Artificial Intelligence 314 (2023) 103808

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

Answering regular path queries mediated by unrestricted $\mathcal{SQ}$ ontologies

Víctor Gutiérrez-Basulto[a,*], Yazmín Ibáñez-García[a], Jean Christoph Jung[b], Filip Murlak[c]

[a] School of Computer Science and Informatics, Cardiff University, UK
[b] Faculty for Mathematics, Natural Sciences, Economics and Computer Science, University of Hildesheim, Germany
[c] Faculty of Mathematics, Informatics and Mechanics, University of Warsaw, Poland

A R T I C L E   I N F O

Article history:
Received 22 July 2021
Received in revised form 11 October 2022
Accepted 16 October 2022
Available online 21 October 2022

Keywords:
Ontology-mediated query answering
Description logic
Number restrictions
Data complexity

A B S T R A C T

A prime application of description logics is ontology-mediated query answering, with the query language often reaching far beyond instance queries. Here, we investigate this task for positive existential two-way regular path queries and ontologies formulated in the expressive description logic $\mathcal{SQ}_u$, where $\mathcal{SQ}_u$ denotes the extension of the basic description logic $\mathcal{ALC}$ with transitive roles ($\mathcal{S}$) and qualified number restrictions ($\mathcal{Q}$) which can be *unrestrictedly* applied to both non-transitive and transitive roles ($\cdot_u$). Notably, the latter is usually forbidden in expressive description logics. As the main contribution, we show decidability of ontology-mediated query answering in that setting and establish tight complexity bounds, namely 2ExpTime-completeness in combined complexity and coNP-completeness in data complexity. Since the lower bounds are inherited from the fragment $\mathcal{ALC}$, we concentrate on providing upper bounds. As main technical tools we establish a tree-like countermodel property and a characterization of when a query is not satisfied in a tree-like interpretation. Together, these results allow us to use an automata-based approach to query answering.

1. Introduction

The use of ontologies for data management has gained a lot of popularity in various research fields such as knowledge representation and reasoning, and databases. For instance, ontologies have been successfully used to integrate and access data [1,2]. The main characteristic of the ontology-based data access (OBDA) paradigm is that data can be queried through the lens of an ontology which comes with several benefits. On the one hand, the ontology enriches the data with domain knowledge, resulting in more complete answers to queries. On the other hand, the ontology can provide a more convenient vocabulary to the user for formulating queries.

In the context of OBDA, the main reasoning problem is *ontology-mediated query answering*, which is defined as follows: the input consists of data $\mathcal{A}$, a query φ, and an ontology $\mathcal{T}$ and the task is to compute all answers to the query over the data relative to the ontology. In this context, the data $\mathcal{A}$ is assumed to be incomplete, and a tuple $\mathbf{a}$ is an *answer of* φ *over* $\mathcal{A}$ *relative to* $\mathcal{T}$ if $\varphi(\mathbf{a})$ is entailed by $\mathcal{T}$ and $\mathcal{A}$.

* Corresponding author.
E-mail addresses: gutierrezbasultov@cardiff.ac.uk (V. Gutiérrez-Basulto), ibanezgarciay@cardiff.ac.uk (Y. Ibáñez-García), jungj@uni-hildesheim.de (J.C. Jung), f.murlak@uw.edu.pl (F. Murlak).

https://doi.org/10.1016/j.artint.2022.103808

Depending on the application at hand, there are various options both for ontology and query language, that is, formalisms to express ontologies and queries, respectively. For the latter, one typically uses database-inspired query languages such as conjunctive or navigational queries. A popular choice for the former are description logics (DLs) which are a family of logic-based knowledge representation languages [3]. More precisely, they are fragments of first-order logic restricted to unary and binary symbols. Description logics are an attractive choice as ontology languages since they do offer a good balance between expressivity and complexity. Indeed, they provide the logical basis of the web ontology language OWL 2 [4].

Ontology-mediated query answering has been extensively investigated for different combinations of DL and query languages, resulting in a plethora of complexity results and algorithms. One line of research is the study of *expressive DLs* where the main objective is to identify logics including as many features as possible, while supporting decidable reasoning. In particular, considerable effort has been put into the study of query answering in expressive DLs featuring *transitive roles* and *qualified number restrictions* [5–8]. Transitive roles are needed to model intrinsically transitive relations such as hasDescendant or hasPart. Qualified number restrictions can then be used to express numerical properties of concepts as in *every car has (as parts) four wheels.* However, in all the aforementioned works the application of number restrictions to transitive roles is forbidden. Remarkably, this is reflected in the fact that the W3C ontology language OWL 2 forbids such interaction too [9]. As a consequence, users might not be able to properly model knowledge, e.g., about the number of parts a component or a piece has, using OWL languages. As a concrete example, consider the ontology $\mathcal{T}$ consisting of the following statements.

$$\begin{aligned}
\mathsf{Heart} &\sqsubseteq (\geq 1\ \mathsf{hasPart}.\mathsf{MitralValve}) \sqcap \\
&\quad (\leq 1\ \mathsf{hasPart}.\mathsf{MitralValve}) && (1)\\
\mathsf{Heart} &\sqsubseteq (\geq 1\ \mathsf{hasPart}.\mathsf{LeftAtrium}) \sqcap \\
&\quad (\geq 1\ \mathsf{hasPart}.\mathsf{LeftVentricle}) && (2)\\
\mathsf{LeftVentricle} \sqcap \mathsf{LeftAtrium} &\sqsubseteq \bot && (3)\\
\mathsf{LeftVentricle} &\sqsubseteq (\geq 1\ \mathsf{hasPart}.\mathsf{MitralValve}) && (4)\\
\mathsf{LeftAtrium} &\sqsubseteq (\geq 1\ \mathsf{hasPart}.\mathsf{MitralValve}) && (5)
\end{aligned}$$

Intuitively, $\mathcal{T}$ models that the human heart has exactly one mitral valve, by (1), as well as a left atrium and a left ventricle, by (2). By (3), the last two are enforced to be distinct. By (4) and (5), both the left ventricle and the left atrium have as a part a mitral valve. Notably, the axiom (1) applies a non-trivial number restriction to the transitive role hasPart. Obviously, similar modelling issues arise in different application areas, for instance, when modelling that every car has four wheels, or that every human has ten fingers.

This example demonstrates that in important application areas like medicine, there is a need for logics supporting counting on transitive relations. However, existing research shows that it is difficult to design such logics without losing decidability, see Related Work section. Motivated by the above, in this paper is we investigate ontology-mediated query answering in the case when the ontology language supports the application of qualified number restrictions to transitive roles. We focus on the most fundamental Boolean complete such DL which we denote $\mathcal{SQ}_u$ where letter $\mathcal{S}$ denotes the extension of the basic Boolean complete DL $\mathcal{ALC}$ with transitive roles while letter $\mathcal{Q}$ indicates the availability of qualified number restrictions. Throughout the paper we use the subscript $\cdot_u$, for "unrestricted" to avoid confusion with traditional DL nomenclature where, even if both $\mathcal{S}$ and $\mathcal{Q}$ appear in the name of the DL, the interaction of qualified number restrictions and transitive roles is forbidden. As query language, we choose *positive existential two-way regular path queries (P2RPQs)*, which is arguably one of the most expressive query languages, generalizing languages such as conjunctive queries (CQs) [10], unions of CQs, regular path queries (RPQs) [11], and conjunctive two-way regular path queries (C2RPQs) [12]. RPQs and C2RPQs, and thus also P2RPQs are instances of *navigational queries*, which go beyond the expressive power of CQs in that they can express graph reachability. For instance, the query $\varphi(x) = \mathsf{advisorOf}^*(\mathsf{tarski}, x)$ returns all (scientific) descendants of Tarski, using the transitive closure of the advisorOf relation. While (unions of) CQs have been the prime language for ontology-mediated query answering for a long time, navigational queries have recently gained considerable attention [13–16]. Indeed, motivated by applications in the semantic web, the latest W3C standard SPARQL 1.1 includes property paths, related to regular expressions.

1.1. *Contributions and structure of the paper*

The goal of the paper is to pinpoint the computational complexity of answering P2RPQs mediated by $\mathcal{SQ}_u$ ontologies. For this reason, we study the decision problem associated to query answering, *query entailment*, which takes as input only Boolean queries, that is, queries without answer variables, and decides whether the query is entailed over the data and the ontology. We investigate both the combined complexity, where all the data, ontology and query form the input, and data complexity, where query and ontology are assumed to be fixed [17]. Our main results are decidability and tight complexity bounds for the entailment problem. In particular, we show that entailment of P2RPQs mediated by $\mathcal{SQ}_u$ ontologies is 2ExpTime-complete in combined complexity, and coNP-complete in data complexity.

Since the lower bounds follow from previous results for the sublogic $\mathcal{ALC}$ of $\mathcal{SQ}_u$ [18,19], we concentrate on establishing upper bounds.

Our main technical contribution is the development of an *automata-based approach for query entailment* which roughly consists of the following steps:

1. establish a *tree-like countermodel property*, that is, show that if a query φ is not entailed by $\mathcal{T}$ and $\mathcal{A}$, then there is a *tree-like countermodel* witnessing this,
2. construct an automaton $\mathfrak{A}_{\mathcal{T},\mathcal{A}}$ recognizing tree-like models of $\mathcal{T}$ and $\mathcal{A}$,
3. construct an automaton $\mathfrak{A}_{\neg\varphi}$ recognizing tree-like models of $\neg\varphi$.

Observe first that in the above example, the ontology (correctly) enforces that the left atrium and left ventricle *have the same mitral valve* as a part. From a technical perspective, this shows that $\mathcal{SQ}_u$ does *not* enjoy the tree model property. As a consequence, in Step 1, we have to resort to *tree-like* models, defined in terms of tree decompositions. The rationale behind this approach is that query entailment then reduces to the question whether there is a tree-like interpretation that is accepted by both $\mathfrak{A}_{\mathcal{T},\mathcal{A}}$ and $\mathfrak{A}_{\neg\varphi}$, that is, whether the languages recognized by $\mathfrak{A}_{\mathcal{T},\mathcal{A}}$ and $\mathfrak{A}_{\neg\varphi}$ have non-empty intersection.

To obtain our automata-based decision procedure, we thus have to realize each of the steps above. Section 2 provides the necessary background and definitions. In Section 3, we address Step 1 above. More precisely, 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 unravelling operation for $\mathcal{SQ}_u$ that, if the query is not entailed, there is a witness interpretation which has a canonical tree decomposition of bounded degree and width. In Section 4, we develop the foundations for Step 3. In particular, we develop a characterization of when a query is satisfied in a tree-like interpretation. Since P2RPQs are not local, the characterization is based on a careful decomposition of the input query into subqueries. The query is then not satisfied iff there is a certain annotation of the tree-like interpretation with (sets of) subqueries obtained in this way. It is worth mentioning that this characterization does not depend on $\mathcal{SQ}_u$ and is thus of independent interest.

Based on the results in Sections 3 and 4, we provide in Section 5 the automata constructions for Steps 2 and 3. We work with non-deterministic tree automata over infinite trees with weak acceptance conditions. In fact, we only require the existence of an infinite run. For such automata intersection non-emptiness can be decided in polynomial time [20]. Since the size of the automata we construct is bounded doubly exponentially in the size of the input data, ontology, and query, a 2ExpTime upper bound in combined complexity follows. Based on a careful analysis of the constructed automata, we also obtain a coNP upper bound in data complexity. These bounds are tight since entailment of positive existential queries mediated by $\mathcal{ALC}$ ontologies is 2ExpTime-hard in combined complexity [19], and entailment of instance queries mediated by $\mathcal{ALC}$ ontologies is coNP-hard in data complexity [18].

In Section 6 we discuss related work and in Section 7 we conclude and point out some directions for future work.

2. Preliminaries

2.1. The description logic $\mathcal{SQ}_u$

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

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

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

As usual in DLs, ontologies are described by TBoxes and data is described by ABoxes. An $\mathcal{SQ}_u$*-TBox* $\mathcal{T}$ is a finite set of *(general) concept inclusions (CIs)* of the form $C \sqsubseteq D$, where C, D are $\mathcal{SQ}_u$-concepts. An *ABox* $\mathcal{A}$ is a finite non-empty 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 $\{a, b\} \subseteq \mathsf{N_I}$; we denote with $\mathsf{ind}(\mathcal{A})$ the set of individual names occurring in $\mathcal{A}$. Whenever no confusion can arise we will drop the specification of the logic $\mathcal{SQ}_u$ and just talk about TBoxes instead of $\mathcal{SQ}_u$-TBoxes.

The semantics of $\mathcal{SQ}_u$ is defined as usual in terms of interpretations $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$, where $\Delta^\mathcal{I}$ is a non-empty set called the *domain* and $\cdot^\mathcal{I}$ is an *interpretation function* that maps each $A \in \mathsf{N_C}$ to a subset $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$, each $r \in \mathsf{N_R}$ to a subset $r^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$, and each $a \in \mathsf{N_I}$ to an element $a^\mathcal{I} \in \Delta^\mathcal{I}$.

Notice that transitive role names are *not* required to be interpreted as transitive relations; this will be convenient when working with tree decompositions for transitive relations. We denote with $\mathcal{I}^*$ the *transitive closure* of $\mathcal{I}$ defined by setting $\Delta^{\mathcal{I}^*} = \Delta^\mathcal{I}$, $A^{\mathcal{I}^*} = A^\mathcal{I}$ for $A \in \mathsf{N_C}$, $r^{\mathcal{I}^*} = r^\mathcal{I}$ for $r \in \mathsf{N_R^{nt}}$, and $r^{\mathcal{I}^*} = (r^\mathcal{I})^+$ for $r \in \mathsf{N_R^{t}}$. We call an interpretation $\mathcal{I}$ *transitive* if $\mathcal{I} = \mathcal{I}^*$. Naturally, entailment will be defined relative to transitive interpretations only.

The interpretation of complex concepts is defined by taking

$$\begin{aligned}(\bot)^{\mathcal{I}} &= \emptyset\,,\\ (\neg D)^{\mathcal{I}} &= \Delta^{\mathcal{I}} - D^{\mathcal{I}}\,,\\ (C \sqcap D)^{\mathcal{I}} &= C^{\mathcal{I}} \cap D^{\mathcal{I}}\,,\\ (\leqslant n\, r\, B)^{\mathcal{I}} &= \big\{d \in \Delta^{\mathcal{I}} \mid \#\{e \in C^{\mathcal{I}} \mid (d,e) \in r^{\mathcal{I}}\} \le n\big\}\,,\end{aligned}$$

where $\#M$ denotes the cardinality of a set M. An interpretation $\mathcal{I}$ is a *model of a TBox* $\mathcal{T}$, written $\mathcal{I} \models \mathcal{T}$, if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for all CIs $C \sqsubseteq D \in \mathcal{T}$. It is a *model of an ABox* $\mathcal{A}$, written $\mathcal{I} \models \mathcal{A}$, if $\mathsf{ind}(\mathcal{A}) \subseteq \Delta^{\mathcal{I}}$, $a^{\mathcal{I}} = a$ for each $a \in \mathsf{ind}(\mathcal{A})$, $(a,b) \in r^{\mathcal{I}}$ for all $r(a,b) \in \mathcal{A}$, and $a \in A^{\mathcal{I}}$ for all $A(a) \in \mathcal{A}$. The first two conditions constitute the *standard name assumption*; our results carry over to the setting with the *unique name assumption*, where $a^{\mathcal{I}} = b^{\mathcal{I}}$ iff $a = b$ for all $a, b \in \mathsf{ind}(\mathcal{A})$, as well as the setting where individual names are interpreted freely.

An interpretation $\mathcal{I}'$ is a *sub-interpretation* of $\mathcal{I}$, written as $\mathcal{I}' \subseteq \mathcal{I}$, if $\Delta^{\mathcal{I}'} \subseteq \Delta^{\mathcal{I}}$, $A^{\mathcal{I}'} \subseteq A^{\mathcal{I}}$, and $r^{\mathcal{I}'} \subseteq r^{\mathcal{I}}$ for all $A \in \mathsf{N_C}$ and $r \in \mathsf{N_R}$. For $\Sigma \subseteq \mathsf{N_C} \cup \mathsf{N_R}$, $\mathcal{I}$ is a Σ*-interpretation* if $A^{\mathcal{I}} = \emptyset$ and $r^{\mathcal{I}} = \emptyset$ for all $A \in \mathsf{N_C} - \Sigma$ and $r \in \mathsf{N_R} - \Sigma$. The *restriction of* $\mathcal{I}$ *to signature* Σ is the maximal Σ-interpretation $\mathcal{I}'$ with $\mathcal{I}' \subseteq \mathcal{I}$. The *restriction of* $\mathcal{I}$ *to domain* Δ is the maximal sub-interpretation of $\mathcal{I}$ with domain Δ. The union $\mathcal{I} \cup \mathcal{J}$ of $\mathcal{I}$ and $\mathcal{J}$ is an interpretation such that $\Delta^{\mathcal{I} \cup \mathcal{J}} = \Delta^{\mathcal{I}} \cup \Delta^{\mathcal{J}}$, $A^{\mathcal{I} \cup \mathcal{J}} = A^{\mathcal{I}} \cup A^{\mathcal{J}}$, and $r^{\mathcal{I} \cup \mathcal{J}} = r^{\mathcal{I}} \cup r^{\mathcal{J}}$ for all $A \in \mathsf{N_C}$ and $r \in \mathsf{N_R}$.

2.2. Normal form

Throughout the paper we will use a convenient TBox normal form. A concept inclusion is in *normal form* if it takes one of the following shapes:

$$\textstyle\bigsqcap_i A_i \sqsubseteq \bigsqcup_j B_j, \quad A \sqsubseteq (\leqslant n\, r\, B), \quad A \sqsubseteq (\geqslant n\, r\, B),$$

where $A, A_i \in \mathsf{N_C} \cup \{\top\}$, $B \in \mathsf{N_C}$, $B_j \in \mathsf{N_C} \cup \{\bot\}$, and $r \in \mathsf{N_R}$. Note that $\top$, $\sqcup$, and $(\geqslant n\, r\, B)$ are now treated as parts of the syntax, not abbreviations. A TBox $\mathcal{T}$ is in *normal form* if all CIs are in normal form and at-most restrictions involving transitive role names are propagated: with each CI of the form $A \sqsubseteq (\leqslant n\, r\, B)$ with $A \neq \top$ and $r \in \mathsf{N_R^t}$, $\mathcal{T}$ also contains CIs

$$A \sqsubseteq (\leqslant 0\, r\, A') \quad \text{and} \quad \top \sqsubseteq A \sqcup A'$$

for some concept name A'. Note that the latter is equivalent to requiring that $\mathcal{T}$ contains $A \sqsubseteq (\leqslant 0\, r\, \neg A)$, which, however, is not in normal form. Note also that for $A = \top$ the corresponding CI $\top \sqsubseteq (\leqslant 0\, r\, \neg\top)$ is a tautology, which explains why it is not required in the normal form. Every TBox can be normalized in polynomial time by introducing fresh concept names for complex concepts and axiomatizing them accordingly. To make this precise, let us say that a TBox $\mathcal{T}'$ is a *conservative extension* of a TBox $\mathcal{T}$ if each model of $\mathcal{T}'$ is a model of $\mathcal{T}$ and each transitive model of $\mathcal{T}$ can be expanded to a model of $\mathcal{T}'$.

Proposition 1. *For every $\mathcal{SQ}_u$-TBox $\mathcal{T}$ one can compute in polynomial time an $\mathcal{SQ}_u$-TBox $\mathcal{T}'$ in normal form such that $\mathcal{T}'$ is a conservative extension of $\mathcal{T}$.*

Proof. Let us introduce a fresh concept name X_C for every concept C that occurs in $\mathcal{T}$, and additionally a fresh concept name $X_{(\geqslant (n+1)\, r\, C)}$ for each concept of the form $(\leqslant n\, r\, C)$ that occurs in $\mathcal{T}$. Let $\mathcal{T}'$ be the set of the following concept inclusions:

- $X_\bot \sqsubseteq \bot$, $\top \sqsubseteq X_\top$;
- $X_A \sqsubseteq A$, $A \sqsubseteq X_A$ for all concept names A used in $\mathcal{T}$;
- $X_{C \sqcap D} \sqsubseteq X_C$, $X_{C \sqcap D} \sqsubseteq X_D$, $X_C \sqcap X_D \sqsubseteq X_{C \sqcap D}$ for all $C \sqcap D$ that occur in $\mathcal{T}$;
- $X_{\neg C} \sqcap X_C \sqsubseteq \bot$, $\top \sqsubseteq X_{\neg C} \sqcup X_C$ for all $\neg C$ that occur in $\mathcal{T}$;
- $\top \sqsubseteq X_{(\leqslant n\, r\, C)} \sqcup X_{(\geqslant (n+1)\, r\, C)}$, $X_{(\leqslant n\, r\, C)} \sqsubseteq (\leqslant n\, r\, X_C)$, $X_{(\geqslant (n+1)\, r\, C)} \sqsubseteq (\geqslant (n+1)\, r\, X_C)$ for all $(\leqslant n\, r\, C)$ that occur in $\mathcal{T}$;
- $X_{(\leqslant n\, r\, C)} \sqsubseteq (\leqslant 0\, r\, X_{(\geqslant (n+1)\, r\, C)})$ for all $(\leqslant n\, r\, C)$ with $r \in \mathsf{N_R^t}$ that occur in $\mathcal{T}$;
- $X_C \sqsubseteq X_D$ for all $C \sqsubseteq D \in \mathcal{T}$.

Observe that $\mathcal{T}'$ is in normal form: all concept inclusions are in normal form and the penultimate bullet ensures that at-most restrictions involving transitive role names are propagated. Clearly, $\mathcal{T}'$ can be computed from $\mathcal{T}$ in polynomial time. It is routine to verify that $\mathcal{T}'$ is a conservative extension of $\mathcal{T}$. $\square$

2.3. Ontology-mediated query entailment

We next introduce the query language. We concentrate on Boolean queries, that is, queries without answer variables. The extension to queries with answer variables is standard; see, for example, [6]. A *positive existential two-way regular path query (P2RPQ)* is a first-order formula

$$\varphi = \exists \mathbf{x}\, \psi(\mathbf{x})$$

such that $\psi(\mathbf{x})$ is constructed using $\wedge$ and $\vee$ over atoms of the form $\mathcal{E}(t, t')$ where t, t' are variables from $\mathbf{x}$ or individual names from $\mathsf{N_I}$, and $\mathcal{E}$ is a *path expression* defined by the grammar

$$\mathcal{E}, \mathcal{E}' ::= r \mid r^- \mid A? \mid \mathcal{E}^* \mid \mathcal{E} \cup \mathcal{E}' \mid \mathcal{E} \circ \mathcal{E}',$$

where $r \in \mathsf{N_R}$ and $A \in \mathsf{N_C}$. Thus, $\mathcal{E}$ is essentially a regular expression over the (infinite) alphabet $\{r, r^- \mid r \in \mathsf{N_R}\} \cup \{A? \mid A \in \mathsf{N_C}\}$. The set of individual names in φ is denoted with $\mathsf{ind}(\varphi)$.

The semantics of P2RPQs is defined via matches. Let us fix a P2RPQ $\varphi = \exists \mathbf{x}\, \psi(\mathbf{x})$ and an interpretation $\mathcal{I}$. A *match for* φ *in* $\mathcal{I}$ is a function

$$\pi : \mathbf{x} \cup \mathsf{ind}(\varphi) \to \Delta^{\mathcal{I}}$$

such that $\pi(a) = a$, for all $a \in \mathsf{ind}(\varphi)$, and $\mathcal{I}, \pi \models \psi(\mathbf{x})$ under the standard semantics of first-order logic extended with a rule for atoms of the form $\mathcal{E}(t, t')$. More formally, we define:

- $\mathcal{I}, \pi \models \psi_1 \vee \psi_2$ iff $\mathcal{I}, \pi \models \psi_1$ or $\mathcal{I}, \pi \models \psi_2$;
- $\mathcal{I}, \pi \models \psi_1 \wedge \psi_2$ iff $\mathcal{I}, \pi \models \psi_1$ and $\mathcal{I}, \pi \models \psi_2$;
- $\mathcal{I}, \pi \models \mathcal{E}(t, t')$ iff $(\pi(t), \pi(t')) \in \mathcal{E}^{\mathcal{I}}$, where $\mathcal{E}^{\mathcal{I}}$ is defined inductively as follows:

$$\begin{aligned} (r^-)^{\mathcal{I}} &= \{(e, d) \mid (d, e) \in r^{\mathcal{I}}\} & (A?)^{\mathcal{I}} &= \{(d, d) \mid d \in A^{\mathcal{I}}\} \\ (\mathcal{E}^*)^{\mathcal{I}} &= (\mathcal{E}^{\mathcal{I}})^* & (\mathcal{E}_1 \cup \mathcal{E}_2)^{\mathcal{I}} &= \mathcal{E}_1^{\mathcal{I}} \cup \mathcal{E}_2^{\mathcal{I}} \\ (\mathcal{E}_1 \circ \mathcal{E}_2)^{\mathcal{I}} &= \mathcal{E}_1^{\mathcal{I}} \circ \mathcal{E}_2^{\mathcal{I}} \end{aligned}$$

We write $\mathcal{I} \models \varphi$ if there is a match for φ in $\mathcal{I}$. A query φ is *entailed over an ABox* $\mathcal{A}$ *relative to a TBox* $\mathcal{T}$, denoted as

$$\mathcal{T}, \mathcal{A} \models \varphi\,,$$

if $\mathcal{I} \models \varphi$ for every model $\mathcal{I}$ of $\mathcal{T}$ and $\mathcal{A}$ with $\mathcal{I}^* = \mathcal{I}$. That is, entailment is defined as usual, relative to all interpretations that respect the transitivity of role names.

Now, we are ready to define the reasoning problem that we study throughout the paper, *ontology-mediated query entailment*.

Input: ABox $\mathcal{A}$, $\mathcal{SQ}_u$-TBox $\mathcal{T}$, and a P2RPQ φ.
Question: Is φ entailed over $\mathcal{A}$ relative to $\mathcal{T}$?

We shall assume that the input TBox is in normal form. This is possible because replacing a TBox with its conservative extension does not affect entailment of queries over the original concept and role names. Complexity bounds are not affected either, because a normalized conservative extension of a given $\mathcal{SQ}_u$-TBox can be computed in polynomial time, by Proposition 1.

It is standard to show that it is without loss of generality to assume that input queries use no individual names [21]. In a nutshell, we can eliminate each individual name a from the query by replacing it uniformly with a fresh variable x_a and adding to the query the conjunct $A_a(x_a)$ for a fresh concept name A_a; if a is mentioned in the ABox, we also add the assertion $A_a(a)$ to the ABox. The modified query is entailed over the modified ABox if and only if the original query is entailed over the original ABox [21]. We stress that in spite of the input query using no individual names, our decision procedure will manipulate queries with individual names at intermediate stages.

We shall explore both *combined* and *data complexity* of the ontology-mediated query entailment problem [17]. For the former, the complexity is measured in terms of the sizes of the TBox, ABox, and query, while for the latter, it is measured only in the size of the ABox. We assume a natural binary encoding of the input. We write $\|\mathcal{A}\|$, $\|\mathcal{T}\|$, and $\|\varphi\|$ for the sizes of the representations of the ABox $\mathcal{A}$, the TBox $\mathcal{T}$, and the query φ, respectively.

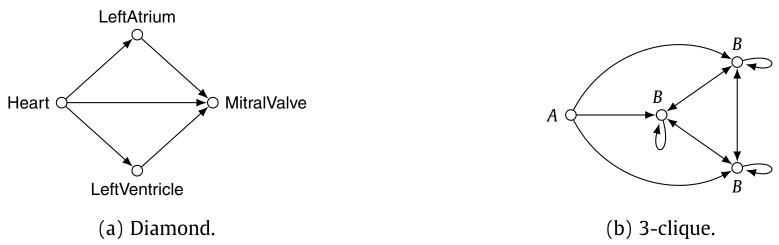


Fig. 1. Structures enforced by $\mathcal{SQ}_u$ TBoxes in Example 1.

2.4. Tree decompositions

A *tree* is a prefix-closed subset $T \subseteq (\mathbb{N} - \{0\})^*$. A node $w \in T$ is a *child* of $v \in T$ and v is the *parent* of w if $w = v \cdot i$ for some $i \in \mathbb{N}$. We write $\mathsf{children}(v)$ for the set of children of v. The *degree* of v is $|\mathsf{children}(v)|$. We say that the node ε, corresponding to the empty sequence, is the *root* of T. A subset $S \subseteq T$ is *connected* if every two nodes in S are connected by a path of nodes from S (possibly going up and down the tree). Every non-empty connected subset S of T can be seen as a tree and has a unique root: the element of S that is closest to the root of T.

We use the standard notion of tree decompositions. Observe, however, that here it is essential to allow for interpretations that do not interpret transitive role names by transitive relations, since transitive relations are dense in the sense that they do not admit a tree decomposition of small width even if the transitive relation is induced by a relation of small width. Formally, a *tree decomposition* $\mathfrak{T}$ *of an interpretation* $\mathcal{I}$ is a pair $(T, \mathfrak{I})$ where T is a tree and $\mathfrak{I}$ is a function that assigns a Σ_v-interpretation $\mathcal{I}_v = (\Delta_v, \cdot^{\mathcal{I}_v})$ to each $v \in T$ such that

(D$_1$) $\mathcal{I} = \bigcup_{v \in T} \mathcal{I}_v$; and
(D$_2$) for every $d \in \Delta^{\mathcal{I}}$, the set $\{v \in T \mid d \in \Delta_v\}$ is connected in T.

We often blur the distinction between a node v of T and the associated interpretation $\mathcal{I}_v$, using the term *bag* for both. The *width* of $\mathfrak{T}$ is $\sup_{v \in T} |\Delta_v| - 1$; the *degree* of $\mathfrak{T}$ is $\sup_{v \in T} |\mathsf{children}(v)|$. Because $\{v \in T \mid d \in \Delta_v\}$ is non-empty and connected for each $d \in \Delta^{\mathcal{I}}$, there is a unique bag v closest to the root ε such that $d \in \Delta_v$. We say that d is *fresh* in this bag, and write $F(v)$ for the set of all elements fresh in v. Note that $F(\varepsilon) = \Delta_\varepsilon$.

In this work we additionally assume that

(D$_3$) for all $u, v \in T$, the restrictions of $\mathcal{I}_u$ and $\mathcal{I}_v$ to domain $\Delta_u \cap \Delta_v$ and signature $\Sigma_u \cap \Sigma_v$ coincide.

This is without loss of generality because one can always replace $\mathcal{I}_v$ by the restriction of $\mathcal{I}$ to domain Δ_v and signature Σ_v.

3. Tree-like countermodel property

In this section we show a tree-like countermodel property for $\mathcal{SQ}_u$: we show that if a query is not entailed over an ABox $\mathcal{A}$ relative to a TBox $\mathcal{T}$, then there is a countermodel with a tree decomposition of bounded width and degree. Let us first demonstrate why $\mathcal{SQ}_u$ does not enjoy the tree model property.

Example 1. Reconsider the TBox $\mathcal{T}$ defined in the introduction. The number restrictions ensure that every model of $\mathcal{T}$ satisfying Heart contains the structure in Fig. 1a, which is clearly not tree-shaped. In fact, in $\mathcal{SQ}_u$ one can even enforce a clique. Consider the TBox

$$\mathcal{T}' = \{A \sqsubseteq (\geqslant 3\, r\, B), B \sqsubseteq (\geqslant 3\, r\, B), \top \sqsubseteq (\leqslant 3\, r\, B), A \sqcap B \sqsubseteq \bot\},$$

where $r \in \mathsf{N}_\mathsf{R}^t$ is a transitive role name. In every model of $\mathcal{T}'$, each element satisfying A is the root of a structure depicted in Fig. 1b where the elements satisfying B form an r-clique; that is, there are r-edges in both directions between the three elements satisfying B. Observe that the number 3 in $\mathcal{T}'$ can be replaced by any number n; thus, large cliques can be enforced.

Consequently, the best we can hope for is a *tree-like* countermodel property; that is, we have to move from trees to tree decompositions. Intuitively, fragments of the countermodel that cannot be unravelled, like the clique in Example 1, will constitute bags. Note that Example 1 shows that bags may need to be large, even exponentially large due to the binary representation of numbers. For the automata-based decision procedure to yield optimal upper bounds, it is useful to consider *canonical* decompositions which we define next.

3.1. Canonical tree decompositions

Processing generic tree decompositions is costly. For instance, to verify that the represented interpretation is a model of an ABox, the automaton needs to maintain the set of assertions to be witnessed in the current subtree, which leads to (at least) exponential data complexity. Similarly, to verify an at-most restriction or an at-least restriction for a single element, the automaton needs to scan unbounded fragments of the tree decomposition for the relevant neighbours. Canonical tree decompositions are designed to make these tasks easier.

Let us fix an ABox $\mathcal{A}$ and an $\mathcal{SQ}_u$ TBox $\mathcal{T}$ in normal form. In canonical tree decompositions elements will be accompanied by certain key neighbours. For an interpretation $\mathcal{I}$ and $r \in \mathsf{N}_\mathsf{R}^\mathsf{t}$, we say that $e \in \Delta^\mathcal{I}$ is a *directly relevant r-successor of* $d \in \Delta^\mathcal{I}$ if $(d, e) \in r^{\mathcal{I}^*}$ and there is a concept inclusion $A \sqsubseteq (\leqslant n\, r\, B)$ in $\mathcal{T}$ such that $d \in A^\mathcal{I}$ and $e \in B^\mathcal{I}$. For an interpretation $\mathcal{I}$, an element $d \in \Delta^\mathcal{I}$, and $r \in \mathsf{N}_\mathsf{R}^\mathsf{t}$, the set $\mathsf{rel}_r^\mathcal{I}(d)$ of *relevant r-successors of d in $\mathcal{I}$* is the least set that contains d and is closed under directly relevant r-successors. For instance, for the TBox $\mathcal{T}'$ from Example 1 and the interpretation $\mathcal{I}$ shown in Fig. 1b, if $A^\mathcal{I} = \{d\}$ and $B^\mathcal{I} = \{e_1, e_2, e_3\}$, then e_1, e_2, e_3 are directly relevant r-successors of d and $\mathsf{rel}_r^\mathcal{I}(d) = \{d, e_1, e_2, e_3\}$. The following is an immediate consequence of the definition.

Lemma 1. *For all $r \in \mathsf{N}_\mathsf{R}^\mathsf{t}$, $d \in \Delta^\mathcal{I}$, and $e \in \mathsf{rel}_r^\mathcal{I}(d)$, we have $\mathsf{rel}_r^\mathcal{I}(e) \subseteq \mathsf{rel}_r^\mathcal{I}(d)$.*

Crucially, in models of $\mathcal{T}$ the sets of relevant r-successors have bounded size.

Lemma 2. *If $\mathcal{I} \models \mathcal{T}$, then for each $r \in \mathsf{N}_\mathsf{R}^\mathsf{t}$ and $d \in \Delta^\mathcal{I}$, $|\mathsf{rel}_r^\mathcal{I}(d)| \leq 2^{\mathsf{poly}(\|\mathcal{T}\|)}$.*

Proof. Let us arrange the elements of $\mathsf{rel}_r^\mathcal{I}(d)$ into a tree T, as follows. Let d be the root of T and then apply the following exhaustively: Choose a leaf e and add as its children all elements $f \in \mathsf{rel}_r^\mathcal{I}(d)$ that are directly relevant r-successors of e but are not yet in T. This way all elements of $\mathsf{rel}_r^\mathcal{I}(d)$ will find their place in T. Now, consider the labelling $\ell : \mathsf{rel}_r^\mathcal{I}(d) \to 2^{\mathsf{N_C}}$ given by

$$\ell(e) = \left\{ B \mid A \sqsubseteq (\leqslant n\, r\, B) \in \mathcal{T}, e \in A^\mathcal{I} \right\} .$$

Let f be a child of e. By construction, we have

- $\ell(e) \subseteq \ell(f)$ if f is a leaf in T, and
- $\ell(e) \subsetneq \ell(f)$ if f is an inner node in T.

Because ℓ only uses concept names occurring in $\mathcal{T}$, the depth of T is bounded linearly in $\|\mathcal{T}\|$. Because $\mathcal{I} \models \mathcal{T}$, the number of directly relevant r-successors of any given element is bounded exponentially in $\|\mathcal{T}\|$: recall that numbers in at-most restrictions are represented in binary. Hence, the branching of T is at most exponential in $\|\mathcal{T}\|$. Overall, the size of T is at most $2^{\mathsf{poly}(\|\mathcal{T}\|)}$. □

Canonical tree decompositions are formalized in Definition 1 below. Intuitively, each non-root bag v keeps track of the interpretation of all concept names, but only a single role name r_v. Condition (C$_1$) ensures that all ABox assertions are already witnessed in the root bag. Conditions (C$_2$)–(C$_4$) express that $\mathcal{T}$ is respected locally: (C$_2$) ensures that all elements satisfy all concept inclusions that do not mention role names, (C$_3$) ensures that suitable at-most restrictions are locally satisfied for fresh elements, and (C$_4$) does the same for at-least restrictions. Conditions (C$_2$) and (C$_4$) imply that the corresponding concept inclusions are globally satisfied in the represented interpretation. For (C$_3$) this is no longer true, because more distant bags may add further successors, possibly violating some at-most restrictions. This is remedied by condition (C$_5$), which restricts the ways in which neighbouring bags may overlap: if they represent different role names or the same non-transitive role name, then they share a single element, but if they represent the same transitive role name, they share an element accompanied by all its relevant successors.

Definition 1. A tree decomposition $\mathfrak{T} = (T, \mathfrak{I})$ is *$(\mathcal{T}, \mathcal{A})$-canonical* if $\Sigma_\varepsilon = \mathsf{N_C} \cup \mathsf{N_R}$ and for each $w \in T - \{\varepsilon\}$ there exists $r_w \in \mathsf{N_R}$ such that $\Sigma_w = \mathsf{N_C} \cup \{r_w\}$ and for each $v \in T$,

(C$_1$) $\mathcal{I}_\varepsilon \models \mathcal{A}$;
(C$_2$) $\mathcal{I}_v$ satisfies all CIs in $\mathcal{T}$ of the forms $\bigsqcap_i A_i \sqsubseteq \bigsqcup_j B_j$ and $A \sqsubseteq (\leqslant n\, r\, B)$ with $r \in \mathsf{N}_\mathsf{R}^\mathsf{t}$;
(C$_3$) for each concept inclusion $A \sqsubseteq (\leqslant n\, r\, B)$ in $\mathcal{T}$ with $r \in \mathsf{N}_\mathsf{R}^\mathsf{nt}$ and each $d \in F(v) \cap A^{\mathcal{I}_v}$, the element d has at most n r-successors satisfying B in $\mathcal{I}_v \cup \bigcup_{w \in \mathsf{children}(v)} \mathcal{I}_w$;
(C$_4$) for each concept inclusion $A \sqsubseteq (\geqslant n\, r\, B)$ in $\mathcal{T}$ and each $d \in F(v) \cap A^{\mathcal{I}_v}$ there exists a child w of v such that the element d has at least n r-successors satisfying B in $\mathcal{I}_v \cup \mathcal{I}_w$;
(C$_5$) for each child w of v there exists $d \in F(v)$ such that

- if $r_w \in \mathsf{N}_\mathsf{R}^\mathsf{t}$ and $r_w \in \Sigma_v$, then $\Delta_v \cap \Delta_w = \mathsf{rel}_{r_w}^{\mathcal{I}_v}(d) = \mathsf{rel}_{r_w}^{\mathcal{I}_w}(d)$,
- if $r_w \in \mathsf{N}_\mathsf{R}^\mathsf{nt}$ or $r_w \notin \Sigma_v$, then $\Delta_v \cap \Delta_w = \{d\}$.

Because $\mathcal{T}$ is in normal form, the sets of relevant successors are faithfully represented in each bag of a $(\mathcal{T}, \mathcal{A})$-canonical tree decomposition.

Lemma 3. *Let $\mathfrak{T} = (T, \mathfrak{I})$ be a $(\mathcal{T}, \mathcal{A})$-canonical tree decomposition of $\mathcal{J}$. Then $\mathsf{rel}_r^{\mathcal{I}_v}(d) = \mathsf{rel}_r^{\mathcal{J}}(d)$ for all $v \in T$, $d \in \Delta_v$, and $r \in \mathsf{N}_\mathsf{R}^\mathsf{t} \cap \Sigma_v$.*

Proof. As the first step towards the claim of the lemma, note that from (C_5) it follows by Lemma 1 that for all $u, v \in T$, $d \in \Delta_u \cap \Delta_v$, and $r \in \mathsf{N}_\mathsf{R}^\mathsf{t} \cap \Sigma_u \cap \Sigma_v$,

$$\mathsf{rel}_r^{\mathcal{I}_u}(d) = \mathsf{rel}_r^{\mathcal{I}_v}(d)\,. \tag{6}$$

Next, we show that if an element $f \in \Delta^{\mathcal{J}}$ is a directly relevant r-successor of an element $e \in \Delta^{\mathcal{J}}$ in $\mathcal{J}$, then $f \in \mathsf{rel}_r^{\mathcal{I}_u}(e)$ for all $u \in T$ with $e \in \Delta_u$ and $r \in \Sigma_u$. The proof is by induction on the length of the shortest r-path connecting e to f in $\mathcal{J}$. If $(e, f) \in r^{\mathcal{J}}$, then $(e, f) \in r^{\mathcal{I}_v}$ for some $v \in T$ with $r \in \Sigma_v$, and it follows immediately that $f \in \mathsf{rel}_r^{\mathcal{I}_v}(e)$. By (6), $f \in \mathsf{rel}_r^{\mathcal{I}_u}(e)$ for all $u \in T$ with $e \in \Delta_u$. Suppose now that the claim holds for all e and f connected by an r-path in $\mathcal{J}$ of length at most k. Take e and f connected by an r-path in $\mathcal{J}$ of length $k+1$. The first element on this path is an r-successor e' of e, connected to f with a path of length at most k. Take $v \in T$ such that $(e, e') \in r^{\mathcal{I}_v}$ and $r \in \Sigma_v$. By the definition of directly relevant r-successors, there exists a concept inclusion $A \sqsubseteq (\leqslant n\, r\, B)$ in $\mathcal{T}$, such that $e \in A^{\mathcal{J}}$ and $f \in B^{\mathcal{J}}$. Consequently, also $e \in A^{\mathcal{I}_v}$. Because at-most restrictions involving transitive roles are propagated, we can conclude from (C_2) that $e' \in A^{\mathcal{I}_v}$, and so $e' \in A^{\mathcal{J}}$. This means that f is a directly relevant r-successor of e' in $\mathcal{J}$. By the inductive hypothesis, $f \in \mathsf{rel}_r^{\mathcal{I}_v}(e')$, and consequently $(e', f) \in r^{\mathcal{I}_v^*}$. Hence, also $(e, f) \in r^{\mathcal{I}_v^*}$ and it follows that f is a directly relevant r-successor of e in $\mathcal{I}_v$. Consequently, $f \in \mathsf{rel}_r^{\mathcal{I}_v}(e)$. By (6), $f \in \mathsf{rel}_r^{\mathcal{I}_u}(e)$ for all $u \in T$ with $e \in \Delta_u$ and $r \in \Sigma_u$. This completes the proof of the claim.

Let us now see that $\mathsf{rel}_r^{\mathcal{I}_v}(d) = \mathsf{rel}_r^{\mathcal{J}}(d)$ for all $v \in T$ with $d \in \Delta_v$ and $r \in \mathsf{N}_\mathsf{R}^\mathsf{t} \cap \Sigma_v$. The inclusion $\mathsf{rel}_r^{\mathcal{I}_v}(d) \subseteq \mathsf{rel}_r^{\mathcal{J}}(d)$ holds because $\mathcal{I}_v$ is the restriction of $\mathcal{J}$ to signature Σ_v and domain Δ_v. For the converse inclusion, it suffices to see that for each $e \in \mathsf{rel}_r^{\mathcal{I}_v}(d)$ and $f \in \Delta^{\mathcal{J}}$, if f is a directly relevant r-successor of e in $\mathcal{J}$, then $f \in \mathsf{rel}_r^{\mathcal{I}_v}(d)$. By the claim above, $f \in \mathsf{rel}_r^{\mathcal{I}_v}(e)$. By Lemma 1, $\mathsf{rel}_r^{\mathcal{I}_v}(e) \subseteq \mathsf{rel}_r^{\mathcal{I}_v}(d)$, and we are done. □

Using Lemma 3, it is easy to show that respecting $\mathcal{T}$ locally in a canonical tree decomposition is sufficient to ensure that the decomposed interpretation is a model of $\mathcal{T}$.

Lemma 4. *If $\mathcal{J}$ has a $(\mathcal{T}, \mathcal{A})$-canonical tree decomposition, then $\mathcal{J}^* \models \mathcal{T} \cup \mathcal{A}$.*

Proof. Let $\mathfrak{T} = (T, \mathfrak{I})$ be a canonical tree decomposition of $\mathcal{J}$. By (C_1), $\mathcal{I}_\varepsilon \models \mathcal{A}$. Because $\mathcal{I}_\varepsilon \subseteq \mathcal{J} \subseteq \mathcal{J}^*$, we have $\mathcal{J}^* \models \mathcal{A}$.

Next, we show that each CI from $\mathcal{T}$ holds in $\mathcal{J}^*$. CIs of the form $\bigsqcap_i A_i \sqsubseteq \bigsqcup_j B_j$ hold in $\mathcal{J}^*$ because they hold in $\mathcal{I}_v$ for each $v \in T$. For at-least restrictions, all necessary witnesses for element d are ensured by the condition (C_4) applied to the node v with $d \in F(v)$. It remains to deal with at-most restrictions.

Take a concept inclusion $A \sqsubseteq (\leqslant n\, r\, B)$ from $\mathcal{T}$ involving a non-transitive role name r and $d \in F(v) \cap A^{\mathcal{J}}$ for some bag v. It follows from (C_5) that r-successors of d can be present only in v and its child bags. Consequently, the condition (C_3) ensures that d has at most n r-successors in $\mathcal{J}$ belonging to $B^{\mathcal{J}}$, and the same holds in $\mathcal{J}^*$.

Finally, take a concept inclusion $A \sqsubseteq (\leqslant n\, r\, B)$ from $\mathcal{T}$ involving a transitive role name r and $d \in \Delta_v \cap A^{\mathcal{J}}$ for some bag v with $r \in \Sigma_v$. By definition, all r-successors of d in $\mathcal{J}^*$ belonging to $B^{\mathcal{J}}$ are included in $\mathsf{rel}_r^{\mathcal{J}}(d)$. By Lemma 3, $\mathsf{rel}_r^{\mathcal{J}}(d) = \mathsf{rel}_r^{\mathcal{I}_v}(d)$, so their number is bounded by n owing to (C_2). □

3.2. Tree-like countermodel property

We are now ready to establish the tree-like countermodel property. We employ a refined variant of unravelling to ensure not only bounded treewidth of the model, but also canonicity of the associated tree-decomposition, which will be instrumental in the construction of the automaton verifying the ABox and the TBox. Note however that the decomposition does not represent the countermodel itself, but an interpretation $\mathcal{J}$ whose transitive closure $\mathcal{J}^*$ is a countermodel. This will make the job of the query automaton harder, forcing it to compute the transitive closure on the fly.

Example 2. Consider the ABox $\mathcal{A} = \{A(a)\}$ and the TBox

$$\begin{aligned}\mathcal{T} = \{\, & A \sqsubseteq (\geqslant 1\, r\, B)\,,\ B \sqsubseteq (\geqslant 1\, r\, C)\,,\ C \sqsubseteq (\geqslant 1\, r\, D)\,,\\ & \top \sqsubseteq (\leqslant 1\, r\, D)\,,\ B \sqsubseteq (\geqslant 1\, s\, A)\,\}\,,\end{aligned}$$

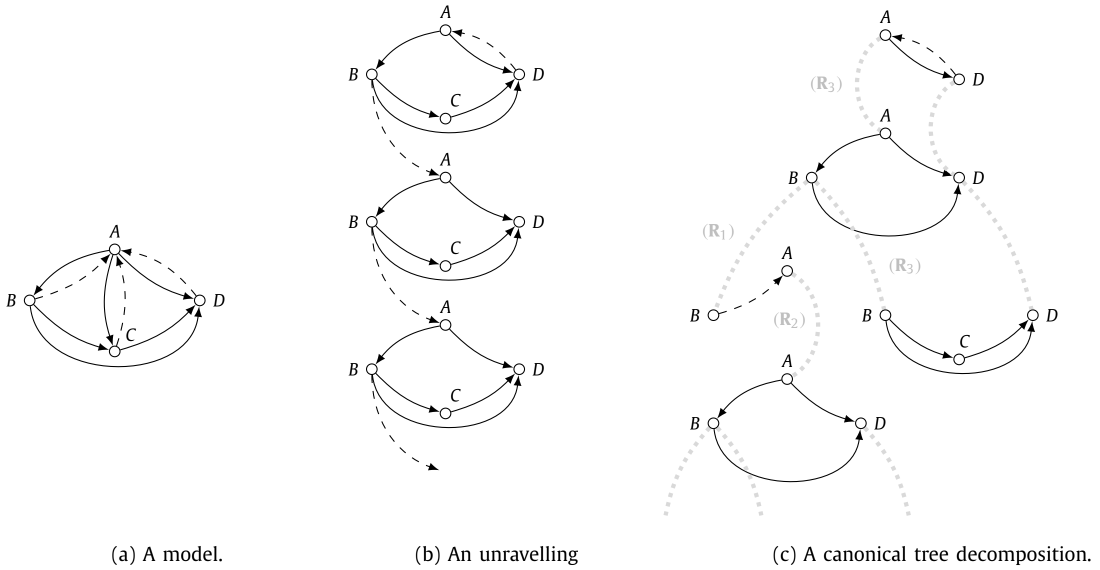


Fig. 2. A model of the TBox in Example 2, an unravelling, and its canonical decomposition.

where r is a transitive role name and s a non-transitive role name.

Let $\mathcal{I}$ be the instance shown in Fig. 2a, where solid and dashed lines represent r-edges and s-edges, respectively, and the unique element satisfying A is the individual name a. It is straightforward to check that $\mathcal{I} \models \mathcal{T} \cup \mathcal{A}$. Observe that the unique element d satisfying D is a directly relevant r-successor of each element in $\mathcal{I}$ and for each $e \in \Delta^{\mathcal{I}}$ we have $\mathsf{rel}_r^{\mathcal{I}}(e) = \{d, e\}$.

Fig. 2b shows an unravelling $\mathcal{J}$ of $\mathcal{I}$. We shall explain later how it is obtained. For now, note that our unravelling operation does not preserve neighbourhoods: for instance, elements satisfying C do not have s-neighbours satisfying A any more. Indeed, only neighbours explicitly required by the TBox will be copied. Note also that $\mathcal{J}$ is not transitive, as the transitive role name r is not interpreted as a transitive relation. This is because the unravelling operation will be constructing directly a tree decomposition (of bounded bag size) and only edges between elements falling inside the same bag will be included.

Fig. 2c shows a canonical tree decomposition of $\mathcal{J}$ of degree 2 and width 2; that is, the underlying tree is binary and bags have size at most 3. Bags are depicted as if they were disjoint, but elements connected by a dotted grey line should be identified with each other. For example, the root bag shares both its elements with its child bag. Note that elements satisfying A and C never meet in the same bag, which explains why there are no edges between them in $\mathcal{J}$, even though their originals in $\mathcal{I}$ are connected with an s-edge and an r-edge.

Theorem 1. *Let $\mathcal{A}$ be an ABox, $\mathcal{T}$ an $\mathcal{SQ}_u$ TBox in normal form, and φ a P2RPQ such that $\mathcal{T}, \mathcal{A} \not\models \varphi$. Then there exists an interpretation $\mathcal{J}$ with a $(\mathcal{T}, \mathcal{A})$-canonical tree decomposition of width and degree at most $\|\mathcal{A}\| \cdot 2^{\mathsf{poly}(\|\mathcal{T}\|)}$ such that $\mathcal{J}^* \models \mathcal{T} \cup \mathcal{A}$, and $\mathcal{J}^* \not\models \varphi$. Moreover, one can additionally guarantee that each non-root bag in the tree decomposition has size and degree bounded by $2^{\mathsf{poly}(\|\mathcal{T}\|)}$.*

Proof. Let us fix an interpretation $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{T} \cup \mathcal{A}$ and $\mathcal{I} \not\models \varphi$. To build a canonical tree decomposition $\mathfrak{T}$, we unravel $\mathcal{I}$. We start from the interpretation of the ABox together with its relevant successors and then apply the extension rules ($\mathbf{R}_1$)–($\mathbf{R}_3$) below: ($\mathbf{R}_1$) performs unravelling of non-transitive role names, ($\mathbf{R}_2$) takes care of the change of roles to a transitive one, and ($\mathbf{R}_3$) realizes further unravelling of transitive role names. Each application of a rule will add a new bag, holding an interpretation obtained by restricting $\mathcal{I}$ and then replacing some elements d with fresh copies d'. In such case we shall call d *the original of* d'. The rules will ensure that mapping each element to its original gives a homomorphism from the constructed interpretation to $\mathcal{I}$. We shall illustrate the construction using the ABox $\mathcal{A}$, TBox $\mathcal{T}$, and instance $\mathcal{I}$ described in Example 2.

For the root bag, we take the restriction of $\mathcal{I}$ to the domain

$$\Delta_\varepsilon = \mathsf{ind}(\mathcal{A}) \cup \bigcup_{r \in \mathsf{N}_\mathsf{R}^\mathsf{t}} \bigcup_{a \in \mathsf{ind}(\mathcal{A})} \mathsf{rel}_r^{\mathcal{I}}(a)\,.$$

In our example, $\mathsf{ind}(\mathcal{A}) = \{a\}$, $\mathsf{rel}_r^{\mathcal{I}}(a) = \{a, d\}$, and the resulting root bag is shown in Fig. 2c; note that it is the only bag that has unrestricted signature and represents all roles. Then, we use the following rules ($\mathbf{R}_1$)–($\mathbf{R}_3$) *ad infinitum*, applying each rule only once to each previously added bag v: for each fresh element d in v and each role r, exactly one rule will fire and it will add (at most) one bag containing all r-successors of d that are needed to satisfy the at-least restrictions enforced by the TBox (see Fig. 2c).

($\mathbf{R}_1$) For each $r \in \mathsf{N}_\mathsf{R}^{\mathsf{nt}}$ and each $d' \in F(v)$, let $d \in \Delta^\mathcal{I}$ be the original of d' (possibly $d = d'$) and let W_0 be the set of originals of all r-successors of d' in $\mathcal{I}_v$. Pick a minimal set $W \subseteq \Delta^\mathcal{I}$ containing $\{d\} \cup W_0$ such that for each $A \sqsubseteq (\geqslant n\, r\, B)$ in $\mathcal{T}$, if $d \in A^\mathcal{I}$, then d has at least n different r-successors in $B^\mathcal{I} \cap W$. Let $\mathcal{I}_{r,d'}$ be the restriction of $\mathcal{I}$ to the signature $\Sigma_w = \mathsf{N}_\mathsf{C} \cup \{r\}$ and domain $\Delta_w = \{d\} \cup (W - W_0)$, with d replaced by d' and each $e \in W - W_0$ replaced by a fresh copy e'. If $\mathcal{I}_{r,d'} \nsubseteq \mathcal{I}_v$, add $\mathcal{I}_{r,d'}$ as a child bag of v.

($\mathbf{R}_2$) For each $r \in \mathsf{N}_\mathsf{R}^{\mathsf{t}} - \Sigma_v$ and each $d' \in F(v)$, let $d \in \Delta^\mathcal{I}$ be the original of d'. Pick a minimal set $W \subseteq \Delta^\mathcal{I}$ containing $\mathsf{rel}_r^\mathcal{I}(d)$ such that for each $A \sqsubseteq (\geqslant n\, r\, B)$ in $\mathcal{T}$, if $d \in A^\mathcal{I}$, then d has at least n different r-successors in $B^\mathcal{I} \cap W$. Let $\mathcal{I}_{r,d'}$ be the restriction of $\mathcal{I}$ to the signature $\Sigma_w = \mathsf{N}_\mathsf{C} \cup \{r\}$ and domain $\Delta_w = \bigcup_{e \in W} \mathsf{rel}_r^\mathcal{I}(e)$, with d replaced by d', and each $f \in \bigcup_{e \in W} \mathsf{rel}_r^\mathcal{I}(e) - \{d\}$ replaced by a fresh copy f'. (Because $d \in W$, a copy of $\mathsf{rel}_r^\mathcal{I}(d)$ is included in $\mathcal{I}_{r,d'}$.) If $\mathcal{I}_{r,d'} \nsubseteq \mathcal{I}_v$, add $\mathcal{I}_{r,d'}$ as a child bag of v.

($\mathbf{R}_3$) For each $r \in \mathsf{N}_\mathsf{R}^{\mathsf{t}} \cap \Sigma_v$, proceed as in ($\mathbf{R}_2$) taking the same restriction of $\mathcal{I}$ but replace each element $f \in \mathsf{rel}_r^\mathcal{I}(d)$ by its copy f' from $\mathcal{I}_v$, and each element $f \in \bigcup_{e \in W} \mathsf{rel}_r^\mathcal{I}(e) - \mathsf{rel}_r^\mathcal{I}(d)$ by a fresh copy f'.

Clearly, $\mathfrak{T}$ is a tree decomposition: Conditions (D$_1$)–(D$_2$) are straightforward, and Condition (D$_3$) holds because each $\mathcal{I}_v$ is isomorphic to a restriction of $\mathcal{I}$. Let us see that $\mathfrak{T}$ is canonical. The signatures Σ_v clearly satisfy the required conditions. Condition (C$_1$) holds because $\mathcal{I}_\varepsilon$ contains the restriction of $\mathcal{I}$ to $\mathsf{ind}(\mathcal{A})$. Condition (C$_2$) holds because each $\mathcal{I}_v$ is isomorphic to a restriction of $\mathcal{I}$.

Let us check (C$_3$). Take a concept inclusion $A \sqsubseteq (\leqslant n\, r\, B)$ from $\mathcal{T}$ involving a non-transitive role name r and $d' \in F(v) \cap A^{\mathcal{I}_v}$ for some bag v. Because r is a non-transitive role name and rules are applied only once to each node v, r-successors of d' can be present only in v and the unique child w of v with $d' \in \Delta_w$ and $r \in \Sigma_w$, created using rule ($\mathbf{R}_1$). Hence, we can work with $\mathcal{I}_v \cup \mathcal{I}_w$. We claim that the originals of any two different r-successors of d' in $\mathcal{I}_v \cup \mathcal{I}_w$ are also different. Let $e', f' \in \Delta_v \cup \Delta_w$ be r-successors of d', and let $e, f \in \Delta^\mathcal{I}$ be their originals. If $e', f' \in \Delta_v$ or $e', f' \in \Delta_w$, then the claim follows because $\mathcal{I}_v$ and $\mathcal{I}_w$ are isomorphic to restrictions of $\mathcal{I}$. Assume that $e' \in \Delta_v$ and $f' \in \Delta_w$. Then, it follows from ($\mathbf{R}_1$) that $e \in W_0$ and $f \in \{d\} \cup (W - W_0)$. Hence, $e \neq f$ unless $e = f = d$. But in the latter case, ($\mathbf{R}_1$) ensures that $e' = f' = d'$, which completes the proof of the claim. Now, because $\mathcal{I}_v$ and $\mathcal{I}_w$ are isomorphic to restrictions of $\mathcal{I}$, it follows that $d \in A^\mathcal{I}$ and that the originals of r-successors of d' belonging to $B^{\mathcal{I}_v \cup \mathcal{I}_w}$ are r-successors of d belonging to $B^\mathcal{I}$. By the claim above, the number of r-successors of d' belonging to $B^{\mathcal{I}_v \cup \mathcal{I}_w}$ is bounded by the number of r-successors of d belonging to $B^\mathcal{I}$, which is at most n because $\mathcal{I} \models \mathcal{T}$.

Condition (C$_4$) holds because all necessary witnesses are provided by rule ($\mathbf{R}_1$) if the involved role name is non-transitive, or by rules ($\mathbf{R}_2$) and ($\mathbf{R}_3$) if it is transitive. Condition (C$_5$) is easy to verify based on ($\mathbf{R}_1$)–($\mathbf{R}_3$).

By Lemma 2, $|\mathsf{rel}_r^\mathcal{I}(d)| \leq 2^{\mathsf{poly}(\|\mathcal{T}\|)}$ for all $r \in \mathsf{N}_\mathsf{R}^{\mathsf{t}}$ and $d \in \Delta^\mathcal{I}$. It follows that

$$|\Delta_\varepsilon| \leq |\mathsf{ind}(\mathcal{A})| \cdot N_1 \cdot 2^{\mathsf{poly}(\|\mathcal{T}\|)},$$

where N_1 is the number of transitive role names that occur in $\mathcal{T}$. Examining the rules ($\mathbf{R}_1$)–($\mathbf{R}_3$) we see that for all $w \in T - \{\varepsilon\}$,

$$|\Delta_w| \leq (1 + |\mathcal{T}| \cdot N_2) \cdot 2^{\mathsf{poly}(\|\mathcal{T}\|)},$$

where N_2 is the maximal number in at-least restrictions in $\mathcal{T}$. That is, $|\Delta_\varepsilon| \leq \|\mathcal{A}\| \cdot 2^{\mathsf{poly}(\|\mathcal{T}\|)}$ and $|\Delta_w| \leq 2^{\mathsf{poly}(\|\mathcal{T}\|)}$ for $w \neq \varepsilon$. The same bounds hold for the degrees of ε and w, because each bag has at most N_3 child bags for each element of the domain, where N_3 is the number of role names that occur in $\mathcal{T}$.

Let $\mathcal{J} = \bigcup_{v \in T} \mathcal{I}_v$. By Lemma 4, $\mathcal{J}^* \models \mathcal{T} \cup \mathcal{A}$. The function mapping each $d' \in \Delta^\mathcal{J}$ to its original $d \in \Delta^\mathcal{I}$ gives a homomorphism from $\mathcal{J}$ to $\mathcal{I}$, and consequently also from $\mathcal{J}^*$ to $\mathcal{I}$. It follows that $\mathcal{J}^* \not\models \varphi$. $\square$

This concludes the technical development of this section. The tree-like countermodel property from Theorem 1 shows that in our search for counter-models we can restrict our attention to tree-like models of $\mathcal{T}$ and $\mathcal{A}$ that admit canonical tree decompositions. In Section 5, we construct a tree automaton that works over (appropriately encoded) tree decompositions and accepts the input iff it is $(\mathcal{T}, \mathcal{A})$-canonical. The forthcoming Section 4 lays foundations for the second ingredient of the decision procedure, which is checking that the interpretation represented by the tree decomposition falsifies the query; the corresponding automaton is also constructed in Section 5.

4. Matching queries in tree decompositions

The goal of this section is to characterize when a query is satisfied in an interpretation $\mathcal{J}$, which is given by its tree decomposition $(T, \mathfrak{I})$, in a way that lends itself to a tree automata implementation. Our approach is rather general since it works for arbitrary tree decompositions; in particular, it does not rely on the canonicity of the tree decomposition. We believe that, as such, it is of independent interest.

More specifically, we aim to use non-deterministic tree automata over infinite trees (a precise definition will be given in Section 5). Since such automata are inherently local, we face the challenge of dealing with non-locality of P2RPQs. Let us illustrate this using a simple example. Consider the query

$$q = \exists x \exists y\, A(x) \wedge r^*(x, y) \wedge B(y).$$

Informally, q has a match in the interpretation $\mathcal{J}$ given by a tree decomposition $(T, \mathfrak{I})$ if there are elements a, b such that $a \in A^{\mathcal{J}}$, $b \in B^{\mathcal{J}}$, and there is an r-path from a to b. Such a path will typically use elements from different bags from $(T, \mathfrak{I})$ and it might go "up and down" in the tree decomposition. Moreover, a and b can be arbitrarily far away from each other in the tree decomposition. These points make it difficult for non-deterministic tree automata to check the existence of such paths.

We resort to the following strategy: rather than treating the input query φ as a whole, we shall consider *splits* of φ into pieces. Intuitively, a split of a query φ is a set of queries P such that φ has a match if all the queries in P have a *local* match, that is, a match in a single bag. Thus, the idea is to *annotate* the nodes of tree decompositions with sets of queries which have local matches and analyze how these pieces can be put together. It will then be the case that a query φ is not entailed by $(T, \mathfrak{I})$ iff there is a valid annotation that does not contain φ. Let us formalize this idea.

To ease the technical development in this section, we make two simplifying assumptions. First, we focus on *conjunctive two-way regular path queries (C2RPQs)* which are P2RPQs of the form

$$\exists \mathbf{x}\, \mathcal{E}_1(t_1, t_1') \wedge \ldots \wedge \mathcal{E}_n(t_n, t_n'),$$

that is, P2RPQs that do not use disjunction $\vee$. In what follows, we denote C2RPQs with letters $p, q, \ldots$. Moreover, we skip the existential quantifiers, view all variables of a C2RPQ q, denoted as $\mathsf{var}(q)$, as implicitly existentially quantified, and often treat a C2RPQ as the set of atoms that occur in it. Establishing the characterization for C2RPQs is sufficient because every P2RPQ is equivalent to a disjunction of C2RPQs, see Section 5 below.

Second, we work with a representation of C2RPQs based on non-deterministic finite automata (NFA). Under that representation, a C2RPQ is a conjunction of atoms of the form $\mathfrak{B}(t, t')$, where $\mathfrak{B}$ is an NFA. We assume that all NFA are over a finite alphabet Ω with

$$\Omega \subseteq \{r, r^- \mid r \in \mathsf{N_R}\} \cup \{A? \mid A \in \mathsf{N_C}\},$$

and use a single initial state and a single final state. Given an NFA $\mathfrak{B}$ and two states s, s' of $\mathfrak{B}$, we denote with $\mathfrak{B}_{s,s'}$ the NFA obtained from $\mathfrak{B}$ by making s the initial state and s' the unique final state. We adjust the definition of *match* to the NFA-based representation by letting $\mathcal{I}, \pi \models \mathfrak{B}(t, t')$ iff for some $n \in \mathbb{N}$ there exist $\nu_1, \ldots, \nu_n \in \Omega$ and $a_0, \ldots, a_n \in \Delta^{\mathcal{I}}$ such that

- $a_0 = \pi(t)$, $a_n = \pi(t')$;
- the word $\nu_1 \ldots \nu_n$ is accepted by $\mathfrak{B}$;
- for all $i \in \{1, \ldots, n\}$, we have if $\nu_i = A?$, then $a_{i-1} = a_i \in A^{\mathcal{I}}$, if $\nu_i = r$, then $(a_{i-1}, a_i) \in r^{\mathcal{I}}$, and if $\nu_i = r^-$, then $(a_i, a_{i-1}) \in r^{\mathcal{I}}$.

As before, for a C2RPQ p we write $\mathcal{I} \models p$ if there is a match of p in $\mathcal{I}$. For a set P of C2RPQs we write $\mathcal{I} \models P$ if $\mathcal{I} \models p$ for each $p \in P$.

We can now formally define what we mean by a split. Let us fix a domain Δ. We say that a C2RPQ p' is a Δ-*instantiation* of a C2RPQ p if it can be obtained from p by consistently replacing an arbitrary number of existentially quantified variables in p with elements from Δ. A Δ-*subdivision* of an atom $\mathfrak{B}(t, t')$ is a conjunction of the form

$$\mathfrak{B}_{s_0,s_1}(t_0, t_1) \wedge \mathfrak{B}_{s_1,s_2}(t_1, t_2) \wedge \cdots \wedge \mathfrak{B}_{s_n,s_{n+1}}(t_n, t_{n+1})$$

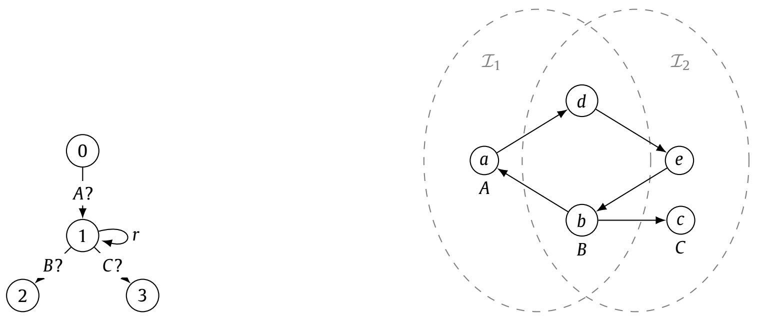
where $n \in \mathbb{N}$, $s_0, s_1, \ldots, s_{n+1}$ are states of $\mathfrak{B}$, s_0 is the initial state, s_{n+1} is the final state, $t_1, t_2, \ldots, t_n \in \Delta$, $t = t_0$, and $t' = t_{n+1}$. Then, p' is a Δ-*subdivision* of p if it can be obtained from p by replacing some atoms with their Δ-subdivisions.

Definition 2. Let p be a C2RPQ and let P be a set of C2RPQs obtained by partitioning (the set of atoms of) a Δ-subdivision of a Δ-instantiation of p. Then, P is called a Δ-*split of* p if

- $\mathsf{var}(p_1) \cap \mathsf{var}(p_2) = \emptyset$ for all different $p_1, p_2 \in P$;
- each $p \in P$ either is a single ground atom over Δ or contains no ground atoms over Δ.

We write $\mathsf{cl}(p, \Delta)$ for the union of all Δ-splits of p.

We illustrate the notion in the following example.



(a) NFA $\mathfrak{B}$.   (b) Interpretation $\mathcal{I} = \mathcal{I}_1 \cup \mathcal{I}_2$.

Fig. 3. The automaton and interpretation in Example 3.

Example 3. Consider the NFA $\mathfrak{B}$ shown in Fig. 3a and the C2RPQ

$$q = \mathfrak{B}_{0,2}(x,y) \wedge \mathfrak{B}_{1,1}(y,x) \wedge \mathfrak{B}_{1,3}(y,z)\,.$$

Note that $\mathfrak{B}_{0,2}$, $\mathfrak{B}_{1,1}$, and $\mathfrak{B}_{1,3}$ correspond to path expressions $A? \circ r^* \circ B?$, r^*, and $r^* \circ C?$, respectively. Consequently, q checks whether there are elements x, y, z satisfying A, B, C, respectively, such that x and y lie on an r-cycle, and z is r-reachable from y. The function π mapping x to a, y to b, and z to c is a match of q in the interpretation $\mathcal{I}$ shown in Fig. 3b. Let $\Delta = \{b, d\}$. Then, queries

$$q_1 = \mathfrak{B}_{0,2}(x,b) \wedge \mathfrak{B}_{1,1}(b,x) \wedge \mathfrak{B}_{1,3}(b,z) \text{ and}$$
$$q_2 = \mathfrak{B}_{0,2}(d,b) \wedge \mathfrak{B}_{1,1}(b,d) \wedge \mathfrak{B}_{1,3}(b,z)$$

are Δ-instantiations of q (the second cannot be matched in $\mathcal{I}$). Moreover, the query

$$\hat{q}_1 = \mathfrak{B}_{0,1}(x,d) \wedge \mathfrak{B}_{1,2}(d,b) \wedge \mathfrak{B}_{1,1}(b,x) \wedge \mathfrak{B}_{1,3}(b,z)$$

is a Δ-subdivision of q_1, obtained by replacing the atom $\mathfrak{B}_{0,2}(x,b)$ with its subdivision $\mathfrak{B}_{0,1}(x,d) \wedge \mathfrak{B}_{1,2}(d,b)$, where $\mathfrak{B}_{0,1}$ and $\mathfrak{B}_{1,2}$ correspond to path expressions $A? \circ r^*$ and $r^* \circ B?$, respectively. It can be verified that the sets

$$P = \left\{\mathfrak{B}_{0,1}(x,d) \wedge \mathfrak{B}_{1,1}(b,x) \wedge \mathfrak{B}_{1,3}(b,z)\,,\ \mathfrak{B}_{1,2}(d,b)\right\},$$
$$P' = \left\{\mathfrak{B}_{0,1}(x,d) \wedge \mathfrak{B}_{1,1}(b,x)\,,\ \mathfrak{B}_{1,3}(b,z)\,,\ \mathfrak{B}_{1,2}(d,b)\right\}$$

are Δ-splits of q. Note that, by the second point in Definition 2, the ground atom $\mathfrak{B}_{1,2}(d,b)$ must occur as a single atom in the split, and, by the first point, atoms that share a variable must end up in the same element of the split. Note also that P' can be partitioned into $P_1 = \left\{\mathfrak{B}_{0,1}(x,d) \wedge \mathfrak{B}_{1,1}(b,x)\right\}$ and $P_2 = \left\{\mathfrak{B}_{1,3}(b,z)\,,\ \mathfrak{B}_{1,2}(d,b)\right\}$ such that $\mathcal{I}_1 \models P_1$ and $\mathcal{I}_2 \models P_2$, where $\mathcal{I}_1$ and $\mathcal{I}_2$ are as indicated in Fig. 3b. We invite the reader to look at this with the intuition of splits provided above in mind, that is, to view $\mathcal{I}_1$ and $\mathcal{I}_2$ as two neighbouring bags in a tree decomposition. Then P_i has a match in $\mathcal{I}_i$, for $i = 1, 2$ which induces the match of q in the union $\mathcal{I}_1 \cup \mathcal{I}_2$. We show in Lemma 5 (2) below that it is always possible to find a $\Delta^{\mathcal{I}_1} \cap \Delta^{\mathcal{I}_2}$-split of a query satisfied in $\mathcal{I}_1 \cup \mathcal{I}_2$.

In the following lemma we collect some useful properties of splits.

Lemma 5. *Let p be a C2RPQ.*

1. *Let $\mathcal{I}$ be an interpretation such that $\mathcal{I} \models P$ for some $\Delta^{\mathcal{I}}$-split P of p. Then $\mathcal{I} \models p$.*
2. *Let $\mathcal{I}_1$ and $\mathcal{I}_2$ be interpretations such that $\mathcal{I}_1 \cup \mathcal{I}_2 \models p$. Then there exists a $\Delta^{\mathcal{I}_1} \cap \Delta^{\mathcal{I}_2}$-split $P_1 \cup P_2$ of p such that $\mathcal{I}_1 \models P_1$ and $\mathcal{I}_2 \models P_2$.*
3. *Each Δ-split of a query from $\mathsf{cl}(p, \Delta)$ is a subset of $\mathsf{cl}(p, \Delta)$.*
4. *If Δ is finite and p has n atoms, each using an automaton with at most m states, then each $q \in \mathsf{cl}(p, \Delta)$ has at most $2n$ atoms and $|\mathsf{cl}(p, \Delta)| \leq (6 \cdot m^2 \cdot |\Delta|^2)^n$.*

Let us comment on the claims of Lemma 5 before we move on to its proof. Item (1) makes precise the intuition of splits discussed at the beginning of the section: a match of a split of p induces a match of p; the correctness of our characterization relies on this. Item (2) states that one can always split matched p in a way compatible with a given decomposition of the interpretation; it will be used to show completeness of the characterization. Item (3) states that $\mathsf{cl}(p, \Delta)$ is closed under taking splits, which is also needed for the completeness proof. Finally, (4) provides upper bounds both on the cardinality of $\mathsf{cl}(p, \Delta)$ and on the sizes of the queries therein; this is not relevant for the characterization itself, but it will matter for establishing precise complexity upper bounds in Section 5. The proof of the lemma, given below, is straightforward. The reader might prefer to skip it and first see how the lemma is used to prove the correctness and completeness of the characterization (Lemma 6).

Proof. (1) If p' is a $\Delta^{\mathcal{I}}$-instantiation of p such that $\mathcal{I} \models p'$, then $\mathcal{I} \models p$. By the semantics of NFA-based C2RPQs, if p'' is a $\Delta^{\mathcal{I}}$-subdivision of p' such that $\mathcal{I} \models p''$, then $\mathcal{I} \models p'$. Finally, if P is a partition of p'' such that $\mathcal{I} \models P$, then $\mathcal{I} \models p''$. Because each $\Delta^{\mathcal{I}}$-split of p is a partition of a $\Delta^{\mathcal{I}}$-subdivision of a $\Delta^{\mathcal{I}}$-instantiation of p, the claim follows.

(2) Let π be a match of p in $\mathcal{I}_1 \cup \mathcal{I}_2$. Let p' be the instantiation obtained from p by replacing every $x \in \mathsf{var}(p)$ satisfying $\pi(x) \in \Delta^{\mathcal{I}_1} \cap \Delta^{\mathcal{I}_2}$ with $\pi(x)$. We shall construct a $\Delta^{\mathcal{I}_1} \cap \Delta^{\mathcal{I}_2}$-subdivision $\widehat{p}$ of p' such that for each atom $\mathfrak{B}(t, t')$ in $\widehat{p}$, either $\mathcal{I}_1, \pi \models \mathfrak{B}(t, t')$ or $\mathcal{I}_2, \pi \models \mathfrak{B}(t, t')$. Based on $\widehat{p}$ we can define the desired $\Delta^{\mathcal{I}_1} \cap \Delta^{\mathcal{I}_2}$-split $P_1 \cup P_2$ of p by letting P_i contain all ground atoms over $\Delta^{\mathcal{I}_1} \cap \Delta^{\mathcal{I}_2}$ that are satisfied in $\mathcal{I}_i$, as well as the conjunction q_i of all remaining atoms satisfied in $\mathcal{I}_i$ under the match π; note that $\mathsf{var}(q_1) \cap \mathsf{var}(q_2) = \emptyset$ because $\pi(x) \notin \Delta^{\mathcal{I}_1} \cap \Delta^{\mathcal{I}_2}$ for all $x \in \mathsf{var}(q)$. Let us see how to construct $\widehat{p}$. Consider an atom $\mathfrak{B}(t, t')$ of p and assume that s and s' are initial and final states, respectively, of $\mathfrak{B}$. Because $\mathcal{I}_1 \cup \mathcal{I}_2, \pi \models \mathfrak{B}(t, t')$, there exist $\nu_1, \nu_2, \ldots, \nu_n \in \Omega$, elements $\pi(t) = a_0, a_1 \ldots, a_n = \pi(t')$, and states $s = s_0, s_1, \ldots, s_n = s'$ of the automaton $\mathfrak{B}$ such that for all $i \in \{1, \ldots, n\}$ the following conditions hold: (a) (s_{i-1}, ν_i, s_i) is a transition of $\mathfrak{B}$; (b) if $\nu_i = A?$, then $a_{i-1} = a_i \in A^{\mathcal{I}_1 \cup \mathcal{I}_2}$; (c) if $\nu_i = r$, then $(a_{i-1}, a_i) \in r^{\mathcal{I}_1 \cup \mathcal{I}_2}$; (d) if $\nu_i = r^-$, then $(a_i, a_{i-1}) \in r^{\mathcal{I}_1 \cup \mathcal{I}_2}$. Let $0 = i_0 < i_1 < i_2 < \ldots i_k < i_{k+1} = n$ be such that

- $a_i \in \Delta^{\mathcal{I}_1} \cap \Delta^{\mathcal{I}_2}$ for all $i \in \{i_1, i_2, \ldots, i_k\}$; and
- $a_i \notin \Delta^{\mathcal{I}_1} \cap \Delta^{\mathcal{I}_2}$ for all $i \notin \{0\} \cup \{i_1, i_2, \ldots, i_k\} \cup \{n\}$.

We ensure the desired property of $\widehat{p}$ by replacing $\mathfrak{B}(t, t')$ with its subdivision

$$\mathfrak{B}_{s, s_{i_1}}(t, a_{i_1}) \wedge \mathfrak{B}_{s_{i_1}, s_{i_2}}(a_{i_1}, a_{i_2}) \wedge \ldots \wedge \mathfrak{B}_{s_{i_k}, s'}(a_{i_k}, t').$$

Indeed, for each $j \in \{0, 1, \ldots, k\}$, $\{a_{i_j+1}, a_{i_j+2}, \ldots, a_{i_{j+1}-1}\} \subseteq \Delta^{\mathcal{I}_\ell} - (\Delta^{\mathcal{I}_1} \cap \Delta^{\mathcal{I}_2})$ for some $\ell \in \{1, 2\}$. Consequently, the conditions (a)–(d) above hold for all $i \in \{i_j, i_j + 1, \ldots, i_{j+1} - 1\}$ with $\mathcal{I}_1 \cup \mathcal{I}_2$ replaced by $\mathcal{I}_\ell$, witnessing that the jth atom of the subdivision of $\mathfrak{B}(t, t')$ holds in $\mathcal{I}_\ell$ under the match π.

(3) Clearly, a Δ-instantiation of a Δ-instantiation of p is a Δ-instantiation p. Similarly, a Δ-subdivision of a Δ-subdivision of p is a Δ-subdivision of p. Consequently, if P is a Δ-split of p and P' is a Δ-split of $p' \in P$, then $\big(P - \{p'\}\big) \cup P'$ is a Δ-split of p; that is, $\big(P - \{p'\}\big) \cup P' \subseteq \mathsf{cl}(p, \Delta)$ and the claim follows.

(4) It is not difficult to see that $\mathsf{cl}(p, \Delta)$ is contained in the set of queries that can be obtained from p by replacing each atom $\mathfrak{B}(t_1, t_2)$ with one of the following, assuming that s_1, s_2 are the initial and final states, respectively, of $\mathfrak{B}$: (a) nothing; (b) itself; (c) $\mathfrak{B}_{s_1, s_1'}(t_1, d_1)$ for some $d_1 \in \Delta$ and some state s_1' of $\mathfrak{B}$; (d) $\mathfrak{B}_{s_2', s_2}(d_2, t_2)$ for some $d_2 \in \Delta$ and some state s_2' of $\mathfrak{B}$; (e) $\mathfrak{B}_{s_1', s_2'}(d_1, d_2)$ for some $d_1, d_2 \in \Delta$ and some states s_1', s_2' of $\mathfrak{B}$; (f) $\mathfrak{B}_{s_1, s_1'}(t_1, d_1) \wedge \mathfrak{B}_{s_2', s_2}(d_2, t_2)$ for some $d_1, d_2 \in \Delta$ and some states s_1', s_2' of $\mathfrak{B}$. Consequently, each query $q \in \mathsf{cl}(p, \Delta)$ has at most $2n$ atoms and $|\mathsf{cl}(p, \Delta)| \leq (2 + 2 \cdot |\mathfrak{B}| \cdot |\Delta| + 2 \cdot |\mathfrak{B}|^2 \cdot |\Delta|^2)^n \leq (6 \cdot m^2 \cdot |\Delta|^2)^n$. $\square$

Recall that the goal is to characterize when a C2RPQ q has *no* match into some tree-like interpretation represented by its tree decomposition, and that for this purpose we want to *annotate* the nodes in the tree decomposition with all relevant queries that 'locally' have a match, and ask for an annotation that avoids q. Here, a query is relevant if it is an element of a split of q. We next give a formal definition of annotations. Since we will need this not only for single C2RPQs but for *unions (disjunctions) of C2RPQs*, we directly do it for sets of C2RPQs.

Definition 3. Let Q be a set of C2RPQs and let $\mathsf{cl}(Q, \Delta) = \bigcup_{q \in Q} \mathsf{cl}(q, \Delta)$ for each set Δ. A *Q-annotation* of a tree decomposition $(T, \mathfrak{I})$ is a mapping Φ that assigns to each $v \in T$ a set $\Phi(v) \subseteq \mathsf{cl}(Q, \Delta_v)$ such that, for each $v \in T$,

(A_1) $\big\{p \in \mathsf{cl}(Q, \Delta_v) \mid \mathcal{I}_v \models p\big\} \subseteq \Phi(v)$;
(A_2) if some Δ_v-split of some $p \in \mathsf{cl}(Q, \Delta_v)$ is a subset of $\Phi(v)$, then $p \in \Phi(v)$;
(A_3) for each neighbour (child or parent) w of v,

$$\Phi(w) \cap \mathsf{cl}(Q, \Delta_v \cap \Delta_w) = \Phi(v) \cap \mathsf{cl}(Q, \Delta_v \cap \Delta_w).$$

Condition (A_1) states that all queries satisfied in a bag $\mathcal{I}_v$ are contained in $\Phi(v)$. Condition (A_2) is responsible for piecing queries together from their splits, within $\Phi(v)$. Finally, condition (A_3) synchronizes neighbouring bags: intuitively, it says that if some relevant query has a match in a neighbouring bag w, then this fact should be known in bag v, and vice versa. In particular, it means that in a valid annotation all $\Phi(v)$ agree on the set of Boolean queries.

The main result of this section, Theorem 2 below, provides the announced characterization and will be the basis for developing tree automata that accept a tree decomposition iff the represented interpretation is a countermodel for all queries from Q.

Theorem 2. *Let Q be a set of C2RPQs without individual names and $(T, \mathfrak{I})$ a tree decomposition of some $\mathcal{J}$. Then, $\mathcal{J} \not\models \bigvee_{q \in Q} q$ iff $(T, \mathfrak{I})$ admits a Q-annotation Φ with $Q \cap \bigcup_{v \in T} \Phi(v) = \emptyset$.*

For the proof of the theorem, let us fix an interpretation $\mathcal{J}$ together with an arbitrary (not necessarily canonical) tree decomposition $(T, \mathfrak{I})$ thereof, and a set Q of C2RPQs. Because the mapping $\Phi(v) = \mathsf{cl}(Q, \Delta_v)$, for all $v \in T$, is a Q-annotation of $(T, \mathfrak{I})$ and pointwise intersection of an arbitrary family of Q-annotations of $(T, \mathfrak{I})$ is a Q-annotation of $(T, \mathfrak{I})$, there exists a unique pointwise minimal Q-annotation Φ_Q of $(T, \mathfrak{I})$. Intuitively, this minimal annotation Φ_Q assigns to every node v all queries from $\mathsf{cl}(Q, \Delta_v)$ which have a match in $\mathcal{J}$.

Lemma 6. *For each $v \in T$ and $p \in \mathsf{cl}(Q, \Delta_v)$,*

$$\mathcal{J} \models p \iff p \in \Phi_Q(v).$$

Based on Lemma 6 and the fact that $Q \subseteq \mathsf{cl}(Q, \Delta)$, it is straightforward to prove the theorem. Suppose first that $(T, \mathfrak{I})$ admits a Q-annotation Φ with $Q \cap \bigcup_{v \in T} \Phi(v) = \emptyset$. Since Φ_Q is pointwise contained in Φ, it satisfies $Q \cap \bigcup_{v \in T} \Phi_Q(v) = \emptyset$. Thus, no query from Q is contained in $\Phi_Q(v)$, for any $v \in T$. Lemma 6 implies that $\mathcal{J} \not\models \bigvee_{q \in Q} q$. Conversely, suppose that $\mathcal{J} \not\models \bigvee_{q \in Q} q$. Lemma 6 yields $q \notin \Phi_Q(v)$ for any $q \in Q$ and $v \in T$, and thus, Φ_Q witnesses $Q \cap \bigcup_{v \in T} \Phi_Q(v) = \emptyset$.

We conclude the section with the missing proof of Lemma 6.

Proof. Let us begin with the right-to-left implication. It is straightforward to see that Φ_Q can be obtained as the limit of the following process: for each v start with $\Phi_Q(v) = \{p \in \mathsf{cl}(Q, \Delta_v) \mid \mathcal{I}_v \models p\}$ and then repeat the following steps *ad infinitum*:

- for each v, each neighbour w of v, and each $p \in \Phi_Q(w) \cap \mathsf{cl}(Q, \Delta_v \cap \Delta_w)$, add p to $\Phi_Q(v)$;
- for each v and each $p \in \mathsf{cl}(Q, \Delta_v)$ that admits a Δ_v-split into queries from $\Phi_Q(v)$, add p to $\Phi_Q(v)$.

By straightforward induction on the number of steps taken before $p \in \Phi_Q(v)$ is added to $\Phi_Q(v)$ one can show that $\mathcal{J} \models p$, using Lemma 5 (1).

For the converse, we only rely on Φ_Q being a Q-annotation, not on its minimality. A *support* for p in $(T, \mathfrak{I})$ is a connected set $V \subseteq T$ such that $\bigcup_{u \in V} \mathcal{I}_u \models p$. Because each match of p relies on a finite number of nodes and edges in $\mathcal{J}$, it induces a *finite* support for p in $(T, \mathfrak{I})$. Hence, we can proceed by induction on the size of the support for p.

The base case is that $p \in \mathsf{cl}(Q, \Delta_v)$ and $\mathcal{I}_u \models p$ for some $u \in T$. By (A$_1$), $p \in \Phi_Q(u)$. Because bags containing any given set Δ form a connected subtree in T, using (A$_3$) we get that $p \in \Phi_Q(v)$.

For the inductive step, take $m > 1$ and assume that the claim holds for all v and all queries in $\mathsf{cl}(Q, \Delta_v)$ admitting a support of size at most $m - 1$. Take any $v \in T$ and $p \in \mathsf{cl}(Q, \Delta_v)$ with a support V of size m. We claim there exists a node $v_1 \in V$ such that $p \in \mathsf{cl}(Q, \Delta_{v_1})$. Let us assume that this is the case and see how to proceed from there. Because V is connected and contains at least two elements, some neighbour v_2 of v_1 belongs to V. Removing the edge between v_1 and v_2 induces a partition of V into two nonempty connected sets V_1 and V_2 such that $v_i \in V_i$ and $|V_i| < m$ for $i = 1, 2$. Let $\mathcal{I}_i = \bigcup_{u \in V_i} \mathcal{I}_u$ for $i = 1, 2$. Then $\mathcal{I}_1 \cup \mathcal{I}_2 \models p$ and Lemma 5 (2) yields a $\Delta^{\mathcal{I}_1} \cap \Delta^{\mathcal{I}_2}$-split $P_1 \cup P_2$ of p such that $\mathcal{I}_i \models P_i$ for $i = 1, 2$. It follows that V_i is a support for each query from P_i for $i = 1, 2$. Consequently, each query from $P_1 \cup P_2$ admits a support of size at most $m - 1$. Moreover, because $(T, \mathfrak{I})$ is a tree decomposition, $\Delta^{\mathcal{I}_1} \cap \Delta^{\mathcal{I}_2} \subseteq \Delta_{v_1}$, which means that $P_1 \cup P_2$ is a Δ_{v_1}-split of p. By Lemma 5 (3), it follows that $P_1 \cup P_2 \subseteq \mathsf{cl}(Q, \Delta_{v_1})$. By the induction hypothesis, $P_1 \cup P_2 \subseteq \Phi_Q(v_1)$. By (A$_2$), $p \in \Phi_Q(v_1)$. Like in the base case, it follows that $p \in \Phi_Q(v)$.

It remains to prove the claim. Because V is a connected subset of the tree T, it is a tree too. Let v_1 be that node in V from which the unique simple path to v is the shortest: either v itself, or the root of V, or one of its leaves. In order to show that $p \in \mathsf{cl}(Q, \Delta_{v_1})$ it suffices to show that $\mathsf{ind}(p) \subseteq \Delta_{v_1}$. Let $d \in \mathsf{ind}(p)$. Then $d \in \Delta_v$ and also Δ_u for some $u \in V$ because each individual name used in p must occur in some node of each support of p. Because each connected set of nodes containing v and u must also contain v_1, we conclude that $d \in \Delta_{v_1}$. This completes the proof of the claim and the whole lemma. $\square$

5. Query entailment via automata

We shall now exploit the main results of the previous two sections in order to establish optimal complexity upper bounds. To describe the approach, let us fix an ABox $\mathcal{A}$, an $\mathcal{SQ}_u$ TBox $\mathcal{T}$ in normal form, and a P2RPQ φ. By Theorem 1, φ is entailed over $\mathcal{A}$ relative to $\mathcal{T}$ iff there exists a $(\mathcal{T}, \mathcal{A})$-canonical tree decomposition of width and degree at most $\|\mathcal{A}\| \cdot 2^{\mathsf{poly}(\|\mathcal{T}\|)}$, representing an interpretation $\mathcal{J}$ such that $\mathcal{J}^* \not\models \varphi$. We will follow an automata-based approach in the sense that we will effectively construct a non-deterministic tree automaton recognizing such tree decompositions, and thus reduce query entailment to the emptiness of these tree automata. More precisely, we will construct a tree automaton $\mathfrak{A}$ that takes as input tree decompositions of width and degree as specified above and verifies the following conditions:

1. the tree decomposition in the input is $(\mathcal{T}, \mathcal{A})$-canonical, and
2. φ is not satisfied in the (transitive closure $\mathcal{J}^*$ of the) interpretation $\mathcal{J}$ represented by the tree decomposition in the input.

By Theorem 1, it is then the case that

$$\mathfrak{A} \text{ accepts some input} \quad \text{iff} \quad \mathcal{T}, \mathcal{A} \not\models \varphi,$$

which provides the promised reduction of query entailment to the non-emptiness problem of the underlying automata model.

We split the construction of $\mathfrak{A}$ into two steps. First, in Lemma 7, we construct an automaton recognizing $(\mathcal{T}, \mathcal{A})$-canonical tree decompositions; recall that, by Lemma 4, the transitive closures of the represented interpretations are models of $\mathcal{T} \cup \mathcal{A}$. Then, in Lemma 8, we construct another automaton recognizing tree decompositions of interpretations whose transitive closures do not satisfy φ. In the construction of the latter automaton, we rely on the characterization of query matches in tree decompositions established in Theorem 2: essentially, the automaton will just guess the Q-annotation in its states. Finally, we take the product of the two automata to obtain the automaton $\mathfrak{A}$ satisfying items 1 and 2 above. In order to obtain optimal bounds we have to ensure that the involved automata are of the right size and that they can be constructed in the required time. We analyze this general algorithm in terms of combined complexity in Theorem 3 and in terms of data complexity in Theorem 4 below.

While the described approach is relatively simple, the development below is complicated by a certain mismatch between the capabilities of tree automata and the nature of tree decompositions. On the one hand, the input alphabet (that is, the set of allowed node labels) of a tree automaton has to be finite. On the other hand, in any bounded-width tree decomposition of an infinite interpretation, there will be infinitely many different node labels since the tree decomposition has to encompass the whole domain of the interpretation. Hence, tree automata cannot run directly on tree decompositions. This is a recurring problem when working with tree automata that process tree decompositions and the standard solution to this problem, used for instance in [22], is to *encode* tree decompositions by reusing elements according to a certain policy, described below.

We first introduce the necessary notions for tree automata. A *k-ary Ω-labelled tree* is a pair (T, τ) where T is a tree each of whose nodes has at most k successors and $\tau : T \to \Omega$ assigns a letter from the alphabet Ω to each node. A *non-deterministic tree automaton (NTA)* over k-ary trees is a tuple $\mathfrak{A} = (Q, \Omega, I, \Lambda)$, where Q is a finite set of states, Ω is a finite alphabet, $I \subseteq Q$ is the set of initial states, and $\Lambda \subseteq \bigcup_{i \leq k}(Q \times \Omega \times Q^i)$ is a set of transitions. A *run* r on a k-ary Ω-labelled tree (T, τ) is a Q-labelled tree (T, r) such that $r(\varepsilon) \in I$ and, for every $x \in T$ with successors $x_1, \ldots, x_m$, there is a transition $(r(x), \tau(x), r(x_1), \ldots, r(x_m)) \in \Lambda$. We say that $\mathfrak{A}$ *recognizes* the set of all Ω-labelled trees that admit a run. This corresponds to the weak parity condition where all states have rank 0. For such automata language intersection can be implemented simply by using the product construction and emptiness can be tested in polynomial time [20].

We next describe the announced encoding of tree decompositions of bounded width using a finite alphabet. Intuitively, two different elements in the tree decomposition can be represented by the same element in the encoding if they never occur in the same bag nor in two neighbouring bags. Then, if some element is used in two different nodes u and v of the encoding, then it represents the same actual element in both nodes iff it is present in all nodes on the unique shortest path between u and v.

To formalize this intuition, let $N \geq |\mathsf{ind}(\mathcal{A})|$ be a bound on the bag size. We shall use a fixed domain Δ_N such that $|\Delta_N| = 2N + 2$ and $\mathsf{ind}(\mathcal{A}) \subseteq \Delta_N$. Let Σ_C, $\Sigma_\mathsf{R}^\mathsf{t}$, $\Sigma_\mathsf{R}^\mathsf{nt}$ be the sets of concept names, transitive role names, and non-transitive role names used in $\mathcal{T}$ and $\mathcal{A}$, and let $\Sigma_\mathsf{R} = \Sigma_\mathsf{R}^\mathsf{t} \cup \Sigma_\mathsf{R}^\mathsf{nt}$. The automaton's alphabet will be the set Ω_N of all pairs $(\Sigma, \mathcal{J})$ such that $\Sigma \subseteq \Sigma_\mathsf{C} \cup \Sigma_\mathsf{R}$ and $\mathcal{J}$ is a Σ-interpretation with $\Delta^\mathcal{J} \subseteq \Delta_N$. Note that such $\mathcal{J}$ can be finitely represented and that $|\Omega_N| \leq 2^{\mathsf{poly}(N, \|\mathcal{T}\|)}$. Let (T, τ) be an Ω_N-labelled tree. We use Σ_v and $\mathcal{J}_v$ to refer to the two components of the label $\tau(v)$, that is, $\tau(v) = (\Sigma_v, \mathcal{J}_v)$. Given an element $d \in \Delta_N$, we say that $v, w \in T$ are *d-connected* iff $d \in \Delta^{\mathcal{J}_u}$ for all u on the unique shortest path from v to w. In case $d \in \Delta^{\mathcal{J}_v}$, we use $[v]_d$ to denote the set of all w which are d-connected to v. We call (T, τ) *$\mathcal{A}$-consistent* if $\mathsf{ind}(\mathcal{A}) \subseteq \Delta^{\mathcal{J}_\varepsilon}$. An $\mathcal{A}$-consistent Ω_N-labelled tree (T, τ) *represents* the tree decomposition $(T, \mathfrak{I})$ where $\mathfrak{I}$ assigns to each node $v \in T$ a Σ_v-interpretation $\mathcal{I}_v$ with

$$\Delta_v = \left\{(d, [v]_d) \mid d \in \Delta^{\mathcal{J}_v}\right\},$$
$$A^{\mathcal{I}_v} = \left\{(d, [v]_d) \mid d \in A^{\mathcal{J}_v}\right\},$$
$$r^{\mathcal{I}_v} = \left\{((d, [v]_d), (e, [v]_e)) \mid (d, e) \in r^{\mathcal{J}_v}\right\},$$

for all concept names A and role names r occurring in $\mathcal{T}$ and $\mathcal{A}$; let $\mathcal{I}_{(T,\tau)} = \bigcup_{w \in T} \mathcal{I}_w$ be the underlying interpretation. Modulo a harmless bijection we can assume that $a = (a, [\varepsilon]_a)$ for all $a \in \mathsf{ind}(\mathcal{A})$. Note that $[\varepsilon]_a$ is well-defined due to $\mathcal{A}$-consistency. Conversely, given an interpretation $\mathcal{I}$ and a tree decomposition $(T, \mathfrak{I})$ of $\mathcal{I}$ of width and degree at most N, one can construct an $\mathcal{A}$-consistent Ω_N-labelled tree (T, τ) such that $\mathcal{I}_{(T,\tau)}$ is isomorphic to $\mathcal{I}$ [22]. As encoding and decoding preserves the degree, it suffices to consider N-ary trees throughout.

Lemma 7. *Given $\mathcal{A}$, $\mathcal{T}$, and $N \geq |\mathsf{ind}(\mathcal{A})|$, one can compute a nondeterministic tree automaton recognizing the set of encodings of $(\mathcal{T}, \mathcal{A})$-canonical tree decompositions of width and degree at most N, in time $O\left(2^{\mathsf{poly}(N, \|\mathcal{T}\|)}\right)$.*

Proof. Verifying that the input tree is an encoding of a $(\mathcal{T}, \mathcal{A})$-canonical tree decomposition is not difficult. Condition (D_1) is always satisfied for some $\mathcal{I}$. Condition (D_2) is guaranteed by the way we encode and decode tree decompositions. The remaining condition (D_3), as well as the conditions of $(\mathcal{T}, \mathcal{A})$-canonicity (Definition 1), can be checked by looking at the labels of each node and its neighbours. We implement this by storing the label of the parent, guessing the labels of children, checking that they satisfy all necessary conditions, and then verifying that we have guessed correctly. That is, the set of states is $\Omega_N \times \Omega_N$. For the set of initial states we take

$$\left\{\big((\Sigma, \mathcal{J}_0), (\Sigma_\mathsf{C} \cup \Sigma_\mathsf{R}, \mathcal{J})\big) \in \Omega_N \times \Omega_N \;\middle|\; \mathcal{J} \models \mathcal{A}\,,\, \Delta^{\mathcal{J}_0} \cap \Delta^{\mathcal{J}} = \emptyset\right\};$$

the interpretation $\mathcal{J}_0$ with $\Delta^{\mathcal{J}_0} \cap \Delta^{\mathcal{J}} = \emptyset$ will emulate the behaviour of the interpretation with the empty domain, which is appropriate for the non-existent parent of the root; the condition $\mathcal{J} \models \mathcal{A}$ ensures (C_1). Transitions are of the form

$$\big((\omega_0, \omega), \omega, (\omega, \omega_1), \ldots, (\omega, \omega_k)\big) \in (\Omega_N)^2 \times \Omega_N \times \underbrace{(\Omega_N)^2 \times \cdots \times (\Omega_N)^2}_{k},$$

where $k \leq N$, ω is the label of the current node, ω_0 is the stored label of the parent, and $\omega_1, \omega_2, \ldots, \omega_k$ are the guessed labels of the children. The transition above is present only if

$$(\omega_0, \omega, \omega_1, \ldots, \omega_k) = \Big(\big(\Sigma_0, \mathcal{J}_0\big),\ \big(\Sigma, \mathcal{J}\big),\ \big(\Sigma_\mathsf{C} \cup \{r_1\}, \mathcal{J}_1\big),\ \ldots,\ \big(\Sigma_\mathsf{C} \cup \{r_k\}, \mathcal{J}_k\big)\Big)$$

for some $r_1, r_2, \ldots, r_k \in \Sigma_\mathsf{R}$ such that

(D'_3) for each $i \in \{1, 2, \ldots, k\}$, the restrictions of $\mathcal{J}$ and $\mathcal{J}_i$ to signature $\Sigma \cap \Sigma_i$ and domain $\Delta^{\mathcal{J}} \cap \Delta^{\mathcal{J}_i}$ coincide;

(C'_2) $\mathcal{J}$ satisfies all CIs in $\mathcal{T}$ of the forms $\bigsqcap_i A_i \sqsubseteq \bigsqcup_j B_j$ and $A \sqsubseteq (\leqslant n\, r\, B)$ with $r \in \Sigma_\mathsf{R}^\mathsf{t}$;
(C'_3) for each concept inclusion $A \sqsubseteq (\leqslant n\, r\, B)$ in $\mathcal{T}$ with $r \in \Sigma_\mathsf{R}^{\mathsf{nt}}$ and each $d \in A^{\mathcal{I}_v} - \Delta^{\mathcal{J}_0}$ in $\mathcal{J}' = \mathcal{J} \cup \bigcup_{i=1}^{k} \mathcal{J}_i$ the element d has at most n r-successors in $B^{\mathcal{J}'}$;
(C'_4) for each concept inclusion $A \sqsubseteq (\geqslant n\, r\, B)$ in $\mathcal{T}$ and each $d \in A^{\mathcal{I}_v} - \Delta^{\mathcal{J}_0}$ there exists $i \in \{1, 2, \ldots, k\}$ such that in $\mathcal{J} \cup \mathcal{J}_i$ the element d has at least n r-successors in $B^{\mathcal{J} \cup \mathcal{J}_i}$;
(C'_5) for each $i \in \{1, 2, \ldots, k\}$ there exists $d \in \Delta^{\mathcal{J}} - \Delta^{\mathcal{J}_0}$ such that either $r_i \in \Sigma_\mathsf{R}^{\mathsf{nt}} \cup (\Sigma_\mathsf{R}^\mathsf{t} - \Sigma)$ and $\Delta^{\mathcal{J}} \cap \Delta^{\mathcal{J}_i} = \{d\}$ or $r_i \in \Sigma_\mathsf{R}^\mathsf{t} \cap \Sigma$ and $\Delta^{\mathcal{J}} \cap \Delta^{\mathcal{J}_i} = \mathsf{rel}_{r_i}^{\mathcal{J}}(d) = \mathsf{rel}_{r_i}^{\mathcal{J}_i}(d)$.

Note that the above conditions are simply translations of conditions (D_3) and (C_2)–(C_5) from the level of tree decompositions to the level of their encodings.

It remains to argue that the automaton can be constructed within the claimed time. Note first that $|\Omega_N| \leq 2^{\mathsf{poly}(N, \|\mathcal{T}\|)}$ and the elements of Ω_N can be enumerated in time $2^{\mathsf{poly}(N, \|\mathcal{T}\|)}$. Hence, the alphabet and the set of states the automaton can be computed in time $2^{\mathsf{poly}(N, \|\mathcal{T}\|)}$. Conditions (D'_3) and (C'_2)–(C'_5) above can be tested in polynomial time in the total size of the representation of $\omega_0, \omega, \omega_1, \omega_2, \ldots \omega_k$ and $\mathcal{T}$. Consequently, the transition relation can be computed in time $2^{\mathsf{poly}(N, \|\mathcal{T}\|)}$ by iterating over all possible choices of $\omega_0, \omega, \omega_1, \omega_2, \ldots \omega_k$ and filtering out the ones that violate either of these conditions. Similarly, the conditions defining the set of initial states can be verified in time polynomial in the size of the representation of the state and $\mathcal{A}$. Because $|\mathsf{ind}(\mathcal{A})| \leq N$, it follows that the set of initial states can be computed in time $2^{\mathsf{poly}(N, \|\mathcal{T}\|)}$. Thus, the whole automaton can be constructed in time $2^{\mathsf{poly}(N, \|\mathcal{T}\|)}$. □

Lemma 8. *Given a P2RPQ φ without individual names and a positive integer N, one can compute in time $2^{\mathsf{poly}(\|\mathcal{T}\|, N^{\|\varphi\|}, \|\varphi\|^{\|\varphi\|})}$ a nondeterministic tree automaton recognizing the set of encodings of tree decompositions of width and degree at most N of interpretations $\mathcal{J}$ such that $\mathcal{J}^* \not\models \varphi$.*

Proof. In order to be able to evaluate φ directly over $\mathcal{J}$, rather than over $\mathcal{J}^*$, for each transitive role name r, we replace each occurrence of r in φ with $r \circ r^*$, and each occurrence of r^- with $r^- \circ (r^-)^*$. This ensures that $\mathcal{J}^* \models \varphi$ iff $\mathcal{J} \models \varphi$, without affecting the semantics of φ over $\mathcal{J}^*$.

The next step is to move to the NFA-based representation. Path expressions in φ are regular expressions over the alphabet comprising all r, r^-, and $A?$ used in φ. It is well-known that every regular expression can be converted in polynomial time to an equivalent NFA with linearly many states [23]. Without loss of generality, one can assume that the automaton has only one initial and one final state. Hence, we can replace each atom $\mathcal{E}(t, t')$ with $\mathfrak{B}(t, t')$ where $\mathfrak{B}$ is an NFA equivalent to $\mathcal{E}$ (viewed as a regular expression).

Finally, it is well-known that φ can be rewritten (in exponential time) as a disjunction of at most $\|\varphi\|^{\|\varphi\|}$ C2RPQs of size at most $\|\varphi\|$. Let $q_1 \vee q_2 \vee \cdots \vee q_\ell$ be such an equivalent formulation of φ and let $Q = \{q_1, q_2, \ldots, q_\ell\}$. By Theorem 2, if $(T, \mathfrak{I})$ is a tree decomposition of some $\mathcal{J}$ then $\mathcal{J} \not\models \varphi$ iff

$$(T, \mathfrak{I}) \text{ admits a } Q\text{-annotation } \Phi \text{ such that } Q \cap \bigcup_{v \in T} \Phi(v) = \emptyset. \tag{7}$$

Thus, it suffices to construct an automaton recognizing encodings of tree decompositions $(T, \mathfrak{I})$ satisfying (7).

Intuitively, the automaton will guess a Q-annotation of the encoded tree decomposition and verify that it avoids Q as required in (7). Recall that all bags of the tree decomposition are encoded using elements from a fixed finite domain Δ_N: the elements of Δ_N are reused to represent different elements in different bags, the rule being that $d \in \Delta_N$ represents the same element in two bags iff d is present in each bag on the unique simple path between these bags. Consequently, it suffices to consider Δ_N-splits of queries from Q. Because these queries use no individual names, their Δ_N-splits use only individual names from Δ_N. In order to check correctness of the guessed annotation, the automaton needs access to the domains of children of the current node: we let the automaton guess them and verify them later on. Thus, the states of the automaton are pairs (Δ, Φ) such that $\Delta \subseteq \Delta_N$ and $\Phi \subseteq \mathsf{cl}(Q, \Delta) - Q$; all states are initial and final. Transitions are of the form

$$\big((\Delta_0, \Phi_0), (\Sigma, \mathcal{J}), (\Delta_1, \Phi_1), (\Delta_2, \Phi_2), \ldots, (\Delta_k, \Phi_k)\big),$$

where $\Delta_0 = \Delta^{\mathcal{J}}$, $\Phi_0 \subseteq \mathsf{cl}(Q, \Delta_0)$ and

(A_1') $\big\{p \in \mathsf{cl}(Q, \Delta_0) \mid \mathcal{J} \models p\big\} \subseteq \Phi_0$;
(A_2') if some Δ_0-split of some $p \in \mathsf{cl}(Q, \Delta_0)$ is a subset of Φ_0, then $p \in \Phi_0$;
(A_3') for each $i \in \{1, 2, \ldots, k\}$,

$$\Phi_0 \cap \mathsf{cl}(Q, \Delta_0 \cap \Delta_i) = \Phi_i \cap \mathsf{cl}(Q, \Delta_0 \cap \Delta_i).$$

From each run over the input tree (T, τ) we can extract a Q-annotation of the encoded tree decomposition $(T, \mathfrak{I})$: in each node $v \in T$ we take the second component $\Phi_0 \subseteq \mathsf{cl}(Q, \Delta_0)$ of the corresponding state (Δ_0, Φ_0) and replace constants from $\Delta_0 \subseteq \Delta_N$ by the corresponding constants from Δ_v. Conditions (A_1'), (A_2'), (A_3') above are direct translations of (A_1), (A_2), (A_3) to the encodings. Because neighbouring bags agree over the encoding of shared elements, it is easy to see that the extracted labelling of T is a correct Q-annotation of $(T, \mathfrak{I})$. Similarly, we can show that each Q-annotation of a tree decomposition $(T, \mathfrak{I})$ can be turned into an accepting run over each encoding of $(T, \mathfrak{I})$.

Recall that we have $|\Omega_N| \leq 2^{\mathsf{poly}(N, \|\mathcal{T}\|)}$ and that queries in Q have size at most $\|\varphi\|$, the automata used to represent path expressions in these queries have at most $m = O(\|\varphi\|)$ states, and $|Q| \leq \|\varphi\|^{\|\varphi\|}$. Using Lemma 5 (4), we get

$$|\mathsf{cl}(Q, \Delta_N)| \leq |Q| \cdot \big(6 \cdot m^2 \cdot (2N)^2\big)^{\|\varphi\|} \leq \mathsf{poly}\big(N^{\|\varphi\|}, \|\varphi\|^{\|\varphi\|}\big).$$

Consequently, the number of states in the constructed tree automaton is bounded by

$$2^{|\Delta_N|} \cdot 2^{|\mathsf{cl}(Q, \Delta_N)|} \leq 2^{\mathsf{poly}\left(N^{\|\varphi\|}, \|\varphi\|^{\|\varphi\|}\right)}$$

and the number of transitions is bounded by

$$\sum_{k=0}^{N} |\Omega_N| \cdot \left(2^{|\Delta_N|} \cdot 2^{|\mathsf{cl}(Q, \Delta_N)|}\right)^{k+1} \leq 2^{\mathsf{poly}\left(\|\mathcal{T}\|, N^{\|\varphi\|}, \|\varphi\|^{\|\varphi\|}\right)},$$

where k stands for the possible numbers of children of the current node. Condition (A_1') can be tested in time $\mathsf{poly}(|\mathsf{cl}(Q, \Delta_N)|, N^{\|\varphi\|}, \|\mathcal{T}\|)$ because

$$\mathsf{cl}(Q, \Delta_0) \subseteq \mathsf{cl}(Q, \Delta_N),$$

the size of the domain of $\mathcal{J}$ is linear in N, and the size of queries in $\mathsf{cl}(Q, \Delta_0)$ is linear in $\|\varphi\|$. Conditions (A_2') and (A_3') can be tested in time polynomial in the total size of the representations of the objects they mention. It follows that the whole automaton can be constructed in time $2^{\mathsf{poly}(\|\mathcal{T}\|, N^{\|\varphi\|}, \|\varphi\|^{\|\varphi\|})}$. $\square$

The main results are now obtained by putting the pieces together. We begin with a simple argument establishing the combined complexity.

Theorem 3. *Entailment of P2RPQs relative to $\mathcal{SQ}_u$ TBoxes is* 2ExpTime-*complete.*

Proof. Let $\mathcal{A}$, $\mathcal{T}$, and φ be the input ABox, $\mathcal{SQ}_u$ TBox, and P2RPQ. As explained in Section 2.3, we can assume that $\mathcal{T}$ is in normal form and φ contains no individual names. Let $N \le \|\mathcal{A}\| \cdot 2^{\mathsf{poly}(\|\mathcal{T}\|)}$ be a constant bounding the width and degree of tree decompositions, as guaranteed by Theorem 1; it can be easily computed from $\mathcal{T}$ and $\mathcal{A}$. We will say that a tree decomposition is N-bounded if its width and degree is at most N. By Lemma 7, we can construct in time $O(2^{\mathsf{poly}(N,\|\mathcal{T}\|)})$ an automaton $\mathfrak{A}_{\mathcal{T},\mathcal{A}}$ recognizing $(\mathcal{T},\mathcal{A})$-canonical N-bounded tree decompositions. By Lemma 8, we can construct in time $2^{\mathsf{poly}(\|\mathcal{T}\|,N^{\|\varphi\|},\|\varphi\|^{\|\varphi\|})}$ an automaton $\mathfrak{A}_{\neg\varphi}$ recognizing N-bounded tree decomposition of interpretations whose transitive closures falsify φ. The product $\mathfrak{A}$ of these automata recognizes $(\mathcal{T},\mathcal{A})$-canonical N-bounded tree decompositions of interpretations whose transitive closures falsify φ. From Theorem 1 it follows that $\mathcal{T},\mathcal{A} \models \varphi$ iff $\mathfrak{A}$ does not accept anything. We construct the product automaton and test its emptiness in time polynomial in the size of $\mathfrak{A}_{\mathcal{T},\mathcal{A}}$ and $\mathfrak{A}_{\neg\varphi}$. Overall, this gives a decision procedure whose combined complexity is doubly exponential. The matching lower bound is inherited from positive existential query answering in the sublogic $\mathcal{ALC}$ [19]. □

The algorithm described in the proof of Theorem 3 explicitly constructs automata that are exponential in the size of the ABox. In order to establish a tight upper bound for data complexity, we will show that one can avoid constructing the whole automaton; the combined complexity of the resulting algorithm will be still doubly exponential.

Theorem 4. *Entailment of P2RPQs relative to $\mathcal{SQ}_u$ TBoxes is* coNP*-complete in terms of data complexity.*

Proof. As the lower bound carries over from the entailment of instance queries in $\mathcal{ALC}$ [18], we only need to prove the upper bound. Consider an ABox $\mathcal{A}$, an $\mathcal{SQ}_u$ TBox $\mathcal{T}$ in normal form, and a P2RPQ φ without individual names. We shall rely on the additional claim of Theorem 1. Let $M \le 2^{\mathsf{poly}(\|\mathcal{T}\|)}$ be a constant bounding the size and the degree of non-root bags, as in Theorem 1, and let it be large enough to ensure that the size and the degree of the root bag is at most $N = \|\mathcal{A}\| \cdot M$. Again, M is computable from $\mathcal{T}$ and $\mathcal{A}$. We shall reuse the automata $\mathfrak{A}_{\mathcal{T},\mathcal{A}}$ and $\mathfrak{A}_{\neg\varphi}$ constructed in Lemmas 7 and 8. The idea is that we universally guess initial levels of the input tree and the corresponding fragment of a run of the product automaton of $\mathfrak{A}_{\mathcal{T},\mathcal{A}}$ and $\mathfrak{A}_{\neg\varphi}$ and then check that it cannot be completed to a tree accepted by the product automaton. The latter check will only explore limited fragments of the product automaton, computable in time independent of $\mathcal{A}$. Notice that in each child of the root the bag size and the degree is already independent of $\mathcal{A}$, but the state still stores full information about the root bag; this information is only forgotten when the next transition is taken. This is why we guess three initial levels, not just two.

We begin by guessing a tree of depth 2 in which the root has at most $\|\mathcal{A}\| \cdot M$ children, and each of those has at most M children. For each guessed node v, we guess a label $(\Sigma_v, \mathcal{I}_v)$ such that $\Sigma_v \subseteq \Sigma_{\mathsf{C}} \cup \Sigma_{\mathsf{R}}$ and $\mathcal{I}_v$ is a Σ_v-interpretation with domain $\Delta_v \subseteq \Delta_N$. Moreover, $|\Delta_v| \le M$ for all nodes v except the root. For each guessed node v we also guess the corresponding state $\big(((\Sigma'_v, \mathcal{I}'_v), (\Sigma_v, \mathcal{I}_v)), (\Delta_v, \Phi_v)\big)$ of the product automaton, where $\mathcal{I}'_v$ is a Σ'_v-interpretation with domain $\Delta'_v \subseteq \Delta_N$ for some $\Sigma'_v \subseteq \Sigma_{\mathsf{C}} \cup \Sigma_{\mathsf{R}}$, and $\Phi_v \subseteq \mathsf{cl}(\Delta_v, Q)$ with Q the set of C2RPQs obtained by rewriting the input query φ as a disjunction of C2RPQs. As we have seen in the proof of Lemma 8, $|\Phi_v| \le |\mathsf{cl}(\Delta_v, Q)| \le \mathsf{poly}\big(|\Delta_v|^{\|\varphi\|}, \|\varphi\|^{\|\varphi\|}\big)$ and each query from $\Phi_v \subseteq \mathsf{cl}(\Delta_v, Q)$ has size $O(\|\varphi\|)$. Moreover, $\|\mathcal{I}_v\|, \|\mathcal{I}'_v\| \le \mathsf{poly}(N, \|\mathcal{T}\|)$. Because we consider $\mathcal{T}$ and φ fixed, it follows that we can store the guessed fragment of the input tree and the corresponding run in space polynomial in $\|\mathcal{A}\|$. As we have seen in the proofs of Lemmas 7 and 8, we can also check in polynomial time that the state in the root is initial and that the transition relation is respected. It remains to check that the guessed fragments of the input tree and the corresponding run cannot be extended to a full tree and a full accepting run. This is equivalent to checking that for some grandchild v of the root, the product automaton does not accept anything from the state guessed for v. We claim that this test can be performed in time independent of $\|\mathcal{A}\|$ for each grandchild v of the root. As the number of such nodes v is polynomial in $\|\mathcal{A}\|$, this gives a coNP decision procedure (in terms of data complexity).

To see why the claim holds, let us fix a grandchild v of the root. Recall that we only need to consider tree decompositions with the size of non-root bags bounded by M. Whether the automaton accepts depends only on the tree decomposition, not on the encoding. Consequently, we can assume that all bags in the subtree rooted at v are encoded using a fixed domain Δ^v_M such that $\Delta_v \subseteq \Delta^v_M \subseteq \Delta_N$ and $|\Delta^v_M| \le 2M$. Let $\mathfrak{A}^v_{\mathcal{T},\mathcal{A}}$ and $\mathfrak{A}^v_{\neg\varphi}$ be the automata defined just like $\mathfrak{A}_{\mathcal{T},\mathcal{A}}$ and $\mathfrak{A}_{\neg\varphi}$, but with Δ_N replaced by Δ^v_M, and the initial states set to $\big((\Sigma'_v, \mathcal{I}'_v), (\Sigma_v, \mathcal{I}_v)\big)$ and (Δ_v, Φ_v), respectively. These automata do not depend on $\mathcal{A}$ any more: apart from $\mathcal{T}$ and φ, they depend only on $\big((\Sigma'_v, \mathcal{I}'_v), (\Sigma_v, \mathcal{I}_v)\big)$ and (Δ_v, Φ_v). Because $|\Delta^v| \le |\Delta^v_M| \le M$ and $\|\mathcal{I}_v\|, \|\mathcal{I}'_v\| \le \mathsf{poly}(M, \|\mathcal{T}\|)$, the product automaton of $\mathfrak{A}^v_{\mathcal{T},\mathcal{A}}$ and $\mathfrak{A}^v_{\neg\varphi}$ can be computed and tested for emptiness in time constant with respect to $\|\mathcal{A}\|$. □

6. Related work

We begin with related work on fragments of FO with counting and transitivity. Then we move to description logics with counting and transitivity. Finally, we discuss existing work on answering navigational queries, such as regular path queries or extensions thereof, mediated by description logic ontologies.

6.1. Related work on fragments of FO with counting and transitivity

In general, extending expressive decidable fragments of FO with counting and transitivity has turned out to be challenging because their combination easily leads to undecidability. There has been some work on the extension of decidable first-order logic fragments, such as the guarded fragment, with transitivity and counting, see e.g., [24,25]. However, to ensure decidability of the satisfiability problem the interaction of counting and transitivity is severely restricted [24]. In the context of existential rules, several efforts have been recently made to design languages with decidable QA supporting transitivity [26–28]. However, extending decidable existential rules languages with counting (even without transitive relations) easily leads to undecidability [29]. Similarly, extending the unary negation fragment of FO with counting leads to undecidability [30,31]. In this sense our positive results are an important step to close the distance to the undecidability frontier. In fact, besides extensions based on $\mathcal{SQ}_u$, we are not aware of more expressive decidable logics supporting counting and transitivity.

6.2. Related work on DLs with number restrictions on transitive roles

The initial investigations on $\mathcal{SQ}_u$ and extensions thereof concentrated on the limits of decidability of the satisfiability problem. Decision procedures for satisfiability in $\mathcal{SQ}_u$ and its extension $\mathcal{SOQ}_u$ with nominals were provided in [32] and [33], respectively; in both cases, the procedure gives a non-elementary upper bound. Shortly thereafter, concept satisfiability in $\mathcal{SQ}_u$ restricted to a single transitive role name was shown to be NExpTime-complete [34]; in fact, the result was shown for *graded modal logics over transitive frames*, which is a notational variant. The techniques for proving the NExpTime-upper bound were then adapted to prove also tight complexity bounds for the more general problem of TBox satisfiability (*global consequence*, in modal logic parlance) in the presence of an arbitrary number of transitive and non-transitive roles, and even in the presence of nominals: both for $\mathcal{SQ}_u$ and $\mathcal{SOQ}_u$, TBox satisfiability is NExpTime-complete [35]. Further, it was shown that even modest extensions of $\mathcal{SQ}_u$ such as with role inclusions or inverse roles result in logics with an undecidable TBox satisfiability problem [32], which implies that query answering is undecidable as well. In fact, already supporting number restrictions on transitive roles in the lightweight *DL-Lite*$_{core}^{\mathcal{HN}}$ [36] leads to undecidability [35]. Remarkably, it has been observed that sometimes decidability of TBox satisfiability can be regained by disallowing the application of at-most restrictions to inverse transitive roles (but still allowing the application of at-least restrictions) [37,38].

Recall that $\mathcal{SQ}_u$ does not enjoy the tree model property, which has ramifications here as well. Indeed, it has been shown that when transitive roles are interpreted as the transitive closure of the successor relation induced by a tree, satisfiability in $\mathcal{SQ}_u$ is decidable in NExpTime even if role inclusions are allowed [39]. This is in marked contrast with the mentioned undecidability result for $\mathcal{SQ}_u$ with role inclusions [32].

6.3. Related approaches to navigational queries

Possibly closest to our approach is [8], where an automata-based approach consisting of Steps 1–3 described in Section 1.1 is followed, too. There, the query automaton $\mathfrak{A}_{\neg\varphi}$ is obtained from an initial automaton via a projection and a subsequent complementation operation. This approach to constructing $\mathfrak{A}_{\neg\varphi}$ is applicable here as well, but does not lead to optimal complexity, due to the higher treewidth. In [14], a *query rewriting approach* is followed. While this is in principle orthogonal to ours, it faces the same challenge, non-locality of path queries, and thus still needs a form of query decompositions or splits. Sometimes, the query can also be rewritten or incorporated into the ontology. For example, if the considered DL allows for the *regular role inclusion constructor*, usually abbreviated with $\mathcal{R}$, then entailment of regular path queries can be reduced to entailment of conjunctive queries [13,40]. Finally, a query decomposition which is in spirit close to ours was recently developed for the extension of the unary negation fragment of first-order logic [31] with regular path expressions [16]; this extension captures answering (unions of) C2RPQs mediated by $\mathcal{S}$ ontologies, where $\mathcal{S}$ denotes the extension of $\mathcal{ALC}$ with transitive roles.

A certain form of non-locality is also present in conjunctive queries with transitive atoms. Such queries are usually considered in ontology-mediated query answering when the ontology language subsumes $\mathcal{S}$. It is known that in some cases entailment of CQs with transitive relations is indeed more complex than without. For instance, entailment of CQs with transitive atoms relative to $\mathcal{SHQ}$ ontologies is 2ExpTime-complete [6,7], while it is ExpTime-complete [41] without transitive atoms. In fact, this increase in complexity is already witnessed in $\mathcal{S}$ itself, for which entailment of CQs with transitive atoms is coNExpTime-hard [7].

7. Conclusions

We have studied the problem of answering positive two-way regular path queries (P2RPQs) relative to $\mathcal{SQ}_u$, that is, unrestricted $\mathcal{SQ}$, ontologies. Our main results are decidability and tight complexity bounds, both in data complexity and in combined complexity.

These results are both of practical and theoretical interest. From a practical point of view, our complexity results meet the application demands and open up the possibility to include a profile based on $\mathcal{SQ}_u$ 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 a theoretical perspective, some of the techniques we introduced here are of broader interest. Specifically, the annotations and the query automaton are developed independently of the ontology and can be applied for any ontology language enjoying the tree-like countermodel property. For instance, we recently generalized the tree-like countermodel property to the extensions of $\mathcal{SQ}_u$ with nominals ($\mathcal{SOQ}_u$) and with *controlled inverses* ($\mathcal{SIQ}_u^-$), where at-most restrictions are only applicable to role names, but not to inverses [38]. For these logics, recognizing tree-like models of $\mathcal{T}$ and $\mathcal{A}$ requires a different automaton $\mathfrak{A}_{\mathcal{T},\mathcal{A}}$, but the automaton $\mathfrak{A}_{\neg\varphi}$ constructed in Section 5 can be reused, yielding the same complexity upper bounds.

There are various directions of future work. With respect to the query language, we believe that our techniques can be lifted to *nested* P2RPQs, which extend P2RPQs with a form of *tests*. The feature of nesting has been introduced to navigational queries in the context of SPARQL [42], but has also been studied in the ontology-mediated setting to some extent [43,44]. It would also be interesting to look at more restricted query languages. For example, for CQs there is a gap in combined complexity between our 2ExpTime upper bound and the best known lower bound, which is coNExpTime and holds already for $\mathcal{S}$ ontologies [7].

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgements

Filip Murlak was supported by the NCN grant 2018/30/E/ST6/00042.

References

[1] M. Benedikt, B.C. Grau, E.V. Kostylev, Logical foundations of information disclosure in ontology-based data integration, Artif. Intell. 262 (2018) 52–95.
[2] G. Xiao, D. Calvanese, R. Kontchakov, D. Lembo, A. Poggi, R. Rosati, M. Zakharyaschev, Ontology-based data access: A survey, in: J. Lang (Ed.), Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI), ijcai.org, 2018, pp. 5511–5519.
[3] F. Baader, I. Horrocks, C. Lutz, U. Sattler, An Introduction to Description Logic, Cambridge University Press, 2017.
[4] W3C OWL Working Group, OWL 2 Web Ontology Language, Available at http://www.w3.org/TR/owl2-overview/, 2009.
[5] B. Glimm, I. Horrocks, U. Sattler, Unions of conjunctive queries in SHOQ, in: G. Brewka, J. Lang (Eds.), Proceedings of the 11th International Conference on Principles of Knowledge Representation and Reasoning (KR), AAAI Press, 2008, pp. 252–262.
[6] B. Glimm, C. Lutz, I. Horrocks, U. Sattler, Conjunctive query answering for the description logic SHIQ, J. Artif. Intell. Res. 31 (2008) 157–204.
[7] T. Eiter, C. Lutz, M. Ortiz, M. Simkus, Query answering in description logics with transitive roles, in: C. Boutilier (Ed.), Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI), 2009, pp. 759–764.
[8] D. Calvanese, T. Eiter, M. Ortiz, Answering regular path queries in expressive description logics via alternating tree-automata, Inf. Comput. 237 (2014) 12–55.
[9] W3C, OWL Web Ontology Language: Use Cases and Requirements, Available at https://www.w3.org/TR/webont-req/, 2004.
[10] A.K. Chandra, P.M. Merlin, Optimal implementation of conjunctive queries in relational data bases, in: J.E. Hopcroft, E.P. Friedman, M.A. Harrison (Eds.), Proceedings of the 9th Annual ACM Symposium on Theory of Computing (STOC), ACM, 1977, pp. 77–90.
[11] P. Barceló, Querying graph databases, in: R. Hull, W. Fan (Eds.), Proceedings of the 32nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS), ACM, 2013, pp. 175–188.
[12] D. Calvanese, G.D. Giacomo, M. Lenzerini, M.Y. Vardi, Containment of conjunctive regular path queries with inverse, in: A.G. Cohn, F. Giunchiglia, B. Selman (Eds.), Proceedings of the 7th International Conference on Principles of Knowledge Representation and Reasoning(KR), Morgan Kaufmann, 2000, pp. 176–185.
[13] G. Stefanoni, B. Motik, M. Krötzsch, S. Rudolph, The complexity of answering conjunctive and navigational queries over OWL 2 EL knowledge bases, J. Artif. Intell. Res. 51 (2014) 645–705.
[14] M. Bienvenu, M. Ortiz, M. Simkus, Regular path queries in lightweight description logics: Complexity and algorithms, J. Artif. Intell. Res. 53 (2015) 315–374.
[15] J. Baget, M. Bienvenu, M. Mugnier, M. Thomazo, Answering conjunctive regular path queries over guarded existential rules, in: C. Sierra (Ed.), Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI), ijcai.org, 2017, pp. 793–799.
[16] J.C. Jung, C. Lutz, M. Martel, T. Schneider, Querying the unary negation fragment with regular path expressions, in: B. Kimelfeld, Y. Amsterdamer (Eds.), Proceedings of the 21st International Conference on Database Theory (ICDT), in: LIPIcs, vol. 98, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018, pp. 15:1–15:18.
[17] M.Y. Vardi, The complexity of relational query languages (extended abstract), in: H.R. Lewis, B.B. Simons, W.A. Burkhard, L.H. Landweber (Eds.), Proceedings of the 14th Annual ACM Symposium on Theory of Computing (STOC), ACM, 1982, pp. 137–146.
[18] A. Schaerf, On the complexity of the instance checking problem in concept languages with existential quantification, J. Intell. Inf. Syst. 2 (3) (1993) 265–278.
[19] M. Ortiz, M. Simkus, Revisiting the hardness of query answering in expressive description logics, in: R. Kontchakov, M. Mugnier (Eds.), Proceedings of the 8th International Conference on Web Reasoning and Rule Systems (RR), in: Lecture Notes in Computer Science, vol. 8741, Springer, 2014, pp. 216–223.
[20] J. Neumann, A. Szepietowski, I. Walukiewicz, Complexity of weak acceptance conditions in tree automata, Inf. Process. Lett. 84 (4) (2002) 181–187.
[21] S. Tessaris, Questions and answers: reasoning and querying in description logic, Ph.D. thesis, University of Manchester, 2001.
[22] E. Grädel, I. Walukiewicz, Guarded fixed point logic, in: Proceedings of the 14th Annual IEEE Symposium on Logic in Computer Science (LICS), IEEE Computer Society, 1999, pp. 45–54.

[23] M. Fürer, The complexity of the inequivalence problem for regular expressions with intersection, in: J.W. de Bakker, J. van Leeuwen (Eds.), Proceedings of the 7th Colloquium on Automata, Languages and Programming (ICALP), in: Lecture Notes in Computer Science, vol. 85, Springer, 1980, pp. 234–245.
[24] L. Tendera, Counting in the two variable guarded logic with transitivity, in: V. Diekert, B. Durand (Eds.), Proceedings of the 22nd Annual Symposium on Theoretical Aspects of Computer Science (STACS), in: Lecture Notes in Computer Science, vol. 3404, Springer, 2005, pp. 83–96.
[25] I. Pratt-Hartmann, The two-variable fragment with counting and equivalence, Math. Log. Q. 61 (6) (2015) 474–515.
[26] G. Gottlob, A. Pieris, L. Tendera, Querying the guarded fragment with transitivity, in: F.V. Fomin, R. Freivalds, M.Z. Kwiatkowska, D. Peleg (Eds.), Proceedings of the 40th International Colloquium on Automata, Languages, and Programming (ICALP), in: Lecture Notes in Computer Science, vol. 7966, Springer, 2013, pp. 287–298.
[27] J. Baget, M. Bienvenu, M. Mugnier, S. Rocher, Combining existential rules and transitivity: Next steps, in: Q. Yang, M.J. Wooldridge (Eds.), Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI), AAAI Press, 2015, pp. 2720–2726.
[28] A. Amarilli, M. Benedikt, P. Bourhis, M. Vanden Boom, Query answering with transitive and linear-ordered data, J. Artif. Intell. Res. 63 (2018) 191–264.
[29] A. Calì, D. Lembo, R. Rosati, On the decidability and complexity of query answering over inconsistent and incomplete databases, in: F. Neven, C. Beeri, T. Milo (Eds.), Proceedings of the 22nd ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS), ACM, 2003, pp. 260–271.
[30] D. Danielski, E. Kieronski, Finite satisfiability of unary negation fragment with transitivity, in: P. Rossmanith, P. Heggernes, J. Katoen (Eds.), Proceedings of the 44th International Symposium on Mathematical Foundations of Computer Science (MFCS), in: LIPIcs, vol. 138, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019, pp. 17:1–17:15.
[31] L. Segoufin, B. ten Cate, Unary negation, Log. Methods Comput. Sci. 9 (3) (2013).
[32] Y. Kazakov, U. Sattler, E. Zolin, How many legs do I have? non-simple roles in number restrictions revisited, in: N. Dershowitz, A. Voronkov (Eds.), Proceedings of 14th International Conference on Logic for Programming, AI, and Reasoning (LPAR), in: Lecture Notes in Computer Science, vol. 4790, Springer, 2007, pp. 303–317.
[33] M. Kaminski, G. Smolka, Terminating tableaux for $\mathcal{SOQ}$ with number restrictions on transitive roles, in: Proceedings of 6th International Conference on Theoretical Computer Science (IFIP TCS), in: IFIP Advances in Information and Communication Technology, vol. 323, Springer, 2010, pp. 213–228.
[34] Y. Kazakov, I. Pratt-Hartmann, A note on the complexity of the satisfiability problem for graded modal logics, in: Proceedings of the 24th Annual IEEE Symposium on Logic in Computer Science (LICS), IEEE Computer Society, 2009, pp. 407–416.
[35] V. Gutiérrez-Basulto, Y.A. Ibáñez-García, J.C. Jung, Number restrictions on transitive roles in description logics with nominals, in: S.P. Singh, S. Markovitch (Eds.), Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI), AAAI Press, 2017, pp. 1121–1127.
[36] A. Artale, D. Calvanese, R. Kontchakov, M. Zakharyaschev, The DL-Lite family and relations, J. Artif. Intell. Res. 36 (2009) 1–69.
[37] B. Bednarczyk, E. Kieronski, P. Witkowski, Completing the picture: Complexity of graded modal logics with converse, Theory Pract. Log. Program. 21 (4) (2021) 493–520.
[38] T. Gogacz, V. Gutiérrez-Basulto, Y. Ibáñez-García, J.C. Jung, F. Murlak, On finite and unrestricted query entailment beyond $\mathcal{SQ}$ with number restrictions on transitive roles, in: S. Kraus (Ed.), Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI), 2019, pp. 1719–1725.
[39] L. Schröder, D. Pattinson, How many toes do I have? parthood and number restrictions in description logics, in: G. Brewka, J. Lang (Eds.), Proceedings of the 11th International Conference on Principles of Knowledge Representation and Reasoning (KR), AAAI Press, 2008, pp. 307–317.
[40] B. Bednarczyk, S. Rudolph, Worst-case optimal querying of very expressive description logics with path expressions and succinct counting, in: S. Kraus (Ed.), Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI), ijcai.org, 2019, pp. 1530–1536.
[41] C. Lutz, The complexity of conjunctive query answering in expressive description logics, in: A. Armando, P. Baumgartner, G. Dowek (Eds.), Proceedings of 4th International Joint Conference on Automated Reasoning (IJCAR), in: Lecture Notes in Computer Science, vol. 5195, Springer, 2008, pp. 179–193.
[42] J. Pérez, M. Arenas, C. Gutiérrez, nSPARQL: A navigational language for RDF, J. Web Semant. 8 (4) (2010) 255–270.
[43] M. Bienvenu, D. Calvanese, M. Ortiz, M. Simkus, Nested regular path queries in description logics, in: C. Baral, G.D. Giacomo, T. Eiter (Eds.), Proceedings of 14th International Conference on Principles of Knowledge Representation and Reasoning (KR), AAAI Press, 2014.
[44] P. Bourhis, M. Krötzsch, S. Rudolph, How to best nest regular path queries, in: M. Bienvenu, M. Ortiz, R. Rosati, M. Simkus (Eds.), Proceedings of the 27th International Workshop on Description Logics, CEUR Workshop Proceedings, 2014, pp. 404–415.