Approximation and Semantic Tree-Width of Conjunctive Regular Path Queries

Diego Figueira

Univ. Bordeaux, CNRS, Bordeaux INP, LaBRI, UMR5800, F-33400 Talence, France

Rémi Morvan

Univ. Bordeaux, CNRS, Bordeaux INP, LaBRI, UMR5800, F-33400 Talence, France

Abstract

We show that the problem of whether a query is equivalent to a query of tree-width k is decidable, for the class of Unions of Conjunctive Regular Path Queries with two-way navigation (UC2RPQs). A previous result by Barceló, Romero, and Vardi [5] has shown decidability for the case $k = 1$, and here we show that decidability in fact holds for any arbitrary $k > 1$. The algorithm is in 2ExpSpace, but for the restricted but practically relevant case where all regular expressions of the query are of the form a^* or $(a_1 + \cdots + a_n)$ we show that the complexity of the problem drops to Π_2^p.

We also investigate the related problem of approximating a UC2RPQ by queries of small tree-width. We exhibit an algorithm which, for any fixed number k, builds the maximal under-approximation of tree-width k of a UC2RPQ. The maximal under-approximation of tree-width k of a query q is a query q' of tree-width k which is contained in q in a maximal and unique way, that is, such that for every query q'' of tree-width k, if q'' is contained in q then q'' is also contained in q'.

2012 ACM Subject Classification Information systems → Query languages for non-relational engines; Theory of computation → Parameterized complexity and exact algorithms; Theory of computation → Database query processing and optimization (theory)

Keywords and phrases graph databases, conjunctive regular path queries, semantic optimization, tree-width, containment, approximation

Digital Object Identifier 10.4230/LIPIcs.ICDT.2023.15

Related Version *Full Version*: `https://arxiv.org/abs/2212.01679`

☞ This pdf contains internal links: clicking on a notion leads to its *definition.*[1]

1 Introduction

Graph databases are abstracted as edge-labeled directed graphs $G = \langle V(G), E(G)\rangle$, where nodes of $V(G)$ represent entities and labeled edges $E(G) \subseteq V(G) \times \mathbb{A} \times V(G)$ represent relations between these entities, with $\mathbb{A}$ being a fixed finite alphabet. For instance, Figure 1 depicts a graph database, whose nodes are authors and papers, on the alphabet $\mathbb{A} = \{\text{wrote}, \text{advised}\}$. Edges $x \xrightarrow{\text{wrote}} y$ indicate that the person x wrote the paper y, while edges $x \xrightarrow{\text{advised}} y$ indicate that person x was the Ph.D. advisor of person y.

Being a subclass of relational databases, graph databases can be queried by the predominant query language of *conjunctive queries*, a.k.a. CQs, which consists of the closure under projection of conjunctions of atoms of the form $x \xrightarrow{a} y$ for some letter $a \in \mathbb{A}$. For instance, the conjunctive query

$$\gamma_1(x, y) = x \xrightarrow{\text{wrote}} z \wedge y \xrightarrow{\text{wrote}} z$$

[1] This result was achieved by using the `knowledge` package and its companion tool `knowledge-clustering`.

26th International Conference on Database Theory (ICDT 2023).
Editors: Floris Geerts and Brecht Vandevoort; Article No. 15; pp. 15:1–15:19
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

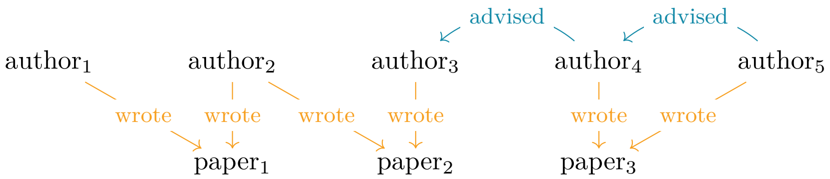


Figure 1 A graph database with eight nodes and eight edges on a two-letter alphabet.

returns, when evaluated on the graph database G defined in Figure 1, all pairs of nodes (u, v) such that u is a co-author of v. Each variable not appearing in the left-hand side of the definition of a conjunctive query (in this example, z) is existentially quantified. Note that every CQ can be seen as a graph database, where each atom is an edge; hence, we sometimes use graph database terminology for CQs.

The expressive power of CQs is somewhat limited, since CQs cannot express, for example, transitive closure. Since the ability to navigate paths is of importance in many graph database scenarios, most modern graph query languages support, as a central querying mechanism, conjunctive regular path queries, or CRPQs for short. CRPQs are defined analogously to conjunctive queries, except that their atoms are now of the form $x \xrightarrow{L} y$ where L is an arbitrary regular language over the alphabet $\mathbb{A}$. For instance the evaluation of the CRPQ $\gamma_2(x, y) = x \xrightarrow{\text{wrote}} z \wedge z' \xrightarrow{\text{wrote}} z \wedge y \xrightarrow{(\text{advised})^*} z'$ on G yields every pair of persons (u, v) such that u is a co-author of a "scientific descendant" of v.

Formally, a *CRPQ* γ is defined as a tuple $\bar{z} = (z_1, \dots, z_n)$ of *output variables*[2] together with a conjunction of *atoms* of the form $\bigwedge_{j=1}^{m} x_j \xrightarrow{L_j} y_j$, where each L_j is a regular language. The set of all variables occurring in γ, namely[3] $\{z_1, \dots, z_n\} \cup \{x_1, y_1, \dots, x_m, y_m\}$, is denoted by $\mathit{vars}(\gamma)$. Given a database G, we say that $(u_1, \dots, u_n)$ *satisfies* γ on G if there is a mapping $f\colon \mathit{vars}(\gamma) \to V(G)$ such that $u_i = f(z_i)$ for all $1 \leqslant i \leqslant n$, and for each $1 \leqslant j \leqslant m$, there exists a path from $f(x_i)$ to $f(y_i)$ in G, labelled by a word from L_i (if the path is empty, the labelled word is ε). The *evaluation* of γ on G is then the set of all tuples that satisfy γ. For example, $(\text{author}_2, \text{author}_5)$ satisfies γ_2 on the graph database G of Figure 1 via the function that maps x to author$_2$, y to author$_5$, z to paper$_2$, and z' to author$_3$.

The language of CRPQ can be extended to navigate edges in both directions. Consider the database $G^{\pm}$ obtained from G by adding, for every edge $x \xrightarrow{a} y$ in G, an extra edge $y \xrightarrow{a^-} x$. We obtain a graph database on the alphabet $\mathbb{A}^{\pm} = \mathbb{A} \dot{\cup} \mathbb{A}^-$ where $\mathbb{A}^- = \{a^- \mid a \in \mathbb{A}\}$. We then define the syntax of a *CRPQ with two-way navigation*, or *C2RPQ*, as a CRPQ on the alphabet $\mathbb{A}^{\pm}$. Its *evaluation* is defined as the evaluation of the CRPQ on $G^{\pm}$. For instance, the evaluation of the C2RPQ $\gamma_3(x, y) = x \xrightarrow{(\text{wrote}\cdot\text{wrote}^-)^*} y$ on the graph database of Figure 1 returns all pairs of individuals linked by a chain of co-authorship. It includes $(\text{author}_1, \text{author}_3)$ or $(\text{author}_1, \text{author}_1)$ but not $(\text{author}_1, \text{author}_4)$. If a query has no output variables we call it *Boolean*, and its evaluation can either be the set $\{()\}$, in which case we say that G satisfies the query, or the empty set $\{\}$. For example, G satisfies $\gamma_4() = x \xrightarrow{\text{wrote}} y$ if, and only if, the database contains one author together with the paper they wrote. We denote the set of atoms of a C2RPQ γ by $\text{Atoms}(\gamma)$, and by $\|\gamma\|$ we denote its number of atoms, i.e., $|\text{Atoms}(\gamma)|$.

[2] For technical reasons (see the definition of expansion) we allow for a variable to appear multiple times.
[3] We neither assume disjointness nor inclusion between $\{z_1, \dots, z_n\}$ and $\{x_1, y_1, \dots, x_m, y_m\}$

Finally, a *union of CQs* (*UCQs*) [resp. *union of CRPQs* (*UCRPQs*), resp. *union of C2RPQs* (*UC2RPQs*)] is defined as a finite set of CQs [resp. CRPQs, resp. C2RPQs], whose tuples of output variables have all the same arity. The evaluation of a union is defined as the union of its evaluations, for instance:

$$\Gamma_5 = \gamma_5^1(x,y) \lor \gamma_5^2(x,y) \text{ where } \gamma_5^1(x,y) = x \xrightarrow{\text{wrote}} y \text{ and } \gamma_5^2(x,y) = x \xleftarrow{\text{advised}} z \land z \xrightarrow{\text{wrote}} y$$

evaluates to the set of pairs (x,y) such that y is a paper written by either x or their advisor. *Infinitary unions* are defined analogously, except that we allow for potentially infinite unions.

For a more detailed introduction to CRPQs, we refer the reader to [10]. For a more general introduction to different query languages for graph databases – including CRPQs – see [6], and for a more practical approach, see [1].

Given two UC2RPQ Γ and Γ', we say that Γ is *contained* in Γ', denoted by $\Gamma \subseteq \Gamma'$ if for every graph database G, for every tuple $\bar{u}$ of G, if $\bar{u}$ satisfies Γ on G, then so does Γ'. The *containment problem* for UC2RPQs is the problem of, given two UC2RPQs Γ and Γ', to decide if $\Gamma \subseteq \Gamma'$. When Γ is contained in Γ' and vice versa, we say that Γ and Γ' are *semantically equivalent*, denoted by $\Gamma \equiv \Gamma'$. The *evaluation problem* for UC2RPQ is the problem of, given a C2RPQ γ, a graph database G and a tuple $\bar{u}$ of elements of G, whether $\bar{u}$ satisfies γ on G.

Queries of small tree-width

It is known that the evaluation problem for UC2RPQ is NP-complete, just as for conjunctive queries [9]. However, queries whose underlying structure looks like a tree – formally, queries of bounded tree-width – can be evaluated in polynomial time.

Tree-width is a measure of how much a graph differs from a tree, introduced by Arnborg and Proskurowski [2]. Formally, a *tree decomposition* of a C2RPQ γ is a pair $(T, \mathbf{v})$ where T is a tree and $\mathbf{v} : V(T) \to \wp(\mathit{vars}(\gamma))$ is a function that associates to each node of T, called *bag*, a set of variables of γ. When $x \in \mathbf{v}(b)$ we shall say that the bag $b \in V(T)$ *contains* the variable x. Further, it must satisfy the following three properties:

- each variable x of γ is contained in at least one bag of T;
- for each atom $x \xrightarrow{L} y$ of γ, there is at least one bag of T that contains both x and y; and
- for each variable x of γ, the set of bags of T containing x is a connected subset of $V(T)$.

The *width* of $(T, \mathbf{v})$ is the maximum of $|\mathbf{v}(b)| - 1$ when b ranges over $V(T)$. The *tree-width* of γ is the minimum of the width of all tree decompositions of γ. We denote by $\mathcal{T}\!w_k$ the set of all C2RPQ of tree-width at most k. The tree-width of a UC2RPQ is simply the maximum of the tree-width of its C2RPQs. An example of tree decomposition of width 2 is given in Figure 3 on Page 12. For a gentle introduction to tree-width, see [14, §3.6].

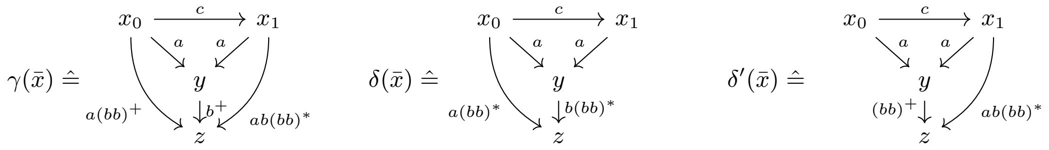
▶ **Proposition 1.1** (Folklore, see e.g. [15, Theorem IV.3]). *For each $k \geqslant 1$, the evaluation problem for UC2RPQs of tree-width at most k is in polynomial time.*

In practice, graph databases tend to be huge and often changing, while queries are in comparison very small. This motivates the following question, given some natural $k \geqslant 1$:

Given a UC2RPQ Γ, is it equivalent to a UC2RPQ Γ' of tree-width at most k?
That is, does it have *semantic tree-width* at most k?

This problem is called the *semantic tree-width k problem*. Should it be decidable in a constructive way – that is, decidable, and if the answer is positive, we can compute a witnessing Γ' from Γ – , then one could, once and for all, compute Γ' from Γ and, whenever one wants to evaluate Γ on a database, evaluate Γ' instead.

▶ **Example 1.2.** Consider the following CRPQs, where $\bar{x} = (x_0, x_1, y, z)$:



The underlying graph of $\gamma(\bar{x})$ being the directed 4-clique, $\gamma(\bar{x})$ has tree-width 3. We claim that $\gamma(\bar{x})$ is equivalent to the UCRPQ $\delta(\bar{x}) \vee \delta'(\bar{x})$, and hence has semantic tree-width 2.

Indeed, given a graph database satisfying $\gamma(\bar{x})$ via some mapping μ, it suffices to make a case disjunction on whether the number of b-labelled atoms in the path from $\mu(y)$ to $\mu(z)$ is even or odd. In the first case, the atom $x_0 \xrightarrow{a(bb)^+} z$ becomes redundant since we can deduce the existence of such a path from the conjunction $x \xrightarrow{a} y \xrightarrow{(bb)^+} z$, and hence the database satisfies $\delta(\bar{x})$ via μ. Symmetrically, in the second case, the atom $x_1 \xrightarrow{b(bb)^*} z$ becomes redundant, and the database satisfies $\delta'(\bar{x})$ via μ. Thus, $\gamma(\bar{x})$ is contained, and hence equivalent (the other containment being trivial), to the UCRPQ $\delta(\bar{x}) \vee \delta'(\bar{x})$ of tree-width 2.

For conjunctive queries, the semantic tree-width k problem can be effectively decided quite easily – in fact, CQs enjoy the effective existence of unique minimal queries [9, Theorem 12] which happen to also minimize the tree-width. For CRPQs and UC2RPQs, the question is far more challenging, and it has only been solved for the case $k = 1$ by Barceló, Romero, and Vardi [5, Theorem 6.1]. We solve the problem for every other $k > 1$, left open in [5, §7] [15, §VI-(3)]:

▶ **Theorem 1.3.** *For each $k \geqslant 1$, the semantic tree-width k problem is decidable. Moreover, it lies in* 2ExpSpace *and is* ExpSpace*-hard.*

Amusingly, our proof for $k > 1$ cannot be stretched to capture the case $k = 1$: the two approaches seem to be intrinsically incompatible.

▶ Remark 1.4. To simplify proofs, we assume that all the regular languages are described via non-deterministic finite automata (NFA) instead of regular expressions, which does not affect any of our complexity bounds. However, for readability all our examples will be given in terms of regular expressions.

Moreover, we also show that for any class $\mathcal{L}$ of regular languages over $\mathbb{A}^{\pm}$ satisfying some mild hypothesis ("closure under sublanguages"), if $\Gamma \in \text{UC2RPQ}(\mathcal{L})$ has semantic tree-width $k > 1$, then Γ is equivalent to a UC2RPQ($\mathcal{L}$) of tree-width at most k, where UC2RPQ($\mathcal{L}$) denotes the class of all UC2RPQs whose atoms are all labelled by languages from $\mathcal{L}$. In other words, if a query can be defined with labels in $\mathcal{L}$, and if this query is equivalent to a query of small tree-width, then it is also equivalent to a query of small tree-width with labels in $\mathcal{L}$.

For a NFA $\mathcal{A}$ and two states q, q' thereof, we denote by $\mathcal{A}[q, q']$ the *sublanguage* of $\mathcal{A}$ recognized when considering q as initial state and $\{q'\}$ as set of final states. We say that $\mathcal{L}$ is *closed under sublanguages* if (i) it contains every language of the form $\{a\}$, where $a \in \mathbb{A}$ is any (positive) letter such that either a or a^- occur in a word of a language of $\mathcal{L}$, and (ii) for every language $L \in \mathcal{L}$ there exists a NFA $\mathcal{A}_L$ such that every sublanguage $\mathcal{A}_L[q, q']$ distinct from $\varnothing$ and $\{\varepsilon\}$ belongs to $\mathcal{L}$.

To the best of our knowledge, all classes of regular expressions that have been considered in the realm of regular path queries (see, e.g., [11, §1]) are closed under sublanguages. In particular, this is the case for the class $\big\{\{a_1 + \ldots + a_n\} \mid a_1, \ldots, a_n \in \mathbb{A}\big\} \cup \big\{a^* \mid a \in \mathbb{A}\big\}$, which will be our focus of study in Section 6. Moreover, even if some class $\mathcal{L}$ is not closed under sublanguages, such as for example $\{(aa)^*\}$, then it is contained is a class closed under sublanguages – $\{a, a(aa)^*, (aa)^*\}$ in this example – , whose size[4] is polynomial in the size of the original class. See the full version for more examples.

▶ **Theorem 1.5.** *Assume that $\mathcal{L}$ is closed under sublanguages. For any query $\Gamma \in$ UC2RPQ($\mathcal{L}$) and $k > 1$, the following are equivalent:*

1. *Γ is equivalent to an infinitary union of conjunctive queries of tree-width at most k;*
2. *Γ has semantic tree-width at most k;*
3. *Γ is equivalent to a* UC2RPQ($\mathcal{L}$) *of tree-width at most k.*

The implications (3) ⇒ (2) ⇒ (1) immediately follow from the definition of the semantic tree-width. On the other hand, the implications (1) ⇒ (2) and (2) ⇒ (3) are surprising, since they are both trivially false when $k = 1$. We defer the proof of this claim to Section 2 (see Remark 2.5) as we first need a few tools to manipulate CRPQs.

The proofs of both Theorems 1.3 and 1.5 rely on our key lemma (Lemma 3.8), which states essentially that every UC2RPQ has a computable "maximal under approximation" by a UC2RPQ of tree-width k. Formally, the key lemma has the following corollary:

▶ **Corollary 3.9.** *For each $k > 1$ and for each class $\mathcal{L}$ closed under sublanguages, for each query $\Gamma \in$ UC2RPQ($\mathcal{L}$), there exists $\Gamma' \in$ UC2RPQ($\mathcal{L}$) of tree-width k such that $\Gamma' \subseteqq \Gamma$, and for every $\Delta \in$ UC2RPQ, if Δ has tree-width k and $\Delta \subseteqq \Gamma$, then $\Delta \subseteqq \Gamma'$. Moreover, Γ' is computable from Γ in* ExpSpace*.*

The proof of our key lemma spans over Sections 3–5: in Section 3, we introduce necessary notions to formally state it, and deduce Theorems 1.3 and 1.5 from it; in Section 4 we introduce the central notion of tagged tree decompositions of C2RPQs homomorphisms, and building on it, we finally describe the constructions used to prove the key lemma in Section 5.

Finally, in Section 6, given the high complexity of semantic tree-width k problem, we focus on the case of CRPQs using some simple regular expressions (SRE), and show that the complexity of this problem is much lower:

▶ **Theorem 6.1.** *For $k > 1$, the semantic tree-width k problem for* UCRPQ(SRE) *is in Π_2^p.*

A discussion on differences with Barceló, Romero and Vardi's contributions and open questions are left for Section 7.

2 Homomorphisms, refinements, and expansions

Before attacking the statement of our key lemma in Section 3, we first give a few elementary definitions on C2RPQs in this section. A *homomorphism* f from a C2RPQ $\gamma(x_1, \dots, x_m)$ to a C2RPQ $\gamma'(y_1, \dots, y_m)$ is a mapping from *vars*(γ) to *vars*(γ') such that $f(x) \xrightarrow{L} f(y)$ is an atom of γ' for every atom $x \xrightarrow{L} y$ of γ, and further $f(x_i) = y_i$ for every i. Such a homomorphism h is *strong onto* if for every atom $x' \xrightarrow{L} y'$ of γ' there is an atom $x \xrightarrow{L} y$ of γ such that $f(x) = x'$ and $f(y) = y'$. We write $\gamma \xrightarrow{hom} \gamma'$ if there is a homomorphism from γ to γ', and $\gamma \xrightarrow{hom}\!\!\!\!\!\rightarrow \gamma'$ if there is a strong onto homomorphism. It is easy to see that if $\gamma \xrightarrow{hom} \gamma'$ then $\gamma' \subseteqq \gamma$, and in the case where γ, γ' are CQs this is an "if and only if" [9, Lemma 13].

[4] Defined as the sum of the number of states of the minimal automaton of the languages of $\mathcal{L}$.

Some intuitions on maximal under-approximations. Given a conjunctive query γ, the union of all conjunctive queries that are contained in γ is semantically equivalent to the union $\bigvee\{\gamma' \mid \gamma \xrightarrow{hom}\!\!\!\twoheadrightarrow \gamma'\}$. Naturally, this statement borders on the trivial since γ' belongs to this union. It becomes interesting when we add a restriction: given a class $\mathcal{C}$ of CQs (to which γ may not belong) closed under subqueries, then $\bigvee\{\gamma' \in \mathcal{C} \mid \gamma \xrightarrow{hom}\!\!\!\twoheadrightarrow \gamma'\}$ is the maximal under-approximation[5] of γ by finite unions of conjunctive queries of $\mathcal{C}$[6]. As a consequence, we deduce that for each $k \geqslant 1$, the maximal under-approximation of a CQ by a finite union of CQs of tree-width at most k is computable, and hence we can effectively decide if some CQ is equivalent to a query of tree-width at most k. For more details on approximations of CQs, see [3].

Unfortunately, these results cannot be straightforwardly extended to conjunctive regular path queries: intuitively, taking homomorphic images can be understood as "simplifying" the query (we reduce its number of variables, but this may make the query strictly contained in the original one). This is because CRPQs have an implicit quantification of variables: for instance, the CQ $\gamma(x,y) = x \xrightarrow{a} z \xrightarrow{b} y$ can be rewritten as the CRPQ $\gamma'(x,y) = x \xrightarrow{ab} y$. Coming back to our previous Example 1.2, another way of seeing that $\delta(\bar{x}) \subsetneqq \gamma(\bar{x})$ is by observing that we can obtain $\delta(\bar{x})$ as the result of the following two operations:

- in $\gamma(\bar{x})$, replace the atom $x_1 \xrightarrow{ab(bb)^*} z$ with $x_1 \xrightarrow{a} t \xrightarrow{b(bb)^*} z$, where t is a fresh existentially quantified variable;
- identify variables t with y.[7]

The last operation consists in taking homomorphic images, and the first one amounts to making explicit an implicit quantification. We formalize the first operation by introducing the notion of "refinement" which we will later use, in Section 3, to introduce the notion of maximal under-approximations for CRPQs. We will come back this example once all these notions will have been formally defined (cf. Example 3.4).

Refinements. An *atom m-refinement* of a C2RPQ atom $\gamma(x,y) = x \xrightarrow{L} y$ where L is given by the NFA $\mathcal{A}_L$ is any C2RPQ of the form

$$\rho(x,y) = x \xrightarrow{L_1} t_1 \xrightarrow{L_2} \dots \xrightarrow{L_{n-1}} t_{n-1} \xrightarrow{L_n} y \tag{1}$$

where $1 \leqslant n \leqslant m$, $t_1, \dots, t_{n-1}$ are fresh (existentially quantified) variables, and $L_1, \dots, L_n$ are such that there exists a sequence $(q_0, \dots, q_n)$ of states of $\mathcal{A}_L$ such that q_0 is initial, q_n is final, and for each i, L_i is either of the form (i) $\mathcal{A}_L[q_i, q_{i+1}]$, or (ii) $\{a\}$ if the letter $a \in \mathbb{A}$ belongs to $\mathcal{A}_L[q_i, q_{i+1}]$, or (iii) $\{a^{-1}\}$ if $a^{-1} \in \mathbb{A}^-$ belongs to $\mathcal{A}[q_i, q_{i+1}]$. Additionally, if $\varepsilon \in L$, the equality atom "$x = y$" is also an *atom m-refinement* (see the full version for more details on these), Thus, an *atom m-refinement* can be either of the form (1) or "$x = y$". By convention, $t \xrightarrow{a^-} t'$ is a shorthand for $t' \xrightarrow{a} t$. As a consequence, the underlying graph of an atom m-refinement of the form (1) is not necessarily a directed path. By definition, note that $L_1 \cdots L_n \subseteq L$ and hence $\rho \subsetneqq \gamma$ for any atom m-refinement ρ of γ. An *atom refinement* is an atom m-refinement for some m.

[5] A formalization of what "maximal under-approximation" means is given in Remark 3.2.

[6] The proof is straightforward: by definition, this union is finite (a finite CQ has only finitely many homomorphic images), and is contained in γ. Moreover, if $\gamma' \in \mathcal{C}$ is contained in γ, then there exists a homomorphism $f\colon \gamma \to \gamma'$. Then $\gamma \xrightarrow{hom}\!\!\!\twoheadrightarrow f(\gamma)$ and $f(\gamma) \in \mathcal{C}$ since it is a subquery of $\gamma' \in \mathcal{C}$ and $\mathcal{C}$ is closed under subqueries. We conclude by noting that γ' is contained in $f(\gamma)$.

[7] We actually obtain two atoms from y to z: one label by b^+ and one by $b(bb)^*$, but since $b(bb)^* \subseteq b^+$ we can discard the b^+ atom preserving the query semantics.

▶ **Definition 2.1.** *Given an atom refinement $\rho = x \xrightarrow{L_1} t_1 \xrightarrow{L_2} \ldots \xrightarrow{L_{n-1}} t_{n-1} \xrightarrow{L_n} y$ of $\gamma = x \xrightarrow{L} y$ as in* (1)*, define a* contraction *of ρ between t_i and t_j, where $0 \leqslant i, j \leqslant n$ and $j > i + 1$, is any C2RPQ of the form:*

$$\rho' = x \xrightarrow{L_1} t_1 \xrightarrow{L_2} \ldots \xrightarrow{L_i} \boldsymbol{t_i} \xrightarrow{\boldsymbol{K}} \boldsymbol{t_j} \xrightarrow{L_{j+1}} \ldots \xrightarrow{L_{n-1}} t_{n-1} \xrightarrow{L_n} y$$

such that $K = \mathcal{A}[q_i, q_j]$. Then every contraction ρ' of ρ is a refinement of γ, and $\rho \subseteqq \rho' \subseteqq \gamma$. Informally, we will abuse the notation and write $[L_i \cdots L_j]$ to denote the language K – even if this language does not only depend on $L_i \cdots L_j$.

▶ **Example 2.2.** Let $\gamma(x, y) = x \xrightarrow{(aa^-)^*} y$ be a C2RPQ atom, where $(aa^-)^*$ is implicitly represented by its minimal automaton. Then $\rho(x, y)$ is a refinement of refinement length seven of $\gamma(x, y)$ and $\rho'(x, y)$ is a contraction of $\rho(x, y)$, where:

$$\begin{aligned}\rho(x, y) &= x \xrightarrow{a} t_1 \xrightarrow{(a^-a)^*} t_2 \xrightarrow{(a^-a)^*} t_3 \xleftarrow{a} t_4 \xrightarrow{(aa^-)^*} t_5 \xrightarrow{(aa^-)^*a} t_6 \xleftarrow{a} y,\\ \rho'(x, y) &= x \xrightarrow{a} t_1 \xrightarrow{(a^-a)^*a^-} t_4 \xrightarrow{(aa^-)^*} y.\end{aligned}$$

On the other hand, $\rho''(x, y) = x \xrightarrow{a} t_1 \xleftarrow{a} y$ is not a contraction of $\rho(x, y)$.

An *m-refinement* of a C2RPQ $\gamma(\bar{x}) = \bigwedge_i x_i \xrightarrow{L_i} y_i$ is any query resulting from: 1) replacing every atom by one of its m-refinements, and 2) should some m-refinements be equality atoms, collapsing equal variables and getting rid of equalities, in a rather standard way – more details are given in the full version. A *refinement* is an m-refinement for some m. Note that any atom m-refinements is, by definition, also an atom m'-refinements when $m < m'$: as a consequence, in the refinement of a C2RPQ the atom refinements need not have the same length. For instance, both $\rho(x, x) = x \xrightarrow{c} x$ and $\rho'(x, y) = x \xrightarrow{a} t_1 \xrightarrow{a} y \xleftarrow{c} y$ are refinements of $\gamma(x, y) = x \xrightarrow{a^*} y \xleftarrow{c} x$. For a given C2RPQ γ, let $\mathrm{Ref}^{\leqslant m}(\gamma)$ be the set of all m-refinements of γ, and $\mathrm{Ref}(\gamma)$ be the set of all its refinements. Given a refinement $\rho(\bar{x})$ of $\gamma(\bar{x})$, its *refinement length* is the least integer m such that $\rho(\bar{x}) \in \mathrm{Ref}^{\leqslant m}(\gamma)$. Note that if the automaton representing a language L has more than one final state, for instance the minimal automaton for $L = a^+ + b^+$, then $x \xrightarrow{L} y$ is not a refinement of itself. However, it will always be equivalent to a union of refinements: in this example, $x \xrightarrow{a^+ + b^+} y$ is equivalent to the union of $x \xrightarrow{a^+} y$ and $x \xrightarrow{b^+} y$, which are both refinements of the original C2RPQ.

Expansions. Remember that a C2RPQ whose languages are $\{a\}$ or $\{a^-\}$ for $a \in \mathbb{A}$ is in effect a CQ. The *expansions* of a C2RPQ γ is the set $\mathrm{Exp}(\gamma)$ of all CQs which are refinements of γ. In other words, an expansion of γ is any CQ obtained from γ by replacing each atom $x \xrightarrow{L} y$ by a path $x \xrightarrow{w} y$ for some word $w \in L$. For instance, $\xi(x, y) = x \xrightarrow{a} t_1 \xleftarrow{a} t_2 \xrightarrow{a} t_3 \xleftarrow{a} y$ is an expansion of $\rho(x, y) = x \xrightarrow{(aa^-)^*} y$.

Any C2RPQ is equivalent to the infinitary union of its expansions. In light of this, the semantics for UC2RPQ can be rephrased as follows. Given a UC2RPQ Γ and a graph database G, the evaluation of Γ over G, denoted by $\Gamma(G)$, is the set of tuples $\bar{v}$ of nodes for which there is $\xi \in \mathrm{Exp}(\Gamma)$ such that $\xi \xrightarrow{hom} (G, \bar{v})$. Similarly, containment of UC2RPQs can also be characterized in terms of expansions:

▶ **Proposition 2.3** (Folklore, see e.g. [12, Proposition 3.2] or [8, Theorem 2])**.** *Let Γ_1 and Γ_2 be UC2RPQs. Then the following are equivalent*

- *$\Gamma_1 \subseteq \Gamma_2$;*
- *for every $\xi_1 \in \mathrm{Exp}(\Gamma_1)$, $\xi_1 \subseteqq \Gamma_2$;*
- *for every $\xi_1 \in \mathrm{Exp}(\Gamma_1)$ there is $\xi_2 \in \mathrm{Exp}(\Gamma_2)$ such that $\xi_2 \xrightarrow{hom} \xi_1$.*

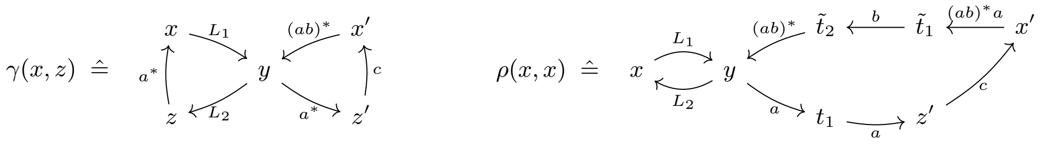


Figure 2 On the left-hand side, a CRPQ γ of tree-width 2. On the right-hand side, a refinement ρ of γ, whose refinement length is three, which is also of tree-width 2.

Hence, every expansion of γ is also a refinement of γ. Moreover, if ρ is a refinement of γ, then $\rho \subseteq \gamma$, and γ is semantically equivalent to the infinitary union of all its refinements, and to the infinitary union of all its expansions.

Our approach to proving Theorems 1.3 and 1.5 and the key lemma heavily rely on refinements. One crucial property that these objects satisfy is that they preserve tree-width k, unless $k = 1$. This is the main reason why our approach cannot capture the case $k = 1$, solved by Barceló, Romero and Vardi [5].

▷ Fact 2.4. Let $k > 1$ and let γ be a C2RPQ of tree-width at most k. Then any refinement of γ has tree-width at most k.

This fact is illustrated on a example in Figure 2.

Proof sketch. The underlying graph of a refinement of γ is obtained from the underlying graph of γ by either contracting some edges (when dealing with equality atoms), or by replacing a single edge by a path of edges (where the non-extremal nodes are new nodes). Both operations preserve the property "having tree-width at most k" when $k > 1$. Details are given in the full version. ◁

For $k = 1$, the property fails: for instance the CRPQ $\gamma(x) = x \xrightarrow{a^*} x$ has tree-width at most 1 (in fact it has tree-width 0), but its refinement $\rho(x) = x \xrightarrow{a^*} t_1 \xrightarrow{a^*} t_2 \xrightarrow{a^*} x$ has tree-width two.

Before introducing maximal under-approximations, we can now show that the statement of Theorem 1.5 is indeed false for $k = 1$.

▶ Remark 2.5. (1) $\not\Rightarrow$ (2) when $k = 1$: consider the CRPQ $\gamma(x, y) = x \xrightarrow{a^*} y \wedge y \xrightarrow{b} x$ of tree-width 1, and hence of semantic tree-width 1, and observe that it is not equivalent to any infinitary union of conjunctive queries of tree-width 1 – this can be proven by considering, for example, the expansion $x \xrightarrow{a} z \xrightarrow{a} y \wedge y \xrightarrow{b} x$ of $\gamma(x, y)$ and applying Proposition 2.3.

(2) $\not\Rightarrow$ (3) when $k = 1$: By [5, Proposition 6.4] the CRPQ of semantic tree-width 1 $\gamma(x) \mathrel{\hat{=}} x \xleftarrow{a} z \xrightarrow{a} y \wedge x \xrightarrow{b} y \equiv x \xrightarrow{ba^- a} x$ is not equivalent to any UCRPQ of tree-width 1. Hence, the implication is false when $\mathcal{L}$ is the class of regular languages over $\mathbb{A}^{\pm}$ that do not use any letter of the form a^-.

3 Maximal under-approximations of bounded tree-width

In this section, we state our key technical result, Lemma 3.8. Essentially, we follow the same structure as Theorem 1.5: given a C2RPQ γ and an integer $k > 1$, we start by consider its maximal under-approximation by infinitary unions of conjunctive queries of tree-width k (Definition 3.1), and then show that this query can in fact be expressed as a UC2RPQ of tree-width k whose atoms are obtained by taking sublanguages from γ (Lemma 3.8).

For the definitions of this section, let us fix any class $\mathcal{C}$ of C2RPQ queries.

▶ **Definition 3.1** (Maximal under-approximation). *Let γ be a C2RPQ. The* maximal under-approximation *of γ by infinitary unions of $\mathcal{C}$-queries, is* $\mathrm{App}_{\mathcal{C}}(\gamma) \mathrel{\hat{=}} \{\alpha \in \mathcal{C} \mid \alpha \subseteqq \gamma\}$.

For intuition, we refer the reader to paragraph "An intuition on maximal under-approximations" at the beginning of Section 2.

▶ Remark 3.2. Observe that $\mathrm{App}_{\mathcal{C}}(\gamma)$ is an infinitary union of $\mathcal{C}$-queries, that $\mathrm{App}_{\mathcal{C}}(\gamma) \subseteqq \gamma$, and that for every infinitary union of $\mathcal{C}$-queries Δ, if $\Delta \subseteqq \gamma$, then $\Delta \subseteqq \mathrm{App}_{\mathcal{C}}(\gamma)$ (i.e., it is the unique maximal under-approximation). Similarly, the maximal under-approximation of a UC2RPQ is simply the union of the maximal under-approximations of the C2RPQs thereof.

Unfortunately, the fact that a query α is part of this union, $\alpha \in \mathrm{App}_{\mathcal{C}}(\gamma)$, does not yield any useful information on the shape of α – we merely know that $\alpha \subseteqq \gamma$. The rest of this subsection is dedicated to introducing another infinitary union of $\mathcal{C}$-queries, namely $\mathrm{App}^{\star}_{\mathcal{C}}(\gamma) \subseteq \mathrm{App}_{\mathcal{C}}(\gamma)$, in which queries $\alpha \in \mathrm{App}^{\star}_{\mathcal{C}}(\gamma)$ come together with a witness – a homomorphism – of their containment in γ.

▶ **Definition 3.3.** *The maximal under-approximation of γ by infinitary unions of homomorphically-smaller $\mathcal{C}$-queries is*

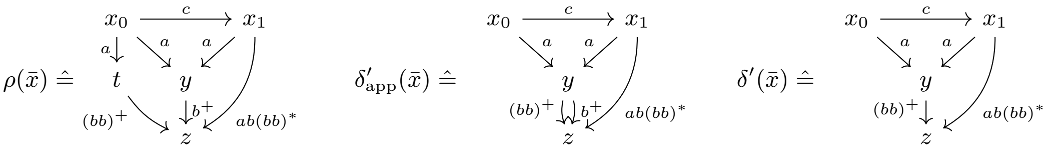
$$\mathrm{App}^{\star}_{\mathcal{C}}(\gamma) \mathrel{\hat{=}} \{\alpha \in \mathcal{C} \mid \exists \rho \in \mathrm{Ref}(\gamma), \exists f\colon \rho \xrightarrow{hom} \!\!\!\!\rightarrow \alpha\}. \tag{2}$$

For instance, let $\mathcal{C}$ be the class of all CRPQs and let $\gamma(x,y) \mathrel{\hat{=}} x \xrightarrow{a^*} y \xrightarrow{a} z \xrightarrow{a^*} x \wedge y \xrightarrow{a^*} x$ be the CRPQ which asks for all pairs of nodes that belong to the same non-empty cycle of a's[8]. Then $\gamma'(x,y) = x \xrightarrow{a^*} y \xrightarrow{a} z \xrightarrow{a^*} x$ is an element of $\mathrm{App}^{\star}_{\mathcal{C}}(\gamma)$ since there is a strong onto homomorphism from the refinement

$$\rho(x,y) = \big(x \xrightarrow{a^*} y \xrightarrow{a} z \xrightarrow{a^*} x \wedge y \xrightarrow{a} z' \xrightarrow{a^*} x\big) \in \mathrm{Ref}(\gamma(x,y))$$

to $\gamma'(x,y)$, mapping z and z' to z.

▶ **Example 3.4** (Example 1.2, cont'd). Both $\delta(\bar{x})$ and $\delta'(\bar{x})$ are semantically equivalent to queries in $\mathrm{App}^{\star}_{\mathcal{T}\!w_2}(\gamma(\bar{x}))$. Indeed, starting from $\gamma(\bar{x})$, we can refine $x_0 \xrightarrow{a(bb)^+} z$ into $x_1 \xrightarrow{a} t \xrightarrow{(bb)^+} z$. Denote by $\rho(\bar{x})$ the query obtained. Then merge variables t and y: this new query $\delta'_{\mathrm{app}}(\bar{x})$ is equivalent to $\delta'(\bar{x})$.



Clearly, $\mathrm{App}^{\star}_{\mathcal{C}}(\gamma)$ – whose queries are informally called *approximations* – is included, and thus semantically contained, in $\mathrm{App}_{\mathcal{C}}(\gamma)$, since $\rho \subseteqq \gamma$ and $\alpha \subseteqq \rho$ in (2). In fact, under some assumptions on $\mathcal{C}$, the converse containment also holds.

▶ **Observation 3.5.** *If $\mathcal{C}$ is closed under expansions, then for any C2RPQ γ, we have* $\mathrm{App}_{\mathcal{C}}(\gamma) \equiv \mathrm{App}^{\star}_{\mathcal{C}}(\gamma)$.

Proof. This follows immediately from Proposition 2.3 – note that in the definition of $\mathrm{App}^{\star}_{\mathcal{C}}(\gamma)$ we work with strong onto homomorphisms, but we can always restrict homomorphisms to their image to make them strong onto, without changing the expressiveness of the query. ◀

[8] There exists CRPQs with fewer variables that expresses the same property, but this is irrelevant here.

Observe then, by Fact 2.4, that the class $\mathcal{T}\!w_k$ of all C2RPQs of tree-width at most k is closed under refinements and hence under expansions, provided that k is greater or equal to 2. As an immediate consequence, we have:

▶ **Corollary 3.6.** *For $k \geqslant 2$, for all C2RPQ γ, $\mathrm{App}_{\mathcal{T}\!w_k}(\gamma) \equiv \mathrm{App}^{\star}_{\mathcal{T}\!w_k}(\gamma)$.*

▶ **Example 3.7** (Counter-example for $k = 1$)**.** Consider the followings queries:

$$\gamma(x,y) \ \hat{=}\ \begin{tikzcd} z \arrow[r, "(ba)^*"] & z' \arrow[d, "b"] \\ x \arrow[u, "a"] \arrow[r, bend left, "(ab)^+"] & y \arrow[l, bend left, "c"] \end{tikzcd} \qquad \text{and} \qquad \delta(x,y) \ \hat{=}\ \begin{tikzcd} x \arrow[r, bend left, "(ab)^+"] & y \arrow[l, bend left, "c"] \end{tikzcd}.$$

We claim that $\mathrm{App}_{\mathcal{T}\!w_1}(\gamma) \not\subseteq \mathrm{App}^{\star}_{\mathcal{T}\!w_1}(\gamma)$ since $\delta(x,y) \in \mathrm{App}_{\mathcal{T}\!w_1}(\gamma)$ but $\delta(x,y) \not\subseteq \mathrm{App}^{\star}_{\mathcal{T}\!w_1}(\gamma)$. Details are given in the full version.

By definition, $\mathrm{App}^{\star}_{\mathcal{T}\!w_k}(\gamma)$ is an infinitary union of C2RPQs. We show that, in fact, $\mathrm{App}^{\star}_{\mathcal{T}\!w_k}(\gamma)$ is always equivalent to a *finite* union of C2RPQs. This is done by bounding the length of the refinements occurring in the definition of $\mathrm{App}^{\star}_{\mathcal{T}\!w_k}(\gamma)$. For a natural m, let $\mathrm{App}^{\star,\leqslant m}_{\mathcal{C}}(\Gamma) \ \hat{=}\ \{\alpha \in \mathcal{C} \mid \exists \rho \in \mathrm{Ref}^{\leqslant m}(\gamma),\ \exists f \colon \rho \xrightarrow{hom} \alpha\}$. Our main technical lemma is then the following:

▶ **Lemma 3.8** (*Key lemma*)**.** *For $k \geqslant 1$ and C2RPQ γ, we have $\mathrm{App}_{\mathcal{T}\!w_k}(\gamma) \equiv \mathrm{App}^{\star,\leqslant \ell}_{\mathcal{T}\!w_k}(\gamma)$, where $\ell = \Theta(\|\gamma\|^2 \cdot (k+1)^{\|\gamma\|+1})$.*

▶ **Corollary 3.9.** *For each $k > 1$ and for each class $\mathcal{L}$ closed under sublanguages, for each query $\Gamma \in \mathrm{UC2RPQ}(\mathcal{L})$, there exists $\Gamma' \in \mathrm{UC2RPQ}(\mathcal{L})$ of tree-width k such that $\Gamma' \subseteqq \Gamma$, and for every $\Delta \in \mathrm{UC2RPQ}$, if Δ has tree-width k and $\Delta \subseteqq \Gamma$, then $\Delta \subseteqq \Gamma'$. Moreover, Γ' is computable from Γ in* ExpSpace.

Using Lemma 3.8 as a black box – which will be proven in Section 5 – , we can now give a proof of Theorems 1.3 and 1.5. The upper bound of Theorem 1.3 follows directly from Corollary 3.9: to test whether a query Γ is of semantic tree-width k, it suffices to test the containment $\Gamma \subseteqq \Gamma'$, where Γ' is the maximal under-approximation given by Corollary 3.9. The containment problem being in ExpSpace [12, 8], we obtain:

▶ **Lemma 3.10.** *For $k \geqslant 1$, the semantic tree-width k problem for UC2RPQ is in* 2ExpSpace.

An ExpSpace lower bound follows by a straightforward adaptation from the ExpSpace lower bound for the case $k = 1$ [5, Proposition 6.2].

▶ **Lemma 3.11.** *The semantic tree-width k problem is* ExpSpace*-hard, even if restricted to Boolean CRPQs.*

We can now rely on the equivalence $\mathrm{App}_{\mathcal{T}\!w_k}(\gamma) \equiv \mathrm{App}^{\star,\leqslant \ell}_{\mathcal{T}\!w_k}(\gamma)$ to prove Theorem 1.5.

Proof of Theorem 1.5. The implications (3) ⇒ (2) ⇒ (1) are straightforward: they follow directly from Fact 2.4. For (1) ⇒ (3), note that (1) implies that $\Gamma \equiv \mathrm{App}_{\mathcal{T}\!w_k}(\Gamma)$, and by Lemma 3.8, $\mathrm{App}_{\mathcal{T}\!w_k}(\Gamma) \equiv \Delta \ \hat{=}\ \bigvee_{\gamma \in \Gamma} \mathrm{App}^{\star,\leqslant \ell_\gamma}_{\mathcal{T}\!w_k}(\gamma)$, so Γ is equivalent to the latter. Since queries of Δ are obtained as homomorphic images of refinements of Γ, all of which are labelled by sublanguages of $\mathcal{L}$, and since $\mathcal{L}$ is closed under sublanguages, it follows that hence Γ is equivalent to a $\mathrm{UC2RPQ}(\mathcal{L})$ of tree-width k. ◀

We are left with the proof of Lemma 3.8, which will take the next two sections. Since $\mathrm{App}_{\mathcal{T}w_k}(\gamma) \equiv \mathrm{App}^{\star}_{\mathcal{T}w_k}(\gamma)$ by Corollary 3.6 and since $\mathrm{App}^{\star,\leqslant\ell}_{\mathcal{T}w_k}(\gamma)$ is a subset of $\mathrm{App}_{\mathcal{T}w_k}(\gamma)$, we only need to show that $\mathrm{App}^{\star}_{\mathcal{T}w_k}(\gamma) \subseteqq \mathrm{App}^{\star,\leqslant\ell}_{\mathcal{T}w_k}(\gamma)$. Formally, this means that for all $\alpha \in \mathcal{T}w_k$, if there exists $\rho \in \mathrm{Ref}(\gamma)$ and $f\colon \rho \xrightarrow{hom}\!\!\!\!\rightarrow \alpha$, then there exists $\alpha' \in \mathcal{T}w_k$ such that $\alpha \subseteqq \alpha'$ and there exists $\rho' \in \mathrm{Ref}^{\leqslant\ell}(\gamma)$ and $f'\colon \rho' \xrightarrow{hom}\!\!\!\!\rightarrow \alpha'$. We prove this by "massaging" the homomorphism $f\colon \rho \xrightarrow{hom}\!\!\!\!\rightarrow \alpha$, by looking where each atom refinement of γ is sent on α relative to a "well-behaved" tree decomposition of α of width k. Next, we introduce the notion of tagged tree decomposition to have a precise handle on this information.

4 Intermezzo: tagged tree decompositions

▶ **Definition 4.1.** *Let $f\colon \rho \to \alpha$ be a homomorphism between two C2RPQs. A* tagged tree decomposition *of f is a triple $(T, \mathbf{v}, \mathbf{t})$ where $(T, \mathbf{v})$ is a tree decomposition of α, and $\mathbf{t}$ is a mapping $\mathbf{t}\colon \mathrm{Atoms}(\gamma) \to V(T)$, called* tagging, *such that $\mathbf{v}(\mathbf{t}(e))$ contains both $f(x)$ and $f(y)$ for each atom $e = x \xrightarrow{\lambda} y \in \mathrm{Atoms}(\gamma)$.*

In other words, $\mathbf{t}$ gives, for each atom of γ, a witnessing bag that contains it, in the sense that it contains the image by f of the atom source and target. By definition, given a tree decomposition $(T, \mathbf{v})$ of α and a homomorphism $f\colon \rho \to \alpha$, there is always one way (usually many) of extending $(T, \mathbf{v})$ into a tagged tree decomposition of f.

▷ Fact 4.2. Let $(T, \mathbf{v}, \mathbf{t})$ be a tagged tree decomposition of some homomorphism $f\colon \rho \to \alpha$. Let T' be the smallest connected subset of T induced by the image of $\mathbf{t}$. Then $(T', \mathbf{v}|_{T'}, \mathbf{t})$ is still a tagged tree decomposition of f, whose width is at most the width of $(T, \mathbf{v}, \mathbf{t})$.

We extend the notion of tagging to paths: a formal definition can be found in the full version, and the notion is illustrated in Figure 3. In the context of a (nice) tagged tree decomposition $(T, \mathbf{v}, \mathbf{t})$ of $f\colon \rho \xrightarrow{hom} \alpha$, given a path π of ρ, say $x_0 \xrightarrow{\lambda_1} x_1 \xrightarrow{\lambda_2} \cdots \xrightarrow{\lambda_n} x_n$ (in blue in Figure 3a), the *path induced* by π, denoted $\mathbf{t}[\pi]$, is informally defined as the following "path" in $T \times \alpha$, seen as a sequence of pairs from $V(T) \times \mathit{vars}(\alpha)$:

- it starts with the bag $\mathbf{t}(x_0 \xrightarrow{\lambda_1} x_1)$ of T and the variable $f(x_0)$ of α; and it continues with $(\mathbf{t}(x_0 \xrightarrow{\lambda_1} x_1), f(x_1))$ (corresponding to the first blue edge in b_1 of Figure 3a);
- it then follows the shortest path in T (unique, since it is a tree) that goes to the bag $\mathbf{t}(x_1 \xrightarrow{\lambda_2} x_2)$, while staying in $f(x_1)$ in α (in Figure 3a, it follows the blue path: $(b_2, x_2), (b_3, x_2))$ and it traverses the atom $x_1 \xrightarrow{\lambda_2} x_2$ (i.e., we go to (b_3, x_3));
- it continues in the same way for all other atoms of the path, ending up with the bag $\mathbf{t}(x_{n-1} \xrightarrow{\lambda_n} x_n)$ and the variable $f(x_n)$ of α.

By construction, note that the constructed sequence $(b_i, z_i)_i$ is such that $z_i \in \mathbf{v}(b_i)$. Moreover, given a bag b of T and a variable z of α, we say that $\mathbf{t}[\pi]$ *leaves* b at z when $z = z_i$ and $b = b_i$ for some i, and either b_{i+1} is undefined, or distinct from b. For example, in Figure 3a, $\tau[\pi]$ leaves b_1 at x_1 and leaves b_2 at x_1 and at x_4. We say that an induced path is *cyclic* if it contains three positions $i < j < i'$ such that i and i' contain the same bag but j contains a different bag (the one in Figure 3a is cyclic because it passes twice by the bag b_2).

For technical reasons – the proof of Claim 5.2 – , we need to use the classical notion of *nice tree decomposition* (see e.g. [13, Definition 13.1.4, page 149]), which is a tree decomposition $(T, \mathbf{v})$ such that any bag b that is not a leaf either: 1) has exactly two children b_1, b_2 such that $\mathbf{v}(b_1) = \mathbf{v}(b) = \mathbf{v}(b_2)$, or 2) has exactly one child b', and $\mathbf{v}(b')$ is obtained from $\mathbf{v}(b)$ by either adding a single vertex, or by removing a single vertex. A C2RPQ has tree-width k if and only if it has a nice tree decomposition of width at most k [13, Lemma 13.1.2, page 149]. In the context of a nice tree decomposition of width k, a *full bag* is any bag of size

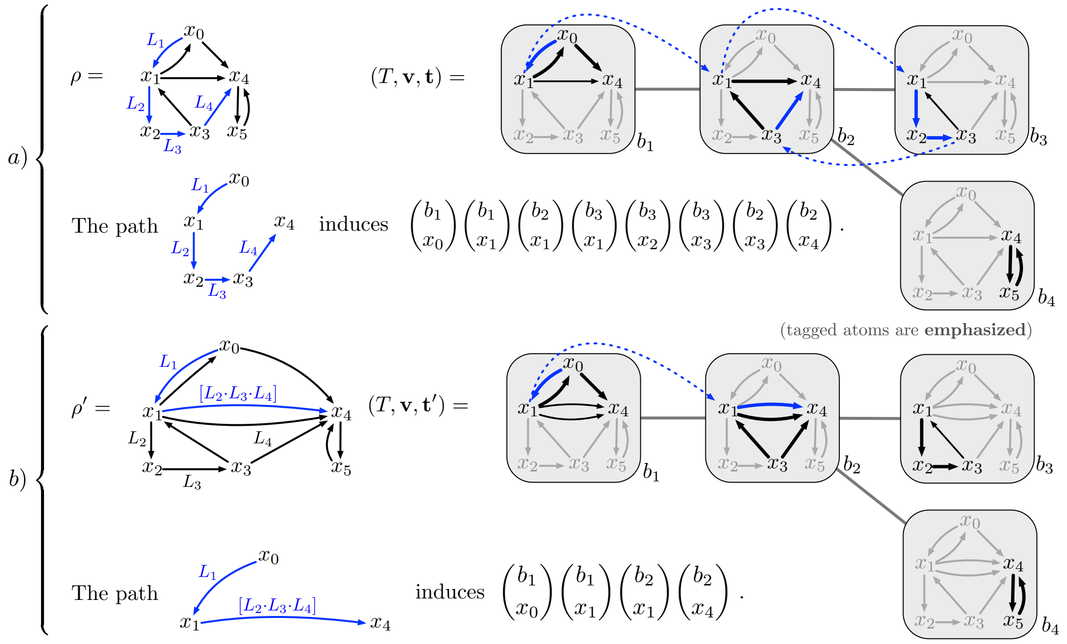

a)
$\rho =$
x_0 x_1 x_2 x_3 x_4 x_5
L_1 L_2 L_3 L_4
$(T, \mathbf{v}, \mathbf{t}) =$
b_1 b_2 b_3 b_4
The path
induces
$\binom{b_1}{x_0}\binom{b_1}{x_1}\binom{b_2}{x_1}\binom{b_3}{x_1}\binom{b_3}{x_2}\binom{b_3}{x_3}\binom{b_2}{x_3}\binom{b_2}{x_4}$.
(tagged atoms are **emphasized**)
b)
$\rho' =$
$[L_2{\cdot}L_3{\cdot}L_4]$
$(T, \mathbf{v}, \mathbf{t}') =$
The path
induces
$\binom{b_1}{x_0}\binom{b_1}{x_1}\binom{b_2}{x_1}\binom{b_2}{x_4}$.

Figure 3 *a*) Non-acyclic path induced by some path in ρ (left-hand side), in a tagged tree decomposition (right-hand side) of $f\colon \rho \xrightarrow{hom} \alpha$ in the case where $\alpha = \rho$ and f is the identity homomorphism $\mathrm{id}_\rho\colon \rho \xrightarrow{hom} \rho$.
b) Suppose that the path in ρ is the image of an atom refinement of γ. Then the cycle in the induced path can be avoided by adding an atom $x_1 \xrightarrow{[L_2\cdot L_3\cdot L_4]} x_4$ to ρ, obtaining the path of ρ', whose induced path is acyclic.

$k+1$. Note that any *non-branching path* – i.e. a path whose non-extremal bags have degree 2 – in a nice tree decomposition with n bags must have at least $\lfloor n/2 \rfloor$ bags which are not full. Finally, we define a *nice tagged tree decomposition* of $f\colon \rho \to \alpha$ to be a tagged tree decomposition of f that is also a nice tree decomposition of α.

5 Proof of the key lemma

We can now start to describe the constructions used to prove Lemma 3.8. We call a *trio* to a triple (α, ρ, f) such that $\alpha \in \mathcal{T}w_k$, $\rho \in \mathrm{Ref}(\gamma)$ and f is a strong onto homomorphism from γ to α. For clarity, we will denote such a trio by simply "$f\colon \rho \overset{hom}{\twoheadrightarrow} \alpha$". Using this terminology, in order to prove Lemma 3.8, we must show that for every trio $f\colon \rho \overset{hom}{\twoheadrightarrow} \alpha$, there exists another trio $f'\colon \rho' \overset{hom}{\twoheadrightarrow} \alpha'$ such that $\alpha \subsetneqq \alpha'$ and $\rho' \in \mathrm{Ref}^{\leqslant \ell}(\gamma)$. Our first construction, which will ultimately allow us to bound the size of atom refinements, shows that we can assume w.l.o.g. that they induce acyclic paths in a nice tagged tree decomposition of f.

▷ Claim 5.1. For any trio $f\colon \rho \overset{hom}{\twoheadrightarrow} \alpha$, there exists a trio $f'\colon \rho' \overset{hom}{\twoheadrightarrow} \alpha'$ and a nice tagged tree decomposition $(T', \mathbf{v}', \mathbf{t}')$ of width at most k of f' such that $\alpha \subsetneqq \alpha'$, $\|\rho'\| \leqslant \|\rho\|$ and every atom refinement of ρ' induces an acyclic path in the tree T', in which case we say that $(T', \mathbf{v}', \mathbf{t}')$ is *locally acyclic* .

The construction behind Claim 5.1 is quite simple, and is described and proven in Section 5, and illustrated in Figure 3.

Informal proof of Claim 5.1. Start with a trio $f\colon \rho \xrightarrow{hom}\!\!\!\!\rightarrow \alpha$, and let $(T, \mathbf{v}, \mathbf{t})$ be a nice tagged tree decomposition of f. Consider an atom refinement $\pi \mathrel{\hat=} z_0 \xrightarrow{L_1} z_1 \xrightarrow{L_2} \cdots \xrightarrow{L_n} z_n$ in ρ of some atom $x \xrightarrow{L} y$ (with $z_0 \mathrel{\hat=} x$ and $z_n \mathrel{\hat=} y$), and assume that it induces a cyclic path in T, as in Figure 3a. It means that some variables z_i and z_j are mapped by f to the same bag of T, somewhere along the path induced by π. It suffices then to contract ρ by replacing the atoms $z_i \xrightarrow{L_{i+1}} \cdots \xrightarrow{L_j} z_j$ by a single atom $z_i \xrightarrow{[L_{i+1} \cdots L_j]} z_j$ (in Figure 3, $z_i = x_1$ and $z_j = x_4$). We thus obtain a new refinement ρ' of γ. Then define α' be simply adding an atom $f(z_i) \xrightarrow{L_{i+1} \cdots L_j} f(z_j)$, see Figure 3b. The definitions of f' and $(T', \mathbf{v}', \mathbf{t}')$ are then straightforward – potentially, α' should be restricted to the image of $f'\colon \rho' \to \alpha'$ so that f' is still strong onto. Crucially, $\alpha \subsetneq \alpha'$, and α' still has tree-width at most k since we picked $f(z_i)$ and $f(z_j)$ so that they belonged to the same bag of T. ◁

Ultimately, Claim 5.1 will allow us to give a bound on the number of leaves of a nice tagged tree decomposition of a trio. The following claim – which is significantly more technical than the foregoing – will give us a bound on the height of a decomposition.

▷ Claim 5.2. Let $f\colon \rho \xrightarrow{hom}\!\!\!\!\rightarrow \alpha$ be a trio and $(T, \mathbf{v}, \mathbf{t})$ be a locally acyclic nice tagged tree decomposition of width at most k of f. Then there is a trio $f'\colon \rho' \xrightarrow{hom}\!\!\!\!\rightarrow \alpha'$ and a nice tagged tree decomposition $(T', \mathbf{v}', \mathbf{t}')$ of width at most k of f' such that:

- $\alpha \subsetneq \alpha'$,
- $(T', \mathbf{v}', \mathbf{t}')$ is locally acyclic w.r.t. f', and
- the size of the longest non-branching path in T is at most $\Theta(\|\gamma\| \cdot (k+1)^{\|\gamma\|+1})$.

To prove Claim 5.2, we will try to find, in a long non-branching path, some kind of shortcut. The piece of information that is relevant to finding this shortcut is what we call the profile of a bag.

▶ **Definition 5.3.** *Given a trio $f\colon \rho \xrightarrow{hom}\!\!\!\!\rightarrow \alpha$ and a nice tagged tree decomposition $(T, \mathbf{v}, \mathbf{t})$ of f, for each bag b of T, we say that:*

- *b is* "atomic" *if there is at least one atom $e \in \mathbf{t}^{-1}(b)$ and at least one variable x of e such that $x \in vars(\gamma)$, i.e., the atom e is not in the "middle" part of an atom refinement;*
- *otherwise, when b is non-atomic, we assign to each variable $z \in \mathbf{v}(b) \subseteq V(\alpha)$ a* type

$$\text{type}_z \mathrel{\hat=} \Big\{ x \xrightarrow{L} y \textit{ atom of } \gamma \;\Big|\; \begin{array}{l}\textit{the path induced by the atom refinement}\\ \textit{of } x \xrightarrow{L} y \textit{ in } \rho \textit{ leaves } b \textit{ at } z\end{array} \Big\};$$

then the profile *of b is the multiset of the types of z where z ranges over $\mathbf{v}(b)$.*

The rest of the proof consists in two parts: first, we show that if two non-atomic bags b and b' occurring in some non-branching path of T have the same profile, then we can essentially replace the path between b and b' by a path of constant length (Subclaim 5.4): while this construction is quite elementary, it motivates the intricate definition of the profile of a bag; then, we show that in every non-branching path, if it is sufficiently long, then we can find b and b' satisfying the aforementioned property: this part simply relies on a basic combinatorial argument (see the full version for details).

▷ Subclaim 5.4. Suppose there are two bags b and b' such that: (i) they contain at most k nodes (i.e., not full bags), (ii) they have the same profile, (iii) there is a non-branching path in T between these bags, and (iv) no bags of the path between b and b' (both included) are atomic. Then, there exists a trio $f'\colon \rho' \xrightarrow{hom}\!\!\!\!\rightarrow \alpha'$ and a nice tagged tree decomposition of f' of width at most k that can be obtained by replacing the non-branching path between b and b' in the nice tagged tree-decomposition of $f\colon \rho \xrightarrow{hom}\!\!\!\!\rightarrow \alpha$ by another non-branching path of at most $2k+1$ bags, such that $\alpha \subsetneq \alpha'$.

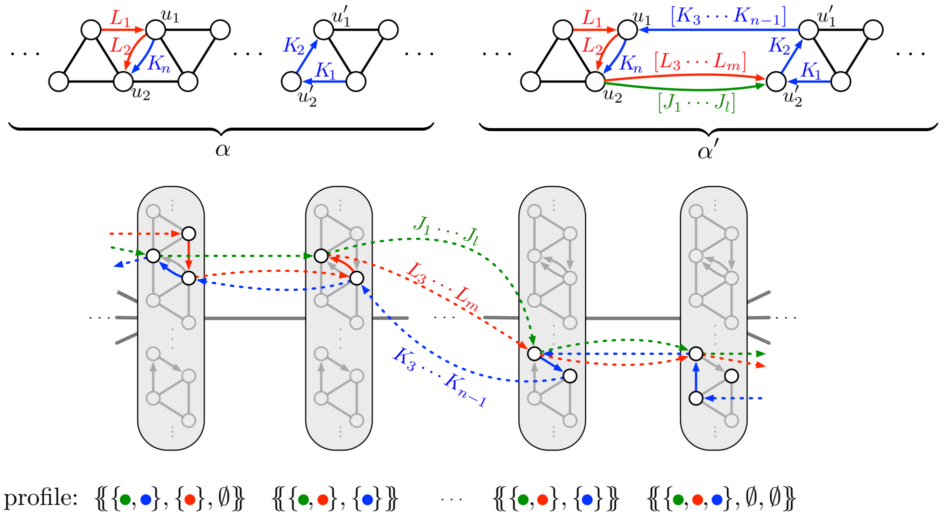


Figure 4 A long non-branching path in the tree decomposition of width 2 of an approximation α. There are two non-full bags in the path with the same profile, and thus the query α can be simplified to α' by applying contractions to the atom refinements involved.

The basic idea behind Subclaim 5.4 is that we the definition of profile was carefully design so that we could contract every refinements between b and b', while preserving every desirable properties on the trio. The construction is illustrated in Figure 4, and both an informal proof and a formal proof can be found in the full version. We can now describe key steps in the proof of Claim 5.2. A formal proof can also be found in the full version.

Proof sketch of Claim 5.2. We claim that, starting from a trio $f\colon \rho \overset{hom}{\twoheadrightarrow} \alpha$ and a locally acyclic nice tagged tree decomposition of f, if we can find a long non-branching path, then an elementary argument (see the full version) yields the existence of two bags b and b' on this path, satisfying the assumptions of Subclaim 5.4, and sufficiently far apart that the construction described in Subclaim 5.4 strictly shortens the path between b and b'. Overall, the iterative application of this construction, which preserves both the niceness and the local acyclicity of the tagged tree decomposition, and only produces bigger approximations (in the sense of containment), yields a trio $f'\colon \rho' \overset{hom}{\twoheadrightarrow} \alpha'$ with a locally acyclic nice tagged tree decomposition of width at most k, whose non-branching paths are all "small", and such that $\alpha \subseteqq \alpha'$. ◁

Finally, our main lemma follows from Claims 5.1 and 5.2.

Proof of Lemma 3.8. In order to show $\mathrm{App}_{\mathscr{Tw}_k}(\gamma) \subseteqq \mathrm{App}^{\star,\leqslant\ell}_{\mathscr{Tw}_k}(\gamma)$ – the other containment being trivial – , pick a trio $f\colon \rho \to \alpha$. Applying Claim 5.1 and then Claim 5.2 yields the existence of a trio $f'\colon \rho' \to \alpha'$ together with a nice tagged tree decomposition $(T', \mathbf{v}', \mathbf{t}')$ of f' such that $\alpha \subseteqq \alpha'$ and $(T', \mathbf{v}', \mathbf{t}')$ is locally acyclic, and any non-branching path in T' has length at most $\Theta(\|\gamma\| \cdot (k+1)^{\|\gamma\|})$.

Moreover, we can assume w.l.o.g., by applying Fact 4.2, that every leaf of T' is tagged by at least one atom of ρ'. The local acyclicity of T' implies that if b be a leaf of T', and $\pi \mathrel{\hat{=}} x \xrightarrow{L_1} t_1 \xrightarrow{L_2} \cdots \xrightarrow{L_{n-1}} t_{n-1} \xrightarrow{L_n} y$ is an atom refinement in ρ' of some atom $x \xrightarrow{L} y$ of γ, then if b is tagged by one atom of π this atom must either be $z_0 \xrightarrow{L_1} z_1$ or $z_{n-1} \xrightarrow{L_n} z_n$ by local acyclicity – see e.g. Figure 3 for a visual proof. The number of such atoms in ρ' being bounded by $2\|\gamma\|$, we conclude that T' has at most $2\|\gamma\|$ leaves.

Then, observe that a tree with at most p leaves and whose non-branching paths have length at most q is of height at most[9] $p \cdot q - 1$. We conclude that the height of T' is $\Theta(\|\gamma\|^2 \cdot (k+1)^{\|\gamma\|})$. Using again the local acyclicity of T', observe that the refinement length of ρ' is at most twice the height of T', and hence $\rho' \in \mathrm{Ref}^{\leqslant \ell}(\gamma)$ where $\ell = \Theta(\|\gamma\|^2 \cdot (k+1)^{\|\gamma\|})$. In other words, $\alpha' \in \mathrm{App}^{\star,\leqslant \ell}_{\mathcal{T}\mathfrak{w}_k}(\gamma)$. Hence, we have shown that for all $\alpha \in \mathrm{App}^{\star}_{\mathcal{T}\mathfrak{w}_k}(\gamma)$, there exists $\alpha' \in \mathrm{App}^{\star,\leqslant \ell}_{\mathcal{T}\mathfrak{w}_k}(\gamma)$ such that $\alpha \subseteq \alpha'$. ◀

6 Queries over simple regular expressions

A *simple regular expression*, or *SRE*, is a regular expression the form a^* for some letter $a \in \mathbb{A}$ or of the form $a_1 + \cdots + a_m$ for some $a_1, \ldots, a_m \in \mathbb{A}$.

Let UCRPQ(SRE) be the set of all UCRPQ whose languages are expressed via SRE expressions. Observe that UCRPQ(SRE) is semantically closed under concatenation, that is, concatenations of SRE expressions can be also expressed in the language. For example, $\gamma(x,y) = x \xrightarrow{a^* \cdot (a+b) \cdot b^*} y$ is equivalent to $\gamma'(x,y) = x \xrightarrow{a^*} z \wedge z \xrightarrow{a+b} z' \wedge z' \xrightarrow{b^*} y$. One interest of UCRPQ(SRE) comes from the fact that it is used widely in practice, as recent studies on SPARQL query logs on Wikidata, DBpedia and other sources show that this kind of regular expressions cover a majority of the queries investigated, e.g., 75% of the "property paths" (C2RPQ atoms) of the corpus of 1.5M queries of [7, Table 15]. An additional interest comes from the fact that the containment problem for UCRPQ(SRE) is much better behaved than for general UCRPQs, since it is in Π^p_2 [11, Corollary 5.2], that is, just one level up the polynomial hierarchy compared to the CQ containment problem, which is in NP [9], and in sharp contrast with the costly ExpSpace-complete CRPQ containment problem [8, 12].

We devote this section to showing the following result.

▶ **Theorem 6.1.** *For $k > 1$, the semantic tree-width k problem for* UCRPQ(SRE) *is in* Π^p_2.

Observe that simple regular expressions are closed under sublanguages. Hence, in the light of Theorem 1.5, the maximal under-approximation of a UCRPQ(SRE) query by infinitary unions of CQs of tree-width k is always equivalent to a UCRPQ(SRE) query of tree-width k. We will explain how the construction of the maximal under-approximation of the previous section can be exploited to improve the complexity from 2ExpSpace down to Π^p_2.

6.1 Summary queries

We will first show that the maximal under-approximation of tree-width k of a UC2RPQ can be expressed as a union of polynomial sized "summary" queries. Each summary query represents a union of exponentially-bounded C2RPQs sharing some *common structure*. These are normal UC2RPQ queries extended with some special kind of atoms, called "path-l approximations". Intuitively, a path-l approximation is a maximal under-approximation of tree-width l of queries of the form $\bigwedge_i x_i \xrightarrow{L_i} y_i$ such that $x_i \neq y_j$ for all i,j. Path-l approximations may require an exponential size when represented as UCRPQs. Formally, a *path-l approximation* is a query of the form "$\mathbf{P}_l(X,Y,\gamma)$" where: (i) X, Y, are two disjoint sets of variables of size at most l, (ii) $\gamma(\bar{z})$ is a conjunction of atoms $\bigwedge_{1 \leqslant i \leqslant n} A_i(x_i, y_i)$ where $\bar{z}$ contains all variables of $X \cup Y$, (iii) each A_i is a C2RPQ atom of the form $x_i \xrightarrow{L} y_i$ or $y_i \xrightarrow{L} x_i$ such that x_i is in X and y_i is in Y. We give its semantics in terms of infinitary

[9] The length of a path being its number of nodes, and with the convention that the height of a single node is zero.

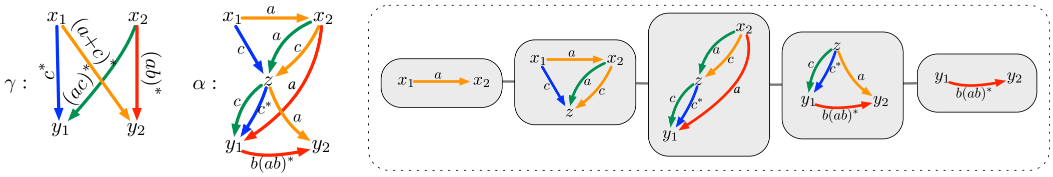


Figure 5 The query $\mathbf{P}_l(\{x_1, x_2\}, \{y_1, y_2\}, \gamma)$ (where $l = 2$) on the left contains the approximation α, witnessed by the path decomposition on the right.

unions of CQs. A query like the one before is defined to be equivalent to the (infinitary) union of all queries $\alpha(\bar{z}) \in \text{App}_{\mathcal{Tw}_k}(\gamma)$ that admit a path decomposition of width l having the bag X at the root and Y at the leaf – where a *path decomposition* is defined to be any tree decomposition whose underlying tree is a path. See Figure 5 for an example.

We now simply define a *k-summary query* as a C2RPQ extended with path-l approximation atoms for any $l \leqslant k$, with the expected semantics. The important property of summary queries is that they are exponentially more succinct than the UC2RPQ counterpart for expressing maximal under-approximations, as the next lemma shows.

▶ **Proposition 6.2.** *For every class $\mathcal{L}$ closed under sublanguages, and for every* C2RPQ($\mathcal{L}$) *γ,* $\text{App}_{\mathcal{Tw}_k}(\gamma)$ *can be expressed as a union of polynomial-sized k-summary queries having only* C2RPQ($\mathcal{L}$) *atoms. Further, one can test in* NP *if a summary query is part of this union. We call* $\text{App}^{\text{zip}}_{\mathcal{Tw}_k}(\gamma)$ *to any such a union of summary queries.*

Informal proof. As corollary of the proof of Lemma 3.8, we can assume to have $\text{App}_{\mathcal{Tw}_k}(\gamma)$ expressed as a union of C2RPQ($\mathcal{L}$) with a nice tree decomposition of width k with a linear number of leaves, and hence it suffices to replace non-branching paths with path-l approximations. Concretely, for any such C2RPQ α having a witnessing tree decomposition with a long non-branching path, the tree must contain a sub-path whose every bag is non-atomic, and such that it starts and ends in bags of size at most k (by the niceness property). Such a non-atomic non-branching path can be "compressed" by replacing the subquery corresponding to the path with a corresponding path-l approximation query. The resulting summary query will contain α and in turn be contained in γ. Simultaneously applying such replacement to all non-atomic non-branching paths of maximal length then yields a polynomial sized summary query.

Further, given a k-summary query σ, one can test in NP whether there exists an element of $\text{App}^{\star}_{\mathcal{Tw}_k}(\gamma)$ that leads to such summary query by the process just mentioned. This is done by first checking that σ is of the "right shape" (essentially a query of tree-width k when disregarding the path-l approximation atoms), and that it is contained in γ via a polynomial refinement $\rho \in \text{Ref}(\gamma)$ and a strong onto homomorphism to the query resulting from replacing $\mathbf{P}_l(X, Y, \delta)$ atoms with δ in σ. ◀

6.2 Semantic tree-width problem

With the previous results in place, we now show that the semantic tree-width k problem is in Π^p_2 for UCRPQ(SRE), for every $k > 1$.

▶ **Theorem 6.1.** *For $k > 1$, the semantic tree-width k problem for* UCRPQ(SRE) *is in* Π^p_2.

Proof. It suffices to show the statement for any CRPQ(SRE) γ. Remember that γ is of semantic tree-width k if, and only if, $\gamma \subseteqq \text{App}^{\text{zip}}_{\mathcal{Tw}_k}(\gamma)$. The first ingredient to this proof is the fact that this containment has a polynomial counter-example property:

▷ Claim 6.3. If $\gamma \not\subseteq \mathrm{App}^{\mathrm{zip}}_{\mathcal{T}\!w_k}(\gamma)$ then there is a polynomial-sized expansion ξ of γ such that $\xi \not\subseteq \mathrm{App}^{\mathrm{zip}}_{\mathcal{T}\!w_k}(\gamma)$.

This is because any path $x_0 \xrightarrow{a} \cdots \xrightarrow{a} x_m$ in an expansion ξ of γ such that $m > \|\gamma\|$ and $\xi \subseteq \mathrm{App}^{\mathrm{zip}}_{\mathcal{T}\!w_k}(\gamma)$ can be "pumped" to an even longer path of any length greater than m obtaining another expansion ξ' such that $\xi' \subseteq \mathrm{App}^{\mathrm{zip}}_{\mathcal{T}\!w_k}(\gamma)$. This implies that a minimal counterexample must be of polynomial size.

The second ingredient is that testing whether CQ is a counterexample is in NP.

▷ Claim 6.4. The problem of testing, given a CQ γ, whether $\gamma \subseteq \mathrm{App}^{\mathrm{zip}}_{\mathcal{T}\!w_k}(\gamma)$, is in NP.

Informal proof. We first guess a polynomial-sized k-summary query δ_{zip} and test in NP that it is part of $\mathrm{App}^{\mathrm{zip}}_{\mathcal{T}\!w_k}(\gamma)$ by Proposition 6.2. We now guess a valuation $\mu : \mathit{vars}(\delta_{\mathrm{zip}}) \to \mathit{vars}(\gamma)$ and test that it is a homomorphism $\xi \xrightarrow{hom} \gamma$, where ξ is an expansion of the CRPQ resulting from discarding the path-l approximation atoms of δ_{zip}, which can be done in NL. It remains to check that each atom $\mathbf{P}_l(X, Y, \hat{\delta})$ of δ has an expansion $\hat{\xi}$ such that $h : \hat{\xi} \xrightarrow{hom} \gamma$ for a homomorphism such that $h(x) = \mu(x)$ for all $x \in X \cup Y$. This can be done via an NL algorithm using $l + 1$ pointers to traverse the width-l nice path decomposition of $\hat{\xi}$, simultaneously guessing $\hat{\xi}$, the expansion of γ which homomorphically maps to $\hat{\xi}$, and the valuation of h. ◁

As a consequence of the two claims, we obtain a Σ^p_2 algorithm for non-containment of $\gamma \subseteq \mathrm{App}^{\mathrm{zip}}_{\mathcal{T}\!w_k}(\gamma)$. We first guess an expansion ξ of γ of polynomial size, and we then test $\xi \not\subseteq \mathrm{App}^{\mathrm{zip}}_{\mathcal{T}\!w_k}(\gamma)$ in coNP. This gives a Π^p_2 algorithm for the semantic tree-width k problem. ◀

7 Discussion

We have studied the definability and approximation of UC2RPQ queries by queries of bounded tree-width and shown that the maximal under-approximation in terms of an infinitary union of conjunctive queries of tree-width $k > 1$ can be always effectively expressed as a UC2RPQ of tree-width k (Corollary 3.9). However, while the semantic tree-width 1 problem as shown to be ExpSpace-complete [5, Theorem 6.1, Proposition 6.2], we have left a gap between our lower and upper bounds in Theorem 1.3.

▷ Question 7.1. For $k > 1$, is the semantic tree-width k problem ExpSpace-complete?

We also do not know whether the Π^p_2 bound on the semantic tree-width k problem for UCRPQ(SRE) has a matching lower bound. The known lower bound for the UCRPQ(SRE) containment problem [11, Theorem 5.1] does not seem to be useful to be used in a reduction, since it necessitates queries of arbitrary high tree-width.

It is worth stressing that in [5] two-way navigation plays a crucial part to prove the existence of the maximal under-approximation by a UC2RPQ of tree-width 1 [5, Theorem 5.2], but this feature plays no role whatsoever in our proof – see Theorem 1.5 and Remark 2.5. Moreover, $\mathcal{T}\!w_k$ queries enjoy the very nice property of being closed under refinement when $k > 1$ but not when $k = 1$ – see Example 3.7 – which forces Barceló, Romero, and Vardi to introduce the notion of "pseudoacyclicity" [5, §5.2.1], namely the greatest subclass of $\mathcal{T}\!w_1$ closed under refinement, while we can directly work with the rather comfortable $\mathrm{App}^{\star}_{\mathcal{T}\!w_k}(\gamma)$. On the other hand, graphs of tree-width $k > 1$ being combinatorially less trivial than graphs of tree-width 1, our proof must carefully handle this information, using tagged tree decompositions of Section 4. Similarly to [5, Theorem 6.3] for the case $k = 1$, our results implies that for each $k > 1$ the evaluation problem for UC2RPQs Γ of semantic tree-width k is fixed-parameter tractable (FPT) in the size of the query, i.e. in $O(|G|^{k+1} \cdot f(|\Gamma|))$ for a computable function f, where G is the database given as input. This improves the dependence on the size of the database, namely $O(|G|^{2k+2} \cdot f(|\Gamma|))$, proven by Romero, Barceló and Vardi [15, Corollary IV.12].

▷ Question 7.2 (Also mentionned in [15, §IV-(4))]). Does every r.e. class of CRPQs with FPT evaluation has bounded semantic tree-width?

Finally, as a consequence of the existence of a minimal equivalent CQ [9, Theorem 12], a CQ is equivalent to a CQ of tree-width at most k if, and only if, it is equivalent to a finite union of CQs of tree-width at most k. Example 1.2 suggests that this is false for CRPQs. Both for $k = 1$ (see [5, §6.5]) and $k \geqslant 2$, we do not if the problem of whether a given CRPQ is equivalent to a single CRPQ of tree-width at most k is decidable. More generally, while we know that there exists CRPQs γ_1, γ_2 such that $\gamma_1 \vee \gamma_2$ is not equivalent to a single CRPQ[10], we have a very limited understanding of how much union adds to the expressive power of CRPQs. This begs the following question:

▷ Question 7.3. Is the problem of whether a given UCRPQ (resp. UC2RPQ) is equivalent to a single CRPQ (resp. C2RPQ) decidable?

References

1 Renzo Angles, Marcelo Arenas, Pablo Barceló, Aidan Hogan, Juan Reutter, and Domagoj Vrgoč. Foundations of modern query languages for graph databases. *ACM Comput. Surv.*, 50(5), September 2017. doi:10.1145/3104031.

2 Stefan Arnborg, Derek G Corneil, and Andrzej Proskurowski. Complexity of finding embeddings in a k-tree. *SIAM Journal on Algebraic Discrete Methods*, 8(2):277–284, 1987.

3 Pablo Barceló, Leonid Libkin, and Miguel Romero. Efficient approximations of conjunctive queries. In *Proceedings of the 31st ACM SIGMOD-SIGACT-SIGAI symposium on Principles of Database Systems*, pages 249–260, 2012. doi:10.1145/2213556.2213591.

4 Pablo Barceló and Miguel Romero. The Complexity of Reverse Engineering Problems for Conjunctive Queries. In Michael Benedikt and Giorgio Orsi, editors, *20th International Conference on Database Theory (ICDT 2017)*, volume 68 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 7:1–7:17, Dagstuhl, Germany, 2017. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.ICDT.2017.7.

5 Pablo Barceló, Miguel Romero, and Moshe Y. Vardi. Semantic acyclicity on graph databases. *SIAM Journal on computing*, 45(4):1339–1376, 2016. doi:10.1137/15M1034714.

6 Pablo Barceló Baeza. Querying graph databases. In *Proceedings of the 32nd ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, PODS '13, pages 175–188, New York, NY, USA, 2013. Association for Computing Machinery. doi:10.1145/2463664.2465216.

7 Angela Bonifati, Wim Martens, and Thomas Timm. An analytical study of large SPARQL query logs. *VLDB Journal*, 29(2):655–679, 2020. doi:10.1007/s00778-019-00558-9.

8 Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Moshe Y. Vardi. Containment of conjunctive regular path queries with inverse. In *Principles of Knowledge Representation and Reasoning (KR)*, pages 176–185, 2000.

[10] See [4, Figure 1 & Example 21, p. 15]: by denoting $\gamma_1, \gamma_2, \gamma_3$ the CQs in the top-left, below-left and right parts of Figure 1, respectively, their example shows that any CRPQ that contains both γ_1 and γ_2 must also contain γ_3. As γ_3 is contained in neither γ_1 nor γ_2, our claim follows.

9 Ashok K. Chandra and Philip M. Merlin. Optimal implementation of conjunctive queries in relational data bases. In John E. Hopcroft, Emily P. Friedman, and Michael A. Harrison, editors, *Proceedings of the 9th Annual ACM Symposium on Theory of Computing, May 4-6, 1977, Boulder, Colorado, USA*, pages 77–90. ACM, 1977. doi:10.1145/800105.803397.

10 Diego Figueira. Foundations of graph path query languages – course notes for the reasoning web summer school 2021. In *Reasoning Web. Declarative Artificial Intelligence – 17th International Summer School 2021, Leuven, Belgium, September 8-15, 2021, Tutorial Lectures*, volume 13100 of *Lecture Notes in Computer Science*, pages 1–21. Springer, 2021. doi:10.1007/978-3-030-95481-9_1.

11 Diego Figueira, Adwait Godbole, S. Krishna, Wim Martens, Matthias Niewerth, and Tina Trautner. Containment of simple conjunctive regular path queries. In *Principles of Knowledge Representation and Reasoning (KR)*, 2020. URL: https://hal.archives-ouvertes.fr/hal-02505244.

12 Daniela Florescu, Alon Levy, and Dan Suciu. Query containment for conjunctive queries with regular expressions. In *ACM Symposium on Principles of Database Systems (PODS)*, pages 139–148. ACM Press, 1998. doi:10.1145/275487.275503.

13 Ton Kloks. *Treewidth: computations and approximations.* Lecture Notes in Computer Science. Springer, 1994. doi:10.1007/BFb0045375.

14 Jaroslav Nešetřil and Patrice Ossona de Mendez. *Sparsity – Graphs, Structures, and Algorithms*, volume 28 of *Algorithms and combinatorics.* Springer, 2012. doi:10.1007/978-3-642-27875-4.

15 Miguel Romero, Pablo Barceló, and Moshe Y. Vardi. The homomorphism problem for regular graph patterns. In *Annual Symposium on Logic in Computer Science (LICS)*, pages 1–12. IEEE Computer Society Press, 2017. doi:10.1109/LICS.2017.8005106.