Conjunctive Queries Determinacy and Rewriting

Daniel Pasailă
LSV, ENS Cachan
61, avenue du president Wilson
94235 Cachan cedex, France
daniel.pasaila@gmail.com

ABSTRACT

The problem of whether a query Q can be answered using a set of views $\mathbf{V}$ studies the possibility of computing Q when only the answers to the given set of views are available. In information-theoretic terms, we say that $\mathbf{V}$ determines Q iff for any two databases D_1, D_2, $\mathbf{V}(D_1) = \mathbf{V}(D_2)$ implies $Q(D_1) = Q(D_2)$. In the case that $\mathbf{V}$ determines Q, we also study the existence of equivalent rewritings of Q in terms of $\mathbf{V}$ in a specific rewriting language. Having a view language $\mathcal{V}$ and a query language $\mathcal{Q}$ we say that $\mathcal{V}$-to-$\mathcal{Q}$ determinacy is decidable if there is an algorithm which, given a view $\mathbf{V} \in \mathcal{V}$ and a query $Q \in \mathcal{Q}$, outputs whether $\mathbf{V}$ determines Q.

We focus on the case where the views and the query are defined by subclasses of conjunctive queries and investigate in which cases $\mathbf{V}$ determines Q and the existence of equivalent rewritings of Q in terms of $\mathbf{V}$. We define the class of CQ_{cgraph} queries as binary CQ queries whose body, if viewed as an undirected graph, is connected. Next, we establish necessary conditions for determinacy in the CQ_{cgraph}-to-CQ_{cgraph} case. We also show that CQ_{chain}-to-CQ_{cgraph} determinacy is decidable, extending the previous decidability result for CQ_{chain}-to-CQ_{chain}, where CQ_{chain} denotes the class of binary CQ queries whose body is a simple path between the two free variables. Finally, we provide an algorithm which, starting with a set of CQ_{cgraph} views $\mathbf{V}$ and an integer k, generates a set of CQ_{cgraph} queries that are determined by $\mathbf{V}$ and have their size bounded by k.

Categories and Subject Descriptors

H.2 [**Database Management**]: Languages—*Query languages*

General Terms

Theory

Keywords

conjunctive queries, first order logic

ICDT 2011, March 21–23, 2011, Uppsala, Sweden.

1. INTRODUCTION

The problem of whether a set of queries on a database provides enough information for answering another query has received considerable attention in different contexts related to data management. This has been a central topic in data integration [5, 10, 17] where the problem is formulated in terms of rewriting a specific query using a given set of queries, called views. The same problem is encountered in semantic caching [8], when answers to a given set of queries are stored in a cached data container. When a new query arises, it must be determined whether the query can be answered from the cache, or the complete database needs to be interrogated. Security and privacy provide another context for the same problem [16]. Assuming that there exists a set of queries which is considered to be private information, and that some data from a database is made accessible through a set of views, it must be verified that the views do not provide enough information for answering the queries.

Whatever the context is, a fundamental question is how to formalize the fact that a set of views $\mathbf{V}$ contains enough information for answering a specific query Q. We say that $\mathbf{V}$ *determines* Q, denoted $\mathbf{V} \twoheadrightarrow Q$, iff for all database instances D_1 and D_2, $\mathbf{V}(D_1) = \mathbf{V}(D_2)$ implies $Q(D_1) = Q(D_2)$ [14]. Indeed, $\mathbf{V} \twoheadrightarrow Q$ implies that for any database instance D, if $V(D)$ is provided, $Q(D)$ can be computed by finding any database instance D' with $V(D') = V(D)$, and computing $Q(D')$. We must note that the above definition holds both in the context of unrestricted (finite or infinite) or restricted (finite) database instances. However, in this paper we only consider the finite case.

In the case that a set of views determines a query, we are also interested to know if there exists an equivalent rewriting of the query using the views, and the language in which this rewriting can be done. However, the definition of determinacy does not say anything about how the query could be rewritten using the views. The existence of such a rewriting depends on the languages of the query and of the views. Thus, having a query language $\mathcal{Q}$ and a view language $\mathcal{V}$, we say that a language $\mathcal{R}$ is complete for $\mathcal{V}$-to-$\mathcal{Q}$ rewritings iff for any query $Q \in \mathcal{Q}$ and any set of views $\mathbf{V} \in \mathcal{V}$, $\mathbf{V} \twoheadrightarrow Q$ implies that there exists a rewriting $Q_{\mathbf{V}} \in \mathcal{R}$ such that for any database instance D, $Q(D) = Q_{\mathbf{V}}(\mathbf{V}(D))$.

In this paper we consider subclasses of conjunctive queries (CQ for short) as query and view languages, and study the decidability of determinacy and the existence of equivalent

rewritings. The decidability of CQ-to-CQ determinacy and the completeness of first order logic (FO) for CQ-to-CQ rewritings is currently an open problem. We define a subclass of conjunctive queries, CQ_{cgraph}, which contains all binary CQ queries whose body, if viewed as an undirected graph, is connected. Our contributions are the following:

1. We identify necessary conditions for the CQ_{cgraph}-to-CQ_{cgraph} determinacy in the form of structural properties of the graph $\mathbf{V}([Q])$, where $[Q]$ is the database instance induced by the body of Q. This way we know that if the properties do not hold, then the views do not determine the query. We also show that our conditions are not sufficient for determinacy, by providing an appropriate example.
2. We prove that CQ_{chain}-to-CQ_{cgraph} determinacy is decidable and that FO is complete for CQ_{chain}-to-CQ_{cgraph} rewritings, extending the previous result for the CQ_{chain}-to-CQ_{chain} case [2]. CQ_{chain} queries are defined as binary CQ queries where the body, if viewed as an undirected graph, is a chain between the two arguments of the query.
3. Having a CQ_{cgraph} query $Q(x, y)$, we denote by $size(Q)$ the number of nodes in the graph $[Q]$. We say that Q is minimal if there is no homomorphism h from $[Q]$ to a strict subgraph of $[Q]$ with $h(x) = x$ and $h(y) = y$. Having a set $\mathbf{V}$ of CQ_{cgraph} views and an integer k, we provide an algorithm which generates a set of minimal CQ_{cgraph} queries $\mathbf{Q}_{\mathbf{V},k}$ such that each $Q \in \mathbf{Q}_{\mathbf{V},k}$ satisfies $\mathbf{V} \twoheadrightarrow Q$ and $size(Q) \leq k$. Moreover, the algorithm also provides FO rewritings for all queries in $\mathbf{Q}_{\mathbf{V},k}$. We also give an example of a CQ_{cgraph} query Q and a set of CQ_{cgraph} views $\mathbf{V}$ such that $\mathbf{V} \twoheadrightarrow Q$ and $Q \notin \mathbf{Q}_{\mathbf{V},size(Q)}$, which shows that the algorithm is not complete in the sense that not all determined queries are generated.

Although we do not prove this, we believe that the above results for CQ_{cgraph} can be extended to CQ as well.

1.1 Related work

The problem of answering queries using views appears in several contexts with different assumptions on views and rewritings. In this paper we study only the case of equivalent query rewritings with *exact* view definitions, meaning that the views contain exactly the set of tuples given by their definitions, and that only equivalent rewritings are of interest. Other settings which are studied in the literature include:

1. Sound views: here the views provide only a subset of tuples from the answer corresponding to their definitions.
2. Maximally contained rewritings: the language in which the rewritings are done is fixed, thus a rewriting which contains only a subset of tuples in the answer is required instead of an equivalent one.

In [14] the problem of language completeness and determinacy is studied extensively for query and view languages ranging from FO to CQ, in the case of equivalent rewritings with exact views. It is easy to prove that if satisfiability of sentences in the query language $\mathcal{Q}$ is undecidable, then determinacy is also undecidable. The same holds if the validity of sentences in the view language $\mathcal{V}$ is undecidable. This directly implies that FO-to-FO determinacy is undecidable [14]. As a consequence of Craig's Interpolation Theorem [6], it follows that FO is complete for FO-to-FO rewritings in the unrestricted case. However, this result does not extend to the finite case, where the FO-to-FO query answering problem is Turing complete. In [14] it is also shown that every language complete for $\exists FO$-to-FO rewritings must be powerful enough to express all queries in $\exists SO \cap \forall SO$, leading to a $NP \cap co - NP$ lower bound.

Despite the fact that UCQ (unions of conjunctive queries) is a much weaker language than FO, UCQ-to-UCQ determinacy remains undecidable and any complete language for UCQ-to-CQ rewritings must be able to express non-monotonic queries [14]. This implies that UCQ is not complete for UCQ-to-UCQ rewritings, nor other more powerful languages such as Datalog. Moreover, this result holds even if the database relations, views, and query are unary.

The case of CQ is particularly important in the context of answering queries using views. Despite the fact that it has been extensively studied, it remains open whether CQ-to-CQ determinacy is decidable and whether FO is complete for CQ-to-CQ rewritings. In [14, 2] examples show that CQ is not complete for CQ-to-CQ rewritings. Recently, subclasses of CQ where determinacy is decidable were identified. In [14] it is shown that determinacy is decidable and that CQ is complete for rewritings in the case of unary views and when the view set contains only one path. In [2] it is shown that determinacy is decidable and that FO is complete for the case where both views and query are paths. In [13] decidability of determinacy and completeness for rewritings are shown for PF (packed fragment), a restriction of first-order logic.

There is also a large amount of work regarding aspects of equivalent query rewritings using exact views, especially for CQs and $UCQs$. In [12] it is shown that it is $NP-complete$ to decide whether an $(U)CQ$ query has an equivalent $(U)CQ$ rewriting using $(U)CQ$ views, some polynomial-time subcases being identified for CQ in [7]. Binding patterns [15], arithmetic comparisons [3] and recursive queries [9] are also studied for CQ queries and views.

2. PRELIMINARIES

2.1 Basic definitions and notations

A database schema σ is a finite set of relational symbols. Each relational symbol $R \in \sigma$ has associated a nonnegative arity $ar(R)$. Assuming an infinite domain dom, a database instance D over σ defines, for all $R \in \sigma$, a finite relation $D(R)$ of arity $ar(R)$ using elements from dom. The active domain of an instance D, denoted by $adom(D)$, is the set of elements from dom which are used in D. A database instance can also be seen as a relational structure with the universe $adom(D)$. We denote by $I(\sigma)$ the set of all instances over σ. If σ contains only binary relations, then each instance $D \in I(\sigma)$ can be seen as a labeled graph. The set of nodes of the graph is $adom(D)$ and the set of labeled edges $E(D)$ is given

by all tuples (x_1, x_2, R) where $(x_1, x_2) \in D(R)$, meaning that there exists an edge between x_1 and x_2 in the graph D labeled with the relation R.

Let σ and $\sigma_{\mathbf{V}}$ be database schemas. A query is defined as a computable mapping from instances of an input schema to instances of an output schema that commute with isomorphisms of the domain. A view $\mathbf{V}$ from $I(\sigma)$ to $I(\sigma_{\mathbf{V}})$ is a set of queries from $I(\sigma)$ to $I(\{V\})$, for each $V \in \sigma_{\mathbf{V}}$. Let Q be a query over σ. Then we say that $\mathbf{V}$ determines Q, denoted $\mathbf{V} \twoheadrightarrow Q$, iff for each $D_1, D_2 \in I(\sigma), \mathbf{V}(D_1) = \mathbf{V}(D_2)$ implies $Q(D_1) = Q(D_2)$. Let R be a query over $\sigma_{\mathbf{V}}$. Then, R is an equivalent rewriting of Q using $\mathbf{V}$, denoted by $Q \Rightarrow_{\mathbf{V}} R$, iff for all database instances $D \in I(\sigma)$, $Q(D) = R(\mathbf{V}(D))$. Let $\mathcal{Q}$ be a query language and $\mathcal{V}$ be a view language. We call $\mathcal{R}$ complete for $\mathcal{V}$-to-$\mathcal{Q}$ rewritings iff for any $Q \in \mathcal{Q}$ and $V \in \mathcal{V}, V \twoheadrightarrow Q$ implies that there exists $R \in \mathcal{R}$ such that $Q \Rightarrow_{\mathbf{V}} R$.

2.2 Query and view languages

In this paper consider the following subsets of conjunctive queries, which are defined as follows.

DEFINITION 2.1 (CQ). *Let Q be a database query over σ. Then, Q is a conjunctive query (CQ) if it is a conjunction of the form $\exists \bar{y}(R_1(\bar{x_1}), \cdots, R_k(\bar{x_k}))$, where $\bar{y} \subseteq (\bar{x_1} \cup \cdots \cup \bar{x_k})$. The variables in $\bar{y}$ are called undistinguished, while the others are called free. If $\bar{z}$ is the set of free variables, we define $vars(Q) = \bar{y} \cup \bar{z}$ to be the set of variables in Q. Also, let $[Q] \in I(\sigma)$ denote the database instance induced by the body of Q, where $(x_1, \cdots, x_k) \in [Q](R)$ iff $R(x_1, \cdots, x_k)$ is an atom in Q.*

We can rewrite Q in a more convenient manner as

$$Q(\bar{z}) \leftarrow R_1(\bar{x_1}), \cdots, R_k(\bar{x_k}),$$

where $\bar{z}$ is called the *head* of the query, and the right part is called the *body*. As explained before, if all the relations $R \in \sigma$ are binary, $[Q]$ can be seen as a labeled graph.

Next we define chain queries [2] and connected graph queries as subsets of CQs.

DEFINITION 2.2 (CQ_{chain}). *A CQ query is called chain query if it has exactly two free variables and if it is defined over binary atoms which, when viewed as a labeled graph, form a simple path from one argument of the query to the other. The language of all chain queries is denoted as CQ_{chain}.*

DEFINITION 2.3 (CQ_{cgraph}). *A CQ query is called a connected graph query if it is defined over binary atoms which, when viewed as an undirected graph, is connected. Moreover, the query is restricted to have two free variables. The language of all connected graph queries is denoted as CQ_{cgraph}.*

EXAMPLE 2.1. *The database query given by $\phi_1(x, y) \leftarrow R_1(x, z_1), R_2(z_1, z_2), R_1(z_2, y)$ is a chain query, since the body of the query is a simple path from x to y. The query $\phi_2(x, y) \leftarrow R(x, z_1), R(z_2, y)$ is not a chain query, since there exists no path between x and y, and neither is $\phi_3(x, y) \leftarrow R_1(x, z_1), R_2(z_1, z_2), R_1(z_2, z_3), R_2(z_3, z_1), R_1(z_1, y)$, because the body of the query is not a simple path between x and y. We also have $\phi_1, \phi_3 \in CQ_{cgraph}$ and $\phi_2 \notin CQ_{cgraph}$.*

DEFINITION 2.4 (MINIMALITY). *Let $Q(x, y) \in CQ_{cgraph}$ be a query. Q is called minimal iff there is no homomorphism $h : [Q] \to [Q']$ with $h(x) = x$ and $h(y) = y$, for some Q' such that $[Q']$ is a strict subgraph of $[Q]$.*

In other words, a CQ_{cgraph} query Q is minimal if we cannot obtain an equivalent query by removing atoms from the body of Q. We can assume that all the queries we are working with are minimal, since for any CQ_{cgraph} query Q, a minimal query equivalent to Q always exists.

Finally, we define the weakly connected components of an instance defined over a schema which contains only binary relations. This notion is used in some of the proofs later in the paper.

DEFINITION 2.5 (WEAK CONNECTIVITY). *Let $D \in I(\sigma)$ be an instance over a database schema σ. A pair of nodes $(u, v) \in adom(D)$ is called weakly connected if there exists an undirected path in D from u to v. A weakly connected component of D is a maximal subgraph in which any pair of nodes is weakly connected.*

3. CQ_{CGRAPH}-TO-CQ_{CGRAPH} DETERMINACY CONDITIONS

As it was previously shown in [2], connectivity inside the body of the query plays an important role in determinacy. The connectivity graph of a query is defined as the graph which contains a node for each atom and an edge between two atoms which share a variable. Indeed, if the connectivity graph of a CQ query Q has a weakly connected component S which does not contain any tuple in $\mathbf{V}([Q])$, then a set of views $\mathbf{V}$ determines Q only if there is a view $V \in \mathbf{V}$ whose connectivity graph contains a connected component isomorphic to S. In this section we analyze the CQ_{cgraph}-to-CQ_{cgraph} case and we show that connectivity properties of the graph $\mathbf{V}([Q])$ also play an important role in determinacy. Let σ be a database schema and $Q(x, y)$ a CQ_{cgraph} query over σ. First, we show that if a CQ_{cgraph} view $\mathbf{V}$ determines Q, then there exists an undirected path between x and y in $\mathbf{V}([Q])$. Then, we show how to compute partial results to a query more efficiently when $\mathbf{V}([Q])$ is divided into smaller weakly connected components. Finally we state our second necessary condition for determinacy: if $\mathbf{V}$ determines Q then the removal of any edge from Q must reflect in the weakly connected component of x.

We start with an example that illustrates the fact that if x and y are not weakly connected in $\mathbf{V}([Q])$, then the view does not determine the query.

EXAMPLE 3.1. *Let $Q(x, y) \leftarrow a(x, z_1), b(z_1, z_2), c(z_2, z_3), d(z_3, z_1), e(z_1, y)$ be a binary database query over the schema*

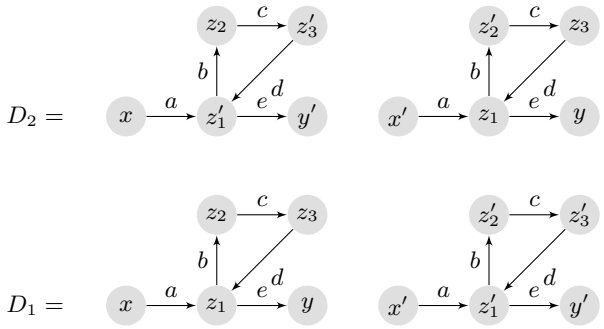

Figure 1: The two instances D_1 and D_2 with $\mathbf{V}(D_1) = \mathbf{V}(D_2)$ and $Q(D_1) \neq Q(D_2)$ for the views and query shown in Example 3.1.

$\sigma = \{a, b, c, d, e\}$. Then, consider the view $\mathbf{V} = \{V_1, V_2\}$ with $V_1(x, y) \leftarrow a(x, z_1), b(z_1, y)$ and $V_2(x, y) \leftarrow d(x, z_1), e(z_1, y)$. Thus, we have that $\mathbf{V}([Q]) = \{V_1(x, z_2), V_2(z_3, y)\}$. Notice that x and y are not in the same weakly connected component which, using theorem 3.1 shown below, implies that $\mathbf{V}$ does not determine Q.

Next we will construct the two instances D_1 and D_2 with $\mathbf{V}(D_1) = \mathbf{V}(D_2)$ and $Q(D_1) \neq Q(D_2)$, obtained as described in the proof of Theorem 3.1. We have $D_1 = D_1^1 \cup D_1^2$, where $D_1^1 = \{a(x, z_1), b(z_1, z_2), c(z_2, z_3), d(z_3, z_1), e(z_1, y)\}$ and $D_1^2 = \{a(x', z_1'), b(z_1', z_2'), c(z_2', z_3'), d(z_3', z_1'), e(z_1', y')\}$. Thus, it directly follows that $Q(D_1) = \{(x, y), (x', y')\}$. Following the construction in the proof, we have $D_2 = D_2^1 \cup D_2^2$, where $D_2^1 = \{a(x, z_1'), b(z_1', z_2), c(z_2, z_3'), d(z_3', z_1'), e(z_1', y')\}$ and $D_2^2 = \{a(x', z_1), b(z_1, z_2'), c(z_2', z_3), d(z_3, z_1), e(z_1, y)\}$. It follows that $Q(D_2) = \{(x, y'), (x', y)\}$, thus $Q(D_1) \neq Q(D_2)$. It can also be seen that $V(D_1) = V(D_2)$.

We are now ready to give the following theorem.

THEOREM 3.1. *Let σ and $\sigma_{\mathbf{V}}$ be database schemas which contain only binary relations. Let $Q(x, y) \in CQ_{cgraph}$ be a query over σ and $\mathbf{V}$ be a CQ_{cgraph} view from $I(\sigma)$ to $I(\sigma_{\mathbf{V}})$. Then $\mathbf{V} \twoheadrightarrow Q$ implies that x and y are weakly connected in $\mathbf{V}([Q])$.*

PROOF. Suppose by contradiction that x and y are not weakly connected in $\mathbf{V}([Q])$. We construct two database instances D_1 and D_2 such that $\mathbf{V}(D_1) = \mathbf{V}(D_2)$ and $Q(D_1) \neq Q(D_2)$, which contradicts $\mathbf{V} \twoheadrightarrow Q$. For any element $u \in adom([Q])$ we associate a fresh element u'. For the rest of the proof, let S_x be the set of nodes in the weakly connected component of x in $\mathbf{V}([Q])$ and let $S_x' = \{u' \mid u \in S_x\}$.

First, we construct D_1 with the domain $adom(D_1) = \{u, u' \mid u \in adom([Q])\}$. For each $R \in \sigma$ we define

$$D_1(R) = \{(u, v), (u', v') \mid (u, v) \in [Q](R)\}.$$

Therefore, D_1 basically consists of two disjoint subgraphs isomorphic to $[Q]$. From this it follows that $\{(x, y), (x', y')\} \subseteq Q(D_1)$ and that $\{(x, y'), (x', y)\} \cap Q(D_1) = \emptyset$.

For constructing D_2, let $adom(D_2) = adom(D_1)$. Then, let $D_2(R) = \{(u, v), (u', v') \mid (u, v) \in [Q](R), |\{u, v\} \cap S_x| \neq 1\} \cup \{(u, v'), (u', v) \mid (u, v) \in [Q](R), |\{u, v\} \cap S_x| = 1\}$, for all $R \in \sigma$. The key thing to notice here is that D_2 still contains two disjoint subgraphs isomorphic to $[Q]$. We will show that $D_2 = D_2^1 \cup D_2^2$, where D_2^1 and D_2^2 are disjoint subgraphs with the following properties:

1. $adom(D_2^1) = (adom([Q]) - S_x) \cup S_x'$ and $adom(D_2^2) = S_x \cup \{u' \mid u \in (adom([Q]) - S_x)\}$,
2. the mappings $h : [Q] \to D_2^1$ and $h' : [Q] \to D_2^2$ with $h(u) = u', h'(u) = u$ for all $u \in S_x$ and $h(v) = v, h'(v) = v'$, for all $v \notin S_x$ are isomorphisms.

Since $x \in S_x$ and $y \notin S_x$ it follows that $h(x) = x'$ and $h(y) = y$. Thus, we have that $(x', y) \in Q(D_2)$ which implies $Q(D_1) \neq Q(D_2)$. Finally, the existence of both h and h' with the fact that D_1^1 and D_1^2 are disjoint concludes that $\mathbf{V}(D_1) = \mathbf{V}(D_2)$.

Let D_2^1 and D_2^2 with domains $adom(D_2^1) = (adom([Q]) - S_x) \cup S_x'$ and $adom(D_2^2) = S_x \cup \{u' \mid u \in (adom([Q]) - S_x)\}$ be the subgraphs of D_2 where the labeled edges sets $E(D_2^1)$ and $E(D_2^2)$ are restrictions of $E(D_2)$ for $adom(D_2^1)$ and $adom(D_2^2)$. Using the definition of D_2, it follows that D_2^1 and D_2^2 are indeed two disjoint subgraphs such that $D_2 = D_2^1 \cup D_2^2$. It remains now to show that the mappings h and h' shown above are indeed isomorphisms. We will only show it for h, since for h' it is analogous.

First, we will show that h is a homomorphism from $[Q]$ to D_2^1. For that, consider any $R \in \sigma$ and any pair $(u, v) \in [Q](R)$. Then, we distinguish the following cases:

- if $u \in S_x$ and $v \in S_x$ then $h(u) = u', h(v) = v'$ and we have that $(u', v') \in D_2^1(R)$ from the definitions of D_2 and D_2^1,
- if $u \notin S_x$ and $v \notin S_x$ then $h(u) = u, h(v) = v$, which concludes,
- if $u \in S_x$ and $v \notin S_x$ then $h(u) = u', h(v) = v$ and we have that $(u', v) \in D_2^1(R)$ from the definitions of D_2 and D_2^1,
- if $u \notin S_x$ and $v \in S_x$ then we proceed analogous to the previous case.

For showing that h^{-1} is a homomorphism from D_2^1 to $[Q]$, consider any $R \in \sigma$ and any pair $(p, q) \in D_2^1(R)$. Then, we distinguish the following cases:

- if $(p, q) = (u', v')$, where $u', v' \in S_x'$ then we have that $h^{-1}(u') = u$ and $h^{-1}(v') = v$; since $(u', v') \in D_2(R)$, the definition of D_2 implies that $(u, v) \in [Q](R)$,
- if $(p, q) = (u, v)$, where $u, v \in (adom([Q]) - S_x)$, then we have that $h^{-1}(u) = u$ and $h^{-1}(v) = v$, which concludes,

- if $(p,q) = (u',v)$, where $u' \in S'_x, v \in (adom([Q]) - S_x)$ then we have that $h^{-1}(u') = u$ and $h^{-1}(v) = v$; since $(u',v) \in D_2(R)$, the definition of D_2 implies that $(u,v) \in [Q](R)$,
- if $(p,q) = (u,v')$, where $u \in (adom([Q]) - S_x), v' \in S'_x$ then we proceed analogous to the previous case.

□

Example 3.1 illustrates the constructions described in the proof of theorem 3.1, in the case that the views do not determine the query. The following lemma follows as a direct implication of the above theorem.

LEMMA 3.1. *Let σ and $\sigma_{\mathbf{V}}$ be schemas which contain only binary relations and let $D \in I(\sigma)$ be an instance. Let $Q(x,y) \in CQ_{cgraph}$ be a query over σ and $\mathbf{V}$ be a view from $I(\sigma)$ to $I(\sigma_{\mathbf{V}})$ such that $\mathbf{V} \twoheadrightarrow Q$. Then, for every pair $(u,v) \in Q(D)$, u and v are weakly connected in $\mathbf{V}(D)$.*

PROOF. From $(u,v) \in Q(D)$ we have that there exists a homomorphism h from $[Q]$ to D with $h(x) = u$ and $h(y) = v$. This implies that we also have a homomorphism h' from $\mathbf{V}([Q])$ to $\mathbf{V}(D)$ with $h'(x) = u$ and $h'(y) = v$. The conclusion follows now from theorem 3.1. □

The following lemma relates isomorphic weakly connected components of views of different database instances, by showing that they should have similar impact on the results of the query.

LEMMA 3.2. *Let σ and $\sigma_{\mathbf{V}}$ be database schemas which contain only binary relations and let $P, P' \in I(\sigma)$ be database instances. Let $Q(x,y) \in CQ_{cgraph}$ be a query over σ and $\mathbf{V}$ be a view from $I(\sigma)$ to $I(\sigma_{\mathbf{V}})$ such that $\mathbf{V} \twoheadrightarrow Q$. Let S be a weakly connected component of $\mathbf{V}(P)$ and S' be a weakly connected component of $\mathbf{V}(P')$ isomorphic to S. Then, for any $u, v \in S$ and any isomorphism h from S to S' we have that $(u,v) \in Q(P)$ iff $(h(u), h(v)) \in Q(P')$.*

PROOF. The first thing to note is that it suffices to prove just one direction of the equivalence: indeed, assuming that $(u,v) \in Q(P)$ implies $(h(u), h(v)) \in Q(P')$, the other direction follows analogous.

Let $(u,v) \in S$ such that $(u,v) \in Q(P)$. Suppose by contradiction that $(h(u), h(v)) \notin Q(P')$. Then, we will provide two instances D_1 and D_2 such that $V(D_1) = V(D_2)$ and $Q(D_1) \neq Q(D_2)$, which contradicts $\mathbf{V} \twoheadrightarrow Q$. As follows we show how D_1 and D_2 are constructed.

Suppose, wlog, that $adom(P) \cap adom(P') = \emptyset$. For constructing D_1, let $adom(D_1) = adom(P) \cup adom(P')$ and, for all $R \in \sigma$, let $D_1(R) = P(R) \cup P'(R)$. Thus, D_1 contains two isomorphic copies of P. Using $(u,v) \in Q(P)$, it follows that $(u,v) \in Q(D_1)$.

For the second instance consider $adom(D_2) = adom(P) \cup adom(P')$. Then, for all $R \in \sigma$ let $D_2(R)$ be the reunion of the following sets:

1. $\{(p,q) \mid (p,q) \in P(R), |\{p,q\} \cap S| \neq 1\}$ and $\{(p,q) \mid (p,q) \in P'(R), |\{p,q\} \cap S'| \neq 1\}$,
2. $\{(h(p),q) \mid (p,q) \in P(R), \{p,q\} \cap S = \{p\}\}$ and symmetrically $\{(p,h(q)) | (p,q) \in P(R), \{p,q\} \cap S = \{q\}\}$,
3. $\{(h^{-1}(p),q) \mid (p,q) \in P'(R), \{p,q\} \cap S' = \{p\}\}$ and symmetrically $\{(p,h^{-1}(q)) | (p,q) \in P'(R), \{p,q\} \cap S' = \{q\}\}$.

As in the proof of theorem 3.1, we have that $D_2 = D_2^1 \cup D_2^2$, where D_2^1 and D_2^2 are disjoint subgraphs with the following properties:

- $adom(D_1) = S' \cup (adom(P) - adom(S))$, $adom(D_2) = S \cup (adom(P') - adom(S'))$,
- the mapping $f : P \to D_2^1$ with $f(p) = p$, for all $p \in adom(P) - adom(S)$ and $f(p) = h(p)$ for all $p \in adom(S)$ is an isomorphism,
- the mapping $f' : P' \to D_2^2$ with $f(p) = p$, for all $p \in adom(P') - adom(S')$ and $f(p) = h^{-1}(p)$ for all $p \in adom(S')$ is an isomorphism.

Since $(h(u), h(v)) \notin Q(P')$ it follows $(f'(h(u)), f'(h(v))) \notin Q(D_2)$, using the fact that f' is an isomorphism. Since $f'(h(u)) = u$ and $f'(h(v)) = v$, we have $(u,v) \notin Q(D_2)$. This implies that $Q(D_1) \neq Q(D_2)$. Finally, the existence of both f and f', with the fact that D_1^1 and D_1^2 are disjoint concludes that $\mathbf{V}(D_1) = \mathbf{V}(D_2)$. The fact that D_2^1 and D_2^2 indeed exist follows in the same lines as shown in the proof of theorem 3.1. □

We note that we can use lemmas 3.1 and 3.2 for computing partial query results more efficiently in the case where $\mathbf{V}([Q])$ is divided into smaller weakly connected components. Let D be a database instance over σ and $u \in adom(D)$ be a fixed element from the domain of D. Suppose that, taking $\mathbf{V}(D)$ as input, we are interested in computing the set $R = \{v \mid (u,v) \in Q(D), v \in adom(D)\}$. Let S_u denote the weakly connected component of u in $\mathbf{V}(D)$. Let D' be a database instance where $\mathbf{V}(D')$ has a weakly connected component S' such that there exists an isomorphism $f : S_u \to S'$. Let $R' = \{v \mid (f(u), v) \in Q(D'), v \in adom(S')\}$. From lemma 3.1 we have that for all $v \in R$, v is weakly connected to u, thus $R \subseteq adom(S_u)$. Then, using lemma 3.2, it follows that $R = f^{-1}(R')$. In the case that S_u is significantly smaller than $\mathbf{V}(D)$, computing D' is easier than guessing a database instance D'' with $\mathbf{V}(D'') = \mathbf{V}(D)$ and computing $Q(D'') = Q(D)$.

We are now ready to give another necessary condition for determinacy which follows directly from lemma 3.2.

THEOREM 3.2. *Let σ and $\sigma_{\mathbf{V}}$ be database schemas which contain only binary relations. Let $Q(x,y) \in CQ_{cgraph}$ be a minimal query over σ and $\mathbf{V}$ be a view from $I(\sigma)$ to $I(\sigma_{\mathbf{V}})$. Let S_x be the weakly connected component of x in $\mathbf{V}([Q])$. For every labeled edge $e \in E([Q])$ we denote by $Q - e$ the query obtained by removing edge e and by S_x^e the weakly component of x in $[Q - e]$. Then, $\mathcal{V} \twoheadrightarrow Q$ implies that for every $e \in E([Q]), S_x \neq S_x^e$.*

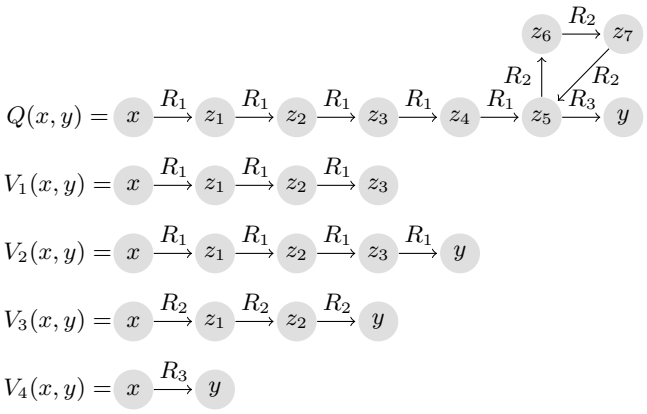

Figure 2: The query and the views used in example 4.1.

PROOF. Suppose there exists an edge e such that $S_x = S_x^e$. Using lemma 3.2, this implies that $(x, y) \in Q([Q - e])$. Thus, there exists a homomorphism from $[Q]$ to $[Q - e]$, which is a contradiction since Q is minimal. □

The following example shows that the necessary conditions from theorems 3.1 and 3.2 are not sufficient for CQ_{cgraph}-to-CQ_{cgraph} determinacy.

EXAMPLE 3.2. *Let $Q(x,y) \leftarrow R(x,z_1), R(z_1,z_2), R(z_2,z_3), R(z_3,z_1), R(z_1,y)$ be a database query over the schema $\sigma = \{R\}$. Then, consider the view $\mathbf{V} = \{V\}$ where $V(x,y) \leftarrow R(x,z_1), R(z_1,z_2), R(z_2,z_3), R(z_3,z_4), R(z_4,y)$ is a database query over σ. We have that $\mathbf{V}$ does not determine Q. However, the determinacy conditions described in theorems 3.1 and 3.2 hold, which shows their insufficiency for determinacy.*

4. DECIDABILITY OF DETERMINACY IN THE CQ_{CHAIN}-TO-CQ_{CGRAPH} CASE

In [2] the following decidability result for CQ_{chain}-to-CQ_{chain} determinacy was given:

THEOREM 4.1. *Let $\mathbf{V}$ be a set of CQ_{chain} views and $Q(x,y)$ be a CQ_{chain} query. Then $\mathbf{V} \twoheadrightarrow Q$ iff x and y are weakly connected in $\mathbf{V}([Q])$. Moreover, FO is complete for CQ_{chain}-to-CQ_{chain} rewritings.*

In this section we will extend the above result by showing that CQ_{chain}-to-CQ_{cgraph} determinacy is decidable as well. We start with an example which illustrates the intuition behind our algorithm.

EXAMPLE 4.1. *Let $\sigma = \{R_1, R_2, R_3\}$ be a database schema and let $Q(x,y) \leftarrow R_1(x,z_1), R_1(z_1,z_2), R_1(z_2,z_3), R_1(z_3,z_4), R_1(z_4,z_5), R_2(z_5,z_6), R_2(z_6,z_7), R_2(z_7,z_5), R_3(z_5,y)$ over σ be a CQ_{cgraph} query, as shown in figure 2. Also, let $\mathbf{V} = \{V_1, V_2, V_3, V_4\}$ be a set of CQ_{chain} views over σ given by*

$$
\begin{aligned}
V_1(x,y) &\leftarrow R_1(x,z_1), R_1(z_1,z_2), R_1(z_2,y),\\
V_2(x,y) &\leftarrow R_1(x,z_1), R_1(z_1,z_2), R_1(z_2,z_3), R_1(z_3,y),\\
V_3(x,y) &\leftarrow R_2(x,z_1), R_2(z_1,z_2), R_2(z_2,y),\\
V_4(x,y) &\leftarrow R_3(x,y).
\end{aligned}
$$

As shown later in lemmas 4.1 and 4.2 we have that if $\mathbf{V}$ determines Q then $\mathbf{V}$ also determines $Q_1(x,y) \leftarrow R_3(x,y)$, $Q_2(x,y) \leftarrow R_2(x,z_1), R_2(z_1,z_2), R_2(z_2,y)$ and $Q_3(x,y) \leftarrow R_1(x,z_1), R_1(z_1,z_2), R_1(z_2,z_3), R_1(z_3,z_4), R_1(z_4,y)$. It could be observed that $[Q_1]$, $[Q_2]$ and $[Q_3]$ are subgraphs of $[Q]$ (up to the renaming of their free variables). After the appropriate renaming of their arguments, they are simple paths or simple cycles in $[Q]$ such that each edge linking a node outside $[Q_i]$ to $[Q_i]$ $(1 \leq i \leq 3)$ has one end in one of the arguments of Q_i. Moreover, $[Q_1]$, $[Q_2]$ and $[Q_3]$ are maximal subgraphs with the above properties. We show later in lemmas 4.1 and 4.2 that if $\mathbf{V}$ determines Q then $\mathbf{V}$ determines all queries that satisfy similar properties.

Using theorem 4.1 we have that $\mathbf{V}$ determines each of the above queries. Indeed, we can rewrite Q_1 as $Q_1(x,y) = \exists z_1(V_2(x,z_1) \wedge \forall z_2(V_1(z_2,z_1) \rightarrow V_2(z_2,y)))$. For the other two we have the trivial rewritings $Q_2(x,y) = V_3(x,y)$ and $Q_3(x,y) = V_4(x,y)$. Finally, it can be seen that the existence of the above rewritings is also sufficient for determinacy, since we have that $Q(x,y) = \exists z_1(Q_1(x,z_1) \wedge Q_2(z_1,z_1) \wedge Q_3(z_1,y))$.

Let $Q(x,y)$ be a CQ_{cgraph} query and $\mathbf{V}$ be a set of CQ_{chain} views. We show that if $\mathbf{V}$ determines Q then $\mathbf{V}$ determines all queries that satisfy similar properties to those shown in the above example. We have also seen that, for the previous example, this was also sufficient. At the end of the section we will prove that this is always the case, which concludes that CQ_{chain}-to-CQ_{cgraph} determinacy is decidable.

As follows, the inner grade of a node $u \in adom([Q])$ is the number of triplets (v,u,R), where $v \in adom([Q]), R \in \sigma$ such that $(v,u) \in [Q](R)$. Conversely, the outer grade of a node u is the number of triplets (u,v,R), where $v \in adom([Q]), R \in \sigma$ such that $(u,v) \in [Q](R)$. We are now ready to give the following lemmas.

LEMMA 4.1. *Let σ and $\sigma_{\mathbf{V}}$ be database schemas which contain only binary relations. Let $Q(x,y) \in CQ_{cgraph}$ be a minimal query over σ and $\mathbf{V} \in CQ_{chain}$ be a view from $I(\sigma)$ to $I(\sigma_{\mathbf{V}})$. Then, $\mathbf{V} \twoheadrightarrow Q$ implies that $\mathbf{V}$ determines all CQ_{chain} queries $Q'(z_0,z_k) \leftarrow R_1(z_0,z_1), \cdots, R_k(z_{k-1},z_k)$ that satisfy the following properties:*

1. *$[Q']$ is a subgraph of $[Q]$*
2. *the nodes $z_1, \cdots, z_{k-1}$ have the inner and outer grades equal to 1 in $[Q]$,*
3. *the nodes z_0 and z_k are either in the head of Q or have the inner or outer grade not equal to 1.*

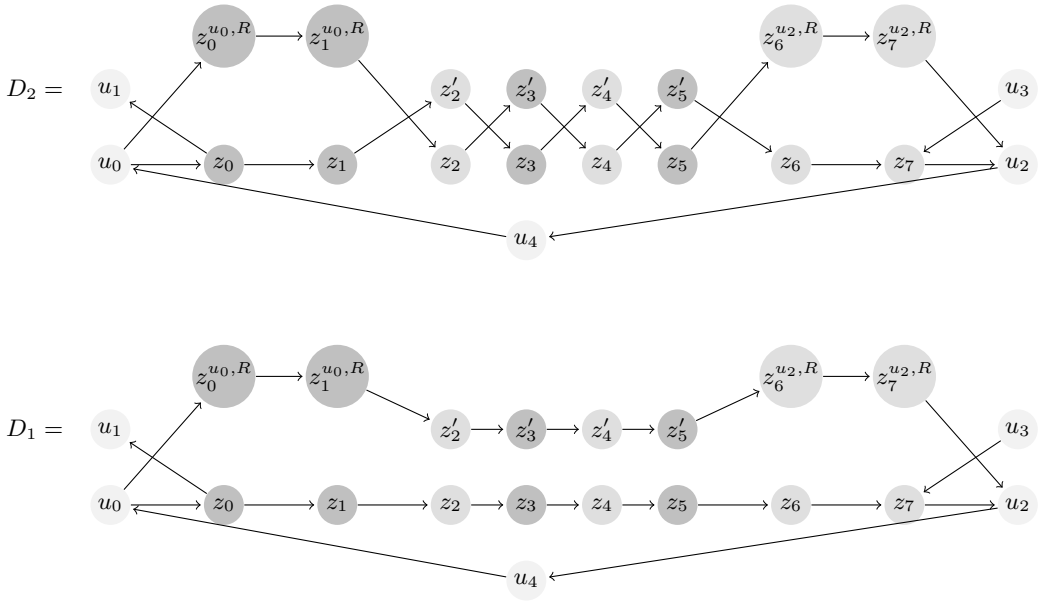

Figure 3: D_1 and D_2 constructed as shown in the proof of lemma 4.1. The general setting is explained in example 4.2

PROOF. Suppose that there exists a query $Q'(z_0, z_k) \leftarrow R_1(z_0, z_1), \cdots, R_k(z_{k-1}, z_k)$ that satisfies the above properties such that $\mathbf{V}$ does not determine Q'. Thus, we have that the chain $R_1(z_0, z_1), \cdots, R_{k-1}(z_{k-2}, z_{k-1}), R_k(z_{k-1}, z_k)$ is a subgraph of $[Q]$ and that the nodes $z_1, \cdots, z_{k-1}$ do not have other incident edges in $[Q]$ except those from the chain. Next we will provide two instances D_1 and D_2, with $V(D_1) = V(D_2)$ and $Q(D_1) \neq Q(D_2)$ which contradict $\mathbf{V} \twoheadrightarrow Q$. For the rest of the proof, let S denote the weakly connected component of z_0 in $V([Q']), i_{min} = \min_{0 \leq i \leq k} z_i \notin S$ and $i_{max} = \max_{0 \leq i \leq k} z_i \in S$.

For the first instance D_1, let $adom(D_1) = adom([Q]) \cup \{z'_i \mid i_{min} \leq i \leq i_{max}\} \cup \{z_i^{u,R} \mid 0 \leq i < i_{min}, R \in \sigma, (u, z_0) \in R([Q])\} \cup \{z_i^{u,R} \mid i_{max} < i \leq k, R \in \sigma, (z_k, u) \in R([Q])\}$. Now let the set of edges of D_1 be

$$
\begin{aligned}
E(D_1) \;=\;& [Q] && (1)\\
\cup\;& \{R(z'_i, z'_{i+1}) \mid R \in \sigma, (z_i, z_{i+1}) \in [Q](R), \\
& \quad i_{min} \leq i < i_{max}\} && (2)\\
\cup\;& \{R(u, z_0^{u,R}), \\
& \quad R_1(z_0^{u,R}, z_1^{u,R}), \ldots, R_{i_{min}}(z_{i_{min}-1}^{u,R}, z'_{i_{min}}) \mid \\
& \quad R \in \sigma, u \in adom([Q]) - \{z_0, \cdots, z_k\}, \\
& \quad (u, z_0) \in [Q](R)\} && (3)\\
\cup\;& \{R_{i_{max}+1}(z'_{i_{max}}, z_{i_{max}+1}^{u,R}), \ldots, R(z_k^{u,R}, u) \mid \\
& \quad R \in \sigma, u \in adom([Q]) - \{z_0, \cdots, z_k\}, \\
& \quad (z_k, u) \in [Q](R)\} && (4)
\end{aligned}
$$

For the second instance, let $adom(D_2) = adom(D_1)$, with the set of edges

$$
\begin{aligned}
E(D_2) \;=\;& \{R(u, v) \mid R \in \sigma, (u, v) \in [Q](R), \\
& \quad u, v \in adom[Q] - \{z_1, \cdots, z_{k-1}\}\} && (5)\\
\cup\;& \{R_{i+1}(z_i, z_{i+1}) \mid |\{z_i, z_{i+1}\} \cap S| \neq 1, \\
& \quad 0 \leq i < k\} && (6)\\
\cup\;& \{R_{i+1}(z'_i, z'_{i+1}) \mid |\{z_i, z_{i+1}\} \cap S| \neq 1, \\
& \quad i_{min} \leq i < i_{max}\} \cup && (7)\\
\cup\;& \{R_{i+1}(z_i, z'_{i+1}) \mid |\{z_i, z_{i+1}\} \cap S| = 1, \\
& \quad 0 \leq i < k\} && (8)\\
\cup\;& \{R_{i+1}(z'_i, z_{i+1}) \mid |\{z_i, z_{i+1}\} \cap S| = 1, \\
& \quad i_{min} \leq i \leq i_{max}\} && (9)\\
\cup\;& \{R(u, z_0^{u,R}), \\
& \quad R_1(z_0^{u,R}, z_1^{u,R}), \cdots, R_{i_{min}}(z_{i_{min}-1}^{u,R}, z_{i_{min}}) \mid \\
& \quad R \in \sigma, u \in adom([Q]) - \{z_0, \cdots, z_k\}, \\
& \quad (u, z_0) \in [Q](R)\} && (10)\\
\cup\;& \{R_{i_{max}+1}(z_{i_{max}}, z_{i_{max}+1}^{u,R}), \cdots, R(z_k^{u,R}, u) \mid \\
& \quad R \in \sigma, u \in adom([Q]) - \{z_0, \cdots, z_k\}, \\
& \quad (z_k, u) \in [Q](R)\} && (11)
\end{aligned}
$$

From equation 1 it follows that $(x, y) \in Q(D_1)$. Next we will show that $(x, y) \notin Q(D_2)$, which implies $Q(D_1) \neq Q(D_2)$. Finally we will sketch the proof for $V(D_1) = V(D_2)$.

First, for every $0 \leq i \leq k$, consider $U_i = \{z_i\} \cup \{z'_i | z'_i \in adom(D_2)\} \cup \{z_i^{u,R} | R \in \sigma, u \in adom([Q]), z_i^{u,R} \in adom(D_2)\}$. Suppose there exists h, a homomorphism from $[Q]$ to D_2 with $h(x) = x$ and $h(y) = y$. Then, we can construct h', a homomorphism from $[Q]$ to $[Q]$ as follows:

$$
h'(u) = \begin{cases} z_i, \ if \ h(u) \in U_i \\ h(u), \ otherwise \end{cases} \quad ,
$$

for all $u \in adom([Q])$. It follows trivially that $h'(x) = x$ and $h'(y) = y$.

As follows, we show that h cannot exist. First, notice that $|h([Q]) \cap U_i| = 1, 0 \leq i \leq k$, since otherwise h' would contradict the minimality assumption on Q. Thus, let $u_i = h([Q]) \cap U_i$. Then, we have that

$$(u_i, u_{i+1}) \in R_{i+1}(D_2), 0 \leq i < k. \quad (12)$$

To see this, suppose there exists $0 \leq i < k$ such that $(u_i, u_{i+1}) \notin R_{i+1}([D_2])$. Then h' would be a homomorphism over $[Q] \rightarrow [Q^e]$, where Q^e is the query obtained from Q by removing the edge $e = R_{i+1}(z_i, z_{i+1})$. Thus, this would contradict the minimality assumption on Q.

We know that $u_0 \in U_0$ and $u_k \in U_k$. As follows, we will show that $u_0 = z_0$ and $u_k = z_k$. We will only show the first equality, since the same arguments apply for the second equality as well. Suppose that $u_0 \neq z_0$. Then, we distinguish two cases for z_0, that follow the statement of the lemma:

- $z_0 = x$: in this case, we have from the definition of h that $h(x) = x$, thus $h(x) = z_0$ which implies $u_0 = z_0$,
- the inner grade or outer grade of z_0 is different than 1 : if the inner grade of z_0 is 0 and the outer grade is 1 then $|U_0| = 1$ which implies $u_0 = z_0$. Otherwise, it follows that the outer grade or the inner grade of z_0 is bigger than 1. Suppose $u_0 \neq z_0$. It follows that the inner and outer grade of u_0 are at most 1, which implies that there exists an edge e such that h' is a homomorphism over $[Q] \rightarrow [Q^e]$, where Q^e is the query obtained from Q by removing edge e. However, this contradicts the minimality assumption on Q.

Thus, we now have that $u_0 = z_0$ and $u_k = z_k$. From 12 it follows that the chain $R_1(z_0, u_1), R_2(u_1, u_2), \cdots, R_k(u_{k-1}, z_k)$ is a subgraph of D_2. Since for all $1 \leq i \leq k-1$ we have that $u_i \in U_i$, it follows from 8 and 9 that $z_k \in S$. Using theorem 4.1, this contradicts the fact that $\mathbf{V}$ does not determine Q'.

Next we sketch the proof for $\mathbf{V}(D_1) = \mathbf{V}(D_2)$. For $\mathbf{V}(D_1) \subseteq \mathbf{V}(D_2)$ consider a CQ_{chain} view $V \in \mathbf{V}$ given by$V(v_0, v_n) \leftarrow R_1(v_0, v_1), \cdots, R_n(v_{n-1}, v_n) \in \mathbf{V}$, and let h be a homomorphism from $[V]$ to D_1. Let $h([V])$ be denoted by $P = R_1(h(v_0), h(v_1)), \cdots, R_k(h(v_{n-1}), h(v_n))$. From the construction of D_2 it follows that there exists a chain subgraph of D_2, denoted P', which is isomorphic to P and starts and ends in the same nodes as P (recall that $adom(D_1) = adom(D_2)$). This implies that there exists a homomorphism h' from $[V]$ to D_2 such that $h(v_0) = h'(v_0)$ and $h(v_n) = h'(v_n)$. The reasoning for $\mathbf{V}(D_2) \subseteq \mathbf{V}(D_1)$ is analogous. □

EXAMPLE 4.2. *In this example we illustrate the construction of D_1 and D_2 as shown in the proof of lemma 4.1. Let*

$$\begin{aligned} Q(u_0, u_2) \leftarrow\ & R(u_0, z_0), R(z_0, u_1), R_1(z_0, z_1), R_2(z_1, z_2), \\ & R_3(z_2, z_3), R_4(z_3, z_4), R_5(z_4, z_5), R_6(z_5, z_6), \\ & R_7(z_6, z_7), R(z_7, u_2), R(u_3, z_7), R(u_2, u_4), \\ & R(u_4, u_0). \end{aligned}$$

be a database query over $\sigma = \{R, R_1, \cdots, R_7\}$ and let the set $\mathbf{V}$ of CQ_{chain} views be

$$\begin{aligned} V_1(x, y) &\leftarrow R_1(x, z_1), R_2(z_1, z_2), R_3(z_2, y), \\ V_2(x, y) &\leftarrow R_2(x, z_1), R_3(z_1, y), \\ V_3(x, y) &\leftarrow R_4(x, z_1), R_5(z_1, y), \\ V_4(x, y) &\leftarrow R_3(x, z_1), R_4(z_1, y), \\ V_5(x, y) &\leftarrow R_5(x, z_1), R_6(z_1, y), \\ V_6(x, y) &\leftarrow R_5(x, z_1), R_6(z_1, z_2), R_7(z_2, y), \\ V_7(x, y) &\leftarrow R(x, y). \end{aligned}$$

Next, consider the CQ_{chain} query

$$\begin{aligned} Q'(z_0, z_7) \leftarrow\ & R_1(z_0, z_1), R_2(z_1, z_2), R_3(z_2, z_3), R_4(z_3, z_4), \\ & R_5(z_4, z_5), R_6(z_5, z_6), R_7(z_6, z_7). \end{aligned}$$

We have that $[Q']$ is a subgraph of $[Q]$. Moreover, the nodes $z_1, \cdots, z_6$ have the inner and outer grades equal to 1 in $[Q]$. On the other hand, z_0 and z_7 have outer, respectively inner grades different then 1. Thus, Q' satisfies the hypothesis of Lemma 4.1. Using Theorem 4.1 it follows that $\mathbf{V}$ does not determine Q' since u and v are not connected by a undirected path in $V([Q'])$. Indeed, we have that

$$\begin{aligned} \mathbf{V}([Q']) = \{&V_1(z_0, z_3), V_2(z_1, z_3), V_3(z_3, z_5), V_4(z_2, z_4), \\ &V_5(z_4, z_6), V_6(z_4, z_7)\}, \end{aligned}$$

with $S = \{z_0, z_1, z_3, z_5\}$, where S is the weakly connected component of z_0 in $V([Q'])$. Using Lemma 4.1, we have that any CQ_{chain} view which determines Q also determines Q', thus $\mathbf{V}$ does not determine Q. The construction of D_1 and D_2 as shown in the proof the lemma can be seen in Figure 3. The fact that $Q(D_1) \neq Q(D_2)$ follows from $(u_0, u_2) \in Q(D_1)$ and $(u_0, u_2) \notin Q(D_2)$. Also, we have that

$$\begin{aligned} \mathbf{V}(D_1) = &V_7(u_0, z_0), V_7(z_0, u_1), V_7(z_7, u_2), V_7(u_3, z_7), \\ &V_7(u_2, u_4), V_7(u_4, u_0), V_7(u_0, z_0^{u_0,R}), \\ &V_7(z_7^{u_2,R}, u_2), V_1(z_0, z_3), V_2(z_1, z_3), \\ &V_3(z_3, z_5), V_4(z_2, z_4), V_5(z_4, z_6), \\ &V_6(z_4, z_7), V_1(z_0^{u_0,R}, z_3'), V_2(z_1^{u_0,R}, z_3'), \\ &V_3(z_3', z_5'), V_4(z_2', z_4'), V_5(z_4', z_6^{u_2,R}), \\ &V_6(z_4', z_7^{u_2,R}), \end{aligned}$$

and $V(D_2) = V(D_1)$. As defined in the proof of lemma 4.1, $U_0 = \{z_0, z_0^{u_0,R}\}, U_1 = \{z_1, z_1^{u_0,R}\}, U_i = \{z_i, z_i'\}$ for $2 \leq i \leq 5$, $U_6 = \{z_6, z_6^{u_2,R}\}$ and $U_7 = \{z_7, z_7^{u_2,R}\}$. All the edges from U_i to $U_{i+1}, 0 \leq i < 7$, are labeled with R_{i+1}, and all the other edges are labeled with R.

LEMMA 4.2. *Let σ and $\sigma_{\mathbf{V}}$ be database schemas which contain only binary relations. Let $Q(x, y) \in CQ_{cgraph}$ be a minimal query over σ and $\mathbf{V} \in CQ_{chain}$ be a view from $I(\sigma)$ to $I(\sigma_{\mathbf{V}})$. Then, $\mathbf{V} \twoheadrightarrow Q$ implies that $\mathbf{V}$ determines all CQ_{chain} queries $Q'(x, y) \leftarrow R_1(x, z_1), \cdots, R_k(z_{k-1}, y)$ over σ that satisfy the following properties:*

1. *there exists $z_0 \in adom([Q])$ such that the simple cycle $[Q'(z_0, z_0)]$ is a subgraph of $[Q]$,*

2. *the nodes $z_1, \cdots, z_{k-1}$ have the inner and outer grades equal to 1 in $[Q]$.*

PROOF SKETCH. The proof follows the same lines as the proof of lemma 4.1. However, the construction presented there needs to be adapted for the cycle case. □

It follows that the necessary conditions expressed in lemmas 4.1 and 4.2 are also sufficient for determinacy. Thus, we give the decidability theorem for CQ_{chain}-to-CQ_{cgraph} determinacy:

THEOREM 4.2. *Having a set of CQ_{chain} views $\mathbf{V}$ and a CQ_{cgraph} query Q it is decidable whether $\mathbf{V} \twoheadrightarrow Q$ in polynomial time, provided that the query is minimal. Moreover, we have that FO is complete for CQ_{chain}-to-CQ_{cgraph} rewritings.*

PROOF SKETCH. The key thing to note here is that we can partition the set of edges of $[Q]$ into simple cycles and simple paths that satisfy the hypothesis of lemmas 4.1 and 4.2. This partition can be obtained in linear time since it requires only a traversal of the graph $[Q]$. Let $\mathbf{P}$ denote the set of CQ_{chain} queries which correspond to each element of the partition. Using lemmas 4.1 and 4.2 we have that if $\mathbf{V}$ determines Q then $\mathbf{V}$ determines all queries in $\mathbf{P}$. Testing if a CQ_{chain} query $P \in \mathbf{P}$ is determined by $\mathbf{V}$, as well as computing an equivalent FO rewriting, can be done using theorem 4.1 in $O(|adom([P])| \cdot S)$, where S denotes the sum of $|adom(V)|$ for each $V \in \mathbf{V}$. Since we have to check determinacy for each $P \in \mathbf{P}$, the total complexity of the algorithm is $O(|adom([Q])| \cdot S + |E([Q])|)$, where $|E([Q])|$ denotes the number of edges in $[Q]$.

If this necessary condition holds, then we have an FO rewriting for each query in $P \in \mathbf{P}$ using $\mathbf{V}$. This suffices for obtaining an equivalent FO rewriting of Q in terms of $\mathbf{V}$. Indeed, the rewriting can be obtained by quantifying existentially the nodes shared by multiple partitions, as shown in example 4.1. □

5. GENERATING QUERIES DETERMINED BY A SET OF CQ_{CGRAPH} VIEWS

In this section we will analyze how queries determined by a set of CQ_{cgraph} views can be computed. Let σ and $\sigma_{\mathbf{V}}$ be database schemas which contain only binary relations and $\mathbf{V} \in CQ_{cgraph}$ be a view from $I(\sigma)$ to $I(\sigma_{\mathbf{V}})$. We will provide an algorithm which, starting from the set of queries in $\mathbf{V}$ and an integer k, computes a set $\mathbf{Q}_{\mathbf{V},k}$ of queries such that all $Q \in \mathbf{Q}_{\mathbf{V},k}$ are minimal CQ_{cgraph} queries that satisfy $\mathbf{V} \twoheadrightarrow Q$ and $size(Q) \leq k$, where $size(Q)$ is the cardinal of $adom([Q])$. Moreover, the algorithm provides an FO rewriting for each Q in terms of $\mathbf{V}$.

For two graphs G_1 and G_2 we define the graph $G_1 - G_2$ to be the graph with the same set of nodes G_1 and the set of edges $E(G_1 - G_2) = E(G_1) - E(G_2)$.

DEFINITION 5.1 (u, v-CENTERED SUBGRAPH). *Let $Q \in CQ_{cgraph}$ be a query and let $u, v \in adom(Q)$. Then, the u, v-centered subgraph of $[Q]$, denoted as $\mathcal{C}_{u,v}(Q)$ is the subgraph of $[Q]$ containing all the nodes p such that there exists an undirected path from u to v in $[Q]$ that contains p and that uses u and v exactly once. The set of edges $E(\mathcal{C}_{u,v}(Q))$ is the natural restriction of $E([Q])$ on the set of nodes of $\mathcal{C}_{u,v}(Q)$.*

DEFINITION 5.2 (u, v-LEFT SUBGRAPH). *Given a query $Q \in CQ_{cgraph}$ and $u, v \in adom(Q)$, the u, v-left centered subgraph of $[Q]$, denoted as $\mathcal{C}^u_{u,v}(Q)$, is the weakly connected component of u in the graph $[Q] - \mathcal{C}_{u,v}(Q)$.*

DEFINITION 5.3 (u, v-RIGHT SUBGRAPH). *Having $Q \in CQ_{cgraph}$ and $u, v \in adom(Q)$, the u, v-right centered subgraph of $[Q]$, denoted as $\mathcal{C}^v_{u,v}(Q)$, is the weakly connected component of v in the graph $[Q] - \mathcal{C}_{u,v}(Q)$.*

From the above definitions it directly follows that for any query $Q \in CQ_{cgraph}$ and for any $u, v \in adom(Q)$ we have that $E([Q]) = E(C_{u,v}(Q)) \cup E(C^u_{u,v}(Q) \cup E(C^v_{u,v}(Q))$, where $E(G)$ denotes the edge set of the graph G.

EXAMPLE 5.1. *Let $\sigma = \{R\}$, where R is a binary relation. Let $Q(x, y) \leftarrow R(x, z_5), R(z_6, z_5), R(z_6, x), R(x, z_1), R(y, z_1), R(y, z_3), R(z_3, z_4), R(z_4, y)$ be a CQ_{cgraph} query over σ. We have $C_{x,y}(Q)$ the subgraph of $[Q]$ having nodes $\{x, z_1, y\}$, with the edges $\{R(x, z_1), R(y, z_1)\}$. We also have $C^x_{x,y}$ as the graph with the nodes $\{x, z_5, z_6\}$ having the set of edges $\{R(x, z_5), R(z_6, z_5), R(z_6, y)\}$. Finally, $C^y_{x,y}$ is the graph with nodes $\{y, z_3, z_4\}$ and edges $\{R(y, z_3), R(z_3, z_4), R(z_4, y)\}$.*

LEMMA 5.1. *Let σ be a database schema. Let $Q_1(x, y)$, $Q_2(x, y)$ and $Q_3(x, y)$ be CQ_{cgraph} queries over σ that satisfy the following properties:*

1. *There exists $z_1 \in adom(Q_1)$ such that x is a node in $C^{z_1}_{z_1,y}(Q_1)$ and there exists an isomorphism h_1 from $C_{x,y}(Q_2)$ to $C_{z_1,y}(Q_1)$, with $h_1(x) = z_1, h_1(y) = y$.*
2. *There exists a homomorphism h'_1 from $C^x_{x,y}(Q_2)$ to $[Q_1]$, with $h'_1(x) = z_1$.*
3. *There exists $z_2 \in adom(Q_3)$ such that y is a node in $C^{z_2}_{x,z_2}(Q_3)$ and there exists an isomorphism h_2 from $C_{x,y}(Q_2)$ to $C_{x,z_2}(Q_3)$, with $h_2(x) = x, h_2(y) = z_2$.*
4. *There exists a homomorphism h'_2 from $C^y_{x,y}(Q_2)$ to $[Q_3]$, with $h'_2(y) = z_2$.*
5. *There exists a homomorphism h_3 from $C^x_{x,y}(Q_3)$ to $[Q_2]$ with $h_3(x) = x$ and a homomorphism h_4 from $C^y_{x,y}(Q_1)$ to $[Q_3]$ with $h_4(y) = z_2$.*

then the query $Q(x, y) = \exists v_2(Q_1(x, v_2) \wedge \exists v_4(Q_2(v_4, v_2)) \wedge \forall v_1((Q_2(v_1, v_2) \rightarrow Q_3(v_1, y))))$ is a CQ_{cgraph} query over σ.

PROOF. We will start the proof by defining a new query $Q' \in CQ_{cgraph}$ over σ. After that we will prove that Q' is indeed equivalent to Q.

Assume, wlog, that $adom([Q_1]) \cap adom([Q_2]) = \{x, y\}$. From items 1 and 3 it follows that there exists an isomorphism h''_2

from $C_{x,z_2}(Q_3)$ to $C_{z_1,y}(Q_1)$ with $h_2''(x) = z_1$ and $h_2''(z_2) = y$. Then, let Q_3' be the CQ_{cgraph} query with $adom([Q_3']) = (adom([Q_3]) - (C_{x,z_2}(Q_3) - \{z_2\})) \cup h_2''(C_{x,z_2}(Q_3) - \{z_2\})$. For any $u \in adom([Q_3'])$, let

$$f(u) = \begin{cases} h_2''^{-1}(u), \ if \ u \in h_2''(C_{x,z_2}(Q_3) - \{z_2\}) \\ u, \ otherwise \end{cases}.$$

Thus, for all $u \in adom([Q_3'])$ we have $f(u) \in adom([Q_3])$. The set of edges $E([Q_3'])$ is obtained as follows: for all $R \in \sigma$ and $u, v \in adom([Q_3'])$, $R(u,v) \in E([Q_3'])$ iff $R(f(u), f(v)) \in E([Q_3])$. Let Q_1' be the query equivalent to Q_1, where the node y is renamed to z_2. Finally, let Q' be the CQ_{cgraph} query with $adom([Q']) = V(C^x_{x,z_2}(Q_1')) \cup V(C_{x,z_2}(Q_1')) \cup V(C^{z_2}_{x,z_2}(Q_3'))$, where $V(G)$ denotes the set of nodes of the graph G. The set of edges $E([Q'])$ is the natural restriction of $E([Q_1']) \cup E([Q_3'])$ on $adom([Q'])$. As follows, we will prove that $Q(x,y)$ is equivalent to $Q'(x,y)$.

Let $D \in I(\sigma)$ be a database instance. First, we will show that $(u,v) \in Q'(D)$ implies $(u,v) \in Q(D)$. Thus, there exists a homomorphism $g : [Q'] \to D$ with $g(x) = u$ and $g(y) = v$. From the construction of Q' we have that there exists $v_2 \in adom([Q'])$ such that $(x, v_2) \in Q_1([Q'])$ and that there exists $v_4 \in adom([Q'])$ such that $(v_4, v_2) \in Q_2([Q'])$. Indeed, this follows using items 1, 2, 3, 4 of the hypothesis of the lemma, by taking $v_2 = z_2$ and $v_4 = z_1$. Thus, using the homomorphism g it follows that there exists $g(v_2) \in adom([D])$ and $g(v_4) \in adom([D])$ such that $(u, g(v_2)) \in Q_1(D)$ and $(g(v_4), g(v_2)) \in Q_2(D)$. It remains now to show that for any v_1, $((Q_2(v_1, g(v_2)) \to Q_3(v_1, y)))$. This follows from items 3 and 5 and the construction of Q'. Indeed, we have that there exists a homomorphism g' from $[Q_3]$ to D with $g'(x) = v_1$ and $g'(y) = y$, which concludes $(v_1, y) \in Q_3(D)$.

It remains to show that $(u,v) \in Q(D)$ implies $(u,v) \in Q'(D)$. Thus, it follows that there exists $v_2 \in adom(D)$ such that $Q_1(u, v_2)$ is satisfied. Let g' be a homomorphism from $[Q_1]$ to D with $g'(x) = u$ and $g'(y) = v_2$. Using item 1 and the fact that $\exists v_4(Q_2(v_4, v_2))$ it follows that $Q_2(g'(z_1), v_2)$ is satisfied, where z_1 is chosen as in item 1. From the fact that $\forall v_1((Q_2(v_1, v_2) \to Q_3(v_1, y)))$ it follows that $Q_3(g'(z_1), v)$ is satisfied. Thus, there exists a homomorphism g'' from $[Q_3]$ to D with $g''(x) = g'(z_1)$ and $g''(y) = v$. The conclusion follows from the fact that we can construct the homomorphism f' from $[Q']$ to D, where

$$f'(p) = \begin{cases} g'(p), \ if \ p \in C^{z_1}_{z_1,z_2}(Q') \\ g''(p), \ otherwise \end{cases},$$

with $f'(x) = u$ and $f'(y) = v$. □

Next we give an example of how the lemma can be applied.

Example 5.2. *Let $\sigma = \{R_1, R_2, R_3, R_4\}$ be a database schema with binary relations. Let $\mathbf{V} = \{V_1, V_2, V_3\}$ be a view over $I(\sigma)$, where V_1, V_2, V_3 are defined as follows:*

$$\begin{aligned} V_1(x,y) &\leftarrow R_1(x,z_1), R_2(z_1,y), \\ V_2(x,y) &\leftarrow R_2(x,y), R_4(y,z_1), \\ V_3(x,y) &\leftarrow R_2(x,z_2), R_3(z_2,y), R_4(z_2,z_3). \end{aligned}$$

Indeed, V_1, V_2 and V_3 satisfy the conditions of lemma 5.1. Thus, we have that $\mathbf{V} \twoheadrightarrow Q$, where

$$Q(x,y) \leftarrow R_1(x,z_1), R_2(z_1,z_2), R_3(z_2,y), R_4(z_2,z_3)$$

is a CQ_{cgraph} query over σ. Indeed, $Q(x,y) = \exists v_2(V_1(x,v_2) \wedge \exists v_4(V_2(v_4,v_2)) \wedge \forall v_1((V_2(v_1,v_2) \to V_3(v_1,y))))$.

The following lemmas are immediate and show other ways of obtaining new queries which can be rewritten in terms of a given set of queries.

Lemma 5.2. *Let σ be a database schema with binary relations, and let $Q(x,y)$ be a CQ_{cgraph} query over σ. Then, $Q'(x,y) = Q(y,x)$ is also a CQ_{cgraph} query over σ.*

Lemma 5.3. *Let σ be a database schema with binary relations, and let $Q_1(x,y)$ and $Q_2(x,y)$ be CQ_{cgraph} queries over σ. Then, the query $Q(x,y) = Q_1(x,y) \wedge Q_2(x,y)$ is a CQ_{cgraph} query.*

Lemma 5.4. *Let σ be a database schema with binary relations and let G be an undirected connected graph with the vertex set $V(G) = \{x, y, v_1, \cdots, v_n\}$. For every $(p,q) \in E(G)$ we assign a CQ_{cgraph} query $\varphi_{p,q}$ over σ where p and q are free variables. Then, the query given by the formula $Q(x,y) = \exists v_1 \cdots \exists v_n(\wedge_{p,q \in E(G)} \varphi_{p,q})$ is a CQ_{cgraph} query.*

Proof Sketch. Suppose, wlog, that for any p_1, q_1, p_2, q_2 such that $(p_1,q_1) \in E(G)$ and $(p_2,q_2) \in E(G)$ with $(p_1,q_1) \neq (p_2,q_2)$ we have $adom(\varphi_{p_1,q_1}) \cap adom(\varphi_{p_2,q_2}) = \{p_1, q_1\} \cap \{p_2, q_2\}$. Then, Q is equivalent to the CQ_{cgraph} query $Q'(x,y)$ where $E[Q'] = \cup_{(u,v) \in E(G)} E([\varphi_{u,v}])$. □

Finally, we give the algorithm which takes as input the view $\mathbf{V}$ and an integer k and outputs a set of queries $\mathbf{Q}_{\mathbf{V},k}$. We will use the following subroutines:

- $combineThree(Q_1, Q_2, Q_3)$: given CQ_{cgraph} queries Q_1, Q_2 and Q_3 returns false if the three queries cannot be combined, or the combined query Q otherwise, as shown in lemma 5.1,
- $invert(Q_1)$: given a CQ_{cgraph} query Q_1, the inverse query is returned, as shown in lemma 5.2,
- $combineTwo(Q_1, Q_2)$: given CQ_{cgraph} queries Q_1 and Q_2, returns the combined query Q as shown in lemma 5.3,
- $combinePoints(G, \varphi)$: given the graph G and the assignation φ, it returns the combined query Q as shown in lemma 5.4.

We assume that the above subroutines return only minimal queries. Thus, a minimization operation must be performed before returning the result of each.

The pseudocode is shown in algorithm 5.1. The termination of the algorithm is ensured by the fact that the number

Algorithm 5.1 Generating queries determined by a set of views $\mathbf{V}$ which have the size smaller than k.

1: $stop \leftarrow false$, $\mathbf{Q}_{\mathbf{V},k} \leftarrow \mathbf{V}$
2: **while** $stop = false$ **do**
3: $stop \leftarrow true$
4: **if** $\exists Q_1, Q_2, Q_3 \in \mathbf{Q}_{\mathbf{V},k}$ such that $combineThree(Q_1, Q_2, Q_3) \neq false$ **then**
5: $Q' \leftarrow combineThree(Q_1, Q_2, Q_3)$
6: **if** $Q' \notin \mathbf{Q}_{\mathbf{V},k} \wedge size(Q') \leq k$ **then**
7: $\mathbf{Q}_{\mathbf{V},k} \leftarrow \mathbf{Q}_{\mathbf{V},k} \cup Q'$, $stop \leftarrow false$
8: **end if**
9: **end if**
10: **if** $\exists Q \in \mathbf{Q}_{\mathbf{V},k}$ where $invert(Q) \notin \mathbf{Q}_{\mathbf{V},k} \wedge size(invert(Q)) \leq k$ **then**
11: $\mathbf{Q}_{\mathbf{V},k} \leftarrow \mathbf{Q}_{\mathbf{V},k} \cup invert(Q)$, $stop \leftarrow false$
12: **end if**
13: **if** $\exists Q_1, Q_2 \in \mathbf{Q}_{\mathbf{V},k}$ where $combineTwo(Q_1, Q_2) \notin \mathbf{Q}_{\mathbf{V},k} \wedge size(combineTwo(Q_1, Q_2)) \leq k$ **then**
14: $\mathbf{Q}_{\mathbf{V},k} \leftarrow \mathbf{Q}_{\mathbf{V},k} \cup combineTwo(Q_1, Q_2)$, $stop \leftarrow false$
15: **end if**
16: **if** $\exists G, \varphi$ where $|V(G)| \leq 2 * k$ and φ is an assignation such that for each $(u, v) \in E(G), \varphi_{u,v} \in \mathbf{Q}_{\mathbf{V},k}$ **then**
17: $Q' \leftarrow combinePoints(G, \varphi)$
18: **if** $Q' \notin \mathbf{Q}_{\mathbf{V},k} \wedge size(Q') \leq k$ **then**
19: $\mathbf{Q}_{\mathbf{V},k} \leftarrow \mathbf{Q}_{\mathbf{V},k} \cup Q'$, $stop \leftarrow false$
20: **end if**
21: **end if**
22: **end while**

of CQ_{cgraph} queries over σ which are smaller than k is finite. We must note that the appartenance test made in lines $6, 10, 13$ and 18 tests if there exists a query $Q' \in \mathbf{Q}_{\mathbf{V},k}$ such that there exists an isomorphism $h : [Q] \rightarrow [Q']$ with $h(x) = x$ and $h(y) = y$, and not for equal query graphs. We also stress the fact that the bound of $2k$ points used in line 16 is indeed optimal, in the sense that the algorithm would not produce more queries with a higher bound. This can be seen by analyzing the query graph $[Q']$ obtained as described in the proof of lemma 5.4. Indeed, the minimization of Q' consists of removing edges from the original graph $[Q']$. Thus, we can impose the condition that for each $u \in V(G)$ there exists $v \in V(G)$ such that the subgraph $[\varphi_{u,v}]$ is not completely removed from $[Q']$ in the minimization process, since otherwise we could obtain an equivalent query by choosing a graph G with fewer nodes. The conclusion now follows from the fact that the minimized query should have at most k nodes.

Example 5.3 shows how the algorithm starts with a set of views and produces a specific query.

EXAMPLE 5.3. *Let $\sigma = \{R_a, R_b, R_c, R_d\}$ be a database schema with binary relations. Let $\mathbf{V} = \{V_1, V_2, V_3\}$ be a view over $I(\sigma)$, where V_1, V_2, V_3 are binary CQ_{cgraph} queries defined as follows:*

$$\begin{aligned} V_1(x,y) &\leftarrow R_a(x,z_1), R_b(z_1,y), R_d(z_1,z_2), \\ V_2(x,y) &\leftarrow R_b(x,y), \\ V_3(x,y) &\leftarrow R_b(x,z_1), R_c(z_1,y), R_d(x,z_2). \end{aligned}$$

Also, consider the CQ_{cgraph} query

$$Q(x,y) \leftarrow R_a(x,z_1), R_b(z_1,z_2), R_d(z_1,z_3), R_c(z_2,y)$$

over the database schema σ. It can be seen that Q can be rewritten in terms of $\mathbf{V}$ as

$$Q(x,y) = \exists v_1(V_3(v_1,y) \wedge \forall v_2(V_2(v_1,v_2) \rightarrow V_3(x,v_2))).$$

Notice that this query is indeed generated by the algorithm by first inverting V_1, V_2 and V_3, then using lemma 5.1 for obtaining a query Q', which again, inverted, leads to Q. Thus, even if V_1, V_2 and V_3 do not directly satisfy the properties of lemma 5.1 the query is generated by the algorithm using the inversion procedure in advance.

Next, we give an example which shows that if item 5 from lemma 5.1 is not satisfied, then the views may not determine the query.

EXAMPLE 5.4. *Let $\sigma = \{R_a, R_b, R_c, R_d\}$ be a database schema with binary relations. Let $\mathbf{V} = \{V_1, V_2, V_3\}$ be a view over $I(\sigma)$, where V_1, V_2, V_3 are binary CQ_{cgraph} queries defined as follows:*

$$\begin{aligned} V_1(x,y) &\leftarrow R_a(x,z_1), R_b(z_1,y), R_d(z_1,z_2), R_d(y,z_3) \\ V_2(x,y) &\leftarrow R_b(x,y), \\ V_3(x,y) &\leftarrow R_b(x,z_1), R_c(z_1,y), R_d(z_1,z_2), R_d(x,z_3). \end{aligned}$$

We have that V_1, V_2 and V_3 satisfy hypothesis 1 to 4 of lemma 5.1, but not 5. Thus, we have the CQ_{cgraph} query

$$\begin{aligned} Q(x,y) \leftarrow\ & R_a(x,z_1), R_b(z_1,z_2), R_d(z_1,z_3), \\ & R_c(z_2,y), R_d(z_2,z_4) \end{aligned}$$

obtained as in the proof of lemma 5.1. However, it can be seen that $\mathbf{V}$ does not determine Q. Indeed, the above algorithm does not generate Q starting from $\mathbf{V}$.

A natural question that rises is the following: given a set $\mathbf{V}$ of CQ_{cgraph} views and a CQ_{cgraph} query Q, does the algorithm generate all the queries determined by $\mathbf{V}$ with the size smaller than Q? It turns out that this is not the case, as shown in the following example.

EXAMPLE 5.5. *Consider the CQ_{cgraph} query*

$$\begin{aligned} Q(x,y) \leftarrow\ & a(x,z_3), b(z_3,z_1), c(z_1,z_2), d(z_2,z_4), e(z_4,y), \\ & f(z_1,z_5), f(z_2,z_6) \end{aligned}$$

over the signature $\sigma = \{a, b, c, d, e, f\}$. Then, consider the CQ_{cgraph} views

$$\begin{aligned} V_1(x,y) &\leftarrow a(x,z_1), b(z_1,y), f(y,z_2), \\ V_2(x,y) &\leftarrow d(x,z_1), e(z_1,y), f(x,z_2), \\ V_3(x,y) &\leftarrow b(x,y), \\ V_4(x,y) &\leftarrow d(x,y), \\ V_5(x,y) &\leftarrow b(x,z_1), c(z_1,z_2), d(z_2,y), f(z_1,z_3), f(z_2,z_4). \end{aligned}$$

We have that $\mathbf{V} \twoheadrightarrow Q$ due to the following rewriting of Q in terms of $\mathbf{V}$:

$$\begin{aligned} Q(x,y) = \ & \exists z_1(V_1(x,z_1) \wedge \exists z_2(V_2(z_2,y) \wedge \\ & \forall z_3 \forall z_4((V_3(z_3,z_1) \wedge V_4(z_2,z_4)) \rightarrow V_5(z_3,z_4)))) \end{aligned}$$

However, the algorithm does not generate Q since Lemma 5.1 *does not handle nested universal quantifiers, as needed in this rewriting.*

6. CONCLUSION

The problem whether CQ-to-CQ determinacy is decidable remains open, several well behaved subclasses being identified in the meantime. Currently there is a common belief that results for CQ_{cgraph} may be naturally extended to CQ, thus a result regarding CQ_{cgraph}-to-CQ_{cgraph} determinacy decidability would be an important milestone in solving the original problem for CQs.

7. ACKNOWLEDGMENTS

Many thanks to Luc Segoufin and Ştefan Ciobâcă for insightful discussions and for carefully reviewing the paper. I am also grateful to Foto Afrati for providing useful comments as a shepherd. Thanks also to the anonymous reviewers for their constructive critiques and suggestions.

8. REFERENCES

[1] S. Abiteboul and O. M. Duschka. Complexity of answering queries using materialized views. In *PODS*, pages 254–263. ACM Press, 1998.

[2] F. N. Afrati. Rewriting conjunctive queries determined by views. In L. Kucera and A. Kucera, editors, *MFCS*, volume 4708 of *Lecture Notes in Computer Science*, pages 78–89. Springer, 2007.

[3] F. N. Afrati, C. Li, and V. Pavlaki. Data exchange in the presence of arithmetic comparisons. In A. Kemper, P. Valduriez, N. Mouaddib, J. Teubner, M. Bouzeghoub, V. Markl, L. Amsaleg, and I. Manolescu, editors, *EDBT*, volume 261 of *ACM International Conference Proceeding Series*, pages 487–498. ACM, 2008.

[4] F. N. Afrati, C. Li, and J. D. Ullman. Generating efficient plans for queries using views. In *SIGMOD Conference*, pages 319–330, 2001.

[5] R. Bayardo, W. Bohrer, R. S. Brice, A. Cichocki, J. Fowler, A. Helal, V. Kashyap, T. Ksiezyk, G. MMartin, M. H. Nodine, M. Rashid, M. Rusinkiewicz, R. Shea, C. Unnikrishnan, A. Unruh, and D. Woelk. Infosleuth: Semantic integration of information in open and dynamic environments (experience paper). In J. Peckham, editor, *SIGMOD Conference*, pages 195–206. ACM Press, 1997.

[6] C. C. Chang and H. J. Keisler. *Model Theory*. North-Holland, 1977.

[7] C. Chekuri and A. Rajaraman. Conjunctive query containment revisited. *Theor. Comput. Sci.*, 239(2):211–229, 2000.

[8] S. Dar, M. J. Franklin, B. T. Jónsson, D. Srivastava, and M. Tan. Semantic data caching and replacement. In T. M. Vijayaraman, A. P. Buchmann, C. Mohan, and N. L. Sarda, editors, *VLDB*, pages 330–341. Morgan Kaufmann, 1996.

[9] O. M. Duschka and M. R. Genesereth. Answering recursive queries using views. In *PODS*, pages 109–116. ACM Press, 1997.

[10] D. Florescu, A. Y. Levy, I. Manolescu, and D. Suciu. Query optimization in the presence of limited access patterns. In A. Delis, C. Faloutsos, and S. Ghandeharizadeh, editors, *SIGMOD Conference*, pages 311–322. ACM Press, 1999.

[11] L. M. Haas, D. Kossmann, E. L. Wimmers, and J. Yang. Optimizing queries across diverse data sources. In M. Jarke, M. J. Carey, K. R. Dittrich, F. H. Lochovsky, P. Loucopoulos, and M. A. Jeusfeld, editors, *VLDB*, pages 276–285. Morgan Kaufmann, 1997.

[12] A. Y. Levy, A. O. Mendelzon, Y. Sagiv, and D. Srivastava. Answering queries using views. In *Proceedings of the Fourteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, May 22-25, 1995, San Jose, California*, pages 95–104. ACM Press, 1995.

[13] M. Marx. Queries determined by views: pack your views. In L. Libkin, editor, *PODS*, pages 23–30. ACM, 2007.

[14] A. Nash, L. Segoufin, and V. Vianu. Determinacy and rewriting of conjunctive queries using views: A progress report. In *ICDT*, pages 59–73, 2007.

[15] A. Rajaraman, Y. Sagiv, and J. D. Ullman. Answering queries using templates with binding patterns. In *PODS*, pages 105–112, 1995.

[16] K. Stoffel and T. Studer. Provable data privacy. In K. V. Andersen, J. K. Debenham, and R. Wagner, editors, *DEXA*, volume 3588 of *Lecture Notes in Computer Science*, pages 324–332. Springer, 2005.

[17] J. D. Ullman. Information integration using logical views. *Theor. Comput. Sci.*, 239(2):189–210, 2000.