Source inconsistency and incompleteness in data integration

Domenico Lembo, Maurizio Lenzerini, Riccardo Rosati
Dipartimento di Informatica e Sistemistica
Università di Roma "La Sapienza"
Via Salaria 113, I-00198 Roma, Italy
lastname@dis.uniroma1.it

Abstract

Data integration is the problem of providing users with a unified view of heterogeneous data sources, called *global schema*. In general, data at the sources may result incomplete in providing answers to queries issued to a data integration system, and may be inconsistent with respect to integrity constraints expressed over the global schema. In this paper we present a general semantics for data integration in the presence of incomplete information sources and inconsistency of data with respect to constraints of a general form over the global schema. We define a method for query processing under the above semantics, when the constraints over the global schema are key constraints and foreign key constraints, and characterize the computational complexity of the query processing problem in such a setting.

1 Introduction

A data integration system deals with the problem of combining the data residing at different sources, and providing the user with a unified view of them, called *global schema*. Users formulate their queries over the global schema, and the system provides the answers to such queries on the basis of the data stored at the sources, i.e., provides solutions to the *query processing* problem. An important aspect that has to be taken into account in order to process a query is the form of the *mapping* that specifies the relationship between the sources and the global schema. By exploiting such a mapping, the system can access the appropriate sources and answer queries, thus freeing the users from both having to locate the data relevant for their queries, and knowing how data are structured at the sources, or how data are to be merged and reconciled to fit into the global schema.

There are basically two approaches for specifying the mapping. The first approach, called *global-as-view* (GAV), requires that a view over the sources is associated with every element of the global schema, so that its meaning is specified in terms of the data residing at the sources.

Conversely, the second approach, called *local-as-view* (LAV), requires the sources to be defined as views over the global schema.

Whatever form of the mapping or method to process queries is adopted, in a data integration context the data stored in autonomous and heterogeneous sources are considered as a whole, and they may result inconsistent due to the presence of integrity constraints expressed over the global schema. Generally speaking, when data are considered locally, they must satisfy only the integrity constraints specified in the source to which they belong, but they are not required to satisfy the integrity constraints expressed in the global schema. Hence, when data sources are considered as part of an integration system, and a set of global requirements is expressed in the system, the data might present inconsistencies that have to be properly taken into account during query processing.

Another aspect is that the set of available sources might not store exactly the data needed to answer a query posed to the system, so that the mapping cannot be considered *exact*. It might be that the data that can be retrieved at the sources are either a subset or a superset of the data that satisfy the corresponding portion of the global schema, and the mapping has to be considered *sound* or *complete*, respectively. Hence, query processing in a data integration system is, in general, a form of reasoning with incomplete information [12].

In this paper, we restrict our analysis to the GAV approach, which is generally considered sufficiently simple and effective for practical purposes. We present a framework for data integration in GAV that allows for the specification of integrity constraints of general form over the global schema, and we define a semantics for the integration system that is able to deal with incomplete and inconsistent data. Then, we propose a method for query processing under the above semantics, when the constraints over the global schema are key constraints and foreign key constraints, and we show the computational characterization of the query processing problem in such a setting.

Our semantics follows the principles that underlie the notion of *database repair* introduced by previous work in the area of inconsistent databases. However, such studies basically apply to a single database setting [4, 2, 3], and the proposed techniques can be employed in a data integration setting only by assuming an "exact" interpretation of the mapping [11, 7, 9]. Conversely, our approach considers a pure data integration framework, and our semantics seriously takes into account the interpretation of the mapping. Specifically, in order to capture the incomplete nature of the sources, we interpret the mapping as sound rather than exact, and in order to deal with inconsistent data we make the further assumption that this soundness can be suitably relaxed: hence, we model the mapping in a way that we call *loosely-sound*.

Moreover, to the best of our knowledge, our method for query answering is the first one able to deal with foreign key constraints in the global schema: indeed, the methods proposed in the literature for query answering in inconsistent databases (see e.g. [9, 3]) consider the class of "universally quantified" constraints only.

The paper is organized as follows. In Section 2, we present our data integration framework. The loosely-sound semantics of the mapping is defined in Section 3. In Section 4, we propose our method for query processing in the presence of key and foreign key constraints over the global schema, and characterize the computational complexity of query processing in such a setting. Finally, Section 5 describes some related works and Section 6 provides some discussions and concludes the paper.

2 Framework

In this section, we describe a formal framework for data integration systems based on the relational model [1]. For the sake of simplicity, we consider to have a fixed alphabet Γ of constants representing real world objects, and will take into account only databases having Γ as domain. Furthermore, we adopt the so-called *unique name assumption*, i.e., we assume that different constants denote different objects. Formally, a data integration system $\mathcal{I}$ is a triple $\langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, where:

- $\mathcal{G}$ is the *global schema* expressed in the relational model with integrity constraints. Henceforth, we indicate with $\mathcal{A}_\mathcal{G}$ the alphabet of the relation symbols of $\mathcal{G}$ and with $\mathcal{C}_\mathcal{G}$ the set of integrity constraints specified on $\mathcal{A}_\mathcal{G}$.
- $\mathcal{S}$ is the *source schema*, constituted by the schemas of the various sources that are part of the data integration system. We assume that the sources are relational, and no integrity constraint is expressed on the source schema. Henceforth, we indicate with $\mathcal{A}_\mathcal{S}$ the alphabet of the relation symbols of $\mathcal{S}$. Obviously, $\mathcal{A}_\mathcal{S}$ is disjoint from $\mathcal{A}_\mathcal{G}$.
- $\mathcal{M}$ is the *mapping* between the global and the source schema. In our framework the mapping is defined in the GAV approach, i.e., each relation in $\mathcal{G}$ is associated with a *view*, i.e., a query, over the sources. We assume that the language used to express queries in the mapping is *positive* DATALOG [1], i.e., recursive DATALOG without negation, over the alphabet $\mathcal{A}_\mathcal{S}$. We indicate the mapping as a set of assertions of the form $\langle r, V \rangle$, where r is a relation and V is the associated view over the source schema.

In order to specify the semantics of a data integration system $\mathcal{I}$, we start by considering a *source database* for $\mathcal{I}$, i.e., a database $\mathcal{D}$ for the source schema $\mathcal{S}$. Based on $\mathcal{D}$, we now specify which is the information content of the global schema $\mathcal{G}$. We call *global database* for $\mathcal{I}$ any database for $\mathcal{G}$. A global database $\mathcal{B}$ for $\mathcal{I}$ is said to be *legal* with respect to $\mathcal{D}$, if:

- $\mathcal{B}$ is coherent with $\mathcal{G}$, i.e., it satisfies the integrity constraints in $\mathcal{C}_\mathcal{G}$.
- $\mathcal{B}$ satisfies the mapping wrt $\mathcal{D}$. We will explain the exact meaning of the notion of satisfaction of the mapping in the next section.

Given a source database $\mathcal{D}$ for $\mathcal{I}$, the semantics of $\mathcal{I}$ wrt $\mathcal{D}$, denoted $sem(\mathcal{I}, \mathcal{D})$, is defined as follows:

$$sem(\mathcal{I}, \mathcal{D}) = \{ \mathcal{B} \mid \mathcal{B} \text{ is a legal global database for } \mathcal{I} \text{ wrt } \mathcal{D} \}$$

A *query* over the global schema q is a formula that provides the specification of which data to retrieve from the integration system, i.e., q is intended to extract a set of tuples of elements of Γ. Thus, every query has an associated arity. While for the views in the mapping we make use of positive DATALOG queries expressed over the alphabet $\mathcal{A}_\mathcal{S}$, for the queries over the global schema we restrict our analysis to *conjunctive queries* [1] expressed over the alphabet $\mathcal{A}_\mathcal{G}$.

In order to define the semantics of a query over the global schema, we have to consider that several global databases may exist that are legal for $\mathcal{I}$ wrt $\mathcal{D}$, and we

are interested in computing the set of tuples that satisfy the query in all databases in $sem(\mathcal{I}, \mathcal{D})$. Formally, we call *certain answers* of a query q of arity n wrt $\mathcal{I}$ and $\mathcal{D}$, the set $q^{\mathcal{I},\mathcal{D}}$ defined as follows:

$$q^{\mathcal{I},\mathcal{D}} = \{(c_1, \ldots, c_n) \mid \text{for each } \mathcal{DB} \in sem(\mathcal{I}, \mathcal{D}),\ (c_1, \ldots, c_n) \in q^{\mathcal{DB}}\ \}$$

where $q^{\mathcal{DB}}$ denotes the result of evaluating q in the database $\mathcal{DB}$.

Example 1 Consider a data integration system $\mathcal{F} = \langle \mathcal{G}^{\mathcal{F}}, \mathcal{S}^{\mathcal{F}}, \mathcal{M}^{\mathcal{F}} \rangle$ referring to the context of football teams. The global schema $\mathcal{G}^{\mathcal{F}}$ consists of the relation predicates $\mathsf{player}(Pcode, Pname, Pteam)$ and $\mathsf{team}(Tcode, Tname, Tleader)$. For the moment we assume that no constraint is defined in the global schema. The source schema $\mathcal{S}^{\mathcal{F}}$ consists of the schemas of three sources comprising the relation s_1 of arity 4, and the relations s_2 and s_3, both of arity 3. Finally, the mapping $\mathcal{M}^{\mathcal{F}}$ is defined by the two assertions

$$\begin{array}{ll} \langle \mathsf{player}, & \mathsf{player}(X, Y, Z) \leftarrow \mathsf{s}_1(X, Y, Z, W) \rangle \\ \langle \mathsf{team}, & \mathsf{team}(X, Y, Z) \leftarrow \mathsf{s}_2(X, Y, Z) \vee \mathsf{s}_3(X, Y, Z) \rangle \end{array}$$

where $\mathsf{team}(X, Y, Z) \leftarrow \mathsf{s}_2(X, Y, Z) \vee \mathsf{s}_3(X, Y, Z)$ is equivalent to the DATALOG program

$$\begin{array}{l} \mathsf{team}(X, Y, Z) \leftarrow \mathsf{s}_2(X, Y, Z) \\ \mathsf{team}(X, Y, Z) \leftarrow \mathsf{s}_3(X, Y, Z) \end{array}$$

■

3 Semantics of the Mapping

As we said before, in this section we turn our attention on the notion of mapping satisfaction. We recall the reader that in the GAV approach the mapping between the global and the source schema is given by associating each global relation with a view over the sources. The intended meaning of such association is that the view represents the best way to characterize the instances of the global relation using the relations in the sources.

Generally speaking, given a data integration system $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, in order to satisfy the mapping a global database $\mathcal{B}$ has to respect the *assumptions* adopted for the views. More specifically, given a source database $\mathcal{D}$ and a mapping assertion $\langle r, V \rangle$, the assumption adopted for the view V is intended to establish how to interpret the set of the tuples that $\mathcal{B}$ assigns to r, i.e., $r^{\mathcal{B}}$, with respect to the tuples retrieved by the view V, i.e., $V^{\mathcal{D}}$. Traditionally, in GAV data integration systems, the views are considered exact. This means that each view provides exactly the data needed to satisfy the corresponding relation of the global schema. However, in more general cases, the views provide only a subset of the data that satisfy the corresponding relations of the global schema, thus they have to be considered sound. Formally, we have that $\mathcal{B}$ satisfies the sound mapping assertion $\langle r, V \rangle$ if $V^{\mathcal{D}} \subseteq r^{\mathcal{B}}$. We say that $\mathcal{B}$ satisfies the mapping $\mathcal{M}$ wrt $\mathcal{D}$ if $\mathcal{B}$ satisfies every assertion in $\mathcal{M}$.

It is immediate to see that, by simply evaluating each view over the source database $\mathcal{D}$, we obtain a global database that actually satisfies the sound mapping. We call such database the *retrieved global database* $ret(\mathcal{I}, \mathcal{D})$.

Notice that, in general, $ret(\mathcal{I}, \mathcal{D})$ might not satisfy the constraints on the global schema. Furthermore, according to the above definition of mapping satisfaction, it could happen that no global database exists that is legal for $\mathcal{I}$ wrt $\mathcal{D}$, because data retrieved from the sources cannot be completely reconciled in the global schema in such a way that both the constraints on the global schema and the mapping are satisfied. For example, this happens when a key constraint specified in the global schema is violated by the tuples retrieved from the sources, since the assumption of sound views does not allow us to disregard any tuple from the retrieved global database.

A more general approach would be to provide a formalization that is able to support query processing even when the data at the sources are incoherent with respect to the integrity constraints on the global schema. The basic idea is to consider those global databases that satisfy the integrity constraints in the global schema and that approximate the satisfaction of the mapping $\mathcal{M}$, i.e., that are *as sound as possible*. One way to formalize this idea is to distinguish between *strictly-sound* mappings, as the ones considered before, and *loosely-sound* mappings, in which the assumption of soundness is suitably relaxed. Then, given a source database $\mathcal{D}$ and a mapping $\mathcal{M} = \{\langle r_1, V_1\rangle \ \ldots \ \langle r_n, V_n\rangle\}$, we define an ordering over the global databases for $\mathcal{I}$ that are coherent with $\mathcal{G}$.

If $\mathcal{B}_1$ and $\mathcal{B}_2$ are two such databases, we say that $\mathcal{B}_1$ is *better* than $\mathcal{B}_2$ wrt $\mathcal{D}$, denoted as $\mathcal{B}_1 \gg_{\mathcal{D}} \mathcal{B}_2$, iff there exists $i \in \{1, \ldots, n\}$ such that

- $(r_i^{\mathcal{B}_1} \cap V_i^{\mathcal{D}}) \supset (r_i^{\mathcal{B}_2} \cap V_i^{\mathcal{D}})$, and
- $(r_j^{\mathcal{B}_1} \cap V_j^{\mathcal{D}}) \supseteq (r_j^{\mathcal{B}_2} \cap V_j^{\mathcal{D}})$ for $j = 1, \ldots, n$;

Intuitively, this means that the portion of $ret(\mathcal{I}, \mathcal{D})$ contained in the global database is greater in $\mathcal{B}_1$ than in $\mathcal{B}_2$, i.e., $\mathcal{B}_1$ approximates the sound mapping better than $\mathcal{B}_2$. It is easy to verify that the relation $\gg_{\mathcal{D}}$ is a partial order. With this notion in place, we say that a global database $\mathcal{B}$ coherent with $\mathcal{G}$ satisfies the mapping if $\mathcal{B}$ is maximal wrt $\gg_{\mathcal{D}}$, i.e., for no other global database $\mathcal{B}'$ coherent with $\mathcal{G}$, we have that $\mathcal{B}' \gg_{\mathcal{D}} \mathcal{B}$. Hence, with reference to the definition given in Section 2, a global database for $\mathcal{I}$ is said to be *legal* wrt $\mathcal{D}$, if $\mathcal{B}$ is coherent with $\mathcal{G}$ and is maximal wrt $\gg_{\mathcal{D}}$.

Example 2 Let $E = \langle \mathcal{G}^E, \mathcal{S}^E, \mathcal{M}^E \rangle$ be a data integration system, in which $\mathcal{G}^E$ consists of the relation predicate $\mathtt{g}(X, Y)$, $\mathcal{S}^E$ consists of two relations $\mathtt{s}_1$ and $\mathtt{s}_2$, both of arity two, and $\mathcal{M}^E$ is defined by the single assertion $\langle \mathtt{g}, \mathtt{g}(X,Y) \leftarrow \mathtt{s}_1(X,Y) \vee \mathtt{s}_2(X,Y)\rangle$. Suppose that we have an alphabet containing (among other symbols) the constants a, b, c, d, e, f, g, and that $\mathcal{D}$ is a source database such that $\mathtt{s}_1^{\mathcal{D}} = \{(a,b),(c,d)\}$ and $\mathtt{s}_2^{\mathcal{D}} = \{(a,e)\}$. Finally, assume that the only integrity constraint on $\mathcal{G}^E$ is a key constraint stating that X is the key of $\mathtt{g}$.

Under such assumptions we have that $g^{ret(E,\mathcal{D})} = \{(a,b),(c,d),(a,e)\}$, and that the tuples (a,b) and (a,e) violate the key constraint on g. It is easy to see that the following global databases are three examples of legal databases for E under the loosely-sound semantics:

- $\mathcal{B}_1$ such that $\mathtt{g}^{\mathcal{B}_1} = \{(a,b),(c,d)\}$,
- $\mathcal{B}_2$ such that $\mathtt{g}^{\mathcal{B}_2} = \{(a,e),(c,d)\}$,

- $\mathcal{B}_3$ such that $\mathbf{g}^{\mathcal{B}_3} = \{(a, e), (c, d), (f, g)\}$,

while the following global databases are not legal for E:

- $\mathcal{B}_4$ such that $\mathbf{g}^{\mathcal{B}_4} = \{(a, e), (a, b)\}$, since it violates the key constraint;
- $\mathcal{B}_5$ such that $\mathbf{g}^{\mathcal{B}_5} = \{(a, e), (f, g)\}$, since we have that $\mathcal{B}_3 \gg_{\mathcal{D}} \mathcal{B}_5$.

■

It is immediate to verify that, if there exists a legal database for $\mathcal{I}$ wrt $\mathcal{D}$ under the strictly-sound semantics, then the strictly-sound and the loosely-sound semantics coincide, in the sense that the set of legal databases is the same for the two semantics.

4 Query Processing

In this section we consider the particular situation in which two kinds of constraints are specified in the global schema:

- *Key constraints*: given a relation r in the schema, a key constraint over r is expressed in the form $key(r) = \mathbf{X}$, where $\mathbf{X}$ is a set of attributes of r. We assume that there is exactly one key constraint for each relation. Such a constraint is satisfied in a global database $\mathcal{B}$ if for each $t_1, t_2 \in r^{\mathcal{B}}$ with $t_1 \neq t_2$ we have $t_1[\mathbf{X}] \neq t_2[\mathbf{X}]$, where $t[\mathbf{X}]$ is the projection of the tuple t over $\mathbf{X}$;
- *Foreign key constraints*: a foreign key constraint is a statement of the form $r_1[\mathbf{X}] \subseteq r_2[\mathbf{Y}]$, where r_1 and r_2 are relations, $\mathbf{X}$ is a sequence of distinct attributes of r_1, and $\mathbf{Y}$ is a sequence formed by the distinct attributes forming the key of r_2. Such a constraint is satisfied in a global database $\mathcal{B}$ if for each tuple t_1 in $r_1^{\mathcal{B}}$ there exists a tuple t_2 in $r_2^{\mathcal{B}}$ such that $t_1[\mathbf{X}] = t_2[\mathbf{Y}]$.

The problem of query answering in this setting has been recently studied under the strictly-sound semantics of the mapping. In particular, it is possible to compute the certain answers of a conjunctive query q over the global schema by means of a query reformulation algorithm, presented in [5, 6], that transforms the original query q into a new query $exp_{\mathcal{G}}(q)$ over the global schema. Roughly, the algorithm is based on the idea of expressing foreign key constraints in terms of rules of a logic program with functional symbols (used as Skolem functions). Then, such a logic program is used to generate the *partial evaluation tree* of the query q, whose non-empty leaves constitute the reformulation $exp_{\mathcal{G}}(q)$ of the query q. Notably, $exp_{\mathcal{G}}(q)$ is independent of the source database $\mathcal{D}$; moreover, the algorithm has polynomial data complexity.

We now study the problem of computing certain answers to a query in this setting under a loosely-sound semantics of the mapping. The difference with respect to the previous case can be seen in a situation in which there exists no global database that both is coherent with $\mathcal{G}$ and satisfies the mapping wrt $\mathcal{D}$: in our setting, this corresponds to a case in which $ret(\mathcal{I}, \mathcal{D})$ violates the key constraints in $\mathcal{G}$, i.e., there exist $r \in \mathcal{G}$ and $t_1, t_2 \in r^{ret(\mathcal{I}, \mathcal{D})}$ such that $key(r) = \mathbf{X}$, $t_1[\mathbf{X}] = t_2[\mathbf{X}]$, and $t_1 \neq t_2$.

Under the strictly-sound semantics, this means that there are no legal databases for $\mathcal{I}$ wrt $\mathcal{D}$.

Conversely, under a loosely-sound semantics, it is immediate to see that there always exists a legal database for $\mathcal{I}$ wrt $\mathcal{D}$, because we are allowed to eliminate tuples from $ret(\mathcal{I},\mathcal{D})$ in order to satisfy the constraints, and key constraints and foreign key constraints can always be satisfied by suitably restricting the set of tuples in the database. However, the semantics implies that the legal databases are the ones that are "as sound as possible", thus we have to consider only databases coherent with the constraints and that "minimize" elimination of tuples from $ret(\mathcal{I},\mathcal{D})$. Notice that, in order to satisfy key constraints, we are forced to eliminate tuples from $ret(\mathcal{I},\mathcal{D})$, while, in general, foreign key constraints can be satisfied either by adding new tuples or by eliminating tuples from $ret(\mathcal{I},\mathcal{D})$: by the loosely-sound semantics, such constraints must be satisfied by adding new tuples.

Our method for computing the certain answers to a query q can be informally explained as follows: we identify the databases corresponding to the legal databases for $\mathcal{I}'$ wrt $\mathcal{D}$, where $\mathcal{I}'$ is obtained from $\mathcal{I}$ by eliminating all foreign key constraints in $\mathcal{G}$. It is immediate to see that each such database $\mathcal{B}'$ is "contained" in $ret(\mathcal{I},\mathcal{D})$, i.e. if $t \in r^{\mathcal{B}'}$ then $t \in r^{ret(\mathcal{I},\mathcal{D})}$ for each t and for each r. Then, we make use of the query reformulation technique for the strictly-sound semantics mentioned above [6], since it can be shown that t is a certain answer of q wrt $\mathcal{I}$ and $\mathcal{D}$ iff $t \in (exp_{\mathcal{G}}(q))^{\mathcal{B}'}$ for each database $\mathcal{B}'$ for $\mathcal{I}'$ that is legal wrt $\mathcal{D}$.

Specifically, we resort to DATALOG$^\neg$ under stable model semantics [10, 8], a well-known extension of DATALOG that allows for using negation in the body of program rules. In particular, we define a DATALOG$^\neg$ program $P(\mathcal{I},\mathcal{D})$ that allows for computing the legal databases for $\mathcal{I}'$ wrt $\mathcal{D}$. The DATALOG$^\neg$ program $P(\mathcal{I},\mathcal{D})$ is obtained by adding to the set of facts $\mathcal{D}$ the following set of rules:

- for each assertion $\langle r, V\rangle \in \mathcal{M}$, with

$$V = r(\vec{\mathbf{x}}) \;\leftarrow\; conj_1(\vec{\mathbf{x}},\vec{\mathbf{y}}_1) \vee \cdots \vee conj_m(\vec{\mathbf{x}},\vec{\mathbf{y}}_m)$$

  the rules

$$\begin{aligned} r_{\mathcal{D}}(\vec{\mathbf{x}}) &\leftarrow conj_1(\vec{\mathbf{x}},\vec{\mathbf{y}}_1) \\ &\cdots \\ r_{\mathcal{D}}(\vec{\mathbf{x}}) &\leftarrow conj_m(\vec{\mathbf{x}},\vec{\mathbf{y}}_m) \end{aligned}$$

- for each relation $r \in \mathcal{G}$, the rules

$$\begin{aligned} r(\vec{\mathbf{x}},\vec{\mathbf{y}}) &\leftarrow r_{\mathcal{D}}(\vec{\mathbf{x}},\vec{\mathbf{y}}) \,,\; not\, \overline{r}(\vec{\mathbf{x}},\vec{\mathbf{y}}) \\ \overline{r}(\vec{\mathbf{x}},\vec{\mathbf{y}}) &\leftarrow r(\vec{\mathbf{x}},\vec{\mathbf{z}}) \,,\; \vec{\mathbf{y}} \neq \vec{\mathbf{z}} \end{aligned}$$

  where in $r(\vec{\mathbf{x}},\vec{\mathbf{y}})$ the variables in $\vec{\mathbf{x}}$ correspond to the attributes constituting the key of the relation r, and $\vec{\mathbf{y}} \neq \vec{\mathbf{z}}$ if there exists i such that $Y_i \neq Z_i$.

Informally, for each relation r, $P(\mathcal{I},\mathcal{D})$ contains (i) a relation $r_{\mathcal{D}}$ that represents $r^{ret(\mathcal{I},\mathcal{D})}$; (ii) a relation r that represents a subset of $r^{ret(\mathcal{I},\mathcal{D})}$ that is consistent with

the key constraint for r; (iii) an auxiliary relation $\overline{r}$. The above rules force each stable model M of $P(\mathcal{I},\mathcal{D})$ to be such that r^M is a maximal subset of tuples from $r^{ret(\mathcal{I},\mathcal{D})}$ that are consistent with the key constraint for r.

Then, we add the $exp_{\mathcal{G}}(q)$ to the program $P(\mathcal{I},\mathcal{D})$, thus obtaining a DATALOG$^\neg$ program that allows us to compute the certain answers to the original query.

Theorem 4.1 *Let $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$ be a data integration system, let q be a query posed to $\mathcal{I}$, $\mathcal{D}$ be a source database for $\mathcal{I}$, and t be a tuple of constants of the same arity as q. Then, $t \in q^{\mathcal{I},\mathcal{D}}$ if and only if $t \in q^M$ for each stable model M of the* DATALOG$^\neg$ *program $P(\mathcal{I},\mathcal{D}) \cup \{exp_{\mathcal{G}}(q)\}$.*

Finally, we are able to give a computational characterization of the problem of computing certain answers to queries in our data integration setting (i.e., loosely-sound mapping, key constraints and foreign key constraints).

Theorem 4.2 *Let $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$ be a data integration system, let q be a query posed to $\mathcal{I}$, $\mathcal{D}$ be a source database for $\mathcal{I}$, and t be a tuple of constants of the same arity as q. The problem of deciding whether $t \in q^{\mathcal{I},\mathcal{D}}$ is coNP-complete wrt data complexity.*

Proof sketch. Membership in coNP follows from the Theorem 4.1, and from the fact that query answering in DATALOG$^\neg$ is coNP-complete in data complexity, while coNP-hardness can be easily proven by a reduction of the validity problem in propositional logic to the above problem.

Example 1 (cont.) We assume now that the following constraints are defined in the global schema:

$$\begin{array}{rcl} key(\mathsf{player}) & = & \{Pcode\} \\ key(\mathsf{team}) & = & \{Tcode\} \\ \mathsf{player}[Pteam] & \subseteq & \mathsf{team}[Tcode] \\ \mathsf{team}[Tleader] & \subseteq & \mathsf{player}[Pcode]. \end{array}$$

Moreover, we consider the extension $\mathcal{D}$ for the sources shown in Figure 1.

$\mathsf{s}_1^{\mathcal{D}}$:

9	*Batistuta*	*RM*	31
10	*Rivaldo*	*BC*	29

$\mathsf{s}_2^{\mathcal{D}}$:

RM	*Roma*	10
BC	*Barcelona*	8

$\mathsf{s}_3^{\mathcal{D}}$:

RM	*Roma*	9

Figure 1: Data at the sources in the example

The extension of the source s_2 and the foreign key constraint $\mathsf{team}[Tleader] \subseteq \mathsf{player}[Pcode]$ impose that 8 is the code of some player. Nonetheless, the source s_1 does not say anything about the name and the team of such player. Since the views are loosely-sound, we can consider as legal databases for the data integration system all the global databases that provide the name and the team of the player.

Furthermore, the tuples stored in the source s_2 together with the tuple in the source s_3 violate the key constraint $key(\mathsf{team}) = \{Tcode\}$. According to the loosely-sound semantics, for all the legal global databases $\mathcal{B}$ we have that $\mathsf{team}^{\mathcal{B}}$ contains either the tuple $\langle RM, Roma, 9\rangle$ or the tuple $\langle RM, Roma, 10\rangle$.

Suppose now that the user query is the following: $\mathsf{q}(X) \leftarrow \mathsf{player}(X, Y, Z)$. By evaluating the partial evaluation tree of q, according to the procedure presented in [6], the following expansion of the query q is obtained:

$$\begin{aligned} \mathsf{q}(X) &\leftarrow \mathsf{player}(X, Y, Z) \\ \mathsf{q}(X) &\leftarrow \mathsf{team}(W_1, W_2, X) \end{aligned}$$

Intuitively, we see that the expanded query searches for player codes not only in the relation player, but also in team, where, due to the foreign key constraint $\mathsf{team}[Tleader] \subseteq \mathsf{player}[Pcode]$, it is known that player codes are stored.

By the above Theorem 4.1, we obtain the DATALOG$^\neg$ program $P \cup exp_{\mathcal{G}}(q)$ for computing the certain answers of q. Such a program contains the facts in $\mathcal{D}$ and the following rules:

$$\begin{aligned} \mathsf{player}_{\mathcal{D}}(X, Y, Z) &\leftarrow \mathsf{s}_1(X, Y, Z, W) \\ \mathsf{team}_{\mathcal{D}}(X, Y, Z) &\leftarrow \mathsf{s}_2(X, Y, Z) \\ \mathsf{team}_{\mathcal{D}}(X, Y, Z) &\leftarrow \mathsf{s}_3(X, Y, Z) \\ \mathsf{player}(X, Y, Z) &\leftarrow \mathsf{player}_{\mathcal{D}}(X, Y, Z)\ ,\ \mathit{not}\ \overline{\mathsf{player}}(X, Y, Z) \\ \overline{\mathsf{player}}(X, Y, Z) &\leftarrow \mathsf{player}(X, V, W)\ ,\ Y \neq V \\ \overline{\mathsf{player}}(X, Y, Z) &\leftarrow \mathsf{player}(X, V, W)\ ,\ Z \neq W \\ \mathsf{team}(X, Y, Z) &\leftarrow \mathsf{team}_{\mathcal{D}}(X, Y, Z)\ ,\ \mathit{not}\ \overline{\mathsf{team}}(X, Y, Z) \\ \overline{\mathsf{team}}(X, Y, Z) &\leftarrow \mathsf{team}(X, V, W)\ ,\ Y \neq V \\ \overline{\mathsf{team}}(X, Y, Z) &\leftarrow \mathsf{team}(X, V, W)\ ,\ Z \neq W \\ \mathsf{q}(X) &\leftarrow \mathsf{player}(X, Y, Z) \\ \mathsf{q}(X) &\leftarrow \mathsf{team}(W_1, W_2, X) \end{aligned}$$

By computing the stable models of the above program, it can be seen that the set of certain answers to the query q is $\{8, 9, 10\}$.

■

5 Related Work

Several recent works deal with the problem of database inconsistency. For example, in [4] the notion of *consistent query answers* is defined, which can be seen as a specialization to the case of a single database of our notion of *certain answers* given in Section 2. No method for computing consistent answers is provided.

In [11] the authors describe an operator for *merging databases* under constraints which allows one to obtain a maximal amount of information from each database by means of a majority criterion used in case of conflict. Even if a large set of constraints is considered, namely the constraints that can be expressed as first-order formulas, the computational complexity of the merging procedure is not explored, and no algorithm to compute consistent query answers is provided. Furthermore, the problem of dealing with incomplete databases is not taken into account. Notice also that, differently from all the other studies mentioned in the following, this approach relies on a cardinality based ordering between databases (rather than on a set containment one).

In [2] the authors define an algorithm for consistent query answers in inconsistent databases based on the notion of *residues*, originally defined in the context of semantic query optimization. The method is proved to be sound and complete only for the class of universally quantified binary constraints, i.e., constraints that involve two database relations. In [3] the same authors propose a new method that can handle arbitrary universally quantified constraints by specifying the database repairs into *logic rules with exceptions* (LPe).

Finally, [9] proposes a technique to deal with inconsistencies that is based on the reformulation of integrity constraints into a disjunctive DATALOG program with two different forms of negation: negation as failure and classical negation. Such a program can be used both to repair databases, i.e., modify the data in the databases in order to make the integrity constraints be satisfied, and to compute consistent query answers. The technique is proved to be sound and complete for universally quantified constraints.

It is worth noticing that none of the above works provides solutions for the case of existentially quantified constraints, as, for example, the foreign key constraints that we have treated in Section 4. To the best of our knowledge, our method is the first one to deal with foreign key constraints.

Moreover, the studies concerning data inconsistencies basically apply to a single database setting. With regard to the works that deal with inconsistencies in database integration, the activity of integrating data is commonly conceived as a two-phase process, where the first one aims at populating the global schema, and the second one is intended to repair possible inconsistencies among data. Hence, traditional approaches implicitly interpret the mapping as an exact one. In this sense, our approach is more general, since it is based on the semantics of the mapping, which we have assumed to be loosely-sound.

6 Discussion and Conclusions

As we already said in the above section, several semantics underlying the notion of consistent query answers have been proposed in the literature. Consider, for example, the semantics defined in [11], which is based on the notion of cardinality based ordering between databases, or the semantics presented in [2, 3, 9], which, on the contrary, relies on an ordering based on set containment.

As in these last works, the loosely-sound semantics proposed in this paper is based on set containment between global databases. According to such an ordering, given a data integration system $\mathcal{I}$ and a source database $\mathcal{D}$, a global database $\mathcal{B}_1$ is said to be better than a global database $\mathcal{B}_2$ if the portion of $ret(\mathcal{I}, \mathcal{D})$ contained in $\mathcal{B}_2$ is a subset of the portion of $ret(\mathcal{I}, \mathcal{D})$ contained in $\mathcal{B}_1$. In other words, coherently with the notion of sound mapping, in order to prove if $\mathcal{B}_1$ is better than $\mathcal{B}_2$ we do not take into account the tuples of $\mathcal{B}_1$ and $\mathcal{B}_2$ that are not in $ret(\mathcal{I}, \mathcal{D})$.

However, this is only one of the several semantics that can be defined to deal with inconsistent and incomplete data. For instance, our definition of ordering between databases could be refined to the aim of comparing also global databases that differ

only for the tuples that are not in $ret(\mathcal{I}, \mathcal{D})$, and we could consider a new semantics that relies on this different definition. In this case, if $\mathcal{B}_1$ and $\mathcal{B}_2$ are two databases coherent with the global schema, we say that $\mathcal{B}_1$ is *better* than $\mathcal{B}_2$ wrt $\mathcal{D}$, as usually denoted as $\mathcal{B}_1 \gg_{\mathcal{D}} \mathcal{B}_2$, iff one of the two following conditions holds:

1. there exists $i \in \{1, \ldots, n\}$ such that
   - $(r_i^{\mathcal{B}_1} \cap V_i^{\mathcal{D}}) \supset (r_i^{\mathcal{B}_2} \cap V_i^{\mathcal{D}})$, and
   - $(r_j^{\mathcal{B}_1} \cap V_j^{\mathcal{D}}) \supseteq (r_j^{\mathcal{B}_2} \cap V_j^{\mathcal{D}})$ for $j = 1, \ldots, n$;
2. $(r_k^{\mathcal{B}_1} \cap V_k^{\mathcal{D}}) = (r_k^{\mathcal{B}_2} \cap V_k^{\mathcal{D}})$ for $k = 1, \ldots, n$ and there exists $i \in \{1, \ldots, n\}$ such that
   - $(r_i^{\mathcal{B}_1} - V_i^{\mathcal{D}}) \subset (r_i^{\mathcal{B}_2} - V_i^{\mathcal{D}})$, and
   - $(r_j^{\mathcal{B}_1} - V_j^{\mathcal{D}}) \subseteq (r_j^{\mathcal{B}_2} - V_j^{\mathcal{D}})$ for $j = 1, \ldots, n$.

Intuitively, this means that $\mathcal{B}_1 \gg_{\mathcal{D}} \mathcal{B}_2$ iff $\mathcal{B}_1$ is "closer" to $ret(\mathcal{I}, \mathcal{D})$ than $\mathcal{B}_2$. With reference to Example 2, it is easy to see that, according to the new definition of the semantics, the global database $\mathcal{B}_3$ is not legal for E, since now we have that $\mathcal{B}_2 \gg_{\mathcal{D}} \mathcal{B}_3$, and the only global legal databases are $\mathcal{B}_1$ and $\mathcal{B}_2$.

If $sem_1(\mathcal{I}, \mathcal{D})$ is the semantics of a data integration system based on the ordering defined in Section 3, and $sem_2(\mathcal{I}, \mathcal{D})$ is the semantics based on the refined ordering, it can be shown that

$$sem_2(\mathcal{I}, \mathcal{D}) = \{ \mathcal{B} \mid \mathcal{B} \in sem_1(\mathcal{I}, \mathcal{D}) \text{ and for each } \mathcal{B}' \in sem_1(\mathcal{I}, \mathcal{D}) \ \mathcal{B}' \not\subset \mathcal{B}\}$$

Hence, $sem_2(\mathcal{I}, \mathcal{D})$ is the set of the minimal databases of $sem_1(\mathcal{I}, \mathcal{D})$. Therefore, for each query q, the certain answers to q are the same in the two semantics, hence the method proposed in Section 4 to compute the set of certain answers to q when key and foreign key constraints are expressed over the global schema can be used also when we adopt $sem_2(\mathcal{I}, \mathcal{D})$ instead of $sem_1(\mathcal{I}, \mathcal{D})$. The situation is different if we are interested in computing the *possible answers* to a query q, i.e., the set of n-tuples t such that $t \in q^{\mathcal{DB}}$ for *some* database $\mathcal{DB} \in sem(\mathcal{I}, \mathcal{D})$, where n is the arity of q. In general, the set of possible answers to q according to $sem_2(\mathcal{I}, \mathcal{D})$ is contained in the set of possible answers according to $sem_1(\mathcal{I}, \mathcal{D})$.

This example illustrates that the approach described in this paper is general enough to be easily extended beyond the setting presented here. In particular, other different assumptions (complete, exact) on the nature of the mapping between the sources and the global schema can be captured. Furthermore, we believe that we can generalize our approach to also take into account LAV mappings.

On the other hand, in order to cope with the high computational complexity of query processing in the described setting, we are investigating possible restrictions of our framework to obtain particular and practical cases that allow for more efficient query processing.

References

[1] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases.* Addison Wesley Publ. Co., Reading, Massachussetts, 1995.

[2] Marcelo Arenas, Leopoldo E. Bertossi, and Jan Chomicki. Consistent query answers in inconsistent databases. In *Proc. of the 18th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS'99)*, pages 68–79, 1999.

[3] Marcelo Arenas, Leopoldo E. Bertossi, and Jan Chomicki. Specifying and querying database repairs using logic programs with exceptions. In *Proc. of the 4th Int. Conf. on Flexible Query Answering Systems (FQAS'00)*, pages 27–41. Springer, 2000.

[4] François Bry. Query answering in information systems with integrity constraints. In *IFIP WG 11.5 Working Conf. on Integrity and Control in Information System.* Chapman & Hall, 1997.

[5] Andrea Calì, Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. Accessing data integration systems through conceptual schemas. In *Proc. of the 20th Int. Conf. on Conceptual Modeling (ER 2001)*, 2001.

[6] Andrea Calì, Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. Data integration under integrity constraints. In *Proc. of the 14th Conf. on Advanced Information Systems Engineering (CAiSE 2002)*, 2002. To appear.

[7] Phan Minh Dung. Integrating data from possibly inconsistent databases. In *Proc. of the 4th Int. Conf. on Cooperative Information Systems (CoopIS'96)*, pages 58–65, 1996.

[8] Thomas Eiter, Georg Gottlob, and Heikki Mannilla. Disjunctive Datalog. *ACM Trans. on Database Systems*, 22(3):364–418, 1997.

[9] Gianluigi Greco, Sergio Greco, and Ester Zumpano. A logic programming approach to the integration, repairing and querying of inconsistent databases. In *Proc. of the 17th Int. Conf. on Logic Programming (ICLP'01)*, volume 2237 of *Lecture Notes in Artificial Intelligence*, pages 348–364. Springer, 2001.

[10] Phokion G. Kolaitis and Christos H. Papadimitriou. Why not negation by fixpoint? *J. of Computer and System Sciences*, 43(1):125–144, 1991.

[11] Jinxin Lin and Alberto O. Mendelzon. Merging databases under constraints. *Int. J. of Cooperative Information Systems*, 7(1):55–76, 1998.

[12] Ron van der Meyden. Logical approaches to incomplete information. In Jan Chomicki and Günter Saake, editors, *Logics for Databases and Information Systems*, pages 307–356. Kluwer Academic Publisher, 1998.