}_{\mathsf{Self}})$-run by adding to it a root labeled $(x, p^-{}_{\mathsf{Self}})$ (note that $\delta_\mathcal{T}(p^-{}_{\mathsf{Self}}, L(x)) = (0, p_{\mathsf{Self}})$ by definition).

(C2) *Inductive step*: This step is practically identical the one of (C1). In the case of role difference one needs to make a case distinction between a role name p and its inverse p^-, analogously to the induction base.

(C3) *Induction base*: For a role name $p \in \mathbf{R}$, we observe that, since $i, j \in \Delta^{\mathcal{I}_\mathbf{T}}$ and by the construction of $\mathcal{I}_\mathbf{T}$, $p_{ij} \in L(\varepsilon)$ iff $(i,j) \in p^{\mathcal{I}_\mathbf{T}}$. Then the rest is as above: the existence of an accepting (ε, p_{ij})-run implies $p_{ij} \in L(\varepsilon)$ and conversely, if $p_{ij} \in L(\varepsilon)$, then there is an accepting (ε, p_{ij})-run (comprising a single node ε with $r(\varepsilon) = (\varepsilon, p_{ij})$). For S of the form p^- with $p \in \mathbf{R}$: (*If*) In an accepting (ε, p^-_{ij})-run the root has a child k such that $r(k) = (\varepsilon, p_{ji})$, which implies $p_{ji} \in L(\varepsilon)$ and hence $(j,i) \in p^{\mathcal{I}_\mathbf{T}}$ and $(i,j) \in p^{-\mathcal{I}_\mathbf{T}}$. (*Only If*) Since $(i,j) \in p^{-\mathcal{I}_\mathbf{T}}$ implies $(j,i) \in p^{\mathcal{I}_\mathbf{T}}$, we know we have $p_{ji} \in L(\varepsilon)$ and, as shown above, there is an accepting (ε, p_{ji})-run, which can be easily extended to an (ε, p^-_{ij})-run by adding to it a root labeled (ε, p^-_{ij}) (note that $\delta_\mathcal{T}(p^-_{ij}, L(\varepsilon)) = (0, p_{ji})$ by definition).

(C3) *Inductive step*: This is again analogous to (C1) and (C2).

Using these claims, one can show the lemma by structural induction on C.

Induction base: Let $C = A \in \mathbf{C}$. By the transitions in item III of Table 4 (which are the only ones given for states $A \in \mathbf{C}$) there is an accepting (x, A)-run iff $A \in L(x)$, and by construction of $\mathcal{I}_\mathbf{T}$ we have $A \in L(x)$ iff $x \in A^{\mathcal{I}_\mathbf{T}}$.

Inductive step: We must distinguish several cases due to the large number of constructors of $\mathcal{ZIQ}$.

- C is of the form $\neg A$, for a concept name A (recall that all concepts in $Cl(C_\mathcal{T})$ are in NNF). The proof is analogous to the base case: we have $A \notin L(x)$ iff $x \in \neg A^{\mathcal{I}_\mathbf{T}}$, and by the transitions in item III of Table 4 there is an accepting $(x, \neg A)$-run iff $A \notin L(x)$.
- C is of the form $D \sqcap D'$ or $D \sqcup D'$. The proof is analogous as for intersection and union of simple roles.
- C is of the form $\leqslant nS.D$ or of the form $\geqslant nS.D$.
  We rely on the following auxiliary claims. Here, we use the notion of *potential neighbors* and (S,D)*-neighbors* defined above. Note that, by definition, the potential neighbors include all successors of a node, also the 'dummy' nodes in $\mathbf{T}$ that do not correspond to an object in $\Delta^{\mathcal{I}_\mathbf{T}}$. For an individual node, its potential neighbors also include all the level 1 nodes, both actual individual nodes and 'dummy' nodes. Note that, in contrast, the (S,D)*-neighbors* are necessarily elements of $\Delta^{\mathcal{I}_\mathbf{T}}$. Moreover, we order the potential neighbors of a node x as follows: the first $b_\mathcal{K}$ ones are its successors, in the order they occur in $\mathbf{T}$. If there is no $a \in \mathbf{I}_\mathcal{K}$ such that $a^{\mathcal{I}_\mathbf{T}} = x$, then the $(b_\mathcal{K}+1)$-th is the node itself, and the $(b_\mathcal{K}+2)$-th is its predecessor. Otherwise, the $(b_\mathcal{K}+1)$-th to $(2b_\mathcal{K})$-th are the level 1 nodes, in the order they appear in $\mathbf{T}$.
  (C4) Let $x \in \Delta^{\mathcal{I}_\mathbf{T}}$ and let $\langle \geqslant nS.D, i, j\rangle \in Q_{num}$ (resp., $\langle \leqslant nS.D, i, j\rangle \in Q_{num}$). Then there is an accepting $(x, \langle \geqslant nS.D, i, j\rangle)$-run (resp., $(x, \langle \leqslant nS.D, i, j\rangle)$-run) of $\mathbf{A}_\mathcal{T}$ over $\mathbf{T}$ iff there are at least (resp., at most) $n-j$ many (S,D)-neighbors of x among its potential neighbors beyond the i-th one.
  (C5) Let $\langle a, \geqslant nS.D, i, j\rangle \in Q_{\mathcal{A}\_num}$ (resp., $\langle a, \leqslant nS.D, i, j\rangle \in Q_{\mathcal{A}\_num}$). Then there is an accepting $(\varepsilon, \langle a, \gtrless nS.D, i, j\rangle)$-run of $\mathbf{A}_\mathcal{T}$ over $\mathbf{T}$ iff there are at least (resp., at most) $n-j$ many (S,D)-neighbors of $a^{\mathcal{I}_\mathbf{T}}$ among its potential neighbors beyond the $(b_\mathcal{K}+i)$-th.
  Both claims are easy to show, since the transitions in item IV2 of Table 5 ensure that all potential neighbors are navigated in the right order. The i counter is always increased, while the j counter is increased iff the i-th potential neighbor is actually an (S,D)-neighbor. The correctness of the latter check follows easily from the way both disjuncts are defined, relying on the Claims (C1)–(C3) for S and on the inductive hypothesis for D.
  For each number restriction $\gtrless nS.C$, the automaton will first switch to $\langle \gtrless nS.C, 0, 0\rangle$. From there, by Claims (C4) and (C5), it will correctly count its potential neighbors. An inspection of the transitions in item IV3 of Table 5 reveals that the run can be successfully completed iff the number restriction holds at the corresponding node.
- C is of the form $\exists S.\mathsf{Self}$. Recall that $x \in \exists S.\mathsf{Self}^{\mathcal{I}_\mathbf{T}}$ iff $(x,x) \in S^{\mathcal{I}_\mathbf{T}}$, and that $\delta_\mathcal{T}(\exists S.\mathsf{Self}, \sigma) = (0, S_{\mathsf{Self}})$ is the only transition specified for states of the form $\exists S.\mathsf{Self}$. We can apply Claim (C2). For the (*If*) direction, we have that $x \in \exists S.\mathsf{Self}^{\mathcal{I}_\mathbf{T}}$ implies the existence of an accepting (x, S_{Self})-run, which can be extended into an $(x, \exists S.\mathsf{Self})$-run by simply adding a root labeled $(x, \exists S.\mathsf{Self})$. For the converse, in an accepting $(x, \exists S.\mathsf{Self})$-run the root necessarily has a child labeled (x, S_{Self}), which can be viewed as the root of an accepting (x, S_{Self})-run.
- C is of the form $\exists S.D$ or $\forall S.D$ for a simple role S. Then the claim follows from the fact that the transition function treats these concepts as equivalent number restrictions.
  (*If*) $x \in (\exists S.D)^{\mathcal{I}_\mathbf{T}}$, then $x \in (\geqslant 1S.D)^{\mathcal{I}_\mathbf{T}}$, so we can take an accepting $(x, \geqslant 1S.D)$-run, which we have shown to exist and extend it into an accepting $(x, \exists S.D)$-run by adding a root with label $(x, \exists S.D)$. Similarly, adding an $(x, \forall S.D)$-labeled root to an accepting $(x, \leqslant 0S.\sim D)$-run yields an accepting $(x, \forall S.D)$-run. (*Only If*) Conversely, if we remove the root of an accepting $(x, \exists S.D)$-run we obtain a subtree that is an accepting $(x, \geqslant 1S.D)$-run. As we have shown, this implies that

$x \in (\geqslant 1 S.D)^{\mathcal{I}_\mathbf{T}}$ and thus $x \in (\exists S.D)^{\mathcal{I}_\mathbf{T}}$. Similarly, an accepting $(x, \forall S.D)$-run contains an $(x, \leqslant 0 S. \sim D)$-run as subtree which implies $x \in (\leqslant 0 S. \sim D)^{\mathcal{I}_\mathbf{T}}$ and $x \in (\forall S.D)^{\mathcal{I}_\mathbf{T}}$.

- C is of the form $\forall(R \cup R').D$, $\forall(R \circ R').D$, $\forall id(D).D'$, $\exists(R \cup R').D$, $\exists(R \circ R').D$, or $\exists id(D).D'$. From these states $\delta_\mathcal{T}$ moves to states corresponding to syntactically simpler but semantically equivalent expressions, for which the induction hypothesis applies. The proof for all of all these cases is analogous to the previous case: we can build an accepting (x, D)-run by adding a root to runs for the equivalent expressions, and every (x, D)-run contains sub-runs for the corresponding expressions.
- C is of the form $\forall R^*.D$. We know that $x \in (\forall R^*.D)^{\mathcal{I}_\mathbf{T}}$ iff it holds that $x \in (D)^{\mathcal{I}_\mathbf{T}}$ and $x \in (\forall R.\forall R^*.D)^{\mathcal{I}_\mathbf{T}}$. Showing that there is a (possibly infinite) accepting $(x, \forall R^*.D)$-run iff $x \in (D)^{\mathcal{I}_\mathbf{T}}$ and $x \in (\forall R.\forall R^*.D)^{\mathcal{I}_\mathbf{T}}$ is not hard. For the only-if direction, we start from a root ε with $r(\varepsilon) = (x, \forall R^*.D)$, and give it a child 1 with $r(1) = (x, D)$ and a child 2 with $r(2) = (x, \forall R.\forall R^*.D)$. We use the induction hypothesis to argue that (i) there exists an (x, D)-run, and (ii) if there exists an accepting $(x, \forall R^*.D)$-run, then there exists an accepting $(x, \forall R.\forall R^*.D)$-run. We add these as subruns, and when we reach the $(x, \forall R.\forall R^*.D)$ run, we simply repeat the above construction. Also the converse can be shown in a straightforward way, by using the structure of an accepting $(x, \forall R^*.D)$-run to show that $x \in (D)^{\mathcal{I}_\mathbf{T}}$ and $x \in (\forall R.\forall R^*.D)^{\mathcal{I}_\mathbf{T}}$. Note that the accepting $(x, \forall R^*.D)$-run may contain arbitrarily many $(x, \forall R^*.D)$-runs as subruns.
- C is of the form $\exists R^*.D$. One can proceed analogously to the previous case to show that the existence of a possibly infinite (not necessarily accepting) $(x, \exists R^*.D)$-run is equivalent to having $x \in (D)^{\mathcal{I}_\mathbf{T}}$ or $x \in (\exists R.\exists R^*.D)^{\mathcal{I}_\mathbf{T}}$. It is only left to observe that the acceptance condition is satisfied iff there is no path in the $(x, \exists R^*.D)$-run where $r(w) = (x, \exists R^*.D)$ for infinitely many nodes w. By inspecting the transition function we see that this is the case iff there is some w' with $r(w') = (y, D)$ (which implies $y \in (D)^{\mathcal{I}_\mathbf{T}}$), and there is a path from x to y in $\mathcal{I}_\mathbf{T}$ (which may contain only x itself) such that the run visited all elements z on this path and verified $(x, z) \in (R^*)^{\mathcal{I}_\mathbf{T}}$. This shows that there is an accepting $(x, \exists R^*.D)$-run iff there is some y such that $(x, y) \in (R^*)^{\mathcal{I}_\mathbf{T}}$ and $y \in (D)^{\mathcal{I}_\mathbf{T}}$, as desired. □

References

[1] D. Calvanese, T. Eiter, 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), 2007, pp. 391–396.

[2] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, P.F. Patel-Schneider (Eds.), The Description Logic Handbook: Theory, Implementation and Applications, Cambridge University Press, 2003.

[3] B. Cuenca Grau, I. Horrocks, B. Motik, B. Parsia, P.P.-S.U. Sattler, OWL 2: The next step for OWL, J. Web Semant. 6 (4) (2008) 309–322.

[4] I. Horrocks, O. Kutz, U. Sattler, The even more irresistible $\mathcal{SROIQ}$, in: Proc. of the 10th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2006), AAAI Press, 2006, pp. 57–67.

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

[6] S. Bechhofer, F. van Harmelen, J. Hendler, I. Horrocks, D.L. McGuinness, P.F. Patel-Schneider, L.A. Stein, OWL Web Ontology Language reference – W3C recommendation, Tech. rep., World Wide Web Consortium, available at, http://www.w3.org/TR/owl-ref/, February 2004.

[7] D. Calvanese, G. De Giacomo, 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), 1998, pp. 149–158.

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

[9] M. Krötzsch, S. Rudolph, P. Hitzler, Conjunctive queries for a tractable fragment of OWL 1.1, in: Proc. of the 6th Int. Semantic Web Conf. (ISWC 2007), in: Lect. Notes Comput. Sci., vol. 4825, Springer, 2007, pp. 310–323.

[10] R. Rosati, On conjunctive query answering in $\mathcal{EL}$, in: Proc. of the 20th Int. Workshop on Description Logic (DL 2007), in: CEUR Electronic Workshop Proc., vol. 250, 2007, pp. 451–458, http://ceur-ws.org/.

[11] A. Krisnadhi, C. Lutz, Data complexity in the $\mathcal{EL}$ family of description logics, in: Proc. of the 20th Int. Workshop on Description Logic (DL 2007), in: CEUR Electronic Workshop Proc., vol. 250, 2007, pp. 88–99, http://ceur-ws.org/.

[12] D. Calvanese, G.D. Giacomo, D. Lembo, M. Lenzerini, R. Rosati, Tractable reasoning and efficient query answering in description logics: The DL-Lite family, J. Autom. Reason. 39 (3) (2007) 385–429.

[13] B. Glimm, I. Horrocks, C. Lutz, U. Sattler, Conjunctive query answering for the description logic $\mathcal{SHIQ}$, J. Artif. Intell. Res. 31 (2008) 151–198.

[14] B. Glimm, S. Rudolph, Nominals, inverses, counting, and conjunctive queries or: Why infinity is your friend, J. Artif. Intell. Res. 39 (2010) 429–481.

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

[16] M. Ortiz, D. Calvanese, T. Eiter, Data complexity of query answering in expressive description logics via tableaux, J. Autom. Reason. 41 (1) (2008) 61–98.

[17] P. Buneman, Semistructured data, in: Proc. of the 16th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS'97), 1997, pp. 117–121.

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

[19] D. Calvanese, G. De Giacomo, M. Lenzerini, M.Y. Vardi, Rewriting of regular expressions and regular path queries, J. Comput. Syst. Sci. 64 (3) (2002) 443–465.

[20] M. San Martin, C. Gutierrez, Representing, querying and transforming social networks with RDF/SPARQL, in: Proc. of the 6th European Semantic Web Conf. (ESWC 2009), in: Lect. Notes Comput. Sci., vol. 5554, Springer, 2009, pp. 293–307.

[21] S. Hagedorn, K.-U. Sattler, Discovery querying in Linked Open Data, in: Workshop Proc. of the Joint 2013 EDBT/ICDT Conferences, ACM Press, 2013, pp. 38–44.

[22] O. Shmueli, Equivalence of Datalog queries is undecidable, J. Log. Program. 15 (3) (1993) 231–241.

[23] D. Calvanese, R. Rosati, Answering recursive queries under keys and foreign keys is undecidable, in: Proc. of the 10th Int. Workshop on Knowledge Representation meets Databases (KRDB 2003), in: CEUR Electronic Workshop Proc., vol. 79, 2003, http://ceur-ws.org/.

[24] A.Y. Levy, M.-C. Rousset, Combining Horn rules and description logics in CARIN, Artif. Intell. 104 (1–2) (1998) 165–209.

[25] S. Harris, A. Seaborne, SPARQL 1.1 Query Language, W3C Recommendation, World Wide Web Consortium, available at http://www.w3.org/TR/sparql11-query, March 2013.

[26] A. Berglund, et al., XML Path Language (XPath) 2.0 (Second Edition), W3C Recommendation, World Wide Web Consortium, available at http://www.w3.org/TR/xpath20, December 2010.

[27] D. Calvanese, G. De Giacomo, M. Lenzerini, M.Y. Vardi, Reasoning on regular path queries, SIGMOD Rec. 32 (4) (2003) 83–92.

[28] G. Grahne, 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), 2003, pp. 111–122.
[29] A. Deutsch, V. Tannen, Optimization properties for classes of conjunctive regular path queries, in: G. Ghelli, G. Grahne (Eds.), Proc. of the 8th Int. Workshop on Database Programming Languages (DBPL 2001), in: Lect. Notes Comput. Sci., vol. 2397, Springer, 2001, pp. 21–39.
[30] D. Calvanese, G. De Giacomo, M. Lenzerini, 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), 2000, pp. 176–185.
[31] S. Abiteboul, V. Vianu, Regular path queries with constraints, J. Comput. Syst. Sci. 58 (3) (1999) 428–452.
[32] D. Florescu, A. Levy, 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), 1998, pp. 139–148.
[33] D. Calvanese, T. Eiter, M. Ortiz, Regular path queries in expressive description logics with nominals, in: C. Boutilier (Ed.), Proc. of the 21st Int. Joint Conf. on Artificial Intelligence (IJCAI 2009), 2009, pp. 714–720.
[34] S. Tobies, PSPACE reasoning for graded modal logics, J. Log. Comput. 11 (1) (2001) 85–106.
[35] D. Calvanese, G. De Giacomo, M. Lenzerini, 2ATAs make DLs easy, in: Proc. of the 15th Int. Workshop on Description Logic (DL 2002), CEUR Electronic Workshop Proceedings, 2002, pp. 107–118, http://ceur-ws.org/Vol-53/.
[36] I. Horrocks, S. Tessaris, A conjunctive query language for description logic ABoxes, in: Proc. of the 17th Nat. Conf. on Artificial Intelligence (AAAI 2000), 2000, pp. 399–404.
[37] U. Hustadt, B. Motik, 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), 2004, pp. 21–35.
[38] W. Thomas, Automata on infinite objects, in: J. van Leeuwen (Ed.), Handbook of Theoretical Computer Science, Vol. B, Elsevier Science Publishers, 1990, pp. 133–192, Ch. 4.
[39] M.Y. Vardi, The taming of converse: Reasoning about two-way computations, in: R. Parikh (Ed.), Proc. of the 4th Workshop on Logics of Programs, in: Lect. Notes Comput. Sci., vol. 193, Springer, 1985, pp. 413–424.
[40] M.Y. Vardi, P. Wolper, Automata-theoretic techniques for modal logics of programs, J. Comput. Syst. Sci. 32 (1986) 183–221.
[41] 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), in: Lect. Notes Comput. Sci., vol. 1443, Springer, 1998, pp. 628–641.
[42] O. Kupferman, U. Sattler, M.Y. Vardi, The complexity of the graded μ-calculus, in: Proc. of the 18th Int. Conf. on Automated Deduction (CADE 2002), in: Lect. Notes Comput. Sci., vol. 2392, Springer, 2002, pp. 423–437.
[43] P. Bonatti, C. Lutz, A. Murano, M.Y. Vardi, The complexity of enriched μ-calculi, Log. Methods Comput. Sci. 4 (3:11) (2008) 1–27.
[44] D. Calvanese, G. De Giacomo, 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), 1999, pp. 84–89.
[45] S. Tobies, Complexity results and practical algorithms for logics in knowledge representation, Ph.D. thesis, LuFG Theoretical Computer Science RWTH-Aachen, Germany, 2001.
[46] C. Lutz, Inverse roles make conjunctive queries hard, in: Proc. of the 20th Int. Workshop on Description Logic (DL 2007), in: CEUR Electronic Workshop Proc., vol. 250, 2007, pp. 100–111, http://ceur-ws.org/.
[47] T. Eiter, C. Lutz, M. Ortiz, M. Šimkus, Query answering in description logics with transitive roles, in: C. Boutilier (Ed.), Proc. of the 21st Int. Joint Conf. on Artificial Intelligence (IJCAI 2009), 2009, pp. 759–764.
[48] A.K. Chandra, 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), 1977, pp. 77–90.
[49] D. Calvanese, G. De Giacomo, M. Lenzerini, Conjunctive query containment and answering under description logics constraints, ACM Trans. Comput. Log. 9 (3) (2008) 22.1–22.31.
[50] D.E. Muller, P.E. Schupp, Alternating automata on infinite trees, Theor. Comput. Sci. 54 (1987) 267–276.
[51] E.A. Emerson, C.S. Jutla, Tree automata, mu-calculus and determinacy, in: Proc. of the 32nd Annual Symp. on the Foundations of Computer Science (FOCS'91), 1991, pp. 368–377.
[52] D.E. Muller, P.E. Schupp, Simulating alternating tree automata by nondeterministic automata: New results and new proofs of the theorems of Rabin, McNaughton and Safra, Theor. Comput. Sci. 141 (1–2) (1995) 69–107.
[53] O. Kupferman, M.Y. Vardi, Weak alternating automata and tree automata emptiness, in: Proc. of the 30th ACM SIGACT Symp. on Theory of Computing (STOC'98), ACM Press, 1998, pp. 224–233.
[54] S. Rudolph, M. Krötzsch, P. Hitzler, Cheap Boolean role constructors for description logics, in: Proc. of the 11th Eur. Conference on Logics in Artificial Intelligence (JELIA 2008), in: Lect. Notes Comput. Sci., vol. 5293, 2008, pp. 362–374.
[55] M.J. Fischer, R.E. Ladner, Propositional dynamic logic of regular programs, J. Comput. Syst. Sci. 18 (1979) 194–211.
[56] M. Ortiz, An automata-based algorithm for description logics around $\mathcal{SRIQ}$, in: Proc. of the 4th Latin American Workshop on Non-Monotonic Reasoning (LANMR 2008), in: CEUR Electronic Workshop Proc., vol. 408, 2008, pp. 1–15, http://ceur-ws.org/.
[57] K. Schild, A correspondence theory for terminological logics: Preliminary report, in: Proc. of the 12th Int. Joint Conf. on Artificial Intelligence (IJCAI'91), 1991, pp. 466–471.
[58] M. Bienvenu, D. Calvanese, M. Ortiz, M. Šimkus, Nested regular path queries in description logics, in: Proc. of the 14th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2014), AAAI Press, 2014.
[59] I. Horrocks, U. Sattler, Decidability of $\mathcal{SHIQ}$ with complex role inclusion axioms, Artif. Intell. 160 (1) (2004) 79–104.
[60] I. Horrocks, U. Sattler, S. Tobies, Reasoning with individuals for the description logic $\mathcal{SHIQ}$, in: D. McAllester (Ed.), Proc. of the 17th Int. Conf. on Automated Deduction (CADE 2000), in: Lect. Notes Comput. Sci., vol. 1831, Springer, 2000, pp. 482–496.
[61] 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), 2008, pp. 274–284.
[62] P. Barcelò, L. Libkin, A.W. Lin, P.T. Wood, Expressive languages for path queries over graph-structured data, ACM Trans. Database Syst. 37 (4) (2012) 31.
[63] I. Horrocks, U. Sattler, A tableau decision procedure for $\mathcal{SHOIQ}$, J. Autom. Reason. 39 (3) (2007) 249–276.
[64] U. Hustadt, B. Motik, U. Sattler, Reasoning in description logics by a reduction to Disjunctive Datalog, J. Autom. Reason. 39 (3) (2007) 351–384.
[65] M. Ortiz, S. Rudolph, M. Simkus, Query answering in the Horn fragments of the description logics $\mathcal{SHOIQ}$ and $\mathcal{SROIQ}$, in: Proc. of the 22nd Int. Joint Conf. on Artificial Intelligence (IJCAI 2011), 2011, pp. 1039–1044.
[66] D. Calvanese, D. Carbotta, M. Ortiz, A practical automata-based technique for reasoning in expressive description logics, in: Proc. of the 22nd Int. Joint Conf. on Artificial Intelligence (IJCAI 2011), 2011, pp. 798–804.
[67] T. Eiter, C. Lutz, M. Ortiz, M. Simkus, Query answering in description logics: The knots approach, in: H. Ono, M. Kanazawa, R.J.G.B. de Queiroz (Eds.), Proc. of the 16th Int. Workshop on Logic, Language, Information and Computation (WoLLIC 2009), in: Lect. Notes Comput. Sci., vol. 5514, Springer, 2009, pp. 26–36.
[68] R.S. Streett, E.A. Emerson, An automata theoretic decision procedure for the propositional μ-calculus, Inf. Comput. 81 (1989) 249–264.