Consistent Queries over Databases with Integrity Constraints

Luciano Caroprese
DEIS Universitá della Calabria, Italy

Cristian Molinaro
DEIS Universitá della Calabria, Italy

Irina Trubitsyna
DEIS Universitá della Calabria, Italy

Ester Zumpano
DEIS Universitá della Calabria, Italy

INTRODUCTION

Integrating data from different sources consists of two main steps, the first in which the various relations are merged together, and the second in which some tuples are *removed* (or *inserted*) from the resulting database in order to satisfy integrity constraints. There are several ways to integrate databases or possibly distributed information sources, but whatever integration architecture we choose, the heterogeneity of the sources to be integrated causes subtle problems. In particular, the database obtained from the integration process may be inconsistent with respect to integrity constraints, that is, one or more integrity constraints are not satisfied. Integrity constraints represent an important source of information about the real world. They are usually used to define constraints on data (functional dependencies, inclusion dependencies, etc.) and have, nowadays, a wide applicability in several contexts such as semantic query optimization, cooperative query answering, database integration, and view update.

Since the satisfaction of integrity constraints cannot generally be guaranteed, if the database is obtained from the integration of different information sources, in the evaluation of queries, we must compute answers that are consistent with the integrity constraints. The following example shows a case of inconsistency.

Example 1: Consider the following database schema consisting of the single binary relation *Teaches (Course, Professor)* where the attribute *Course* is a key for the relation. Assume there are two different instances for the relations *Teaches*, D1={(c1,p1),(c2,p2)} and D2={(c1,p1),(c2,p3)}. The two instances satisfy the constraint that *Course* is a key, but from their union we derive a relation that does not satisfy the constraint since there are two distinct tuples with the same value for the attribute *Course*.

In the integration of two conflicting databases simple solutions could be based on the definition of preference criteria such as a partial order on the source information or a majority criterion (Lin & Mendelzon, 1996). However, these solutions are not generally satisfactory, and more useful solutions are those based on (1) the computation of "repairs" for the database, and (2) the computation of consistent answers (Arenas, Bertossi, & Chomicki, 1999).

The computation of repairs is based on the definition of minimal sets of insertion and deletion operations so that the resulting database satisfies all constraints. The computation of consistent answers is based on the identification of tuples satisfying integrity constraints and on the selection of tuples matching the goal. For instance, for the integrated database of *Example 1*, we have two alternative repairs consisting in the deletion of one of the tuples *(c2,p2)* and *(c2,p3)*. The consistent answer to a query over the relation *Teaches* contains the unique tuple *(c1,p1)* so that we do not know which professor teaches course *c2*. Therefore, it is very important, in the presence of inconsistent data, not only to compute the set of consistent answers, but also to know which facts are unknown and if there are possible repairs for the database.

BACKGROUND

Several proposals considering the integration of databases as well as the computation of queries over inconsistent databases have been provided in the literature (Agarwal, Keller, Wiederhold, & Saraswat, 1995; Arenas et al., 1999; Arenas, Bertossi, & Chomicki, 2000; Bry, 1997; Dung, 1996; Greco & Zumpano, 2000; Lin, 1996; Lin & Mendelzon, 1996; Lembo, Lenzerini, & Rosati, 2002; Lenzerini, 2002; Wijsen, 2003). Most of the techniques for computing queries over inconsistent databases work for restricted cases, and only

recently have there been proposals to consider more general constraints. This section provides an informal description of the main techniques proposed in the literature.

- Lin and Mendelzon (1996) proposed an approach taking into account the majority view of the knowledge bases in order to obtain a new relation that is consistent with the integrity constraints. The technique proposes a formal semantics to merge first order theories under a set of constraints.
  However, the "merging by majority" technique does not resolve conflicts in all cases since information is not always present in the majority of the databases, and, therefore, it is not always possible to choose between alternative values. Moreover, the use of the majority criteria involves discarding inconsistent data and hence the loss of potentially useful information.
- Arenas et al. (1999) introduced a logical characterization of the notion of consistent answer in a possibly inconsistent database. The technique is based on the computation of an equivalent query $T_\omega(Q)$ derived from the source query Q. The definition of $T_\omega(Q)$ is based on the notion of residue developed in the context of semantic query optimization.
  More specifically, for each literal B, appearing in some integrity constraint, a residue *Res(B)* is computed. Intuitively, *Res(B)* is a universal quantified first order formula that must be true, because of the constraints, if B is true.
  The technique, more general than the previous ones, has been shown to be complete for universal binary integrity constraints and universal quantified queries. However, the rewriting of queries is complex since the termination conditions are not easy to detect and the computation of answers generally is not guaranteed to be polynomial.
- Arenas et al. (2000) proposed an approach consisting in the use of a Logic Program with exceptions (LPe) for obtaining consistent query answers. An LPe is a program with the syntax of an extended logic program (ELP), that is, in it we may find both logical (or strong) negation ($\neg$) and procedural negation (not). In this program, rules with a positive literal in the head represent a sort of general default, whereas rules with a logically negated head represent exceptions. The semantic of an LPe is obtained from the semantics for ELPs, by adding extra conditions that assign higher priority to exceptions. The method, given a set of integrity constraints ICs and an inconsistent database instance, consists in the direct specification of database repairs in a logic programming formalism. The resulting program will have both negative and positive exceptions, strong and procedural negations, and disjunctions of literals in the head of some of the clauses, that is, it will be a disjunctive extended logic program with exceptions. As shown by Arenas et al. (1999), the method considers a set of integrity constraints, IC, written in the standard format $\vee_{i=1}^{n} P_i(x_i) \vee \vee_{i=1}^{m} (\neg Q_i(y_i) \vee \varphi$, where φ is a formula containing only built-in predicates, and there is an implicit universal quantification in front. This method specifies the repairs of the database, D, that violate IC, by means of a logical program with exceptions, Π^D. In Π^D, for each predicate P a new predicate P' is introduced, and each occurrence of P is replaced by P'.
  The method can be applied to a set of domain independent binary integrity constraints *IC*, that is, the constraint can be checked w.r.t. satisfaction by looking to the active domain, and in each *IC* appear at most two literals.
- Calì, Calvanese, De Giacomo, and Lenzerini (2002), Lembo et al. (2002), and Lenzerini (2002) proposed a framework for data integration that allows to specify a general form of integrity constraints over the global schema, and it is defined a semantics for data integration in the presence of incomplete and inconsistent information sources. Moreover, it is defined as a method for query processing under the previous semantics when key constraints and foreign key constraints are defined upon the global schema.
  Formally, a data integration system I is a triple <G, S, $M_{G,S}$>, where G is the global schema, S is the source schema, and $M_{G,S}$ is the mapping between G and S. More specifically, the *global schema* is expressed in the relational model with both key and foreign key constraints; the *source schema* is expressed in the relational model without integrity constraints; and the *mapping* is defined between the global and the source schema, that is, each relation in G is associated with a view, that is, a query over the sources. The semantics of a data integration system is given by considering a source database D for I, that is, a database for the source schema S containing relation r^D for each source r in S.
  Any database G is a *global database* for I, and it is said *legal* w.r.t. D if:
  - It satisfies the integrity constraints defined on G.
  - It satisfies the mapping w.r.t. D, that is, for each relation r in G, the set of tuples r^B that B assigns to r is a subset of the set of tuples $\rho(r)^D$ computed by the associated query $\rho(r)$ over D: $\rho(r)^D \subseteq r^B$.

  In this framework, the semantics of I w.r.t. a source database D, denoted $sem^D(I, D)$, is given in terms of a set of databases. In particular, $sem^D(I, D) = \{ B \mid B$ *is a legal global database for I, w.r.t. D}*. If $sem^D(I, D) \neq \emptyset$, then I is said to be consistent w.r.t. D.