The Complexity of Reverse Engineering Problems for Conjunctive Queries

Pablo Barceló[1] **and Miguel Romero**[2]

1 **Center for Semantic Web Research & Department of Computer Science, University of Chile, Santiago de Chile, Chile**
pbarcelo@dcc.uchile.cl

2 **Center for Semantic Web Research & Department of Computer Science, University of Chile, Santiago de Chile, Chile**
mromero@dcc.uchile.cl

Abstract

Reverse engineering problems for conjunctive queries (CQs), such as query by example (QBE) or definability, take a set of user examples and convert them into an explanatory CQ. Despite their importance, the complexity of these problems is prohibitively high (coNEXPTIME-complete). We isolate their two main sources of complexity and propose relaxations of them that reduce the complexity while having meaningful theoretical interpretations. The first relaxation is based on the idea of using existential pebble games for approximating homomorphism tests. We show that this characterizes QBE/definability for CQs up to treewidth k, while reducing the complexity to EXPTIME. As a side result, we obtain that the complexity of the QBE/definability problems for CQs of treewidth k is EXPTIME-complete for each $k \geq 1$. The second relaxation is based on the idea of "desynchronizing" direct products, which characterizes QBE/definability for unions of CQs and reduces the complexity to coNP. The combination of these two relaxations yields tractability for QBE and characterizes it in terms of unions of CQs of treewidth at most k. We also study the complexity of these problems for conjunctive regular path queries over graph databases, showing them to be no more difficult than for CQs.

1998 ACM Subject Classification H.2.3 [Database Management] Query Languages

Keywords and phrases reverse engineering, conjunctive queries, query by example, definability, treewidth, complexity of pebble games

Digital Object Identifier 10.4230/LIPIcs.ICDT.2017.7

1 Introduction

Reverse engineering is the general problem of abstracting user examples into an explanatory query. An important instance of this problem corresponds to *query-by-example* (QBE) for a query language $\mathcal{L}$. In QBE, the system is presented with a database $\mathcal{D}$ and n-ary relations S^+ and S^- over $\mathcal{D}$ of *positive* and *negative* examples, respectively. The question is whether there exists a query q in $\mathcal{L}$ such that its evaluation $q(\mathcal{D})$ over $\mathcal{D}$ contains all the positive examples (i.e., $S^+ \subseteq q(\mathcal{D})$) but none of the negative ones (i.e., $q(\mathcal{D}) \cap S^- = \emptyset$). In case such q exists, it is also desirable to return its result $q(\mathcal{D})$. Another version of this problem assumes that the system is given the set S^+ of positive examples only, and the question is whether there is a query q in $\mathcal{L}$ that precisely defines S^+, i.e., $q(\mathcal{D}) = S^+$. This is often known as the *definability problem* for $\mathcal{L}$. As of late, QBE and definability have received quite some attention in different contexts; e.g., for first-order logic and the class of conjunctive queries over relational databases [26, 23, 19, 7, 2, 24, 22]; for regular path queries over graph databases [1, 6]; for SPARQL queries over RDF [3]; and for tree patterns over XML [10, 20].

In data management, a particularly important instance of QBE and definability corresponds to the case when $\mathcal{L}$ is the class of conjunctive queries (CQs). Nevertheless, the relevance of such instance is counterbalanced by its inherent complexity: Both QBE and definability for CQs are coNEXPTIME-complete [24, 22]. Moreover, in case that a CQ-*explanation* q for S^+ and S^- over $\mathcal{D}$ exists (i.e., a CQ q such that $S^+ \subseteq q(\mathcal{D})$ and $q(\mathcal{D}) \cap S^- = \emptyset$ for QBE), it might take double exponential time to compute its result $q(\mathcal{D})$. While several heuristics have been proposed that alleviate this complexity in practice [26, 23, 19, 7], up to date there has been (essentially) no theoretical investigation identifying the sources of complexity of these problems and proposing principled solutions for them. The general objective of this article is to make a first step in such direction.

A semantic characterization of QBE for CQs has been known for a long time in the community. Formally, there exists a CQ q such that $S^+ \subseteq q(\mathcal{D})$ and $q(\mathcal{D}) \cap S^- = \emptyset$ (i.e., a CQ-explanation) if and only if (essentially) the following *QBE test for CQs* succeds:

- QBE test for CQs: For each tuple $\bar{b}$ in S^- it is the case that $\prod_{\bar{a} \in S^+} (\mathcal{D}, \bar{a}) \not\to (\mathcal{D}, \bar{b})$, i.e., $\prod_{\bar{a} \in S^+} (\mathcal{D}, \bar{a})$ does not homomorphically map to $(\mathcal{D}, \bar{b})$. (Here, $\prod$ denotes the usual direct product of databases with distinguished tuples of constants).

(A similar test characterizes CQ-definability, save that now $\bar{b}$ is an arbitrary tuple over $\mathcal{D}$ outside S^+). Moreover, in case there is a CQ-explanation q for S^+ and S^- over $\mathcal{D}$, then there is a *canonical* such explanation given by the CQ whose body corresponds to $\prod_{\bar{a} \in S^+} (\mathcal{D}, \bar{a})$. As shown by Willard [24], the QBE test for CQs yields optimal bounds for determining (a) the existence of a CQ-explanation q for S^+ and S^- over $\mathcal{D}$ (namely, coNEXPTIME), and (b) the size of such q (i.e., exponential). More important, it allows to identify the two main sources of complexity of the problem, each one of which increases its complexity by one exponential:

1. The construction of the canonical explanation $\prod_{\bar{a} \in S^+} (\mathcal{D}, \bar{a})$, which takes exponential time in the combined size of $\mathcal{D}$ and S^+.
2. The homomorphism test $\prod_{\bar{a} \in S^+} (\mathcal{D}, \bar{a}) \to (\mathcal{D}, \bar{b})$ for each tuple $\bar{b} \in S^-$. Since, in general, checking for the existence of a homomorphism is an NP-complete problem, this step involves an extra exponential blow up.

Our contributions: We propose relaxations of the QBE test for CQs that alleviate one or both sources of complexity and have meaningful theoretical interpretations in terms of the QBE problem (our results also apply to definability). They are based on standard approximation notions for the homomorphism test and the construction of the direct product $\prod_{\bar{a} \in S^+} (\mathcal{D}, \bar{a})$, as found in the context of constraint satisfaction and definability, respectively.

1. We start by relaxing the second source of complexity, i.e., the one given by the homomorphism tests of the form $\prod_{\bar{a} \in S^+} (\mathcal{D}, \bar{a}) \to (\mathcal{D}, \bar{b})$, for $\bar{b} \in S^-$. In order to approximate the notion of homomorphism, we use the *strong consistency tests* often applied in the area of constraint satisfaction [13]. As observed by Kolaitis and Vardi [18], such consistency tests can be recast in terms of the *existential pebble game* [17], first defined in the context of database theory as a tool for studying the expressive power of Datalog, and also used to show that CQs of bounded treewidth can be evaluated efficiently [12].
   As opposed to the homomorphism test, checking for the existence of a *winning duplicator strategy* in the existential k-pebble game on $(\mathcal{D}, \bar{a})$ and $(\mathcal{D}', \bar{b})$, denoted $(\mathcal{D}, \bar{a}) \to_k (\mathcal{D}', \bar{b})$, can be solved in polynomial time for each fixed $k > 1$ [17]. Therefore, replacing the homomorphism test $\prod_{\bar{a} \in S^+} (\mathcal{D}, \bar{a}) \to (\mathcal{D}, \bar{b})$ with its "approximation" $\prod_{\bar{a} \in S^+} (\mathcal{D}, \bar{a}) \to_k (\mathcal{D}, \bar{b})$ reduces the complexity of the QBE test for CQs to EXPTIME. Furthermore, this approximation has a neat theoretical interpretation: The relaxed version of the QBE

test accepts the input given by $(\mathcal{D}, S^+, S^-)$ if and only if there is a CQ-explanation q for S^+ and S^- over $\mathcal{D}$ such that q is of *treewidth at most* $(k-1)$. While the latter is not particularly surprising in light of the strong existing connections between the existential k-pebble game and the evaluation of CQs of treewidth at most $(k-1)$ [12], we believe our characterization to be of conceptual importance.

Interestingly, when this relaxed version of the QBE test yields a CQ-explanation q of treewidth at most $(k-1)$, its result $q(\mathcal{D})$ can be evaluated in exponential time (recall that for general CQs this might require double exponential time).

2. We then prove that the previous bound is optimal, i.e., checking whether the relaxed version of the QBE test accepts the input given by $(\mathcal{D}, S^+, S^-)$, or, equivalently, if there is a CQ-explanation q for S^+ and S^- over $\mathcal{D}$ of treewidth at most k, for each $k \geq 1$, is EXPTIME-complete. (This also holds for the definability problem for CQs of treewidth at most k). Intuitively, this states that relaxing the second source of complexity of the test by using existential pebble games does not eliminate the first one.
3. Finally, we look at the second source of complexity, i.e., the construction of the exponential size canonical explanation $\prod_{\bar{a} \in S^+}(\mathcal{D}, \bar{a})$. While it is not clear which techniques are better suited for approximating this construction, we look at a particular one that appears in the context of definability: Instead of constructing the synchronized product $\prod_{\bar{a} \in S^+}(\mathcal{D}, \bar{a})$ with respect to all tuples in S, we look at them one by one. That is, we check whether for each tuples $\bar{a} \in S^+$ and $\bar{b} \in S^-$ it is the case that $(\mathcal{D}, \bar{a}) \not\to (\mathcal{D}, \bar{b})$. By using a characterization developed in the context of definability [1], we observe that this relaxed version of the QBE test is coNP-complete and has a meaningful interpretation: It corresponds to finding explanations based on *unions* of CQs. Moreover, when combined with the previous relaxation (i.e., replacing the homomorphism test $(\mathcal{D}, \bar{a}) \to (\mathcal{D}, \bar{b})$ with $(\mathcal{D}, \bar{a}) \to_k (\mathcal{D}, \bar{b})$) we obtain tractability. This further relaxed test corresponds to finding explanations over the set of unions of CQs of treewidth at most $(k-1)$.

We then switch to study QBE in the context of graph databases, where CQs are often extended with the ability to check whether two nodes are linked by a path whose label satisfies a given regular expression. This gives rise to the class of *conjunctive regular path queries*, or CRPQs (see, e.g., [11, 8, 25, 5]). CRPQ-definability was first studied by Antonopulos et al. [1]. In particular, it is shown that CRPQ-definability is in EXPSPACE by exploting automata-based techniques, in special, pumping arguments. Our contributions in this context are the following:

1. We first provide a QBE test for CRPQs in the spirit of the one for CQs given above. With such characterization we prove that QBE and definability for CRPQs are in coNEXPTIME, improving the EXPSPACE upper bound of Antonopoulos et al. This tells us that these problems are at least not more difficult than for CQs.
2. We also develop relaxations of the QBE test for CRPQs based on the existential pebble game and the "desynchronization" of the direct product $\prod_{\bar{a} \in S^+}(\mathcal{D}, \bar{a})$. As before, we show that they reduce the complexity of the test and have meaningful interpretations in terms of the class of queries we use to construct explanations.

Organization: Preliminaries are in Section 2. A review of QBE/definability for CQs is provided in Section 3. Relaxations of the homomorphism tests are studied in Section 4 and the desynchronization of the direct product in Section 5. In Section 6 we consider QBE/definability for CRPQs. Future work is presented in Section 7.

2 Preliminaries

Databases, homomorphisms, and direct products. A *schema* is a finite set of relation symbols, each one of which has an associated arity $n > 0$. A *database* over schema σ is a finite set of atoms of the form $R(\bar{a})$, where R is a relation symbol in σ of arity $n > 0$ and $\bar{a}$ is an n-ary tuple of constants. We slightly abuse notation, and sometimes write $\mathcal{D}$ also for the set of elements mentioned in $\mathcal{D}$.

Let $\mathcal{D}$ and $\mathcal{D}'$ be databases over the same schema σ. A *homomorphism* from $\mathcal{D}$ to $\mathcal{D}'$ is a mapping h from the elements of $\mathcal{D}$ to the elements of $\mathcal{D}'$ such that for every atom $R(\bar{a})$ in $\mathcal{D}$ it is the case that $R(h(\bar{a})) \in \mathcal{D}'$. We often need to talk about distinguished tuples of elements in databases. We then write $(\mathcal{D}, \bar{a})$ to define the pair that corresponds to the database $\mathcal{D}$ and the tuple $\bar{a}$ of elements in $\mathcal{D}$. Let $\bar{a}$ and $\bar{b}$ be n-ary $(n \geq 0)$ tuples of elements in $\mathcal{D}$ and $\mathcal{D}'$, respectively. A homomorphism from $(\mathcal{D}, \bar{a})$ to $(\mathcal{D}', \bar{b})$ is a homomorphism from $\mathcal{D}$ to $\mathcal{D}'$ such that $h(\bar{a}) = \bar{b}$. We write $(\mathcal{D}, \bar{a}) \to (\mathcal{D}', \bar{b})$ if there is a homomorphism from $(\mathcal{D}, \bar{a})$ to $(\mathcal{D}', \bar{b})$. Checking if $(\mathcal{D}, \bar{a}) \to (\mathcal{D}', \bar{b})$ is a well-known NP-complete problem.

In this work, the notion of *direct product* of databases is particularly important. Let $\bar{a} = (a_1, \dots, a_n)$ and $\bar{b} = (b_1, \dots, b_n)$ be n-ary tuples of elements over A and B, respectively. Their direct product $\bar{a} \otimes \bar{b}$ is the n-ary tuple $((a_1, b_1), \dots, (a_n, b_n))$ over $A \times B$. If $\mathcal{D}$ and $\mathcal{D}'$ are databases over the same schema σ, we define $\mathcal{D} \otimes \mathcal{D}'$ to be the following database over σ:

$$\{R(\bar{a} \otimes \bar{b}) \mid R \in \sigma,\ R(\bar{a}) \in \mathcal{D},\ \text{and}\ R(\bar{b}) \in \mathcal{D}'\}.$$

Further, we use $(\mathcal{D}, \bar{a}) \otimes (\mathcal{D}', \bar{b})$ to denote the pair $(\mathcal{D} \otimes \mathcal{D}', \bar{a} \otimes \bar{b})$, and write $\prod_{1 \leq i \leq m} (\mathcal{D}_i, \bar{a}_i)$ as a shorthand for $(\mathcal{D}_1, \bar{a}_1) \otimes \cdots \otimes (\mathcal{D}_m, \bar{a}_m)$. This is allowed since $\otimes$ is an associative operation.

The elements in the tuple $\prod_{1 \leq i \leq m} \bar{a}_i$ may or may not appear in $\prod_{1 \leq i \leq m} \mathcal{D}_i$. If they do appear, we call $\prod_{1 \leq i \leq m} (\mathcal{D}_i, \bar{a}_i)$ *safe*. The notion of safeness is important in our work for reasons that will become apparent later. The next example better explains this notion:

▶ **Example 1.** If $\mathcal{D} = \{R(a, b), S(c, d)\}$, $\bar{a}_1 = (a, b)$, and $\bar{a}_2 = (c, d)$, then $(\mathcal{D}, \bar{a}_1) \otimes (\mathcal{D}, \bar{a}_2)$ is unsafe. In fact, $\bar{a}_1 \otimes \bar{a}_2 = \big((a, c), (b, d)\big)$ and $\mathcal{D} \otimes \mathcal{D} = \{R((a, a), (b, b)), S((c, c), (d, d))\}$. That is, none of the elements in $\bar{a}_1 \otimes \bar{a}_2$ belongs to $\mathcal{D} \otimes \mathcal{D}$. ◀

It is worth remarking that the direct product $\otimes$ defines the least upper bound in the lattice of databases defined by the notion of homomorphism. In particular:

1. $\prod_{1 \leq i \leq m} (\mathcal{D}_i, \bar{a}_i) \to (\mathcal{D}_i, \bar{a}_i)$ for each $1 \leq i \leq m$, and
2. if $(\mathcal{D}, \bar{a}) \to (\mathcal{D}_i, \bar{a}_i)$ for each $1 \leq i \leq m$, then $(\mathcal{D}, \bar{a}) \to \prod_{1 \leq i \leq m} (\mathcal{D}_i, \bar{a}_i)$.

Conjunctive queries. A *conjunctive query* (CQ) q over relational schema σ is an FO formula of the form:

$$\exists \bar{y} \big(R_1(\bar{x}_1) \wedge \cdots \wedge R_m(\bar{x}_m)\big), \tag{1}$$

such that (a) each $R_i(\bar{x}_i)$ is an atom over σ, for $1 \leq i \leq m$, and (b) $\bar{y}$ is a sequence of variables taken from the $\bar{x}_i$'s. In order to ensure domain-independence for queries, we only consider CQs without constants. We often write $q(\bar{x})$ to denote that $\bar{x}$ is the sequence of *free* variables of q, i.e., the ones that do not appear existentially quantified in $\bar{y}$.

Let $\mathcal{D}$ be a database over σ. We define the evaluation of a CQ $q(\bar{x})$ of the form (1) over $\mathcal{D}$ in terms of the homomorphisms from $\mathcal{D}_q$ to $\mathcal{D}$, where $\mathcal{D}_q$ is the *canonical database* of q, that is, $\mathcal{D}_q$ is the database $\{R_1(\bar{x}_1), \dots, R_m(\bar{x}_m)\}$ that contains all atoms in q. The *evaluation of* $q(\bar{x})$ *over* $\mathcal{D}$, denoted $q(\mathcal{D})$, contains exactly those tuples $h(\bar{x})$ such that h is a homomorphism from $\mathcal{D}_q$ to $\mathcal{D}$.

CQs of bounded treewidth. The evaluation problem for CQs (i.e., determining whether $q(\mathcal{D}) \neq \emptyset$, given a database $\mathcal{D}$ and a CQ q) is NP-complete, but becomes tractable for several syntactically defined classes. One of the most prominent such classes corresponds to the CQs of *bounded treewidth* [9]. Recall that treewidth is a graph-theoretical concept that measures how much a graph resembles a tree (see, e.g., [14]). For instance, trees have treewidth one, cycles treewidth two, and K_k, the clique on k elements, treewidth $k-1$.

Formally, let $G = (V, E)$ be an undirected graph. A *tree decomposition* of G is a pair (T, λ), where T is a tree and λ is a mapping that assigns a nonempty set of nodes in V to each node t in T, for which the following holds:

1. For each $v \in V$ it is the case that the set of nodes $t \in T$ such that $v \in \lambda(t)$ is connected.
2. For each edge $\{u, v\} \in E$ there exists a node $t \in T$ such that $\{u, v\} \subseteq \lambda(t)$.

The *width* of (T, λ) corresponds to $(\max\{|\lambda(t)| \mid t \in T\}) - 1$. The treewidth of G is then defined as the minimum width of its tree decompositions.

We define the treewidth of a CQ $q = \exists \bar{y} \bigwedge_{1 \leq i \leq m} R_i(\bar{x}_i)$ as the treewidth of the *Gaifman graph* of its existentially quantified variables. Recall that this is the undirected graph whose vertices are the existentially quantified variables of q (i.e., those in $\bar{y}$) and there is an edge between distinct existentially quantified variables y and y' if and only they appear together in some atom of q, that is, they both appear in a tuple $\bar{x}_i$ for $1 \leq i \leq m$. For $k \geq 1$, we denote by $\mathsf{TW}(k)$ the class of CQs of treewidth at most k. It is known that the evaluation problem for the class $\mathsf{TW}(k)$ (for each fixed $k \geq 1$) can be solved in polynomial time [9, 12].

The QBE and definability problems. Let $\mathcal{C}$ be a class of queries (e.g., the class CQ of all conjunctive queries, or $\mathsf{TW}(k)$ of CQs of treewidth at most k). Suppose that $\mathcal{D}$ is a database and S^+ and S^- are n-ary relations over $\mathcal{D}$ of positive and negative examples, respectively. A $\mathcal{C}$*-explanation* for S^+ and S^- over $\mathcal{D}$ is a query q in $\mathcal{C}$ such that $S^+ \subseteq q(\mathcal{D})$ and $q(\mathcal{D}) \cap S^- = \emptyset$. Analogously, a $\mathcal{C}$*-definition* of S^+ over $\mathcal{D}$ is a query q in $\mathcal{C}$ such that $q(\mathcal{D}) = S^+$. The *query by example* and *definability* problems for $\mathcal{C}$ are as follows:

PROBLEM :	$\mathcal{C}$-QUERY-BY-EXAMPLE (resp., $\mathcal{C}$-DEFINABILITY)
INPUT :	A database $\mathcal{D}$ and n-ary relations S^+ and S^- over $\mathcal{D}$ (resp., a database $\mathcal{D}$ and an n-ary relation S^+ over $\mathcal{D}$)
QUESTION :	Is there a $\mathcal{C}$-explanation for S^+ and S^- over $\mathcal{D}$? (resp., is there a $\mathcal{C}$-definition of S^+ over $\mathcal{D}$?)

3 Query by example and definability for CQs

Let us start by recalling what is known about these problems for CQs. We first establish characterizations of the notions of CQ-explanations/definitions based on the following tests:

- QBE test for CQs: Takes as input a database $\mathcal{D}$ and n-ary relations S^+ and S^- over $\mathcal{D}$. It accepts if and only if:
  1. $\prod_{\bar{a} \in S^+}(\mathcal{D}, \bar{a})$ is safe, and
  2. $\prod_{\bar{a} \in S^+}(\mathcal{D}, \bar{a}) \not\to (\mathcal{D}, \bar{b})$ for each tuple $\bar{b} \in S^-$.
- Definability test for CQs: Takes as input a database $\mathcal{D}$ and an n-ary relation S^+ over $\mathcal{D}$. It accepts if and only if:
  1. $\prod_{\bar{a} \in S^+}(\mathcal{D}, \bar{a})$ is safe, and
  2. $\prod_{\bar{a} \in S^+}(\mathcal{D}, \bar{a}) \not\to (\mathcal{D}, \bar{b})$ for each n-ary tuple $\bar{b}$ over $\mathcal{D}$ that is not in S^+.

The following characterizations are considered to be folklore in the literature:

▶ **Proposition 2.** *The following statements hold:*
1. *Let $\mathcal{D}$ be a database and S^+, S^- relations over $\mathcal{D}$. There is a* CQ*-explanation for S^+ and S^- over $\mathcal{D}$ if and only if the QBE test for CQs accepts $\mathcal{D}$, S^+, and S^-.*
2. *Let $\mathcal{D}$ be a database and S^+ a relation over $\mathcal{D}$. There is a* CQ*-definition for S^+ over $\mathcal{D}$ if and only if the definability test for CQs accepts $\mathcal{D}$ and S^+.*

This provides us with a simple method for obtaining a coNEXPTIME upper bound for CQ-QUERY-BY-EXAMPLE and CQ-DEFINABILITY. Let us concentrate on the first problem (a similar argument works for the second one). Assume that S^+ and S^- are relations of positive and negative examples over a database $\mathcal{D}$. It follows from Proposition 2 that to check that there is *not* CQ-explanation for S^+ and S^- over $\mathcal{D}$, we need to either show that $\prod_{\bar{a}\in S^+}(\mathcal{D},\bar{a})$ is unsafe or guess a tuple $\bar{b} \in S^-$ and a homomorphism h from $\prod_{\bar{a}\in S^+}(\mathcal{D},\bar{a})$ to $(\mathcal{D},\bar{b})$. Since $\prod_{\bar{a}\in S^+}(\mathcal{D},\bar{a})$ is of exponential size, checking its safety can be carried out in exponential time. On the other hand, the guess of h is also of exponential size, and therefore checking that h is indeed a homomorphism from $\prod_{\bar{a}\in S^+}(\mathcal{D},\bar{a})$ to $(\mathcal{D},\bar{b})$ can be performed in exponential time. The whole procedure can then be carried out in NEXPTIME. As it turns out, this bound is also optimal:

▶ **Theorem 3.** [24, 22] *The problems* CQ-QUERY-BY-EXAMPLE *and* CQ-DEFINABILITY *are* coNEXPTIME*-complete.*

The lower bound for CQ-DEFINABILITY was established by Willard using a complicated reduction from the complement of a tiling problem. A simpler proof was then obtained by ten Cate and Dalmau [22]. Their techniques also establish a lower bound for CQ-QUERY-BY-EXAMPLE. Notably, these lower bounds hold even when S^+ and S^- are unary relations.

The cost of evaluating CQ-explanations. Recall that in query by example not only we want to find a CQ-explanation q for S^+ and S^- over $\mathcal{D}$, but also compute its result $q(\mathcal{D})$ if possible. It follows from the proof of Proposition 2 that in case there is a CQ-explanation for S^+ and S^- over $\mathcal{D}$, then we can assume such CQ to be $\prod_{\bar{a}\in S^+}(\mathcal{D},\bar{a})$, i.e., the CQ whose set of atoms is $\mathcal{D}^{|S^+|}$ and whose tuple of free variables is $\prod_{\bar{a}\in S^+}\bar{a}$ (notice that we are using here the assumption that $\prod_{\bar{a}\in S^+}(\mathcal{D},\bar{a})$ is safe, i.e., that the free variables in $\prod_{\bar{a}\in S^+}\bar{a}$ do in fact appear in the atoms in $\mathcal{D}^{|S^+|}$). The CQ $\prod_{\bar{a}\in S^+}(\mathcal{D},\bar{a})$ is known as the *canonical* CQ-explanation. We could then simply evaluate this canonical CQ-explanation over $\mathcal{D}$ in order to meet the requirements of query by example. This, however, takes double exponential time since $\prod_{\bar{a}\in S^+}(\mathcal{D},\bar{a})$ itself is of exponential size. It is not known whether there are better algorithms for computing the result of *some* CQ-explanation, but the results in this section suggest that this is unlikely.

Size of CQ explanations and definitions. It follows from the previous observations that CQ-explanations are of at most exponential size (by taking the canonical CQ-explanation as witness). The same holds for CQ-definitions. Interestingly, these bounds are optimal:

▶ **Proposition 4.** [24, 22] *The following statements hold:*
1. *If there is a* CQ*-explanation for S^+ and S^- over $\mathcal{D}$, then there is a* CQ*-explanation of at most exponential size; namely, $\prod_{\bar{a}\in S^+}(\mathcal{D},\bar{a})$. Similarly, for* CQ*-definitions.*
2. *There is a family $(\mathcal{D}_n, S_n^+, S_n^-)_{n\geq 0}$ of tuples of databases $\mathcal{D}_n$ and relations S_n^+ and S_n^- over $\mathcal{D}_n$, such that (a) the combined size of $\mathcal{D}_n$, S_n^+, and S_n^- is polynomial in n, (b) there is a* CQ*-explanation for S_n^+ and S_n^- over $\mathcal{D}_n$, and (c) the size of the smallest such* CQ*-explanation is at least 2^n. Similarly, for* CQ*-definitions.*

Sources of complexity. The QBE test performs the following steps on input $(\mathcal{D}, S^+, S^-)$: (1) It computes $\prod_{\bar{a} \in S^+}(\mathcal{D}, \bar{a})$, and (2) it checks whether $\prod_{\bar{a} \in S^+}(\mathcal{D}, \bar{a})$ is unsafe or it is the case that $\prod_{\bar{a} \in S^+}(\mathcal{D}, \bar{a}) \to (\mathcal{D}, \bar{b})$ for some $\bar{b} \in S^-$. The definability test is equivalent, but the homomorphism test is then extended to each tuple over $\mathcal{D}$ but outside S^+. Two sources of complexity are involved in these tests, each one of which incurs in one exponential blow up: (a) The construction of $\prod_{\bar{a} \in S^+}(\mathcal{D}, \bar{a})$, and (b) the homomorphism tests $\prod_{\bar{a} \in S^+}(\mathcal{D}, \bar{a}) \to (\mathcal{D}, \bar{b})$. In order to alleviate the high complexity of the tests we thus propose relaxations of these two sources of complexity. The proposed relaxations are based on well-studied approximation notions with strong theoretical support. As such, they give rise to clean reformulations of the notions of CQ-explanations/definitions. We start with the homomorphism test in the following section.

4 A relaxation of the homomorphism test

We use an approximation technique for the homomorphism test based on the existential pebble game. This technique finds several applications in database theory [17, 12] and can be shown to be equivalent to the strong consistency tests for homomorphism approximation used in the area of constraint satisfaction [18]. The complexity of the (existential) pebble game is by now well-understood [15, 16]. We borrow several techniques used in such analysis to understand the complexity of our problems. We also prove some results on the complexity of such games that are of independent interest. We define the existential pebble game below.

The existential pebble game. Let $k > 1$. The existential k-pebble game is played by the spoiler and the duplicator on pairs $(\mathcal{D}, \bar{a})$ and $(\mathcal{D}', \bar{b})$, where $\mathcal{D}$ and $\mathcal{D}'$ are databases over the same schema and $\bar{a}$ and $\bar{b}$ are n-ary $(n \geq 0)$ tuples over $\mathcal{D}$ and $\mathcal{D}'$, respectively. The spoiler plays on $\mathcal{D}$ only, and the duplicator responds on $\mathcal{D}'$. In the first round the spoiler places his pebbles $\mathtt{p}_1, \dots, \mathtt{p}_k$ on (not necessarily distinct) elements $c_1, \dots, c_k$ in $\mathcal{D}$, and the duplicator responds by placing his pebbles $\mathtt{q}_1, \dots, \mathtt{q}_k$ on elements $d_1, \dots, d_k$ in $\mathcal{D}'$. In every further round, the spoiler removes one of his pebbles, say $\mathtt{p}_i$, for $1 \leq i \leq k$, and places it on an element of $\mathcal{D}$, and the duplicator responds by placing his corresponding pebble $\mathtt{q}_i$ on some element of $\mathcal{D}'$. The duplicator wins if he has a *winning strategy*, i.e., he can indefinitely continue playing the game in such way that at each round, if $c_1, \dots, c_k$ and $d_1, \dots, d_k$ are the elements covered by pebbles $\mathtt{p}_1, \dots, \mathtt{p}_k$ and $\mathtt{q}_1, \dots, \mathtt{q}_k$ on $\mathcal{D}$ and $\mathcal{D}'$, respectively, then

$$\big((c_1, \dots, c_k, \bar{a}), (d_1, \dots, d_k, \bar{b})\big)$$

is a *partial homomorphism* from $\mathcal{D}$ to $\mathcal{D}'$. Recall that this means that for every atom of the form $R(\bar{c}) \in \mathcal{D}$, where each element c of $\bar{c}$ appears in $(c_1, \dots, c_k, \bar{a})$, it is the case that $R(\bar{d}) \in \mathcal{D}'$, where $\bar{d}$ is the tuple that is obtained from $\bar{c}$ by replacing each element c of $\bar{c}$ by its corresponding element d in $(d_1, \dots, d_k, \bar{b})$. If such winning strategy for the duplicator exists, we write $(\mathcal{D}, \bar{a}) \to_k (\mathcal{D}', \bar{b})$.

It is easy to see that the relations $\to_k$, for $k > 1$, provide an approximation of the notion of homomorphism in the following sense:

$$\to \subsetneq \dots \subsetneq \to_{k+1} \subsetneq \to_k \subsetneq \cdots \subsetneq \to_2 .$$

Furthermore, these approximations are convenient from a complexity point of view: While checking for the existence of a homomorphism from $(\mathcal{D}, \bar{a})$ to $(\mathcal{D}', \bar{b})$ is NP-complete, checking for the existence of a winning strategy for the duplicator in the existential k-pebble game can be solved efficiently:

▶ **Proposition 5.** [17] *Fix $k > 1$. Checking if $(\mathcal{D}, \bar{a}) \rightarrow_k (\mathcal{D}', \bar{b})$, given databases $\mathcal{D}$ and $\mathcal{D}'$ and n-ary tuples $\bar{a}$ and $\bar{b}$ over $\mathcal{D}$ and $\mathcal{D}'$, respectively, can be solved in polynomial time.*

Furthermore, there is an interesting connection between the existential pebble game and the evaluation of CQs of bounded treewidth as established in the following proposition:

▶ **Proposition 6.** [4] *Fix $k \geq 1$. Consider databases $\mathcal{D}$ and $\mathcal{D}'$ over the same schema and n-ary tuples $\bar{a}$ and $\bar{b}$ over $\mathcal{D}$ and $\mathcal{D}'$, respectively. Then $(\mathcal{D}, \bar{a}) \rightarrow_{k+1} (\mathcal{D}', \bar{b})$ if and only if for each CQ $q(\bar{x})$ in $\mathsf{TW}(k)$ such that $|\bar{x}| = n$ the following holds:*

$$\bar{a} \in q(\mathcal{D}) \implies \bar{b} \in q(\mathcal{D}'),$$

or, equivalently, $(\mathcal{D}_q, \bar{x}) \rightarrow (\mathcal{D}, \bar{a})$ implies $(\mathcal{D}_q, \bar{x}) \rightarrow (\mathcal{D}', \bar{b})$, where as before $\mathcal{D}_q$ is the database that contains all the atoms of q.

Moreover, in case that $(\mathcal{D}, \bar{a}) \not\rightarrow_{k+1} (\mathcal{D}', \bar{b})$ there exists an exponential size CQ $q(\bar{x})$ in $\mathsf{TW}(k)$ such that $\bar{a} \in q(\mathcal{D})$ but $\bar{b} \notin q(\mathcal{D}')$.

The relaxed test. We study the following relaxed version of the QBE test for CQs that replaces the notion of homomoprhism $\rightarrow$ with its approximation $\rightarrow_k$, for a fixed $k > 1$:

- k-pebble QBE test for CQs: Takes as input a database $\mathcal{D}$ and n-ary relations S^+ and S^- over $\mathcal{D}$. It accepts if and only if:
  1. $\prod_{\bar{a} \in S^+} (\mathcal{D}, \bar{a})$ is safe, and
  2. $\prod_{\bar{a} \in S^+} (\mathcal{D}, \bar{a}) \not\rightarrow_k (\mathcal{D}, \bar{b})$ for each tuple $\bar{b} \in S^-$.

Analogously, we define the k-pebble definability test for CQs. It immediately follows from the fact that the relation $\rightarrow_k$ can be decided in polynomial time (Proposition 5) that the k-pebble tests for CQs reduce the complexity of the general test from coNEXPTIME to EXPTIME. Later, in Section 4.2, we show that this is optimal.

4.1 A characterization of the k-pebble tests for CQs

Using Proposition 6 we can now establish the theoretical meaningfulness of the relaxed tests: They admit a clean characterization in terms of the CQs of bounded treewidth. In fact, recall that the QBE (resp., definability) test for CQs precisely characterizes the existence of CQ-explanations (resp., CQ-definitions). As we show next, their relaxed versions based on the existential $(k+1)$-pebble game preserve these characterizations up to treewidth k:

▶ **Theorem 7.** *Fix $k \geq 1$. Consider a database $\mathcal{D}$ and n-ary relations S^+ and S^- over $\mathcal{D}$.*

1. *There is a $\mathsf{TW}(k)$-explanation for S^+ and S^- over $\mathcal{D}$ if and only if the $(k+1)$-pebble QBE test for CQs accepts $\mathcal{D}$, S^+, and S^-.*
2. *There is a $\mathsf{TW}(k)$-definition for S^+ over $\mathcal{D}$ if and only if the $(k+1)$-pebble definability test for CQs accepts $\mathcal{D}$ and S^+.*

Proof. We concentrate on explanations (the proof for definitions is analogous). From left to right, assume for the sake of contradiction that q is a $\mathsf{TW}(k)$-explanation for S^+ and S^- over $\mathcal{D}$, yet the $(k+1)$-pebble QBE test for CQs fails over $\mathcal{D}$, S^+, and S^-. Since there is a $\mathsf{TW}(k)$-explanation for S^+ and S^- over $\mathcal{D}$, we have from Proposition 2 that $\prod_{\bar{a} \in S^+} (\mathcal{D}, \bar{a})$ is safe. Therefore, it must be the case that $\prod_{\bar{a} \in S^+} (\mathcal{D}, \bar{a}) \rightarrow_{k+1} (\mathcal{D}, \bar{b})$ for some $\bar{b} \in S^-$. Since $S^+ \subseteq q(\mathcal{D})$ it is the case that $\bar{a} \in q(\mathcal{D})$ for each $\bar{a} \in S^+$. That is, $(\mathcal{D}_q, \bar{x}) \rightarrow (\mathcal{D}, \bar{a})$ for each $\bar{a} \in S^+$. Due to basic properties of direct products, this implies that $(\mathcal{D}_q, \bar{x}) \rightarrow \prod_{\bar{a} \in S^+} (\mathcal{D}, \bar{a})$.

From Proposition 6 we conclude that $(\mathcal{D}_q, \bar{x}) \to (\mathcal{D}, \bar{b})$, i.e., $\bar{b} \in q(\mathcal{D})$. This is a contradiction since $\bar{b} \in S^-$ and $q(\mathcal{D}) \cap S^- = \emptyset$.

From right to left, assume that the $(k+1)$-pebble QBE test for CQs accepts $\mathcal{D}$, S^+, and S^-, i.e., $\prod_{\bar{a}\in S^+}(\mathcal{D}, \bar{a})$ is safe and for every tuple $\bar{b} \in S^-$ it is the case that $\prod_{\bar{a}\in S^+}(\mathcal{D}, \bar{a}) \not\to_{k+1} (\mathcal{D}, \bar{b})$. Since $\prod_{\bar{a}\in S^+}(\mathcal{D}, \bar{a})$ is safe we can apply Proposition 6, which tells us that for each $\bar{b} \in S^-$ there is a CQ $q_{\bar{b}}(\bar{x})$ such that $(\mathcal{D}_{q_{\bar{b}}}, \bar{x}) \to \prod_{\bar{a}\in S^+}(\mathcal{D}, \bar{a})$ but $(\mathcal{D}_{q_{\bar{b}}}, \bar{x}) \not\to (\mathcal{D}, \bar{b})$. Suppose first that $S^- \neq \emptyset$ and let:

$$q(\bar{x}) \ := \ \bigwedge_{\bar{b}\in S^-} q_{\bar{b}}(\bar{x}).$$

It is easy to see that $q(\bar{x})$ is well-defined (since S^- is nonempty) and can be expressed as a CQ in $\mathsf{TW}(k)$. For the latter we simply use fresh existentially quantified variables for each CQ $q_{\bar{b}}$ such that $\bar{b} \in S^-$ and then move all existentially quantified variables in $\bigwedge_{\bar{b}\in S^-} q_{\bar{b}}(\bar{x})$ to the front. We now prove that $q(\bar{x})$ is a $\mathsf{TW}(k)$-explanation for S^+ and S^- over $\mathcal{D}$. It easily follows that $(\mathcal{D}_q, \bar{x}) \to \prod_{\bar{a}\in S^+}(\mathcal{D}, \bar{a})$ from the fact that $(\mathcal{D}_{q_{\bar{b}}}, \bar{x}) \to \prod_{\bar{a}\in S^+}(\mathcal{D}, \bar{a})$ for each $\bar{b} \in S^-$. But then $(\mathcal{D}_q, \bar{x}) \to (\mathcal{D}, \bar{a})$ for each $\bar{a} \in S^+$. This means that $\bar{a} \in q(\mathcal{D})$ for each $\bar{a} \in S^+$, i.e., $S^+ \subseteq q(\mathcal{D})$. Assume now for the sake of contradiction that $q(\mathcal{D}) \cap S^- \neq \emptyset$, that is, there is a tuple $\bar{b} \in q(\mathcal{D}) \cap S^-$. Then $(\mathcal{D}_q, \bar{x}) \to (\mathcal{D}, \bar{b})$, which implies that $(\mathcal{D}_{q_{\bar{b}}}, \bar{x}) \to (\mathcal{D}, \bar{b})$. This is a contradiction. The case when $S^- = \emptyset$ can be proved using similar techniques. ◀

4.2 The complexity of the k-pebble tests for CQs

As mentioned before, the k-pebble tests for CQs can be evaluated in exponential time. We show here that such bounds are also optimal:

▶ **Theorem 8.** *Deciding whether the k-pebble QBE test for CQs accepts $(\mathcal{D}, S^+, S^-)$ is* EXPTIME*-complete for each $k > 1$. Similarly, for the k-pebble definability test for CQs. This holds even if restricted to the case when S^+ and S^- are unary relations.*

As a corollary to Theorems 7 and 8, we obtain the following interesting result:

▶ **Corollary 9.** *The problems* $\mathsf{TW}(k)$-QUERY-BY-EXAMPLE *and* $\mathsf{TW}(k)$-DEFINABILITY *are* EXPTIME*-complete for each fixed $k \geq 1$. This holds even if restricted to the case when the relations to be explained/defined are unary.*

We now provide a brief outline of the main ideas used for proving the lower bounds in Theorem 8. Let us first notice that in the case of the general QBE/definability tests for CQs, a coNEXPTIME lower bound is obtained in [22] as follows:

1. It is first shown that the following *product homomorphism problem* (PHP) is NEXPTIME-hard: Given databases $\mathcal{D}_1, \dots, \mathcal{D}_m$ and $\mathcal{D}$, is it the case that $\prod_{1\leq i\leq m} \mathcal{D}_i \to \mathcal{D}$?
2. It is then shown that there is an easy polynomial-time reduction from PHP to the problem of checking whether the QBE/definability test fails on its input.

The ideas used for proving (2) can be easily adapted to show that there is a polynomial-time reduction from the following *relaxed* version of PHP to the problem of checking whether the k-pebble QBE/definability test fails on its input:

PROBLEM :	k-PEBBLE PHP (for $k > 1$)
INPUT :	Databases $\mathcal{D}_1, \dots, \mathcal{D}_m$ and $\mathcal{D}$ over the same schema
QUESTION :	Is it the case that $\prod_{1\leq i\leq m} \mathcal{D}_i \to_k \mathcal{D}$?

We establish that this relaxed version of PHP is EXPTIME-complete for each fixed $k > 1$:

▶ **Theorem 10.** *The problem k-PEBBLE PHP is EXPTIME-complete for each fixed $k > 1$.*

To prove this result, we exploit techniques from [16, 15] that study the complexity of pebble games. In particular, it is shown in [16] that for each fixed $k > 1$, checking whether $\mathcal{D} \to_k \mathcal{D}'$ is P-complete. The proof uses an involved reduction from the *monotone circuit value problem*, that is, given a monotone circuit C, it constructs two databases $\mathcal{D}_C$ and $\mathcal{D}'_C$ such that the value of C is 1 if and only if $\mathcal{D}_C \to_k \mathcal{D}'_C$.

In our case, to show that k-PEBBLE PHP is EXPTIME-hard for each fixed $k > 1$, we reduce from the following well-known EXPTIME-complete problem: Given an alternating Turing machine M and a positive integer n, decide whether M accepts the empty tape using n space. The latter problem can be easily recast as a circuit value problem: We can construct a circuit $C_{M,n}$ such that the value of $C_{M,n}$ is 1 if and only if M accepts the empty tape using n space. The main idea of our reduction is to construct databases $\mathcal{D}_1, \ldots, \mathcal{D}_m$ and $\mathcal{D}$, given M and n, such that:

$$\prod_{1 \le i \le m} \mathcal{D}_i \to_k \mathcal{D} \iff \mathcal{D}_{C_{M,n}} \to_k \mathcal{D}'_{C_{M,n}},$$

where $\mathcal{D}_{C_{M,n}}$ and $\mathcal{D}'_{C_{M,n}}$ are defined as in [16].

A natural approach then is to construct $\mathcal{D}_1, \ldots, \mathcal{D}_m, \mathcal{D}$ such that $\prod_{1 \le i \le m} \mathcal{D}_i$ and $\mathcal{D}$ roughly coincide with $\mathcal{D}_{C_{M,n}}$ and $\mathcal{D}'_{C_{M,n}}$. However, there is a problem with this: the databases $\mathcal{D}_{C_{M,n}}$ and $\mathcal{D}'_{C_{M,n}}$ closely resemble the circuit $C_{M,n}$, but the size of $C_{M,n}$ is exponential in $|M|$ and n, and so are the sizes of $\mathcal{D}_{C_{M,n}}$ and $\mathcal{D}'_{C_{M,n}}$. Although it is possible to codify the exponential size database $\mathcal{D}_{C_{M,n}}$ using a product of polynomial size databases $\mathcal{D}_1, \ldots, \mathcal{D}_m$, we cannot do the same with the exponential size $\mathcal{D}'_{C_{M,n}}$ using $\mathcal{D}$ only. To overcome this, we need to extend the techniques in [16] and show that the complexity of the existential k-pebble game is P-complete even over a *fixed template*:

▶ **Lemma 11.** *For each fixed $k > 1$, there is a database $\mathcal{D}_k$ that only depends on k, such that the following problem is* P*-complete: Given a database $\mathcal{D}$, decide whether $\mathcal{D} \to_k \mathcal{D}_k$.*

To prove this, we again use a reduction from the circuit value problem that given a circuit C constructs a database $\tilde{\mathcal{D}}_C$ such that C takes value 1 if and only if $\tilde{\mathcal{D}}_C \to_k \mathcal{D}_k$. We then use the following idea to prove that k-PEBBLE PHP is EXPTIME-complete: Given M and n, we construct in polynomial time databases $\mathcal{D}_1, \ldots, \mathcal{D}_m$ and $\mathcal{D}$ such that $\prod_{1 \le i \le m} \mathcal{D}_i$ and $\mathcal{D}$ roughly coincide with $\tilde{\mathcal{D}}_{C_{M,n}}$ and $\mathcal{D}_k$, respectively. It then follows that:

$$\prod_{1 \le i \le m} \mathcal{D}_i \to_k \mathcal{D} \iff \tilde{\mathcal{D}}_{C_{M,n}} \to_k \mathcal{D}_k \iff M \text{ accepts the empty tape using } n \text{ space.}$$

4.3 Evaluating the result of TW(k)-explanations

Recall that computing the result of CQ-explanations might require double exponential time. For TW(k)-explanations, instead, we can do this in single exponential time.

▶ **Theorem 12.** *Fix $k \ge 1$. There is a single exponential time algorithm that, given a database $\mathcal{D}$ and n-ary relations S^+ and S^- over $\mathcal{D}$, does the following:*

1. *It checks whether there is a TW(k)-explanation for S^+ and S^- over $\mathcal{D}$, and*
2. *if the latter holds, it computes the evaluation $q(\mathcal{D})$ of one such TW(k)-explanation q.*

Proof. We first check in exponential time the existence of one such $\mathsf{TW}(k)$-explanation for S^+ and S^- over $\mathcal{D}$ using the $(k+1)$-pebble QBE test for CQs. If such $\mathsf{TW}(k)$-explanation exists, we compute in exponential time the set S^e of all n-ary tuples $\bar{b}$ over $\mathcal{D}$ such that $\prod_{\bar{a}\in S^+}(\mathcal{D},\bar{a}) \rightarrow_{k+1} (\mathcal{D},\bar{b})$. Notice, in particular, that $S^+ \subseteq S^e$ and $S^e \cap S^- = \emptyset$. Moreover, it can be shown that $S^e = q(\mathcal{D})$ for some $\mathsf{TW}(k)$-explanation q for S^+ and S^- over $\mathcal{D}$. ◀

Notably, the previous result computes the result of a $\mathsf{TW}(k)$-explanation q for S^+ and S^- over $\mathcal{D}$ without explicitly computing q. One might wonder whether it is possible to also include q in the output of the algorithm. The answer is negative, and the reason is that $\mathsf{TW}(k)$-explanations/definitions can be double exponentially large in the worst case:

▶ **Proposition 13.** *Fix $k \geq 1$. The following holds:*

1. *Assume that there is a $\mathsf{TW}(k)$-explanation for S^+ and S^- over $\mathcal{D}$. Then there is one such $\mathsf{TW}(k)$-explanation of at most double exponential size.*
2. *There is a family $(\mathcal{D}_n, S_n^+, S_n^-)_{n\geq 0}$ of tuples of databases $\mathcal{D}_n$ and relations S_n^+ and S_n^- over $\mathcal{D}_n$, such that (a) the combined size of $\mathcal{D}_n$, S_n^+, and S_n^- is polynomial in n, (b) there is a $\mathsf{TW}(k)$-explanation for S_n^+ and S_n^- over $\mathcal{D}_n$, and (c) the size of the smallest such $\mathsf{TW}(k)$-explanation is at least 2^{2^n}.*

The same holds for $\mathsf{TW}(k)$-definitions.

Proof. From the proof of Theorem 7, whenever there is a $\mathsf{TW}(k)$-explanation for S^+ and S^- over $\mathcal{D}$ this can be assumed to be the CQ $q = \bigwedge_{\bar{b}\in S^-} q_{\bar{b}}(\bar{x})$. From Proposition 6, each such $q_{\bar{b}}$ is of exponential size in the combined size of $\prod_{\bar{a}\in S^+}(\mathcal{D},\bar{a})$ and $(\mathcal{D},\bar{b})$, i.e., double exponential in the size of $\mathcal{D}$, S^+ and S^-. Thus, the size of q is at most double exponential in that of $\mathcal{D}$, S^+ and S^-. The lower bound follows by inspection of the proof of Theorem 8. ◀

Notice that this establishes a difference with CQ-explanations/definitions, which are at most of exponential size (see Proposition 4).

5 Desynchronizing the direct product

We now look at the other source of complexity for the QBE and definability tests for CQs: The construction of the direct product $\prod_{\bar{a}\in S^+}(\mathcal{D},\bar{a})$. It is a priori not obvious how to define reasonable approximations of this construction with a meaningful theoretical interpretation. As a first step in this direction, we look at a simple idea that has been applied in the study of CQ-definability: We "desynchronize" this direct product and consider each tuple $\bar{a} \in S^+$ in isolation. This leads to the following relaxed test:

- Desynchronized QBE test for CQs: Takes as input a database $\mathcal{D}$ and n-ary relations S^+, S^- over $\mathcal{D}$. It accepts iff for each $\bar{a} \in S^+$ and $\bar{b} \in S^-$ it is the case that $(\mathcal{D},\bar{a}) \not\rightarrow (\mathcal{D},\bar{b})$.

Similarly, we define the desynchronized definability test for CQs. Notice that, unlike the previous tests we have presented in the paper, the desynchronized tests do not require any safeness condition (for reasons we explain below).

It follows from [1] that these tests capture the notion of explanations/definitions for the class of *unions* of CQs (UCQs). Recall that a UCQ is a formula Q of the form $\bigvee_{1\leq i\leq m} q_i(\bar{x})$, where the $q_i(\bar{x})$'s are CQs over the same schema. The evaluation $Q(\mathcal{D})$ of Q over database $\mathcal{D}$ corresponds to $\bigcup_{1\leq i\leq m} q_i(\mathcal{D})$. We denote by UCQ the class of UCQs. We then obtain the following:

▶ **Theorem 14** (implicit in [1]). *Consider a database $\mathcal{D}$ and n-ary relations S^+ and S^- over $\mathcal{D}$. There is a UCQ-explanation for S^+ and S^- over $\mathcal{D}$ if and only if the desynchronized*

QBE test for CQs accepts $\mathcal{D}$, S^+, and S^-. Similarly, for the UCQ*-definitions of S^+ and the desynchronized definability test for CQs.*

In this case, the *canonical* UCQ-explanation/definition corresponds to $Q = \bigcup_{\bar{a} \in S^+} (\mathcal{D}, \bar{a})$. This explains why no safeness condition is required on the desynchronized tests, as each pair of the form $(\mathcal{D}, \bar{a})$, for $\bar{a} \in S^+$, is safe by definition. Notice that Q consists of polynomially many CQs of polynomial size. Its evaluation $Q(\mathcal{D})$ over a database $\mathcal{D}$ can thus be computed in single exponential time (as opposed to the double exponential time needed to evaluate the canonical CQ-explanation $\prod_{\bar{a} \in S^+} (\mathcal{D}, \bar{a})$).

It is easy to see that the desynchronization of the direct product reduces the complexity of the general tests from coNEXPTIME to coNP. It follows from [1] that this bound is optimal. As a corollary to Theorem 14 we thus obtain that QBE/definability for UCQs are coNP-complete:

▶ **Proposition 15.** [1] *The following statements hold:*
1. *Deciding whether the desynchronized QBE test for CQs accepts $(\mathcal{D}, S^+, S^-)$ is* coNP*-complete. Similarly, for the desynchronized definability test for CQs.*
2. UCQ-QUERY-BY-EXAMPLE *and* UCQ-DEFINABILITY *are* coNP*-complete.*

5.1 Combining both relaxations

By combining both relaxations (replacing homomorphism tests with relations $\rightarrow_k$, for $k > 1$, and desynchronizing direct products) we obtain the *desynchronized k-pebble QBE (resp., definability) test for CQs.* Its definition coincides with that of the desynchronized QBE (resp., definability) test for CQs given above, save that now the homomorphism test $(\mathcal{D}, \bar{a}) \rightarrow (\mathcal{D}, \bar{b})$ is replaced by $(\mathcal{D}, \bar{a}) \rightarrow_k (\mathcal{D}, \bar{b})$. As is to be expected from the previous charaterizations, this test captures definability by the class of UCQs of bounded treewidth. Formally, let $\mathsf{UTW}(k)$ be the class of unions of CQs in $\mathsf{TW}(k)$ (for $k \geq 1$). Then:

▶ **Theorem 16.** *Fix $k \geq 1$. Consider a database $\mathcal{D}$ and n-ary relations S^+ and S^- over $\mathcal{D}$. There is a* $\mathsf{UTW}(k)$*-explanation for S^+ and S^- over $\mathcal{D}$ if and only if the desynchronized $(k+1)$-pebble QBE test for CQs accepts $\mathcal{D}$, S^+, and S^-. Similarly, for the* $\mathsf{UTW}(k)$*-definitions of S^+ and the desynchronized $(k+1)$-pebble definability test for CQs.*

Furthermore, in case there is a $\mathsf{UTW}(k)$-explanation for S^+ and S^- over $\mathcal{D}$ (resp., a $\mathsf{UTW}(k)$-definition of S^+ over $\mathcal{D}$), then there is one such explanation/definition given by a union of polynomially many CQs in $\mathsf{TW}(k)$, each one of which is of at most exponential size.

Interestingly, the combination of both relaxations yields tractability for the QBE test. In contrast, the definability test remains coNP-complete. The difference lies on the fact that the QBE test only needs to perform a polynomial number of tests of the form $(\mathcal{D}, \bar{a}) \rightarrow_k (\mathcal{D}, \bar{b})$ for each $\bar{a} \in S^+$ (one for each tuple $\bar{b} \in S^-$), while the definability test needs to perform exponentially many such tests (one for each tuple $\bar{b}$ outside S^+). Then:

▶ **Proposition 17.** *The following statements hold:*
1. *Deciding whether the desynchronized k-pebble QBE test for CQs accepts $(\mathcal{D}, S^+, S^-)$ can be solved in polynomial time for each fixed $k > 1$. As a consequence,* $\mathsf{UTW}(k)$-QUERY-BY-EXAMPLE *is in polynomial time for each fixed $k \geq 1$.*
2. *If a* $\mathsf{UTW}(k)$*-explanation for S^+ and S^- over $\mathcal{D}$ exists, we can compute the evaluation $Q(\mathcal{D})$ of one such explanation Q in exponential time.*
3. *Deciding whether the desynchronized k-pebble definability test for CQs accepts $(\mathcal{D}, S^+)$ is* coNP*-complete for each fixed $k > 1$. As a consequence,* $\mathsf{UTW}(k)$-DEFINABILITY *is* coNP*-complete for each $k \geq 1$.*

6 Conjunctive regular path queries

We now switch to study the QBE and definability problems in the context of graph databases. Let Σ be a finite alphabet. Recall that a *graph database* $\mathcal{G} = (V, E)$ over Σ consists of a finite set V of nodes and a set $E \subseteq V \times \Sigma \times V$ of directed edges labeled in Σ (i.e., $(v, a, v') \in E$ represents the fact that there is an a-labeled edge from node v to node v' in $\mathcal{G}$). A *path* in $\mathcal{G}$ is a sequence

$$\eta \;=\; v_0 a_1 v_1 a_2 v_2 \dots v_{k-1} a_k v_k, \qquad \text{for } k \geq 0,$$

such that $(v_{i-1}, a_i, v_i) \in E$ for each $1 \leq i \leq k$. The *label* of η, denoted $\mathsf{label}(\eta)$, is the word $a_1 a_2 \dots a_k$ in Σ^*. Notice that v is a path for each node $v \in V$. The label of such path is the empty word ε.

The basic navigational mechanism for querying graph databases is the class of *regular path queries*, or RPQs (see, e.g., [25, 5]). An RPQ L over alphabet Σ is a regular expression over Σ. The *evaluation* $L(\mathcal{G})$ of L over graph database $\mathcal{G}$ consists of those pairs (v, v') of nodes in $\mathcal{G}$ such that there is a path η in $\mathcal{G}$ from v to v' whose label $\mathsf{label}(\eta)$ satisfies L. The analogue of CQs in the context of graph databases is the class of *conjunctive* RPQs, or CRPQs [8]. Formally, a CRPQ γ over Σ is an expression of the form:

$$\exists \bar{z}(L_1(x_1, y_1) \wedge \dots \wedge L_m(x_m, y_m)),$$

where each L_i is a RPQ over Σ, for $1 \leq i \leq m$, and $\bar{z}$ is a tuple of variables among $\{x_1, y_1, \dots, x_m, y_m\}$. We write $\gamma(\bar{x})$ to denote that $\bar{x}$ is the tuple of free variables of γ. A homomorphism from γ to the graph database $\mathcal{G}$ is a mapping h from $\{x_1, y_1, \dots, x_m, y_m\}$ to the nodes of $\mathcal{G}$, such that $(h(x_i), h(y_i)) \in L_i(\mathcal{G})$ for each $1 \leq i \leq m$. The evaluation $\gamma(\mathcal{G})$ of $\gamma(\bar{x})$ over $\mathcal{G}$ is the set of tuples $h(\bar{x})$ such that h a homomorphism from γ to $\mathcal{G}$. We denote the class of CRPQs by CRPQ.

6.1 The QBE and definability tests for CRPQs

We present QBE/definability tests for CRPQs in the same spirit than the tests for CQs, save that we now use a notion of *strong homomorphism* from a product $\prod_{1 \leq i \leq n} \mathcal{G}_i$ of directed graphs to a single directed graph $\mathcal{G}$. This notion preserves, in a precise sense defined below, the languages defined by pairs of nodes in $\prod_{1 \leq i \leq n} \mathcal{G}_i$. Interestingly, these tests yield a coNEXPTIME upper bound for the QBE/definability problems for CRPQs, which improves the EXPSPACE upper bound from [1]. In conclusion, QBE/definability for CRPQs is no more difficult than for CQs.

We start with some notation. Let v and v' be nodes in a graph database $\mathcal{G}$. We define the following language in Σ^*:

$$L^{\mathcal{G}}_{v,v'} \;:=\; \{\mathsf{label}(\eta) \mid \eta \text{ is a path in } \mathcal{G} \text{ from } v \text{ to } v'\}.$$

Moreover, if $\mathcal{G}_1 = (V_1, E_1)$ and $\mathcal{G}_2 = (V_2, E_2)$ are graph databases over Σ, their direct product $\mathcal{G}_1 \otimes \mathcal{G}_2$ is the graph database (V, E) such that $V = V_1 \times V_2$ and there is an a-labeled edge in E from node (v_1, v_2) to node (v_1', v_2') if and only if $(v_1, a, v_2) \in E_1$ and $(v_1', a, v_2') \in E_2$.

Let then $\mathcal{G}_1, \dots, \mathcal{G}_n$ and $\mathcal{G}$ be graph databases over Σ. A *strong homomorphism* from $\prod_{1 \leq i \leq n} \mathcal{G}_i$ to $\mathcal{G}$ is a mapping h from the nodes of $\prod_{1 \leq i \leq n} \mathcal{G}_i$ to the nodes of $\mathcal{G}$ such that for each pair $\bar{v} = (v_1, \dots, v_n)$ and $\bar{v}' = (v_1', \dots, v_n')$ of nodes in $\prod_{1 \leq i \leq n} \mathcal{G}_i$, it is the case that:

$$L^{\mathcal{G}_i}_{v_i, v_i'} \;\subseteq\; L^{\mathcal{G}}_{h(\bar{v}), h(\bar{v}')}, \qquad \text{for some coordinate } i \text{ with } 1 \leq i \leq n.$$

We write $\prod_{1\leq i\leq n}\mathcal{G}_i \Rightarrow \mathcal{G}$ when there is a strong homomorphism h from $\prod_{1\leq i\leq n}\mathcal{G}_i$ to $\mathcal{G}$. Note that in this case, h must also be a (usual) homomorphism from $\prod_{1\leq i\leq n}\mathcal{G}_i$ to $\mathcal{G}$, i.e., $\prod_{1\leq i\leq n}\mathcal{G}_i \Rightarrow \mathcal{G}$ implies $\prod_{1\leq i\leq n}\mathcal{G}_i \to \mathcal{G}$. The next example shows that the converse does not hold in general:

▶ **Example 18.** Let $\vec{C}_n$ be the directed cycle of length n over $\{1, 2, \ldots, n\}$. We assume $\vec{C}_n$ to be represented as a graph database over the unary alphabet $\Sigma = \{a\}$. We then have that $\vec{C}_2 \otimes \vec{C}_3 \to \vec{C}_6$, since $\vec{C}_2 \otimes \vec{C}_3$ is isomorphic to $\vec{C}_6$ as shown below (we omit the labels):

$$\begin{tikzcd}
(1,1) \arrow[dr] & (1,2) \arrow[dr] & (1,3) \arrow[dll] \\
(2,1) \arrow[ur] & (2,2) \arrow[ur] & (2,3) \arrow[ull]
\end{tikzcd}$$

On the other hand, $\vec{C}_2 \otimes \vec{C}_3 \not\Rightarrow \vec{C}_6$. To see this, take e.g. the homomorphism h defined as

$$\{(1,1) \mapsto 1,\ (2,2) \mapsto 2,\ (1,3) \mapsto 3,\ (2,1) \mapsto 4,\ (1,2) \mapsto 5,\ (2,3) \mapsto 6\}.$$

This is not a strong homomorphism as witnessed by the pair $(1,1)$ and $(2,2)$. Indeed, we have that:

$$\big(h(1,1) = 1 \text{ and } h(2,2) = 2\big) \text{ but } \big(L_{1,2}^{\vec{C}_2} \not\subseteq L_{1,2}^{\vec{C}_6} \text{ and } L_{1,2}^{\vec{C}_3} \not\subseteq L_{1,2}^{\vec{C}_6}.\big)$$

The reason is that $aaa \in L_{1,2}^{\vec{C}_2}$, $aaaa \in L_{1,2}^{\vec{C}_3}$, but none of these words is in $L_{1,2}^{\vec{C}_6}$. The same holds for any homomorphism $h : \vec{C}_2 \otimes \vec{C}_3 \to \vec{C}_6$. ◀

If $(\mathcal{G}_1, \bar{a}_1), \ldots, (\mathcal{G}_n, \bar{a}_n)$ and $(\mathcal{G}, \bar{b})$ are graph databases with distinguished tuple of elements, then we write $\prod_{1\leq i\leq n}(\mathcal{G}_i, \bar{a}_i) \Rightarrow (\mathcal{G}, \bar{b})$ if there is a strong homomorphism h from $\prod_{1\leq i\leq n}\mathcal{G}_i$ to $\mathcal{G}$ such that $h(\bar{a}_1 \otimes \cdots \otimes \bar{a}_n) = \bar{b}$. Next we present our tests for CRPQs:

- QBE test for CRPQs: Takes as input a graph database $\mathcal{G}$ and n-ary relations S^+ and S^- over $\mathcal{G}$. It accepts if and only if $\prod_{\bar{a}\in S^+}(\mathcal{G}, \bar{a}) \not\Rightarrow (\mathcal{G}, \bar{b})$ for each tuple $\bar{b} \in S^-$.
- Definability test for CRPQs: Takes as input a graph database $\mathcal{G}$ and an n-ary relation S^+ over $\mathcal{G}$. It accepts if and only if $\prod_{\bar{a}\in S^+}(\mathcal{G}, \bar{a}) \not\Rightarrow (\mathcal{G}, \bar{b})$ for each n-ary tuple $\bar{b} \notin S^+$.

As it turns out, our tests characterize the non-existence of CRPQ-explanations/definitions. (Notice that unlike Proposition 2, we need no safety conditions on QBE/definability tests for CRPQs for this characterization to hold).

▶ **Theorem 19.** *The following hold:*

1. *Let $\mathcal{G}$ be a database and S^+, S^- relations over $\mathcal{G}$. There is a* CRPQ*-explanation for S^+ and S^- over $\mathcal{G}$ if and only if the QBE test for CRPQs accepts $\mathcal{G}$, S^+, and S^-.*
2. *Let $\mathcal{G}$ be a database and S^+ a relation over $\mathcal{G}$. There is a* CRPQ*-definition for S^+ over $\mathcal{G}$ if and only if the definability test for CRPQs accepts $\mathcal{G}$ and S^+.*

Since containment of regular languages can be checked in polynomial space [21], it is straightforward to check that both tests can be carried out in coNEXPTIME. Thus:

▶ **Theorem 20.** CRPQ-QUERY-BY-EXAMPLE *and* CRPQ-DEFINIBILITY *are in* coNEXPTIME.

Whether these problems are complete for coNEXPTIME is left as an open question.

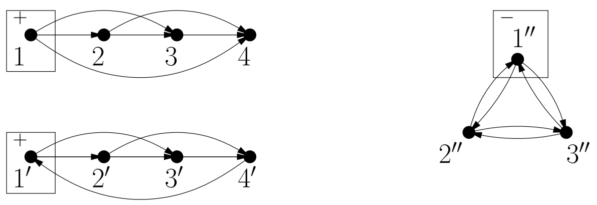

Figure 1 The graph database $\mathcal{G}$ from Example 21.

CRPQ vs UCQ explanations. It is easy to see that if there is a CRPQ-explanation for S^+ and S^- over $\mathcal{G}$, then there is also a UCQ-explanation [1]. One may wonder then if QBE for CRPQs and UCQs coincide. If this was the case, we would directly obtain a coNP upper bound for CRPQ-QUERY-BY-EXAMPLE from Proposition 15 (which establishes that UCQ-QUERY-BY-EXAMPLE is in coNP). The next example shows that this is not the case:

▶ **Example 21.** Consider the graph database $\mathcal{G}$ over $\Sigma = \{a\}$ given by the three connected components depicted in Figure 1 (we omit the labels). Let $S^+ = \{1, 1'\}$ and $S^- = \{1''\}$. Clearly, $(\mathcal{G}, 1) \not\to (\mathcal{G}, 1'')$ and $(\mathcal{G}, 1') \not\to (\mathcal{G}, 1'')$, since the underlying graph of each component on the left-hand side is a clique of size 4, while the one on the right-hand side is a clique of size 3. It follows that there is a UCQ-explanation for S^+ and S^- over $\mathcal{G}$. On the other hand, a straightforward construction shows that $(\mathcal{G}, 1) \otimes (\mathcal{G}, 1') \Rightarrow (\mathcal{G}, 1'')$. The intuition is that, since $(4', 1')$ and $(1, 4)$ have opposite direction, they do not synchronize in the product and, thus, the product does not contain a clique of size 4. We conclude that there is no CRPQ-explanation for S^+ and S^- over $\mathcal{G}$. ◀

6.2 Relaxing the QBE and definability tests for CRPQs

In this section, we develop relaxations of the tests for CRPQs based on the ones we studied for CQs in the previous sections. Let us start by observing that desynchronizing the direct product trivializes the problem in this case: In fact, as expected the *desynchronized QBE/definability tests for CRPQs* characterize QBE/definability for the *unions* of CRPQs (UCRPQ). It is known, on the other hand, that QBE/definability for UCRPQ and UCQ coincide [1]. The results then follow directly from the ones obtained in Section 5 for UCQs. In particular, UCRPQ-QUERY-BY-EXAMPLE and UCRPQ-DEFINABILITY are coNP-complete.
+

We thus concentrate on the most interesting case, which is the relaxation of the homomorphism tests. In order to approximate the strong homomorphism test, we consider a variant of the existential pebble game. Fix $k > 1$. Let $(\mathcal{G}_1, \bar{a}_1), \dots, (\mathcal{G}_n, \bar{a}_n)$ and $(\mathcal{G}, \bar{b})$ be graph databases over Σ with distinguished tuples of elements. We define $\bar{a} := \bar{a}_1 \otimes \cdots \otimes \bar{a}_n$. The *strong existential k-pebble game* on $\prod_{1\leq i\leq n}(\mathcal{G}_i, \bar{a}_i)$ and $(\mathcal{G}, \bar{b})$ is played as the existential k-pebble game on $\prod_{1\leq i\leq n}(\mathcal{G}_i, \bar{a}_i)$ and $(\mathcal{G}, \bar{b})$, but now, at each round, if $c_1, \dots, c_k$ and $d_1, \dots, d_k$ are the elements covered by pebbles on $\prod_{1\leq i\leq n}\mathcal{G}_i$ and $\mathcal{G}$, respectively, then the duplicator needs to ensure that $((c_1, \dots, c_k, \bar{a}), (d_1, \dots, d_k, \bar{b}))$ is a *strong partial homomorphism* from $\prod_{1\leq i\leq n}\mathcal{G}_i$ and $\mathcal{G}$. This means that for every pair $\bar{v} = (v_1, \dots, v_n)$ and $\bar{v}' = (v_1', \dots, v_n')$ of nodes in $\prod_{1\leq i\leq n}\mathcal{G}_i$ that appear in $(c_1, \dots, c_k, \bar{a})$, if u and u' are the elements in $(d_1, \dots, d_k, \bar{b})$ that correspond to $\bar{v}$ and $\bar{v}'$, respectively, then:

$$L^{\mathcal{G}_i}_{v_i, v_i'} \subseteq L^{\mathcal{G}}_{u,u'}, \quad \text{for some coordinate } i \text{ with } 1 \leq i \leq n.$$

We write $\prod_{1\leq i\leq n}(\mathcal{G}_i, \bar{a}_i) \Rightarrow_k (\mathcal{G}, \bar{b})$ if the duplicator has a winning strategy in the strong existential k-pebble game on $\prod_{1\leq i\leq n}(\mathcal{G}_i, \bar{a}_i)$ and $(\mathcal{G}, \bar{b})$.

By replacing the notion of strong homomorphism $\Rightarrow$ with its approximation $\Rightarrow_k$, for a fixed $k > 1$, we can then define the following relaxed test:

- k-pebble QBE test for CRPQs: Takes as input a graph database $\mathcal{G}$ and n-ary relations S^+ and S^- over $\mathcal{G}$. It accepts iff $\prod_{\bar{a}\in S^+}(\mathcal{G},\bar{a}) \not\Rightarrow_k (\mathcal{G},\bar{b})$ for each tuple $\bar{b} \in S^-$.

The k-pebble definability test for CRPQs is defined analogously. As in the case of CQs, these tests characterize the existence of CRPQs-explanations/definitions of treewidth at most k. Formally, the treewidth of a CRPQ $\gamma = \exists\bar{y} \bigwedge_{1\leq i\leq m} L_i(x_i, y_i)$ is the treewidth of the undirected graph that contains as nodes the existentially quantified variables of γ, i.e., those in $\bar{y}$, and whose set of edges is $\{\{x_i, y_i\} \mid 1 \leq i \leq m,\ x_i \neq y_i\}$. We denote by $\mathsf{TW}_{\mathrm{crpq}}(k)$ the class of CRPQs of treewidth at most k (for $k \geq 1$). Then:

▶ **Theorem 22.** *Fix $k \geq 1$. Consider a database $\mathcal{G}$ and n-ary relations S^+ and S^- over $\mathcal{G}$.*

1. *There is a $\mathsf{TW}_{\mathrm{crpq}}(k)$-explanation for S^+ and S^- over $\mathcal{G}$ if and only if the $(k+1)$-pebble QBE test for CRPQs accepts $\mathcal{G}$, S^+ and S^-.*
2. *There is a $\mathsf{TW}_{\mathrm{crpq}}(k)$-definition for S^+ over $\mathcal{G}$ if and only if the $(k+1)$-pebble definability test for CRPQs accepts $\mathcal{G}$ and S^+.*

Using similar ideas as for the existential k-pebble game, it is possible to prove that the problem of checking whether $\prod_{1\leq i\leq n}(\mathcal{G}_i, \bar{a}_i) \Rightarrow_k (\mathcal{G}, \bar{b})$, given $(\mathcal{G}_1, \bar{a}_1), \ldots, (\mathcal{G}_n, \bar{a}_n)$ and $(\mathcal{G}, \bar{b})$, can be solved in exponential time for each fixed $k > 1$. We then obtain that the k-pebble QBE/definability tests for CRPQs take exponential time, and from Theorem 22 that $\mathsf{TW}_{\mathrm{crpq}}(k)$-QUERY-BY-EXAMPLE and $\mathsf{TW}_{\mathrm{crpq}}(k)$-DEFINABILITY are in EXPTIME (same than for $\mathsf{TW}(k)$ as stated in Corollary 9). We also obtain an exponential upper bound on the cost of evaluating a $\mathsf{TW}_{\mathrm{crpq}}(k)$-explanation (in case it exists):

▶ **Proposition 23.** *Fix $k \geq 1$. The following statements hold:*

1. $\mathsf{TW}_{\mathrm{crpq}}(k)$-QUERY-BY-EXAMPLE *and* $\mathsf{TW}_{\mathrm{crpq}}(k)$-DEFINABILITY *are in* EXPTIME.
2. *Moreover, in case that there is a $\mathsf{TW}_{\mathrm{crpq}}(k)$-explanation of S^+ and S^- over $\mathcal{G}$, the evaluation $\gamma(\mathcal{G})$ of one such explanation γ over $\mathcal{G}$ can be computed in exponential time.*

7 Future work

We have left some problems open. The most notable one is determining the precise complexity of QBE/definability for CRPQs (resp., CRPQs of bounded treewidth). We have only obtained upper bounds for these problems that show that they are no more difficult than for CQs, but proving matching lower bounds seems challenging.

An interesting line for future research is studying what to do when no explanation/definition exists for a set of examples. In such cases one might want to compute a query that minimizes the "error", e.g., the number of misclassified examples. We plan to study whether the techniques presented in this paper can be extended to deal with such problems.

Acknowledgements. We are grateful to Leonid Libkin for their helpful comments in earlier versions of the paper and to Timos Antonopoulos for enlightening discussions about the notion of definability. We are also indebted to the reviewers of this article who helped us improving the presentation. In particular, one of the revieweres identified a subtle but important issue with the QBE tests presented in the paper that would have gone unnoticed otherwise. Barceló and Romero are funded by the Millennium Nucleus Center for Semantic Web Research under Grant NC120004. Romero is also funded by a Conicyt PhD scholarship.

References

1 Timos Antonopoulos, Frank Neven, and Frédéric Servais. Definability problems for graph query languages. In *ICDT*, pages 141–152, 2013.

2 Marcelo Arenas and Gonzalo I. Díaz. The exact complexity of the first-order logic definability problem. *ACM Trans. Database Syst.*, 41(2), to 2016.

3 Marcelo Arenas, Gonzalo I. Díaz, and Egor V. Kostylev. Reverse engineering sparql queries. In *WWW*, 2016.

4 Albert Atserias, Phokion G. Kolaitis, and Moshe Y. Vardi. Constraint propagation as a proof system. In *CP*, pages 77–91, 2004.

5 Pablo Barceló. Querying graph databases. In *PODS*, pages 175–188, 2013.

6 Angela Bonifati, Radu Ciucanu, and Aurélien Lemay. Learning path queries on graph databases. In *EDBT*, pages 109–120, 2015.

7 Angela Bonifati, Radu Ciucanu, and Slawek Staworko. Learning join queries from user examples. *ACM Trans. Database Syst.*, 40(4):24, 2016.

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

9 Chandra Chekuri and Anand Rajaraman. Conjunctive query containment revisited. *Theor. Comput. Sci.*, 239(2):211–229, 2000.

10 Sara Cohen and Yaacov Y. Weiss. Learning tree patterns from example graphs. In *ICDT*, pages 127–143, 2015.

11 Mariano P. Consens and Alberto O. Mendelzon. Graphlog: a visual formalism for real life recursion. In *PODS*, pages 404–416, 1990.

12 Víctor Dalmau, Phokion G. Kolaitis, and Moshe Y. Vardi. Constraint satisfaction, bounded treewidth, and finite-variable logics. In *CP*, pages 310–326, 2002.

13 Rina Dechter. From local to global consistency. *Artif. Intell.*, 55(1):87–108, 1992.

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

15 Martin Grohe. Equivalence in finite-variable logics is complete for polynomial time. *Combinatorica*, 19(4):507–532, 1999.

16 Phokion G. Kolaitis and Jonathan Panttaja. On the complexity of existential pebble games. In *CSL*, pages 314–329, 2003.

17 Phokion G. Kolaitis and Moshe Y. Vardi. On the expressive power of datalog: Tools and a case study. *J. Comput. Syst. Sci.*, 51(1):110–134, 1995.

18 Phokion G. Kolaitis and Moshe Y. Vardi. A game-theoretic approach to constraint satisfaction. In *AAAI*, pages 175–181, 2000.

19 Hao Li, Chee-Yong Chan, and David Maier. Query from examples: An iterative, data-driven approach to query construction. *PVLDB*, 8(13):2158–2169, 2015.

20 Slawek Staworko and Piotr Wieczorek. Characterizing XML twig queries with examples. In *ICDT*, pages 144–160, 2015.

21 Larry J. Stockmeyer and Albert R. Meyer. Word problems requiring exponential time: Preliminary report. In *STOC*, pages 1–9, 1973.

22 Balder ten Cate and Víctor Dalmau. The product homomorphism problem and applications. In *ICDT*, pages 161–176, 2015.

23 Quoc Trung Tran, Chee Yong Chan, and Srinivasan Parthasarathy. Query reverse engineering. *VLDB J.*, 23(5):721–746, 2014.

24 Ross Willard. Testing expressibility is hard. In *CP*, pages 9–23, 2010.

25 Peter T. Wood. Query languages for graph databases. *SIGMOD Record*, 41(1):50–60, 2012.

26 Meihui Zhang, Hazem Elmeleegy, Cecilia M. Procopiuc, and Divesh Srivastava. Reverse engineering complex join queries. In *SIGMOD*, pages 809–820, 2013.