The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Answering Regular Path Queries over $\mathcal{SQ}$ Ontologies

Víctor Gutiérrez-Basulto
Cardiff University, UK
gutierrezbasultov@cardiff.ac.uk

Yazmín Ibáñez-García
TU Wien, Austria
yazmin.garcia@tuwien.ac.at

Jean Christoph Jung
Universität Bremen, Germany
jeanjung@uni-bremen.de

Abstract

We study query answering in the description logic $\mathcal{SQ}$ supporting qualified number restrictions on both transitive and non-transitive roles. Our main contributions are a tree-like model property for $\mathcal{SQ}$ knowledge bases and, building upon this, an optimal automata-based algorithm for answering positive existential regular path queries in 2ExpTime.

1 Introduction

The use of ontologies to access data has gained a lot of popularity in various research fields such as knowledge representation and reasoning, and databases. In the ontology-based data access (OBDA) scenario, ontologies are often encoded using description logic languages (DLs); as a consequence, a large amount of research on the query answering problem (QA) over DL ontologies has been conducted. In particular, several efforts have been put into the study of the query answering problem in DLs featuring *transitive roles* and *number restrictions* (Glimm, Horrocks, and Sattler 2008; Glimm et al. 2008; Eiter et al. 2009; Calvanese, Eiter, and Ortiz 2009; 2014). However, in all these works the application of number restrictions to transitive roles is forbidden. This is also reflected in the fact that the W3C ontology language OWL 2 does not allow for this interaction.[1] Unfortunately, this comes as a shortcoming in crucial DL application areas like medicine and biology in which many terms are defined and classified according to the number of components they contain or have as a part, in a transitive sense (Wolstencroft et al. 2005; Rector and Rogers 2006; Stevens et al. 2007). For instance, the ontology $\mathcal{T}$ below describes that the human heart has as a part (hPt) exactly one mitral valve (MV), a left atrium (LA) and a left ventricle (LV); and the latter two (enforced to be distinct) also have as a part a mitral valve. Thus, the left atrium and left ventricle have to share the mitral valve.

$$\mathcal{T} = \{\ \mathsf{Heart} \sqsubseteq (= 1\ \mathsf{hPt}.\mathsf{MV}) \sqcap \exists \mathsf{hPt}.\mathsf{LA} \sqcap \exists \mathsf{hPt}.\mathsf{LV},$$
$$\mathsf{LV} \sqcap \mathsf{LA} \sqsubseteq \bot, \quad \mathsf{LV} \sqsubseteq \exists \mathsf{hPt}.\mathsf{MV}, \quad \mathsf{LA} \sqsubseteq \exists \mathsf{hPt}.\mathsf{MV}\ \}.$$

The lack of investigations of query answering in DLs of this kind is partly because (i) the interaction of these features with other traditional constructors often leads to undecidability of the standard reasoning tasks (e.g., satisfiability) (Horrocks, Sattler, and Tobies 2000); and (ii) for those DLs known to be decidable, such as $\mathcal{SQ}$ and $\mathcal{SOQ}$ (Kazakov, Sattler, and Zolin 2007; Kaminski and Smolka 2010), only recently tight complexity bounds were obtained (Gutiérrez-Basulto, Ibáñez-García, and Jung 2017a). Moreover, these features, even with restricted interaction, pose additional challenges for devising decision procedures since they lead to the loss of properties, such as the tree model property, which make the design of algorithms for QA simpler. Clearly, these issues are exacerbated if number restrictions are imposed on transitive roles.

Traditionally, most of the research in OBDA has focused on answering conjunctive queries. However, *navigational* queries have recently gained a lot of attention (Stefanoni et al. 2014; Bienvenu, Ortiz, and Simkus 2015; Baget et al. 2017) since they are key in various applications. For instance, in biomedicine they are used to retrieve specific paths from protein, cellular and disease networks (Dogrusoz et al. 2009; Lysenko et al. 2016). A prominent class of navigational queries is that of *regular path queries* (Florescu, Levy, and Suciu 1998), where paths are specified by a regular expression. Indeed, motivated by applications in the semantic web, the latest W3C standard SPARQL 1.1 includes property paths, related to regular expressions.

The objective of this paper is to start the research on query answering in DLs supporting qualified number restrictions over transitive roles. We study the entailment problem of *positive existential two-way regular path queries* (Calvanese et al. 2000) over $\mathcal{SQ}$ ontologies, thus generalizing both conjunctive and regular path queries. To this end, we pursue an *automata-based approach for query answering* using two-way alternating tree automata (2ATA) (Vardi 1998). This roughly consists of three steps (Calvanese, Eiter, and Ortiz 2014): (i) show that, if a query φ is not entailed by the knowledge base $\mathcal{K}$, there is a *tree-like* interpretation witnessing this, (ii) devise an automaton $\mathfrak{A}_\mathcal{K}$ which accepts precisely the tree-like interpretations of $\mathcal{K}$, (iii) devise an automaton $\mathfrak{A}_\varphi$ which accepts a tree-like interpretation iff it satisfies φ. Query entailment is then reduced to the question whether $\mathfrak{A}_\mathcal{K}$ accepts a tree that is not accepted by $\mathfrak{A}_\varphi$. In this paper, we significantly adapt and extend each step to $\mathcal{SQ}$, resulting in an algorithm running in 2ExpTime, even

[1] https://www.w3.org/TR/webont-req/

for *binary* coding of numbers. A matching lower bound follows from positive existential QA in $\mathcal{ALC}$ (Calvanese, Eiter, and Ortiz 2014). More precisely, for step (i) we develop the notion of *canonical tree decompositions* which intuitively are tree decompositions tailored to handle the interaction of transitivity and number restrictions. We then show via a novel unraveling operation for $\mathcal{SQ}$ that, if the query is not entailed, there is a witness interpretation which has a canonical tree decomposition of width bounded exponentially in the size of $\mathcal{K}$, cf. Section 3. These canonical tree decompositions are crucial in order to construct a small 2ATA $\mathfrak{A}_{\mathcal{K}}$ in step (ii), which is done in Section 4.1. For step (iii), we propose in Section 4.2 a novel technique for answering regular path queries directly using a 2ATA $\mathfrak{A}_\varphi$ since a naive application of the techniques from (Calvanese, Eiter, and Ortiz 2014) does not lead to optimal complexity, because of the large width of the decompositions.

An extended version with appendix can be found under www.informatik.uni-bremen.de/tdki/research/papers.html.

2 Preliminaries

Syntax. We consider a vocabulary consisting of countably infinite disjoint sets of *concept names* $\mathsf{N_C}$, *role names* $\mathsf{N_R}$, and *individual names* $\mathsf{N_I}$, and assume that $\mathsf{N_R}$ is partitioned into two countably infinite sets of *non-transitive role names* $\mathsf{N_R^{nt}}$ and *transitive role names* $\mathsf{N_R^{t}}$. The syntax of $\mathcal{SQ}$-*concepts* C, D is given by the rule

$$C, D ::= A \mid \neg C \mid C \sqcap D \mid (\leqslant n\, r.C)$$

where $A \in \mathsf{N_C}$, $r \in \mathsf{N_R}$, and n is a number given in binary. We use $(\geqslant n\, r.C)$ as an abbreviation for $\neg(\leqslant n{-}1\, r.C)$, and other standard abbreviations like $\bot$, $\top$, $C \sqcup D$, $\exists r.C$, $\forall r.C$. Concepts of the form $(\leqslant n\ r.C)$ and $(\geqslant n\ r.C)$ are called *at-most restrictions* and *at-least restrictions*, respectively.

An $\mathcal{SQ}$*-TBox (ontology)* $\mathcal{T}$ is a finite set of *concept inclusions* $C \sqsubseteq D$ where C, D are $\mathcal{SQ}$-concepts. An *ABox* is a finite set of *concept* and *role assertions* of the form $A(a)$, $r(a, b)$ where $A \in \mathsf{N_C}$, $r \in \mathsf{N_R}$ and $\{a, b\} \subseteq \mathsf{N_I}$; $\mathsf{ind}(\mathcal{A})$ denotes the set of individual names occurring in $\mathcal{A}$. A *knowledge base (KB)* is a pair $\mathcal{K} = (\mathcal{T}, \mathcal{A})$.

Semantics. An *interpretation* $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ consists of a non-empty *domain* $\Delta^\mathcal{I}$ and an *interpretation function* $\cdot^\mathcal{I}$ mapping concept names to subsets of the domain and role names to binary relations over the domain such that transitive role names are mapped to transitive relations. The interpretation function is extended to complex concepts by defining $(\neg C)^\mathcal{I} = \Delta^\mathcal{I} \setminus C^\mathcal{I}$, $(C \sqcap D)^\mathcal{I} = C^\mathcal{I} \cap D^\mathcal{I}$, and

$$(\leqslant n\, r.C)^\mathcal{I} = \{d \in \Delta^\mathcal{I} \mid |\{e \in C^\mathcal{I} \mid (d, e) \in r^\mathcal{I}\}| \leq n\}.$$

For ABoxes $\mathcal{A}$ we adopt the *standard name assumption (SNA)*, that is, $a^\mathcal{I} = a$, for all $a \in \mathsf{ind}(\mathcal{A})$, but we strongly conjecture that our results hold without it. The satisfaction relation $\models$ is defined as usual by taking $\mathcal{I} \models C \sqsubseteq D$ iff $C^\mathcal{I} \subseteq D^\mathcal{I}$, $\mathcal{I} \models A(a)$ iff $a \in A^\mathcal{I}$, and $\mathcal{I} \models r(a, b)$ iff $(a, b) \in r^\mathcal{I}$. An interpretation $\mathcal{I}$ is a *model* of a TBox $\mathcal{T}$, denoted $\mathcal{I} \models \mathcal{T}$, if $\mathcal{I} \models \alpha$ for all $\alpha \in \mathcal{T}$; it is a model of an ABox $\mathcal{A}$, written $\mathcal{I} \models \mathcal{A}$, if $\mathcal{I} \models \alpha$ for all $\alpha \in \mathcal{A}$; it is a model of a KB $\mathcal{K}$ if $\mathcal{I} \models \mathcal{T}$ and $\mathcal{I} \models \mathcal{A}$.

Query Language. A *positive existential regular path query (PRPQ)* is a formula $\varphi = \exists \mathbf{x}\, \psi(\mathbf{x})$ where ψ is constructed using $\wedge$ and $\vee$ over atoms of the form $\mathcal{E}(t, t')$ where t, t' are variable or constant names, $\mathcal{E}$ is a regular expression over $\{r, r^- \mid r \in \mathsf{N_R}\} \cup \{A? \mid A \in \mathsf{N_C}\}$, and the tuple $\mathbf{x}$ denotes precisely the free variables in ψ. Note that atoms $A(t)$ are captured using $A?(t, t)$.

We denote with I_φ the set of constant names in φ. A *match for* φ *in* $\mathcal{I}$ is a function $\pi : \mathbf{x} \cup I_\varphi \to \Delta^\mathcal{I}$ such that $\pi(a) = a$, for all $a \in I_\varphi$ and $\mathcal{I}, \pi \models \psi(\mathbf{x})$ under the standard semantics of first-order logic extended with the following rule for atoms of the form $\mathcal{E}(t, t')$: $\mathcal{I}, \pi \models \mathcal{E}(t, t')$ if there is a word $\nu_1 \cdots \nu_n \in L(\mathcal{E})$ and a sequence $d_0, \ldots, d_n \in \Delta^\mathcal{I}$ such that $d_0 = \pi(t)$, $d_n = \pi(t')$, and for all $i \in [1, n]$ we have that (i) if $\nu_i = A?$, then $d_{i-1} = d_i \in A^\mathcal{I}$, and (ii) if $\nu_i = r$ (resp., $\nu_i = r^-$), then $(d_{i-1}, d_i) \in r^\mathcal{I}$ (resp., $(d_i, d_{i-1}) \in r^\mathcal{I}$). A query φ is *entailed by a KB* $\mathcal{K}$, denoted as $\mathcal{K} \models \varphi$, if there is a match for φ in every model $\mathcal{I}$ of $\mathcal{K}$. The *query entailment problem* asks whether a KB $\mathcal{K}$ entails a PRPQ φ. It is well-known that the *query answering problem* can be reduced to query entailment, and that PRPQs are *preserved under homomorphisms*, that is, if $\mathcal{I} \models \varphi$ and there is a homomorphism from $\mathcal{I}$ to $\mathcal{J}$, then also $\mathcal{J} \models \varphi$.

Additional Notation for Transitive Roles. Given some interpretation $\mathcal{I}$, $\mathcal{I}|_\Delta$ denotes the restriction of $\mathcal{I}$ to domain $\Delta \subseteq \Delta^\mathcal{I}$. For $d \in \Delta^\mathcal{I}$ and $r \in \mathsf{N_R^t}$, the *r-cluster of* d *in* $\mathcal{I}$, denoted by $Q_{\mathcal{I},r}(d)$, is the set containing d and all elements $e \in \Delta^\mathcal{I}$ such that both $(d, e) \in r^\mathcal{I}$ and $(e, d) \in r^\mathcal{I}$. We call a set $\mathbf{a} \subseteq \Delta^\mathcal{I}$ an *r-cluster in* $\mathcal{I}$ if $\mathbf{a} = Q_{\mathcal{I},r}(d)$ for some $d \in \Delta^\mathcal{I}$, and an *r-root cluster* if additionally $(d, e) \in r^\mathcal{I}$ for all $d \in \mathbf{a}$ and $e \in \Delta^\mathcal{I} \setminus \mathbf{a}$. Note that both a single element without an r-loop and a single element with an r-loop are r-clusters of size 1; otherwise r-clusters can be viewed as r-cliques.

3 Tree Decompositions

Existing algorithms for QA in expressive DLs, e.g., $\mathcal{SHIQ}$ (without number restrictions on transitive roles), exploit the fact that for answering queries it suffices to consider *canonical models* that are forest-like, roughly consisting of an interpretation of the ABox and a collection of tree-shaped interpretations whose roots are elements of the ABox. We start with showing that for $\mathcal{SQ}$ this tree-model property is lost.

Example 1. *The number restrictions in* $\mathcal{T}$*, cf. Section 1, force that every model of* $\mathcal{T}$ *satisfying* Heart *contains the structure in Fig. 1(a). Moreover, in* $\mathcal{SQ}$ *clusters can be enforced. Let* $\mathcal{T}'$ *be the following TBox, where* $r \in \mathsf{N_R^t}$*:*

$$\{A \sqsubseteq (= 3\, r.B), B \sqsubseteq (= 3\, r.B), A \sqsubseteq \neg B\}.$$

Then, in every model of $\mathcal{T}'$*, an element satisfying* A *roots the structure depicted in Fig. 1(b), where the elements satisfying* B *form an* r*-cluster.*

Nevertheless, we will establish a *tree-like* model property for $\mathcal{SQ}$, showing that it suffices to consider such models for query entailment. We first introduce a basic form of tree decompositions suited for transitive roles. A *tree* is a prefix-closed subset $T \subseteq (\mathbb{N} \setminus \{0\})^*$. A node $w \in T$ is a successor

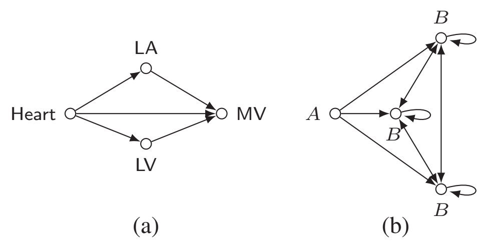

Figure 1: Example 1

of $v \in T$ and v is a predecessor of w if $w = v \cdot i$ for some $i \in \mathbb{N}$. We denote with $w \cdot -1$ the predecessor of w, if it exists.

Definition 1. *A* tree decomposition *of an interpretation $\mathcal{I}$ is pair $(T, \mathfrak{I})$ where T is a tree and $\mathfrak{I}$ is a function that assigns an interpretation $\mathfrak{I}(w) = (\Delta_w, \cdot^{\mathfrak{I}(w)})$ to every $w \in T$, and the following conditions are satisfied:*

1. $\Delta^{\mathcal{I}} = \bigcup_{w \in T} \Delta_w$*;*
2. *for every $w \in T$, we have $\mathfrak{I}(w) = \mathcal{I}|_{\Delta_w}$;*
3. $r^{\mathcal{I}} = \chi_r$ *for* $r \in \mathsf{N}_\mathsf{R}^{nt}$ *and* $r^{\mathcal{I}} = \chi_r^+$ *for* $r \in \mathsf{N}_\mathsf{R}^{t}$*, where* $$\chi_r = \bigcup_{w \in T} r^{\mathfrak{I}(w)};$$
4. *for every $d \in \Delta^{\mathcal{I}}$, the set $\{w \in T \mid d \in \Delta_w\}$ is connected in T.*

The width *of $(T, \mathfrak{I})$ is the maximum domain size of interpretations that occur in the range of $\mathfrak{I}$ minus 1, that is,* $\sup_{w \in T} |\Delta_w| - 1$. *Its* outdegree *is the outdegree of T.*

Unfortunately, this basic tree decomposition does not yet enable tree automata to count over transitive roles (with a small number of states) since the r-successors of an element, say $d \in \Delta^{\mathcal{I}}$, are scattered in the decomposition; see Section 4.1 for further details. To address this, we extend tree decompositions with a third component $\mathfrak{r}$ which assigns to every node $w \in T \setminus \{\varepsilon\}$ a role name $\mathfrak{r}(w)$ and $\perp$ to the root ε. Intuitively, a node labeled with $r = \mathfrak{r}(w)$ is responsible for capturing r-successors of some element(s) in the predecessor of w.

We need some additional notation. Let $(T, \mathfrak{I}, \mathfrak{r})$ be such an extended tree decomposition, and let $w \in T$ and $r \in \mathsf{N}_\mathsf{R}$. We say that $d \in \Delta_w$ *is fresh in* w if $w = \varepsilon$ or $d \notin \Delta_{w \cdot -1}$, and *$r$-fresh in* w if $r = \mathfrak{r}(w)$ and it is either fresh or $r \neq \mathfrak{r}(w \cdot -1)$. We denote with $F(w)$ and $F_r(w)$ the set of all fresh and r-fresh elements in w, respectively. Intuitively, $F_r(w)$ contains all elements which are allowed to have fresh r-successors in the successor nodes of w. Indeed, the following stronger form of tree decompositions implies (among other things) that, for all d and r, there is a unique w with $d \in F_r(w)$.

Definition 2. *An extended tree decomposition $\mathfrak{T} = (T, \mathfrak{I}, \mathfrak{r})$ of an interpretation $\mathcal{I}$ is* canonical *if the following conditions are satisfied for every $w \in T$ with $r = \mathfrak{r}(w)$ and every successor v of w with $s = \mathfrak{r}(v)$:*

(C$_1$) if $(d, e) \in s_1^{\mathfrak{I}(v)}$, then $s_1 = s$, or $d = e$ and $s_1 \in \mathsf{N}_\mathsf{R}^t$;

(C$_2$) if $s \in \mathsf{N}_\mathsf{R}^{nt}$, then $\Delta_v = \{d, e\}$, for some $d \in F(w)$, $e \in F(v)$, and $s^{\mathfrak{I}(v)} = \{(d, e)\}$;

(C$_3$) if $s \in \mathsf{N}_\mathsf{R}^t$ and $r \notin \{\perp, s\}$, there are $d \in F(w)$ and an r-root cluster $\mathbf{a}$ in $\mathfrak{I}(v)$ such that $\Delta_w \cap \Delta_v = \{d\}$ and $d \in \mathbf{a}$; moreover, there is no successor $v' \neq v$ of w satisfying this for d and $\mathfrak{r}(v') = s$;

(C$_4$) if $s \in \mathsf{N}_\mathsf{R}^t$ and $r \in \{\perp, s\}$, then there is an s-root cluster $\mathbf{a}$ in $\mathfrak{I}(v)$ with:

- *(a) $\mathbf{a} \subseteq F_s(w)$;*
- *(b) $\mathbf{a}$ is an s-cluster in $\mathfrak{I}(w)$;*
- *(c) for all $d \in \mathbf{a}$ and $(d, e) \in s^{\mathfrak{I}(w)}$, we have $e \in \Delta_v$;*
- *(d) for all $(d, e) \in s^{\mathfrak{I}(v)}$, $d \in \mathbf{a} \cup F(v)$ or $e \notin F(v)$.*

Definition 2 imposes restrictions on the structural relation between interpretations at neighboring nodes. Condition (C$_1$) expresses that the interpretation at a node labeled with $\mathfrak{r}(w) = r$ interprets essentially only r non-empty (among role names). Condition (C$_2$) is in analogy with standard unravelling over non-transitive roles (Baader et al. 2003). Condition (C$_3$) reflects that interpretations at neighboring nodes with different $\mathfrak{r}$-components do only interact via single elements. Most interestingly, Condition (C$_4$) plays the role of (C$_2$), but for transitive roles. Note that (C$_4$) is based on r-clusters since they can be enforced, see Example 1 above.

3.1 Tree-like Model Property for $\mathcal{SQ}$

As our first main result, we show a *tree-like model property*, in particular, that every model can be *unraveled* into a canonical decomposition of small width. The proof is via a novel unraveling operation tailored for the logic $\mathcal{SQ}$ and canonical decompositions.

Theorem 1. *Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be an $\mathcal{SQ}$ KB and φ a PRPQ with $\mathcal{K} \not\models \varphi$. There is a model $\mathcal{J}$ of $\mathcal{K}$ and a canonical tree decomposition $(T, \mathfrak{I}, \mathfrak{r})$ of $\mathcal{J}$ with (i) $\mathcal{J} \not\models \varphi$, (ii) $\mathfrak{I}(\varepsilon) \models \mathcal{A}$, and (iii) width and outdegree of $(T, \mathfrak{I})$ are bounded by $O(|\mathcal{A}| \cdot 2^{p(|\mathcal{T}|)})$, for some polynomial p.*

Before outlining the proof of Theorem 1, we introduce some additional notation. The *width of an interpretation $\mathcal{I}$* is the minimum k such that $|Q_{\mathcal{I},r}(d)| \leq k$ for all $d \in \Delta^{\mathcal{I}}$, $r \in \mathsf{N}_\mathsf{R}^t$. Moreover, for a transitive role r, we say that e is a *direct r-successor of* d if $(d, e) \in r^{\mathcal{I}}$ but $e \notin Q_{\mathcal{I},r}(d)$, and for each f with $(d, f), (f, e) \in r^{\mathcal{I}}$, we have $f \in Q_{\mathcal{I},r}(d)$ or $f \in Q_{\mathcal{I},r}(e)$; if r is non-transitive, then e is a *direct r-successor of* d if $(d, e) \in r^{\mathcal{I}}$. The *breadth of* $\mathcal{I}$ is the maximum k such that there are $d, d_1, \ldots, d_k$ and a role name r, all d_i are direct r-successors of d, and

– if r is non-transitive, then $d_i \neq d_j$ for all $i \neq j$;
– if r is transitive, then $Q_{\mathcal{I},r}(d_i) \neq Q_{\mathcal{I},r}(d_j)$, for $i \neq j$.

Let now be $\mathcal{I} \models \mathcal{K}$ and $\mathcal{I} \not\models \varphi$. As PRPQs are preserved under homomorphisms, the following lemma implies that we can assume without loss of generality that $\mathcal{I}$ is of bounded width and breadth. The proof of this lemma adapts a result in (Kazakov and Pratt-Hartmann 2009).

Lemma 1. *For each $\mathcal{I} \models \mathcal{K}$, there is a sub-interpretation $\mathcal{I}'$ of $\mathcal{I}$ with $\mathcal{I}' \models \mathcal{K}$ and width and breadth of $\mathcal{I}'$ are bounded by $O(|\mathcal{A}| + 2^{p(|\mathcal{T}|)})$.*

Let $\mathsf{cl}(\mathcal{T})$ be the set of all subconcepts occurring in $\mathcal{T}$, closed under single negation. For each transitive role r, define a binary relation $\rightsquigarrow_{\mathcal{I},r}$ on $\Delta^{\mathcal{I}}$, by taking $d \rightsquigarrow_{\mathcal{I},r} e$ if there is some $(\leqslant n\ r.C) \in \mathcal{T}$ such that $d \in (\leqslant n\ r.C)^{\mathcal{I}}$, $e \in C^{\mathcal{I}}$, and $(d,e) \in r^{\mathcal{I}}$. Based on the transitive, reflexive closure $\rightsquigarrow^*_{\mathcal{I},r}$ of $\rightsquigarrow_{\mathcal{I},r}$, we define, for every $d \in \Delta^{\mathcal{I}}$, the set $\mathsf{Wit}_{\mathcal{I},r}(d)$ of *r-witnesses for d* by:

$$\mathsf{Wit}_{\mathcal{I},r}(d) = \textstyle\bigcup_{e \mid d \rightsquigarrow^*_{\mathcal{I},r} e} Q_{\mathcal{I},r}(e).$$

Intuitively, $\mathsf{Wit}_{\mathcal{I},r}(d)$ contains all r-witnesses of at-most restrictions of some element d, and due to using $\rightsquigarrow^*_{\mathcal{I},r}$, also all witnesses of at-most restrictions of those witnesses and so on. For the stated bounds, it is important that the size of $\mathsf{Wit}_{\mathcal{I},r}(d)$ is bounded as follows:

Lemma 2. *For every $d \in \Delta^{\mathcal{I}}$ and transitive r, we have $|\mathsf{Wit}_{\mathcal{I},r}(d)| \leq |\mathcal{A}| \cdot 2^{q(|\mathcal{T}|)}$, for some polynomial q.*

We describe now the construction of the interpretation $\mathcal{J}$ and its tree decomposition via a possibly infinite unraveling process. Elements of $\Delta^{\mathcal{J}}$ will be either of the form a with $a \in \mathsf{ind}(\mathcal{A})$ or of the form d_x with $d \in \Delta^{\mathcal{I}}$ and some index x. We usually use δ to refer to domain elements in $\mathcal{J}$ (in either form), and define a function $\tau : \Delta^{\mathcal{J}} \to \Delta^{\mathcal{I}}$ by setting $\tau(\delta) = \delta$, for all $\delta \in \mathsf{ind}(\mathcal{A})$, and $\tau(\delta) = d$, for all δ of the form d_x.

To start the construction of $\mathcal{J}$ and $(T, \mathfrak{I}, \mathfrak{r})$, initialize the domain $\Delta^{\mathcal{J}}$ with $\mathsf{ind}(\mathcal{A}) \cup \bigcup_{r \in \mathsf{N}^t_\mathsf{R}} \Delta^r$, where the sets Δ^r are defined as

$$\Delta^r = \{d_r \mid d \in \textstyle\bigcup_{a \in \mathsf{ind}(\mathcal{A})} \mathsf{Wit}_{\mathcal{I},r}(a) \setminus \mathsf{ind}(\mathcal{A})\}.$$

Concept and role names are interpreted in a way such that $\mathcal{J}|_{\mathsf{ind}(\mathcal{A})} = \mathcal{I}|_{\mathsf{ind}(\mathcal{A})}$, and for all $r \in \mathsf{N}^t_\mathsf{R}$ and all $\delta, \delta' \in \mathsf{ind}(\mathcal{A}) \cup \Delta^r$, we have

$$\begin{aligned} &\delta \in A^{\mathcal{J}} \Leftrightarrow \tau(\delta) \in A^{\mathcal{I}}, \text{ for all } A \in \mathsf{N_C}, \quad \text{and} \\ &(\delta, \delta') \in r^{\mathcal{J}} \Leftrightarrow (\tau(\delta), \tau(\delta')) \in r^{\mathcal{I}}. \end{aligned} \qquad (\dagger)$$

Now, initialize $(T, \mathfrak{I}, \mathfrak{r})$ with $T = \{\varepsilon\}$, $\Delta_\varepsilon = \Delta^{\mathcal{J}}$, and $\mathfrak{r}(\varepsilon) = \bot$. This first step ensures that all witnesses of ABox individuals appear in the root.

In the inductive step, we extend $\mathcal{J}$ and $(T, \mathfrak{I}, \mathfrak{r})$ by applying the following rules exhaustively in a fair way.

R$_1$ Let r be non-transitive, $w \in T$, $\delta \in F(w)$, and d a direct r-successor of $\tau(\delta)$ in $\mathcal{I}$ with $\{\delta, d\} \not\subseteq \mathsf{ind}(\mathcal{A})$. Then, add a fresh successor v of w to T, add the fresh element d_v to $\Delta^{\mathcal{J}}$, extend $\mathcal{J}$ by adding $(\delta, d_v) \in r^{\mathcal{J}}$ and $d_v \in A^{\mathcal{J}}$ iff $d \in A^{\mathcal{I}}$, for all $A \in \mathsf{N_C}$, and set $\Delta_v = \{\delta, d_v\}$ and $\mathfrak{r}(v) = r$.

R$_2$ Let r be transitive, $w \in T$, and $\delta_0 \in F(w)$ such that:

(a) $w = \varepsilon$ and $\delta_0 \in \Delta^s$, for some transitive $s \neq r$, or

(b) $w \neq \varepsilon$ and $\mathfrak{r}(w) \neq r$.

Then add a fresh successor v of w to T, and define

$$\Delta = \{e_v \mid e \in \mathsf{Wit}_{\mathcal{I},r}(\tau(\delta_0)) \setminus \{\tau(\delta_0)\}\}.$$

Extend the domain of $\mathcal{J}$ with Δ and the interpretation of concept and role names such that $(\dagger)$ is satisfied for all $\delta, \delta' \in \Delta \cup \{\delta_0\}$. Finally, set $\Delta_v = \Delta \cup \{\delta_0\}$ and $\mathfrak{r}(v) = r$.

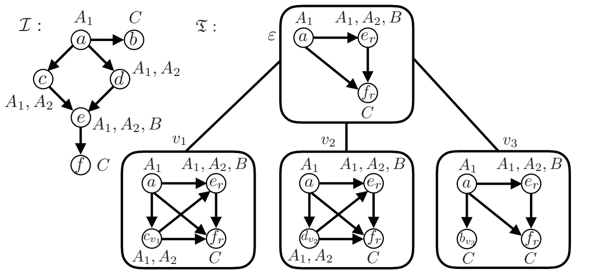

Figure 2: Example 2

R$_3$ Let r be transitive, $w \in T$, $\mathbf{a} \subseteq F_r(w)$ an r-cluster in $\mathfrak{I}(w)$ such that:

(a) $w = \varepsilon$ and $\mathbf{a} \subseteq \Delta^r \cup \mathsf{ind}(\mathcal{A})$, or

(b) $w \neq \varepsilon$ and $\mathfrak{r}(w) = r$.

If there is a direct r-successor e of $\tau(\delta)$ in $\mathcal{I}$ for some $\delta \in \mathbf{a}$ such that $(\delta, \delta') \notin r^{\mathcal{J}}$ for any δ' with $\tau(\delta') = e$, then add a fresh successor v of w to T, and define

$$\begin{aligned} &\Delta = \{f_v \mid f \in \mathsf{Wit}_{\mathcal{I},r}(e) \setminus \mathsf{Wit}_{\mathcal{I},r}(\tau(\delta))\} \quad \text{and} \\ &\Delta_v = \Delta \cup \mathbf{a} \cup \{\delta'' \mid r(\delta', \delta'') \in \mathfrak{I}(w) \text{ for some } \delta' \in \mathbf{a}\}. \end{aligned}$$

Then extend the domain of $\mathcal{J}$ with Δ and the interpretation of concept names such that $(\dagger)$ is satisfied for all $\delta \in \mathbf{a} \cup \Delta$ and $\delta' \in \Delta_v$. Finally, set $\mathfrak{r}(v) = r$.

To finish the construction, let $\mathcal{J}$ be the interpretation obtained in the limit, and set $\mathfrak{I}(w) = \mathcal{J}|_{\Delta_w}$, for all $w \in T$. It is verified in the appendix that $(T, \mathfrak{I}, \mathfrak{r})$ and $\mathcal{J}$ satisfy the conditions from Theorem 1. Notably, τ is a homomorphism from $\mathcal{J}$ to $\mathcal{I}$, thus $\mathcal{J} \not\models \varphi$, due to preservation under homomorphisms.

Rules **R**$_1$–**R**$_3$ are, respectively, in one-to-one correspondence with Conditions (C$_2$)–(C$_4$) in Definition 2. In particular, **R**$_1$ implements the well-known unraveling procedure for non-transitive roles. **R**$_2$ is used to change the 'role component' for transitive roles by creating a fresh node whose interpretation contains all witnesses of the chosen element δ. Finally, **R**$_3$ describes how to unravel direct r-successors in case of transitive roles r. In the definition of Δ it is taken care that witnesses which are 'inherited' from predecessors are not introduced again, in order to preserve at-most restrictions.

We finish the section with an illustrating example.

Example 2. *Let $\mathcal{K}$ be the following KB, where $r \in \mathsf{N}^t_\mathsf{R}$:*

$$(\{A_1 \sqsubseteq (\leqslant 1\ r.B), A_2 \sqsubseteq (\leqslant 1\ r.C)\}, \{A_1(a)\}).$$

*Figure 2 shows a model $\mathcal{I}$ of $\mathcal{K}$ and a canonical decomposition $\mathfrak{T}$ of its unraveling (transitivity connections are omitted). In the initialization phase, the interpretation $\mathfrak{I}(\varepsilon)$ is constructed starting from individual a. Since $a \rightsquigarrow_{\mathcal{I},r} e$ and $e \rightsquigarrow_{\mathcal{I},r} f$, we have $\mathsf{Wit}_{\mathcal{I},r}(a) = \{e, f\}$, thus e_r and f_r are added in this phase. The interpretations $\mathfrak{I}(v_i)$ are introduced using **R**$_3$: In all cases Δ_ε is the cluster $\mathbf{a}$ and $\delta = a$; and, e.g., $\Delta = \{c_{v_1}\}$ for $\mathfrak{I}(v_1)$.*

4 Automata-Based Query Entailment

In this section, we devise an automata-based decision procedure for query entailment in $\mathcal{SQ}$. We start with the necessary background about the used automata model.

Alternating Tree Automata. A tree is *k-ary* if each node has *exactly* k successors. For brevity, we set $[k] = \{-1, 0, \ldots, k\}$. Let Σ be a finite alphabet. A Σ*-labeled tree* is a pair (T, τ) with T a tree and $\tau : T \to \Sigma$ assigns a letter from Σ to each node. A *two-way alternating tree automaton (2ATA)* over Σ-labeled k-ary trees is a tuple $\mathfrak{A} = (Q, \Sigma, q_0, \delta, F)$ where Q is a finite set of *states*, $q_0 \in Q$ is an *initial state*, δ is the *transition function*, and F is the *(parity) acceptance condition* (Vardi 1998). The transition function maps a state q and an input letter $a \in \Sigma$ to a positive Boolean formula over the constants true and false, and variables from $[k] \times Q$. The semantics is given in terms of *runs*, see appendix. As usual, $L(\mathfrak{A})$ denotes the set of trees accepted by $\mathfrak{A}$. Emptiness of $L(\mathfrak{A})$ can be checked in exponential time in the number of states of $\mathfrak{A}$ (Vardi 1998).

General Picture. The leading thought is as follows. If $\mathcal{K} \not\models \varphi$, then, by Theorem 1, there is a model $\mathcal{J}$ of $\mathcal{K}$ and a canonical tree decomposition thereof with small width and outdegree such that $\mathcal{J} \not\models \varphi$. The idea is to design 2ATAs $\mathfrak{A}_{\mathsf{can}}$, $\mathfrak{A}_{\mathcal{K}}$, and $\mathfrak{A}_{\varphi}$ which accept canonical tree decompositions, (tree-like) models of the KB $\mathcal{K}$, and (tree-like) models of the query φ, respectively. Query answering is then reduced to the question whether some tree is accepted by $\mathfrak{A}_{\mathsf{can}}$ and $\mathfrak{A}_{\mathcal{K}}$, but not by $\mathfrak{A}_{\varphi}$. As we shall see, these automata have size exponential in $\mathcal{K}$ and can be constructed in double exponential time. Since 2ATAs can be complemented and intersected in polynomial time, the automaton $\mathfrak{A}_{\mathsf{can}} \wedge \mathfrak{A}_{\mathcal{K}} \wedge \neg\mathfrak{A}_{\varphi}$ is of exponential size, and can be constructed in double exponential time. Checking it for nonemptiness can thus be done in double exponential time. A matching lower bound is inherited from positive existential query answering in $\mathcal{ALC}$ (Calvanese, Eiter, and Ortiz 2014). We thus obtain our main result.

Theorem 2. *PRPQ entailment over $\mathcal{SQ}$-knowledge bases is* 2EXPTIME*-complete.*

Encoding Tree Decompositions. As the underlying interpretation might be infinite, 2ATAs cannot directly work over tree decompositions. Thus, for the desired approach to work, it is crucial to *encode* tree decompositions using a finite alphabet. To this aim, we use an approach similar to (Grädel and Walukiewicz 1999).

Throughout this section, fix a knowledge base $\mathcal{K}$ and let K and k be the bounds on width and outdegree, respectively, obtained in Theorem 1. Then, fix a finite set Δ having $2K$ elements with $\mathsf{ind}(\mathcal{A}) \subseteq \Delta$, and define $\Sigma = \{\bullet\} \cup \Sigma'$, where Σ' is the set of all pairs $(\mathcal{I}, x)$ such that $\mathcal{I}$ is an interpretation where only symbols from $\mathcal{K}$ are interpreted non-empty, $\Delta^{\mathcal{I}} \subseteq \Delta$, $|\Delta^{\mathcal{I}}| \leq K$, and x is either a role name from $\mathcal{K}$ or $\perp$. The symbol $\bullet \in \Sigma$ is used to encode non-existing branches (tree decompositions are not necessarily uniformly branching).

Let (T, τ) be a Σ-labeled tree with Σ as above. For convenience, we use $\mathcal{I}_w$ and r_w to refer to the single components of τ in a node w with $\tau(w) \neq \bullet$, that is, $\tau(w) = (\mathcal{I}_w, r_w)$. Given an element $d \in \Delta$, we say that $v, w \in T$ are d*-connected* iff $d \in \Delta^{\mathcal{I}_u}$ for all u on the unique shortest path from v to w. In case $d \in \Delta^{\mathcal{I}_w}$, we use $[w]_d$ to denote the set of all v which are d-connected to w. We call (T, τ) *consistent* if ε is the only node with $r_\varepsilon = \perp$ and $(\mathcal{I}_w)|_D = (\mathcal{I}_v)|_D$ for all neighbors $v, w \in T$ and $D = \Delta^{\mathcal{I}_w} \cap \Delta^{\mathcal{I}_v}$. A consistent Σ-labeled tree (T, τ) *represents* a triple $(T, \mathfrak{I}, \mathfrak{r})$ of width at most K as follows. The domain underlying $(T, \mathfrak{I}, \mathfrak{r})$ is the set of all elements $[w]_d$ with $w \in T$ and $d \in \Delta^{\mathcal{I}_w}$, and for every $w \in T$, the interpretation $\mathfrak{I}(w)$ is defined as:

$$\Delta_w = \{[w]_d \mid d \in \Delta^{\mathcal{I}_w}\}, \quad A^{\mathfrak{I}(w)} = \{[w]_d \mid d \in A^{\mathcal{I}_w}\},$$
$$r^{\mathfrak{I}(w)} = \{([w]_d, [w]_e) \mid (d, e) \in r^{\mathcal{I}_w}\},$$

for all concept names A and role names r occurring in $\mathcal{K}$; and $\mathfrak{r}(w)$ is just r_w. We denote with $\mathcal{I}_{(T,\tau)}$ the interpretation $\bigcup_{w \in T} \mathcal{I}_w$; clearly, $(T, \mathfrak{I}, \mathfrak{r})$ is a tree decomposition of $\mathcal{I}_{(T,\tau)}$. As a convention, we use $[\varepsilon]_a$ to represent each ABox individual $a \in \mathsf{ind}(\mathcal{A})$ in the encoding. Based on the size $2K$ of Δ, it is not hard to verify that, conversely, for every width K tree decomposition of some $\mathcal{I}$, there is a consistent (T, τ) such that $\mathcal{I}_{(T,\tau)}$ is isomorphic to $\mathcal{I}$.

It is easy to devise a 2ATA $\mathfrak{A}_{\mathsf{can}}$ which accepts an input (T, τ) iff it is consistent and the represented tree decomposition $(T, \mathfrak{I}, \mathfrak{r})$ is canonical. We thus concentrate on the most challenging automata $\mathfrak{A}_{\mathcal{K}}$ and $\mathfrak{A}_{\varphi}$.

4.1 Knowledge Base Automaton $\mathfrak{A}_{\mathcal{K}}$

The automaton $\mathfrak{A}_{\mathcal{K}}$ is the intersection of two automata $\mathfrak{A}_{\mathcal{A}}$ and $\mathfrak{A}_{\mathcal{T}}$ verifying that the input satisfies the ABox and the TBox, respectively. Note that, by Point (ii) of Theorem 1, we can assume that the ABox is satisfied in the root; thus, an automaton $\mathfrak{A}_{\mathcal{A}}$ checking whether $\mathcal{I}_{(T,\tau)} \models \mathcal{A}$ just has to check the label $\tau(\varepsilon)$, see the appendix.

For the design of the automaton $\mathfrak{A}_{\mathcal{T}}$, assume w.l.o.g. that $\mathcal{T}$ is of the form $\{\top \sqsubseteq C_{\mathcal{T}}\}$ and $C_{\mathcal{T}}$ is in negation normal form. We present the main ideas of the construction of $\mathfrak{A}_{\mathcal{T}}$, see the appendix for further details. In its 'outer loop', the automaton visits every domain element d in state $C_{\mathcal{T}}(d)$. This is realized using the initial state q_0, and states of the form $D(d)$, D a sub-concept of $C_{\mathcal{T}}$ and $d \in \Delta$ via the following transitions for every $(\mathcal{I}, x) \in \Sigma$:

$$\delta(q_0, (\mathcal{I}, x)) = \textstyle\bigwedge_{1 \leq i \leq K}(i, q_0) \wedge \bigwedge_{d \in \Delta^{\mathcal{I}}}(0, C_{\mathcal{T}}(d))$$
$$\delta(q_0, \bullet) = \mathsf{true}$$

If $\mathfrak{A}_{\mathcal{T}}$ visits w in a state $D(d)$ this presents the obligation to verify that, in the represented model, $[w]_d$ satisfies D. The Boolean operations are dealt with using the following transitions, for every $(\mathcal{I}, x) \in \Sigma$:

$$\delta(A(d), (\mathcal{I}, x)) = \text{if } d \in A^{\mathcal{I}}, \text{ then } \mathsf{true} \text{ else } \mathsf{false}$$
$$\delta(\neg A(d), (\mathcal{I}, x)) = \text{if } d \notin A^{\mathcal{I}}, \text{ then } \mathsf{true} \text{ else } \mathsf{false}$$
$$\delta((C_1 \sqcup C_2)(d), (\mathcal{I}, x)) = (0, C_1(d)) \vee (0, C_2(d))$$
$$\delta((C_1 \sqcap C_2)(d), (\mathcal{I}, x)) = (0, C_1(d)) \wedge (0, C_2(d))$$

For states of the form $(\sim\, n\ r.D)(d)$ we have to be more careful. The naive approach for counting the number of r-successors of d satisfying D would be to count the number of r-successors satisfying D in the interpretation associated to the current node, and then move to all other nodes where d appears. Since interpretations associated to neighboring nodes might overlap, to avoid double counting, we have to store (in the states) all elements that have already been counted in the current node before changing the node. However, since the domain in each node has size exponential in $|\mathcal{T}|$, we need doubly exponentially many states for this task. Since this naive approach does not result in optimal complexity, we pursue an alternative approach, based on canonicity, leading to only exponentially many states.

Our approach is based on characterizing how r-successors of an element can be uniquely identified in canonical tree decompositions. Assume some $(T, \tau) \in L(\mathfrak{A}_{\mathsf{can}})$ and let r be a role name. In what follows, we assume that the notions of 'fresh' and 'r-fresh' are lifted to the encoding in the straightforward way. An *r-path from $[w]_d$ to $[v]_e$ in (T, τ)* is a sequence $d_0, w_0, d_1, \ldots, w_{n-1}, d_n$ such that $d = d_0$, $e = d_n$, $w_0 \in [w]_d$, $w_{n-1} \in [v]_e$, and $(d_i, d_{i+1}) \in r^{\mathcal{I}_{w_i}}$, for all $0 \leq i < n$. It is *downward* if, for all $0 < i < n$, w_i is a successor of w_{i-1} and d_i is contained in an r-root cluster of w_i. We then have:

Lemma 3. *For $(T, \tau) \in L(\mathfrak{A}_{\mathsf{can}})$, we have $([w]_d, [v]_e) \in r^{\mathcal{I}_{(T,\tau)}}$ iff one of the following is true:*

- *r is non-transitive and $(d, e) \in r^{\mathcal{I}_\varepsilon}$ or $(d, e) \in r^{\mathcal{I}_v}$, d is fresh in w, and v is a successor of w, or*
- *r is transitive, and there is an r-path $d_0, w_0, \ldots, d_n$ from $[w]_d$ to $[v]_e$ such that one of the following holds:*
  - ***A*** *$d_0 \in F_r(w_0) \cup F_r(w_0 \cdot -1)$, $d_1 \in F_r(w_0)$, and $d_0, \ldots, d_n$ is downward, or*
  - ***B*** *$d_0 \in F_r(w_0)$, $d_1 \notin F_r(w_0)$, and if $n > 1$, then $d_1, \ldots, d_n$ is downward and $w_1 \cdot -1 \in [w]_{d_1}$ is an ancestor of w_0 such that $d_1 \in F_r(w_1 \cdot -1)$.*

This lemma suggests the following approach for verifying the obligation $(\sim\, n\ r.D)(d)$ at some node w. If r is non-transitive, 'navigate' with the automaton to the (unique!) w^* such that $d \in F(w^*)$ and count the r-successors of d in the successors v of w^*, or in ε. If r is transitive, navigate with the automaton to the unique w^* such that $d \in F_r(w^*)$ and change to a state $q^*_{(\sim n\ r.D),d}$, starting from which $\mathfrak{A}_\mathcal{T}$ systematically scans the r-successors according to **A** and **B**. We concentrate on verifying at-least restrictions, at-most restrictions are completely complementary.

Assume $\tau(w^*) = (\mathcal{I}, x)$, and let $\mathbf{a}_1, \ldots, \mathbf{a}_\ell$ be all r-clusters in $\mathcal{I}$ reachable from d (including $Q_{\mathcal{I},r}(d)$), and let $a_1, \ldots, a_\ell$ be representatives of these clusters. Moreover, let N be the set of all tuples $\mathbf{n} = (n_1, \ldots, n_\ell)$ such that $\sum_i n_i = n$. Then, the transition $\delta(q^*_{(\geqslant n\ r.D),d}, (\mathcal{I}, x))$ is defined as

$$\bigvee_{\mathbf{n} \in N} \bigvee_{X \subseteq [1,\ell]} \bigwedge_{i \in X} (0, q^{\mathbf{A}}_{(\geqslant n_i\ r.D),a_i}) \wedge \bigwedge_{i \in [1,\ell] \setminus X} (0, q^{\mathbf{B}}_{(\geqslant n_i\ r.D),a_i}).$$

Thus, $\mathfrak{A}_\mathcal{T}$ guesses a distribution of n to the reachable clusters. Moreover, it guesses from which clusters it starts paths of the shape **A** and **B**. For both guesses, it verifies that the chosen a_i is r-fresh (for **A**) or not (for **B**), and continues in states $q^{\downarrow}_{(\geqslant n\ r.D)}$ and $q^{\uparrow}_{(\geqslant n\ r.D)}$, respectively. This is done using the following transitions:

$$\delta(q^{\mathbf{A}}_{(\geqslant n\ r.D),d}, (\mathcal{I}, x)) = (0, F_{r,d}) \wedge (0, q^{\downarrow}_{(\geqslant n\ r.D),d})$$

$$\delta(q^{\mathbf{B}}_{(\geqslant n\ r.D),d}, (\mathcal{I}, x)) = (0, \overline{F}_{r,d}) \wedge (-1, q^{\uparrow}_{(\geqslant n\ r.D),d})$$

$$\delta(F_{r,d}, (\mathcal{I}, \bot)) = \mathsf{true}$$

$$\delta(F_{r,d}, (\mathcal{I}, x)) = \mathsf{false} \qquad \text{if } x \notin \{r, \bot\}$$

$$\delta(F_{r,d}, (\mathcal{I}, r)) = (-1, F'_{r,d})$$

$$\delta(F'_{r,d}, (\mathcal{I}, x)) = \begin{cases} \mathsf{true} & \text{if } x \notin \{r, \bot\} \text{ or } d \notin \Delta^\mathcal{I}, \\ \mathsf{false} & \text{otherwise,} \end{cases}$$

and complementary transitions for $\overline{F}_{r,d}$. Now, in states $q^{\uparrow}_{(\geqslant n\ r.D),d}$, the automaton goes up until it finds the world where d is r-fresh (corresponding to $w_1 \cdot -1$ in **B**) and looks for downward paths starting from there. This is done by taking setting $\delta(q^{\uparrow}_{(\geqslant n\ r.D),d}, (\mathcal{I}, x)) = \mathsf{false}$ whenever $d \notin \Delta^\mathcal{I}$, and otherwise:

$$\delta(q^{\uparrow}_{(\geqslant n\ r.D),d}, (\mathcal{I}, x)) = (0, q^{\mathbf{A}}_{(\geqslant n\ r.D),d}) \vee (0, q^{\mathbf{B}}_{(\geqslant n\ r.D),d}).$$

It thus remains to describe transitions for states of the form $q^{\downarrow}_{(\geqslant n\ r.D),d}$ at some node w. Such situations represent the obligation to find n r-successors along downward paths from d. Note that the transitions before ensure that $d \in F_r(w)$. In this case, the automaton guesses how many of the n successors it will find locally in the current cluster (using states $p^{\mathsf{loc}}_{m,r,D,d}$), and how many are to be found in successor nodes (using $p^{\mathsf{succ}}_{(\geqslant m\ r.D)}$). Formally, let M be the set of all tuples $\mathbf{m} = (m_0, \ldots, m_k)$ with $\sum_i m_i = n$, and define the transition for $\delta(q^{\downarrow}_{(\geqslant n\ r.D),d}, (\mathcal{I}, x))$ as:

$$\bigvee_{\mathbf{m} \in M} \Big((0, p^{\mathsf{loc}}_{m_0,r,D,d}) \wedge \bigwedge_{i \in [1,k]} (i, p^{\mathsf{succ}}_{(\geqslant m_i\ r.D),d}) \Big)$$

States of the form $p^{\mathsf{loc}}_{n,r,D,d}$ are used to verify that in $Q_{\mathcal{I},r}(d)$ there are n elements satisfying D:

$$\delta(p^{\mathsf{loc}}_{n,r,D,d}, (\mathcal{I}, x)) = \bigvee_{Y \subseteq Q_{\mathcal{I},r}(d), |Y| = n} \bigwedge_{e \in Y} D(e).$$

It remains to give the transitions for states $p^{\mathsf{succ}}_{(\geqslant m\ r.D)}$. To start, we set $\delta(p^{\mathsf{succ}}_{(\geqslant n\ r.D),d}, \sigma) = \mathsf{true}$, whenever $n = 0$; $\delta(p^{\mathsf{succ}}_{(\geqslant n\ r.D),d}, \bullet) = \mathsf{false}$; and $\delta(p^{\mathsf{succ}}_{(\geqslant n\ r.D),d}, (\mathcal{I}, x)) = \mathsf{false}$ whenever $x \neq r$ or d is not in a root cluster of $\mathcal{I}$. For all other cases, let $\mathbf{a}_1, \ldots, \mathbf{a}_\ell$ be all r-clusters reachable from d, except $Q_{\mathcal{I},r}(d)$, let N be again the set of all $\mathbf{n} = (n_1, \ldots, n_\ell)$ such that $\sum_i n_i = n$, and include the transition

$$\delta(p^{\mathsf{succ}}_{(\geqslant n\ r.D),d}, (\mathcal{I}, x)) = \bigvee_{\mathbf{n} \in N} \bigwedge_{i \in [1,\ell]} (0, q^{\mathbf{A}}_{(\geqslant n_i\ r.D),a_i}).$$

Using the parity condition, we make sure that states $q^{\downarrow}_{(\geqslant n\ r.D),d}$ with $n \geq 1$ are not suspended forever, that is, eventualities are finally satisfied.

Lemma 4. *For every $(T, \tau) \in L(\mathfrak{A}_{\mathsf{can}})$, we have $(T, \tau) \in L(\mathfrak{A}_\mathcal{T})$ iff $\mathcal{I}_{(T,\tau)} \models \mathcal{T}$. It can be constructed in time double exponential in $|\mathcal{K}|$, and has exponentially many states in $|\mathcal{K}|$.*

4.2 Query Automaton $\mathfrak{A}_\varphi$

In previous work, we have observed that the approach for the query automaton taken in (Calvanese, Eiter, and Ortiz 2014) leads to a 2ATA with double exponentially many states in $\mathcal{K}$, and thus not to optimal complexity (Gutiérrez-Basulto, Ibáñez-García, and Jung 2017b). We thus take an alternative approach by first giving an intermediate characterization for when a query has a match, and then show how to exploit this to build a 2ATA with exponentially many states.

Fix a P2RPQ $\varphi = \exists \mathbf{x}\, \psi(\mathbf{x})$. Note first that since for every regular expression $\mathcal{E}$ over some alphabet Γ, one can construct in polynomial time an equivalent non-deterministic finite automaton (NFA) $\mathfrak{B} = (Q_\mathfrak{B}, \Gamma, s_{0\mathfrak{B}}, \Delta_\mathfrak{B}, F_\mathfrak{B})$ (Fürer 1980), we generally assume an NFA-based representation, that is, atoms in φ take the shape $\mathfrak{B}(t, t')$, $\mathfrak{B}$ an NFA. For states $s, s' \in Q_\mathfrak{B}$, write $\mathfrak{B}_{s,s'}$ for the NFA that is obtained from $\mathfrak{B}$ by taking s as initial state and $\{s'\}$ as the set of final states. To give semantics to the automata based representation, we define $\mathcal{I} \models \mathfrak{B}(a, b)$ iff $\mathcal{I} \models \mathcal{E}_\mathfrak{B}(a, b)$, where $\mathcal{E}_\mathfrak{B}$ is a regular expression equivalent to $\mathfrak{B}$.

A *conjunctive regular path query (CRPQ)* is a PRPQ which does not use $\vee$. It is well-known that the PRPQ φ is equivalent to a disjunction $q_1 \vee \ldots \vee q_n$ of CRPQs, where n is exponential in $|\varphi|$. Given a CRPQ p, we denote with $\hat{p}$ the equivalent CRPQ obtained from p by replacing every occurrence of r or r^-, r transitive, with $r \cdot r^*$ or $r^- \cdot (r^-)^*$, respectively. Let (T, τ) be a consistent Σ-labeled tree. In the appendix, we show the following characterization.

Lemma 5. *A function $\pi : \mathbf{x} \cup I_\varphi \to \Delta^{\mathcal{I}_{(T,\tau)}}$ with $\pi(a) = [\varepsilon]_a$, for every $a \in I_\varphi$, is a match for φ in $\mathcal{I}_{(T,\tau)}$ iff there is a q_i such that for every $\mathfrak{B}(t, t')$ in $\hat{q}_i$, there is a sequence*

$$(d_0, s_0), w_1, (d_1, s_1), w_2, \ldots, w_n, (d_n, s_n),$$

where $(d_i, s_i) \in \Delta \times Q_\mathfrak{B}$ and $w_i \in T$ and such that:

(a) $s_0 = s_{0\mathfrak{B}}$, $s_n \in F_\mathfrak{B}$,

(b) $\pi(t) = [w_1]_{d_0}$, $\pi(t') = [w_n]_{d_n}$, *and*

(c) for every $i \in [1, n]$, we have $d_{i-1}, d_i \in \Delta^{\mathcal{I}_{w_i}}$, $w_i \in [w_{i-1}]_{d_{i-1}}$ if $i > 1$, and $\mathcal{I}_{w_i} \models \mathfrak{B}_{s_{i-1},s_i}(d_{i-1}, d_i)$.

We will refer to such sequences as *witness sequences*. The lemma suggests the following approach. In order to check whether φ has a match in $\mathcal{I}_{(T,\tau)}$, the automaton guesses a q_i and tries to find the witness sequences characterizing a match. For this purpose, $\mathfrak{A}_\varphi$ uses as states triples $\langle p, V_l, V_r\rangle$ such that $p \subseteq \hat{q}_i$, $I_p = \emptyset$, and:

– V_l and V_r are sets of expressions of the form $(d, s) \to_\mathfrak{B} x$ and $x \to_\mathfrak{B} (d, s)$, respectively, where $\mathfrak{B}$ is the automaton of some atom $\mathfrak{B}(t, t')$ in $\hat{q}_i$, $s \in Q_\mathfrak{B}$, $d \in \Delta$, $x \in \mathsf{var}(p)$.

Intuitively, when the automaton visits a node w in state $\langle p, V_l, V_r\rangle$, this represents the obligation that each atom $\mathfrak{B}(x, y)$ in p still has to be processed in the sense that all variables occuring in p will be instantiated in the subtree rooted at w, and

– for each $(d, s) \to_\mathfrak{B} x \in V_l$, $\mathfrak{A}_\varphi$ tries to find a suffix of the witness sequence for $\mathfrak{B}(t, t')$ starting with (d, s),

– for each $x \to_\mathfrak{B} (d, s) \in V_r$, $\mathfrak{A}_\varphi$ tries to find a prefix of the witness sequence for $\mathfrak{B}(t, t')$ ending with (d, s).

We describe verbally how the automaton $\mathfrak{A}_\varphi$ acts when visiting a node w in state $\langle p, V_l, V_r\rangle$; the complete transition function is given in the appendix. First, $\mathfrak{A}_\varphi$ non-deterministically chooses a partition $S_0, \ldots, S_k$ (with S_i possibly empty, for all i) of $\mathsf{var}(p)$ and values $d_x \in \Delta^{\mathcal{I}_w}$ for all $x \in S_0$. Intuitively, S_0 contains the variables that are to be instantiated in w, and S_i contains the variables that are to be instantiated in the subtree rooted at $w \cdot i$. Based on the taken choice, $\mathfrak{A}_\varphi$ determines states $\langle p^i, V_l^i, V_r^i\rangle$ which are then sent to the respective successors $i \in [1, k]$ of w. Using the parity condition, we enforce that every variable is instantiated after finitely many of such steps.

We demonstrate on several examples how to compute the states $\langle p^i, V_l^i, V_r^i\rangle$ from $S_0, \ldots, S_k$ and d_x for all $x \in S_0$.

– Assume some $\mathfrak{B}(x, y) \in p$ with $x, y \in S_0$. In this case, $\mathfrak{A}_\varphi$ guesses some $f \in F_\mathfrak{B}$ and verifies (using another set of states) that there is a witness sequence for $\mathfrak{B}(x, y)$ starting with $(d_x, s_{0\mathfrak{B}})$ and ending with (d_y, s_f).

– Assume $\mathfrak{B}(x, y) \in p$ and $x, y \in S_i$ for some $i > 0$. In this case, just put $\mathfrak{B}(x, y)$ into p^i.

– For an atom $\mathfrak{B}(x, y) \in p$ with $x \in S_0$ and $y \in S_i$ for $i > 0$, $\mathfrak{A}_\varphi$ guesses an intermediate tuple (d, s), verifies that there is a witness sequence from $(d_x, s_{0\mathfrak{B}})$ to (d, s) and adds $x \to_\mathfrak{B} (d, s)$ to V_r^i.

– For the treatment of V_l (V_r is similar), assume $(d, s) \to_\mathfrak{B} x \in V_l$. If $x \in S_0$, $\mathfrak{A}_\varphi$ verifies that the sequence has a suffix from (d, s) to (d_x, s_f), for some $s_f \in F_\mathfrak{B}$. If $x \in S_i$, $i > 0$, $\mathfrak{A}_\varphi$ guesses an intermediate pair (d', s'), verifies that there is an infix between (d, s) and (d', s') and includes $(d', s') \to_\mathfrak{B} x \in V_l^i$.

We show in the appendix how to verify the existence of an infix of a witness sequence between two pairs (d, s) and (d', s') as required in the first, third and last item using only exponentially many states. Regarding number of states, observe that there are only exponentially many disjuncts (and thus states) q_i and exponentially many states of the form $\langle p, V_l, V_r\rangle$ as described.

We refer the reader to the appendix for the complete construction and a proof of the following lemma.

Lemma 6. *There is a 2ATA $\mathfrak{A}_\varphi$ such that for every $(T, \tau) \in L(\mathfrak{A}_{\mathsf{can}})$, we have $(T, \tau) \in L(\mathfrak{A}_\varphi)$ iff $\mathcal{I}_{(T,\tau)} \models q$. It can be constructed in exponential time in $|\varphi| + |\mathcal{K}|$ and has exponentially in $|\varphi| + |\mathcal{K}|$ many states.*

5 Discussion and Future Work

The obtained results are both of practical and theoretical interest. From the practical point of view, our complexity results and application demands open up the possibility to include a profile based on $\mathcal{SQ}$ to OWL 2. Note that there is no increase in the computational complexity in comparison with that of $\mathcal{SQ}$ *without* counting over transitive roles. From the theoretical perspective, our techniques are useful for several future lines of research. First, the unraveling lays the groundwork for studying extensions of $\mathcal{SQ}$ with other DL constructors. Second, the technique underlying the query automaton works for standard tree decompositions (it does not rely on canonicity) of bounded outdegree, even if the width is high (exponential in our case). We thus believe that this technique is useful for query answering in other DLs. Finally, the gained understanding of the model-theoretic characteristics of $\mathcal{SQ}$ is an important step towards the development of more practical decision procedures.

As future work, we will tackle the following four interesting problems: (i) The *data complexity* of deciding entailment of PRPQs in $\mathcal{SQ}$. The present techniques give only exponential bounds, but we expect coNP-completeness. (ii) The complexity of deciding entailment of *conjunctive queries (CQs)* in $\mathcal{SQ}$. The proposed automata-based approach yields the same upper bound for PRPQs or CQs, but we expect it to be easier for CQs. (iii) The complexity of deciding query entailment in generalizations of $\mathcal{SQ}$ with *role composition* or regular expressions on roles; or with nominals and (controlled) inverses. (iv) The complexity of query entailment in $\mathcal{SQ}$ over *finite* models. Indeed, $\mathcal{SQ}$ lacks *finite controlability*, that is, query entailment in the finite does not coincide with unrestricted query entailment:

Example 3. *Consider* $\mathcal{A} = \emptyset$, $\mathcal{T} = \{\top \sqsubseteq \exists r.\top\}$, *and* $\varphi = \exists x\, r(x,x)$ *for some* $r \in \mathsf{N}_{\mathsf{R}}^t$. *Clearly,* $(\mathcal{T},\mathcal{A}) \not\models \varphi$, *but for every finite model* $\mathcal{I}$ *of* $(\mathcal{T},\mathcal{A})$, *we have* $\mathcal{I} \models \varphi$.

Acknowledgments

The first author was funded by EU's Horizon 2020 programme under the Marie Skłodowska-Curie grant 663830, the second one by the FWF project P30360, and the third one by the ERC grant 647289 CODA.

References

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

Baget, J.; Bienvenu, M.; Mugnier, M.; and Thomazo, M. 2017. Answering conjunctive regular path queries over guarded existential rules. In *In Proc. of IJCAI-17*, 793–799.

Bienvenu, M.; Ortiz, M.; and Simkus, M. 2015. Regular path queries in lightweight description logics: Complexity and algorithms. *J. Artif. Intell. Res. (JAIR)* 53:315–374.

Calvanese, D.; De Giacomo, G.; Lenzerini, M.; and Vardi, M. Y. 2000. Containment of conjunctive regular path queries with inverse. In *Proc. of KR-00*, 176–185.

Calvanese, D.; Eiter, T.; and Ortiz, M. 2009. Regular path queries in expressive description logics with nominals. In *Proc. of IJCAI-09*, 714–720.

Calvanese, D.; Eiter, T.; and Ortiz, M. 2014. Answering regular path queries in expressive description logics via alternating tree-automata. *Inf. Comput.* 237:12–55.

Dogrusoz, U.; Cetintas, A.; Demir, E.; and Babur, O. 2009. Algorithms for effective querying of compound graph-based pathway databases. *BMC Bioinformatics* 10(1):376.

Eiter, T.; Lutz, C.; Ortiz, M.; and Simkus, M. 2009. Query answering in description logics with transitive roles. In *Proc. of IJCAI-09*, 759–764.

Florescu, D.; Levy, A. Y.; and Suciu, D. 1998. Query containment for conjunctive queries with regular expressions. In *Proc. of PODS-98*, 139–148.

Fürer, M. 1980. The complexity of the inequivalence problem for regular expressions with intersection. In *Proc. of ICALP-80*, 234–245.

Glimm, B.; Lutz, C.; Horrocks, I.; and Sattler, U. 2008. Conjunctive query answering for the description logic SHIQ. *J. Artif. Intell. Res. (JAIR)* 31:157–204.

Glimm, B.; Horrocks, I.; and Sattler, U. 2008. Unions of conjunctive queries in SHOQ. In *Proc. of KR-08*, 252–262.

Grädel, E., and Walukiewicz, I. 1999. Guarded fixed point logic. In *Proc. of LICS-99*, 45–54.

Gutiérrez-Basulto, V.; Ibáñez-García, Y.; and Jung, J. C. 2017a. Number restrictions on transitive roles in description logics with nominals. In *Proc. of AAAI-17*.

Gutiérrez-Basulto, V.; Ibáñez-García, Y.; and Jung, J. C. 2017b. On query answering in description logics with number restrictions on transitive roles. In *Proc. of DL-17*.

Horrocks, I.; Sattler, U.; and Tobies, S. 2000. Practical reasoning for very expressive description logics. *Logic Journal of the IGPL* 8(3):239–263.

Kaminski, M., and Smolka, G. 2010. Terminating tableaux for $\mathcal{SOQ}$ with number restrictions on transitive roles. In *Proc. of the 6th IFIP TC*, 213–228.

Kazakov, Y., and Pratt-Hartmann, I. 2009. A note on the complexity of the satisfiability problem for graded modal logics. In *Proc. of LICS-09*, 407–416.

Kazakov, Y.; Sattler, U.; and Zolin, E. 2007. How many legs do I have? Non-simple roles in number restrictions revisited. In *Proc. of LPAR-07*, 303–317.

Lysenko, A.; Roznovăţ, I. A.; Saqi, M.; Mazein, A.; Rawlings, C. J.; and Auffray, C. 2016. Representing and querying disease networks using graph databases. *BioData Mining* 9(1):23.

Rector, A. L., and Rogers, J. 2006. Ontological and practical issues in using a description logic to represent medical concept systems: Experience from GALEN. In *Proc. of RW-06*, 197–231.

Stefanoni, G.; Motik, B.; Krötzsch, M.; and Rudolph, S. 2014. The complexity of answering conjunctive and navigational queries over OWL 2 EL knowledge bases. *J. Artif. Intell. Res.* 51:645–705.

Stevens, R.; Aranguren, M. E.; Wolstencroft, K.; Sattler, U.; Drummond, N.; Horridge, M.; and Rector, A. L. 2007. Using OWL to model biological knowledge. *International Journal of Man-Machine Studies* 65(7):583–594.

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

Wolstencroft, K.; Brass, A.; Horrocks, I.; Lord, P.; Sattler, U.; Turi, D.; and Stevens, R. 2005. A little semantic web goes a long way in biology. In *Proc. of ISWC-05*.