Lossless Regular Views

Diego Calvanese
Giuseppe De Giacomo
Maurizio Lenzerini
Dipartimento di Informatica e Sistemistica
Università di Roma "La Sapienza"
Via Salaria 113, I-00198 Roma, Italy
lastname@dis.uniroma1.it

Moshe Y. Vardi
Department of Computer Science
Rice University, P.O. Box 1892
Houston, TX 77251-1892, U.S.A.
vardi@cs.rice.edu

ABSTRACT

If the only information we have on a certain database is through a set of views, the question arises of whether this is sufficient to answer completely a given query. We say that the set of views is lossless with respect to the query, if, no matter what the database is, we can answer the query by solely relying on the content of the views. The question of losslessness has various applications, for example in query optimization, mobile computing, data warehousing, and data integration. We study this problem in a context where the database is semistructured, and both the query and the views are expressed as regular path queries. The form of recursion present in this class prevents us from applying known results to our case.

We first address the problem of checking losslessness in the case where the views are materialized. The fact that we have the view extensions available makes this case solvable by extending known techniques. We then study a more complex version of the problem, namely the one where we abstract from the specific view extension. More precisely, we address the problem of checking whether, for every database, the answer to the query over such a database can be obtained by relying only on the view extensions. We show that the problem is solvable by utilizing, via automata-theoretic techniques, the known connection between view-based query answering and constraint satisfaction. We also investigate the computational complexity of both versions of the problem.

1. INTRODUCTION

View-based query processing is the problem of computing the answer to a query based on a set of views [16, 22, 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 [15] for a recent survey). Some of the methods are based on the idea of first computing the rewriting of the query with respect to the views, and then evaluating the rewriting over the view extensions (see, for instance, [16]). Other methods, such as [10], use a more direct approach, trying to compute the so-called certain answers, 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. A comparison of the two approaches is reported in [8].

We observe that there are at least two notions of certain answers proposed in the literature, depending on the interpretation of a database being coherent with the views (e.g., [3, 13, 6])[1].

1. One interpretation assumes that the views are sound but not complete. A database is consistent with the views under the sound view assumption if the extension of each view is a subset of the result of evaluating the corresponding view expression over that database. In other words, we assume that views provide only some of the data that are obtainable by the query expressions associated to the views.
2. On the contrary, the other interpretation, called exact view assumption, considers the views to be both sound and complete. A database is consistent with the views under the exact view assumption if the extension of each view coincides with the result of evaluating the corresponding view expression over that database.

Independently from the strategy and the assumption adopted, the question arises of 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 question is relevant for example in mobile computing, where we may be interested in checking whether a set of cached data allows us to derive the requested information without accessing the network. Similarly, in data warehousing, determining if the materialized views are sufficient to

[1] In fact, also a third assumption, called *complete* view assumption, has been proposed in the literature, but we do not consider it here.

In Proceedings of the 21st ACM SIGACT SIGMOD SIGART Symposium on Principles of Database Systems (**PODS 2002**).

completely answer a query is a relevant question, both positively for answering the query if no other data are available, and negatively, for ensuring that the user does not derive sensitive information [14]. Also, the question is important at design time, in particular for the view selection problem [9], where we have to measure the quality of the choice of the views to materialize in the data warehouse. Finally, in data integration, we may be interested in checking whether the relevant queries can be answered by accessing only a given set of sources. As pointed out in [17], this may help in the design of the data integration system, in particular, by selecting a minimal subset of sources to access without losing query-answering power.

The definition of losslessness relies on that of certain answers: a set of views is *lossless* with respect to a query, if for every database, we can answer the query over that database by computing the certain answers based on the view extensions. It follows that there are two versions of losslessness, namely, losslessness under the sound view assumption, and losslessness under the exact view assumption.

1. The first version is obviuosly weaker than the second one. If views $\mathcal{V}$ are lossless with respect to a query Q under the sound view assumption, then we know that, from the intensional point of views, $\mathcal{V}$ contain enough information to completely answer Q, even though the possible incompleteness of the view extensions may prevent us form obtaining all the answers that Q would get from the database.

2. On the other hand, if $\mathcal{V}$ are lossless with respect to a query Q under the exact view assumption, then we know that they contain enough information to completely answer Q, both from the intensional and from the extensional point of view.

Recent papers address, either explicitly or implicitly, the question of losslessness. Interestingly, all of the results they present refer to the case of losslessness under the sound view assumption.

In [14], the authors introduce the notion of "information content" of materialized views, and exploit this notion to derive results on losslessness for a restricted class of aggregate queries. In [19], the problem of "relative containment" is studied for variants of conjunctive queries. A query is contained in another query relative to a set of views, if, for each extension of the views, the certain answers to the former query are a subset of the certain answers to the latter. It is shown that relative containment is Π_2^P complete in the case of conjunctive queries and views. In [18], the results on relative containment are extended to the case where views have limited access patterns. In [17], several results are presented on the notion of "p-containment". A view set V is said to be p-contained in another view set W, i.e., V has at most the answering power of W, if W can answer all queries that can be answered using V. Most results in [19, 17] are based on the fact that, in the case of conjunctive queries and views, the set of certain answers to a given query can be obtained by a nonrecursive Datalog program. This implies, for example, that checking relative containment amounts to checking containment of two nonrecursive Datalog programs. By exploiting this fact, we can conclude that checking losslessness of a set of conjunctive views with respect to a conjunctive query can be done by checking equivalence between the query and the nonrecursive Datalog program computing the corresponding certain answers.

Related to the question of losslessness, is the problem of checking whether the maximally contained rewriting of a query with respect to a set of views is exact, i.e., is equivalent to the query. In [5, 7], methods for checking exactness of regular rewritings are reported for the classes of regular path queries, and regular path queries with inverse. Note, however, that regular rewritings may not be sufficient for computing all certain answers [8], and therefore, checking losslessness of a set of regular path views with respect to a regular path query cannot be reduced to checking the exactness of the rewriting of the query with respect to the views.

In this paper, we study the problem of losslessness in a context where the database is semistructured [2], i.e., has the form a labeled graph, and both the query and the views are expressed as regular path queries. The presence of recursion in regular path queries, although of a limited form, prevents us from applying known results to our case. In particular, all the known results on losslessness checking are valid for settings where the problem can be reduced to checking the equivalence of the query with respect to a certain query expression representing its maximally contained rewriting. As we said before, we cannot directly apply this method in our case.

The main contributions of our work are as follows.

1. First, we address the problem of checking losslessness in the case where the views are materialized, i.e., we have view extensions available. This version of losslessness is particularly relevant in data integration, where views model data sources, and view extensions correspond to the data accessible to the integration system for answering queries. In this case, losslessness should be determined relative to the available data. We present a method that searches for a counterexample to losslessness, i.e., two databases that are both coherent with the view extensions, and that differ in the answers to the query. We show that the availability of view extensions makes this case solvable by extending known techniques.

2. We then study the more general version of the problem, namely the one where we abstract from the specific view extension. As we said before, the losslessness question in this case consists of checking whether, for every database, the answer to the query over such a database can be obtained by computing the certain answers only on the basis of the view extensions. This version of losslessness is the one needed in those situations where only intensional information is available, e.g., at design time of a data warehouse, and in those situations where view extensions are expected to change frequently, so that measuring losslessness on the current view extension only is meaningless. The

search for a counterexample is much more difficult in this case, due to the presence of a quantification over all possible view extensions. Nevertheless, we show that the problem is solvable even in this case, at least under the sound view assumption. In particular, we prove that, under such an assumption, we can restrict our attention to counterexamples that are linear databases, and this allows us to devise a method that uses, via automata-theoretic techniques, the known connection between view-based query answering and constraint satisfaction [8].

We also present a comparison with the case of losslessness under the exact view assumption, for which we have no algorithm at present. Moreover, we show that if a query Q is equivalent to its maximally contained regular rewriting with respect to views $\mathcal{V}$, then $\mathcal{V}$ is lossless with respect to Q under the sound view assumption.

3. Finally, we study the computational complexity of checking losslessness of regular views under the sound views assumption. In particular, we show that, for a fixed view extension, checking losslessness is coNP-complete in data complexity (i.e., with respect to the size of view extensions), and PSPACE-complete wrt the query and the view definitions. In the general case, we show that the problem is PSPACE-complete wrt the view definitions, and we provide an EXPSPACE upper bound with respect to the query.

The paper is organized as follows. Section 2 introduces regular path queries, and provides the definition of losslessness of views. Section 4.1 recalls the connection between view-based query answering and constraint satisfaction [8]. Section 3 presents the method for checking losslessness with respect to a view extension. Section 4 and Section 4.3 address the general version of the problem, for the case of sound views and exact views, respectively. Section 5 presents a discussion on the comparison between the notion of losslessness and the notion of equivalence with the maximally contained rewriting. Fianlly, Section 6 concludes the paper.

2. PRELIMINARIES

A *(semi-structured) database* is a graph whose nodes represent objects, and whose edges are labeled by elements from a given alphabet Σ, which we assume to be finite [4, 1]. Different nodes represent different objects, and an edge from node x to node y labeled by r, denoted (x, r, y), represents the fact that relation r holds between the object represented by x and the object represented by y.

In this paper, we restrict our attention to *regular-path queries* (RPQs). An RPQ Q is expressed as a regular expression or a finite-state automaton over Σ. When evaluated over a database $\mathcal{B}$, an RPQ Q computes the set $Q(\mathcal{B})$ of pairs of nodes connected by a path that conforms to the regular language $L(Q)$ defined by Q. A *path* in $\mathcal{B}$ from x to y (labeled with $r_1 \cdots r_m$) is a sequence of the form

$$(y_0, r_1, y_1, \ldots, y_{m-1}, r_m, y_m)$$

where

- $m \geq 0$, $y_0 = x$, $y_m = y$, and
- for each $i \in \{1, \ldots, m\}$, we have that $r_i \in \Sigma$ and (y_{i-1}, r_i, y_i) in $\mathcal{B}$.

We say that a path $(y_0, r_1, \ldots, r_m, y_m)$ *conforms to* Q if

$$r_1 \cdots r_m \in L(Q).$$

A path is said to be *simple* if no node appears more than once in the corresponding sequence.

Consider a database that is accessible only through a collection of views $\mathcal{V}$, and suppose we want to answer an RPQ only on the basis of our knowledge on the views. Specifically, the collection of views is represented by a tuple $\mathcal{V} = (V_1, \ldots, V_k)$ of *view definitions* and a tuple $\mathcal{E} = (E_1, \ldots, E_k)$ of corresponding *view extensions*. For each i, the i-th view is characterized by the pair (V_i, E_i), where

- view definition V_i is an RPQ over the alphabet Σ;
- view extension E_i is a set of pairs of objects. We assume that objects are represented by constants and we adopt the *unique name assumption* [20], i.e., different constants denote different objects, and therefore different nodes.

We denote by $\Delta_{\mathcal{E}}$ the set of objects appearing in $\mathcal{E}$. Also for a database $\mathcal{B}$, we use $\mathcal{V}(\mathcal{B})$ to denote the view extensions $(V_1(\mathcal{B}), \ldots, V_k(\mathcal{B}))$, and for two databases $\mathcal{B}$ and $\mathcal{B}'$, we use $\mathcal{V}(\mathcal{B}) \subseteq \mathcal{V}(\mathcal{B}')$ to denote that $V_i(\mathcal{B}) \subseteq V_i(\mathcal{B}')$, for $i \in \{1, \ldots, k\}$.

We say that a database $\mathcal{B}$ is *consistent with V_i and E_i under the sound view assumption* if $E_i \subseteq V_i(\mathcal{B})$. A database $\mathcal{B}$ is *consistent with $\mathcal{V}$ and $\mathcal{E}$ under the sound view assumption* if $\mathcal{B}$ is consistent with V_i and E_i under the same assumption, for each $i \in \{1, \ldots, k\}$.

Note that the definition makes it clear that the sound view assumption models a situation where the view extensions provide a subset of the results of applying the view expressions to the database. For a sound view (V, E), all the tuples in E must appear in $V(\mathcal{B})$, but $V(\mathcal{B})$ may contain tuples not in E.

On the other hand, the exact view assumption models a situation where the view extensions provide sound and complete information about which data satisfies the view expressions over the given database. It follows that query answering based on exact views means basically applying the closed world assumption on the views [3]. We say that a database $\mathcal{B}$ is *consistent with V_i and E_i under the exact view assumption* if $E_i = V_i(\mathcal{B})$. A database $\mathcal{B}$ is *consistent with $\mathcal{V}$ and $\mathcal{E}$ under the exact view assumption* if $\mathcal{B}$ is consistent with V_i and E_i under the same assumption, for each $i \in \{1, \ldots, k\}$.

Given view definitions $\mathcal{V}$, view extensions $\mathcal{E}$, and a query Q, the set of *certain answers* to Q under either the sound or the exact view assumption with respect to $\mathcal{V}$ and $\mathcal{E}$ is the set of pairs (c, d) such that $(c, d) \in Q(\mathcal{B})$, for every $\mathcal{B}$ that is consistent with $\mathcal{V}$ and $\mathcal{E}$ under the same assumption. *View-based query answering* consists in deciding whether a given pair (c,d) of objects is a certain answer to Q with respect to $\mathcal{V}$ and $\mathcal{E}$.

Given view definitions $\mathcal{V}$ and a query Q, we denote by $cert^{sound}_{Q,\mathcal{V}}(\cdot)$ ($cert^{exact}_{Q,\mathcal{V}}(\cdot)$, resp.) the function that, for every view extension $\mathcal{E}$ for $\mathcal{V}$, returns the set of certain answers to Q under the sound view assumption (exact view assumption, resp.) with respect to $\mathcal{V}$ and $\mathcal{E}$.

We now define the notion of losslessness of a collection of views with respect to a query. We first refer to the case where we have the view extensions available.

DEFINITION 1. *Given view definitions $\mathcal{V}$, corresponding view extensions $\mathcal{E}$, and a query Q, $\mathcal{V}$ is said to be lossless* with respect to Q relative to $\mathcal{E}$ under the sound view assumption (exact view assumption, resp.)*, if, for every database $\mathcal{B}$ such that $\mathcal{V}(\mathcal{B}) = \mathcal{E}$, we have that $Q(\mathcal{B}) = cert^{sound}_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B}))$ ($Q(\mathcal{B}) = cert^{exact}_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B}))$, resp.).*

EXAMPLE 1. Consider the alphabet $\Sigma = \{0,1\}$, the query

$$Q = 1\cdot 1$$

the views $\mathcal{V} = (V_1, V_2, V_3)$ with

$$\begin{array}{rcl} V_1 & = & 0+1 \\ V_2 & = & 0 \\ V_3 & = & (0+1)^* \end{array}$$

and the extension $\mathcal{E} = (E_1, E_2, E_3)$ with

$$\begin{array}{rcl} E_1 & = & \{(a,b),(b,c)\} \\ E_2 & = & \emptyset \\ E_3 & = & \{(a,b),(a,c),(b,c)\} \end{array}$$

Then $\mathcal{V}$ is *not* lossless with respect to Q relative to $\mathcal{E}$ under the sound view assumption. Consider for example the database $\mathcal{B}$ constituted by the single path $(a,1,b,1,c)$, for which $\mathcal{V}(\mathcal{B}) = \mathcal{E}$. Clearly, $Q(\mathcal{B}) = \{(a,c)\}$, but it is easy to verify that $(a,c) \notin cert^{sound}_{Q,\mathcal{V}}(\mathcal{E})$, take e.g., the database $\mathcal{B}' = (a,0,b,1,c)$.

On the other hand, one can check that $\mathcal{V}$ *is* lossless with respect to Q relative to $\mathcal{E}$ under the exact view assumption. Intuitively, this is due the fact that $E_2 = \emptyset$, and since the views are exact, in each database $\mathcal{B}$ consistent with $\mathcal{V}$ and $\mathcal{E}$, the pairs of nodes (y_1,y_2) such that $(y_1,1,y_2)$ are in $\mathcal{B}$, are precisely those in E_1. ∎

Finally, we extend this notion to the case where we abstract from view extensions. Intuitively, in this case losslessness of $\mathcal{V}$ with respect to Q means that, no matter what the database is, there is no difference between issuing Q to such a database and computing the certain answers to Q on the basis of the data satisfying the views on the database.

DEFINITION 2. *Given view definitions $\mathcal{V}$ and a query Q, $\mathcal{V}$ is said to be lossless* with respect to Q under the sound view assumption (exact view assumption, resp.)*, if for every database $\mathcal{B}$ we have that $Q(\mathcal{B}) = cert^{sound}_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B}))$ ($Q(\mathcal{B}) = cert^{exact}_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B})$, resp.).*

EXAMPLE 2. Consider the alphabet $\Sigma = \{0,1\}$, the query

$$Q = 1\cdot(0+1)\cdot 1$$

the views $\mathcal{V} = (V_1, V_2)$ with

$$\begin{array}{rcl} V_1 & = & 1 \\ V_2 & = & 0\cdot 1 \end{array}$$

Then $\mathcal{V}$ is lossless with respect to Q both under the sound and under the exact view assumption. Intuitively, this is due to the fact that for each database $\mathcal{B}$, a path conforming to Q has either the form $(y_0,1,y_1,1,y_2,1,y_3)$ or the form $(y_0,1,y_1,0,y_2,1,y_3)$. In the first case, the path conforms to $V_1\cdot V_1\cdot V_1$, while in the second case it conforms to $V_1\cdot V_2$. Hence, for each database $\mathcal{B}'$ such that $\mathcal{V}(\mathcal{B}) \subseteq \mathcal{V}(\mathcal{B}')$, we have that $(y_0,y_3) \in Q(\mathcal{B}')$. Notice that, in this case, there is an equivalent rewriting of the query Q with respect to the views $\mathcal{V}$ (cf. Section 5). ∎

3. LOSSLESSNESS WITH RESPECT TO A VIEW EXTENSION

We address the problem of checking, given a query Q, view definitions $\mathcal{V}$, and corresponding view extensions $\mathcal{E}$, whether $\mathcal{V}$ are lossless with respect to Q relative to $\mathcal{E}$. We deal first with losslessness under sound views. Following the definition, we have that $\mathcal{V}$ is not lossless with respect to Q relative to $\mathcal{E}$ under the sound view assumption if and only if there exists a pair of objects c and d such that:

1. there exists a database $\mathcal{B}$ with $\mathcal{V}(\mathcal{B}) = \mathcal{E}$ and $(c,d) \in Q(\mathcal{B})$;
2. $(c,d) \notin cert^{sound}_{Q,\mathcal{V}}(\mathcal{E})$.

Condition (2) corresponds to checking whether (c,d) is not a certain answer to Q under the sound view assumption with respect to $\mathcal{V}$ and $\mathcal{E}$, while condition (1) corresponds to checking whether (c,d) is a *possible answer* to Q under the sound view assumption with respect to $\mathcal{V}$ and $\mathcal{E}$.

Using such a characterization, we devise an algorithm to determine losslessness relative to a given extension. In [6], techniques are given to check both kinds of conditions when both c and d are in $\Delta_\mathcal{E}$. Observe, however, that we need to deal also with the case where c and/or d are not in $\Delta_\mathcal{E}$. To this end, note that, if $(c,d) \notin \Delta_\mathcal{E} \times \Delta_\mathcal{E}$, then trivially $(c,d) \notin cert^{sound}_{Q,\mathcal{V}}(\mathcal{E})$, since there are databases in which c and d do not appear. Hence, for this case we need only to check condition (1). The basic idea to do so, is to include two new objects c_{new} and d_{new} in the extension without changing the extensions of the views in $\mathcal{V}$. These new objects stand as representatives for objects not in $\Delta_\mathcal{E}$.

The algorithm that checks whether there is no pair of objects c and d satisfying conditions (1) and (2) proceeds as follows:

- Iterate over all pairs of objects $(c,d) \in \Delta_\mathcal{E} \times \Delta_\mathcal{E}$, and for each such pair, check whether conditions (1) and (2) hold.
- Add a new object c_{new}, and a new view with extension $\{(c_{new}, c_{new})\}$ and with definition r_{new}, where r_{new}

is a new symbol not appearing in the alphabet Σ of the original views and the query. Iterate over all objects $d \in \Delta_{\mathcal{E}}$, and for each pair (c_{new}, d) and each pair (d, c_{new}), check whether condition (1) holds.

- Add a new pair of objects (c_{new}, d_{new}), and a new view with extension $\{(c_{new}, d_{new})\}$ and with definition r_{new}, where again $r_{new} \notin \Sigma$. Check whether condition (1) holds for (c_{new}, d_{new}).

The algorithm returns false if one of the above checks succeeds, true otherwise.

THEOREM 3. *$\mathcal{V}$ is lossless with respect to Q relative to $\mathcal{E}$ if and only if the algorithm above returns* true.

Each of the checks performed by the algorithm can be done in PSPACE in Q and $\mathcal{V}$ [6]. If we measure the complexity only with respect to view extensions $\mathcal{E}$ (i.e., we consider *data complexity*), then each of the checks can be done in NP in $\mathcal{E}$ [6]. Hence we get the following upper bounds.

THEOREM 4. *Checking losslessness of $\mathcal{V}$ with respect to Q relative to $\mathcal{E}$ under the sound view assumption can be done in PSPACE wrt Q and $\mathcal{V}$, and in coNP wrt $\mathcal{E}$.*

The above complexity bounds are indeed tight.

THEOREM 5. *Checking losslessness of $\mathcal{V}$ with respect to Q relative to $\mathcal{E}$ under the sound view assumption is PSPACE-complete wrt Q, PSPACE-complete wrt $\mathcal{V}$, and coNP-complete wrt $\mathcal{E}$.*

PROOF. By Theorem 4 it is sufficient to show hardness. We show both PSPACE hardness results by reductions from regular expression universality, known to be PSPACE-complete [12].

To show PSPACE-hardness wrt $\mathcal{Q}$, we reduce non-universality of a regular expression R over the alphabet $\{0, 1\}$ to non-losslessness of $\mathcal{V} = (\$\cdot(0+1)^*\cdot\$)$ with respect to $Q = \$\cdot R\cdot\$$ relative to $\mathcal{E} = (\{(c,d)\})$, where \$ is a new symbol. We observe that (c, d) is the only pair of objects satisfying condition (1), and such a pair satisfies also condition (2) if and only if R is not equivalent to $(0 + 1)^*$.

To show PSPACE-hardness wrt $\mathcal{V}$, we reduce non-universality of a regular expression R over the alphabet $\{0, 1\}$ to non-losslessness of $\mathcal{V} = (\$\cdot R\cdot\$)$ with respect to $Q = \$\cdot(0 + 1)^*\cdot\$$ relative to $\mathcal{E} = (\emptyset)$. We observe that there is no pair of objects in $cert^{sound}_{Q,\mathcal{V}}(\mathcal{E})$, and hence all pairs of objects satisfy condition (2). Moreover, there exists a database $\mathcal{B}$ and a pair (c, d) of objects in $\mathcal{B}$ with $\mathcal{V}(\mathcal{B}) = (\emptyset)$ and $(c, d) \in Q(\mathcal{B})$ — i.e, satisfying condition (1) — if and only if R is not equivalent to $(0 + 1)^*$.

To show coNP-hardness wrt $\mathcal{E}$ we can exploit an idea in [6], and reduce graph 3-colorability [12] to non-losslessness relative to an extension. Without loss of generality, we consider 3-colorability of connected graphs with at least one edge. The alphabet is $\Sigma = \{r_{rg}, r_{gr}, r_{rb}, r_{br}, r_{gb}, r_{bg}, r_s, r_e\}$. Intuitively, $r_{\alpha\beta}$ is used to connect two vertices that have colors α and β respectively, and r_s and r_e are two additional symbols. We use three views $\mathcal{V} = (V_s, V_e, V_G)$, where

$$\begin{array}{rcl} V_s & = & r_s \\ V_e & = & r_e \\ V_G & = & r_{rg} + r_{gr} + r_{rb} + r_{br} + r_{gb} + r_{bg} \end{array}$$

The query is $Q = r_s\cdot M\cdot r_e$, where M is a regular expression over the symbols r_{xy} that describes all paths that contain a pair of mismatched color pairs. E.g., the pair $r_{rg}\cdot r_{rb}$ is mismatched, because r_{rg} denotes an edge from a red node to a green node, so it should be followed by r_{gb} or r_{gr}. Now, given a graph $G = (N, E)$ to be checked for 3-colorability and two objects c, d not in N, we define the extension $\mathcal{E} = (E_s, E_e, E_G)$ as follows

$$\begin{array}{rcl} E_s & = & \{(c, a) \mid a \in N\} \\ E_e & = & \{(a, d) \mid a \in N\} \\ E_G & = & \{(a, b), (b, a) \mid (a, b) \in E\} \end{array}$$

Intuitively, E_G represents G given as a symmetric directed graph, while E_s and E_e connect c and d to all nodes of the graph. Since G is connected and contains at least one edge, there is always a database $\mathcal{B}$ such that $\mathcal{V}(\mathcal{B}) = \mathcal{E}$ and $(c, d) \in Q(\mathcal{B})$ — for example, take all edges in E_G to be in r_{gb}. Moreover, (c, d) is the only possible answer. On the other hand, $(c, d) \notin cert^{sound}_{Q,\mathcal{V}}(\mathcal{E})$, i.e., there exists a database on which M and hence Q is empty, if and only if G is 3-colorable. $\square$

The results presented in this section extend immediately to the case of losslessness with respect to a view extension under exact views. By definition, $\mathcal{V}$ is not lossless with respect to Q relative to $\mathcal{E}$ under the exact view assumption if and only if there exists a pair of objects c and d such that:

1. there exists a database $\mathcal{B}$ with $\mathcal{V}(\mathcal{B}) = \mathcal{E}$ and $(c, d) \in Q(\mathcal{B})$;
2. $(c, d) \notin cert^{exact}_{Q,\mathcal{V}}(\mathcal{E})$.

It follows that, to check losslessness in this case, we can simply apply the same algorithm as for sound views, except that, we use the algorithm for computing certain answers with respect to exact views when we check condition (2). So the upper bound is the same as for losslessness under the sound view assumption. With respect to the lower bound, it is easy that the proof of Theorem 5 still applies for exact views. Thus we can conclude with the following theorem.

THEOREM 6. *Checking losslessness of $\mathcal{V}$ with respect to Q relative to $\mathcal{E}$ under the exact view assumption is PSPACE-complete wrt Q, PSPACE-complete wrt $\mathcal{V}$, and coNP-complete wrt $\mathcal{E}$.*

4. LOSSLESSNESS ABSTRACTING FROM VIEW EXTENSIONS

In this section, we deal with checking losslessness abstracting from the view extensions. We discuss losslessness under the sound and the exact view assumptions separately. The technique for checking losslessness under the sound view assumption makes use of the connection between constraint satisfaction and view based query answering [8], which we briefly recall here.

4.1 Constraint-satisfaction and View-Based Query Answering

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 [11].

A *vocabulary* is a set $V = \{R_1, \ldots, R_t\}$ of predicates, each with an associated arity. A *relational structure* $A = (\Delta_A, \cdot^A)$ over V is a *domain* Δ_A together with an *interpretation function* $\cdot^A$ that assigns to each predicate R_i a relation R_i^A of the appropriate arity over Δ_A. A *homomorphism* $h : A \rightarrow B$ between two relational structures A and B over the same vocabulary is a mapping $h : \Delta_A \rightarrow \Delta_B$ such that, if $(c_1, \ldots, c_n) \in R^A$, then $(h(c_1), \ldots, h(c_n)) \in R^B$, for every predicate R in the vocabulary.

Let $\mathcal{A}$ and $\mathcal{B}$ be two classes of finite relational 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 vocabulary, is there a homomorphism $h : A \rightarrow B$? We denote such an instance as $CSP(A, B)$, and if such a homomorphism exists we say that $CSP(A, B)$ is *satisfiable*. When $\mathcal{B}$ consists of a single relational structure B and $\mathcal{A}$ is the set of all relational structures over the vocabulary 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 $A \in \mathcal{A}$.

In [8] a tight relationship between constraint-satisfaction and view-based query answering under the sound view assumption is illustrated. In particular, the following reduction from view-based query answering for RPQs to non-uniform CSP is developed. Given a query Q and view definitions $\mathcal{V} = (V_1, \ldots, V_k)$, the *constraint template* $CT_{Q,\mathcal{V}}$ of Q wrt $\mathcal{V}$ is the structure B defined as follows.

- The vocabulary of B is $\{R_1, \ldots, R_k\} \cup \{U_c, U_d\}$, where each R_i, corresponding to V_i, denotes a binary predicate, and U_c and U_d denote unary predicates.
- Let $A_Q = (\Sigma, S, S_0, \rho, F)$ be a (nondeterministic) automaton for Q. The structure $B = (\Delta_B, \cdot^B)$ is given by:
  - $\Delta_B = 2^S$;
  - $\sigma \in U_c^B$ iff $S_0 \subseteq \sigma$;
  - $\sigma \in U_d^B$ iff $\sigma \cap F = \emptyset$;
  - $(\sigma_1, \sigma_2) \in R_i^B$ iff there exists a word $w \in L(V_i)$ such that $\rho(\sigma_1, w) \subseteq \sigma_2$ — we consider here ρ as extended to sets of states and words in the usual way.

Given view extensions $\mathcal{E} = (E_1, \ldots, E_k)$ and a pair of objects c and d, the *constraint instance* $\mathcal{E}^{c,d}$ of $CSP(CT_{Q,\mathcal{V}})$ is the structure $(\Delta_I, \cdot^I)$ defined as follows:

- $\Delta_I = \Delta_{\mathcal{E}} \cup \{c, d\}$;
- $R_i^I = E_i$, for $i \in \{1, \ldots, k\}$;
- $U_c^I = \{c\}$, and $U_d^I = \{d\}$.

THEOREM 7. [8] *Let Q be a query, $\mathcal{V}$ view definitions, $\mathcal{E}$ corresponding view extensions, and c, d a pair of objects. Then $(c, d) \notin cert_{Q,\mathcal{V}}^{sound}(\mathcal{E})$ if and only if there is a homomorphism from $\mathcal{E}^{c,d}$ to $CT_{Q,\mathcal{V}}$.*

4.2 Losslessness under Sound Views

We show how to check losslessness under the sound view assumption. The technique we propose is based on searching for counterexamples to losslessness. A *counterexample* to losslessness of view definitions $\mathcal{V}$ with respect to a query Q is a database $\mathcal{B}$ containing a pair of objects (c, d) such that $(c, d) \in Q(\mathcal{B})$ but $(c, d) \notin cert_{Q,\mathcal{V}}^{sound}(\mathcal{V}(\mathcal{B}))$. A *linear counterexample* is a counterexample database formed by a single simple path from c to d.

It turns out that, without loss of generality, we can concentrate on linear counterexamples. To show this, we exploit the correspondence between CSP and view-based query answering discussed above. We first prove the following technical lemma.

LEMMA 8. *Let Q be a query, $\mathcal{V}$ view definitions, $\mathcal{E}_1$ and $\mathcal{E}_2$ two corresponding view extensions, and (c, d) a pair of objects. Then, if there exists a homomorphism $h : \mathcal{E}_1^{c,d} \rightarrow \mathcal{E}_2^{h(c),h(d)}$, we have that $(c, d) \in cert_{Q,\mathcal{V}}^{sound}(\mathcal{E}_1)$ implies $(h(c), h(d)) \in cert_{Q,\mathcal{V}}^{sound}(\mathcal{E}_2)$.*

PROOF. If $(h(c), h(d)) \notin cert_{Q,\mathcal{V}}^{sound}(\mathcal{E}_2)$, then, by Theorem 7, there exists a homomorphism $h'(\cdot)$ from the constraint instance $\mathcal{E}_2^{h(c),h(d)}$ to the constraint template $CT_{Q,\mathcal{V}}$. Now, the mapping $h'(h(\cdot))$, obtained by composing $h(\cdot)$ and $h'(\cdot)$, is a homomorphism from $\mathcal{E}_1^{c,d}$ to $CT_{Q,\mathcal{V}}$. But then, again by Theorem 7, $(c, d) \notin cert_{Q,\mathcal{V}}^{sound}(\mathcal{E}_1)$. □

Intuitively, the crux for the above result is that view-based query answering is monotonic in our setting. This means that, if a pair of objects is a certain answer to a query with respect to given view extensions, and we add to such extensions more pairs of objects and/or more equalities (i.e., we identify elements), then the pair of objects is still a certain answer.

Exploiting such a result we can indeed show that we can concentrate on linear counterexamples to losslessness.

THEOREM 9. *If there is a counterexample to losslessness of $\mathcal{V}$ with respect to Q under the sound view assumption, then there is a linear counterexample.*

PROOF. Suppose there exists a database $\mathcal{B}$ such that for some pair of objects c and d, we have that $(c,d) \in Q(\mathcal{B})$ and $(c,d) \notin cert^{sound}_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B}))$. Since $(c,d) \in Q(\mathcal{B})$, there is a path $(y_0, r_1, \ldots, r_m, y_m)$ in $\mathcal{B}$ such that $y_0 = c$, $y_m = d$, and $r_1 \cdots r_m \in L(Q)$. Observe that the y_i's are not necessarily distinct.

Let us define a database $\mathcal{B}'$ consisting of the distinct objects $d_0, \ldots, d_m$ and of the edges (d_{i-1}, r_i, d_i), for $i \in \{1, \ldots, m\}$. Clearly, $(d_0, d_m) \in Q(\mathcal{B}')$. Consider now the extensions of the views. For each pair of objects d_j and d_h in $\mathcal{B}'$, if $(d_j, d_h) \in V_i(\mathcal{B}')$, then $(y_j, y_h) \in V_i(\mathcal{B})$. Thus, the mapping that maps d_i to y_i is a homomorphism from $\mathcal{V}(\mathcal{B}')^{d_0,d_m}$ to $\mathcal{V}(\mathcal{B})^{y_0,y_m}$. Since $(y_0, y_m) \notin cert^{sound}_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B}))$, by Lemma 8 we also have that $(d_0, d_m) \notin cert^{sound}_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B}'))$. $\square$

The fact that we can restrict the attention to linear counterexamples allows us to show decidability fairly easily from known results. By Theorem 7, computing pairs that are not certain answers can be expressed as CSP. On the other hand, CSP can be expressed in existential monadic second-order logic [11], and, by the classical result by Büchi, Elgot, and Trakhtenbrot, monadic second-order sentences on words can be translated to automata [21]. It follows that the set of linear counterexamples is regular, and we can construct an automaton that accepts it and check for its emptiness. This decidability result, however, does not provide us with good upper bounds for complexity. We show below how to obtain such bounds.

Without loss of generality, we search for a linear counterexample $\mathcal{B}$ of the form $(x_0, r_1, x_1, r_2, \ldots, r_m, x_m)$, for some m, such that $r_1 \cdots r_m \in L(Q)$ and thus $(x_0, x_m) \in Q(\mathcal{B})$. By Theorem 7, we have that $(x_0, x_m) \notin cert^{sound}_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B}))$ iff there is a homomorphism from $\mathcal{V}(\mathcal{B})^{x_0,x_m}$ to the constraint template $CT_{Q,\mathcal{V}}$. In other words, $(x_0, x_m) \notin cert^{sound}_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B}))$ 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 automaton $A_Q = (\Sigma, S, S_0, \rho, F)$ for Q such that the following conditions (which we call *CT-conditions*) hold:

- $S_0 \subseteq \ell(x_0)$;
- $\ell(x_m) \cap F = \emptyset$;
- for each pair of objects x_j and x_h in $\mathcal{B}$ and each view definition V_i in $\mathcal{V}$, we have that, if $r_{j+1} \cdots r_h \in L(V_i)$, then there exists a word $w \in L(V_i)$ such that $\rho(\ell(x_j), w) \subseteq \ell(x_h)$.

Thus, we are looking for a word of the form $\ell_0, r_1, \ldots, r_m, \ell_m$, where each ℓ_i is a set of states of A_Q, representing $\ell(x_i)$, and that satisfies these conditions. We construct an automaton $A_{Q,\mathcal{V}}$ that accepts such words and check its emptiness. The automaton is over the alphabet $\Sigma \cup \Lambda$, with $\Lambda = 2^S$, and is obtained as the cross product of two automata, A'_Q and $A'_{\mathcal{V}}$.

- A'_Q accepts words that
  1. consist of an alternation of symbols in Λ and symbols in Σ;
  2. start with a symbol $\ell \in \Lambda$ such that $S_0 \subseteq \ell$;
  3. end with a symbol $\ell \in \Lambda$ such that $\ell \cap F = \emptyset$;
  4. have a projection on Σ that is in $L(Q)$.

  A'_Q can be obtained by intersecting the straightforward automata that check conditions (1), (2), and (3) with an automaton $A_{Q,\Lambda}$ that checks condition (4). $A_{Q,\Lambda}$ can be obtained from A_Q by adding for each state $s \in S$ and each subset $\ell \in \Lambda$ a transition $\rho(s, \ell) = s$. The number of states of A'_Q is polynomial in Q and does not depend on $\mathcal{V}$.

- For the automaton $A'_{\mathcal{V}}$, let us define for each view V_i a binary relation V_i^S on Λ as follows: $(\ell, \ell') \in V_i^S$ if there exists a word $w \in L(V_i)$ such that $\rho(\ell, w) \subseteq \ell'$. Note that for each ℓ, ℓ' we can decide in PSPACE whether $(\ell, \ell') \in V_i^S$ holds[2], so all the relations V_i^S can be constructed in time exponential in the size of A_Q and linear in the size of $\mathcal{V}$. Now we can construct the automaton A''_{V_i} as follows. It reads a word, guesses positions j and $h > j$, checks that $r_{j+1} \cdots r_h \in L(V_i)$ and $(\ell_j, \ell_h) \notin V_i^S$. Clearly the size of this automaton is linear in V_i and exponential in A_Q, since it has to remember ℓ_j along its computation. To obtain $A'_{\mathcal{V}}$ we take the union of the automata A''_{V_i} and complement it.

THEOREM 10. *$\mathcal{V}$ is not lossless with respect to Q under the sound view assumption if and only if the automaton $A_{Q,\mathcal{V}}$ is non-empty.*

PROOF. "$\Rightarrow$" If $\mathcal{V}$ is not lossless with respect to Q, by Theorem 9 there exists a linear counterexample $\mathcal{B}$ of the form $(x_0, r_1, x_1, r_2, \ldots, r_m, x_m)$ such that $r_1 \cdots r_m \in L(Q)$ and thus $(x_0, x_m) \in Q(\mathcal{B})$. Since $(x_0, x_m) \notin cert^{sound}_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B}))$, by Theorem 7 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 $\ell(x_0) r_1 \ell(x_1) r_2 \cdots r_m \ell(x_m)$. Such a word is accepted by both A'_Q and $A'_{\mathcal{V}}$, and hence by $A_{Q,\mathcal{V}}$.

"$\Leftarrow$" If $A_{Q,\mathcal{V}}$ accepts a word $w = \ell_0 r_1 \ell_1 r_2 \cdots r_m \ell_m$, then there exists a linear database $\mathcal{B}$ of the form $(x_0, r_1, x_1, r_2, \ldots, r_m, x_m)$ 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 w is accepted by A'_Q, we have that $r_1 \cdots r_m \in L(Q)$. Since w is accepted by $A'_{\mathcal{V}}$, the CT-conditions hold and hence, by Theorem 7 $(x_0, x_m) \notin cert^{sound}_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B}))$. Hence $\mathcal{V}$ is not lossless with respect to Q. $\square$

The automata theoretic characterization above provides us with an upper bound for verifying losslessness.

THEOREM 11. *Checking losslessness of $\mathcal{V}$ with respect to Q can be done in PSPACE with respect to $\mathcal{V}$, and in EXPSPACE with respect to Q.*

[2]It suffices to check whether $L(V_i)$ is contained in the language accepted by the automaton $(\Sigma, S, \ell, \rho, S \setminus \ell')$ obtained from $A_Q = (\Sigma, S, S_0, \rho, F)$ by changing the initial and final states.

PROOF. The number of states of the automaton A'_Q is polynomial in Q and does not depend on $\mathcal{V}$. All the relations V_i^S can be constructed in time exponential in the size of Q and linear in the size of $\mathcal{V}$. A''_{V_i} has a number of states that is exponential in Q and polynomial in $\mathcal{V}$, and $A'_{\mathcal{V}}$, which requires complementation, has a number of states that is double exponential in Q and exponential in $\mathcal{V}$. Considering that emptiness of automata is NLOGSPACE in the number of states and that the construction of the above automata can be done on the fly while checking for emptiness, we get the claim. □

The following theorems show us that the above upper bounds are indeed tight.

THEOREM 12. *Checking whether a set $\mathcal{V}$ of views is lossless with respect to a query Q under the sound view assumption is PSPACE-complete wrt $\mathcal{V}$.*

PROOF. By Theorem 11 it suffices to show hardness. We reduce again universality of a regular expressions R over the alphabet $\{0, 1\}$ to losslessness of $\mathcal{V} = (\$\cdot R\cdot\$)$ with respect to $Q = \$\cdot(0 \cup 1)^*\cdot\$$. As shown in the proof of Theorem 5, if R is not equivalent to $(0 + 1)^*$, then $\mathcal{V}$ is not lossless with respect to Q relative to the extension $\mathcal{E} = (\emptyset)$. Hence $\mathcal{V}$ is not lossless with respect to Q. On the other hand, if R is equivalent to $(0 + 1)^*$, then the view $\$\cdot R\cdot\$$ is equivalent to Q, and hence $\mathcal{V}$ is trivially lossless. □

THEOREM 13. *Checking whether a set $\mathcal{V}$ of views is lossless with respect to a query Q under the sound view assumption is EXPSPACE-complete wrt Q.*

PROOF. Again by Theorem 11 it suffices to show hardness. We use a reduction from an EXPSPACE-complete tiling problem [23]. A *tiling problem* $(\Delta, \mathit{Hor}, \mathit{Ver})$ is defined by a finite set Δ of tile types, a horizontal adjacency relation $\mathit{Hor} \in \Delta \times \Delta$, and a vertical adjacency relation $\mathit{Ver} \in \Delta \times \Delta$, and consists in determining whether there exists a tiling of a region of the integer plane of size $2^n \times k$, for some k, with tiles of type in Δ such that the adjacency conditions are satisfied. Such a tiling is called a *good* tiling. We reduce the problem of checking whether $(\Delta, \mathit{Hor}, \mathit{Ver})$ admits a good tiling to the problem of checking whether a set of views $\mathcal{V}$ is not lossless with respect to a query Q under the sound view assumption.

By Theorem 9, it is enough to consider linear counterexamples to losslessness, i.e., a database of the form $(x_s, a_0, \ldots, a_m, x_f)$. We'll just refer to the word $a_0 \cdots a_m$ as the *database*.

We encode a tiling by a word in

$$e_{good} = \$\cdot(\#\cdot(0 + 1)^n\cdot\Delta\cdot(0 + 1)^n)^*\cdot\$$$

where \$ is a begin-end marker, # is a block marker, and $(0 + 1)^n$ are n-bit vectors. Such words are called *good*. We refer to each subword of the form $(0+1)^n\cdot t\cdot(0+1)^n$, where t is a tile type in Δ, as a *block*, and each block represents one tile. The idea is that the n-bit vectors provide an address between 0 and $2^n - 1$. We intend the addresses to start from 0 and behave like an n-bit counter. In each block we have two addresses, referred to as *left address* and *right address*. The intention is to have the two addresses be identical. A sequence of blocks is called a *segment*. Each segment where the addresses go once from 0 to 2^{n-1} represents one horizontal row of exponential length.

In addition to \$, #, 0, 1 and the tile types, which we call *old symbols*, we have other symbols, called *new symbols*. As we shall see, new symbols cannot appear in a counterexample database.

The query Q accepts, among other words, all words over the old symbols that (i) do not conform to the intended format, (ii) contain a block with two distinct addresses, (iii) contain successive addresses that are not adjacent, or (iv) contain successive tiles that do not satisfy the horizontal tiling constraint. Also, Q accepts words that mix old and new symbols. We call all of these *errors*. A database that contains an error is a *bad* database. (Note that a database can be neither good nor bad). We use a regular expression e_{bad} of size linear in n and the size of the tiling system to describe bad databases. The expression e_{bad} is the first part of Q.

The challenge is to relate blocks that represent vertically adjacent tiles. Such blocks have the same addresses and have precisely one block with the address 2^{n-1} between them. For this we use views $\mathcal{V}$.

The first view is

$$V_{bad} = e_{bad}$$

If the database $\mathcal{B}$ is bad, then $V_{bad}(\mathcal{B}) = \{(x_s, x_f)\}$. Thus, $(x_s, x_f) \in \mathit{cert}^{sound}_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B}))$. This means that a bad database cannot be a counterexample, because Q accepts all bad words.

Then we have views $V^l_{(i,c)}$, for $i \in \{1, \ldots, n\}$, and $c \in \{0, 1\}$. The intention for one such a view is to connect x_s to a node following a left address where the i-th bit is c. Such views are defined as

$$V^l_{(i,c)} = (\$\cdot(\#\cdot\mathit{block})^*\cdot\#\cdot(0 + 1)^{i-1}\cdot c\cdot(0 + 1)^{n-i}) + (l, i, c)$$

where (l, i, c) is a new symbol. Thus, when we see the pair (x_s, x) in $V^l_{(i,c)}(\mathcal{B})$, we do not know whether x_s and x are connected in $\mathcal{B}$ by a word of the form $\$\cdot(\#\cdot\mathit{block})^*\cdot\#\cdot(0 + 1)^{i-1}\cdot c\cdot(0 + 1)^{n-i}$ or by the single letter (l, i, c). The latter is called a *nonstandard interpretation*.

Analogously, we have views $V^r_{(i,c)}$, for $i \in \{1, \ldots, n\}$, and $c \in 0, 1$, where the intention for one such a view is to connect a node before a right address where the i-th bit is c to x_f. Such views are defined as

$$V^r_{(i,c)} = (0 + 1)^{i-1}\cdot c\cdot(0 + 1)^{n-i}\cdot(\#\cdot\mathit{block})^*\cdot\$ + (r, i, c)$$

where (r, i, c) is a new symbol. Here we can also have a nonstandard interpretation.

Finally, we have a view V_{error} defined as a constrained version of

$$\Big(\sum_{t,t' \mid (t,t') \notin Ver} t \cdot (0+1)^n \cdot (\# \cdot block)^* \cdot \# \cdot (0+1)^n \cdot t'\Big) + (b_1 + \cdots + b_n)$$

where $b_1, \ldots, b_n$ are new symbols. The constraint is that the subword 1^n can occur only once between t and t'. (We can construct a regular expression of size quadratic in n that contains all words in $(0+1)^n \cdot (\# \cdot block)^* \cdot \# \cdot (0+1)^n$ that obey this constraint.) When we see a pair (x, y) in $V_{error}(\mathcal{B})$, we do not know whether x and y are connected in $\mathcal{B}$ by a word of the form $t \cdot (0+1)^n \cdot (\# \cdot block)^* \cdot \# \cdot (0+1)^n \cdot t'$ or by a single letter b_i. The latter is called a nonstandard interpretation.

We can now define the query Q as the sum of terms

$$e_{bad} + e_{good} + \sum_{\substack{i \in \{1,\ldots,n\} \\ c \in \{0,1\}}} V^l_{(i,c)} \cdot b_i \cdot V^r_{(i,c)}$$

We first show that if there is a good tiling, then there is a counterexample database. Let $w = a_s \cdots a_f$ be a word that describes the good tiling. We claim that the database $\mathcal{B} = (x_s, a_s, \ldots, a_f, x_f)$ is a counterexample database. Since w is a good word, clearly (x_s, x_f) is in $Q(\mathcal{B})$. We need to show that (x_s, x_f) is not in $cert^{sound}_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B}))$. For this we need to display a database $\mathcal{B}'$ such that $\mathcal{V}(\mathcal{B}) \subseteq \mathcal{V}(\mathcal{B}')$, but (x_s, x_f) is not in $Q(\mathcal{B}')$.

First note that $V_{bad}(\mathcal{B})$ is empty, since the database is good. The view $V^l_{(i,c)}$ connects x_s to every node x following a left address where the i-th bit is c. In $\mathcal{B}'$ we take the nonstandard interpretation and assume that x_s is connected to x by letter (l, i, c). We handle $V^r_{(i,c)}$ analogously. Finally, suppose that (x, y) is in $V_{error}(\mathcal{B})$. Since $\mathcal{B}$ describes a good tiling, x and y cannot represent vertically adjacent tiles. Thus, the address after x and the address before y must differ on some bit i of the n address bit. In $\mathcal{B}'$ we take the nonstandard interpretation and connect x and y by the single letter b_i.

We now need to show that (x_s, x_f) is not in $Q(\mathcal{B}')$. Clearly, x_s is not connected in $\mathcal{B}'$ to x_f by either a word in e_{bad} or a word in e_{good}. The only other possibility is that x_s is connected in $\mathcal{B}'$ to x_f by a word $(l, i, c) b_i (r, i, c)$, for some $i \in \{1, \ldots, n\}$, and $c \in \{0, 1\}$. But this means that x_s is connected to some x by (l, i, c), x is connected to some y by b_i, and y is connected to x_f by (r, i, c). By construction, x follows a left address where the i-th bit is c, and y precedes a right address where the i-th bit is c. But since x is connected to y by b_i, these addresses need to differ on the i-th bit, which is a contradiction.

It remains to show that only good tilings provide counterexample databases. Assume that there is no good tiling and let $\mathcal{B} = (x_s, a_s, \ldots, a_f, x_f)$ be a database such that $(x_s, x_f) \in Q(\mathcal{B})$. We show that (x_s, x_f) is in $cert^{sound}_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B}))$. Since $(x_s, x_f) \in Q(\mathcal{B})$, there are three possibilities:

1. x_s is connected to x_f by a bad word.
2. x_s is connected to x_f by a good word.
3. x_s is connected to x_f by a word in $V^l_{(i,c)}) \cdot b_i \cdot V^r_{(i,c)})$, for some $i \in \{1, \ldots, n\}$, and $c \in \{0, 1\}$.

In the first case, (x_s, x_f) is in $V_{bad}(\mathcal{B})$. This means that for all databases $\mathcal{B}'$ such that $\mathcal{V}(\mathcal{B}) \subseteq \mathcal{V}(\mathcal{B}')$, we have that x_s is connected to x_f by a bad word also in $\mathcal{B}'$, so $(x_s, x_f) \in Q(\mathcal{B}')$.

In the third case, we have that (x_s, x) is in $V^l_{(i,c)}(\mathcal{B})$, (x, y) is in $V_{error}(\mathcal{B})$, and (y, x_f) is in $V^r_{(i,c)}(\mathcal{B})$. Thus, for all databases $\mathcal{B}'$ such that $\mathcal{V}(\mathcal{B}) \subseteq \mathcal{V}(\mathcal{B}')$, we also have that (x_s, x) is in $V^l_{(i,c)}(\mathcal{B}')$, (x, y) is in $V_{error}(\mathcal{B}')$, and (y, x_f) is in $V^r_{(i,c)}(\mathcal{B}')$. Each of these views can have a standard or nonstandard interpretation in $\mathcal{B}'$. If all interpretations are standard, then x_s is connected to x_f in $\mathcal{B}'$ by a good word, so (x_s, x_f) is in $Q(\mathcal{B}')$. If some of these interpretations are standard and some are nonstandard, then x_s is connected to x_f in $\mathcal{B}'$ by a bad word that mixes old and new symbols, so (x_s, x_f) is in $Q(\mathcal{B}')$. It follows that all three interpretations must be nonstandard, which means that also in $\mathcal{B}'$ we have that x_s is connected to some x by (l, i, c), x is connected to some y by b_i, and y is connected to x_f by (r, i, c). But then we also have that (x_s, x_f) is in $Q(\mathcal{B}')$.

It remains to consider the case that x_s is connected to x_f in $\mathcal{B}$ by a good word w that is not a bad word. Thus, w must be representing a tiling that contains a vertical error. That is, there is some x following an address a and preceding a tile t, and some y following a tile t' and preceding the same address a such that $(t, t') \notin Ver$ even though the two tiles are in vertically adjacent positions. It follows that $(x, y) \in V_{error}(\mathcal{B})$. Consider an arbitrary database $\mathcal{B}'$ such that $\mathcal{V}(\mathcal{B}) \subseteq \mathcal{V}(\mathcal{B}')$. In particular, $(x, y) \in V_{error}(\mathcal{B}')$. We now have two cases to consider: a) We have a nonstandard interpretation in $\mathcal{B}'$, which means that x is connected to y in $\mathcal{B}'$ by some b_i, with $i \in \{1, \ldots, n\}$. b) We have a standard interpretation in $\mathcal{B}'$, which means that x is connected to y in $\mathcal{B}'$ by a word in $t \cdot (0+1)^n \cdot (\# \cdot block)^* \cdot \# \cdot (0+1)^n \cdot t'$.

In case (a), let the i-th bit of the address a be c. We know that (x_s, x) is in $V^l_{(i,c)}(\mathcal{B})$ and (y, x_f) is in $V^r_{(i,c)}(\mathcal{B})$. It follows that (x_s, x) is in $V^l_{(i,c)}(\mathcal{B}')$ and (y, x_f) is in $V^r_{(i,c)}(\mathcal{B}')$. That is, in $\mathcal{B}'$ we have that x_s is connected to x by a word in $V^l_{(i,c)}$, x is connected to y by b_i, and y is connected to x_f by a word in $V^r_{(i,c)}$. It follows that (x_s, x_f) is in $Q(\mathcal{B}')$.

In case (b), choose an arbitrary $i \in \{1, \ldots, n\}$, and let the i-th bit of the address a be c. We know that (x_s, x) is in $V^l_{(i,c)}(\mathcal{B})$ and (y, x_f) is in $V^r_{(i,c)}(\mathcal{B})$. It follows that (x_s, x) is in $V^l_{(i,c)}(\mathcal{B}')$ and (y, x_f) is in $V^r_{(i,c)}(\mathcal{B}')$. If either has a nonstandard interpretation, then we have in $\mathcal{B}'$ a path that mixes old and new symbols, which is bad, so $(x_s, x_f) \in Q(\mathcal{B}')$. If both have a standard interpretation, then we have in $\mathcal{B}'$ a good path from x_s to x_f, so again $(x_s, x_f) \in Q(\mathcal{B}')$. Thus, if there is no good tiling, $\mathcal{B}$ cannot be a counterexample database. □

4.3 Losslessness under Exact Views

We remind the reader that a set of views $\mathcal{V}$ is lossless with respect to Q under the exact view assumption , if for every database $\mathcal{B}$ we have that $Q(\mathcal{B}) = cert^{exact}_{Q,\mathcal{V}}(\mathcal{V}(\mathcal{B}))$. From the definition, it is immediate to verify that losslessness under sound views implies losslessness under exact views.

However, the following example shows that, in the context of regular path queries, losslessness under exact views does *not* imply losslessness under sound views.

EXAMPLE 3. Consider the alphabet $\Sigma = \{0,1\}$, the query Q with

$$L(Q) = (0+1)^3 \setminus \{111\}$$

and the views $\mathcal{V} = (V_1, V_2, V_3, V_4)$ with

$$\begin{array}{rcl} L(V_1) & = & \{0,1\} \\ L(V_2) & = & \{1\} \\ L(V_3) & = & \{01\} \\ L(V_4) & = & \{10\} \end{array}$$

It is easy to see that $\mathcal{V}$ is not lossless with respect to Q under sound views. Indeed, the database

$$\mathcal{B} = (x_1, 0, x_2, 0, x_3, 0, x_4)$$

is a counterexample to losslessness, since $(x_1, x_4) \in Q(\mathcal{B})$ and there exists a database $\mathcal{B}'$

$$\mathcal{B}' = (x_1, 1, x_2, 1, x_3, 1, x_4)$$

such that $\mathcal{V}(\mathcal{B}) \subseteq \mathcal{V}(\mathcal{B}')$, and $(x_1, x_4) \notin Q(\mathcal{B}')$.

We show, however, that $\mathcal{V}$ is lossless with respect to Q under exact views. By contradiction, assume there is a database $\mathcal{B}$ with a pair of objects x_1 and x_4 such that $(x_1, x_4) \in Q(\mathcal{B})$, and there is a database $\mathcal{B}'$ such that $\mathcal{V}(\mathcal{B}') = \mathcal{V}(\mathcal{B})$ but $(x_1, x_4) \notin Q(\mathcal{B}')$. Since $(x_1, x_4) \in Q(\mathcal{B})$, $\mathcal{B}$ contains two objects x_2 and x_3 (possibly non-distinct and possibly coinciding with x_1 or x_4) and a path $(x_1, a, x_2, b, x_3, c, x_4)$, with $a, b, c \in \Sigma$, such that $abc \neq 111$. Since $V_1(\mathcal{B}') = V_1(\mathcal{B})$, we have that $\mathcal{B}'$ contains at least a path $(x_1, e, x_2, f, x_3, g, x_4)$, with $e, f, g \in \Sigma$. Since $(x_1, x_4) \notin Q(\mathcal{B}')$, for all such paths we have that $efg = 111$. However, since $abc \neq 111$, in the case where $a = b = c = 0$ we have that $V_2(\mathcal{B}') \neq V_2(\mathcal{B})$, in the cases where $a = 0$ or $b = 0$ we have that $V_3(\mathcal{B}') \neq V_3(\mathcal{B})$, and in the cases where $b = 0$ or $c = 0$ we have that $V_4(\mathcal{B}') \neq V_4(\mathcal{B})$. In all cases, this contradicts the assumption that $\mathcal{V}(\mathcal{B}') = \mathcal{V}(\mathcal{B})$. ■

Note that, an interesting consequence of the above example is that searching for linear counterexamples is not sufficient for deciding losslessness under exact views. This means that we cannot directly extend our technique to deal with this case.

The question of losslessness under exact views is largely unexplored. To the best of our knowledge, no technique is known for this problem, even for the case of conjunctive queries and views. This question is an interesting direction for continuing the work presented here.

5. RELATIONSHIP TO REWRITING

The notion of losslessness allows us to compare a query with the answers derivable from a collection of views. However, losslessness as introduced in the previous sections is not the only way to perform such a comparison. In particular, previous attempts at defining view losslessness have been in terms of query rewriting. We show here that, in the context of regular path queries, our notion is strictly more general.

We briefly recall the notion of query rewriting in our setting. Given views $\mathcal{V}$, we assume to have an alphabet $\Sigma_{\mathcal{V}}$ consisting of one symbol v_i for each view definition V_i in $\mathcal{V}$. Given a language λ over $\Sigma_{\mathcal{V}}$, we denote by $expand_\Sigma(\lambda)$ the language defined as follows

$$expand_\Sigma(\lambda) = \bigcup_{v_{i_1}\cdots v_{i_n} \in \lambda} \{w_1 \cdots w_n \mid w_h \in L(V_{i_h}), \text{ for } h \in \{1,\ldots,n\}\}$$

If R is a regular language over $\Sigma_{\mathcal{V}}$, we say that R is a *(regular) rewriting* of a query Q with respect to views $\mathcal{V}$ if $expand_\Sigma(L(R)) \subseteq L(Q)$. A rewriting R of Q with respect to $\mathcal{V}$ is a *maximally contained rewriting* if for each rewriting R' of Q with respect to $\mathcal{V}$ we have that $expand_\Sigma(L(R')) \subseteq expand_\Sigma(L(R))$.

Determining whether a query is equivalent to its maximally contained rewriting is 2EXPSPACE-complete [5]. Since checking losslessness (under the sound view assumption) can be done in EXPSPACE, the two notions cannot coincide. The following theorem demonstrates that losslessness is more general.

THEOREM 14. *If Q is equivalent to its maximally contained rewriting with respect to views $\mathcal{V}$, then $\mathcal{V}$ is lossless with respect to Q under the sound (and hence under the exact) view assumption.*

PROOF. Let $\mathcal{B}$ be a database with a path from x to y labeled by $w \in L(Q)$. Since Q is equivalent to its maximally contained rewriting $rew(Q)$ with respect to $\mathcal{V}$, there exist words $w_1 \in L(V_{i_1}), \ldots, w_n \in L(V_{i_n})$ such that $w_1 \cdots w_n = w$ and $v_{i_1} \cdots v_{i_n} \in L(rew(Q))$. Now, let $\mathcal{B}'$ be a database such that $\mathcal{V}(\mathcal{B}) \subseteq \mathcal{V}(\mathcal{B}')$. Then $\mathcal{B}'$ contains a path from x to y labeled by $w'_1 \cdots w'_n$ with $w'_h \in L(V_{i_h})$. Since $v_{i_1} \cdots v_{i_n} \in L(rew(Q))$, the word $w'_1 \cdots w'_n$ belongs to $L(Q)$ and hence $(x, y) \in Q(\mathcal{B}')$. It follows that $\mathcal{V}$ is lossless with respect to Q under the sound view assumption. □

The following example shows a case where the query Q is *not* equivalent to its maximally contained rewriting with respect to views $\mathcal{V}$, and still views $\mathcal{V}$ are lossless with respect to the query under the sound (and hence under the exact) view assumption.

EXAMPLE 4. Consider the alphabet $\Sigma = \{0,1\}$, the query Q with

$$L(Q) = \{0000, 0101, 1010, 1111\}$$

and the views $\mathcal{V} = (V_1, V_2)$, with

$$\begin{array}{rcl} L(V_1) & = & \{0,1\} \\ L(V_2) & = & \{000, 010, 101, 111\} \end{array}$$

It is easy to see that the maximally contained rewriting of Q with respect to $\mathcal{V}$ is empty.

On the other hand, suppose we have a database $\mathcal{B}$ with a path $(x_1, a, x_2, b, x_3, a, x_4, b, x_5)$, with $a, b \in \Sigma$. We have that

- $(x_1, x_5) \in Q(\mathcal{B})$,
- $V_1(\mathcal{B}) = \{(x_1, x_2), (x_2, x_3), (x_3, x_4), (x_4, x_5)\}$, and
- $V_2(\mathcal{B}) = \{(x_1, x_3), (x_2, x_4), (x_3, x_5)\}$.

Consider now a database $\mathcal{B}'$ such that $\mathcal{V}(\mathcal{B}) \subseteq \mathcal{V}(\mathcal{B}')$. Since $V_1(\mathcal{B}) \subseteq V_1(\mathcal{B}')$, $\mathcal{B}'$ contains a path $(x_1, c, x_2, d, x_3, e, x_4, f, x_5)$, with $c, d, e, f \in \Sigma$. Moreover, since $V_2(\mathcal{B}) \subseteq V_2(\mathcal{B}')$, we have that $c = e$ and $d = f$, and therefore

$$(x_1, x_5) \in Q(\mathcal{B}').$$

Hence $\mathcal{V}$ is lossless with respect to Q under the sound view assumption. ∎

6. CONCLUSIONS

In several applications the question arises of whether a set of views are lossless with respect to a query, i.e., whether the information content of the views is sufficient to answer completely a given query. We have addressed this question in a context where the database is semistructured, and both the query and the views are expressed as regular path queries.

We have shown that, in the case where we have the view extensions available, the problem is solvable by extending known techniques. The more general version of the problem, namely the one where we abstract from the specific view extension, can be studied in at least two different settings. In a first setting, losslessness is checked by assuming the views are sound, but not necessarily complete. In this case, we have illustrated a technique for checkng losslessness, and we have stidied the computational complexity of the problem. In a second setting, views are considered exact (i.e., both sound and complete). In this case, the question of losslessness under exact views is largely unexplored, and needs further investigation.

Indeed, we plan to address this problem in the continuation of the work presented here. We are also interested in investigating losslessness (under both sound and exact views) for other classes of queries for semistructured databases, e.g., regular path queries with inverse, and conjunctive regular path queries.

7. ACKNOWLEDGEMENTS

This work was supported in part by NSF grants CCR-9988322, IIS-9908435, IIS-9978135, and EIA-0086264, and by MURST project “Integration, Warehousing, and Mining of Heterogeneous Data Sources” (D2I).

8. REFERENCES

[1] S. Abiteboul. Querying semi-structured data. In *Proc. of the 6th Int. Conf. on Database Theory (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, Los Altos, 2000.

[3] S. Abiteboul and O. Duschka. Complexity of answering queries using materialized views. In *Proc. of the 17th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS'98)*, pages 254–265, 1998.

[4] P. Buneman. Semistructured data. In *Proc. of the 16th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS'97)*, pages 117–121, 1997.

[5] D. Calvanese, G. De Giacomo, M. Lenzerini, and M. Y. Vardi. Rewriting of regular expressions and regular path queries. In *Proc. of the 18th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS'99)*, pages 194–204, 1999.

[6] D. Calvanese, G. De Giacomo, M. Lenzerini, and M. Y. Vardi. Answering regular path queries using views. In *Proc. of the 16th IEEE Int. Conf. on Data Engineering (ICDE 2000)*, pages 389–398, 2000.

[7] D. Calvanese, G. De Giacomo, M. Lenzerini, and M. Y. Vardi. Query processing using views for regular path queries with inverse. In *Proc. of the 19th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS 2000)*, pages 58–66, 2000.

[8] D. Calvanese, G. De Giacomo, M. Lenzerini, and M. Y. Vardi. View-based query processing and constraint satisfaction. In *Proc. of the 15th IEEE Symp. on Logic in Computer Science (LICS 2000)*, pages 361–371, 2000.

[9] R. Chirkova, A. Y. Halevy, and D. Suciu. A formal perspective on the view selection problem. In *Proc. of the 27th Int. Conf. on Very Large Data Bases (VLDB 2001)*, pages 59–68, 2001.

[10] O. M. Duschka and M. R. Genesereth. Answering recursive queries using views. In *Proc. of the 16th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS'97)*, pages 109–116, 1997.

[11] T. Feder and M. Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction. *SIAM J. on Computing*, 28:57–104, 1999.

[12] M. R. Garey and D. S. Johnson. *Computers and Intractability — A guide to NP-completeness.* W. H. Freeman and Company, San Francisco (CA, USA), 1979.

[13] G. Grahne and A. O. Mendelzon. Tableau techniques for querying information sources through global schemas. In *Proc. of the 7th Int. Conf. on Database Theory (ICDT'99)*, volume 1540 of *Lecture Notes in Computer Science*, pages 332–347. Springer, 1999.

[14] S. Grumbach and L. Tininini. On the content of materialized aggregate views. In *Proc. of the 19th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS 2000)*, pages 47–57, 2000.

[15] A. Y. Halevy. Theory of answering queries using views. *SIGMOD Record*, 29(4):40–47, 2000.

[16] A. Y. Levy, A. O. Mendelzon, Y. Sagiv, and D. Srivastava. Answering queries using views. In *Proc. of the 14th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS'95)*, pages 95–104, 1995.

[17] C. Li, M. Bawa, and J. D. Ullman. Minimizing view sets without loosing query-answering power. In *Proc. of the 8th Int. Conf. on Database Theory (ICDT 2001)*, pages 99–103, 2001.

[18] C. Li and E. Chang. On answering queries in the presence of limited access patterns. In *Proc. of the 8th Int. Conf. on Database Theory (ICDT 2001)*, pages 219–233, 2001.

[19] T. D. Millstein, A. Y. Levy, and M. Friedman. Query containment for data integration systems. In *Proc. of the 19th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS 2000)*, pages 67–75, 2000.

[20] R. Reiter. On closed world data bases. In H. Gallaire and J. Minker, editors, *Logic and Databases*, pages 119–140. Plenum Publ. Co., New York, 1978.

[21] W. Thomas. Languages, automata, and logic. In *Handbook of Formal Language Theory*, volume III, pages 389–455. 1997.

[22] J. D. Ullman. Information integration using logical views. In *Proc. of the 6th Int. Conf. on Database Theory (ICDT'97)*, volume 1186 of *Lecture Notes in Computer Science*, pages 19–40. Springer, 1997.

[23] P. van Emde Boas. The convenience of tilings. In A. Sorbi, editor, *Complexity, Logic, and Recursion Theory*, volume 187 of *Lecture Notes in Pure and Applied Mathematics*, pages 331–363. Marcel Dekker Inc., 1997.