Containment of Regular Path Queries under Description Logic Constraints*

Diego Calvanese
KRDB Research Centre
Free University of Bozen-Bolzano
Piazza Domenicani 3, Bolzano, Italy
calvanese@inf.unibz.it

Magdalena Ortiz and **Mantas Šimkus**
Institute of Information Systems
Vienna University of Technology
Favoritenstraße 9-11, Vienna, Austria
ortiz@kr.tuwien.ac.at, simkus@dbai.tuwien.ac.at

Abstract

Query containment has been studied extensively in KR and databases, for different kinds of query languages and domain constraints. We address the longstanding open problem of containment under expressive description logic (DL) constraints for two-way regular path queries (2RPQs) and their conjunctions, which generalize conjunctive queries with the ability to express regular navigation. We show that, surprisingly, functionality constraints alone make containment of 2RPQs already EXPTIME-hard. By employing automata-theoretic techniques, we also provide a matching upper bound that extends to very expressive DL constraints. For conjunctive 2RPQs we prove a further exponential jump in complexity, and provide again a matching upper bound for expressive DLs. Our techniques provide also a solution to the problem of query entailment over DL knowledge bases in which individuals in the ABox may be related through regular role-paths.

1 Introduction

Query containment [Abiteboul *et al.*, 1995], called *subsumption* in AI [Baader *et al.*, 2007], is a fundamental task in contexts like query optimization, information integration, knowledge base verification, and management of semistructured and XML data, cf. [Levy and Rousset, 1998; ten Cate and Lutz, 2009].

The complexity of query containment has been studied extensively, starting from the classical NP-completeness result for plain conjunctive queries (CQs) [Chandra and Merlin, 1977]. For recursive Datalog queries the problem is undecidable [Shmueli, 1993] but it remains decidable when one of the two queries is non-recursive [Sagiv, 1988; Chaudhuri and Vardi, 1997]. Containment has also been addressed for various forms of recursive queries over *graph databases*, i.e., databases consisting of binary relations only. In this setting, which is receiving increased attention [Barceló *et al.*, 2010], the basic querying mechanism is *(two-way) regular path queries (2RPQs)* [Calvanese *et al.*, 2003]. These queries ask for all pairs of objects connected by a path conforming to a regular language over the binary relations, and thus support a restricted form of recursion. 2RPQs have been included in the new version of the SPARQL language for querying ontologies, which is currently under standarisation at W3C.[1]

By employing techniques based on two-way automata over finite words, containment has been shown PSPACE-complete for 2RPQs [Calvanese *et al.*, 2003], and EXPSPACE-complete for conjunctive 2RPQs (C2RPQs), the expressive variant of queries considered here that extends plain CQs by allowing each atom to be a 2RPQ [Calvanese *et al.*, 2000].

We address here the problem of *query containment under constraints*, which amounts to checking whether containment between the answers to two queries holds for all structures (i.e., databases) satisfying a given set of constraints. This problem has been considered, e.g., under various forms of database constraints [Johnson and Klug, 1984], and for XML queries under constraints expressed as DTDs [ten Cate and Lutz, 2009]. We consider constraints expressed in terms of expressive Description Logics (DLs) [Baader *et al.*, 2007], which allow one to capture by means of a TBox complex conditions that hold in a domain of interest.

For plain CQs over TBoxes, containment $Q_1 \subseteq Q_2$ of two queries is equivalent to checking entailment of Q_2 in a knowledge base (KB) whose extensional component (ABox) is directly obtained from Q_1. In general, entailment/containment of CQs in expressive DLs is exponentially harder than inference over plain KBs, e.g., 2EXPTIME-hard for extensions of $\mathcal{ALC}$ with inverse roles [Lutz, 2008], or with role hierarchies and transitivity [Eiter *et al.*, 2009]. Tight upper bounds for entailment/containment of CQs under expressive DL constraints have been obtained using a variety of techniques [Glimm *et al.*, 2008; Lutz, 2008; Eiter *et al.*, 2009], and extended to entailment of C2RPQs [Calvanese *et al.*, 2009]. However, these upper bounds do not easily extend to containment in the presence of regular path constructs, as a 2RPQ or C2RPQ on the left of $Q_1 \subseteq Q_2$, unlike a CQ, cannot be written as an ABox. Using nominals to incorporate Q_1 into the TBox, tight

*This work was partially supported by the Austrian Science Fund (FWF) grant P20840, the Vienna Science and Technology Fund (WWTF) project ICT08-032, and by the EU under the ICT Collaborative Project ACSI (Artifact-Centric Service Interoperation), grant agreement n. FP7-257593.

[1] http://www.w3.org/TR/sparql11-query/

upper bounds for containment of (an extension of) C2RPQs were obtained in [Calvanese *et al.*, 2009], but only for DLs that cannot express functionality constraints, or for the case where inverses are disallowed from both the constraints and the query. The case when both inverses and functionality are allowed remained open until now, since there are no algorithms for reasoning in logics that support both, together with regular expressions and nominals. A technique for containment of C2RPQs under this kind of expressive DL constraints had been proposed in [Calvanese *et al.*, 1998], but it turned out to be incomplete.

We show that, surprisingly, the presence of functionality constraints alone makes containment of 2RPQs EXPTIME-hard (it is PSPACE-complete in the absence of constraints [Calvanese *et al.*, 2003]). This result is based on the ability, under functionality, to generate/traverse a tree using a regular language. We also provide the first algorithm for containment of 2RPQs under functionality contraints, and obtain a tight EXPTIME upper bound that extends to very expressive DL constraints. This is achieved extending ideas used for 2RPQs without constraints [Calvanese *et al.*, 2000; 2003] to the setting of expressive DLs, in order to characterize a non-trivial class of structures over which containment can be decided using automata on infinite trees [Calvanese *et al.*, 2009]. For containment of a 2RPQ in a C2RPQs under functionality constraints alone, we prove a further jump in complexity to 2EXPTIME, and provide again a matching upper bound for containment between C2RPQs under expressive DL constraints. Our techniques provide also a solution to the problem of query entailment over DL KBs in which individuals in the ABox may be related through regular role-paths, since such ABoxes can be encoded as C2RPQs.

2 Preliminaries

We introduce now our notation for standard notions, the specific description logic (DL) that we use as a constraint language, and the class of queries that we consider.

We make use of regular languages, represented by nondeterministic finite state automata (NFAs) or regular expressions. Recall that an NFA over an *alphabet* Σ is a tuple $\alpha = \langle S, \Sigma, \delta, s, f\rangle$, where S is a finite set of *states*, $\delta \subseteq S \times \Sigma \times S$ the *transition relation*, $s \in S$ the *initial state*, and $f \in S$ the *final state*[2]. We use S_α, Σ_α, δ_α, $\mathsf{Ini}(\alpha)$, and $\mathsf{Fin}(\alpha)$ to denote the five components of such an α. We assume that there are no incoming transitions into the initial state, and no outgoing transitions from the final state (i.e., $s_2 \neq \mathsf{Ini}(\alpha)$ and $s_1 \neq \mathsf{Fin}(\alpha)$ for all $(s_1, c, s_2) \in \delta$). This is w.l.o.g., since every NFA can be translated in polynomial time into an NFA with the above properties while preserving the accepted language.

2.1 The Description Logic $\mathcal{ALCIF}b_{reg}$

We define now the DL $\mathcal{ALCIF}b_{reg}$. The vocabulary comprises countably infinite sets $\mathsf{N_R}$ of *role names* and $\mathsf{N_C}$ of *concept names*. If $P \in \mathsf{N_R}$, then P^- is the *inverse* of P, and $\overline{\mathsf{N_R}} = \mathsf{N_R} \cup \{P^- \mid P \in \mathsf{N_R}\}$ is the set of *atomic roles*. Each $R \in \overline{\mathsf{N_R}}$ is a *simple role*, and if R, R' are simple roles, so are $R \cap R'$, $R \cup R'$, and $R \setminus R'$. Every NFA whose alphabet is a set of simple roles is a *role*. Every $A \in \mathsf{N_C}$ is a *concept*, and if R is a simple role, α is a role, and C, C' are concepts, then $C \sqcap C'$, $C \sqcup C'$, $\exists\alpha.C$, $\forall\alpha.C$, $\leqslant 1\, R.C$ and $\geqslant 2\, R.C$ are concepts. An $\mathcal{ALCIF}b_{reg}$ *constraint* is an expression $C \sqsubseteq C'$ where C, C' are concepts. If C is of the form $\top \sqsubseteq \leqslant 1\, R.\top$ we call it a *functionality constraint* (where $\top$ is a shortcut for $C \sqcup \neg C$). A set $\mathcal{T}$ of $\mathcal{ALCIF}b_{reg}$ constraints is called a *TBox*. The *syntactic closure* of $\mathcal{T}$, $\mathsf{cl}(\mathcal{T})$, is defined in the usual way and is assumed to contain concepts in *negation normal form (NNF)* only. Recall that $\mathsf{cl}(\mathcal{T})$ contains every concept in $\mathcal{T}$ and is closed under subconcepts and negation (in NNF).

A structure is a pair $\mathcal{I} = \langle \Delta^\mathcal{I}, \cdot^\mathcal{I}\rangle$ where $\Delta^\mathcal{I} \neq \emptyset$, $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$ for each $A \in \mathsf{N_C}$, and $P^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$ for each $P \in \mathsf{N_R}$. The function $\cdot^\mathcal{I}$ is extended to concepts and roles in the usual way [Baader *et al.*, 2007], we only recall that $\alpha^\mathcal{I} = \{\langle d_1, d_2\rangle \mid d_2 \text{ is an } \alpha\text{-successor of } d_1 \text{ in } \mathcal{I}\}$, where d_2 *is an α-successor of d_1 in $\mathcal{I}$* if there is a word $R_1 \cdots R_n$ accepted by α and a sequence $e_0, \ldots, e_n$ of elements in $\Delta^\mathcal{I}$ such that $e_0 = d_1$, $e_n = d_2$, and $\langle e_{i-1}, e_i\rangle \in R_i^\mathcal{I}$ for $1 \leq i \leq n$. A structure $\mathcal{I}$ is a *model* of $\mathcal{T}$, in symbols $\mathcal{I} \models \mathcal{T}$, if $C^\mathcal{I} \subseteq C'^\mathcal{I}$ for every $C \sqsubseteq C' \in \mathcal{T}$.

2.2 (Conjunctive) 2-way Regular Path Queries

Let V be a countably infinite set of variables. An atom is an expression $\alpha(x, y)$, where $x, y \in \mathsf{V}$ and α is an NFA with alphabet $\Sigma_\alpha = \overline{\mathsf{N_R}}$. A *conjunctive 2-way regular path query (C2RPQ)* Q is an expression

$$Q(x_1, \ldots, x_n) \leftarrow \alpha_1(u_1, v_1), \ldots, \alpha_m(u_m, v_m),$$

where $\{x_1, \ldots, x_n\} \subseteq \{u_1, v_1, \ldots, u_m, v_m\}$ and each $\alpha_i(u_i, v_i)$, $1 \leq i \leq m$, is an atom. We let $\mathsf{At}(Q) = \bigcup_{1\leq i\leq m}\{\alpha_i(u_i, v_i)\}$ and $\mathsf{V}(Q) = \bigcup_{1\leq i\leq m}\{u_i, v_i\}$. A *match* for Q in a structure $\mathcal{I}$ is a mapping $\pi : \mathsf{V}(Q) \to \Delta^\mathcal{I}$ such that $\pi(y)$ is an α-successor of $\pi(x)$ for each $\alpha(x, y) \in \mathsf{At}(Q)$.

The tuple $\langle x_1, \ldots, x_n\rangle$ of *answer variables* of Q is denoted by $\mathsf{AVars}(Q)$. The *answer* to Q over $\mathcal{I}$, denoted $\mathsf{Ans}(Q, \mathcal{I})$, is the set of all n-tuples $\langle d_1, \ldots, d_n\rangle$ such that $\langle d_1, \ldots, d_n\rangle = \langle \pi(x_1), \ldots, \pi(x_n)\rangle$ for some match π for Q in $\mathcal{I}$.

A query is *Boolean* if it has no answer variables. A *2-way regular path query (2RPQ)* is a C2RPQ of the form $Q(x, y) \leftarrow \alpha(x, y)$. A *conjunctive query (CQ)* is a C2RPQ where the regular language of each atom consists of one role.[3]

The Query Containment Problem. Assume a pair Q_1, Q_2 of C2RPQs and a TBox $\mathcal{T}$. We say that Q_1 *is contained in* Q_2 *w.r.t.* $\mathcal{T}$, denoted $Q_1 \subseteq_\mathcal{T} Q_2$, if $\mathsf{Ans}(Q_1, \mathcal{I}) \subseteq \mathsf{Ans}(Q_2, \mathcal{I})$ for each model $\mathcal{I}$ of $\mathcal{T}$. The *query containment problem* is to decide given Q_1, Q_2, and $\mathcal{T}$, whether $Q_1 \subseteq_\mathcal{T} Q_2$.

3 Counter-models

In this section and the next one, we prove our upper bounds for query containment. Assume henceforth C2RPQs Q_1, Q_2 and a TBox $\mathcal{T}$. We refer to any structure $\mathcal{I}$ containing a tuple

[2] We can assume w.l.o.g. that an NFA has a unique final state.

[3] Unary atoms $A(x)$, usually allowed in such queries, can be encoded by $P_A(x, x')$, where P_A is a fresh role associated to the unary predicate A and x' is a fresh variable occurring nowhere else.

$\vec{d}$ such that $\vec{d} \in \mathsf{Ans}(Q_1, \mathcal{I})$ and $\vec{d} \notin \mathsf{Ans}(Q_2, \mathcal{I})$ as a *counter-example* (for "$Q_1 \subseteq Q_2$"). Note that $Q_1 \subseteq_{\mathcal{T}} Q_2$ does not hold (in symbols, $Q_1 \not\subseteq_{\mathcal{T}} Q_2$) iff there is a counter-example $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{T}$. We refer to such a structure as a *counter-model* (for "$Q_1 \subseteq_{\mathcal{T}} Q_2$"). Our first goal is to show that if $Q_1 \not\subseteq_{\mathcal{T}} Q_2$, then there is always a counter-model $\mathcal{I}$ with a "tree-like" structure that can be recognized using a tree-automaton.

Definition 1 (Tree-shaped structures)**.** *A* tree *is any prefix-closed set T of words over the set $\mathbb{N}^+$ of positive integers. A structure $\mathcal{I}$ is a* tree-structure *if $\Delta^{\mathcal{I}}$ is a tree and for each $d_1, d_2 \in \Delta^{\mathcal{I}}$ and atomic role R, $\langle d_1, d_2\rangle \in R^{\mathcal{I}}$ implies $d_2 = d_1 \cdot c$ or $d_1 = d_2 \cdot c$ for some $c \in \mathbb{N}^+$. A structure $\mathcal{I}$ is* tree-shaped *if it is isomorphic to some tree-structure.*

We show that $Q_1 \not\subseteq_{\mathcal{T}} Q_2$ implies that there exists a witnessing counter-model $\mathcal{I}$ that can be decomposed into a small number of tree-shaped structures because it has a small number of *split points*. Intuitively, split points are elements d in $\mathcal{I}$ that have at least 3 neighbors d_1, d_2, d_3 such that the 3 arcs $d \to d_1$, $d \to d_2$, and $d \to d_3$ lie on cycles in $\mathcal{I}$. It is not difficult to see that a structure with n split points can be represented as $n+1$ tree-shaped structures. To understand split points, we note that in the absence of DL constrains, the counter-example $\mathcal{I}$ can be assumed to resemble the structure of Q_1 and to have at most one split point for each variable of Q_1. In the presence of DL constraints more split points may be needed to account for elements that need to be collapsed due to functionality constraints. See Figure 1 for an illustration.

Given a structure $\mathcal{I}$ and a set $S \subseteq \Delta^{\mathcal{I}}$, we use $\mathcal{I}_{|S}$ to denote the restriction of $\mathcal{I}$ to the elements in S. A *path* in a structure $\mathcal{I}$ is a possibly infinite sequence $P = d_0, d_1, d_2, \ldots$ of elements in $\Delta^{\mathcal{I}}$ such that for each d_i in P, where $i > 0$, there is an atomic role R such that $\langle d_{i-1}, d_i\rangle \in R^{\mathcal{I}}$. We say P is *simple* if each element in P occurs exactly once.

Definition 2. *Assume a structure $\mathcal{I}$ and $d_0 \in \Delta^{\mathcal{I}}$. We let $\mathsf{dn}_{\mathcal{I}}(d_0) = \{d_1 \mid \exists R \in \overline{\mathsf{N_R}} : \langle d_0, d_1\rangle \in R^{\mathcal{I}}\}$ be the set of* (direct) neighbors *of d_0 in $\mathcal{I}$. A node $d_1 \in \mathsf{dn}_{\mathcal{I}}(d_0)$ is a* tree-neighbor of d_0 in $\mathcal{I}$ *if the structure $\mathcal{I}_{|\{d_0, d_1\} \cup S}$ is tree-shaped, where S is the set of all nodes $d_n \in \Delta^{\mathcal{I}}$ such that $\mathcal{I}$ has a simple path $d_0, d_1, d_2, \ldots, d_n$ for some $d_2, \ldots, d_{n-1}$.*

Let $\mathsf{tn}_{\mathcal{I}}(d_0)$ denote the set of all tree-neighbors of d in $\mathcal{I}$, and let $\mathsf{cn}_{\mathcal{I}}(d_0) = \mathsf{dn}_{\mathcal{I}}(d_0) \setminus \mathsf{tn}_{\mathcal{I}}(d_0)$ (the latter are cycle neighbors *of d_0). We let $\mathsf{Sp}(\mathcal{I}) = \{d \in \Delta^{\mathcal{I}} \mid |\mathsf{cn}_{\mathcal{I}}(d)| \geq 3\}$ be the* set of split points *in $\mathcal{I}$, and let $\mathsf{OD}(\mathcal{I})$ be the* out-degree *of $\mathcal{I}$, that is the maximum $|\mathsf{dn}_{\mathcal{I}}(d)|$ over all $d \in \Delta^{\mathcal{I}}$.*

Our goal now is to prove the following claim, to which we dedicate the rest of this section.

Proposition 3. *Let $\mathcal{T}$ be a TBox, Q_1 and Q_2 C2RPQs, m the number of atoms of Q_1, and k the total number of states over all atoms of Q_1. If $Q_1 \not\subseteq_{\mathcal{T}} Q_2$, then there is a counter-model $\mathcal{J}$ such that: (a) $|\mathsf{Sp}(\mathcal{J})| \leq |\mathsf{V}(Q_1)| + 2m$, and (b) $\mathsf{OD}(\mathcal{J})| \leq |\mathsf{cl}(\mathcal{T})| + 2k$.*

Proof. For a function $f : B_1 \to B_2$ and a tuple $\vec{t} = \langle b_1, \ldots, b_k\rangle \in (B_1)^k$, we let $f(\vec{t}) = \langle f(b_1), \ldots, f(b_k)\rangle$. Assume for the remainder of the proof that $\vec{z} = \langle z_1, \ldots, z_l\rangle = \mathsf{AVars}(Q_1)$ and $\vec{z}' = \langle z'_1, \ldots, z'_l\rangle = \mathsf{AVars}(Q_2)$.

Assume an arbitrary counter-model $\mathcal{I}$ for $Q_1 \not\subseteq_{\mathcal{T}} Q_2$. The strategy is to show that $\mathcal{I}$ can be reshaped into a counter-model with the desired properties. Since $\mathcal{I}$ is a counter-model, there is a tuple $\vec{v} = \langle v_1, \ldots, v_l\rangle \in (\Delta^{\mathcal{I}})^m$ and a match π for Q_1 such that $\vec{v} = \pi(\vec{z})$ and $\vec{v} \notin \mathsf{Ans}(Q_2, \mathcal{I})$. We assume that all NFAs of Q_1 have mutually disjoint state sets.

Take an arbitrary atom $at = \alpha(x, y)$ of Q_1. Since π is a match for Q_1 in $\mathcal{I}$, we have that $\pi(y)$ is an α-successor of $\pi(x)$ in $\mathcal{I}$. This means there is a sequence $\Gamma_{at} = \langle d_0, s_0\rangle, \ldots, \langle d_n, s_n\rangle$ such that

(a) $\langle d_0, s_0\rangle = \langle \pi(x), \mathsf{Ini}(\alpha)\rangle$, $\langle d_n, s_n\rangle = \langle \pi(y), \mathsf{Fin}(\alpha)\rangle$, and

(b) for each $0 < i \leq n$, there is an atomic role $R \in \overline{\mathsf{N_R}}$ such that $(s_{i-1}, R, s_i) \in \delta_\alpha$ and $\langle d_{i-1}, d_i\rangle \in R^{\mathcal{I}}$.

We assume that in each Γ_{at} there is no $i \neq j$ with $\langle d_i, s_i\rangle = \langle d_j, s_j\rangle$. Note that a Γ_{at} with a repetition of a pair can be shortened to a sequence with no repetition while preserving the satisfaction of (a) and (b) above. Let $\mathsf{Elem}(\Gamma_{at})$ be the set of all elements in Γ_{at}. We let $\mathsf{Elem}(Q_1) = \bigcup_{at \in \mathsf{At}(Q_1)} \mathsf{Elem}(\Gamma_{at})$, and note that $|\mathsf{Elem}(Q_1)| = \Sigma_{at \in \mathsf{At}(Q_1)} |\mathsf{Elem}(\Gamma_{at})|$.

Definition 4 (Induced interpretations)**.** *Assume a set $D \subseteq 2^{\Delta^{\mathcal{I}} \times B}$, where B is a set. We say D is* good *if for all $e \in D$ we have that (i) $e \neq \emptyset$, and (ii) $\langle d_1, s_1\rangle \in e$ and $\langle d_2, s_2\rangle \in e$ implies $d_1 = d_2$. If D is good and $e \in D$, then let $\hat{e}$ be the unique d_e in $\{d \mid \langle d, s\rangle \in e\}$. The interpretation induced by a good D, is $\mathcal{J}_D$ where (i) $\Delta^{\mathcal{J}_D} = D$; (ii) for each atomic concept A, $A^{\mathcal{J}_D} = \{e \in D \mid \hat{e} \in A^{\mathcal{I}}\}$; (iii) for each atomic role R, $R^{\mathcal{J}_D} = \{\langle e_1, e_2\rangle \in D^2 \mid \langle \hat{e}_1, \hat{e}_2\rangle \in R^{\mathcal{I}}\}$. For a good D, let $\xi_D : D \to \Delta^{\mathcal{I}}$ be the function $\xi_D(e) = \hat{e}$.*

As easily seen, if $D \subseteq 2^{\Delta^{\mathcal{I}} \times B}$ is good, then ξ_D is a homomorphism from $\mathcal{J}_D$ to $\mathcal{I}$. We can now proceed with the construction of a counter-model with the desired properties.

Stage 1 (Stretching). We construct a basic structure $\mathcal{J}_{D_0}$ such that $\mathsf{Ans}(Q_1, \mathcal{J}_{D_0}) \not\subseteq \mathsf{Ans}(Q_2, \mathcal{J}_{D_0})$. Intuitively, in $\mathcal{J}_{D_0}$ we 'stretch' the paths along which the atoms of Q_1 are matched, possibly duplicating objects so that each (non initial or final) state of each atom 'visits' a different domain object. We ensure that Q_2 still has no match in the stretched structure. $\mathcal{J}_{D_0}$ need not be a model of $\mathcal{T}$, but we shall fix this later.

For a variable $x \in \mathsf{V}(Q_1)$, let $\mathsf{St}(x)$ be the set of all states s such that there is $\alpha(x_1, x_2) \in \mathsf{At}(Q_1)$ such that *(a)* $x_1 = x$ and $s = \mathsf{Ini}(\alpha)$, or *(b)* $x_2 = x$ and $s = \mathsf{Fin}(\alpha)$. For each $x \in \mathsf{V}(Q_1)$, let $\kappa(x) = \{\langle d, s\rangle \in \mathsf{Elem}(Q_1) \mid s \in \mathsf{St}(x)\}$. Let

$$D_0 = \bigcup_{x \in \mathsf{V}(Q_1)} \{\kappa(x)\} \cup \bigcup_{\substack{\langle d, s\rangle \in \mathsf{Elem}(Q_1) \wedge \\ s \notin \bigcup_{x \in \mathsf{V}(Q_1)} \mathsf{St}(x)}} \{\{\langle d, s\rangle\}\}.$$

Our goal is to show that $\kappa(\vec{z}) \in \mathsf{Ans}(Q_1, \mathcal{J}_{D_0})$ and $\kappa(\vec{z}) \notin \mathsf{Ans}(Q_2, \mathcal{J}_{D_0})$ (recall $\kappa(\vec{z}) = \langle \kappa(z_1), \ldots, \kappa(z_l)\rangle$). Note first of all that D_0 is a good set by construction.

To see that $\kappa(\vec{z}) \in \mathsf{Ans}(Q_1, \mathcal{J}_{D_0})$, we simply show that κ is actually a match for Q_1 in $\mathcal{J}_{D_0}$. For this, let us assume an arbitrary atom $at = \alpha(x, y)$ of Q_1. We must show that $\kappa(y)$ is an α-successor of $\kappa(x)$. To this end,

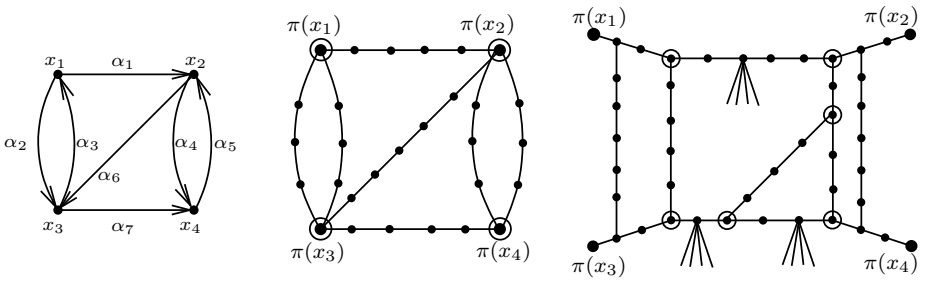

Figure 1: From left to right: an example query Q_1 (an arc $x \to y$ labeled with α stands for the atom $\alpha(x, y)$); the (classic) structure of a counter-example $\mathcal{I}$ for $Q_1 \subseteq Q_2$, for some (in this context irrelevant) query Q_2, in the absence of constraints (4 split points corresponding to $\mathsf{V}(Q_1)$); and a possible structure of $\mathcal{I}$ in the presence of DL constraints (6 split points, tree-shaped structures attached).

from $\Gamma_{at} = \langle d_0, s_0\rangle, \ldots, \langle d_n, s_n\rangle$ define a new sequence $\Gamma' = \langle e_0, s_0\rangle, \ldots, \langle e_n, s_n\rangle$, where each e_i is the unique element in D_0 with $\langle d_i, s_i\rangle \in e_i$. It remains to see:

(a) That $e_0 = \kappa(x)$ and $e_n = \kappa(y)$. Clearly, $s_0 \in \mathsf{St}(x)$ and $s_n \in \mathsf{St}(y)$. Then by the definition of D_0 we have $\langle d_0, s_0\rangle \in \kappa(x)$ and $\langle d_n, s_n\rangle \in \kappa(y)$. Due to the construction of Γ', we have $e_0 = \kappa(x)$ and $e_n = \kappa(y)$.

(b) That $s_0 = \mathsf{Ini}(\alpha)$ and $s_n = \mathsf{Fin}(\alpha)$. This holds by assumption of Γ_{at}.

(c) That for each $0 < i \leq n$, there is a symbol $R \in \overline{\mathsf{N_R}}$ such that $(s_{i-1}, R, s_i) \in \delta_\alpha$ and $\langle e_{i-1}, e_i\rangle \in R^{\mathcal{J}_{D_0}}$. Assume any such i. By assumption (i.e., due to Γ_{at}), there is a symbol $R \in \overline{\mathsf{N_R}}$ such that $(s_{i-1}, R, s_i) \in \delta_\alpha$ and $\langle d_{i-1}, d_i\rangle \in R^{\mathcal{I}}$. We know D_0 is a good set. Furthermore, $\langle d_{i-1}, s_{i-1}\rangle \in e_{i-1}$ and $\langle d_i, s_i\rangle \in e_i$ by construction of Γ'. Then by the construction of $\mathcal{J}_{D_0}$ from D_0 and $\mathcal{I}$, we have $\langle e_{i-1}, e_i\rangle \in R^{\mathcal{J}_{D_0}}$.

For $\kappa(\vec{z}) \notin \mathsf{Ans}(Q_2, \mathcal{J}_{D_0})$, assume towards a contradiction that $\kappa(\vec{z}) \in \mathsf{Ans}(Q_2, \mathcal{J}_{D_0})$. Consider the function ξ_{D_0}. As we know, ξ_{D_0} is a homomorphism from $\mathcal{J}_{D_0}$ to $\mathcal{I}$. Then clearly $\xi_{D_0}(\kappa(\vec{z})) \in \mathsf{Ans}(Q_2, \mathcal{I})$. Then the contradiction follows from the assumption that $\vec{v} \notin \mathsf{Ans}(Q_2, \mathcal{I})$ and the fact that $\xi_{D_0}(\kappa(\vec{z})) = \vec{v}$. Indeed, for any z_i in $\vec{z}$, we have $\xi_{D_0}(\kappa(z_i)) = \pi(z_i)$ and $\pi(z_i) = v_i$.

Stage 2 (Collapsing). This stage transforms $\mathcal{J}_{D_0}$ into a structure that does not violate any functionality constraints. Consider the following rewrite rule **RW** which rewrites a good set $D \subseteq 2^{\Delta^{\mathcal{I}} \times B}$: If there are $e, e_1, e_2 \in D$ and $\leqslant 1\, R.C \in \mathsf{cl}(\mathcal{T})$ such that $\hat{e} \in (\leqslant 1\, R.C)^{\mathcal{I}}$, $\{\langle e, e_1\rangle, \langle e, e_2\rangle\} \subseteq R^{\mathcal{J}_D}$, $\{e_1, e_2\} \subseteq C^{\mathcal{J}_D}$, and $e_1 \neq e_2$, then replace e_1 and e_2 by $e_3 = e_1 \cup e_2$. If D' is a result of applying **RW** on D, then we write $D \to D'$. If $D \to D'$, from $\hat{e} \in (\leqslant 1\, R.C)^{\mathcal{I}}$ it follows that $\hat{e}_1 = \hat{e}_2$, and hence D' is good.

We let $\pi_{D_0}(x) = \kappa(x)$. For each D' that results from rewriting, let $\pi_{D'}(x)$ be the unique $e \in D'$ with $\kappa(x) \subseteq e$. It is not difficult to see that the counter-example is preserved while rewriting D_0 with **RW**. This follows directly from the fact that $\mathcal{J}_{D_0}$ is a counter-example and the following lemma.

Lemma 5. *If (a) $D \to D'$, (b) π_D is a match for Q_1 in $\mathcal{J}_D$ s.t. $\pi_D(\vec{z}) \in \mathsf{Ans}(Q_1, \mathcal{J}_D)$, and (c) $\pi_D(\vec{z}) \notin \mathsf{Ans}(Q_2, \mathcal{J}_D)$, then $\pi_{D'}(\vec{z}) \in \mathsf{Ans}(Q_1, \mathcal{J}_{D'})$ and $\pi_{D'}(\vec{z}) \notin \mathsf{Ans}(Q_2, \mathcal{J}_{D'})$.*

Note that by the construction of D_0, $|\mathsf{Sp}(\mathcal{J}_{D_0})| \leq |\mathsf{V}(Q_1)|$. **RW** may add at most $2m$ new split point, i.e., $|\mathsf{Sp}(\mathcal{J}_{D_1})| \leq |\mathsf{V}(Q_1)| + 2m$. To see this, for a structure $\mathcal{J}$, we let $\mathsf{SD}(\mathcal{J}) = \Sigma_{d \in \mathsf{Sp}(\mathcal{J})}(|\mathsf{cn}_{\mathcal{J}}(d)| - 3)$. Then the bound is immediate from the easy fact that $\mathsf{SD}(\mathcal{J}_{D_0}) \leq 2m$ and the following:

Lemma 6. *If $D \to D'$, then either (a) $|\mathsf{Sp}(\mathcal{J}_{D'})| \leq |\mathsf{Sp}(\mathcal{J}_D)|$, or (b) $|\mathsf{Sp}(\mathcal{J}_{D'})| = |\mathsf{Sp}(\mathcal{J}_D)| + 1$ and $\mathsf{SD}(\mathcal{J}_{D'}) < \mathsf{SD}(\mathcal{J}_D)$.*

Proof. Let e, e_1, e_2, and e_3 witness $D \to D'$ as above. Assume the case $(\star)$ such that $\mathsf{Sp}(\mathcal{J}_{D'}) = \mathsf{Sp}(\mathcal{J}_D) \cup \{e_3\}$. Since $\{e_1, e_2\} \cap D' = \emptyset$, the assumption $(\star)$ implies $e_1 \notin \mathsf{Sp}(\mathcal{J}_D)$ and $e_2 \notin \mathsf{Sp}(\mathcal{J}_D)$. Since $e_3 \in \mathsf{Sp}(\mathcal{J}_{D'})$, we must have $|\mathsf{cn}_{\mathcal{J}_D}(e_1)| = |\mathsf{cn}_{\mathcal{J}_D}(e_2)| = 2$. This implies that $\{e_1, e_2\} \subseteq \mathsf{cn}_{\mathcal{J}_D}(e)$. There can be 3 possible cases:

(a) $|\mathsf{cn}_{\mathcal{J}_D}(e)| > 3$. From $\{e_1, e_2\} \subseteq \mathsf{cn}_{\mathcal{J}_D}(e)$ we know that $|\mathsf{cn}_{\mathcal{J}_{D'}}(e_3)| = 3$ and $|\mathsf{cn}_{\mathcal{J}_{D'}}(e)| = |\mathsf{cn}_{\mathcal{J}_D}(e)| - 1$. Then we obtain $\mathsf{SD}(\mathcal{J}_{D'}) = \mathsf{SD}(\mathcal{J}_D) - 1$.

(b) $|\mathsf{cn}_{\mathcal{J}_D}(e)| = 3$. Then $|\mathsf{cn}_{\mathcal{J}_{D'}}(e)| = 2$. This means $e \in \mathsf{Sp}(\mathcal{J}_D)$ and $e \notin \mathsf{Sp}(\mathcal{J}_{D'})$. Contradiction to $(\star)$.

(c) $|\mathsf{cn}_{\mathcal{J}_D}(e)| < 3$. Assume $\{e_1', e_1''\} = \mathsf{cn}_{\mathcal{J}_D}(e_1)$ and $\{e_2', e_2''\} = \mathsf{cn}_{\mathcal{J}_D}(e_2)$. We clearly have $\mathsf{cn}_{\mathcal{J}_{D'}}(e_3) \subseteq \mathsf{cn}_{\mathcal{J}_D}(e_1) \cup \mathsf{cn}_{\mathcal{J}_D}(e_2)$. Since $\{e_1, e_2\} \subseteq \mathsf{cn}_{\mathcal{J}_D}(e)$, we get $|\mathsf{cn}_{\mathcal{J}_D}(e)| = 2$, $e \in \mathsf{cn}_{\mathcal{J}_D}(e_1)$, and $e \in \mathsf{cn}_{\mathcal{J}_D}(e_2)$. Thus $|\mathsf{cn}_{\mathcal{J}_D}(e_1) \cup \mathsf{cn}_{\mathcal{J}_D}(e_2)| \leq 3$ and $|\mathsf{cn}_{\mathcal{J}_{D'}}(e)| = 1$. Since $e_3 \in \mathsf{cn}_{\mathcal{J}_{D'}}(e)$, we get $e \notin \mathsf{cn}_{\mathcal{J}_{D'}}(e_3)$. From this we obtain that $|\mathsf{cn}_{\mathcal{J}_{D'}}(e_3)| \leq 2$, which contradicts the assumption that $e_3 \in \mathsf{Sp}(\mathcal{J}_{D'})$.

Assume the remaining case where $(\star)$ is false. Note that for all $e' \in D'$, where $e' \neq e_3$, we have $|\mathsf{cn}_{\mathcal{J}_{D'}}(e')| \leq |\mathsf{cn}_{\mathcal{J}_D}(e')|$. Thus $\mathsf{Sp}(\mathcal{J}_{D'}) \subseteq \mathsf{Sp}(\mathcal{J}_D) \cup \{e_3\}$. Thus if $\mathsf{Sp}(\mathcal{J}_{D'}) \neq \mathsf{Sp}(\mathcal{J}_D) \cup \{e_3\}$, then either $e_3 \notin \mathsf{Sp}(\mathcal{J}_{D'})$ or $\mathsf{Sp}(\mathcal{J}_{D'}) \subsetneq \mathsf{Sp}(\mathcal{J}_D)$. In any case, $|\mathsf{Sp}(\mathcal{J}_{D'})| \leq |\mathsf{Sp}(\mathcal{J}_D)|$. □

Let D_1 be the good set that results after applying **RW** exhaustively on D_0. It is not hard to see the $\mathsf{OD}(\mathcal{J}_{D_1})| \leq 2k$ bound, which follows from the following lemma:

Lemma 7. *For all $e \in D_1$, $|e| \leq k$ and $|\mathsf{dn}_{\mathcal{J}_{D_1}}(e)| \leq 2|e|$.*

Stage 3 (Unraveling). Stage 2 guarantees a counter-example $\mathcal{J}_{D_1}$ with few splits points and small out-degree. Functionality cannot be violated in $\mathcal{J}_{D_1}$, because $\mathcal{I} \models \mathcal{T}$ and each node e in $\Delta^{\mathcal{J}_{D_1}}$ satisfies every $\leqslant 1\, R.C \in \mathsf{cl}(\mathcal{T})$ that $\hat{e}$ satisfies in $\mathcal{I}$. Moreover, e also satisfies in $\mathcal{J}_{D_1}$ the same concept names and Boolean combinations thereof as $\hat{e}$ satisfies in $\mathcal{I}$. However, $\mathcal{J}_{D_1}$ might not be a model of $\mathcal{T}$ because nodes might lack neighbors needed to satisfy existential concepts of the forms $\exists\alpha.C$ and $\geqslant 2\, R.C$ in $\mathsf{cl}(\mathcal{T})$. This can be fixed using the standard unraveling technique for $\mathcal{ALCIF}b_{reg}$ [Calvanese *et al.*, 2009], which allows us to satisfy existential concepts by attaching to the nodes of $\mathcal{J}_{D_1}$ tree-shaped structures obtained from $\mathcal{I}$, which have branching $\leq |\mathsf{cl}(\mathcal{T})|$. The resulting structure $\mathcal{J}$ may be infinite, but the number of split points is preserved and the out-degree is still bounded by $|\mathsf{cl}(\mathcal{T})| + 2k$. Furthermore, there is a homomorphism from $\mathcal{J}$ to $\mathcal{I}$ (coinciding with $e \mapsto \hat{e}$ for the nodes in $\mathcal{J}_{D_1}$) that preserves all concepts in $\mathsf{cl}(\mathcal{T})$ and all paths that are relevant for Q_1 and Q_2. $\mathcal{J}$ is the desired counter-model. This concludes the proof of Proposition 3. □

4 Deciding Query Containment

Similarly to tree-shaped structures, we define *forest-shaped* ones. Such a structure is essentially the union of a finite set of tree-shaped structures, which additionally allow any node to be directly related to the root of any tree.

Definition 8 (Forest-shaped structures)**.** *A* forest of degree k with at most n roots*, or simply an (n,k)-forest, is a subset F of $S = \{1,\dots,n\}\cdot\{1,\dots,k\}^*$ such that $w\cdot c \in F$ and $w \in S$ imply $w \in F$. The elements of $F \cap \{1,\dots,n\}$ are called* roots *of F. A structure $\mathcal{I}$ is an* (n,k)-forest structure *if $\Delta^{\mathcal{I}}$ is an (n,k)-forest and for each $d_1, d_2 \in \Delta^{\mathcal{I}}$ and each atomic role R, $\langle d_1, d_2\rangle \in R^{\mathcal{I}}$ implies $\{d_1, d_2\} \cap \{1,\dots,n\} \neq \emptyset$, $d_2 = d_1 \cdot c$, or $d_1 = d_2 \cdot c$ for some $c \in \{1,\dots,k\}$. A structure $\mathcal{I}$ is* (n,k)-forest-shaped *if $\mathcal{I}$ is isomorphic to some (n,k)-forest structure.*

We can now establish a crucial relationship between the counter-model $\mathcal{J}$ described in the previous section and forest-shaped structures: by viewing split points as roots, we can view $\mathcal{J}$ as a forest. More formally:

Theorem 9. *If $\mathcal{J}$ is a structure with $|\mathsf{Sp}(\mathcal{J})| < n$ and $\mathsf{OD}(\mathcal{J}) \leq k$, then $\mathcal{J}$ is (n,k)-forest-shaped.*

Proof. Let $\mathsf{R} = \mathsf{Sp}(\mathcal{J})$ if $\mathsf{Sp}(\mathcal{J}) \neq \emptyset$, and $\mathsf{R} = \{e\}$, where $e \in \Delta^{\mathcal{J}}$ is arbitrary, otherwise. Let $s_1,\dots,s_m$ be an enumeration of R. For each $s_i \in \mathsf{R}$, let $\mathcal{J}_i = \mathcal{J}|S_i$, where S_i is the set of all nodes $d_l \in \Delta^{\mathcal{J}}$ such that $\mathcal{J}$ has a simple path $d_0, d_1, d_2, \dots, d_l$ where $d_0 = s_i$ and either (i) d_i is a tree-neighbour of d_{i-1} for every $1 < i < l$, or (ii) $d_l = s_j \in \mathsf{R}$ with $j \leq i$, and for each d_i, $1 \leq i < l$, $d_i \notin \mathsf{R}$. For each $d \in \Delta^{\mathcal{J}} \setminus \mathsf{R}$ there is exactly one $\mathcal{J}_i$ such that $d \in \Delta^{\mathcal{J}_i}$. Each node $s_i \in \mathsf{R}$ is the root of exactly one tree $\mathcal{J}_i$, and may occur as a leaf of any $\mathcal{J}_j$. Take the interpretation $\mathcal{J}'$, where (a) $\Delta^{\mathcal{J}'} = \bigcup_{s_i \in \mathsf{R}} \Delta^{\mathcal{J}_i}$, (b) for all atomic A, $A^{\mathcal{J}'} = \bigcup_{s_i \in \mathsf{R}} A^{\mathcal{J}_i}$, (c) for all roles R, $R^{\mathcal{J}'} = \bigcup_{s_i \in \mathsf{R}} r^{\mathcal{J}_i}$. It is easy to verify that $\mathcal{J}'$ is isomorphic to $\mathcal{J}$, it is forest-shaped and the roots in $\mathcal{J}'$ are exactly the elements in R. Thus $\mathcal{J}'$ is (m,k)-forest-shaped, and also (n,k)-forest-shaped. □

To recognize the existence of such counter-models, and thus decide query containment, we rely on *2-way alternating automata on infinite trees (2ATA)* [Vardi, 1998] and adapt known techniques for query answering in expressive DLs [Calvanese *et al.*, 2009].

Theorem 10. *Given an $\mathcal{ALCIF}b_{reg}$ TBox $\mathcal{T}$ and C2RPQs Q_1 and Q_2, deciding $Q_1 \subseteq_{\mathcal{T}} Q_2$ is in* 2ExpTime*. If Q_2 is a 2RPQ, deciding $Q_1 \subseteq_{\mathcal{T}} Q_2$ is in* ExpTime*.*

Proof. (Sketch.) Let $m = |\mathsf{At}(Q_1)|$, let k be the number of states over all atoms of Q_1, and let $\vec{z} = \mathsf{AVars}(Q_1)$. By Proposition 3 and Theorem 9, $Q_1 \not\subseteq_{\mathcal{T}} Q_2$ iff there is an (m',k')-forest shaped $\mathcal{J}$ and a match π for Q_1 in $\mathcal{J}$ such that $\pi(\vec{z}) \notin \mathsf{Ans}(Q_2, \mathcal{J})$, where $m' = |\mathsf{V}(Q_1)| + 2m + 1$ and $k' = |\mathsf{cl}(\mathcal{T})| + 2k$.

An (m',k')-forest shaped structure can be represented as a labeled tree with branching degree $\max(m',k')$. With some minor adaptations to the constructions in [Calvanese *et al.*, 2009], we can build a 2ATA $\mathbf{A}$ that accepts a labeled tree $\mathbf{T}$ iff it represents an (m',k')-forest shaped model $\mathcal{J}_{\mathbf{T}}$ of $\mathcal{T}$ where $\mathsf{Ans}(Q_1, \mathcal{J}) \not\subseteq \mathsf{Ans}(Q_2, \mathcal{J})$. Every tree accepted by $\mathbf{A}$ is a counter-example, hence $Q_1 \not\subseteq_{\mathcal{T}} Q_2$ iff the language of $\mathbf{A}$ is not empty. Roughly, $\mathbf{A}$ is obtained by intersecting three automata: $\mathbf{A}_{\mathcal{T}}$ that accepts $\mathbf{T}$ iff $\mathcal{J}_{\mathbf{T}}$ satisfies the constraints in $\mathcal{T}$, $\mathbf{A}_{Q_1}$ that accepts $\mathbf{T}$ iff there is a match π for Q_1 in $\mathcal{J}_{\mathbf{T}}$, and $\mathbf{A}_{\neg Q_2}$ that accepts $\mathbf{T}$ iff there is no match for Q_2 in $\mathcal{J}_{\mathbf{T}}$ giving $\pi(\vec{z})$ as an answer.

In the case where Q_2 is a 2RPQ all three automata are of small size (the state set is polynomial in the combined size of $\mathcal{T}$, Q_1, and Q_2, the alphabet is single exponential, and the size of the acceptance condition is fixed). Hence we can test in single exponential time if the language of $\mathbf{A}$ is empty [Vardi, 1998].

In the case where Q_2 is a C2RPQ, the automata $\mathbf{A}_{\mathcal{T}}$ and $\mathbf{A}_{Q_1}$ are exactly the same as above. However, due to the presence of non-answer (i.e., existentially quantified) variables in Q_2, to obtain $\mathbf{A}_{\neg Q_2}$ we need automata theoretic operations that cause an exponential blow-up in the size of the automata (in particular, we need projection, which requires to transform a 2ATA into an exponentially larger 1-way nondeterministic automaton (1NTA), and then complementation, which causes a blow-up in the size of the 1NTA). This results in the 2ExpTime upper bound. □

5 Lower Bounds

We first recall the DL $\mathcal{ALCI}$, obtained from $\mathcal{ALCIF}b_{reg}$ by disallowing $\leqslant 1\, R.C$ from the syntax and allowing in concepts $\exists R.C$ and $\forall R.C$ only atomic roles R. It is well known that checking whether an $\mathcal{ALCI}$ TBox $\mathcal{T}$ has a model $\mathcal{I}$ is ExpTime-hard [Baader *et al.*, 2007], and checking whether $\mathcal{T}$ has a model $\mathcal{I}$ where a CQ Q has no match is 2ExpTime-hard [Lutz, 2008]. It follows immediately that the bounds in Theorem 10 are tight.

Surprisingly, however, they are tight already for *functional TBoxes*, i.e., very simple TBoxes consisting of *functionality constraints only*, as we show next.

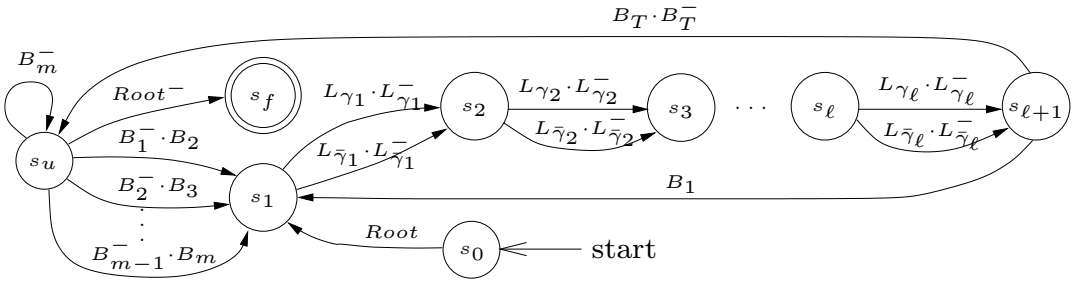

Figure 2: The automaton for the query $Q_1(x, y) \leftarrow \alpha_1(x, y)$.

By examining the reductions which result in the above mentioned lower bounds, it is not hard to see that the bounds also apply if we ask $\mathcal{I}$ to be a *finite-tree*, that is, a tree-shaped structure with a finite domain and branching at most $|\mathsf{cl}(\mathcal{T})|$.

Lemma 11. *For $\mathcal{ALCI}$, (†) checking whether a TBox $\mathcal{T}$ has a finite-tree model is* ExpTime*-hard, and (‡) checking whether a TBox $\mathcal{T}$ has a finite-tree model $\mathcal{I}$ where a Boolean CQ has no match is* 2ExpTime*-hard.*

To prove our lower bounds, we first polynomially reduce problem (†) to containment of two 2RPQs w.r.t. functional TBoxes. Specifically, given an $\mathcal{ALCI}$ TBox $\mathcal{T}$, we show how to construct a functional TBox $\mathcal{T}_f$, and 2RPQs $Q_1(x, y) \leftarrow \alpha_1(x, y)$ and $Q_2(x, y) \leftarrow \alpha_2(x, y)$ such that $\mathcal{T}$ has a finite-tree model iff $Q_1 \not\subseteq_{\mathcal{T}_f} Q_2$. The idea is to use Q_1 and functionality constraints to generate tree-shaped structures encoding interpretations of $\mathcal{T}$, and to use Q_2 to find *errors* in such structures. An error is either a mistake in the encoding of the interpretation or a violation of a constraint of $\mathcal{T}$.

Let $\sim C$ denote the NNF of $\neg C$ and let $\mathsf{cl}'(\mathcal{T}) = \mathsf{cl}(\mathcal{T}) \cup \{R, \neg R \mid R \text{ or } R^- \text{ occurs in } \mathcal{T}\}$. Intuitively, Q_1 is used to generate a finite tree in which each node "stores" the concepts of $\mathsf{cl}(\mathcal{T})$ that the node satisfies and the roles that connect it to its parent. Let $m = |\mathsf{cl}(\mathcal{T})|$, let $\gamma_1, \ldots, \gamma_\ell$ be an enumeration of $\mathsf{cl}'(\mathcal{T})$, and let $\bar{\gamma}$ be the complementary symbol of γ, i.e., *(a)* if γ is a concept, then $\bar{\gamma} = \sim\gamma$; *(b)* if γ is a role, then $\bar{\gamma} = \neg\gamma$; *(c)* if γ is an expression $\neg R$ then $\bar{\gamma} = R$.

We can now define $Q_1(x, y) \leftarrow \alpha_1(x, y)$, where α_1 is the automaton in Figure 2 that is built using an alphabet Σ consisting of the following roles and their inverses: $Root$, B_T, $B_1, \ldots, B_m$, $L_{\gamma_1}, \ldots, L_{\gamma_\ell}$. The TBox is defined as $\mathcal{T}_f = \{\top \sqsubseteq \leqslant 1\, R.\top \mid R \in \Sigma\}$, i.e., all roles are functional.

To understand how α_1 works, a simple but crucial observation is that, if as role R^- is functional, then by starting from a node n of a structure and traversing $R \cdot R^-$ edges one gets back to n. We exploit this to explain how α_1 generates (i.e., recognizes) finite trees encoding interpretations. From the initial state it first makes a $Root$-step to state s_1, from where it starts generating the interpretation. From s_1 it goes to state $s_{\ell+1}$ via intermediate states that ensure that for each γ_i, $1 \leq i \leq \ell$, one of γ_i or $\bar{\gamma}_i$ is selected. Notice that, by the observation above, if α_1 traverses a node n of a structure in state s_1, in each of the states $s_2, \ldots, s_{\ell+1}$ it will again traverse n.

In $s_{\ell+1}$, α_1 can decide whether to continue by generating a B_1-child, or to continue by marking the node as terminal (via B_T). If α_1 chooses the latter, it gets into s_u, from where it continues as follows: *(i)* if the current node is a B_m-successor of its parent, then it moves up in the tree and returns to state s_u; *(ii)* if the current node is a B_i-successor of its parent, where $i < m$, it moves up with B_i^- and then down to generate the B_{i+1}-child of the parent, together with its sub-tree. An example structure for the case $m = 2$ is given in Figure 3, where each β_i, for $1 \leq i \leq \ell$, is either γ_i or $\bar{\gamma}_i$.

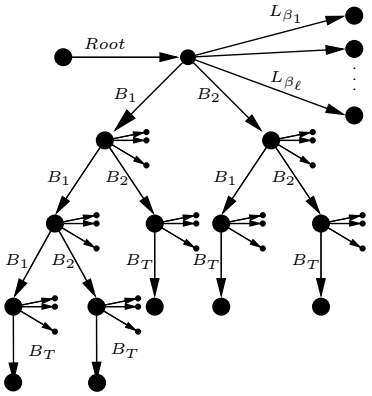

Figure 3: Example structure without structural errors 'generated' by the query Q_1 in Figure 2, for the case where $m = 2$.

There are two possible structural errors in the structure generated by α_1: *(A)* when α_1 reads B_i^- in the state s_u it might "introduce" a fresh B_i^--edge to a child instead of moving up, and *(B)* α_1 might not end up where it started, i.e., $Root$ occurs more than once in the tree. To solve (A) and (B), and to ensure the satisfaction of $\mathcal{T}$, we build α_2, which we describe using regular expressions. For a role R, let $R^{\pm} = R \cdot R^-$ and $R^{\mp} = R^- \cdot R$. Then $\alpha_2 = \Sigma^* \cdot (\alpha_2^1 + \cdots + \alpha_2^8) \cdot \Sigma^*$. Intuitively, a match for Q_2 in a structure generated by α_1 means that the structure has one of eight errors, respectively encoded by the following expressions $\alpha_2^1, \ldots, \alpha_2^8$:

(a) $\alpha_2^1 = \bigcup_{1 \leq i,j \leq m,\, i \neq j} (B_i \cdot B_j^-)$. Tests for error (A).

(b) $\alpha_2^1 = Root^- \cdot \bigcup_{1 \leq i \leq m} B_i^-$. Tests for error (B).

(c) $\alpha_2^3 = \bigcup_{C \sqsubseteq D \in \mathcal{T}} (L_C^{\pm} \cdot L_{\sim D}^{\pm})$. Checks for a violation of $\mathcal{T}$.

(d) $\alpha_2^4 = \bigcup_{C_1 \sqcup C_2 \in \mathsf{cl}(\mathcal{T})} (L^{\pm}_{C_1 \sqcup C_2} \cdot L^{\pm}_{\sim C_1} \cdot L^{\pm}_{\sim C_2})$. Trivial.

(e) $\alpha_2^5 = \bigcup_{C_1 \sqcap C_2 \in \mathsf{cl}(\mathcal{T})} (L^{\pm}_{C_1 \sqcap C_2} \cdot (L^{\pm}_{\sim C_1} + L^{\pm}_{\sim C_2}))$. Trivial.

(f) $\alpha_2^6 = \bigcup_{C \in \mathsf{cl}(\mathcal{T})} (L^{\pm}_{C} \cdot L^{\pm}_{\sim C})$. Trivial.

(g) $\alpha_2^7 = \bigcup_{\forall R.C \in \mathsf{cl}(\mathcal{T})} (L^{\pm}_{\forall R.C} \cdot (\bigcup_{1 \leq i \leq m} (B_i \cdot L^{\pm}_R + L^{\mp}_R \cdot B_i^-)) \cdot L^{\pm}_{\sim C})$.
Intuitively, we have a violation of $\forall R.C$ if we can select a B_i-child that has R and $L_{\sim C}$ in its label, or the node has R^- in its label and its parent has $L_{\sim C}$.

(h) $\alpha_2^8 = \bigcup_{\exists R.C \in \mathsf{cl}(\mathcal{T})} (L^{\pm}_{\exists R.C} \cdot \underbrace{(\mathit{Root}^{\mp} + L^{\pm}_{\neg R^-} + G_1)}_{\star} \cdot \underbrace{(B_T^{\pm} + G_2)}_{\star\star})$,
where $G_1 = (B_1^- \cdot L^{\pm}_{\sim C} \cdot B_1) + \cdots + (B_m^- \cdot L^{\pm}_{\sim C} \cdot B_m)$, and $G_2 = (B_1 \cdot (L^{\pm}_{\neg R} + L^{\pm}_{\sim C}) \cdot B_1^-) \cdots (B_m \cdot (L^{\pm}_{\neg R} + L^{\pm}_{\sim C}) \cdot B_m^-)$.
Intuitively, we have a violation of $\exists R.C$ if *(a)* the parent of the node is not an R-neighbor where C holds (ensured by $\star$), and *(b)* all the children are not R-neighbors where C holds (ensured by $\star\star$).

By this construction we have that $\mathcal{T}$ has a finite-tree model iff $Q_1 \not\subseteq_{\mathcal{T}_f} Q_2$. Then from Theorem 10 we obtain:

Theorem 12. *Containment of two 2RPQs w.r.t. $\mathcal{ALCIF}b_{reg}$ TBoxes is* EXPTIME*-complete, even for TBoxes consisting of functionality constraints only.*

We can exploit the above ideas also to polynomially reduce (‡) to containment of an RPQ in a C2RPQ w.r.t. $\mathcal{ALCIF}b_{reg}$ TBoxes. The query Q_1 on the left and the TBox $\mathcal{T}_f$ remain the same. We assume that all variables in the query Q over $\mathcal{T}$ are different from x, y. For a role R, let $\alpha^R = (\bigcup_{1 \leq i \leq m} B_i) \cdot L^{\pm}_R + L^{\pm}_{R^-} \cdot (\bigcup_{1 \leq i \leq m} B_i^-)$. We build a new query Q' from Q_2 by replacing each atom $R(z_1, z_2)$ by $(\alpha^R + \alpha_2)(z_1, z_2)$, and adding the atom $\Sigma^*(x, y)$. Then $\mathcal{T}$ has a finite-tree counter-model for Q iff $Q_1 \not\subseteq_{\mathcal{T}_f} Q'$. By combining this with Theorem 10, we obtain:

Theorem 13. *Containment of a 2RPQ in a C2RPQ w.r.t. $\mathcal{ALCIF}b_{reg}$ TBoxes is* 2EXPTIME*-complete, even for TBoxes consisting of functionality constraints only.*

6 Regular ABoxes

Apart from the TBox, DL knowledge bases have an assertional component called *ABox*, which is a set of *assertions* $a{:}C$ and $(a, b){:}R$, stating the participation of the (objects denoted by) individual names $a, b, \ldots$ in (the interpretation of) concepts C and roles R. Until now, in DLs that support regular languages over atomic roles as role expressions, ABox assertions $(a, b){:}R$ are restricted to atomic or simple roles R, e.g., in [Calvanese *et al.*, 2009]. The results of this paper yield the first algorithms and optimal complexity bounds for reasoning in the presence of NFAs or regular expressions in the ABox.

Formally, a $\mathcal{ALCIF}b_{reg}$ KB is a pair $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ where $\mathcal{T}$ is a TBox and $\mathcal{A}$ is a set of assertions of the form $(a, b){:}\alpha$ where a, b are *individuals* and α is a role. A structure $\mathcal{I}$ interprets each individual a as an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. We say that $(a, b){:}\alpha$ is *satisfied* in a structure $\mathcal{I}$ if $\langle a^{\mathcal{I}}, b^{\mathcal{I}} \rangle \in \alpha^{\mathcal{I}}$. A model of the KB is a model of $\mathcal{T}$ that satisfies all the assertions in $\mathcal{A}$. Given a Boolean C2RPQ Q, we say that $\mathcal{K}$ *entails* Q, written $\mathcal{K} \models Q$, if Q has a match is every model of $\mathcal{K}$.[4] Given a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ where $\mathcal{A} = \{(a_1, b_1){:}\alpha_1, \ldots, (a_n, b_n){:}\alpha_n\}$, let $Q_{\mathcal{A}} \leftarrow \alpha_1(u_{a_1}, v_{b_1}), \ldots, \alpha_m(u_{b_n}, v_{b_n})$ be a Boolean C2RPQ obtained from $\mathcal{A}$ by using a fresh variable x_a for each individual a. Then, $\mathcal{I}$ is a model of $\mathcal{K}$ iff $\mathcal{I} \models \mathcal{T}$ and there is a match for $Q_{\mathcal{A}}$ in $\mathcal{I}$. Hence, $\mathcal{K}$ is satisfiable iff $Q_{\mathcal{A}} \not\subseteq_{\mathcal{T}} Q_{\bot}$, where $Q_{\bot}$ is a query that does not have a match in any interpretation (such as $Q_{\bot} \leftarrow \alpha_{\bot}(x, y)$ where $\alpha_{\bot}$ is an NFA whose language is the simple role $R \setminus R$). Moreover, given a C2RPQ Q, we can also decide *query entailment* w.r.t. $\mathcal{K}$, by exploiting that $\mathcal{K} \models Q$ iff $Q_{\mathcal{A}} \subseteq_{\mathcal{T}} Q$. Hence we obtain:

Theorem 14. *Let $\mathcal{K}$ be a $\mathcal{ALCIF}b_{reg}$ KB whose ABox consists of assertions $(a, b){:}\alpha$, for an NFA α over the alphabet of simple roles, and let Q be a Boolean C2RPQ. Then (i) deciding whether $\mathcal{K}$ is satisfiable is* EXPTIME*-complete, and (ii) deciding whether $\mathcal{K} \models Q$ is* 2EXPTIME*-complete.*

7 Conclusions

We have closed the open problem of decidability of containment of 2RPQs and of C2RPQs in the presence of constraints, that range from simple functionality constraints to very expressive DL constraints, and have given optimal algorithms for the problem. Indeed, the upper bounds given here extend to the prominent DL $\mathcal{SHIQ}$ that underlies OWL Lite (which can be polynomially encoded into $\mathcal{ALCIF}b_{reg}$ provided that numbers are coded in unary [Calvanese *et al.*, 2009; Rudolph *et al.*, 2008]). They also show decidability of query containment for the fragment $\mathcal{SRIQ}^-$ of $\mathcal{SROIQ}$ that disallows nominals, role (ir)reflexivity axioms, and concepts of the form $\exists \mathsf{Self}.r$. However, since the reduction of $\mathcal{SRIQ}^-$ to $\mathcal{ALCIF}b_{reg}$ is exponential in general [Kazakov, 2008], we only obtain a 2EXPTIME upper bound for 2RPQs and a 3EXPTIME upper bound for C2RPQs which are not known to be tight. We believe that our techniques can be extended with the $\exists \mathsf{Self}.r$ construct to obtain a tight upper bound for the DL $\mathcal{ZIQ}$, which can simulate (with an exponential blow-up) full $\mathcal{SRIQ}$. Our results are also relevant because they provide a necessary step towards solving containment of SPARQL 1.1 queries under the OWL 2 entailment regimes.

References

[Abiteboul *et al.*, 1995] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison Wesley Publ. Co., 1995.

[Baader *et al.*, 2007] Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2nd edition, 2007.

[Barcelò *et al.*, 2010] Bablo Barcelò, Carlos A. Hurtado, Leonid Libkin, and Peter T. Wood. Expressive languages

[4] We assume that concept assertions $a{:}C$ are encoded by replacing them with $\alpha_C(a, a_C)$ and adding $\exists P_C.\top \sqsubseteq C$ to the TBox, where a_C is a fresh individual, P_C is a fresh role name, and α_C is the NFA whose language contains exactly the role name P_C.

for path queries over graph-structured data. In *Proc. of the 29th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS 2010)*, pages 3–14, 2010.

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

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

[Calvanese *et al.*, 2003] Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Moshe Y. Vardi. Reasoning on regular path queries. *SIGMOD Record*, 32(4):83–92, 2003.

[Calvanese *et al.*, 2009] Diego Calvanese, Thomas Eiter, and Magdalena Ortiz. Regular path queries in expressive description logics with nominals. In *Proc. of the 21st Int. Joint Conf. on Artificial Intelligence (IJCAI 2009)*, pages 714–720, 2009.

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

[Chaudhuri and Vardi, 1997] Surajit Chaudhuri and Moshe Y. Vardi. On the equivalence of recursive and nonrecursive datalog programs. *J. of Computer and System Sciences*, 54(1):61–78, 1997.

[Eiter *et al.*, 2009] Thomas Eiter, Carsten Lutz, Magdalena Ortiz, and Mantas Šimkus. Query answering in description logics with transitive roles. In *Proc. of the 21st Int. Joint Conf. on Artificial Intelligence (IJCAI 2009)*, pages 759–764, 2009.

[Glimm *et al.*, 2008] Birte Glimm, Ian Horrocks, Carsten Lutz, and Uli Sattler. Conjunctive query answering for the description logic $\mathcal{SHIQ}$. *J. of Artificial Intelligence Research*, 31:151–198, 2008.

[Johnson and Klug, 1984] David S. Johnson and Anthony C. Klug. Testing containment of conjunctive queries under functional and inclusion dependencies. *J. of Computer and System Sciences*, 28(1):167–189, 1984.

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

[Levy and Rousset, 1998] Alon Y. Levy and Marie-Christine Rousset. Verification of knowledge bases based on containment checking. *Artificial Intelligence*, 101(1-2):227–250, 1998.

[Lutz, 2008] Carsten Lutz. The complexity of conjunctive query answering in expressive description logics. In *Proc. of the 4th Int. Joint Conf. on Automated Reasoning (IJCAR 2008)*, volume 5195 of *Lecture Notes in Artificial Intelligence*, pages 179–193. Springer, 2008.

[Rudolph *et al.*, 2008] Sebastian Rudolph, Markus Krötzsch, and Pascal Hitzler. Terminological reasoning in $\mathcal{SHIQ}$ with ordered binary decision diagrams. In *Proc. of the 23rd AAAI Conf. on Artificial Intelligence (AAAI 2008)*, pages 529–534, 2008.

[Sagiv, 1988] Yehoshua Sagiv. Optimizing Datalog programs. In Jack Minker, editor, *Foundations of Deductive Databases and Logic Programming*, pages 659–698. Morgan Kaufmann, 1988.

[Shmueli, 1993] Oded Shmueli. Equivalence of Datalog queries is undecidable. *J. of Logic Programming*, 15(3):231–241, 1993.

[ten Cate and Lutz, 2009] Balder ten Cate and Carsten Lutz. The complexity of query containment in expressive fragments of XPath 2.0. *J. of the ACM*, 56(6), 2009.

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