Boundedness of Conjunctive Regular Path Queries

Pablo Barceló
Department of Computer Science, University of Chile, Santiago, Chile
IMFD, Santiago, Chile
pbarcelo@dcc.uchile.cl

Diego Figueira
CNRS & LaBRI, Talence, France
diego.figueira@labri.fr

Miguel Romero
Department of Computer Science, University of Oxford, Oxford, UK
miguel.romero@cs.ox.ac.uk

Abstract

We study the boundedness problem for unions of conjunctive regular path queries with inverses (UC2RPQs). This is the problem of, given a UC2RPQ, checking whether it is equivalent to a union of conjunctive queries (UCQ). We show the problem to be ExpSpace-complete, thus coinciding with the complexity of containment for UC2RPQs. As a corollary, when a UC2RPQ is bounded, it is equivalent to a UCQ of at most triple-exponential size, and in fact we show that this bound is optimal. We also study better behaved classes of UC2RPQs, namely *acyclic UC2RPQs of bounded thickness*, and *strongly connected UCRPQs*, whose boundedness problem is, respectively, PSpace-complete and Π_2^P-complete. Most upper bounds exploit results on limitedness for distance automata, in particular extending the model with alternation and two-wayness, which may be of independent interest.

2012 ACM Subject Classification Theory of computation → Database query languages (principles); Theory of computation → Quantitative automata

Keywords and phrases regular path queries, boundedness, limitedness, distance automata

Digital Object Identifier 10.4230/LIPIcs.ICALP.2019.104

Category Track B: Automata, Logic, Semantics, and Theory of Programming

Related Version A full version of this paper can be found at `https://arxiv.org/abs/1904.00850` and `https://hal.archives-ouvertes.fr/hal-02056388`.

Funding Barceló is funded by the Millennium Inst. for Foundational Research on Data and Fondecyt 1170109, and Figueira by ANR project DELTA (grant ANR-16-CE40-0007) and ANR project QUID (grant ANR-18-CE40-0031). This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 714532). The paper reflects only the authors' views and not the views of the ERC or the European Commission. The European Union is not liable for any use that may be made of the information contained therein.

Acknowledgements We are grateful to Thomas Colcombet for helpful discussions and valuable ideas in relation to the results of Section 5.

1 Introduction

Boundedness is an important property of formulas in logics with fixed-point features. At the intuitive level, a formula φ in any such logic is bounded if its *fixed-point depth*, *i.e.*, the number of iterations that are needed to evaluate φ on a structure $\mathbf{A}$, is fixed (and thus it is independent of $\mathbf{A}$). In databases and knowledge representation, boundedness is regarded as an interesting theoretical phenomenon with relevant practical implications [25, 8]. In

46th International Colloquium on Automata, Languages, and Programming (ICALP 2019).
Editors: Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi;
Article No. 104; pp. 104:1–104:15
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

fact, while several applications in these areas require the use of recursive features, actual real-world systems are either not designed or not optimized to cope with the computational demands that such features impose. Bounded formulas, in turn, can be reformulated in non-recursive logics, such as FO, or even as a *union of conjunctive queries* (UCQ) when φ itself is positive. UCQs form the core of most systems for data management and ontological query answering, and, in addition, are the focus of advanced optimization techniques. It has also been experimentally verified in some contexts that recursive features encountered in practice are often used in a somewhat "harmless" way, and that many of such queries are in fact bounded [23]. Thus, checking if a recursive formula φ is bounded, and building an equivalent non-recursive formula φ' when the latter holds, are important optimization tasks.

The study of boundedness for *Datalog* programs, *i.e.*, the least fixed-point extension of the class of UCQs, received a lot of attention during the late 80s and early 90s. Two seminal results established that checking boundedness is undecidable in general for Datalog [22], but becomes decidable for *monadic* Datalog, *i.e.*, those programs in which each intensional predicate is monadic [19]. The past few years have seen a resurgence of interest in boundedness problems. This is due, in part, to the development of the theory of *cost automata* over trees (both finite and infinite) in a series of landmark results, in particular relating to its *limitedness* problem. In a few words, cost automata are generalizations of finite automata associating a *cost* from $\mathbb{N} \cup \{\infty\}$ to every input tree (instead of simply accepting or rejecting). The limitedness problem asks, given a cost automata, whether there is a uniform bound on the cost over all (accepting) input trees. Some deep results establish that checking limitedness is decidable for well-behaved classes of cost automata over trees [18, 35, 36, 7]. Remarkably, for several logics of interest the boundedness problem can be reduced to the limitedness for cost automata in such well-behaved classes. Those reductions have enabled powerful decidability results for the boundedness problem. As an example, it has been shown in this way that boundedness is decidable for monadic second-order logic (MSO) over structures of bounded treewidth [11], which corresponds to an extension of Courcelle's Theorem, and also for the *guarded negation* fragment of least fixed-point logic (LFP), even in the presence of unguarded parameters [6]. Cost automata have also been used to study the complexity of boundedness for guarded Datalog programs [7, 3].

Graph databases is a prominent area of study within database theory, in which the use of recursive queries is crucial [2, 1]. A graph database is a finite edge-labeled directed graph. The most basic navigational querying mechanism for graph databases corresponds to the class of *regular path queries* (RPQs), which check whether two nodes of the graph are connected by a path whose label belongs to a given regular language. RPQs are often extended with the ability to traverse edges in both directions, giving rise to the class of *two-way* RPQs, or 2RPQs [15]. The core of the most popular recursive query languages for graph databases is defined by *conjunctive* 2RPQs, or C2RPQs, which are the closure of 2RPQs under conjunction and existential quantifications [14]. We also consider *unions* of C2RPQs, or UC2RPQs. It can be shown that a UC2RPQ is bounded iff it is equivalent to some UCQ. In spite of the inherent recursive nature of UC2RPQs, their boundedness problem has not been studied in depth. Here we develop such a study by showing the following:

- The boundedness problem for UC2RPQs is ExpSpace-complete. The lower bound holds even for CRPQs. This implies that boundedness is not more difficult than *containment* for UC2RPQs, which was shown to be ExpSpace-complete in [14].
- From our upper bound construction it follows that if a UC2RPQ is bounded, then it is equivalent to a UCQ of triple exponential size. We show that this bound is optimal.
- Finally, we obtain better complexity bounds for some subclasses of UC2RPQs; namely, for acyclic UC2RPQs of *bounded thickness*, in which case boundedness becomes PSpace-complete, and for *strongly connected* UCRPQs, for which it is Π_2^P-complete.

It is important to stress that UC2RPQs can be easily translated into guarded LFP with unguarded parameters, for which boundedness was shown to be decidable by applying sophisticated cost automata techniques as mentioned above. However, the complexity of the boundedness problem for such a logic is currently not well-understood – and it is at least 2EXPTIME-hard [7] – and hence this translation does not yield, in principle, optimal complexity bounds for our problem. To study the boundedness for UC2RPQs, we develop instead techniques especially tailored to UC2RPQs. In fact, since the recursive structure of UC2RPQs is quite tame, their boundedness problem can be translated into the limitedness problem for a much simpler automata model than cost automata on trees; namely, *distance* automata on finite words. Distance automata are nothing more than usual NFAs with two sorts of transitions: costly and non-costly. Such an automaton is limited if there is an integer $k \geq 1$ such that every word accepted by the NFA has an accepting run with at most k costly transitions. A beautiful result in automata theory established the decidability of the limitedness problem for distance automata [24], which is actually in PSPACE [29]. While being a difficult result, by now we have quite transparent proofs of this fact (see, *e.g.*, [26]). We exploit our translation to obtain tight complexity upper bounds for boundedness of UC2RPQs. Some of the proofs in the paper require extending the study of limitedness to *alternating* and *two-way* distance automata, while preserving the PSPACE bound for the limitedness problem. We believe these results to be of independent interest.

Organization of the paper. Section 2 contains preliminaries. We present characterizations of boundedness for UC2RPQs in Section 3 and an application of those to pinpoint the complexity of BOUNDEDNESS for RPQs in Section 4. Distance automata and results about them are given in Section 5. We analyze the complexity of BOUNDEDNESS for general UC2RPQs in Section 6 and present some classes of UC2RPQs with better complexity of BOUNDEDNESS in Section 7. We finish with a discussion in Section 8.

2 Preliminaries

We assume familiarity with *non-deterministic finite automata* (NFA), *two-way* NFA (2NFA), and *alternating finite automata* (AFA) over finite words. We often blur the distinction between an NFA $\mathcal{A}$ and the language $L(\mathcal{A})$ it defines; similarly for regular expressions.

Graph databases and conjunctive regular path queries. A *graph database* over a finite alphabet $\mathbb{A}$ is a finite edge-labelled graph $G = (V, E)$ over $\mathbb{A}$, where V is a finite set of vertices and $E \subseteq V \times \mathbb{A} \times V$ is the set of labelled edges. We write $u \xrightarrow{a} v$ to denote an edge $(u, a, v) \in E$. We define the alphabet $\mathbb{A}^{\pm} := \mathbb{A} \dot{\cup} \mathbb{A}^{-1}$ that extends $\mathbb{A}$ with the set $\mathbb{A}^{-1} := \{a^{-1} \mid a \in \mathbb{A}\}$ of "inverses" of symbols in $\mathbb{A}$. An *oriented path* from u to v in a graph database $G = (V, E)$ over alphabet $\mathbb{A}$ is a pair $\pi = (\sigma, \ell)$ where σ and ℓ are (possibly empty) sequences $\sigma = (v_0, a_1, v_1), (v_1, a_2, v_2), \dots, (v_{k-1}, a_k, v_k) \in V \times \mathbb{A} \times V$, and $\ell = \ell_1, \dots, \ell_k \in \{-1, 1\}$, for $k \geq 0$, such that $u = v_0$, $v = v_k$, and for each $1 \leq i \leq k$, we have that $\ell_i = 1$ implies $(v_{i-1}, a_i, v_i) \in E$; and $\ell_i = -1$ implies $(v_i, a_i, v_{i-1}) \in E$. The *label* of π is the word $b_1 \dots b_k \in (\mathbb{A}^{\pm})^*$, where $b_i = a_i$ if $\ell_i = 1$; otherwise $b_i = a_i^{-1}$. When $k = 0$ the label of π is the empty word ε. If $\ell_i = 1$ for every $1 \leq i \leq k$, we say that π is a *directed path*. Note that in this case, the label of π belongs to $\mathbb{A}^*$.

A *regular path query* (RPQ) over $\mathbb{A}$ is a regular language $L \subseteq \mathbb{A}^*$, which we assume to be given as an NFA. The evaluation of L on a graph database $G = (V, E)$ over $\mathbb{A}$, written $L(G)$, is the set of pairs $(u, v) \in V \times V$ such that there is a directed path from u to v in G

whose label belongs to L. 2RPQs extend RPQs with the ability to traverse edges in both directions. Formally, a 2RPQ L over $\mathbb{A}$ is simply an RPQ over $\mathbb{A}^{\pm}$. The evaluation $L(G)$ of L over a graph database $G = (V, E)$ over $\mathbb{A}$ is the set of pairs $(u, v) \in V \times V$ such that there is an oriented path from u to v in G whose label belongs to L.

Conjunctive 2RPQs (C2RPQs) are obtained by taking the closure of 2RPQs under conjunction and existential quantification, *i.e.*, a C2RPQ over $\mathbb{A}$ is an expression $\gamma := \exists\bar{z}\,\big((x_1 \xrightarrow{L_1} y_1) \wedge \cdots \wedge (x_m \xrightarrow{L_m} y_m)\big)$, where each L_i is a 2RPQ over $\mathbb{A}$ and $\bar{z}$ is a tuple of variables among those in $\{x_1, y_1, \ldots, x_m, y_m\}$. We say that γ is a CRPQ if each L_i is an RPQ. If $\bar{x} = (x^1, \ldots, x^n)$ is the tuple of *free variables* of γ, *i.e.*, those that are not existentially quantified in $\bar{z}$, then the evaluation $\gamma(G)$ of the C2RPQ γ over a graph database G is the set of all tuples $h(\bar{x}) = (h(x^1) \ldots, h(x^n))$, where h ranges over all mappings $h : \{x_1, y_1, \ldots, x_m, y_m\} \to V$ such that $(h(x_i), h(y_i)) \in L_i(G)$ for each $1 \leq i \leq m$.

A *union* of C2RPQs (UC2RPQ) is an expression of the form $\Gamma := \bigvee_{1 \leq i \leq n} \gamma_i$, where the γ_i's are C2RPQ, all of which have exactly the same free variables. The evaluation $\Gamma(G)$ of Γ over a graph database G is $\bigcup_{1 \leq i \leq n} \gamma_i(G)$. We often write $\Gamma(\bar{x})$ to denote that $\bar{x}$ is the tuple of free variables of Γ. A UC2RPQ Γ is *Boolean* if it contains no free variables.

Given UC2RPQs Γ and Γ', we write $\Gamma \subseteq \Gamma'$ if $\Gamma(G) \subseteq \Gamma'(G)$ for each graph database G. Hence, Γ and Γ' are *equivalent* if $\Gamma \subseteq \Gamma'$ and $\Gamma' \subseteq \Gamma$, *i.e.*, $\Gamma(G) = \Gamma'(G)$ for every G.

Boundedness of UC2RPQs. CRPQs, and even UC2RPQs, can easily be expressed in *Datalog*, the least fixed-point extension of the class of *union of conjunctive queries* (UCQs). Hence, we can directly define the boundedness of a UC2RPQ in terms of the boundedness of its equivalent Datalog program, which is a well-studied problem [25]. The latter, however, coincides with being equivalent to some UCQ [31]. In the setting of graph databases, a *conjunctive query* (CQ) over $\mathbb{A}$ is simply a CRPQ over $\mathbb{A}$ of the form $\exists\bar{z} \bigwedge_{1 \leq i \leq m}(x_i \xrightarrow{a_i} y_i)$ where the a_is range over $\mathbb{A} \cup \{\varepsilon\}$. Notice that atoms of the form $x \xrightarrow{\varepsilon} y$ correspond to *equality atoms* $x = y$. Analogously, one can define unions of CQs (UCQs). Note that, modulo equality atoms, a CQ over $\mathbb{A}$ can be seen as a graph database over $\mathbb{A}$. Hence, we shall slightly abuse notation and, in the setting of CQs, use notions defined for graph databases (such as oriented paths).

A UC2RPQ Γ is *bounded* if it is equivalent to some UCQ Φ. In this article we study the complexity of the problem Boundedness, which takes as input a UC2RPQ Γ and asks whether Γ is bounded.

▶ **Example 1.** Consider the Boolean UCRPQ $\Gamma = \gamma_1 \vee \gamma_2$ over the alphabet $\mathbb{A} = \{a, b, c, d\}$ such that $\gamma_1 = \exists x, y\,(x \xrightarrow{L_b} y \wedge x \xrightarrow{L_{b,d}} y)$ and $\gamma_2 = \exists x, y\,(x \xrightarrow{L_d} y \wedge x \xrightarrow{L_{b,d}} y)$, where $L_b := a^+b^+c$, $L_d := ad^+c^+$, and $L_{b,d} := a^+(b + d)c^+$. For $e \in \mathbb{A}$, recall that e^+ denotes the language $e(e^*)$. As we shall explain in Example 4, we have that γ_1 and γ_2 are unbounded. However, Γ is bounded, and in particular, it is equivalent to the UCQ $\Phi = \varphi_1 \vee \varphi_2$, where φ_1 and φ_2 correspond to $\exists x, y\,(x \xrightarrow{abc} y)$ and $\exists x, y\,(x \xrightarrow{adc} y)$, respectively. ◀

3 Characterizations of Boundedness for UC2RPQs

In this section we provide two simple characterizations of when a UC2RPQ is bounded that will be useful to analyze the complexity of Boundedness. Let $\varphi(\bar{x})$ and $\varphi'(\bar{x})$ be CQs over $\mathbb{A}$ with variable sets $\mathcal{V}$ and $\mathcal{V}'$, respectively. Let $=_\varphi$ and $=_{\varphi'}$ be the binary relations induced on $\mathcal{V}$ and $\mathcal{V}'$ by the equality atoms of φ and φ', respectively, and $=_\varphi^*$ and $=_{\varphi'}^*$ be their reflexive-transitive closure. A *homomorphism* from φ to φ' is a mapping $h : \mathcal{V} \to \mathcal{V}'$

such that: (i) $x =^*_\varphi y$ implies $h(x) =^*_{\varphi'} h(y)$; (ii) $h(\bar{x}) = \bar{x}$; and (iii) for each atom $x \xrightarrow{a} y$ in φ with $a \in \mathbb{A}$, there is an atom $x' \xrightarrow{a} y'$ in φ' such that $h(x) =^*_{\varphi'} x'$ and $h(y) =^*_{\varphi'} y'$. We write $\varphi \to \varphi'$ if such a homomorphism exists. It is known that $\varphi \to \varphi'$ iff $\varphi' \subseteq \varphi$ [16].

An *expansion* of a C2RPQ $\gamma(\bar{x})$ over $\mathbb{A}$ is a CQ $\lambda(\bar{x})$ over $\mathbb{A}$ with minimal number of variables and atoms such that (i) λ contains each variable of γ, (ii) for each atom $A = x \xrightarrow{L} y$ of γ, there is an oriented path π_A in λ from x to y with label $w_A \in L$ whose intermediate variables (*i.e.*, those not in $\{x, y\}$) are distinct from one another, and (iii) intermediate variables of different oriented paths π_A and $\pi_{A'}$ are disjoint. Note that the free variables of λ and γ coincide. Intuitively, the expansion λ is obtained from γ by choosing for each atom $A = x \xrightarrow{L} y$ a word $w_A \in L$, and "expanding" $x \xrightarrow{L} y$ into the "fresh oriented path" π_A from x to y with label w_A. When $w_A = \varepsilon$ then λ contains the equality atom $x = y$. An expansion of a UC2RPQ Γ is an expansion of some C2RPQ in Γ. Observe that a (U)C2RPQ is always equivalent to the (potentially infinite) UCQ given by its set of expansions. Even more, it is equivalent to the UCQ defined by its *minimal* expansions, as introduced below.

If λ is an expansion of a UC2RPQ Γ, we define the *size* of λ, denoted by $\|\lambda\|$, to be the number of (non-equality) atoms in λ. We say that λ is *minimal*, if there is no expansion λ' such that $\lambda' \to \lambda$ and $\|\lambda'\| < \|\lambda\|$. Intuitively, an expansion is minimal if its answers cannot be covered by a smaller expansion. We can then establish the following.

▶ **Lemma 2.** *Every UC2RPQ Γ is equivalent to the (potentially infinite) UCQ given by its set of minimal expansions.*

We can now provide our basic characterizations of boundedness.

▶ **Proposition 3.** *The following conditions are equivalent for each UC2RPQ Γ.*

1. *Γ is bounded.*
2. *There is $k \geq 1$ such that for every expansion λ of Γ there exists an expansion λ' of Γ with $\|\lambda'\| \leq k$ such that $\lambda \subseteq \lambda'$ (i.e., such that $\lambda' \to \lambda$).*
3. *Γ has finitely many minimal expansions.*

▶ **Example 4.** Consider the Boolean UCRPQ $\Gamma = \gamma_1 \vee \gamma_2$ over $\mathbb{A} = \{a, b, c, d\}$ from Example 1. To see that γ_1 is unbounded (the case of γ_2 is similar) we can apply Proposition 3. Indeed, the expansions of γ_1 corresponding to $\{\exists x, y\, (x \xrightarrow{ab^n c} y \wedge x \xrightarrow{adc} y) : n \geq 1\}$ are all minimal. On the other hand, Γ is bounded as its minimal expansions correspond to $\exists x, y\, (x \xrightarrow{abc} y \wedge x \xrightarrow{abc} y)$ and $\exists x, y\, (x \xrightarrow{adc} y \wedge x \xrightarrow{adc} y)$. ◀

4 Boundedness for Existentially Quantified RPQs

As a first application of Proposition 3, we study BOUNDEDNESS for CRPQs consisting of a single RPQ; that is, RPQs or existentially quantified RPQs. Let v, w be words over $\mathbb{A}$. Recall that a word v is a *prefix* [resp. *suffix* and *factor*] of w if $w \in v \cdot \mathbb{A}^*$ [resp. $w \in \mathbb{A}^* \cdot v$ and $w \in \mathbb{A}^* \cdot v \cdot \mathbb{A}^*$]. If in addition we have $v \neq w$, then we say that v is a *proper* prefix [resp. suffix and factor] of w. For a language $L \subseteq \mathbb{A}^*$, we define its *prefix-free* sub-language L_{pf} to be the set of words $w \in L$ such that w has no proper prefix in L. Similarly, we define L_{sf} and L_{ff} with respect to the suffix and factor relation. We have the following:

▶ **Proposition 5.** *The following statements hold.*

1. *An RPQ L is bounded iff L is finite.*
2. *A CRPQ $\exists y(x \xrightarrow{L} y)$ [resp. $\exists x(x \xrightarrow{L} y)$] with $x \neq y$ is bounded iff L_{pf} [resp. L_{sf}] is finite.*
3. *A Boolean CRPQ $\exists x, y(x \xrightarrow{L} y)$ with $x \neq y$ is bounded iff L_{ff} is finite.*

▶ **Theorem 6.** *The problem of, given an NFA accepting the language L, checking whether L_{pf} is finite is* PSPACE*-complete. The same holds if we replace L_{pf} by L_{sf} or L_{ff}.*

Proof. We focus on upper bounds, the lower bounds are in the appendix. Given an NFA $\mathcal{A}$ accepting the language L, we can construct an NFA $\mathcal{B}$ of polynomial size in $\mathcal{A}$ that accepts precisely those words that have a proper prefix in L. By complementing and intersecting with $\mathcal{A}$, we obtain an NFA $\mathcal{B}'$ of exponential size in $\mathcal{A}$ that accepts the language L_{pf}. Hence, we only need to check whether the language accepted by $\mathcal{B}'$ is finite, which can be done *on-the-fly* in NL w.r.t. $\mathcal{B}'$, and hence in PSPACE. The other two cases are analogous. ◀

By applying Theorem 6 and Proposition 5, we can now pinpoint the complexity of BOUNDEDNESS for CRPQs with a single RPQ.

▶ **Corollary 7.** *The following statements hold.*

1. BOUNDEDNESS *for RPQs is* NL*-complete.*
2. BOUNDEDNESS *for CRPQs of the form $\exists y(x \xrightarrow{L} y)$, with $x \neq y$, is* PSPACE*-complete. The same holds for CRPQs $\exists x(x \xrightarrow{L} y)$ and Boolean CRPQs $\exists x, y(x \xrightarrow{L} y)$, where $x \neq y$.*

It is not clear, though, how usual automata techniques, as the ones applied in the proof of Theorem 6, can be used to solve BOUNDEDNESS for more complex CRPQs. To solve this problem we develop an approach based on distance automata, as introduced next. Our approach also handles inverses and unions, thus dealing with arbitrary UC2RPQs.

5 Distance Automata

Distance automata [24] (equivalent to weighted automata over the $(\min, +)$-semiring [21], min-automata [12], or $\{\varepsilon, ic\}$-B-automata [17]) are an extension of finite automata which associate to each word in the language a natural number or "cost". They can be represented as non-deterministic finite automata with two sorts of transitions: costly and non-costly. For a given distance automaton, the cost of a run on a word is the number of costly transitions, and the cost of a word $w \in \mathbb{A}^*$ is the minimum cost of an accepting run on w. We will use this automaton model to encode boundedness as the problem of whether there is a uniform bound on the cost of words, known as the *limitedness problem*.

Formally, a *distance automaton* (henceforth *DA*) is a tuple $\mathcal{A} = (\mathbb{A}, Q, q_0, F, \delta)$, where $\mathbb{A}$ is a finite alphabet, Q is a finite set of states, $q_0 \in Q$ is the initial state, $F \subseteq Q$ is the set of finals states and $\delta \subseteq Q \times \mathbb{A} \times \{0, 1\} \times Q$ is the transition relation. A word $w \in \mathbb{A}^*$ is *accepted* by $\mathcal{A}$ if there is an *accepting run* of $\mathcal{A}$ on w, *i.e.*, a (possibly empty) sequence of transitions $\rho = (p_1, a_1, c_1, r_1) \cdots (p_n, a_n, c_n, r_n) \in \delta^*$ with the usual properties: (1) if $\rho = \varepsilon$ then $q_0 \in F$ and $w = \varepsilon$, (2) $p_1 = q_0$ and $r_n \in F$, (3) for every $1 \leq i < n$ we have $r_i = p_{i+1}$, and (4) $w = a_1 \cdots a_n$. The *cost* of the run ρ is $\mathit{cost}(\rho) = c_1 + \cdots + c_n$ (or 0 if $\rho = \varepsilon$); and the cost $\mathit{cost}_{\mathcal{A}}(w)$ of a word w accepted by $\mathcal{A}$ is the minimum cost of an accepting run of $\mathcal{A}$ on w. For convenience, we assume the cost of words not accepted by $\mathcal{A}$ to be 0.

The *limitedness problem* for DA is defined as follows: given a DA $\mathcal{A}$, determine whether $\sup_{w \in \mathbb{A}^*} \mathit{cost}_{\mathcal{A}}(w) < \infty$. This problem is known to be PSPACE-complete.

▶ **Theorem 8** ([28, 29])**.** *The following statements hold:*

1. *The limitedness problem for DA is* PSPACE*-complete.*
2. *If a DA with n states is limited, then $\sup_{w \in \mathbb{A}^*} \mathit{cost}_{\mathcal{A}}(w) \leq 2^{O(n^3)}$.*

We use two extensions of DA: *alternating* and *two-way*. Two-way DA is defined as for NFA, extending the cost function accordingly. The cost of a word is still the minimum over the cost of all (potentially infinitely many) runs. Alternating DA is defined as usual by having two sorts of states: universal and existential. Existential states can be seen as computing the minimum among the cost of all possible continuations of the run, and universal states as computing the maximum (or supremum if the automaton is also two-way). As we will see, these extensions preserve the above PSPACE upper bound for the limitedness problem.

Formally, an *alternating two-way DA with epsilon transitions* (A2DA$^\varepsilon$) over $\mathbb{A}$ is a tuple $\mathcal{A} = (\mathbb{A}, Q_\exists, Q_\forall, q_0, F, \delta)$ is an A2DA$^\varepsilon$ if $q_0 \in Q_\exists$, $F \subseteq Q_\exists$ and

$$\delta \subseteq (Q_\exists \cup Q_\forall) \times (\mathbb{A}^\pm \cup \{\varepsilon\}) \times \{end, \overline{end}\} \times \{0,1\} \times (Q_\exists \cup Q_\forall);$$

where *end* indicates that after reading the letter we arrive at the end of the word (*i.e.*, either the leftmost or the rightmost end) and $\overline{end}$ indicates that we do not. When the automaton $\mathcal{A}$ is two-way, it is convenient to think of its head as being *between* the letter positions of the word, so an *end*-flagged transition can be applied only if it moves the head to be right before the first letter of the word, or right after the last one.

For any given word $w \in \mathbb{A}^*$, consider the edge-labelled graph $G_{\mathcal{A},w} = (V, E)$ over δ, where $V = Q \times \{0, \dots, |w|\}$, with $Q = Q_\exists \cup Q_\forall$, and $E \subseteq V \times \delta \times V$ consists of all edges $(q,i) \xrightarrow{(q,a,e,c,p)} (p,j)$ such that $e = end$ iff $j = 0$ or $j = |w|$ and either (a) $i < |w|$, $a = w[i+1]$, and $j = i+1$; (b) $i > 0$, $a = (w[i])^{-1}$, and $j = i-1$; or (c) $a = \varepsilon$ and $j = i$.

An *accepting run of $\mathcal{A}$ on w from* $(q,i) \in Q \times \{0, \dots, |w|\}$ is a finite (possibly empty) edge-labelled directed rooted tree[1] t over δ and a labelling h from the nodes of t to the nodes of $G_{\mathcal{A},w}$, such that if t is empty then $q \in F$, and otherwise h maps the root of t to (q,i), every leaf of t to $F \times \{0, \dots, |w|\}$, and for every node x of t:

- if (x, α, y) is an (labeled) edge in t for some y, then $(h(x), \alpha, h(y))$ is an edge in $G_{\mathcal{A},w}$;
- if $h(x) \in Q_\forall \times \{0, \dots, |w|\}$, then for every edge $(h(x), \alpha, c)$ in $G_{\mathcal{A},w}$, there is an edge (x, α, y) in t so that $h(y) = c$;
- if $h(x) \in Q_\exists \times \{0, \dots, |w|\}$, then x has at most one child.

Each branch of t with label $(q_1, a_1, e_1, c_1, p_1), \dots, (q_n, a_n, e_n, c_n, p_n)$ has an associated cost of $c_1 + \dots + c_n$; and the cost associated with t is the maximum among the costs of its branches, or 0 if t is empty. The cost $cost_{\mathcal{A}}(w, q, i)$ is the minimum cost of an accepting run on w from (q,i), or 0 if none exists; $cost_{\mathcal{A}}(w)$ is defined as $cost_{\mathcal{A}}(w, q_0, 0)$.

An A2DA$^\varepsilon$ with $\delta \subseteq Q \times (\mathbb{A} \cup \{\varepsilon\}) \times \{end, \overline{end}\} \times \{0,1\} \times Q$ is an *alternating DA* with ε transitions (ADA$^\varepsilon$). An A2DA$^\varepsilon$ with $Q_\forall = \emptyset$ is a *two-way DA* with ε transitions (2DA$^\varepsilon$). An A2DA with both the aforementioned conditions is (equivalent to) a DA with ε transitions (DA$^\varepsilon$). Notice that in the last two cases, accepting runs can be represented as words from δ^* rather than trees. By A2DA (resp., ADA, 2DA, DA) we denote an A2DA$^\varepsilon$ (resp., ADA$^\varepsilon$, 2DA$^\varepsilon$, DA$^\varepsilon$) with no ε-transitions. Note that DA as just defined is in every sense equivalent to the distance automata model we have defined at the beginning of this section – this is why we overload the same "DA" name.

We first observe that 2DA can be transformed into DA while preserving both the language and limitedness problems by adapting the standard "crossing sequence" construction for translating 2NFA into NFA [34]. This fact will be useful for proving the EXPSPACE upper bound for BOUNDEDNESS of general UC2RPQs in Section 6.

▶ **Proposition 9.** *There is an exponential time procedure which for every 2DA $\mathcal{A}$ over $\mathbb{A}$ produces a DA $\mathcal{B}$ over $\mathbb{A}$ such that the languages accepted by $\mathcal{A}$ and $\mathcal{B}$ are the same, and $cost_{\mathcal{B}}(w) \leq cost_{\mathcal{A}}(w) \leq f(cost_{\mathcal{B}}(w))$ for every $w \in \mathbb{A}^*$, where f is a polynomial function that depends on the number of states of $\mathcal{A}$.*

[1] That is, a tree-shaped finite edge-labelled graph over δ with edges directed in the root-to-leaf sense.

Recall that the universality problem for NFAs is known to be PSPACE-complete [27]; and that this bound actually extends to two-way and even alternating automata. We show that, likewise, the limitedness problem remains in PSPACE for A2DA$^{\varepsilon}$. This result will be useful to show in Section 7 that BOUNDEDNESS for the class of acyclic UC2RPQs of bounded thickness is in PSPACE.

▶ **Theorem 10.** *The limitedness problem for A2DA*$^{\varepsilon}$ *is* PSPACE*-complete.*

The novelty of this result is the PSPACE upper bound. In fact, decidability follows from known results, and in particular [7, Theorem 14] claims EXPTIME-membership in the more challenging setup of infinite trees. However, this is obtained via an involved construction spanning through several papers. The proof of Theorem 10, instead, is obtained by the composition of the following reductions:

$$\text{lim. A2DA}^{\varepsilon} \xrightarrow{(1)} \text{lim. A2DA} \xrightarrow{(2)} \text{lim. 2DA} \xrightarrow{(3)} \text{lim. ADA}^{\varepsilon} \xrightarrow{(4)} \text{lim. ADA} \xrightarrow{(5)} \text{lim. DA}.$$

Reductions (1), (3) and (4) are in polynomial time, while reductions (2) and (5), which are basically the same, are in exponential time. Specifically, reductions (2) and (5) preserve the statespace but the size of the alphabet grows exponentially in the number of states and linearly in the size of the source alphabet. However, the alphabet and transition set resulting from these reductions can be succinctly described: letters are encoded in polynomial space, and checking for membership in the transition set is polynomial time computable.

In summary, the composition (1)+(2)+(3)+(4)+(5) yields a DA with the following characteristics: (i) it has a polynomial number of states Q; (ii) it runs on an exponential alphabet $\mathbb{A}$ –and every letter is encoded in polynomial space–; and (iii) one can check in polynomial time whether a tuple $t \in Q \times \mathbb{A} \times \{end, \overline{end}\} \times \{0, 1\} \times Q$ is in its transition relation. This, coupled with Theorem 8, item (2) (which offers a bound depending only on the number of states), provides a polynomial space algorithm for the limitedness of A2DA$^{\varepsilon}$: We can non-deterministically check the existence of a word with cost greater than the single exponential bound N using only polynomial space, by guessing one letter at a time and keeping the set of reachable states together with the associated costs, where each cost is encoded in binary using polynomial space if it is smaller than N, or with a "∞" flag otherwise. The algorithm accepts if at least one final state is reached and the costs of all reachable final states are marked ∞. Since NPSPACE =PSPACE (Savitch's Theorem), Theorem 10 follows.

We now provide a brief description of the reductions used in the proof of Theorem 10.

(1) From A2DA$^{\varepsilon}$ to A2DA. This is a trivial reduction obtained by simulating ε-transitions by reading $a \cdot a^{-1}$ for some $a \in \mathbb{A}$.

(2) From A2DA to 2DA. Given an A2DA $\mathcal{A} = (\mathbb{A}, Q_{\forall}, Q_{\exists}, q_0, F, \delta)$, we build a 2DA $\mathcal{B}$ over a larger alphabet $\mathbb{B}$, where we trade alternation for extra alphabet letters. The alphabet $\mathbb{B}$ consists of triples $(f^{\rightarrow}, a, f^{\leftarrow})$, where $a \in \mathbb{A}$ and $f^{\rightarrow}, f^{\leftarrow} : Q_{\forall} \rightarrow \delta$. The idea is that $f^{\rightarrow}, f^{\leftarrow}$ are "choice functions" for the alternation: whenever we are to the left (resp., right) of a position of the word labelled $(f^{\rightarrow}, a, f^{\leftarrow})$ in state $q \in Q_{\forall}$, instead of exploring all transitions departing from q and taking the maximum cost over all such runs (this is what alternation does in $\mathcal{A}$), $\mathcal{B}$ chooses to just take the transition $f^{\rightarrow}(q)$ (resp., $f^{\leftarrow}(q)$). Note that $\mathbb{B}$ is exponential in the number of states but not in the size of $\mathbb{A}$. In this way, we build a 2DA $\mathcal{B}$ having the same set of states as $\mathcal{A}$ but with a transition function which is essentially deterministic on the states of $Q_{\forall}$. In the end we obtain that

- for every $w \in \mathbb{B}^*$, $cost_{\mathcal{B}}(w) \leq cost_{\mathcal{A}}(w_{\mathbb{A}})$; and
- for every $w \in \mathbb{A}^*$ there is $\widetilde{w} \in \mathbb{B}^*$ so that $\widetilde{w}_{\mathbb{A}} = w$ and $cost_{\mathcal{A}}(w) = cost_{\mathcal{B}}(\widetilde{w})$,

where $w_{\mathbb{A}}$ and $\widetilde{w}_{\mathbb{A}}$ denote the projections onto the alphabet $\mathbb{A}$. This implies that the limitedness problem is preserved.

(3) From 2DA to ADA$^\varepsilon$. We show a polynomial-time translation from 2DA to ADA$^\varepsilon$ which preserves limitedness. In the case of finite automata, there are language-preserving reductions from 2NFA to AFA with a quadratic blowup in the statespace [9, 32]. However, these translations, when applied blindly to reduce from 2DA to ADA$^\varepsilon$, preserve neither the cost semantics nor the limitedness of languages. On the other hand, [10] shows an involved construction that results in a reduction from 2DA to ADA$^\varepsilon$ on *infinite trees*, which preserves limitedness but it is not polynomial in the number of states. We show a translation from 2DA to ADA$^\varepsilon$ which serves our purpose: it preserves limitedness and it is polynomial time computable. The translation is close to the language-preserving reduction from 2NFA to AFA of [32], upgraded to take into account the cost of different alternation branches, somewhat in the same spirit as the *history summaries* from [10].

(4) From ADA$^\varepsilon$ to ADA. This is a straightforward polynomial time reduction which preserves limitedness but – as opposed to (1) – does not preserve the language: we need to add an extra letter to the alphabet in order to make the reduction work in polynomial time.

(5) From ADA to DA. This is exactly the same reduction as (2), noticing that the alphabet will still be single exponential in the original A2DA$^\varepsilon$.

6 Complexity of Boundedness for UC2RPQs

Here we show that BOUNDEDNESS for UC2RPQs is EXPSPACE-complete. We do so by applying distance automata results presented in the previous section on top of the semantic characterizations presented in Section 3. The lower bound applies even for CRPQs. We further show that there is a triple exponential tight bound for the size of the equivalent UCQ of a UC2RPQ (and even CRPQ), whenever this exists. This is summarized in the following theorem. If Γ is a UC2RPQ, we write $\|\Gamma\|$ for the length of an arbitrary reasonable encoding of Γ – in particular, encodings in which regular languages are described through NFA or regular expressions.

▶ **Theorem 11.** *The following statements hold.*

1. BOUNDEDNESS *for UC2RPQs is* EXPSPACE*-complete. The problem remains* EXPSPACE*-hard even for Boolean CRPQs.*
2. *If a UC2RPQ Γ is bounded, there is a UCQ Φ that is equivalent to Γ and such that Φ has at most triple-exponentially many CQs, each one of which is at most of double exponential size with respect to $\|\Gamma\|$.*
3. *There is a family $\{\Gamma_n\}_{n\geq 1}$ of Boolean CRPQs such that for each $n \geq 1$ it is the case that: (1) $\|\Gamma_n\| = O(n)$, (2) Γ_n is bounded, and (3) every UCQ that is equivalent to Γ_n has at least triple-exponentially many CQs with respect to n.*

6.1 Upper bounds

Our upper bound proof builds on top of techniques developed by Calvanese et al. [14] for studying the *containment problem for UC2RPQs*: Given UC2RPQs Γ, Γ', is it the case that $\Gamma \subseteq \Gamma'$? It is shown in [14] that from Γ, Γ' it is possible to construct exponentially sized NFAs $\mathcal{A}_{\Gamma,\Gamma'}$ and $\mathcal{A}'_{\Gamma,\Gamma'}$, such that $\Gamma \subseteq \Gamma'$ iff there is a word in $\mathcal{A}_{\Gamma,\Gamma'} \cap \overline{\mathcal{A}'_{\Gamma,\Gamma'}}$. It is a well-known result that the latter is solvable in NL in the combined size of $(\mathcal{A}_{\Gamma,\Gamma'}, \overline{\mathcal{A}'_{\Gamma,\Gamma'}})$, *i.e.*, in EXPSPACE. We modify this construction to study the boundedness of a given UC2RPQ Γ. In particular, we construct from Γ in exponential time a DA $\mathcal{D}_\Gamma$ such that Γ is bounded iff $\mathcal{D}_\Gamma$ is limited. The result then follows from Theorem 8, which establishes that limitedness for $\mathcal{D}_\Gamma$ can be solved in polynomial space on the number of its states, and thus in EXPSPACE.

▶ **Proposition 12.** *There is a single exponential time procedure that takes as input a UC2RPQ Γ and constructs a DA $\mathcal{D}_\Gamma$ such that Γ is bounded iff $\mathcal{D}_\Gamma$ is limited.*

Proof. Similarly as done in [14], the DA $\mathcal{D}_\Gamma$ will run over encodings of expansions of the UC2RPQ Γ, *i.e.*, words over the alphabet $\mathbb{A}_1 := \mathbb{A}^\pm \cup \mathcal{V} \cup \{\$\}$, where $\mathbb{A}$ is the alphabet of Γ, $\mathcal{V}$ is the set of variables of Γ, and \$ is a fresh symbol. If $\gamma = \exists \bar{z} \bigwedge_{1 \le i \le m} (x_i \xrightarrow{L_i} y_i)$ is a C2RPQ in Γ and λ is the expansion of γ obtained by expanding each $x_i \xrightarrow{L_i} y_i$ into an oriented path π_i from x_i to y_i with label $w_i \in L_i$, then we encode λ as the word

$$w_\lambda = \$x_1 w_1 y_1 \$ x_2 w_2 y_2 \$ \cdots \$ x_m w_m y_m \$ \;\in\; \mathbb{A}_1^*$$

Note how the subword $x_i w_i y_i$ encodes the oriented path π_i. Every position $j \in \{1, \dots, |w_\lambda|\}$ with $w_\lambda[j] \neq \$$ represents a variable in λ: either x_i or y_i if $w_\lambda[j] = x_i$ or $w_\lambda[j] = y_i$, respectively; or the $(\ell+1)$-th variable in the oriented path π_i if $w_\lambda[j]$ is the ℓ-th symbol in the subword w_i. Hence different positions in w_λ could represent the same variable in λ, *e.g.*, in the encoding $\$xabcy\$$, the $5th$ position containing a "c" and the $6th$ position containing a "y", represent the same variable, namely, the last vertex y of the oriented path. It is then easy to build, in polynomial time, an NFA $\mathcal{A}_1$ over $\mathbb{A}_1$ recognizing the language of all such encodings of expansions of Γ. Our automaton $\mathcal{D}_\Gamma$ is the product of $\mathcal{A}_1$ and the DA $\mathcal{C}_\Gamma$ defined below. In particular, $\mathcal{D}_\Gamma$ is limited iff $\mathcal{C}_\Gamma$ is limited over words of the form w_λ, for λ an expansion of Γ.

Fix a disjunct γ of Γ. As in [14], we consider words over the alphabet $\mathbb{A}_2 := \mathbb{A}_1 \times (2^{\mathcal{V}} \cup \{\#\})$ of the form $(\ell_1, \alpha_1) \cdots (\ell_n, \alpha_n)$, such that $w_\lambda = \ell_1 \cdots \ell_n$, for some expansion λ of Γ, and the α_i's are *valid γ-annotations*, *i.e.*, (1) $\alpha_i = \#$ if $\ell_i = \$$, (2) $\alpha_1, \dots, \alpha_n \in 2^{\mathcal{V}}$ induce a partition of the variable set $\mathcal{V}_\gamma$ of γ, and (3) for each free variable $x \in \mathcal{V}_\gamma$ there is some (ℓ_i, α_i) such that $\ell_i = x$ and $x \in \alpha_i$. It is easy to construct an NFA $\mathcal{B}_1^\gamma$ of exponential size that given $w = (\ell_1, \alpha_1) \cdots (\ell_n, \alpha_n)$ with $w_\lambda = \ell_1 \cdots \ell_n$, checks if the α_i's are valid γ-annotations. Note that if the latter holds, then the annotations encode a mapping h_w from $\mathcal{V}_\gamma$ to the variables of λ such that $h_w(\bar{x}) = \bar{x}$, where $\bar{x}$ are the free variables of γ.

Now, given $w = (\ell_1, \alpha_1)(\ell_2, \alpha_2) \cdots (\ell_n, \alpha_n)$ with $w_\lambda = \ell_1 \cdots \ell_n$ and the α_i's being valid γ-annotations, it is shown in [14] that one can construct in polynomial time a 2NFA $\mathcal{B}_2^\gamma$ that checks the existence of an expansion λ' of γ and a homomorphism h from λ' to λ consistent with h_w. For each atom $x \xrightarrow{L} y$ of γ, the automaton $\mathcal{B}_2^\gamma$ guesses an oriented path π in λ from $h_w(x)$ to $h_w(y)$ with label $w' \in L$, directly over the encoding w_λ starting at a position j_x and ending at a position j_y in $\{0, \dots, n\}$ (recall that the head moves in $\{0, \dots, n\}$) with $j_x, j_y > 0$, $w[j_x] = (\ell, \alpha)$, $w[j_y] = (\ell', \alpha')$, $x \in \alpha$ and $y \in \alpha'$. Note that we have two types of transitions: (1) transitions that consume $a \in \mathbb{A}^\pm$ and actually guess an atom of π, and (2) transitions to "jump" from position j to j' in $\{0, \dots, n\}$ representing *equivalent* variables of λ. The latter means that $j, j' > 0$ and either $w_\lambda[j]$ and $w_\lambda[j']$ represents exactly the *same* variable of λ, or $w_\lambda[j]$ and $w_\lambda[j']$ represent variables z, z' of λ such that $z =_\lambda^* z'$, where $=_\lambda^*$ is the reflexive-transitive closure of the relation induced by the equality atoms in λ.

Let $\mathcal{D}_2^\gamma$ be the 2DA obtained from the 2NFA $\mathcal{B}_2^\gamma$ by setting to 0 and 1 the cost of transitions of type (2) and (1), respectively. Hence, for a word w such that the projection of w to $\mathbb{A}_1$ is w_λ, and the one to $(2^{\mathcal{V}} \cup \{\#\})$ is a valid γ-annotation, we have that $cost_{\mathcal{D}_2^\gamma}(w)$ is precisely the minimum size of an expansion λ' that can be mapped to λ via a homomorphism compatible with h_w. By Proposition 9, we can construct in exponential time in $\mathcal{D}_2^\gamma$ a DA $\mathcal{C}_2^\gamma$ accepting the same language as $\mathcal{D}_2^\gamma$ and having an exponential number of states, so that for every word w', we have $cost_{\mathcal{C}_2^\gamma}(w') \le cost_{\mathcal{D}_2^\gamma}(w') \le f(cost_{\mathcal{C}_2^\gamma}(w'))$ for some polynomial function f. Let $\exists\mathcal{C}^\gamma$ be the result of taking the product of $\mathcal{B}_1^\gamma$ and $\mathcal{C}_2^\gamma$ and then projecting over the alphabet $\mathbb{A}_1$.

For every expansion λ of Γ, if λ' is a minimal size expansion of γ such that $\lambda' \to \lambda$, then we obtain that $cost_{\exists\mathcal{C}^\gamma}(w_\lambda) \leq \|\lambda'\| \leq f(cost_{\exists\mathcal{C}^\gamma}(w_\lambda))$. We define our desired $\mathcal{C}_\Gamma$ to be the union of $\exists\mathcal{C}^\gamma$ over all γ in Γ. We have that for every expansion λ, if λ_{min} is a minimal size expansion of Γ such that $\lambda_{min} \to \lambda$, then $cost_{\mathcal{C}_\Gamma}(w_\lambda) \leq \|\lambda_{min}\| \leq f(cost_{\mathcal{C}_\Gamma}(w_\lambda))$. By Proposition 3, item (2), Γ is bounded iff $\|\lambda_{min}\|$ is bounded over all λ. The latter condition holds iff $\mathcal{C}_\Gamma$ is limited over words w_λ, for all expansion λ. By definition, the latter is equivalent to $\mathcal{D}_\Gamma$ being limited. Summing up, we obtain that Γ is bounded iff $\mathcal{D}_\Gamma$ is limited, as required. Note that the whole construction can be done in exponential time. ◀

As a corollary to Proposition 12 and Theorem 8 we obtain the desired upper bound for part (1) of Theorem 11.

▶ **Corollary 13.** Boundedness *for UC2RPQs is in* ExpSpace.

Size of equivalent UCQs. Here we prove part (2) of Theorem 11. Since Γ is bounded we have from Proposition 12 that $\mathcal{D}_\Gamma$ is limited. Then, from Theorem 8 we obtain that the maximum cost that it takes $\mathcal{D}_\Gamma$ over a word is N, where N is exponential in the number of states of $\mathcal{D}_\Gamma$, and thus double exponential in $\|\Gamma\|$ by construction. Therefore, for every expansion λ of Γ, if λ_{min} is a minimal size expansion Γ such that $\lambda_{min} \to \lambda$, then $\|\lambda_{min}\| \leq f(N)$, where f is the polynomial function of the proof of Proposition 12. In particular, all minimal expansions of Γ are of size $\leq f(N)$. By Lemma 2, the UC2RPQ Γ is equivalent to the union of all its minimal expansions. The number of such minimal expansions is thus at most exponential in $f(N)$, and hence triple exponential in $\|\Gamma\|$.

6.2 Lower bounds

We reduce from the 2^n-tiling problem, that is, a tiling problem restricted to 2^n many columns, which is ExpSpace-complete (see, *e.g.*, [14]). We show that for every 2^n-tiling problem T there is a CRPQ γ, computable in polynomial time from T, whose number of minimal expansions is essentially the number of solutions to T in the following sense.

▶ **Lemma 14.** *For every 2^n-tiling problem T with m solutions there is a Boolean CRPQ γ, computable in polynomial time from T, such that the number of minimal expansions of γ is $O((g(|T|) + m)^{n+1})$ and $\Omega(m)$, for some double exponential function g. Further, γ consists of a Boolean CRPQ of the form $\exists x, y \bigwedge_{0 \leq i \leq n}(x \xrightarrow{L_i} y)$, where each L_i is given as a regular expression.*

As a corollary, this yields an ExpSpace lower bound for the boundedness problem (part (1) of Theorem 11), as well as a triple exponential lower bound for the size of the UCQ equivalent to any bounded CRPQ (part (3) of Theorem 11), since one can produce 2^n-tiling problems having triple-exponentially many solutions.

7 Better-behaved Classes of UC2RPQs

Here we present two restrictions of UC2RPQs that exhibit a better behavior in terms of the complexity of Boundedness than the general case, namely, *acyclic UC2RPQs of bounded thickness* and *strongly connected UCRPQs*. The improved bounds are PSpace and Π_2^P, respectively, which turn out to be optimal.

Acyclic UC2RPQs of Bounded Thickness. For any two distinct variables x, y of a C2RPQ γ, we denote by $\text{Atoms}_\gamma(x, y)$ the set of atoms in γ of the form $x \xrightarrow{L} y$ or $y \xrightarrow{L} x$. The *thickness* of a C2RPQ γ is the maximum cardinality of a set of the form $\text{Atoms}_\gamma(x, y)$, for x, y variables of γ with $x \neq y$. The thickness of a UC2RPQ Γ is the maximum thickness over all the C2RPQs in Γ. The *underlying undirected graph* of γ has as vertex set the set of variables of γ and contains an edge $\{x, y\}$ iff $x \neq y$ and $\text{Atoms}_\gamma(x, y) \neq \emptyset$. A C2RPQ γ is *acyclic* if its underlying undirected graph is an acyclic graph (*i.e.*, a forest). A UC2RPQ Γ is acyclic if each C2RPQ in Γ is.

We show next that Boundedness for acyclic UC2RPQs of bounded thickness is PSpace-complete. These classes of UC2RPQs have been previously studied in the literature [4, 5]. In particular, it follows from [5, Theorem 4.2] that the containment problem for the acyclic UC2RPQs of bounded thickness is PSpace-complete, and hence Theorem 15 below shows that Boundedness is not more costly than containment for these classes.

▶ **Theorem 15.** *Fix $k \geq 1$. The problem* Boundedness *is* PSpace*-complete for acyclic UC2RPQs of thickness at most k.*

Proof (sketch). The lower bound follows directly from PSpace-hardness of Boundedness for RPQs (see Corollary 7). For the PSpace upper bound, we follow a similar strategy as in the case of arbitrary UC2RPQs (Section 6.1), *i.e.*, we reduce boundedness of Γ to DA limitedness. The main difference is that, since Γ is acyclic, we can exploit the power of alternation and construct an A2DA$^\varepsilon$ $\mathcal{B}$ (instead of a 2DA, as in the proof of Proposition 12), such that Γ is bounded iff $\mathcal{B}$ is limited. The constant upper bound on the thickness of Γ implies that $\mathcal{B}$ is actually of polynomial size. The result follows then as limitedness of an A2DA$^\varepsilon$ can be decided in PSpace in virtue of Theorem 10. ◀

Both conditions in Theorem 15, *i.e.*, acyclicity and bounded thickness, are necessary. Indeed, it follows from Lemma 14 that Boundedness is ExpSpace-hard even for:

- Boolean acyclic CRPQs.
- Boolean CRPQs of thickness one, whose underlying undirected graph is of *treewidth* two. Recall that the treewidth is a measure of how much a graph resembles a tree (cf., [20]) – acyclic graphs are precisely the graphs of treewidth one.

Indeed, the CRPQs of the form $\exists x, y \bigwedge_i (x \xrightarrow{L_i} y)$ used in Lemma 14 are Boolean and acyclic (but have unbounded thickness). Replacing each $(x \xrightarrow{L_i} y)$ with $(x \xrightarrow{\varepsilon} z_i) \wedge (z_i \xrightarrow{L_i} y)$, yields an equivalent CRPQ of thickness one whose underlying undirected graph has treewidth two.

Strongly Connected UCRPQs. We conclude this section with an even better behaved class of CRPQs in terms of Boundedness. Unlike the previous case, the definition of this class depends on the underlying *directed* graph of a CRPQ γ. This contains a directed edge from variable x to y iff there is an atom in γ of the form $x \xrightarrow{L} y$. A CRPQ γ is *strongly connected* if its underlying directed graph is strongly connected, *i.e.*, every pair of variables is connected by some directed path. A UCRPQ Γ is strongly connected if every CRPQ in Γ is. We can then establish the following.

▶ **Theorem 16.** Boundedness *is Π_2^P-complete for strongly connected UCRPQs.*

8 Discussion and Future Work

The main conclusion of our work is that techniques previously used in the study of containment of UC2RPQs can be naturally leveraged to pinpoint the complexity of Boundedness by using DA instead of NFA. This, however, requires extending results on limitedness to alternating and two-way DA. For all the classes of UC2RPQs studied in the paper we show in fact that the complexity of Boundedness coincides with that of the containment problem. We leave open what is the exact size of UCQ rewritings for the classes of acyclic UC2RPQs of bounded thickness and the strongly connected UCRPQs that are bounded.

The most natural next step is to study Boundedness for the class of *regular queries* (RQs), which are the closure of UC2RPQs under binary transitive closure. RQs are one of the most powerful recursive languages for which containment is decidable in elementary time. In fact, containment of RQs has been proved to be 2EXPSPACE-complete by applying sophisticated techniques based on NFA [33]. We will study if it is possible to settle the complexity of Boundedness for RQs with the help of DA techniques.

Another interesting future line of work is the study of Boundedness for UC2RPQs based on the restricted classes of regular expressions often found in practical applications [13]. As it has been shown lately, the complexity of some query evaluation problems is alleviated under this restriction [30], and it would be nice to see if the same holds for the boundedness problem. This would be good news for the applicability of boundedness techniques in practical applications. In fact, it would be an indication that the high complexity lower bounds obtained in this paper are mostly witnessed by complicated interactions between regular expressions not commonly arising in practice.

References

1 Renzo Angles, Marcelo Arenas, Pablo Barceló, Aidan Hogan, Juan L. Reutter, and Domagoj Vrgoč. Foundations of Modern Query Languages for Graph Databases. *ACM Computing Surveys*, 50(5):68:1–68:40, 2017.

2 Pablo Barceló. Querying graph databases. In *ACM Symposium on Principles of Database Systems (PODS)*, pages 175–188, 2013.

3 Pablo Barceló, Gerald Berger, Carsten Lutz, and Andreas Pieris. First-Order Rewritability of Frontier-Guarded Ontology-Mediated Queries. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1707–1713, 2018.

4 Pablo Barceló, Miguel Romero, and Moshe Y. Vardi. Does Query Evaluation Tractability Help Query Containment? In *ACM Symposium on Principles of Database Systems (PODS)*, pages 188–199, 2014.

5 Pablo Barceló, Miguel Romero, and Moshe Y. Vardi. Semantic Acyclicity on Graph Databases. *SIAM Journal on computing*, 45(4):1339–1376, 2016.

6 Michael Benedikt, Pierre Bourhis, and Michael Vanden Boom. A Step Up in Expressiveness of Decidable Fixpoint Logics. In *Annual IEEE Symposium on Logic in Computer Science (LICS)*, pages 817–826, 2016.

7 Michael Benedikt, Balder ten Cate, Thomas Colcombet, and Michael Vanden Boom. The Complexity of Boundedness for Guarded Logics. In *Annual IEEE Symposium on Logic in Computer Science (LICS)*, pages 293–304. IEEE Computer Society Press, 2015. doi:10.1109/LICS.2015.36.

8 Meghyn Bienvenu, Peter Hansen, Carsten Lutz, and Frank Wolter. First Order-Rewritability and Containment of Conjunctive Queries in Horn Description Logics. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 965–971, 2016.

9 Jean-Camille Birget. State-complexity of finite-state devices, state compressibility and incompressibility. *Mathematical systems theory*, 26(3):237–269, 1993.

10 Achim Blumensath, Thomas Colcombet, Denis Kuperberg, Pawel Parys, and Michael Vanden Boom. Two-way cost automata and cost logics over infinite trees. In *Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), CSL-LICS '14*, pages 16:1–16:9. ACM Press, 2014. `doi:10.1145/2603088.2603104`.

11 Achim Blumensath, Martin Otto, and Mark Weyer. Decidability Results for the Boundedness Problem. *Logical Methods in Computer Science (LMCS)*, 10(3), 2014.

12 Mikołaj Bojańczyk and Szymon Toruńczyk. Deterministic Automata and Extensions of Weak MSO. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FST&TCS)*, volume 4 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 73–84. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2009. `doi:10.4230/LIPIcs.FSTTCS.2009.2308`.

13 Angela Bonifati, Wim Martens, and Thomas Timm. An Analytical Study of Large SPARQL Query Logs. *PVLDB*, 11(2):149–161, 2017.

14 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.

15 Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Moshe Y. Vardi. Rewriting of Regular Expressions and Regular Path Queries. *Journal of Computer and System Sciences (JCSS)*, 64(3):443–465, 2002.

16 Ashok K. Chandra and Philip M. Merlin. Optimal Implementation of Conjunctive Queries in Relational Data Bases. In *Symposium on Theory of Computing (STOC)*, pages 77–90, 1977.

17 Thomas Colcombet. The Theory of Stabilisation Monoids and Regular Cost Functions. In *International Colloquium on Automata, Languages and Programming (ICALP)*, volume 5556 of *Lecture Notes in Computer Science*, pages 139–150. Springer, 2009. `doi:10.1007/978-3-642-02930-1_12`.

18 Thomas Colcombet and Christof Löding. The Nesting-Depth of Disjunctive μ-Calculus for Tree Languages and the Limitedness Problem. In *EACSL Annual Conference on Computer Science Logic (CSL)*, pages 416–430, 2008.

19 Stavros S. Cosmadakis, Haim Gaifman, Paris C. Kanellakis, and Moshe Y. Vardi. Decidable Optimization Problems for Database Logic Programs (Preliminary Report). In *Symposium on Theory of Computing (STOC)*, pages 477–490, 1988.

20 Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2012.

21 Manfred Droste, Werner Kuich, and Heiko Vogler. *Handbook of weighted automata*. Springer Science & Business Media, 2009.

22 Haim Gaifman, Harry G. Mairson, Yehoshua Sagiv, and Moshe Y. Vardi. Undecidable Optimization Problems for Database Logic Programs. *J. ACM*, 40(3):683–713, 1993.

23 Peter Hansen, Carsten Lutz, Inanç Seylan, and Frank Wolter. Efficient Query Rewriting in the Description Logic EL and Beyond. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3034–3040, 2015.

24 Kosaburo Hashiguchi. Limitedness theorem on finite automata with distance functions. *Journal of Computer and System Sciences (JCSS)*, 24(2):233–244, 1982.

25 Gerd G. Hillebrand, Paris C. Kanellakis, Harry G. Mairson, and Moshe Y. Vardi. Tools for Datalog Boundedness. In *ACM Symposium on Principles of Database Systems (PODS)*, pages 1–12, 1991.

26 Daniel Kirsten. Distance desert automata and the star height problem. *Informatique Théorique et Applications (ITA)*, 39(3):455–509, 2005.

27 Dexter Kozen. Lower Bounds for Natural Proof Systems. In *Annual Symposium on Foundations of Computer Science (FOCS)*, pages 254–266, 1977.

28 Hing Leung. Limitedness Theorem on Finite Automata with Distance Functions: An Algebraic Proof. *Theoretical Computer Science (TCS)*, 81(1):137–145, 1991. doi:10.1016/0304-3975(91)90321-R.

29 Hing Leung and Viktor Podolskiy. The limitedness problem on distance automata: Hashiguchi's method revisited. *Theoretical Computer Science (TCS)*, 310(1-3):147–158, 2004. doi:10.1016/S0304-3975(03)00377-3.

30 Wim Martens and Tina Trautner. Evaluation and Enumeration Problems for Regular Path Queries. In *International Conference on Database Theory (ICDT)*, pages 19:1–19:21, 2018.

31 Jeffrey F. Naughton. Data Independent Recursion in Deductive Databases. *Journal of Computer and System Sciences (JCSS)*, 38(2):259–289, 1989.

32 Nir Piterman and Moshe Y. Vardi. From bidirectionality to alternation. *Theoretical Computer Science (TCS)*, 295:295–321, 2003. doi:10.1016/S0304-3975(02)00410-3.

33 Juan L. Reutter, Miguel Romero, and Moshe Y. Vardi. Regular Queries on Graph Databases. *Theoretical Computer Science (TCS)*, 61(1):31–83, 2017.

34 John C. Shepherdson. The reduction of two-way automata to one-way automata. *IBM Journal of Research and Development*, 3(2):198–200, 1959.

35 Michael Vanden Boom. Weak Cost Monadic Logic over Infinite Trees. In *International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 580–591, 2011.

36 Michael Vanden Boom. *Weak cost automata over infinite trees.* PhD thesis, University of Oxford, UK, 2012.