View-based Query Processing: On the Relationship between Rewriting, Answering and Losslessness

Diego Calvanese[1], Giuseppe De Giacomo[2], Maurizio Lenzerini[2], and Moshe Y. Vardi[3]

[1] Facoltà di Scienze e Tecnologie Informatiche
Libera Università di Bolzano/Bozen, Italy
calvanese@inf.unibz.it
[2] Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza", Italy
{degiacomo,lenzerini}@dis.uniroma1.it
[3] Department of Computer Science
Rice University, Houston, U.S.A.
vardi@cs.rice.edu

Abstract. As a result of the extensive research in view-based query processing, three notions have been identified as fundamental, namely rewriting, answering, and losslessness. *Answering* amounts to computing the tuples satisfying the query in all databases consistent with the views. *Rewriting* consists in first reformulating the query in terms of the views and then evaluating the rewriting over the view extensions. *Losslessness* holds if we can answer the query by solely relying on the content of the views. While the mutual relationship between these three notions is easy to identify in the case of conjunctive queries, the terrain of notions gets considerably more complicated going beyond such a query class. In this paper, we revisit the notions of answering, rewriting, and losslessness and clarify their relationship in the setting of semistructured databases, and in particular for the basic query class in this setting, i.e., two-way regular path queries. Our first result is a clean explanation of the relationship between answering and rewriting, in which we characterize rewriting as a "linear approximations" of query answering. We show that applying this linear approximation to the constraint-satisfaction framework yields an elegant automata-theoretic approach to query rewriting. As for losslessness, we show that there are indeed two distinct interpretations for this notion, namely with respect to answering, and with respect to rewriting. We also show that the constraint-theoretic approach and the automata-theoretic approach can be combined to give algorithmic characterization of the various facets of losslessness. Finally, we deal with the problem of coping with loss, by considering mechanisms aimed at explaining lossiness to the user.

1 Introduction

View-based query processing is the problem of computing the answer to a query based on a set of views [27, 31, 3]. This problem has recently received much attention in several application areas, such as mobile computing, query optimization, data warehousing, and data integration. A large number of results have been reported in the last years, and several methods have been proposed (see [25] for a recent survey).

As a result of the extensive research in this area, there is proliferation of notions whose relationship to each other is not clear. Fundamentally, there seems to be two basic approaches to view-based query processing. The first approach, originating with [27], is the *query-rewriting* approach, which is based on the idea of first reformulating the query in terms of the views and then evaluating the rewriting over the view extensions. The other approach, originating with [18], is the *query-answering* approach, which takes a more direct route, trying to compute the so-called *certain tuples*, i.e., the tuples satisfying the query in all databases consistent with the views, on the basis of the view definitions and the view extensions. The relationship between the two approaches has been discussed (e.g., [8, 14]), but not completely clarified, and is often ignored, see for example [27, 5, 21].

A related issue that has been studied in several papers is whether the information content of the views is sufficient to answer completely a given query. We say that a set of views is *lossless* with respect to a query, if, no matter what the database is, we can answer the query by solely relying on the content of the views. This concept has several applications, for example, in view selection [15], where we have to measure the quality of the choice of the views to materialize in the data warehouse, or in data integration, where we may be interested in checking whether the relevant queries can be answered by accessing only a given set of sources [28]. Several papers have addressed the issue of losslessness implicitly [27, 24, 28] or explicitly [11]. It should be noted, however, that losslessness is relative to the manner in which view-based query processing is performed, since the goal is lossless query processing. Thus, there ought to be two distinct notions of losslessness, with respect to query rewriting or with respect to query answering. Recent discussions of losslessness, such as [28, 11], do not reflect this distinction.

One reason for the confusion is that much of the work in this area has focused on using conjunctive queries for both target queries and view definitions, cf. [25]. This setting turns out to be extremely well behaved. In particular, query rewriting and query answering coincide, if we allow the target query to be written as a union of conjunctive queries. Furthermore, losslessness with respect to query rewriting and with respect to query answering also coincide, even if we require rewriting by conjunctive queries (disallowing unions). These results, implicit or explicit in [27], give the impression of a simple "terrain" of notions. Once, however, one goes even slightly beyond conjunctive queries or slightly modifies the view model, the terrain of notions gets considerably more complicated, as has already been observed in [3].

In this paper, we revisit the notions of query answering, query rewriting, and losslessness and clarify their relationship in the setting of semistructured databases, which capture data that do not fit into rigid, predefined schemas, and are best described by graph-based data models [6, 1, 22, 2]. The prevalent model for semistructured data is that of edge-labeled graphs, in which nodes describe data elements and edges describe relationships or values. (Extensions to node-labeled graphs or to node-edge-labeled graphs are straightforward.)

Methods for extracting information from semistructured data necessarily incorporate special querying mechanisms that are not common in traditional database systems. One such basic mechanism is that of *regular-path queries* (RPQs), which retrieves all pairs of nodes in the graph connected by a path conforming to a regular expression [7, 4]. We allow in our regular path queries also the inverse operator. The inverse operator is essential for expressing navigations in the database that traverse the edges both backward and forward [16]. We call such queries *two-way regular path queries* (2RPQs). Such path queries are useful in real settings (see for example [6, 7, 29]), and are part of the core of many query languages for semistructured data [4, 20, 17]. In our earlier work we studied both query answering and query rewriting for 2RPQs [9]. For an introductory survey on 2RPQs, see [13].

Our first result is a clean explanation of the relationship between query rewriting and query answering. We view query answering as the more robust notion among the two, since its definition is in terms of the information content of the view extensions. The certain tuples are the tuples whose presence in the answer logically follows from the view extension. In contrast, query rewriting is motivated by the pragmatic need to access the view extensions using a query language that is close, if not identical, to the language in which the target query and the views were formulated. For example, [27] considered rewriting of conjunctive queries by means of unions of conjunctive queries, [12] considered rewriting of RPQs by means of RPQs, and [9] considered rewriting of 2RPQs using 2RPQs.

The setup we use in this paper is that of *sound views*, in which view extension need not reflect global data completely. Thus, all we require from a view V_i defined in terms of a query Q_i is that its extension E_i with respect to a global database $\mathcal{B}$ is such that $E_i \subseteq Q_i(\mathcal{B})$. This setting corresponds to the long-standing *open-world approach* for querying incomplete information [30]. In this setting query answering can be characterized in terms of constraint satisfaction (or, equivalently, the homomorphism problem [19]), with a constraint template derived from the target query and view definition [14].

It now turns out that rewriting 2RPQs by means of 2RPQs amounts for seeking a "linear approximation" of query answering. That is, we retrieve a pair (c, d) from the view extension only if its inclusion in the answer is logically implied by a single path between c and d in the view extension. (For 2RPQs two-way paths are considered, while for RPQs one-way paths are considered.) We show that applying this linear approximation to the constraint-satisfaction framework yields the elegant automata-theoretic approach to query rewriting of [12], extended naturally to 2RPQs.

Once the relationship between query answering and query rewriting is clarified, we show that there are indeed two distinct notions of losslessness. Losslessness with respect to query rewriting is what has been called *exactness* in [12], while losslessness with respect to query answering, which we view as the more fundamental notion, is what has been studied in [11]. Since query rewriting is an approximation of query answering, exactness is a stronger notion than losslessness; exactness implies losslessness, but not vice versa. Exactness was taken in [12] to be a measure of quality of query rewriting, but we now see that it conflates query rewriting with losslessness. A better way to measure the quality of query rewriting is to measure its quality as an approximation. We say that query rewriting is *perfect* if it is equivalent to query answering. Thus, exactness is the conjunction of perfectness and losslessness (with respect to query answering). We also show that the constraint-theoretic approach and the automata-theoretic approach can be combined to give algorithmic characterization of the three notions: perfectness, losslessness, and exactness.

Finally, we consider lossiness, which we view as the central challenge of view-based query processing, as lossiness is more likely to be the norm rather than the exception. Once a schema designer has learned that a view decomposition is lossy with respect to a certain query, how should this "loss" be dealt with? We believe that database design tools should help users to "cope with loss". In particular, we believe that it would be useful to the user to understand what information is lost by view-based query answering. We discuss a variety of mechanisms aimed at explaining such lossiness to the user.

The paper is organized as follows. In Section 2 we recall the basic notions related to view-based query processing, and in Section 3 we recall the relationship between query answering and constraint satisfaction. In Section 4 we discuss the relationship between answering and rewriting. In Section 5 we study losslessness with respect to rewriting for 2RPQs and in Section 6 losslessness with respect to answering. For the latter we introduce the notion of linear fragment of certain answers. In Section 7 we discuss the relationship between exactness, perfectness, losslessness, and lossiness and conclude the paper. Proof sketches are given in an appendix.

2 Preliminaries

Following the usual approach in semistructured data [2], we define a *semistructured database* as a finite directed graph whose edges are labeled by elements from a given finite alphabet Σ. Each node represents an objects and an edge from object x to object y labeled by r, denoted $r(x, y)$, represents the fact that relation r holds between x and y. Observe that a semistructured database can be seen as a (finite) relational structure over the set Σ of binary relational symbols. A *relational structure* (or simply *structure*) $\mathcal{B}$ over Σ is a pair $(\Delta^{\mathcal{B}}, \cdot^{\mathcal{B}})$, where

$\Delta^{\mathcal{B}}$ is a finite domain and $\cdot^{\mathcal{B}}$ is a function that assigns to each relation symbol in $r \in \Sigma$ a binary relation $r^{\mathcal{B}}$ over $\Delta^{\mathcal{B}}$, also denoted by $r(\mathcal{B})$.

A query is a function from relational structures to relations, assigning to each relational structure over a given alphabet a relation of a certain arity. In this paper we deal mainly with binary queries. A *regular-path query* (RPQ) over Σ is defined in terms of a regular language over Σ. The *answer $Q(\mathcal{B})$ to an RPQ Q over a database $\mathcal{B}$* is the set of pairs of objects connected in $\mathcal{B}$ by a directed path traversing a sequence of edges forming a word in the regular language $L(Q)$ defined by Q.

RPQs allow for navigating the edges of a semistructured databases only in the forward direction. RPQs extended with the ability of navigating database edges backward are called *two-way regular-path queries* (2RPQs) [9]. Formally, we consider an alphabet $\Sigma^{\pm} = \Sigma \cup \{r^- \mid r \in \Sigma\}$ which includes a new symbol r^- for each relation symbol r in Σ. The symbol r^- denotes the *inverse* of the binary relation r. If $p \in \Sigma^{\pm}$, then we use p^- to mean the inverse of p, i.e., if p is r, then p^- is r^-, and if p is r^-, then p^- is r. A 2RPQ over Σ is defined in terms of a regular language over $\Sigma^{\pm}$. The *answer $Q(\mathcal{B})$ to a 2RPQ Q over a database $\mathcal{B}$* is the set of pairs of objects connected in $\mathcal{B}$ by a semipath that conforms to the regular language $L(Q)$. A *semipath* in $\mathcal{B}$ from x to y (labeled with $p_1 \cdots p_n$) is a sequence of the form $(x_0, p_1, x_1, \ldots, x_{n-1}, p_n, x_n)$, where $n \geq 0$, $x_0 = x$, $x_n = y$, and for each x_{i-1}, p_i, x_i, we have that $p_i \in \Sigma^{\pm}$, and, if $p_i = r$ then $(x_{i-1}, x_i) \in r(\mathcal{B})$, and if $p_i = r^-$ then $(x_i, x_{i-1}) \in r(\mathcal{B})$. Intuitively, a semipath $(x_0, p_1, x_1, \ldots, x_{n-1}, p_n, x_n)$ corresponds to a navigation of the database from x_0 to x_n, following edges forward or backward, according to the sequence of edge labels $p_1 \cdots p_n$. Note that the objects in a semipath are not necessarily distinct. A semipath is said to be *simple* if no object in it appears more than once. A *linear database* with endpoints x and y is a database constituted by a single simple semipath from x to y. We say that a semipath $(x_0, p_1, \ldots, p_n, x_n)$ *conforms to* a 2RPQ Q if $p_1 \cdots p_n \in L(Q)$. Summing up, a pair (x, y) of objects is in the answer $Q(\mathcal{B})$ if and only if, by starting from x, it is possible to reach y by navigating on $\mathcal{B}$ according to one of the words in $L(Q)$. The notions above can be extended to *two-way path queries*, which are defined similarly to 2RPQs, but without requiring the language to be regular.

Consider now a semistructured database that is accessible only through a collection of views expressed as 2RPQs, and suppose we need to answer a 2RPQ over the database only on the basis of our knowledge on the views. Specifically, the collection of views is represented by a finite set $\mathcal{V}$ of *view symbols*, each denoting a binary relation. Each view symbol $V \in \mathcal{V}$ has an associated *view definition V^{Σ}*, which is a 2RPQ over Σ. A *$\mathcal{V}$-extension $\mathcal{E}$* is a relational structure over $\mathcal{V}$. We consider views to be *sound* [3, 23], i.e., we model a situation where the extension of the views provides a subset of the results of applying the view definitions to the database. Formally, given a set $\mathcal{V}$ of views and a database $\mathcal{B}$, we use $\mathcal{V}^{\Sigma}(\mathcal{B})$ to denote the $\mathcal{V}$-extension $\mathcal{E}$ such that $V(\mathcal{E}) = V^{\Sigma}(\mathcal{B})$, for each $V \in \mathcal{V}$. We say that a $\mathcal{V}$-extension $\mathcal{E}$ is *sound wrt a database $\mathcal{B}$* if $\mathcal{E} \subseteq \mathcal{V}^{\Sigma}(\mathcal{B})$.

In other words, for a view $V \in \mathcal{V}$, all the tuples in $V(\mathcal{E})$ must appear in $V^{\Sigma}(\mathcal{B})$, but $V^{\Sigma}(\mathcal{B})$ may contain tuples not in $V(\mathcal{E})$.

Given a set $\mathcal{V}$ of views, a $\mathcal{V}$-extension $\mathcal{E}$, and a query Q over Σ, the set of *certain answers* (under sound views) to Q with respect to $\mathcal{V}$ and $\mathcal{E}$ is the set of pairs (x, y) of objects such that $(x, y) \in Q(\mathcal{B})$ for every database $\mathcal{B}$ wrt which $\mathcal{E}$ is sound, i.e., $\mathcal{E} \subseteq \mathcal{V}^{\Sigma}(\mathcal{B})$. *View-based query answering* consists in deciding whether a given pair of objects is a certain answer to Q with respect to $\mathcal{V}$ and $\mathcal{E}$. Given a set $\mathcal{V}$ of views and a query Q, we denote by $cert_{Q,\mathcal{V}}$ the query that, for every $\mathcal{V}$-extension $\mathcal{E}$, returns the set of certain answers to Q with respect to $\mathcal{V}$ and $\mathcal{E}$.

View-based query answering has also been tackled using an indirect approach, called *view-based query rewriting*. According to such an approach, a query Q over the database alphabet is processed by first reformulating Q into an expression of a fixed query language over the view alphabet $\mathcal{V}$ (called *rewriting*), and then evaluating the rewriting over the view extensions. Formally, let Q be a query over the database alphabet, and let Q_r be a query over the view alphabet $\mathcal{V}$. We say that Q_r is a *rewriting of Q under sound views $\mathcal{V}$* (or simply, with respect to views $\mathcal{V}$), if for every database $\mathcal{B}$ and for every $\mathcal{V}$-extension $\mathcal{E}$ with $\mathcal{E} \subseteq \mathcal{V}^{\Sigma}(\mathcal{B})$, we have that $Q_r(\mathcal{E}) \subseteq Q(\mathcal{B})$. Since 2RPQs are monotone, by results in [14] (Proposition 13 and 24), rewritings admit the following simpler characterization. A 2RPQ Q_r is a rewriting of a 2RPQ Q if, for every database $\mathcal{B}$, we have that $Q_r(\mathcal{V}^{\Sigma}(\mathcal{B})) \subseteq Q(\mathcal{B})$. We make use of this characterization in the following.

Obviously, in view-based query rewriting, we are not interested in arbitrary rewritings, but we aim at computing rewritings that capture the original query at best. Let $\mathcal{C}$ be a query class in which rewritings are expressed. A query Q_r in $\mathcal{C}$ is a *$\mathcal{C}$-maximal rewriting* of Q under $\mathcal{V}$ if (i) it is a rewriting of Q under $\mathcal{V}$, and (ii) for each query Q'_r in $\mathcal{C}$ that is a rewriting of Q under $\mathcal{V}$ and for each database $\mathcal{B}$ and each $\mathcal{V}$-extension $\mathcal{E}$ with $\mathcal{E} \subseteq \mathcal{V}^{\Sigma}(\mathcal{B})$, we have that $Q'_r(\mathcal{E}) \subseteq Q_r(\mathcal{E})$. Since in this paper we are focusing on 2RPQs, we are interested in the case where also rewritings are 2RPQs over the view alphabet $\mathcal{V}$, i.e., rewritings are expressed in the same language as queries over the database.

Throughout the paper, we will assume that RPQs are expressed as finite state automata over an appropriate alphabet. Besides standard (one-way) deterministic and non-deterministic finite state automata over words (1DFAs and 1NFAs, respectively), we assume familiarity with two-way automata (2NFAs) [26].

3 Answering and constraint satisfaction

In this work we make use of the tight relationship between view-based query answering for RPQs and 2RPQs and constraint satisfaction, which we recall here.

A *constraint-satisfaction problem (CSP)* is traditionally defined in terms of a set of variables, a set of values, and a set of constraints, and asks whether there is an assignment of the variables with the values that satisfies the constraints. A characterization of CSP can be given in terms of homomorphisms between relational structures [19]. Here we consider relational structures whose relations are of arbitrary arity.

A *homomorphism* $h : A \to B$ between two relational structures A and B over the same alphabet is a mapping $h : \Delta^A \to \Delta^B$ such that, if $(c_1, \ldots, c_n) \in r(A)$, then $(h(c_1), \ldots, h(c_n)) \in r(B)$, for every relation symbol r in the alphabet. Let $\mathcal{A}$ and $\mathcal{B}$ be two classes of structures. The *(uniform) constraint-satisfaction problem* $CSP(\mathcal{A}, \mathcal{B})$ is the following decision problem: given a structure $A \in \mathcal{A}$ and a structure $B \in \mathcal{B}$ over the same alphabet, is there a homomorphism $h : A \to B$? When $\mathcal{B}$ consists of a single structure B and $\mathcal{A}$ is the set of all structures over the alphabet of B, we get the so-called *non-uniform* constraint-satisfaction problem, denoted by $CSP(B)$, where B is fixed and the input is just a structure $A \in \mathcal{A}$. As usual, we use $CSP(B)$ also to denote the set of structures A such that there is a homomorphism from A to B. From the very definition of CSP it follows directly that every $CSP(\mathcal{A}, \mathcal{B})$ problem is in NP.

A tight relationship between non-uniform CSP and view-based query answering for RPQs and 2RPQs has been developed in [10, 14]. Such a relationship is based on the notions of constraint template, associated to the query and view definitions, and constraints instance, associated to the view extension. Formally, given a 2RPQ Q and a set $\mathcal{V}$ of 2RPQ views, the *constraint template* $CT_{Q,\mathcal{V}}$ of Q with respect to $\mathcal{V}$ is the relational structure C defined as follows.

- The alphabet of C is $\mathcal{V} \cup \{U_i, U_f\}$, where each view denotes a binary relation symbol, and U_i and U_f are unary relation symbols.
- Let $A^Q = (\Sigma^{\pm}, S^Q, S_0^Q, \varrho^Q, F^Q)$ be a 1NFA for Q, where $\Sigma^{\pm}$ is the alphabet, S^Q is the set of states, S_0^Q is the set of initial states, ϱ^Q is the transition relation, and F^Q is the set of final states. The structure $C = (\Delta^C, \cdot^C)$ is given by:
  - $\Delta^C = 2^{S^Q}$;
  - $\sigma \in U_i(C)$ iff $S_0^Q \subseteq \sigma$;
  - $\sigma \in U_f(C)$ iff $\sigma \cap F^Q = \emptyset$;
  - for a view $V \in \mathcal{V}$, we have that $(\sigma_1, \sigma_2) \in V^C$ iff there exists a word $q_1 \cdots q_k \in L(V^\Sigma)$ and a sequence $T_0, \ldots, T_k$ of subsets of S^Q such that the following hold:
    1. $T_0 = \sigma_1$ and $T_k = \sigma_2$,
    2. if $s \in T_i$ and $(s, q_{i+1}, t) \in \varrho^Q$ then $t \in T_{i+1}$, for $0 \leq i < k$, and
    3. if $s \in T_i$ and $(s, q_i^-, t) \in \varrho^Q$ then $t \in T_{i-1}$, for $0 < i \leq k$.

Intuitively, the constraint template represents for each view V how the states of A^Q (i.e., of the 1NFA for Q) change when we follow database edges according to what specified by words in $L(V^\Sigma)$. Specifically, the last condition above corresponds to saying that a pair of sets of states (σ_1, σ_2) is in $V(C)$ if and only if there is some word w in $L(V^\Sigma)$ such that the following holds: if we start from a state in σ_1 on the left edge of w and move back and forth on w according to the transitions in A^Q, then, if we end up at the left edge of w we can be only in states in σ_1, and if we end up at the right edge of w we can be only in states in σ_2; similarly, if we start from a state in σ_2 on the right edge of w. Moreover, the sets of states in $U_i(C)$ contain all initial states of A^Q, while the sets of states in $U_f(C)$ do not contain any final state of A^Q. This takes into account that we aim at characterizing counterexamples to view-based query answering, and hence we are interested in not getting to a final state of A^Q, regardless of the initial state from which we start and how we follow transitions.

Observe that, to check the existence of a word $q_1 \cdots q_k \in L(V^\Sigma)$ and of a sequence $T_0, \ldots, T_k$ of subsets of S such that conditions 1–3 above are satisfied, we can resort to a construction analogous to the one in [32]. Hence, such a check can be done in polynomial space in the size of Q, and in fact in nondeterministic logarithmic space in the size of V^Σ.

Given a $\mathcal{V}$-extension $\mathcal{E}$ and a pair of objects c, d, the *constraint instance* $\mathcal{E}^{c,d}$ is the structure $I = (\Delta^I, \cdot^I)$ over the alphabet $\mathcal{V} \cup \{U_i, U_f\}$ defined as follows:

- $\Delta^I = \Delta^\mathcal{E} \cup \{c, d\}$;
- $V(I) = V(\mathcal{E})$, for each $V \in \mathcal{V}$;
- $U_i(I) = \{c\}$, and $U_f(I) = \{d\}$.

The following theorem provides the characterization of view-based query answering in terms of CSP.

Theorem 1 ([14]). *Let Q be a 2RPQ, $\mathcal{V}$ a set of 2RPQ views, $\mathcal{E}$ a $\mathcal{V}$-extension, and c, d a pair of objects. Then, $(c,d) \notin cert_{Q,\mathcal{V}}(\mathcal{E})$ if and only if there is a homomorphism from $\mathcal{E}^{c,d}$ to $CT_{Q,\mathcal{V}}$.*

4 Relationship between rewriting and answering

The relationship between answering and rewriting in view-based query processing is not always well understood. As we said before, one reason for the confusion is that much of the work in this area has focused on a setting based on conjunctive queries, where answering and rewriting coincide. Indeed, if we allow the target query to be written as a union of conjunctive queries (UCQs), then the UCQ-maximal rewriting of the query computes exactly the certain answers. Things get more complicated with RPQs and 2RPQs. Interestingly, we show next that we can use the above characterization of view-based query answering in terms of CSP, to characterize also query rewriting, thus providing a clean explanation of the relationship between answering and rewriting.

A preliminary observation is that one can restrict the attention to linear databases when looking for counterexamples to rewritings.

Lemma 1 ([9]). *Let Q be a 2RPQ, $\mathcal{V}$ a set of 2RPQ views, and w a word over $\mathcal{V}^\pm$. Then w is not a rewriting (note that w can be viewed as a 2RPQ) of Q with respect to $\mathcal{V}$ if and only if there exists a linear database $\mathcal{B}$ with endpoints c and d, and a view extension $\mathcal{E}$ with $\mathcal{E} \subseteq \mathcal{V}^\Sigma(\mathcal{B})$, such that $(c,d) \in w(\mathcal{E})$ [4] but $(c,d) \notin Q(\mathcal{B})$.*

[4] We consider here w as a path-query constituted by a single word.

Making use of this result, we are able to exploit the constraint template itself as a 1NFA that recognizes the words that do not belong to a rewriting. However, we have first to take care of the fact that only direct view symbols appear in the constraint template, while a rewriting is a 1NFA over direct and inverse view symbols. To do so, we extend the constraint template by adding to the alphabet, for each symbol $V \in \mathcal{V}$, also the inverse symbol V^-. Then we define $(\sigma_1, \sigma_2) \in V^{-C}$ if and only if $(\sigma_2, \sigma_1) \in V^C$. We denote the resulting constraint template with $CT^{\pm}_{Q,\mathcal{V}}$. Observe that the construction of $CT^{\pm}_{Q,\mathcal{V}}$ from $CT_{Q,\mathcal{V}}$ takes into account the perfect symmetry that we have when moving along direct and inverse database and view symbols.

Now, $C = CT^{\pm}_{Q,\mathcal{V}}$ can be viewed directly as a 1NFA A^{nr} over $\mathcal{V}^{\pm}$, by taking the domain of C as the set of states of A^{nr}, the extension of U_i and U_f in C respectively as the set of initial and final states, and by deriving the transition relation of A^{nr} from the extension of the various $v \in \mathcal{V}^{\pm}$ as follows: A^{nr} has a transition (σ_1, v, σ_2) if and only if $(\sigma_1, \sigma_2) \in v^C$.

Let A^{rew} be a 1NFA accepting the complement of A^{nr}. Then the following characterization of the 2RPQ-maximal rewriting holds.

Theorem 2. *Let Q be a 2RPQ and $\mathcal{V}$ a set of 2RPQ views. Then A^{rew} is the 2RPQ-maximal rewriting of Q with respect to $\mathcal{V}$.*

The above characterization provides a nice combination of the constraint based [10] and automata theoretic [9] approaches to view-based query processing for 2RPQs, and goes into the heart of view-based rewriting. A (language) rewriting accepts a pair (c, d) if there is a path between c and d such that, if we view this path as a linear view extension, then (c, d) is in the certain answer with respect to this view extension. That means that there is no homomorphism from this path into the constraint template. Indeed, for a path, the existence of a homomorphism into the constraint template means that the path is accepted by the template, viewed as an automaton. Naturally, the difference with view-based query answering, is that we are not limited to linear view extensions only. Suppose that V_i and V_j connect the same pair of objects in a view extension. In rewriting we have to ignore this and allow the choice of distinct pairs of objects for the two views in a counterexample database. Query answering instead takes into account that the two pairs of objects are the same. Thus, query answering is more precise than query rewriting. On the other hand, the simplification introduced by query rewriting allows to have polynomial time evaluation in the size of the data, while query answering is coNP-complete [8].

Finally, observe that the above construction provides also optimal upper bounds for the problems of computing the 2RPQ-maximal rewriting and of determining whether such a rewriting is nonempty [9]. Indeed, the constraint template, and hence the 1NFA A^{nr} can be constructed in EXPTIME and is of exponential size [14]. Hence, its complement A^{rew}, which provides the 2RPQ-maximal rewriting, is of double exponential size and can be constructed in 2EXPTIME. On the other hand, if we only want to check its emptiness, we can complement A^{nr} on the fly, getting an EXPSPACE upper bound. All these bounds are tight [12].

5 losslessness with respect to rewriting

We deal now with the problem of analyzing the loss of information in view-based query processing, and of characterizing the quality of certain answers and of rewritings. For this purpose, we make use of the following basic notions.

- To determine whether the information content of a set of views is sufficient to answer completely a given query, we make use of the notion of losslessness [24, 11]. In [11], a set of views $\mathcal{V}$ is said to be *lossless* with respect to a query Q, if for every database $\mathcal{B}$ we have that $Q(\mathcal{B}) = cert_{Q,\mathcal{V}}(\mathcal{V}^{\Sigma}(\mathcal{B}))$.
- As for rewritings, equivalence of a rewriting to the original query, modulo the view definitions, is called exactness (cf. [27, 12]). Formally, a rewriting Q_r in a certain query class $\mathcal{C}$ is *an exact rewriting* of Q with respect to views $\mathcal{V}$, if for every database $\mathcal{B}$ we have that $Q(\mathcal{B}) = Q_r(\mathcal{V}^{\Sigma}(\mathcal{B}))$.
- Finally, to determine whether we lose answering power by resorting to rewriting, we can compare rewritings with the certain answers, with the aim of checking whether the two are actually equivalent. A rewriting Q_r in a certain query class $\mathcal{C}$ is *a perfect rewriting* of Q with respect to views $\mathcal{V}$, if for every database $\mathcal{B}$ and every view extension $\mathcal{E}$ with $\mathcal{E} \subseteq \mathcal{V}^{\Sigma}(\mathcal{B})$ we have that $cert_{Q,\mathcal{V}}(\mathcal{E}) = Q_r(\mathcal{E})$.

The first notion aims at determining possible loss with respect to view-based query answering, and will be discussed in the next section. The other two notions deal with the loss of information in the case of rewritings, and are discussed below.

In the case of conjunctive queries, the best rewriting of a conjunctive query Q is a union of conjunctive queries. Therefore, checking exactness amounts to verifying whether Q is contained in the UCQ-maximal rewriting. The latter is a, possibly exponential, union of conjunctive queries, each of linear size. Since a conjunctive query is contained in a union of conjunctive queries only if it is contained in one of its disjuncts, it suffices to check whether there is a single conjunctive query in the rewriting that is equivalent to Q, after substituting the view definitions. This can be done in NP in the size of Q. As for perfectness, we already observed that the maximal rewriting computes exactly the certain answers. Therefore, the maximal rewriting is always perfect.

In the case of 2RPQs, things are more complicated. Exactness is studied in [9], where it is shown that verifying the existence of an exact rewriting is 2EXPTIME-complete. On the other hand, perfectness is a new notion, and we provide here a method for checking perfectness of the 2RPQ-maximal rewriting A^{rew} of a query Q. Exploiting the fact that 2RPQs are monotone, by results in [14], this amounts to check whether for all databases $\mathcal{B}$ we have that $cert_{Q,\mathcal{V}}(\mathcal{V}^{\Sigma}(\mathcal{B})) \subseteq A^{rew}(\mathcal{V}^{\Sigma}(\mathcal{B}))$. To do this check, we can in principle directly use the technique in [14] based on *view-based containment* (see [14] for definitions): the 2RPQ-maximal rewriting A^{rew} is a 1NFA of double exponential size in Q, and checking whether for all databases $\mathcal{B}$ we have that $cert_{Q,\mathcal{V}}(\mathcal{V}^{\Sigma}(\mathcal{B})) \subseteq A^{rew}(\mathcal{V}^{\Sigma}(\mathcal{B}))$ amounts to checking whether Q is view-based contained in A^{rew}, which can be done in NEXPTIME in Q and A^{rew} [14]. This gives us a N3EXPTIME upper bound. However, we can do better, by making use of the fact that we have obtained the 1NFA A^{rew} for the rewriting by complementation, and thus by application of the subset construction. This allows us to characterize non-membership in the answer set to A^{rew} by homomorphism into a structure $C = (\Delta^C, \cdot^C)$, called the *rewriting constraint template* $CTR_{A^{rew},\mathcal{V}}$ of A^{rew}, defined as follows:

- The alphabet of C is $\mathcal{V}^{\pm} \cup \{U_i, U_f\}$, where U_i and U_f denote unary relation symbols.
- Let $A^{nr} = (\mathcal{V}^{\pm}, S, S_0, \varrho, F)$ be a 1NFA for the complement of the rewriting (see Section 4). Then
  - $\Delta^C = 2^S$;
  - $\sigma \in U_i^C$ iff $S_0 \subseteq \sigma$;
  - $\sigma \in U_f^C$ iff $\sigma \subseteq F$;
  - $(\sigma_1, \sigma_2) \in r^C$ iff $\varrho(\sigma_1, r) \subseteq \sigma_2$ and $\varrho(\sigma_2, r^-) \subseteq \sigma_1$.

To characterize perfectness of the rewriting in terms of CSP, we need to introduce proper constraint templates (see also [14]). Given the rewriting constraint template $CT_{A^{rew},\mathcal{V}}$, a *proper constraint template* $CT^{\alpha,\beta}_{A^{rew},\mathcal{V}}$ is obtained by eliminating from $\mathcal{U}_i$ all but one element α and from $\mathcal{U}_f$ all but one element β.

Lemma 2. *Let Q be a 2RPQ and $\mathcal{V}$ be a set of 2RPQ views. Then the 2RPQ-maximal rewriting of Q with respect to $\mathcal{V}$ is perfect if and only if for every proper constraint template $CTR^{\alpha,\beta}_{A^{rew},\mathcal{V}}$ of $CTR_{A^{rew},\mathcal{V}}$, there exists a homomorphism from $CTR_{A^{rew},\mathcal{V}}\alpha,\beta$ to $CT_{Q,\mathcal{V}}$.*

The above characterization provides us with a tighter upper bound than the one discussed above.

Theorem 3. *Let Q be a 2RPQ and $\mathcal{V}$ be a set of 2RPQ views. Then checking whether the 2RPQ-maximal rewriting of Q with respect to $\mathcal{V}$ is perfect can be done in N2EXPTIME in the size of Q and in NEXPTIME in the size of $\mathcal{V}^{\Sigma}$.*

We conjecture that such an upper bound is tight.

6 Losslessness with respect to answering

We now turn to verifying losslessness with respect to answering. We want to verify whether a set of views $\mathcal{V}$ is lossless with respect to a query Q, i.e., verifying whether $cert_{Q,\mathcal{V}}$ is equivalent to Q (cf. [11]).

In the case of conjunctive queries, we already observed that the maximal rewriting computes exactly the certain answers. Therefore, losslessness with respect to answering and losslessness with respect to rewriting coincide. The case of RPQs and 2RPQs is much more involved. Losslessness with respect to answering for RPQs was studied in [11]. In the rest of this section we study losslessness with respect to answering for 2RPQs.

The main step toward this goal is to characterize the linear fragment of certain answers. Formally, the *linear fragment of certain answers* $clin_{Q,\mathcal{V}}$ for a 2RPQ Q with respect to a set $\mathcal{V}$ of 2RPQ views is the maximal two-way path query[5] Q' over Σ such that, for every database $\mathcal{B}$ we have that $Q'(\mathcal{B}) \subseteq cert_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B}))$. The following result shows that, in order to characterize the linear fragment of certain answers it is sufficient to restrict the attention to linear databases, i.e., databases constituted by a single semipath.

Lemma 3. *Let Q' be two-way path query. Then, if there is a database $\mathcal{B}$ and a pair of objects (c,d) in $\mathcal{B}$ such that $(c,d) \in Q'(\mathcal{B})$ and $(c,d) \notin cert_{Q,\mathcal{V}}(\mathcal{V}^{\Sigma}(\mathcal{B}))$, then there is a linear database $\mathcal{B}_\ell$ with endpoints c' and d' such that $(c',d') \in Q'(\mathcal{B}_\ell)$ and $(c',d') \notin cert_{Q,\mathcal{V}}(\mathcal{V}^{\Sigma}(\mathcal{B}_\ell))$.*

Hence, to construct the linear fragment of certain answers, we characterize the set of linear databases of the form $\mathcal{B} = (x_0, q_1, x_1, q_2, \ldots, q_m, x_m)$, for some m, such that $(x_0, x_m) \notin cert_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B}))$. By Theorem 1, this holds if and only if there is a homomorphism from the constraint instance $\mathcal{V}(\mathcal{B})^{x_0,x_m}$ to the constraint template $CT_{Q,\mathcal{V}}$. In other words, $(x_0, x_m) \notin cert_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B}))$ if and only if there is a function $\ell(\cdot)$ (i.e., the homomorphism) that labels $x_0, \ldots, x_m$ with sets of states of the 1NFA $A^Q = (\Sigma^{\pm}, S^Q, S_0^Q, \varrho^Q, F^Q)$ for Q such that the following conditions (which we call *CT-conditions*) hold:

- $S_0^Q \subseteq \ell(x_0)$;
- $\ell(x_m) \cap F^Q = \emptyset$;
- for each pair of objects x_j and x_h in $\mathcal{B}$ and each view V in $\mathcal{V}$, we have that, if $(x_j, x_h) \in V^{\Sigma}(\mathcal{B})$ then there exists a word $q_1 \cdots q_k \in L(V^{\Sigma})$ and a sequence $T_0, \ldots, T_k$ of subsets of S^Q such that the following hold:
  1. $T_0 = \ell(x_j)$ and $T_k = \ell(x_h)$,
  2. if $s \in T_i$ and $(s, q_{i+1}, t) \in \varrho^Q$ then $t \in T_{i+1}$, for $0 \leq i < k$, and
  3. if $s \in T_i$ and $(s, q_i^-, t) \in \varrho^Q$ then $t \in T_{i-1}$, for $0 < i \leq k$.

Thus, we are looking for words of the form $\ell_0, q_1, \ldots, q_m, \ell_m$, where each ℓ_i is a set of states of A^Q, representing $\ell(x_i)$, and that satisfies the above conditions. As shown by the following lemma, we can construct a 1NFA that accepts such words, and then project away the ℓ_i transitions.

For a word $w \in \Sigma^{\pm *}$, we denote with $\mathcal{B}_w^{a,b}$ the linear database constituted by a path from a to b spelled by w (with arbitrary intermediate nodes).

Lemma 4. *Let Q be a 2RPQ and $\mathcal{V}$ be a set of 2RPQ views. Then we can construct in double exponential time in Q and $\mathcal{V}^{\Sigma}$ two 1NFAs A^{nlin} and A^{lin} such that:*

- *A^{nlin} accepts all words $w \in \Sigma^{\pm *}$ such that $(a,b) \notin cert_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B}_w^{a,b}))$.*
- *A^{lin} accepts all words $w \in \Sigma^{\pm *}$ such that $(a,b) \in cert_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B}_w^{a,b}))$.*

[5] Recall from Section 2 that two-way path queries are a generalization of 2RPQs in which the language used to define a query is not required to be regular.

Both 1NFAs A^{nlin} and A^{lin} have a number of states that is doubly exponential in both Q and $\mathcal{V}^{\Sigma}$. Obviously, the two automata accept complementary languages. However, in the proof of the above lemma we show how to construct A^{lin} directly, instead of complementing A^{nlin}, to avoid an additional exponential blowup.

Theorem 4. *Let Q be a 2RPQ and $\mathcal{V}$ be a set of 2RPQ views, and A^{nlin} and A^{lin} the 1NFAs defined as above. Then A^{lin} is the linear fragment $clin_{Q,\mathcal{V}}$ of the certain answers of Q with respect to $\mathcal{V}$.*

Corollary 1. *The linear fragment of a 2RPQ with respect to a set of 2RPQ views is a 2RPQ.*

Now we can deal with checking losslessness with respect to answering. To check whether a set $\mathcal{V}$ of 2RPQ views is lossless with respect to a 2RPQ query Q, we have to check whether for all databases $\mathcal{B}$, we have that $Q(\mathcal{B})$ is contained in the certain answers $cert_{Q,\mathcal{V}}(\mathcal{V}^{\Sigma}(\mathcal{B}))$. Since Q is itself a 2RPQ, and hence a path query, it suffices to check whether Q is contained in the linear fragment of the certain answers, i.e., whether for all databases $\mathcal{B}$ we have that $Q(\mathcal{B}) \subseteq clin_{Q,\mathcal{V}}(\mathcal{B})$. By exploiting the characterization of the linear fragment of the certain answers in terms of 1NFAs provided above, we get the following upper bound, which is tight already for RPQs [11].

Theorem 5. *Let Q be a 2RPQ and $\mathcal{V}$ be a set of 2RPQ views. Then checking whether $\mathcal{V}$ is lossless with respect to Q can be done in EXPSPACE in the size of Q and $\mathcal{V}^{\Sigma}$.*

Observe that when we have that a set of views is lossless with respect to a query, we have also, as a side effect, that the linear fragment of certain answers is equivalent to the certain answers, since both are equivalent to the query. Now it is natural to try to understand when the linear fragment of certain answers is equivalent to the certain answers, independently of losslessness with respect to answering. Indeed, in this case, since the certain answers are actually expressible as a 2RPQ over the database, we directly get a characterization of the certain answers in the same language used for expressing the query and thus in terms that are understandable to the user.

Given a 2RPQ Q and a set of 2RPQ views $\mathcal{V}$, checking whether the linear fragment of certain answers is equivalent to the certain answers amounts to checking whether for every database $\mathcal{B}$ we have that $cert_{Q,\mathcal{V}}(\mathcal{V}^{\Sigma}(\mathcal{B})) \subseteq clin_{Q,\mathcal{V}}(\mathcal{B})$. Consider the 1NFA A^{lin}, constructed above, recognizing the linear fragment $clin_{Q,\mathcal{V}}$ of the certain answers of Q. One can verify that the certain answers $cert_{A^{lin},V}$ of A^{lin} with respect to $\mathcal{V}$ are actually equivalent to A^{lin} itself. Hence, the above check amounts to verifying whether for all databases $\mathcal{B}$, we have that $cert_{Q,\mathcal{V}}(\mathcal{V}^{\Sigma}(\mathcal{B})) \subseteq cert_{A^{lin},\mathcal{V}}(\mathcal{V}^{\Sigma}(\mathcal{B}))$. This is a form of view-based containment, and by [14] it can done in NEXPTIME in the size of Q and A^{lin}. Considering that A^{lin} has a number of states that is doubly exponential in the size of Q and $\mathcal{V}^{\Sigma}$, we get the following upper bound.

Theorem 6. *Let Q be a 2RPQ and $\mathcal{V}$ be a set of 2RPQ views. Then checking whether the certain answers $cert_{Q,\mathcal{V}}$ of Q with respect to $\mathcal{V}$ is equivalent to its linear fragment can be done in N3EXPTIME in the size of Q and $\mathcal{V}^{\Sigma}$.*

We conjecture that such an upper bound can be improved.

7 Discussion

In this paper, we have revisited the notions of answering, rewriting and losslessness in the context of view-based query processing in semistructured databases. In particular the richness of (two-way) regular path query allows us to uncover several subtle distinctions between the notions of rewriting and answering, and losslessness with respect to them. Such distinctions are completely blurred when focusing on conjunctive queries, due to the fact that rewriting and answering collapse.

Let Q be a 2RPQ, $\mathcal{V}$ a set of 2RPQ views, and let $R^{max}_{Q,\mathcal{V}}$ denote the 2RPQ-maximal rewriting of Q with respect to $\mathcal{V}$. Then, by definition and by results in [14] exploiting the fact that 2RPQs are monotone, we know that for every database $\mathcal{B}$, the following holds:

$$R^{max}_{Q,\mathcal{V}}(\mathcal{V}^{\Sigma}(\mathcal{B})) \subseteq^{(1)} clin_{Q,\mathcal{V}}(\mathcal{B}) \subseteq^{(2)} cert_{Q,\mathcal{V}}(\mathcal{V}^{\Sigma}(\mathcal{B})) \subseteq^{(3)} Q(\mathcal{B})$$

Notice that we start from a database $\mathcal{B}$ and are evaluating $cert_{Q,\mathcal{V}}$ and $R^{max}_{Q,\mathcal{V}}$ over a particular view extension, namely $\mathcal{V}^{\Sigma}(\mathcal{B})$, instead of an arbitrary view extension $\mathcal{E}$ that is sound with respect to $\mathcal{B}$, i.e., such that $\mathcal{E} \subseteq \mathcal{V}^{\Sigma}(\mathcal{B})$. This is due to the fact that our aim is to understand whether there is loss. It is clear that when $\mathcal{E}$ is a strict subset of $\mathcal{V}^{\Sigma}(B)$ then loss may occur, but this has nothing to do with the "quality" of the views.

It is now of interest to consider the cases in which some or all of the above inclusions are actually equalities, since these correspond to the notions studied in this paper.

1. If $R^{max}_{Q,\mathcal{V}}$ is exact, i.e., is equivalent to Q (modulo the view definitions), then all three inclusions are actually equalities. Hence, not only we have losslessness with respect to rewriting but we also have both that the views are lossless with respect to answering and that $R^{max}_{Q,\mathcal{V}}$ is perfect. Thus exactness of the maximal rewriting is the strongest notion, combining both losslessness of the views and perfectness of the rewriting.
2. If $R^{max}_{Q,\mathcal{V}}$ is perfect, i.e., is equivalent to $cert_{Q,\mathcal{V}}$, then inclusions (1) and (2) are actually equalities. In this case, we also get that $cert_{Q,\mathcal{V}}$ has to coincide with $clin_{Q,\mathcal{V}}$. By Corollary 1 we can conclude that the certain answers are expressible as a 2RPQ over $\mathcal{B}$.
3. If $\mathcal{V}$ is lossless with respect to Q, i.e., we have losslessness with respect to answering, then inclusion (3) is actually an equality. Moreover, in this case, since Q is itself a 2RPQ, and hence is linear, then $cert_{Q,\mathcal{V}}$ has also to be linear and has to coincide with $clin_{Q,\mathcal{V}}$. Hence inclusion (2) is also an equality. In this case we know that there is not loss of information related to the fact that we are answering the query based on a set of views.

4. Finally, if $\mathcal{V}$ is lossy with respect to Q, i.e., we have lossiness with respect to answering, we can check whether inclusion (2) is actually an equality, i.e., whether the certain answers are actually expressible as a 2RPQ over the database. If this is the case, we directly get a characterization of the certain answers in the same language used for expressing the query, namely 2RPQs over the database, and thus in terms that are understandable to the user.

More generally, if $\mathcal{V}$ is lossy with respect to Q and inclusion (2) is a proper inclusion, we would like to provide an explanation for the answers that are actually returned or, equivalently, for the loss of information. Indeed, in this case, we know that there will be at least one view extension such that, in order to show that a tuple is not a certain answer, we need to resort to a non-linear database. It remains to be investigated whether the techniques we provide for doing the check allow one also to extract such a counterexample database to exhibit to the user.

References

1. S. Abiteboul. Querying semi-structured data. In *Proc. of ICDT'97*, pages 1–18, 1997.
2. S. Abiteboul, P. Buneman, and D. Suciu. *Data on the Web: from Relations to Semistructured Data and XML*. Morgan Kaufmann, 2000.
3. S. Abiteboul and O. Duschka. Complexity of answering queries using materialized views. In *Proc. of PODS'98*, pages 254–265, 1998.
4. S. Abiteboul, D. Quass, J. McHugh, J. Widom, and J. L. Wiener. The Lorel query language for semistructured data. *Int. J. on Digital Libraries*, 1(1):68–88, 1997.
5. F. N. Afrati, C. Li, and P. Mitra. Answering queries using views with arithmetic comparisons. In *Proc. of PODS 2002*, pages 209–220, 2002.
6. P. Buneman. Semistructured data. In *Proc. of PODS'97*, pages 117–121, 1997.
7. P. Buneman, S. Davidson, G. Hillebrand, and D. Suciu. A query language and optimization technique for unstructured data. In *Proc. of ACM SIGMOD*, pages 505–516, 1996.
8. D. Calvanese, G. De Giacomo, M. Lenzerini, and M. Y. Vardi. Answering regular path queries using views. In *Proc. of ICDE 2000*, pages 389–398, 2000.
9. D. Calvanese, G. De Giacomo, M. Lenzerini, and M. Y. Vardi. Query processing using views for regular path queries with inverse. In *Proc. of PODS 2000*, pages 58–66, 2000.
10. D. Calvanese, G. De Giacomo, M. Lenzerini, and M. Y. Vardi. View-based query processing and constraint satisfaction. In *Proc. of LICS 2000*, pages 361–371, 2000.
11. D. Calvanese, G. De Giacomo, M. Lenzerini, and M. Y. Vardi. Lossless regular views. In *Proc. of PODS 2002*, pages 58–66, 2002.
12. D. Calvanese, G. De Giacomo, M. Lenzerini, and M. Y. Vardi. Rewriting of regular expressions and regular path queries. *J. of Computer and System Sciences*, 64(3):443–465, 2002.
13. D. Calvanese, G. De Giacomo, M. Lenzerini, and M. Y. Vardi. Reasoning on regular path queries. *SIGMOD Record*, 32(4):83–92, 2003.
14. D. Calvanese, G. De Giacomo, M. Lenzerini, and M. Y. Vardi. View-based query containment. In *Proc. of PODS 2003*, pages 56–67, 2003.

15. R. Chirkova, A. Y. Halevy, and D. Suciu. A formal perspective on the view selection problem. In *Proc. of VLDB 2001*, pages 59–68, 2001.
16. J. Clark and S. DeRose. XML Path Language (XPath) version 1.0 – W3C recommendation 16 november 1999. Technical report, World Wide Web Consortium, 1999. Available at `http://www.w3.org/TR/1999/REC-xpath-19991116`.
17. A. Deutsch, M. F. Fernandez, D. Florescu, A. Levy, and D. Suciu. XML-QL: A query language for XML. Submission to the World Wide Web Consortium, Aug. 1998. Available at `http://www.w3.org/TR/NOTE-xml-ql`.
18. O. M. Duschka and M. R. Genesereth. Answering recursive queries using views. In *Proc. of PODS'97*, pages 109–116, 1997.
19. T. Feder and M. Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction. *SIAM J. on Computing*, 28:57–104, 1999.
20. M. F. Fernandez, D. Florescu, J. Kang, A. Y. Levy, and D. Suciu. Catching the boat with Strudel: Experiences with a web-site management system. In *Proc. of ACM SIGMOD*, pages 414–425, 1998.
21. S. Flesca and S. Greco. Rewriting queries using views. *IEEE Trans. on Knowledge and Data Engineering*, 13(6):980–995, 2001.
22. D. Florescu, A. Levy, and A. Mendelzon. Database techniques for the World-Wide Web: A survey. *SIGMOD Record*, 27(3):59–74, 1998.
23. G. Grahne and A. O. Mendelzon. Tableau techniques for querying information sources through global schemas. In *Proc. of ICDT'99*, volume 1540 of *LNCS*, pages 332–347. Springer, 1999.
24. S. Grumbach and L. Tininini. On the content of materialized aggregate views. In *Proc. of PODS 2000*, pages 47–57, 2000.
25. A. Y. Halevy. Answering queries using views: A survey. *VLDB Journal*, 10(4):270–294, 2001.
26. J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison Wesley Publ. Co., 1979.
27. A. Y. Levy, A. O. Mendelzon, Y. Sagiv, and D. Srivastava. Answering queries using views. In *Proc. of PODS'95*, pages 95–104, 1995.
28. C. Li, M. Bawa, and J. D. Ullman. Minimizing view sets without losing query-answering power. In *Proc. of ICDT 2001*, pages 99–113, 2001.
29. T. Milo and D. Suciu. Index structures for path expressions. In *Proc. of ICDT'99*, volume 1540 of *LNCS*, pages 277–295. Springer, 1999.
30. R. Reiter. On closed world data bases. In H. Gallaire and J. Minker, editors, *Logic and Databases*, pages 119–140. Plenum Publ. Co., 1978.
31. J. D. Ullman. Information integration using logical views. In *Proc. of ICDT'97*, volume 1186 of *LNCS*, pages 19–40. Springer, 1997.
32. M. Y. Vardi. A temporal fixpoint calculus. In *Proc. of POPL'88*, pages 250–259, San Diego (CA, USA), 1988.
33. M. Y. Vardi. A note on the reduction of two-way automata to one-way automata. *Information Processing Letters*, 30(5):261–264, 1989.

A Appendix

Theorem 2. *Let Q be a 2RPQ and $\mathcal{V}$ a set of 2RPQ views. Then A^{rew} is the 2RPQ-maximal rewriting of Q with respect to $\mathcal{V}$.*

Proof. We show that a word w over $\mathcal{V}^{\pm}$ is in the 2RPQ-maximal rewriting of Q with respect to $\mathcal{V}$ if and only if $w \notin L(A^{nr})$.

"$\Rightarrow$" Let $w = v_1 \cdots v_k$ be a word over $\mathcal{V}^{\pm}$ such that $w \in L(A^{nr})$. We construct a (linear) view extension $\mathcal{E}_w$ as follows: we introduce pairwise disjoint objects $c = a_0, a_1, \ldots, a_k = d$, and for each $i \in \{1, \ldots, k\}$,

- $(a_{i-1}, a_i) \in V(\mathcal{E}_w)$, if $v_i = V$, and
- $(a_i, a_{i-1}) \in V(\mathcal{E}_w)$, if $v_i = V^-$.

Consider now the constraint instance $I = \mathcal{E}_w^{c,d}$ obtained from $\mathcal{E}_w$ by defining $U_i(I) = \{c\}$ and $U_f(I) = \{d\}$. It is immediate to verify that, by construction, there is a homomorphism from I to the constraint template $CT_{Q,\mathcal{V}}$. Hence, by Theorem 1, we have that $(c, d) \notin cert_{Q,\mathcal{V}}(\mathcal{E}_w)$, while obviously $(c, d) \in w(\mathcal{E}_w)$. Thus w is not part of a rewriting.

"$\Leftarrow$" Let $w = v_1 \cdots v_k$ be a word over $\mathcal{V}^{\pm}$ that is not part of any rewriting. By Lemma 1, there exists a *linear* database $\mathcal{B}$ containing objects c and d, and a view extension $\mathcal{E}$ with $\mathcal{E} \subseteq \mathcal{V}^{\Sigma}(\mathcal{B})$, such that $(c, d) \in w(\mathcal{E})$ but $(c, d) \notin Q(\mathcal{B})$. Since $(c, d) \in w(\mathcal{E})$ and $\mathcal{E} \subseteq \mathcal{V}^{\Sigma}(\mathcal{B})$, there are objects $c = a_0, a_1, \ldots, a_k = d$ in $\mathcal{B}$ and a word $p = q_1 \cdots q_k$, with each q_i a word over $\Sigma^{\pm}$, such that for each $i \in \{1, \ldots, k\}$, we have that $(a_{i-1}, a_i) \in q_i(\mathcal{B})$, and

- $q_i \in L(V^{\Sigma})$, if $v_i = V$, and
- $q_i \in L((V^{\Sigma})^-)$, if $v_i = V^-$.

We construct from $\mathcal{B}$ a constraint instance $I = \mathcal{E}^{c,d}$ as follows: for each $i \in \{1, \ldots, k\}$,

- $(a_{i-1}, a_i) \in V(I)$, if $v_i = V$, and
- $(a_i, a_{i-1}) \in V(I)$, if $v_i = V^-$,

and $U_i(I) = \{c\}$ and $U_f(I) = \{d\}$. Since $(c, d) \notin Q(\mathcal{B})$, and $\mathcal{E} \subseteq \mathcal{V}(\mathcal{B})$, we also have that $(c, d) \notin cert_{Q,\mathcal{V}}(\mathcal{E})$. Hence there is a homomorphism h from I to $CT_{Q,\mathcal{V}}$, and therefore, for each $i \in \{1, \ldots, k\}$,

- $(h(a_{i-1}), h(a_i)) \in V(CT_{Q,\mathcal{V}})$, if $v_i = V$, and
- $(h(a_i), h(a_{i-1})) \in V(CT_{Q,\mathcal{V}})$, if $v_i = V^-$.

In any case, this means that there are transitions $(h(a_{i-1}), v_i, h(a_i)) \in \varrho^{nr}$ that lead from an initial to a final state of A^{nr}. Hence, $w = v_1 \cdots v_k$ is accepted by A^{nr}. □

Lemma 2. *Let Q be a 2RPQ and $\mathcal{V}$ be a set of 2RPQ views. Then the 2RPQ-maximal rewriting of Q with respect to $\mathcal{V}$ is perfect if and only if for every proper constraint template $CTR^{\alpha,\beta}_{A^{rew},\mathcal{V}}$ of $CTR_{A^{rew},\mathcal{V}}$, there exists a homomorphism from $CTR_{A^{rew},\mathcal{V}}\alpha, \beta$ to $CT_{Q,\mathcal{V}}$.*

Proof. Consider now a view extension $\mathcal{E} = \mathcal{V}^{\Sigma}(\mathcal{B})$ for some database $\mathcal{B}$, and a pair (c, d) that is not in the answer set $A^{rew}(\mathcal{E})$. This means that every semipath $x_0 v1 x_1 \cdots x_n v_n$ from $c = x_0$ to $d = x_n$ in $\mathcal{E}$ can be instantiated by words $w_i \in L(v_i)$ such Q, when evaluated over the linear database $x_0 w_1 x_1 \cdots x_n w_n$, does not accept (c, d). Thus, we can label the objects $x_0, \ldots, x_n$ on the path from c to d in the view extension by sets of states of A^{nr} (recall that A^{nr} is the 1NFA from which we have obtained A^{rew} by complementation) such that the label of c contains the start set and the label of d consists of accepting states of A^{nr}. This holds for all paths. Thus, we should be able to label all the object in the view extension $\mathcal{E}$ by sets of states of A^{nr} such that the label of c contains the start set, the label of d consists of accepting states of A^{nr}, and the edges respect the transition relation of A. In summary, we have mapped the view extension $\mathcal{E}$ into the rewriting constraint template of A^{rew}. Thus, non-membership in the answerset $A^{rew}(\mathcal{E})$ is characterized by homomorphism from $\mathcal{E}^{c,d}$ into the rewriting constraint template of A^{rew}. Now, non-membership in $cert_{Q,\mathcal{V}}(\mathcal{E})$ is also characterized by homomorphism from $\mathcal{E}^{c,d}$ to $CT_{Q,\mathcal{V}}$. Hence, by reasoning as in the proof of Theorem 18 in [14], we get the claim. $\square$

Theorem 3. *Let Q be a 2RPQ and $\mathcal{V}$ be a set of 2RPQ views. Then checking whether the 2RPQ-maximal rewriting of Q with respect to $\mathcal{V}$ is perfect can be done in N2EXPTIME in the size of Q and in NEXPTIME in the size of $\mathcal{V}^{\Sigma}$*

Proof. We have to check whether, for every proper constraint template $CTR^{\alpha,\beta}_{A^{rew},\mathcal{V}}$ of $CTR_{A^{rew},\mathcal{V}}$ there exists a homomorphism to $CT_{Q,\mathcal{V}}$. Constructing the constraint template $CT_{Q,\mathcal{V}}$ is in EXPTIME in the size of Q and polynomial in the size of $\mathcal{V}^{\Sigma}$. Constructing the 1NFA A^{nr} is in EXPTIME in the size of Q and polynomial in the size of $\mathcal{V}^{\Sigma}$. Hence, constructing the rewriting constraint template $CTR_{A^{rew},\mathcal{V}}$ of A^{rew} and each proper constraint template $CTR^{\alpha,\beta}_{A^{rew},\mathcal{V}}$ is in 2EXPTIME in the size of Q and in EXPTIME in the size of $\mathcal{V}^{\Sigma}$. Checking the existence of each homomorphism is NP in the size of the $CTR^{\alpha,\beta}_{A^{rew},\mathcal{V}}$. Moreover, the number of proper constraint templates is $2 \cdot 2^k$, where k is the number of states of A^{nr}, and hence it is doubly exponential in the size of Q. These bounds give us the claim. $\square$

Lemma 3. *Let Q' be two-way path query. Then, if there is a database $\mathcal{B}$ and a pair of objects (c, d) in $\mathcal{B}$ such that $(c, d) \in Q'(\mathcal{B})$ and $(c, d) \notin cert_{Q,\mathcal{V}}(\mathcal{V}^{\Sigma})(\mathcal{B})$, then there is a linear database $\mathcal{B}_\ell$ and a pair of objects (c', d') in $\mathcal{B}_\ell$ such that $(c', d') \in Q'(\mathcal{B}_\ell)$ and $(c', d') \notin cert_{Q,\mathcal{V}}(\mathcal{V}^{\Sigma})(\mathcal{B}_ell)$.*

Proof. The proof is analogous to the proof of Theorem 9 in [11]

Lemma 4. *Let Q be a 2RPQ and $\mathcal{V}$ be a set of 2RPQ views. Then we can construct in double exponential time in both Q and $\mathcal{V}^{\Sigma}$ two 1NFAs A^{nlin} and A^{lin} such that:*

– *A^{nlin} accepts all words $w \in \Sigma^{\pm *}$ such that $(a, b) \notin cert_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B}^{a,b}_w))$.*

– A^{lin} *accepts all words* $w \in \Sigma^{\pm *}$ *such that* $(a,b) \in cert_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B}_w^{a,b}))$.

Proof. We construct a 1DFA $A^{\mathcal{V}}$ over the alphabet $\Sigma^{\pm} \cup \Lambda$, with $\Lambda = 2^{S^Q}$, as follows:

– We construct a 1DFA A^0 accepting words that
  - start with a symbol $\ell \in \Lambda$ such that $S_0^Q \subseteq \ell$, and
  - end with a symbol $\ell \in \Lambda$ such that $\ell \cap F^Q = \emptyset$.

  These conditions can be checked by A^0 in a straightforward manner using a constant number of states.
– For each view $V \in \mathcal{V}$, we construct a 1NFA A^V as follows. We first define a binary relation V^r on Λ as follows: $(\ell, \ell') \in V^r$ if there exists a word $q_1 \cdots q_k \in L(V^{\Sigma})$ and a sequence $T_0, \ldots, T_k$ of subsets of S^Q such that the following hold:
  1. $T_0 = \ell$ and $T_k = \ell'$,
  2. if $s \in T_i$ and $(s, q_{i+1}, t) \in \varrho^Q$ then $t \in T_{i+1}$, for $0 \leq i < k$, and
  3. if $s \in T_i$ and $(s, q_i^-, t) \in \varrho^Q$ then $t \in T_{i-1}$, for $0 < i \leq k$.

  Note that for each ℓ, ℓ' we can decide in PSPACE whether $(\ell, \ell') \in V^r$ holds[6], so all the relations V^r can be constructed in time exponential in the size of A^Q and linear in the size of $\mathcal{V}^{\Sigma}$. From the relation V^r we can immediately construct a 1NFA that accepts a word starting with ℓ and ending with ℓ' if and only if $(\ell, \ell') \notin V^r$.

  We then construct the 1NFA A^V in such a way that it reads a word w, guesses positions j and h, checks that $(\ell_j, \ell_h) \notin V^r$, and checks that $(\ell_j, \ell_h) \in V^{\Sigma}(\mathcal{B}_w)$, where $\mathcal{B}_w$ is a (linear) database obtained from w by considering each symbol ℓ_i as an object. To do so, we first construct a 2NFA that reads a word w, guesses positions j and h, and checks that $(\ell_j, \ell_h) \in V^{\Sigma}(\mathcal{B}_w)$. Note that the 2NFA does not need to consider the actual symbols ℓ_i appearing in w, but only the symbols q_i, hence it has a number of states that is polynomial in V^{Σ} (and does not depend on Q). Then we transform the 2NFA into a 1NFA and modify it in the following way to obtain A^V: when the 1NFA guesses the position j it also remembers ℓ_j and, when it guesses position h, it checks that $(\ell_j, \ell_h) \notin V^r$. The number of states of the 1NFA derived from the 2NFA is exponential, and the necessity to remember ℓ_j along its computation multiplies such a number by at most another exponential in Q. Hence, the overall number of states of A^V is exponential both in Q and in V.
– Finally, we take the union of the automata A^V, for each $V \in \mathcal{V}$, complement it, and intersect with A^0, thus getting $A^{\mathcal{V}}$. Note that such an automaton is deterministic, and has a number of states that is doubly exponential in the size of Q and $\mathcal{V}^{\Sigma}$.

[6] The condition can be verified by checking the nonemptiness of a 1NFA built as in [33]. The 1NFA has an exponential number of states, however nonemptiness can be checked on the fly, without actually building the 1NFA.

Since we are interested in words that consist of an alternation of symbols in Λ and symbols in $\Sigma^{\pm}$, we also construct a 1DFA A^{alt} accepting such words. The number of states of A^{alt} is constant.

Now, using $A^{\mathcal{V}}$ and A^{alt} we can characterize both the linear fragment of the certain answers and its complement.

- By intersecting $A^{\mathcal{V}}$ with A^{alt}, and projecting away the symbols ℓ_i, we obtain an 1NFA A^{nlin}.
- Similarly, by intersecting the *complement*[7] of $A^{\mathcal{V}}$ with A^{alt}, and projecting away the symbols ℓ_i, we obtain an 1NFA A^{lin}.

Finally, we prove now that A^{nlin} accepts a word $w \in \Sigma^{\pm *}$ if and only if $(a, b) \notin cert_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B}_w^{a,b}))$. The proof for A^{lin} is analogous. "$\Leftarrow$" Let $w = q_1 q_2 \cdots q_m$ be a word such that $(x_0, x_m) \notin cert_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B}_w^{x_0,x_m}))$, and let $\mathcal{B}_w^{x_0,x_m}$ be $(x_0, q_1, x_1, q_2, \ldots, q_m, x_m)$. By Theorem 1 there exists a labeling $\ell(\cdot)$ of the objects $x_0, \ldots, x_m$ with states of A^Q such that the CT-conditions hold. Now consider the word $\ell(x_0) q_1 \ell(x_1) q_2 \cdots q_m \ell(x_m)$. One can check that, by construction, such a word is accepted by $A^{\mathcal{V}}$ and A^{alt}. Hence, w is accepted by A^{nlin}.

"$\Rightarrow$" If A^{nlin} accepts a word $w = q_1 q_2 \cdots q_m$, then there exists a word $\ell_0 q_1 \ell_1 q_2 \cdots q_m \ell_m$ accepted by $A^{\mathcal{V}}$ and A^{alt}. Now, consider the linear database $\mathcal{B}$ of the form $(x_0, q_1, x_1, q_2, \ldots, q_m, x_m)$ and a labeling $\ell(\cdot)$ of the objects $x_0, \ldots, x_m$ with states of A^Q defined by $\ell(x_i) = \ell_i$, for $i \in \{0, \ldots, m\}$. Since $\ell_0 q_1 \ell_1 q_2 \cdots q_m \ell_m$ is accepted by $A^{\mathcal{V}}$, the CT-conditions hold and hence, by Theorem 1 $(x_0, x_m) \notin cert_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B}))$. □

Theorem 5. *Let Q be a 2RPQ and $\mathcal{V}$ be a set of 2RPQ views. Then checking whether $\mathcal{V}$ is lossless with respect to Q can be done in EXPSPACE in the size of Q and $\mathcal{V}^{\Sigma}$.*

Proof. By Theorem 4, the linear fragment $clin_{Q,\mathcal{V}}$ of the certain answers is given by A^{lin}. Its complement A^{nlin} is an 1NFA with a number of states that is doubly exponential in Q and $\mathcal{V}^{\Sigma}$. Recall that for a word $w \in \Sigma^{\pm *}$, we denote with $\mathcal{B}_w^{a,b}$ the linear database constituted by a path from a to b spelled by w (with arbitrary intermediate nodes), and that A^{nlin} accepts all words w such that $(a, b) \notin cert_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B}_w^{a,b}))$. To check whether Q is not contained in $clin_{Q,\mathcal{V}}$, it suffices to check whether there is a word $w \in \Sigma^{\pm *}$ such that $(a, b) \in Q(\mathcal{B}_w^{a,b})$ and $w \in L(A^{nlin})$. By Lemma 2 in [13], it suffices to check whether there is a word $w \in L(Q) \cap L(A^{nlin})$, i.e., whether the intersection of Q and A^{nlin} is nonempty. A^{nlin} has a number of states that is double exponential in Q and $\mathcal{V}$. Considering that emptiness of 1NFAs is NLOGSPACE in the number of states and that the construction of A^{nlin} can be done on the fly while checking for emptiness, we get the claim.

[7] Observe that $A^{\mathcal{V}}$ is deterministic, so complementing simply amounts to swapping final and non-final states.