Reasoning in Data Integration Systems: why LAV and GAV are Siblings

Andrea Calì

Dipartimento di Informatica e Sistemistica
Università di Roma "La Sapienza"
Via Salaria 113, I-00198 Roma, Italy
cali@dis.uniroma1.it

Abstract. Data integration consists in providing a uniform access to a set of data sources, through a unified representation of the data called *global schema*; a *mapping* specifies the relationship between the global schema and the sources. Integrity constraints (ICs) are expressed on the global schema to better represent the domain of interest; in general, ICs are not satisfied by the data at the sources. In this paper we address the problem of query answering in GLAV data integration systems, where tuple-generating dependencies are expressed on the global schema. We solve the problem in an intensional fashion, by presenting a rewriting technique that, taking into account both the ICs and the mapping, allows us to compute the answers to a query, expressed over the global schema, by evaluating the rewritten query directly over the sources. Since the GLAV approach is a generalisation of the basic approaches LAV and GAV, we show that query answering under ICs can be done in the same way in LAV and GAV systems, thus proving that LAV and GAV are siblings, and not opposites.

1 Introduction

The task of a data integration system is to provide the user with a unified view, called *global schema*, of a set of heterogeneous data sources. Once the user issues a query over the global schema, the system carries out the task of suitably accessing the different sources and assemble the retrieved data into the final answer to the query. In this context, a crucial issue is the specification of the relationship between the global schema and the sources, which is called *mapping* [15]. There are two basic approaches for specifying a mapping in a data integration system [12, 15]. The first one, called *global-as-view (GAV)*, requires that to each element of the global schema a view over the sources is associated. The second approach, called *local-as-view (LAV)*, requires that to each source a view over the global schema is associated. Besides GAV and LAV, a mixed approach, called GLAV [9, 8], consists in associating views over the global schema to views over the sources.

Since the global schema is a representation of the domain of interest of the system, it needs to be represented by means of a flexible an expressive formalism: to this aim, *integrity constraints* are expressed on it. The data at the sources may not satisfy the constraints on the global schema; in this case a common assumption (which is the one adopted in this paper) is to to consider the sources as *sound*, i.e., they provide a *subset* of the data that satisfy the global schema.

In this paper we address the problem of data integration in the relational context, where the mapping is GLAV, and in the presence of *tuple-generating dependencies (TGDs)* on the

global schema; TGDs are an extension of *inclusion dependencies*, which are an important class of dependencies in database schemata [1, 13].

First, we generalise the notion of *retrieved global database (RGD)* [4] to GLAV systems, showing that in our case the treatment of integrity constraints is based on a *repairing* of the RGD, performed with a procedure called *chase* [13]. The correct answers to a query are obtained by evaluating the query over the repaired retrieved global database; from this perspective, LAV and GAV (and GLAV) systems are analogous with respect to the treatment of integrity constraints. Then, we present a technique for rewriting a query in such a way that the knowledge about integrity constraints is encoded in the rewritten query; in particular, the evaluation of the rewritten query over the retrieved global database returns the same answers as the evaluation of the original query over the repaired RGD: this allows us to avoid the repairing of the RGD. Since in the case of arbitrary TGDs the problem of query answering is undecidable [14], the technique applies to a restricted class of TGDs for which the problem is decidable. Finally, we present a query rewriting technique that takes the mapping into account: the GLAV data integration system is first compiled into an equivalent GAV one. At this point, once we have taken into account the integrity constraints in the previous rewriting step, we can proceed with the traditional rewriting technique for query processing in GAV, i.e., *unfolding*. In this way, we are able to avoid the construction of the RGD.

In conclusion, we are able to take into account the integrity constraints on the global schema and the mapping in two distinct rewriting steps, in a purely intensional fashion. The rewritten query can be thus evaluated on the data sources, providing the correct answers to the query.

2 Framework

In this section we define a logical framework for data integration, based on the relational model with integrity constraints.

Syntax We consider to have an infinite, fixed alphabet Γ of constants representing real world objects, and will take into account only databases having Γ as domain. Formally, a data integration system $\mathcal{I}$ is a triple $\langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, where:

1. $\mathcal{G}$ is the *global schema* expressed in the relational model with integrity constraints. In particular, $\mathcal{G} = \langle \Psi, \Sigma_T \rangle$, where: *(i)* Ψ is a set of relations, each with an associated arity that indicates the number of its attributes. A relation R of arity n is denoted by R/n. *(ii)* Σ_T is a set of *tuple-generating dependencies (TGDs)*. A TGD [1, 8] is a first-order formula of the form

$$\forall \boldsymbol{X} (\exists \boldsymbol{Y}\ \chi(\boldsymbol{X}, \boldsymbol{Y}) \ \rightarrow\ \exists \boldsymbol{Z}\ \psi(\boldsymbol{X}, \boldsymbol{Z}))$$

   where $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}$ are sets of variables, and χ and ψ are conjunctions of atoms whose predicate symbols are in Ψ. Henceforth, for the sake of conciseness, we will omit the quantifiers in TGDs.
2. $\mathcal{S}$ is the *source schema*, constituted by the schemata of the different sources. We assume that the sources are relational, and in particular each source is represented by a relation; furthermore, we assume that no integrity constraint is expressed on the source schema. Considering relational sources is not a restriction, since we may assume that sources that are not relational are suitably presented in relational form by software modules called *wrappers*.

3. $\mathcal{M}$ is the *mapping* between $\mathcal{G}$ and $\mathcal{S}$, specifying the relationship between the global schema and the source schema. The mapping $\mathcal{M}$ is a set of first-order formulae of the form

$$\forall \boldsymbol{X}(\exists \boldsymbol{Y} \varphi_{\mathcal{S}}(\boldsymbol{X}, \boldsymbol{Y}) \rightarrow \exists \boldsymbol{Z} \varphi_{\mathcal{G}}(\boldsymbol{X}, \boldsymbol{Z}))$$

where $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}$ are sets of variables or constants, and $\varphi_{\mathcal{S}}$ and $\varphi_{\mathcal{G}}$ are conjunctions of atoms whose predicate symbols are in $\mathcal{S}$ and Ψ respectively. Henceforth, we will omit quantifiers in mapping formulae. Note that this kind of mapping assertions is a generalisation of both LAV and GAV assertions; in particular, in a LAV assertion a view (conjunction of atoms) over the global schema is associated to a source relation, while in a GAV assertion a view over the source schema is associated to a relation symbol in Ψ. Henceforth, consistently with [9], we will call *GLAV (global-local-as-view)* this approach.

Now we come to queries expressed over the global schema; a n-ary *relational query* (relational query of arity n) is a formula that is intended to specify a set of n-tuples of constants in Γ, that constitute the *answer* to the query. In our setting, we assume that queries over the global schema are expressed in the language of *union of conjunctive queries (UCQs)*. A conjunctive query (CQ) of arity n is a formula of the form $q(\boldsymbol{X}) \leftarrow \omega(\boldsymbol{X}, \boldsymbol{Y})$ where $\boldsymbol{X}$ is a set of variables called *distinguished variables*, $\boldsymbol{Y}$ is a set of symbols that are either variables (called *non-distinguished*) or constants, q is a predicate symbol not appearing in Ψ or $\mathcal{S}$, and ω is a conjunction of atoms whose predicate symbols are in Ψ. The atom $q(\boldsymbol{X})$ is called *head* of the query (denoted $head(q)$), while $\omega(\boldsymbol{X}, \boldsymbol{Y})$ is called *body* (denoted $body(q)$). A UCQ of arity n is a set of conjunctive queries Q such that each $q \in Q$ has the same arity n and uses the same predicate symbol in the head.

Semantics A *database instance* (or simply *database*) $\mathcal{C}$ for a relational schema $\mathcal{R}$ is a set of facts of the form $R(t)$ where R is a relation of arity n in $\mathcal{R}$ and t is an n-tuple of constants of the alphabet Γ. We denote as $R^{\mathcal{C}}$ the set of tuples of the form $\{t \mid R(t) \in \mathcal{C}\}$.

In the following, we shall often make use of the notion of substitution. A *substitution* of variables σ is a partial function that associates to a variable either a constant or a variable, and to each constant the constant itself. Given a formula F and a substitution σ, we denote with $\sigma(F)$ the formula obtained by replacing each variable (or constant) X appearing in F with $\sigma(X)$. Given two substitutions σ_1 and σ_2 we use $\sigma_1\sigma_2(F)$ as a shortcut for $\sigma_1(\sigma_2(F))$.

Given a CQ q of arity n and a database instance $\mathcal{C}$, we denote as $q^{\mathcal{C}}$ the evaluation of q over $\mathcal{C}$, i.e., the set of n-tuples $\overline{t}$ of constants of Γ such that there exists a substitution that sends the atoms of q to facts of $\mathcal{C}$ and the head to $q(\overline{t})$. Moreover, given a UCQ Q, we define the evaluation of Q over $\mathcal{C}$ as $Q^{\mathcal{C}} = \bigcup_{q \in Q} q^{\mathcal{C}}$

Example 1. Consider a data integration system $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, with $\mathcal{G} = \langle \Psi, \Sigma_T \rangle$. The schema Ψ is constituted by the relations $R_1/2$ and $R_2/2$, the source schema by relations $S_1/2, S_2/1$. The set of TGDs Σ_T contains the single TGD $\theta : \; R_1(X, Y) \rightarrow R_1(Y, W), R_2(Y, X)$. The mapping $\mathcal{M}$ consists of the assertions $S_1(X, c) \rightarrow R_1(X, Y), R_2(Y, Z)$ and $S_2(X) \rightarrow R_2(X, Y)$. ■

Now we come to the semantics of a data integration system $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$. Such a semantics is defined by first considering a *source database* for $\mathcal{I}$, i.e., a database $\mathcal{D}$ for the source schema $\mathcal{S}$. We call *global database* for $\mathcal{I}$ any database for $\mathcal{G}$. Given a source database $\mathcal{D}$ for $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, the semantics $sem(\mathcal{I}, \mathcal{D})$ of $\mathcal{I}$ w.r.t. $\mathcal{D}$ is the set of global databases $\mathcal{B}$ for $\mathcal{I}$ such that:

1. $\mathcal{B}$ satisfies the set Σ_T of TGDs in $\mathcal{G}$; in particular, $\mathcal{B}$ satisfies a TGD $\chi(\boldsymbol{X}, \boldsymbol{Y}) \rightarrow \psi(\boldsymbol{X}, \boldsymbol{Z})$ when, if there exists a substitution σ that sends $\chi(\boldsymbol{X}, \boldsymbol{Y})$ to a set of facts of $\mathcal{B}$, then there exists another substitution σ' that sends $\psi(\boldsymbol{X}, \boldsymbol{Z})$ to $\sigma(\chi(\boldsymbol{X}, \boldsymbol{Y}))$ and those of $\chi(\boldsymbol{X}, \boldsymbol{Y})$ to sets of facts of $\mathcal{B}$. In other words, σ' is a generalisation of σ that sends the atoms of $\psi(\boldsymbol{X}, \boldsymbol{Z})$ to sets of facts of $\mathcal{B}$.
2. $\mathcal{B}$ satisfies $\mathcal{M}$ w.r.t. $\mathcal{D}$. In particular, $\mathcal{B}$ satisfies a GLAV mapping $\mathcal{M}$ w.r.t. $\mathcal{D}$ if for each mapping formula $\varphi_{\mathcal{S}}(\boldsymbol{X}, \boldsymbol{Y}) \rightarrow \varphi_{\mathcal{G}}(\boldsymbol{X}, \boldsymbol{Z})$ we have that, if there exists a substitution σ that sends $\varphi_{\mathcal{S}}(\boldsymbol{X}, \boldsymbol{Y})$ to a set of facts of $\mathcal{D}$, then there exists a generalisation σ' of σ that sends $\varphi_{\mathcal{G}}(\boldsymbol{X}, \boldsymbol{Z})$ to a set of facts of $\mathcal{B}$. Note that the above definition amounts to consider the mapping as *sound* but not necessarily complete; intuitively, for each mapping formula, the data retrievable at the sources are a *subset* of the data that satisfy the corresponding fragment of global schema.

We now give the semantics of queries. Formally, given a source database $\mathcal{D}$ for $\mathcal{I}$ we call *certain answers* to a query q of arity n w.r.t. $\mathcal{I}$ and $\mathcal{D}$, the set $cert(Q, \mathcal{I}, \mathcal{D}) = \{\overline{t} \mid \overline{t} \in Q^{\mathcal{B}}$ for each $\mathcal{B} \in sem(\mathcal{I}, \mathcal{D})\}$, or equivalently $cert(Q, \mathcal{I}, \mathcal{D}) = \bigcap_{\mathcal{B} \in sem(\mathcal{I}, \mathcal{D})} Q^{\mathcal{B}}$.

3 Query Answering in GLAV Systems

In this section we present a framework for query answering in the GLAV approach.

Definition 1. *Let $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$ be a GLAV data integration system, and $\mathcal{D}$ a source database for $\mathcal{I}$. The* retrieved global database *$ret(\mathcal{I}, \mathcal{D})$ is defined constructively as follows. Consider a mapping assertion $\varphi_{\mathcal{S}}(\boldsymbol{X}, \boldsymbol{Y}) \rightarrow \varphi_{\mathcal{G}}(\boldsymbol{X}, \boldsymbol{Z})$. For each set H of facts of $\mathcal{D}$ such that there exists a substitution σ that sends the atoms of $\varphi_{\mathcal{S}}(\boldsymbol{X}, \boldsymbol{Y})$ to H: (i) we first define a substitution σ' such that $\sigma'(X_i) = \sigma(X_i)$ for each X_i in $\boldsymbol{X}$, and $\sigma'(Z_j) = z_j$ for each Z_j in $\boldsymbol{Z}$, where z_j is a fresh constant, not introduced before and not appearing in $\mathcal{D}$; (ii) we add to $ret(\mathcal{I}, \mathcal{D})$ the set of facts that are in $\sigma'(\varphi_{\mathcal{G}}(\boldsymbol{X}, \boldsymbol{Z}))$.*

Now we come to the role of integrity constraints. Here, we will consider a restricted class of TGDs, called *weakly-full TGDs*, for which query answering is decidable. In fact, it is known that query answering under general TGDs is undecidable [14].

Definition 2. *A TGD of the form $\chi(\boldsymbol{X}, \boldsymbol{Y}) \rightarrow \psi(\boldsymbol{X}, \boldsymbol{Z})$ is a* weakly-full TGD[1] *(WFTGD) if each $Y_i \in \boldsymbol{Y}$ appears at most once in $\chi(\boldsymbol{X}, \boldsymbol{Y})$.*

Given a retrieved global database $ret(\mathcal{I}, \mathcal{D})$, in general it does not satisfy the integrity constraints on the global schema, expressed in terms of WFTGDs. Due to the assumption of soundness of views, we are allowed to repair such constraints by suitably adding tuples to the RGD. In general, this can be done in several ways, therefore $sem(\mathcal{I}, \mathcal{D})$ consists of several databases. In the case of WFTGDs, as in other classes of dependencies treated in the literature [4, 8], there exists a database that is a representative of all databases in $sem(\mathcal{I}, \mathcal{D})$. Such a database, called *chase*, is constructed by repairing the violations of a set of WFTGDs Σ_T defined on a schema Ψ, by repeatedly applying, as long it is applicable, the *TGD chase rule*; it is denoted by $chase(\Psi, \Sigma_T, \mathcal{B})$.

[1] We recall that a TGD is *full* if its right-hand side has no existentially quantified variables, i.e., all variables appearing in the conjunction ψ appear also in χ [1].

TGD CHASE RULE. Consider a database $\mathcal{B}$ for a schema Ψ, and a TGD θ of the form $\chi(\boldsymbol{X}, \boldsymbol{Y}) \rightarrow \psi(\boldsymbol{X}, \boldsymbol{Z})$. The TGD θ is *applicable* to $\mathcal{B}$ if there is a substitution σ that sends the atoms of $\chi(\boldsymbol{X}, \boldsymbol{Y})$ to tuples of $\mathcal{B}$, and there is no generalisation of σ that sends the atoms of $\psi(\boldsymbol{X}, \boldsymbol{Z})$ to tuples of $\mathcal{B}$. In this case: *(i)* we define a substitution σ' such that $\sigma'(X_i) = \sigma(X_i)$ for each X_i in $\boldsymbol{X}$, and $\sigma'(Z_j) = z_j$ for each Z_j in $\boldsymbol{Z}$, where z_j is a fresh constant of Γ, not already introduced in the construction and not appearing in $\mathcal{B}$; *(ii)* we add to $\mathcal{B}$ the facts of $\sigma'(\varphi_{\mathcal{G}}(\boldsymbol{X}, \boldsymbol{Z}))$ that are not already in $\mathcal{B}$.

Note that in the case of WFTGDs, the chase may be infinite.

Example 2. Consider Example 1, and let $\mathcal{B}$ be a RGD constituted by a single fact $R_1(a, b)$. Let us construct $chase(\Psi, \Sigma_T, \mathcal{B})$: at the first step we add the facts $R_1(b, z_1), R_2(b, a)$; at the second step the facts $R_1(z_1, z_2), R_2(z_1, b)$; note that the construction process is infinite. ■

Finally, we prove that the chase of the RGD, constructed according to the set of WFTGDs Σ_T, is a representative of all databases of $sem(\mathcal{I}, \mathcal{D})$.

Theorem 1. *Consider a data integration system $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, with $\mathcal{G} = \Psi$, where Σ_T is a set of WFTGDs. Let $\mathcal{D}$ be a source database for $\mathcal{I}$, and let Q be a UCQ expressed over the global schema. We have that $cert(Q, \mathcal{I}, \mathcal{D}) = Q^{chase(\Psi, \Sigma_T, ret(\mathcal{I}, \mathcal{D}))}$.*

4 Query rewriting for tuple-generating dependencies

In this section we present a rewriting technique that takes into account the WFTGDs expressed on the global schema. Such a technique is an extension of that presented in [6], which works in the case of inclusion dependencies.

4.1 Preliminaries

Given a conjunctive query q, we say that a variable X is *unbound* in q if it occurs only once in q, otherwise we say that X is *bound* in q. Notice that variables occurring in the head of the query are necessarily bound, since each one of them must also occur in the query body. A *bound term* is either a bound variable or a constant. Analogously to the standard notation used in deductive databases, we adopt the special symbol "$\star$" for all unbound variables in the query q (deductive database systems use the symbol "_").

Definition 3. *Given an atom $g_1 = r(X_1, \ldots, X_n)$ and an atom $g_2 = r(Y_1, \ldots, Y_n)$, we say that g_1* and *g_2* unify *if there exists a variable substitution σ such that $\sigma(g_1) = \sigma(g_2)$. Each such a σ is called* unifier. *Moreover, if g_1 and g_2 unify, we denote as* $\mathrm{mgu}(g_1, g_2)$ a most general unifier (mgu) of g_1 and g_2 *(we recall that σ is a mgu if for every unifier σ' of g_1 and g_2 there exists a substitution γ such that $\sigma' = \gamma\sigma$).*

Definition 4. *Given a set of atoms $G = \{g_1, \ldots, g_k\}$ and a WFTGD $\theta : \chi(\boldsymbol{X}, \boldsymbol{Y}) \rightarrow \psi(\boldsymbol{X}, \boldsymbol{Z})$, we say that θ is* applicable *to G if there exists a substitution σ such that: (i) σ sends the atoms of $\psi(\boldsymbol{X}, \boldsymbol{Z})$ to $g_1, \ldots, g_k$; (ii) for any variable or constant $W \neq \star$ in G (i.e., for any bound term W in G), there is at least one variable X in $\boldsymbol{X}$ such that $\sigma(X) = W$. If such a σ exists, we denote with $\sigma_{G,\theta}$ the most general substitution that verifies the above conditions. Moreover, we denote with rew(G, θ) the conjunction of atoms $\sigma_{G,\theta}(\chi(\boldsymbol{X}, \boldsymbol{Y}))$.*

Algorithm reduce(q, g_1, g_2)
Input: conjunctive query q,
atoms $g_1, g_2 \in body(q)$
such that g_1 and g_2 unify
Output: reduced CQ q'
$q' = q$;
$\sigma := \mathrm{mgu}(g_1, g_2)$;
$body(q') := body(q') - \{g_2\}$;
$q' := \sigma(q')$;
$q' := \tau(q')$;
return q'

Algorithm rewrite(q, G, θ)
Input: conjunctive query q,
set of atoms $G \subseteq body(q)$,
WFTGD θ such that
θ is applicable to G
Output: rewritten CQ q'
$q' := q$;
$body(q') := body(q') - G$;
$q' := \sigma_{G,\theta}(q')$;
$body(q') := body(q') \cup rew(G, \theta)$;
return q'

Fig. 1. The auxiliary algorithms reduce and rewrite

4.2 The algorithm TGDrewrite

In Figure 2 we define the algorithm TGDrewrite that computes the rewriting of a UCQ Q expressed over the global schema. We will show that such a rewriting, when evaluated on the RGD, returns the result of the evaluation of Q over the chase of the RGD, i.e., the certain answers to Q. In the algorithm, it is assumed that unbound variables in the input query Q are represented by the symbol $\star$.

Algorithm TGDrewrite(Ψ, Σ_I, Q)
Input: relational schema Ψ,
set of WFTGDs Σ_T, UCQ Q
Output: rewritten query Q'
$Q' := Q$;
repeat
$Q_{aux} := Q'$;
for each $q \in Q_{aux}$ **do**
(a) **for each** $g_1, g_2 \in body(q)$ **do**
if g_1 and g_2 unify
then $Q' := Q' \cup \{\mathsf{reduce}(q, g_1, g_2)\}$;
(b) **for each** $G \subseteq body(q)$ **do**
for each $\theta \in \Sigma_T$ **do**
if θ is applicable to G
then $Q' := Q' \cup \{\mathsf{rewrite}(q, g, I)\}$
until $Q_{aux} = Q'$;
return Q'

Fig. 2. The algorithm TGDrewrite

More specifically, the algorithm TGDrewrite makes use of the auxiliary algorithms reduce and rewrite; it generates a set of conjunctive queries, whose union is the rewritten query, according to the following rules, applied to Q until the fixpoint is reached. (1) For each pair of atoms g_1 and g_2 that unify in the body of a CQ $q \in Q$, then the algorithm adds to Q the conjunctive query reduce(q, g_1, g_2). Note that the most general unifier $\mathrm{mgu}(g_1, g_2)$ is applied to the whole CQ q. Finally, a function τ replaces with $\star$ each unbound variable symbol; this is necessary in order to guarantee that the generated query has the required form. (2) If there exists a WFTGD θ and a conjunctive query $q \in Q$ such that θ is applicable to a set of atoms $G \subseteq body(q)$, then the algorithm adds to Q the query obtained from q by replacing G with $rew(G, \theta)$ and by applying the substitution $\sigma_{G,\theta}$ to q. This step adds new CQs obtained by applying WFTGDs as rewriting rules (applied from right to left). The above two rules correspond to steps (a) and (b) of the algorithm, which make use of the algorithms reduce and rewrite respectively. Termination is guaranteed by the fact that no new symbols are introduced in the rewriting process, and the number of atoms that can be written with a finite set of relation symbols in Ψ, variables (including the special symbol $\star$) and constants is finite.

Example 3. Consider Example 1 and a CQ $q(X_1) \leftarrow R_1(X_1, X_2), R_2(X_1, X_3)$, represented as $q(X_1) \leftarrow R_1(X_1, \star), R_2(X_1, \star)$. The WFTGD θ is applicable to $G = \{R_1(X_1, \star), R_2(X_1, \star)\}$, and the application of rewrite yields the CQ $q(X_1) \leftarrow R_1(X_3, X_1)$, being $\sigma_{G,\theta} = \{X \rightarrow X_3, Y \rightarrow X_1, W \rightarrow X_2\}$. The rewritten CQ is represented as $q(X_1) \leftarrow R_1(\star, X_1)$, after the application of τ. ■

We can now give the main result about the algorithm TGDrewrite.

Theorem 2. *Consider a data integration system $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, with $\mathcal{G} = \langle \Psi, \Sigma_T \rangle$, a source database $\mathcal{D}$ for $\mathcal{I}$, and a UCQ Q expressed over $\mathcal{G}$. We have that* $\mathsf{TGDrewrite}(Q)^{ret(\mathcal{I},\mathcal{D})} = Q^{chase(\Psi,\Sigma_T,ret(\mathcal{I},\mathcal{D}))}$.

5 Query Rewriting for the Mapping

In the previous section we have shown a technique that allows us to obtain the certain answers by evaluating a rewriting of the query Q over the RGD, thus avoiding the construction of the (possibly infinite) chase. In this section we will present a rewriting technique that allows us to avoid even the construction of the RGD. The technique is based on a transformation of the given GLAV system into an equivalent GAV one, and then on the application of a variant of the known techniques for GAV [15], namely, the *unfolding*.

5.1 From GLAV to GAV

Here we present a technique for compiling a GLAV system into an equivalent GAV one. The notion of equivalence is given in terms of queries, and it is the same as in [3]. In particular, given two integration systems $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$ and $\mathcal{I}' = \langle \mathcal{G}', \mathcal{S}, \mathcal{M}' \rangle$ over the same source schema $\mathcal{S}$ and such that all relations of $\mathcal{G}$ are also relations of $\mathcal{G}'$, we say that $\mathcal{I}'$ is *query-preserving* with respect to $\mathcal{I}$, if for every query q over $\mathcal{I}$ and for every source database $\mathcal{D}$ for $\mathcal{S}$, we have that $cert(q, \mathcal{I}, \mathcal{D}) = cert(q, \mathcal{I}', \mathcal{D})$. The transformation of a GLAV system into a GAV one is an extension of the one presented in [3], which transforms LAV systems without integrity constraints into GAV ones; we improve the technique as follows: *(i)* we transform GLAV systems instead of LAV ones; as already observed in Section 2, the class of GLAV systems is a generalisation of both GAV and LAV; *(ii)* we allow the presence of dependencies belonging to the class of WFTGDs in the original GLAV system.

Let $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$ be the initial GLAV system, with $\mathcal{G} = \langle \Psi, \Sigma_T \rangle$ and $\mathcal{I}' = \langle \mathcal{G}', \mathcal{S}, \mathcal{M}' \rangle$, with $\mathcal{G}' = \langle \Psi', \Sigma_T' \rangle$ be the transformed GAV one. The transformation is performed as follows. (1) The set of sources $\mathcal{S}$ remains unchanged. (2) The global schema $\mathcal{G}'$ is obtained from $\mathcal{G}$ by introducing: *(i)* a new relation R/n_d for each mapping formula $\varphi_{\mathcal{S}}(\boldsymbol{X}, \boldsymbol{Y}) \rightarrow \varphi_{\mathcal{G}}(\boldsymbol{X}, \boldsymbol{Z})$, where n_d is the number of symbols in $\boldsymbol{X} = X_1, \ldots, X_{n_d}$; *(ii)* a new relation $R_{\text{exp}}/(n_d + n_n)$ for each relation R introduced in the previous step, where n_n is the number of symbols in $\boldsymbol{Z} = Z_1, \ldots, Z_{n_n}$ (3) The GAV mapping $\mathcal{M}'$ associates to each global relation R the query $R(X_1, \ldots, X_{n_d}) \leftarrow \varphi_{\mathcal{S}}(\boldsymbol{X}, \boldsymbol{Y})$. We do not associate any query to the remaining global relations. (4) For each mapping assertion $\varphi_{\mathcal{S}}(\boldsymbol{X}, \boldsymbol{Y}) \rightarrow \varphi_{\mathcal{G}}(\boldsymbol{X}, \boldsymbol{Z})$ and its corresponding introduced relation symbol R_{exp}: *(i)* we add the WFTGD $R(X_1, \ldots, X_{n_d}) \rightarrow R_{\text{exp}}(X_1, \ldots, X_{n_d}, Z_1, \ldots, Z_{n_n})$; *(ii)* we add the full TGD $R_{\text{exp}}(X_1, \ldots, X_{n_d}, Z_1, \ldots, Z_{n_n}) \rightarrow \varphi_{\mathcal{G}}(\boldsymbol{X}, \boldsymbol{Z})$.

It is immediate to verify that, given a GLAV integration system $\mathcal{I}$, the corresponding GAV integration system $\mathcal{I}'$ defined as above can be constructed in time that is linear in the size of $\mathcal{I}$. We now show that the above transformation is query-preserving.

Theorem 3. *Let $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$ be a GLAV integration system, with $\mathcal{G} = \langle \Psi, \Sigma_T \rangle$ and let $\mathcal{I}' = \langle \mathcal{G}', \mathcal{S}, \mathcal{M}' \rangle$, with $\mathcal{G} = \langle \Psi', \Sigma'_T \rangle$, be the corresponding GAV integration system defined as above. Then $\mathcal{I}'$ is query-preserving with respect to $\mathcal{I}$.*

5.2 Unfolding

We now go back to query rewriting. The algorithm IDrewrite provides a rewriting of a UCQ Q that returns the certain answers to Q when evaluated over the RGD; however, we are interested in expressing the query in terms of the source relations is $\mathcal{S}$. In order to do this, we define a transformation $unfold$ that "unfolds" the rewritten query by using the GAV mapping $\mathcal{M}$. The following definition formally extends the well-known concept of query unfolding.

Definition 5. *Given a conjunctive query q of the form $q(\boldsymbol{X}) \leftarrow g_1, \ldots, g_k$ over $\mathcal{G}$, and a GAV mapping $\mathcal{M}$, we say that an atom g_i is* unfoldable *in $\mathcal{M}$ if and only if there exists a mapping assertion in $\mathcal{M}$ whose head unifies with g_i. Moreover, we say that q is* unfoldable in $\mathcal{M}$ *if, for each j such that $1 \leq j \leq k$, $\sigma_{j-1} \ldots \sigma_1(g_j)$ is unfoldable in $\mathcal{M}$ with most general unifier σ_j. If q is unfoldable in $\mathcal{M}$, we denote as $unfold(q, \mathcal{M})$ the CQ*

$$\sigma_k \ldots \sigma_1(q(\boldsymbol{X})) \leftarrow unfold(g_1, \mathcal{M}), unfold(\sigma_1(g_2), \mathcal{M}), \ldots, unfold(\sigma_{k-1} \ldots \sigma_1(g_k), \mathcal{M})$$

otherwise, $unfold(q, \mathcal{M}) = \emptyset$. Finally, given a UCQ Q, we define $unfold(Q, \mathcal{M}) = \{unfold(q, \mathcal{M}) \mid q \in Q\}$

Theorem 4. *Let $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$ be an integration system with $\mathcal{G} = \langle \Psi, \Sigma_T \rangle$; let $\mathcal{D}$ be a source database for $\mathcal{I}$, and Q be a UCQ over $\mathcal{G}$. Then, $unfold(\mathsf{TGDrewrite}(\Psi, \Sigma_T, Q), \mathcal{M})^{\mathcal{D}} = cert(Q, \mathcal{I}, \mathcal{D})$.*

6 Discussion

In this paper we have addressed the problem of query answering in GLAV data integration systems, in the presence of WFTGDs, a restricted class of tuple-generating dependencies for which query answering is decidable.

Several works in the literature address the problem of query answering under integrity constraints, both in a single database context [5, 2, 10] and in data integration [7, 8, 6, 14, 4, 11, 16]. In particular, [14] presents a technique for query rewriting under *conjunctive inclusion dependencies (CIDs)*, that are analogous to TGDs; however, this work considers *acyclic* CIDs, so that the problem of having an infinite chase is not addressed. Also, the excellent work of Fagin et al. [8], in the context of *data exchange*, deals with a class of ICs that is incomparable to the one treated in this paper; this work considers a class of TGDs (the *weakly-acyclic* TGDs) together with *equality-generating dependencies*; also in this case, due to the fact that in data exchange the chase is to be materialised, the chase turns out to be finite.

First, we have presented a framework for dealing with integrity constraints in the GLAV approach, based on the notions of retrieved global database [15] and chase [13];

since GLAV is a generalisation of both LAV and GAV, we have thus shown that LAV and GAV are siblings and not opposites, as often stated in the literature. Then, we have presented a further rewriting technique that allows us to take into account the mapping, once it is "compiled" into an equivalent GAV one.

The tractability of our techniques is easily proved, since the evaluation of a UCQ over a database can be done in PTIME in data complexity, i.e., w.r.t. the size of the data; we recall that data complexity is the one that is usually considered in a database context, since the size of the queries is assumed to be much smaller than the size of the data. Due to space limitations, we are not able to give a detailed analysis of the computational complexity of query rewriting in our case.

Acknowledgements This work was supported by the following projects: INFOMIX (IST-2001-33570), funded by the EU; FIRB-MAIS and "Società dell'Informazione", both funded by MIUR. The author wishes to thank Maurizio Lenzerini, Riccardo Rosati and Domenico Lembo for their insightful comments about this material.

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 PODS'99*, pages 68–79.
3. Andrea Calì, Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. On the expressive power of data integration systems. In *Proc. of ER 2002*.
4. Andrea Calì, Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. Data integration under integrity constraints. *Information Systems*, 2003. To appear.
5. Andrea Calì, Domenico Lembo, and Riccardo Rosati. On the decidability and complexity of query answering over inconsistent and incomplete databases. In *Proc. of PODS 2003*.
6. Andrea Calì, Domenico Lembo, and Riccardo Rosati. Query rewriting and answering under constraints in data integration systems. In *Proc. of IJCAI 2003*. To appear.
7. Oliver M. Duschka and Michael R. Genesereth. Answering recursive queries using views. In *Proc. of PODS'97*, pages 109–116.
8. Ronald Fagin, Phokion Kolaitis, Renee J. Miller, and Lucian Popa. Data exchange: Semantics and query answering. In *Proc. of ICDT 2003*, pages 207–224.
9. Marc Friedman, Alon Levy, and Todd Millstein. Navigational plans for data integration. In *Proc. of AAAI'99*, pages 67–73, 1999.
10. Gianluigi Greco, Sergio Greco, and Ester Zumpano. A logic programming approach to the integration, repairing and querying of inconsistent databases. In *Proc. of ICLP'01*, volume 2237 of *LNAI*, pages 348–364. Springer.
11. Jarek Gryz. Query rewriting using views in the presence of functional and inclusion dependencies. *Information Systems*, 24(7):597–612, 1999.
12. Alon Y. Halevy. Answering queries using views: A survey. *VLDB Journal*, 10(4):270–294, 2001.
13. David S. Johnson and Anthony C. Klug. Testing containment of conjunctive queries under functional and inclusion dependencies. *J. of Computer and System Sciences*, 28(1):167–189, 1984.
14. Christoph Koch. Query rewriting with symmetric constraints. In *Proc. of FoIKS'02*, pages 130–147.
15. Maurizio Lenzerini. Data integration: a theoretical perspective. In *Proc. of PODS 2002*.
16. Jinxin Lin and Alberto O. Mendelzon. Merging databases under constraints. *Int. J. of Cooperative Information Systems*, 7(1):55–76, 1998.