Bidirectional Constraints for Exchanging Data: Beyond Monotone Queries

Marcelo Arenas
Dept. of Computer Science
PUC Chile
marenas@ing.puc.cl

Gabriel Diéguez
Dept. of Computer Science
PUC Chile
gsdieguez@uc.cl

Jorge Pérez
Dept. of Computer Science
Universidad de Chile
jperez@dcc.uchile.cl

Abstract

In this paper, we propose to use the language of *bidirectional constraints* to specify schema mappings in the context of data exchange. These constraints impose restrictions over both the source and the target data, and have the potential to minimize the ambiguity in the description of the target data to be materialized. We start by making a case for the usefulness of bidirectional constraints to give a meaningful closed-world semantics for st-tgds, which is motivated by Clark's predicate completion and Reiter's formalization of the closed-world assumption of a logical theory. We then formally study the use of bidirectional constraints in data exchange. In particular, we pinpoint the complexity of the existence-of-solutions and the query evaluation problems in several different scenarios, including in the latter case both monotone and non-monotone queries.

1 Introduction

Data exchange is the process of taking data structured under a source schema and transforming it into data structured under a target schema. A fundamental building block in this process is the notion of a schema mapping, which is a high-level specification that describes how data from the source should be mapped into the target.

In relational databases, schema mappings are usually specified by using a logical language. In particular, most of the research on data exchange [Lenzerini, 2002; Fagin *et al.*, 2005a] has focused on mappings specified by the so-called source-to-target tuple-generating dependencies (st-tgds), which have been also widely used in practical applications [Haas *et al.*, 2005; Bernstein *et al.*, 2006]. An st-tgd is a first-order logic sentence of the form:

$$\forall \bar{x} \forall \bar{y} \big(\varphi_{\mathbf{S}}(\bar{x}, \bar{y}) \rightarrow \exists \bar{z} \psi_{\mathbf{T}}(\bar{x}, \bar{z}) \big),$$

where $\varphi_{\mathbf{S}}(\bar{x}, \bar{y})$ and $\psi_{\mathbf{T}}(\bar{x}, \bar{z})$ are conjunctions of relational atoms from the source and target schemas, respectively. For example, consider a source schema consisting of a relation *Student*(name, course) storing students' names and the names of the courses where they are enrolled in. Moreover, consider a target schema consisting of a relation *Enroll*(sname, cid), storing students' names and the ids of the courses where they are enrolled in, and of a relation *Course*(cid, cname), storing for every course its id and name. In this case, the exchange is driven by the following st-tgd:

$$\forall x \forall y \big(\textit{Student}(x, y) \rightarrow \exists z (\textit{Enroll}(x, z) \wedge \textit{Course}(z, y)) \big). \quad (1)$$

In data exchange, one is given a source database I and a schema mapping $\mathcal{M}$, and then the main problem is to find a *target database* J that is a *solution* for I under $\mathcal{M}$, that is, a valid translation of I according to the constraints imposed by $\mathcal{M}$. For example, consider a mapping $\mathcal{M}$ specified by st-tgd (1) and the following source database I_1:

Student	name	course
	John	AI
	Albert	Databases

Then the following target database J_1 is a possible solution for I_1 under the constraints imposed by $\mathcal{M}$:

Enroll	sname	cid
	John	CS301
	Albert	CS303

Course	cid	cname
	CS301	AI
	CS303	Databases

In fact, I_1 together with J_1 satisfy st-tgd (1) according to the standard first-order logic semantics. Nevertheless, J_1 is not the only target database that satisfies this condition. Consider, for example, the following target database J_2:

Enroll	sname	cid
	John	CS301
	Albert	CS303
	Albert	CS400

Course	cid	cname
	CS301	AI
	CS303	Databases
	CS400	Algorithms

Again, we have that I_1 together with J_2 satisfy st-tgd (1) according to the standard first-order logic semantics. Thus, the database J_2, although less natural than J_1, is also a solution for I_1 according to the constraints imposed by $\mathcal{M}$. This sort of anomaly is caused by the semantics of the implication; the formula used to exchange data is not restricting the possibility of adding arbitrary tuples to *Enroll* and *Course*. Moreover, this formula does not impose any restrictions on the courses' ids; if we replace CS301 by CS401 in the database J_2, we also obtain a solution for I_1 under $\mathcal{M}$. Thus, target databases are allowed to contain *labeled nulls* in this context, which represent values that can be replaced by any constant. In fact, the following target database is a solution for I_1 under $\mathcal{M}$, where $\perp_1$ and $\perp_2$ are distinct null values:

Enroll	sname	cid
	John	$\perp_1$
	Albert	$\perp_2$

Course	cid	cname
	$\perp_1$	AI
	$\perp_2$	Databases

The semantics of st-tgds has raised several issues in data exchange. One of the most prominent is the problem of answering target queries. Given a source database, a schema mapping, and a query Q over the target schema, what should be the answer for Q after the exchange? Given that there are many possible solutions for the source instance, this semantics is not immediately clear.

The *certain answers* semantics has been adopted as the standard semantics for query answering in data exchange. More precisely, a tuple $\bar{t}$ is said to be a certain answer for a target query Q given a source database I and a mapping $\mathcal{M}$, if $\bar{t}$ is an *answer for Q over every possible solution* for I under the constraints imposed by $\mathcal{M}$. Although this semantics gives intuitively correct results for positive queries [Fagin *et al.*, 2003; 2005a], it has a less natural behavior for non-monotone queries [Arenas *et al.*, 2004; Libkin, 2006; Hernich, 2013]. For instance, consider again the source database I_1 and the mapping $\mathcal{M}$ specified by st-tgd (1), and assume that Q is the following target query:

$$\neg\exists z(\textit{Enroll}(\text{John}, z) \wedge \textit{Course}(z, \text{Programming})). \quad (2)$$

This query is asking whether John is not enrolled in Programming. Given the data in I_1, the natural answer to Q would be *true*. However, the actual answer to this query is *false* as there exists a possible solution for I_1 under $\mathcal{M}$ where John is enrolled in Programming.

To tackle this problem, many authors have proposed to restrict the notion of data exchange solution based on some minimality criteria [Fagin *et al.*, 2005b; Libkin, 2006; Afrati and Kolaitis, 2008; Hernich *et al.*, 2011; Libkin and Sirangelo, 2011; Gottlob *et al.*, 2011; Hernich, 2012; 2013], which has led to different alternatives for the semantics of non-monotone queries in this context. Regardless of the usefulness of these proposals, each one of them departs from the well-understood semantics of first-order logic for st-tgds, as initially proposed in [Fagin *et al.*, 2005a]. We are convinced that having a clean semantics for schema mappings based on the semantics of first-order logic has been instrumental in the adoption of this technology. Thus, as opposed to the approaches mentioned above, our main goal in this work is to propose and study a mapping-specification language that can be defined in terms of the usual syntax and semantics of first-order logic (FO), and whose semantics is suitable for both monotone and non-monotone queries.

In [Arenas *et al.*, 2014b], we proposed to use the so-called *bidirectional constraints* to specify schema mappings. A bidirectional constraint is an FO-sentence of the form $\forall\bar{x}\big(\alpha_{\mathbf{S}}(\bar{x}) \leftrightarrow \beta_{\mathbf{T}}(\bar{x})\big)$, where $\alpha_{\mathbf{S}}(\bar{x})$ and $\beta_{\mathbf{T}}(\bar{x})$ are FO-formulas over the source and target schemas, respectively. Thus, these dependencies allow expressing in a natural way what pieces of data should and should not be in the target, reducing the ambiguity when describing what target instances should be materialized. In this paper, we embark on a study of the fundamental properties of these constraints regarding data exchange and non-monotone queries, aiming to meet the requirements mentioned in the previous paragraph. It is important to mention that although this type of dependencies has been previously considered in the literature [Melnik *et al.*, 2008; Arenas *et al.*, 2014b], the properties studied in this paper have not been investigated.

Our contributions. First, in Section 3 we make a case for the usefulness of bidirectional constraints to give a meaningful closed-world semantics for st-tgds. Our proposal is motivated by the classical notion of *predicate completion* [Clark, 1977] and *Reiter's formalization of the closed-world assumption* of a logical theory [Reiter, 1977]. Essentially, we prove that for every schema mapping $\mathcal{M}$ specified by a set of st-tgds, one can compute a *closure* of $\mathcal{M}$ specified by a set of bidirectional constraints which, in a precise sense, completes $\mathcal{M}$ assuming as negative any piece of information for which there is no evidence according to $\mathcal{M}$. For example, for the case of st-tgd (1), the closure is very simple and is given by the following dependency:

$$\forall x\forall y\big(\textit{Student}(x, y) \leftrightarrow \exists z(\textit{Enroll}(x, z) \wedge \textit{Course}(z, y))\big). \quad (3)$$

It is worth noticing that under this bidirectional constraint, we obtain *true* as the answer to the query (2) given the source instance I_1 defined previously, which is the expected answer. Besides, it is worth noticing that although the closure was obtained in this case by just replacing an implication by a double implication, this process is much more involved in general.

Second, we study bidirectional constraints in depth by considering two fundamental data exchange problems: the existence of solutions problem in Section 4, and the query evaluation problem in Section 5. We pinpoint the complexity of these problems in several different scenarios and, in particular, for several different classes of monotone and non-monotone queries in Section 5. We also describe several restricted scenarios for which these problems become tractable.

2 Preliminaries

We assume that data is represented in the relational model. A *relational schema* $\mathbf{R}$, or just *schema*, is a finite set $\{R_1, \ldots, R_n\}$ of relation symbols, with each R_j having a fixed arity k_j. Let $\mathbf{C}$ and $\mathbf{N}$ be countably infinite sets with no elements in common. We refer to the elements in $\mathbf{C}$ as constants and to the elements in $\mathbf{N}$ as labeled nulls, or just nulls. An instance I of $\mathbf{R}$ assigns to each relation symbol R_j a finite k_j-ary relation $R_j^I \subseteq (\mathbf{C} \cup \mathbf{N})^{k_j}$. We denote by $\mathrm{Inst}(\mathbf{R})$ the set of all instances of $\mathbf{R}$. Finally, slightly abusing notation, we use $\mathbf{C}(\cdot)$ to denote a built-in predicate such that $\mathbf{C}(a)$ holds if and only if a is a constant (that is, $a \in \mathbf{C}$).

Query languages. We assume some familiarity with first-order logic (FO), and we focus on queries expressed by using formulas in this logic. Besides FO, the main query languages considered in this paper are the languages of conjunctive queries (CQ), unions of conjunctive queries (UCQ), and the languages obtained from them by adding the equality predicate, the inequality predicate and the negation operator (e.g. $\text{UCQ}^{=}$, $\text{CQ}^{\neq}$ and $\text{UCQ}^{\neg}$). We also consider the class of monotone queries, denoted by MON. This class contains all queries Q over a schema $\mathbf{R}$ that satisfy the following property: given two instances J_1, J_2 of $\mathbf{R}$ such that $J_1 \subseteq J_2$, it holds that $Q(J_1) \subseteq Q(J_2)$. Finally, we assume that all query languages allow the use of built-in predicate $\mathbf{C}$.

Schema mappings. As is customary in the data exchange literature, if we refer to a schema $\mathbf{S}$ as a source schema, then $\mathrm{Inst}(\mathbf{S})$ is defined to be the set of all instances of $\mathbf{S}$ that are constructed by using only elements from $\mathbf{C}$, and if we refer to a schema $\mathbf{T}$ as a target schema, then $\mathrm{Inst}(\mathbf{T})$ is defined as usual (that is, the instances of $\mathbf{T}$ are constructed by using elements from both $\mathbf{C}$ and $\mathbf{N}$). In this article, we use $\mathbf{S}$ to refer to a source schema and $\mathbf{T}$ to refer to a target schema. Besides, in this section we assume that $\mathbf{S}$ and $\mathbf{T}$ are fixed source and target schemas with no relation symbols in common.

Schema mappings are used to define a semantic relationship between a source and a target schema. A schema mapping, or just mapping, $\mathcal{M}$ from $\mathbf{S}$ to $\mathbf{T}$ is a set of pairs (I, J), where I is an instance of $\mathbf{S}$, and J is an instance of $\mathbf{T}$. That is, a mapping $\mathcal{M}$ is just a subset of $\mathrm{Inst}(\mathbf{S}) \times \mathrm{Inst}(\mathbf{T})$. Given an instance I of $\mathbf{S}$, a mapping $\mathcal{M}$ associates to I a set of *possible solutions for* I, denoted by $\textsc{Sol}_{\mathcal{M}}(I)$, given by the set $\{J \in \mathrm{Inst}(\mathbf{T}) \mid (I, J) \in \mathcal{M}\}$. Notice that the set $\textsc{Sol}_{\mathcal{M}}(I)$ represents the possible translations of I according to $\mathcal{M}$.

In practice, schema mappings are represented by using logical formulas. Again, we focus on using fragments of FO to specify mappings. Given a finite set Σ of FO-sentences over vocabulary $\mathbf{S} \cup \mathbf{T}$, we say that a mapping $\mathcal{M}$ from $\mathbf{S}$ to $\mathbf{T}$ is *specified* by Σ if for every pair of instances $(I, J) \in \mathrm{Inst}(\mathbf{S}) \times \mathrm{Inst}(\mathbf{T})$, it holds that $(I, J) \in \mathcal{M}$ if and only if $I \cup J \models \Phi$ for every $\Phi \in \Sigma$, where $I \cup J$ represents the instance of $\mathbf{S} \cup \mathbf{T}$ obtained by taking the union of the relations in I and J (recall that $\mathbf{S}$ and $\mathbf{T}$ have no relation symbols in common). For convenience, we write the last statement as $(I, J) \models \Sigma$. Therefore, we usually refer to a mapping as a triple $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, and to the set of solutions as $\textsc{Sol}_{\mathcal{M}}(I) = \{J \in \mathrm{Inst}(\mathbf{T}) \mid (I, J) \models \Sigma\}$.

Dependencies. Schema mappings from $\mathbf{S}$ to $\mathbf{T}$ are usually defined by *source-to-target dependencies* [Fagin *et al.*, 2005a], that is, by formulas of the form $\forall \bar{x}(\varphi(\bar{x}) \rightarrow \psi(\bar{x}))$, where $\varphi(\bar{x})$ is an FO-formula over $\mathbf{S}$, $\psi(\bar{x})$ is an FO-formula over $\mathbf{T}$, and $\bar{x}$ is the tuple of free variables of both formulas. We usually drop the outermost universal quantification when specifying these constraints, and thus we only write $\varphi(\bar{x}) \rightarrow \psi(\bar{x})$ for the previous formula. Depending on which fragments of FO we use to define formulas $\varphi(\bar{x})$ and $\psi(\bar{x})$, we obtain a wide range of possible fragments of source-to-target dependencies. Given fragments $\mathcal{L}_1$ and $\mathcal{L}_2$ of FO, an $\mathcal{L}_1$-TO-$\mathcal{L}_2$ dependency is a formula of the above form in which $\varphi(\bar{x})$ is an $\mathcal{L}_1$-formula over $\mathbf{S}$ and $\psi(\bar{x})$ is an $\mathcal{L}_2$-formula over $\mathbf{T}$. In [Fagin *et al.*, 2005a], the language of CQ-TO-CQ dependencies was chosen as the preferred formalism for specifying schema mappings, calling them *source-to-target tuple-generating dependencies (st-tgds)*. In this article, we also consider *full* $\mathcal{L}$-TO-CQ dependencies, which are formulas in which the target part is a conjunctive query without existential quantifiers.

In this paper, we study mappings specified by sets of formulas of the following form

$$\forall \bar{x} \left(\varphi(\bar{x}) \leftrightarrow \psi(\bar{x}) \right), \tag{4}$$

where $\varphi(\bar{x})$ is an FO-formula over $\mathbf{S}$, $\psi(\bar{x})$ is an FO-formula over $\mathbf{T}$, and $\bar{x}$ is the tuple of free variables of both formulas. Such a formula is called a *bidirectional constraint* [Arenas *et al.*, 2014b]. We usually drop the outermost universal quantification, and we write $\varphi(\bar{x}) \leftrightarrow \psi(\bar{x})$ for formula (4). We say that Φ is an $\langle \mathcal{L}_1, \mathcal{L}_2 \rangle$-dependency between $\mathbf{S}$ and $\mathbf{T}$, if Φ is of the form (4) with $\varphi(\bar{x})$ in $\mathcal{L}_1$ and $\psi(\bar{x})$ in $\mathcal{L}_2$. When the source and target schemas are clear from the context, we will only talk about $\langle \mathcal{L}_1, \mathcal{L}_2 \rangle$-dependencies. We also consider *full* $\langle \mathcal{L}, \mathrm{CQ} \rangle$-dependencies, defined as before.

Query answering in data exchange. An important aspect in data exchange is how to answer queries over a target schema. The most accepted semantics for query answering is the *certain answers semantics:* given a mapping $\mathcal{M}$ from $\mathbf{S}$ to $\mathbf{T}$, an instance I of $\mathbf{S}$ and a query Q over $\mathbf{T}$, the certain answers of Q with respect to I under $\mathcal{M}$ is the set $\textsc{certain}_{\mathcal{M}}(Q, I) = \bigcap_{J \in \textsc{Sol}_{\mathcal{M}}(I)} Q(J)$. In other words, a tuple $\bar{t} \in \textsc{certain}_{\mathcal{M}}(Q, I)$ if and only if $\bar{t} \in Q(J)$ for every solution J for I under $\mathcal{M}$.

3 Closing Mappings through Bidirectional Constraints

The problem of answering non-monotone queries is of great interest in the data exchange area. In particular, the issue of defining a meaningful semantics for such queries has received a lot of attention, mainly because an open-world approach was followed in the initial definition of the notion of solution in data exchange [Fagin *et al.*, 2005a]. Many authors have proposed restrictions on this notion based on some minimality criteria [Fagin *et al.*, 2005b; Libkin, 2006; Afrati and Kolaitis, 2008; Hernich *et al.*, 2011; Libkin and Sirangelo, 2011; Gottlob *et al.*, 2011; Hernich, 2012; 2013], which has led to some different alternatives for the semantics of non-monotone queries in this context.

Arguably, the main advantage of the data exchange framework proposed in [Fagin *et al.*, 2005a] is that mappings are specified by using st-tgds, whose syntax and semantics are defined in a clean way by relying on the well-understood syntax and semantics of first-order logic. We are convinced that having such a clean framework has been instrumental in the adoption of this technology. Thus, as opposed to the approaches mentioned in the previous paragraph, our main goal in this work is to propose and study a mapping-specification language that can be defined in terms of the usual syntax and semantics of first-order logic, and whose semantics is suitable for both monotone and non-monotone queries.

Bidirectional constraints allow expressing in a natural way what pieces of data should and should not be in the target. This, together with the fact that their syntax and semantics is inherited from first-order logic, makes these constraints quite appealing to meet the aforementioned requirements. Moreover, as shown next, bidirectional constraints can be used to give a closed-world semantics to a mapping specified by st-tgds, as these constraints are expressive enough to encode an extension of the classical notion of *predicate completion* [Clark, 1977] and, at the same time, an adaptation of *Reiter's formalization of the closed-world assumption* [Reiter, 1977] to the case of st-tgds. This gives more evidence that bidirectional constraints form a good mapping-specification language that deserves careful consideration.

Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ be a mapping specified by a set Σ of st-tgds. Then a mapping $\mathcal{M}^\star$ from $\mathbf{S}$ to $\mathbf{T}$ is said to be a *closure* of $\mathcal{M}$ if for every st-tgd $\varphi(\bar{x}) \rightarrow \psi(\bar{x}) \in \Sigma$, every source instance I and every tuple $\bar{t}$ of constants:

if $\bar{t} \in \text{CERTAIN}_{\mathcal{M}}(\psi(\bar{x}), I)$, then $\bar{t} \in \text{CERTAIN}_{\mathcal{M}^\star}(\psi(\bar{x}), I)$; and

if $\bar{t} \notin \text{CERTAIN}_{\mathcal{M}}(\psi(\bar{x}), I)$, then $\bar{t} \in \text{CERTAIN}_{\mathcal{M}^\star}(\neg\psi(\bar{x}), I)$.

To the best of our knowledge, this formalization of the notion of a closure of a mapping is new. Notice that this notion is an extension of Clark's predicate completion [Clark, 1977], as in $\mathcal{M}^\star$ we complete the definition of the heads of the dependencies in Σ. Besides, this notion is also inspired by Reiter's formalization of the closed-world assumption [Reiter, 1977], as if $\psi(\bar{t})$ cannot be inferred from Σ, then $\neg\psi(\bar{t})$ is added into the theory.

A natural question is whether a closure of a mapping $\mathcal{M}$ given by st-tgds can be expressed by using also st-tgds. The following proposition gives a negative answer to this question, even if we are allowed to use FO-TO-CQ dependencies.

Proposition 1. *There is a mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where Σ is a set of st-tgds, for which no mapping $\mathcal{M}' = (\mathbf{S}, \mathbf{T}, \Sigma')$ is a closure, assuming that Σ' is a set of* FO-TO-CQ *dependencies.*

This proposition follows from the fact that for every mapping $\mathcal{M}' = (\mathbf{S}, \mathbf{T}, \Sigma')$ specified by a set Σ' of FO-TO-CQ dependencies, every instance I of $\mathbf{S}$, and every conjunctive query $Q(\bar{x})$ over $\mathbf{T}$, it holds that $\text{CERTAIN}_{\mathcal{M}'}(\neg Q(\bar{x}), I) = \varnothing$. In fact, this property holds even if Σ' is restricted to contain a single *copying dependency* $A(x) \rightarrow A'(x)$. Thus, the class of FO-TO-CQ dependencies is not expressive enough to capture the certain answers semantics for the negation of conjunctive queries, which is required for expressing the closure of a mapping.

Proposition 1 tells us that some restrictions on the target instances considered as solutions have to be imposed in order to obtain the closure of a mapping. Target dependencies have been used in data exchange as a primary way to restrict the spaces of solutions of source instances [Arenas *et al.*, 2014a], so they form a natural alternative to solve this problem. Unfortunately, the following proposition shows that this alternative does not work. Notice that in this proposition we use notation $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma, \Gamma)$ to represent a mapping with target dependencies, where Σ is a set of FO-TO-CQ dependencies and Γ is a set of FO-sentences over $\mathbf{T}$. Moreover, we assume that $J \in \text{SOL}_{\mathcal{M}}(I)$ if and only if $(I, J) \models \Sigma$ and $J \models \Gamma$. Again, this result holds even for a mapping $\mathcal{M}$ specified by a single copying dependency $A(x) \rightarrow A'(x)$.

Proposition 2. *There is a mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where Σ is a set of st-tgds, for which no mapping $\mathcal{M}' = (\mathbf{S}, \mathbf{T}, \Sigma', \Gamma)$ is a closure, assuming that Σ' is a set of* FO-TO-CQ *dependencies and Γ is a set of FO-sentences over $\mathbf{T}$.*

Bidirectional constraints come to the rescue at this point, as it is possible to show that every mapping given by st-tgds has a closure specified by a set of bidirectional constraints. In the rest of this section, we outline the proof of this result. In fact, we provide an algorithm that given a mapping $\mathcal{M}$ specified by a set of st-tgds, computes a mapping $\text{COMP}(\mathcal{M})$ specified by a set of bidirectional constraints such that $\text{COMP}(\mathcal{M})$ is a closure of $\mathcal{M}$.

Our algorithm is based on query rewriting. Next we review the terminology and results about this concept needed in our algorithm. Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where Σ is a set of st-tgds, and assume that Q is a query over $\mathbf{T}$. Then a query Q' over $\mathbf{S}$ is a *rewriting of Q over the source* if for every $I \in \text{Inst}(\mathbf{S})$, it holds that $Q'(I) = \text{CERTAIN}_{\mathcal{M}}(Q, I)$. That is, to obtain the set of certain answers of Q over I under $\mathcal{M}$, we just have to evaluate its rewriting Q' over instance I.

The computation of a rewriting of a conjunctive query is a basic step in the algorithm presented in this section. This problem has been extensively studied in the literature [Levy *et al.*, 1995; Abiteboul and Duschka, 1998] and, in particular, in the data integration context [Halevy, 2000; 2001; Lenzerini, 2002]. In fact, it is known that there exists an exponential-time algorithm QUERYREWRITING that given a mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, with Σ a set of st-tgds, and a conjunctive query Q over $\mathbf{T}$, computes a query Q' over $\mathbf{S}$ such that Q' is in $\text{UCQ}^=$ and Q' is a rewriting of Q over the source.

We are ready to present our algorithm to compute a closure of a mapping. In this algorithm, if $\bar{x} = (x_1, \ldots, x_k)$, then $\mathbf{C}(\bar{x})$ is a shorthand for $\mathbf{C}(x_1) \wedge \cdots \wedge \mathbf{C}(x_k)$, where $\mathbf{C}(\cdot)$ is the unary built-in predicate introduced in the preliminaries which distinguishes between constants and nulls.

Algorithm CLOSURE($\mathcal{M}$)
Input: $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where Σ is a set of st-tgds.
Output: $\text{COMP}(\mathcal{M}) = (\mathbf{S}, \mathbf{T}, \Delta)$, where Δ is a set of bidirectional constraints and $\text{COMP}(\mathcal{M})$ is a closure of $\mathcal{M}$.

(1) Start with Δ as the empty set.
(2) For every st-tgd $\varphi(\bar{x}) \rightarrow \psi(\bar{x}) \in \Sigma$, do the following:
   (a) Use $\text{QUERYREWRITING}(\mathcal{M}, \psi(\bar{x}))$ to compute a query $\alpha(\bar{x})$ that is a rewriting of $\psi(\bar{x})$ over the source.
   (b) Add dependency $\alpha(\bar{x}) \leftrightarrow (\psi(\bar{x}) \wedge \mathbf{C}(\bar{x}))$ to Δ.
(3) Return $\text{COMP}(\mathcal{M}) = (\mathbf{S}, \mathbf{T}, \Delta)$. □

The following theorem establishes the correctness of algorithm CLOSURE. Its proof is direct, and it is based on some general properties regarding the relationship between the solutions under a mapping and the solutions under its closure.

Theorem 1. *Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where Σ is a set of st-tgds. Then* CLOSURE($\mathcal{M}$) *computes a closure of $\mathcal{M}$ in exponential time in the size of Σ, which is specified by a set of $\langle \text{UCQ}^=, \text{CQ} \rangle$-dependencies.*

We conclude this section by pointing out that the previous result provides evidence of the richness of the language of bidirectional constraints. In what follows, we continue our investigation of this language by studying two fundamental problems in data exchange, namely the existence of solutions and the query evaluation problems.

	full $\langle \text{CQ}, \text{CQ} \rangle$-dependencies	full $\langle \text{UCQ}^{=}, \text{CQ} \rangle$-dependencies	$\langle \text{CQ}, \text{CQ} \rangle$-dependencies	$\langle \text{UCQ}^{=}, \text{CQ} \rangle$-dependencies
data complexity	in PTIME	in PTIME	NP-complete	NP-complete
combined complexity	Π_2^P-complete	Π_2^P-complete	NEXPTIME-complete	NEXPTIME-complete

Table 1: The complexity of the existence of solutions problem.

4 The Existence of Solutions Problem

As pointed out before, one of the main problems to be solved in the data exchange area is the issue of computing a valid translation of a source instance according to a mapping. The natural decision problem associated to this issue is what has been called the existence of solutions problem: to decide, given a mapping $\mathcal{M}$ and a source instance I, whether $\text{SOL}_{\mathcal{M}}(I) \neq \varnothing$. For the case of a mapping specified by a set of st-tgds, this problem is trivial as every source instance has a solution. However, for more expressive mapping languages the situation can be different, in particular for the case of bidirectional constraints as shown in the following example.

Example 1. Take a simple setting $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, with $\Sigma = \{A(x) \rightarrow R(x), B(x) \rightarrow R(x)\}$. Then for the source instance $I = \{B(1)\}$, we have that the instance $J = \{R(1)\}$ is a solution under $\mathcal{M}$. Now consider the setting $\mathcal{M}' = (\mathbf{S}, \mathbf{T}, \Delta)$, with Δ consisting of the following bidirectional constraints:

$$A(x) \leftrightarrow R(x) \text{ and } B(x) \leftrightarrow R(x).$$

As opposed to the previous case, I does not have any solution under $\mathcal{M}'$. □

In this section, we embark on a study of the complexity of the existence of solutions problem for different types of bidirectional constraints. Formally, given a class $\mathcal{C}$ of bidirectional constraints, this problem is defined as follows:

Problem:	EXISTENCEOFSOLUTIONS($\mathcal{C}$)
Input:	Mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Delta)$, where $\Delta \subseteq \mathcal{C}$, and an instance I of $\mathbf{S}$
Question:	Is $\text{SOL}_{\mathcal{M}}(I) \neq \varnothing$?

The starting point of our investigation is the class of $\langle \text{CQ}, \text{CQ} \rangle$-dependencies, which is the natural counterpart of the class of st-tgds. Moreover, we consider the class of $\langle \text{UCQ}^{=}, \text{CQ} \rangle$-dependencies, as they are needed in the algorithm in Section 3 to compute the closure of a mapping given by st-tgds. Finally, we consider the restrictions of these two classes to the case of full dependencies, as full source-to-target dependencies are widely used in theory and practice.

It is important to notice that the previous definition corresponds to the *combined complexity* [Vardi, 1982] of the existence of solutions problem. As mapping specifications are usually much smaller than source instances, it is also natural in this context to consider the *data complexity* [Vardi, 1982] of this problem, which is defined by assuming that a mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ is fixed, and then considering the input to be the source instance alone. This problem is denoted by EXISTENCEOFSOLUTIONS($\mathcal{M}$).

The following theorem summarizes both the combined complexity results obtained in our investigation and, for the sake of completeness, the data complexity results obtained in [Arenas *et al.*, 2014b]. The results in this theorem should be read as follows. Let $\mathcal{C}$ be a class of bidirectional constraints. When we refer to the combined complexity of the existence of solutions problem for $\mathcal{C}$, we are talking about the complexity of EXISTENCEOFSOLUTIONS($\mathcal{C}$). On the other hand, when we say that the data complexity of the existence of solutions problem for $\mathcal{C}$ is in a complexity class $\mathcal{N}$, we mean that EXISTENCEOFSOLUTIONS($\mathcal{M}$) $\in \mathcal{N}$ for every mapping $\mathcal{M}$ defined by a set of bidirectional constraints in $\mathcal{C}$. Moreover, when we say that the data complexity of the existence of solutions problem for $\mathcal{C}$ is $\mathcal{N}$-complete, we mean that the data complexity of the existence of solutions problem for $\mathcal{C}$ is in $\mathcal{N}$, and there exists a mapping $\mathcal{M}_0$ defined by a set of bidirectional constraints in $\mathcal{C}$ such that EXISTENCEOFSOLUTIONS($\mathcal{M}_0$) is $\mathcal{N}$-hard.

Theorem 2. *Let $\mathcal{C}$ be any of the classes of bidirectional constraints shown in the header of Table 1. Then the data and combined complexity of the existence of solutions problems for $\mathcal{C}$ are as stated in Table 1.*

The upper bounds in Table 1 for the combined complexity of the existence of solutions problem are derived directly. The lower bound in this table for the case of $\langle \text{CQ}, \text{CQ} \rangle$-dependencies, and thus also for the case of $\langle \text{UCQ}^{=}, \text{CQ} \rangle$-dependencies, is shown via a reduction from TILING [Papadimitriou, 1994], a well-known NEXPTIME-complete problem. The input of this problem is a set of tile types $T = \{t_0, \ldots, t_m\}$, relations $H \subseteq T \times T$ and $V \subseteq T \times T$ (which represent horizontal and vertical adjacency constraints between tile types), and an integer n given in unary. Then the question to answer is whether there exists a tiling of a $2^n \times 2^n$ square with tile types in T, starting with tile type t_0 in the origin and satisfying the constraints imposed by H and V. The lower bound in Table 1 for the case of full $\langle \text{CQ}, \text{CQ} \rangle$-dependencies, and thus also for the case of full $\langle \text{UCQ}^{=}, \text{CQ} \rangle$-dependencies, is shown via a reduction from the problem of determining if a quantified Boolean formula of the form $\forall \bar{x} \exists \bar{y}\, \varphi(\bar{x}, \bar{y})$ is true, where φ is a propositional formula in 3-CNF and $\bar{x}$, $\bar{y}$ form a partition of the variables mentioned in φ. This problem is known to be Π_2^P-complete [Stockmeyer, 1976; Wrathall, 1976].

5 The Query Evaluation Problem

A fundamental problem in data exchange is the computation of certain answers. In this section, we study in depth the complexity of this problem for the case of bidirectional constraints. In particular, we are interested in the complexity of answering non-monotone queries, thus going beyond the more classical scenario of positive queries.

Given a class $\mathcal{C}$ of bidirectional constraints and a class $\mathcal{Q}$ of queries, the problem that we study is defined as follows:

	CQ	MON	$\text{UCQ}^{\neg[1]}$	$\text{CQ}^{\neg[2]}$	$\text{UCQ}^{\neg}$	FO
full $\langle \text{UCQ}^{=}, \text{CQ} \rangle$-dependencies	in PTIME	in PTIME	in PTIME	coNP-complete	coNP-complete	undecidable
$\langle \text{UCQ}^{=}, \text{CQ} \rangle$-dependencies	coNP-complete	coNP-complete	coNP-complete	coNP-complete	coNP-complete	undecidable

Table 2: The data complexity of the certain answers problem.

	CQ	$\text{UCQ}^{\neq}$	$\text{CQ}^{\neg[1]}$	$\text{UCQ}^{\neg[1]}$	$\text{CQ}^{\neg[2]}$	$\text{UCQ}^{\neg}$
full $\langle \text{UCQ}^{=}, \text{CQ} \rangle$-dependencies	Σ_2^P-complete	Σ_2^P-complete	EXPTIME--complete	EXPTIME--complete	coNEXPTIME--complete	coNEXPTIME--complete
$\langle \text{UCQ}^{=}, \text{CQ} \rangle$-dependencies	coNEXPTIME--complete	coNEXPTIME--complete	coNEXPTIME--complete	coNEXPTIME--complete	coNEXPTIME--complete	coNEXPTIME--complete

Table 3: The combined complexity of the certain answers problem.

Problem:	$\text{CERTAINANSWERS}(\mathcal{C}, \mathcal{Q})$
Input:	Mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Delta)$, where $\Delta \subseteq \mathcal{C}$, n-ary query $Q \in \mathcal{Q}$ over $\mathbf{T}$, n-tuple $\bar{a}$, and an instance I over $\mathbf{S}$.
Question:	Is $\bar{a}$ in $\text{CERTAIN}_{\mathcal{M}}(Q, I)$?

As in the previous section, and given that queries are usually much smaller than database instances, we also consider the data complexity of the problem, in which the mapping $\mathcal{M}$ and query Q are considered to be fixed:

Problem:	$\text{CERTAINANSWERS}(\mathcal{M}, Q)$
Input:	n-tuple $\bar{a}$, and an instance I over $\mathbf{S}$.
Question:	Is $\bar{a}$ in $\text{CERTAIN}_{\mathcal{M}}(Q, I)$?

To study the complexity of these problems we consider the class of conjunctive queries (CQ) and its extensions with disjunctions and negations. The issue of negation is specially sensible and the results vary if we consider one, two or many negations in the queries. In particular, we denote by $\text{CQ}^{\neg[n]}$ the class of conjunctive queries with at most n negated atoms, and by $\text{UCQ}^{\neg[n]}$ the class of unions of conjunctive queries with at most n negated atoms per disjunct (recall that $\text{CQ}^{\neg}$ and $\text{UCQ}^{\neg}$ represent the classes of conjunctive queries with negation and unions of conjunctive queries with negation, respectively, without restrictions on the number of negated atoms).

When we refer to the combined complexity of the certain answers problem for $\mathcal{C}$ and $\mathcal{Q}$, we are talking about the complexity of $\text{CERTAINANSWERS}(\mathcal{C}, \mathcal{Q})$. When we say that the data complexity of the certain answers problem for $\mathcal{C}$ and $\mathcal{Q}$ is in a complexity class $\mathcal{N}$, we mean that for every mapping $\mathcal{M}$ defined by some constraints in $\mathcal{C}$ and for every query $Q \in \mathcal{Q}$, the problem $\text{CERTAINANSWERS}(\mathcal{M}, Q)$ is in $\mathcal{N}$. Moreover, we say that the data complexity of the certain answers problem for $\mathcal{C}$ and $\mathcal{Q}$ is $\mathcal{N}$-complete, if the problem is in $\mathcal{N}$ and there exists a mapping $\mathcal{M}_0$ defined by some constraints in $\mathcal{C}$ and a query $Q_0 \in \mathcal{Q}$ such that $\text{CERTAINANSWERS}(\mathcal{M}_0, Q_0)$ is $\mathcal{N}$-hard. Finally, we say that the data complexity of the certain answers problem for $\mathcal{C}$ and $\mathcal{Q}$ is undecidable if there exists a mapping $\mathcal{M}_1$ defined by some constraints in $\mathcal{C}$ and a query $Q_1 \in \mathcal{Q}$ such that $\text{CERTAINANSWERS}(\mathcal{M}_1, Q_1)$ is undecidable.

Theorem 3. *Let $\mathcal{C}$ be either the class of full $\langle \text{UCQ}^{=}, \text{CQ} \rangle$-dependencies or the class of $\langle \text{UCQ}^{=}, \text{CQ} \rangle$-dependencies.*

(1) If $\mathcal{Q} \in \{\text{CQ}, \text{MON}, \text{UCQ}^{\neg[1]}, \text{CQ}^{\neg[2]}, \text{UCQ}^{\neg}, \text{FO}\}$, then the data complexity of the certain answers problem for $\mathcal{C}$ and $\mathcal{Q}$ is as stated in Table 2.

(2) If $\mathcal{Q} \in \{\text{CQ}, \text{UCQ}^{\neq}, \text{CQ}^{\neg[1]}, \text{UCQ}^{\neg[1]}, \text{CQ}^{\neg[2]}, \text{UCQ}^{\neg}\}$, then the combined complexity of the certain answers problem for $\mathcal{C}$ and $\mathcal{Q}$ is as stated in Table 3.

Now we briefly discuss our results, starting with the data complexity of the certain answers problem. The coNP-completeness of the problem for $\langle \text{UCQ}^{=}, \text{CQ} \rangle$-dependencies is consistent with the NP-completeness of the existence of solutions problem for this type of dependencies (shown in Table 1). In fact, the lower bound for the case of conjunctive queries, which is also a lower bound for all the other query languages considered in Table 2, is proved by slightly modifying a reduction from 3-COLORABILITY used in [Arenas *et al.*, 2014b] to prove the NP-hardness (in data complexity) of the existence of solutions problem for the case of $\langle \text{CQ}, \text{CQ} \rangle$-dependencies. The upper bound for $\text{UCQ}^{\neg}$ queries, which is also an upper bound for the query languages in the first four columns of Table 2, is shown by using a modified version of the chase procedure [Maier *et al.*, 1979], inspired by the techniques used in [Fagin *et al.*, 2005a] and [Fuxman *et al.*, 2006]. In the case of full $\langle \text{CQ}, \text{CQ} \rangle$-dependencies, the chase procedure is again used to prove the tractability of the problem for $\text{UCQ}^{\neg[1]}$ queries, and the lower bound for $\text{CQ}^{\neg[2]}$ queries is proved by reducing 3-CNF-SAT to the complement of our problem. It is important to mention that the tractability for the case of monotone queries requires of some techniques tailored to them, since this class of queries has a semantic definition, as opposed to the other query languages in this article that have syntactic definitions. Notice that $\text{UCQ}^{\neq}$ queries are monotone, and thus the tractability for this class is obtained from the tractability for monotone queries. Finally, if we consider FO queries, then the problem becomes undecidable. To prove this result we consider the embedding problem for finite semigroups, which is an undecidable problem [Evans, 1951; 1953; 1978; Gurevich, 1966] that has as input a finite partial semigroup, and asks whether it can be embedded to a finite semigroup. More precisely, we prove the undecidability for the case of FO by constructing a fixed mapping $\mathcal{M}$ and a fixed FO query Q, and then providing a reduction from the embedding problem for finite semigroups to the complement of

CERTAINANSWERS$(\mathcal{M}, Q)$.

In the case of combined complexity, the proofs are similar. We just would like to point out here that for $\text{UCQ}^{\neg[1]}$ queries, the upper bound follows directly from the techniques used in the study of the data complexity for the same class, but not fixing the mapping and the query, while the EXPTIME-hardness is proved by providing a reduction from the evaluation problem for Datalog programs containing a single rule, which is an EXPTIME-complete problem [Gottlob and Papadimitriou, 2003; Kolaitis *et al.*, 2006]. Besides, for the case of $\text{CQ}^{\neg[2]}$ queries, the lower bound is shown by providing a reduction from the complement of the TILING problem defined in Section 4.

The computation of certain answers has been extensively studied for the case of st-tgds [Fagin *et al.*, 2003; Arenas *et al.*, 2004; Libkin, 2006; Hernich, 2013] and also for a setting including target-to-source dependencies [Fuxman *et al.*, 2006]. The latter case could be considered as a generalization of our scenario, as a bidirectional constraint consists of an st-tgd and a target-to-source dependency. However, this derived target-to-source dependency may have disjunctions in the right-hand side, making our setting incomparable with the framework of [Fuxman *et al.*, 2006] where only tuple-generating dependencies are considered. Besides this, neither the combined complexity of the query evaluation problem nor the case of non-monotone queries were considered in [Fuxman *et al.*, 2006]. It is worth mentioning, though, that some of the techniques introduced in [Fuxman *et al.*, 2006] proved to be useful in our case, and inspired some of our results.

5.1 LAV, GAV, and some tractable cases

As it can be seen in Tables 2 and 3, most of the results on the complexity of the query evaluation problem are negative. Thus, it is natural to ask what kind of conditions could be imposed in order to get tractability. In this section, we study the computation of certain answers in two restricted scenarios that have been widely studied in the data integration/exchange literature [Lenzerini, 2002] and have been widely used in practice, namely the *local-as-view* (LAV) and the *global-as-view* (GAV) scenarios.

A set Δ of $\langle \text{UCQ}^{=}, \text{CQ} \rangle$-dependencies is said to be LAV if the left-hand side of every dependency in Δ is a conjunctive query mentioning a single relational atom. Similarly, Δ is said to be GAV if the right-hand side of every dependency in Δ is a single relational atom (not including existential quantifiers). A further restriction is to consider *true-LAV* and *true-GAV* dependencies [Arocena *et al.*, 2010]. In true-LAV, the left-hand side of every dependency must be a conjunctive query mentioning a single relational atom which has no repeated variables, and every source relation must appear in a dependency. Similarly, in true-GAV, the right-hand of every dependency must be a single relational atom with no repeated variables (and no existential quantifiers), and every target relation should appear in the right-hand side of a dependency.

Example 2. Dependency (3) in Section 1 is true-LAV. $\Gamma = \{A(x, y) \wedge B(y) \leftrightarrow R(x, x, y)\}$ is GAV, but not true-GAV. The set Δ in Example 1 is true-LAV and true-GAV. □

Unfortunately, the LAV and true-LAV restrictions do not reduce the complexity of the query evaluation problem. More specifically, all the lower bounds in Table 2 hold for the case of true-LAV (and, thus, for the case of LAV). On the other hand, the query answering problem can be easily solved for the case of true-GAV. In fact, if $\mathcal{M}$ is defined by a set of true-GAV dependencies, then it can be proved that for every instance I such that $\text{SOL}_{\mathcal{M}}(I) \neq \varnothing$, it holds that $|\text{SOL}_{\mathcal{M}}(I)| = 1$. Hence, the certain answers can be obtained by just executing the input query over a single solution.

Proposition 3. CERTAINANSWERS$(\mathcal{M}, Q)$ *is in* PTIME*, for every mapping $\mathcal{M}$ specified by a set of true-GAV $\langle \text{UCQ}^{=}, \text{CQ} \rangle$-dependencies and every query Q in* FO.

For the case of GAV, the query answering problem is no longer trivial as an instance can have many (potentially infinite) solutions. For instance, if Γ is the set of dependencies in Example 2 and $I = \{A(1, 2), B(2)\}$, then for every tuple $R(a, b, c)$ such that $a \neq b$, we have that $J = \{R(1, 1, 2), R(a, b, c)\}$ is a solution for I. In fact, for the case of GAV, it can be proved that the query answering problem is undecidable in data complexity for FO. Nevertheless, the next result shows that for a large class of non-monotone queries, GAV dependencies do have good computational properties with respect to query answering.

Theorem 4. CERTAINANSWERS$(\mathcal{M}, Q)$ *is in* PTIME*, for every mapping $\mathcal{M}$ specified by a set of GAV $\langle \text{UCQ}^{=}, \text{CQ} \rangle$-dependencies and every query Q in* $\text{UCQ}^{\neg}$.

Theorem 4 is proved by providing a polynomial time algorithm for CERTAINANSWERS$(\mathcal{M}, Q)$ which uses a subroutine for the problem of verifying whether a truth assignment satisfies a propositional formula. Given that the latter can be solved in polynomial time, we conclude that CERTAINANSWERS$(\mathcal{M}, Q)$ can be solved in polynomial time.

6 Concluding Remarks and Future Work

We have considered the use of bidirectional constraints to specify schema mappings, showing their usefulness to define the closure of a schema mapping. We have studied two fundamental problems in data exchange (existence of solutions and query evaluation) for these new mapping specifications, obtaining a wide range of results for different scenarios.

Several issues deserve further investigation. First, some properties of the notion of closure of a mapping need to be investigated in more depth, in particular, whether the closure of a mapping is unique, and if this is not the case, what is the relationship between the different closures of a mapping. Second, a deeper study of the connection between our proposal and some existing semantics for non-monotone queries needs to be conducted, in particular in the data exchange context [Libkin, 2006; Hernich, 2013].

Acknowledgements

The authors would like to thank Juan Reutter and the anonymous referees for their comments and suggestions. The authors were partially funded by the Millennium Nucleus Center for Semantic Web Research under Grant NC120004. M. Arenas and G. Diéguez were also funded by Fondecyt grant 1131049.

References

[Abiteboul and Duschka, 1998] Serge Abiteboul and Oliver M. Duschka. Complexity of answering queries using materialized views. In *PODS*, pages 254–263, 1998.

[Afrati and Kolaitis, 2008] Foto N. Afrati and Phokion G. Kolaitis. Answering aggregate queries in data exchange. In *PODS*, pages 129–138, 2008.

[Arenas *et al.*, 2004] Marcelo Arenas, Pablo Barceló, Ronald Fagin, and Leonid Libkin. Locally consistent transformations and query answering in data exchange. In *PODS*, pages 229–240, 2004.

[Arenas *et al.*, 2014a] Marcelo Arenas, Pablo Barceló, Leonid Libkin, and Filip Murlak. *Foundations of Data Exchange*. Cambridge University Press, 2014.

[Arenas *et al.*, 2014b] Marcelo Arenas, Gabriel Diéguez, and Jorge Pérez. Expressiveness and complexity of bidirectional constraints for data exchange. In *Alberto Mendelzon Workshop*, 2014.

[Arocena *et al.*, 2010] Patricia C. Arocena, Ariel Fuxman, and Renée J. Miller. Composing local-as-view mappings: closure and applications. In *ICDT*, pages 209–218, 2010.

[Bernstein *et al.*, 2006] Philip A. Bernstein, Todd J. Green, Sergey Melnik, and Alan Nash. Implementing mapping composition. In *VLDB*, pages 55–66, 2006.

[Clark, 1977] Keith L. Clark. Negation as failure. In *Logic and Data Bases*, pages 293–322, 1977.

[Evans, 1951] T. Evans. The word problem for abstract algebras. *Journal of the London Mathematical Society*, 26:64–71, 1951.

[Evans, 1953] T. Evans. Embeddability and the word problem. *Journal of the London Mathematical Society*, 28:76–80, 1953.

[Evans, 1978] T. Evans. Word problems. *Bulletin of the American Mathematical Society*, 84(5):789–802, 1978.

[Fagin *et al.*, 2003] Ronald Fagin, Phokion G. Kolaitis, Renée J. Miller, and Lucian Popa. Data exchange: Semantics and query answering. In *ICDT*, pages 207–224, 2003.

[Fagin *et al.*, 2005a] Ronald Fagin, Phokion G. Kolaitis, Renée J. Miller, and Lucian Popa. Data exchange: semantics and query answering. *Theor. Comput. Sci.*, 336(1):89–124, 2005.

[Fagin *et al.*, 2005b] Ronald Fagin, Phokion G. Kolaitis, and Lucian Popa. Data exchange: getting to the core. *TODS*, 30(1):174–210, 2005.

[Fuxman *et al.*, 2006] Ariel Fuxman, Phokion G. Kolaitis, Renée J. Miller, and Wang Chiew Tan. Peer data exchange. *TODS*, 31(4):1454–1498, 2006.

[Gottlob and Papadimitriou, 2003] Georg Gottlob and Christos H. Papadimitriou. On the complexity of single-rule datalog queries. *Inf. Comput.*, 183(1):104–122, 2003.

[Gottlob *et al.*, 2011] Georg Gottlob, Reinhard Pichler, and Vadim Savenkov. Normalization and optimization of schema mappings. *VLDB J.*, 20(2):277–302, 2011.

[Gurevich, 1966] Yuri Gurevich. The word problem for certain classes of semigroups. *Algebra and Logic*, 5:25–35, 1966.

[Haas *et al.*, 2005] Laura M. Haas, Mauricio A. Hernández, Howard Ho, Lucian Popa, and Mary Roth. Clio grows up: from research prototype to industrial tool. In *SIGMOD Conference*, pages 805–810, 2005.

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

[Halevy, 2001] Alon Y. Halevy. Answering queries using views: A survey. *VLDB J.*, 10(4):270–294, 2001.

[Hernich *et al.*, 2011] André Hernich, Leonid Libkin, and Nicole Schweikardt. Closed world data exchange. *TODS*, 36(2):14, 2011.

[Hernich, 2012] André Hernich. Computing universal models under guarded tgds. In *ICDT*, pages 222–235, 2012.

[Hernich, 2013] André Hernich. Semantics for non-monotone queries in data exchange and data integration. In *Data Exchange, Information, and Streams*, volume 5 of *Dagstuhl Follow-Ups*, pages 161–184. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2013.

[Kolaitis *et al.*, 2006] Phokion G. Kolaitis, Jonathan Panttaja, and Wang Chiew Tan. The complexity of data exchange. In *PODS*, pages 30–39, 2006.

[Lenzerini, 2002] Maurizio Lenzerini. Data integration: A theoretical perspective. In *PODS*, pages 233–246, 2002.

[Levy *et al.*, 1995] Alon Y. Levy, Alberto O. Mendelzon, Yehoshua Sagiv, and Divesh Srivastava. Answering queries using views. In *PODS*, pages 95–104, 1995.

[Libkin and Sirangelo, 2011] Leonid Libkin and Cristina Sirangelo. Data exchange and schema mappings in open and closed worlds. *J. Comput. Syst. Sci.*, 77(3):542–571, 2011.

[Libkin, 2006] Leonid Libkin. Data exchange and incomplete information. In *PODS*, pages 60–69, 2006.

[Maier *et al.*, 1979] David Maier, Alberto O. Mendelzon, and Yehoshua Sagiv. Testing implications of data dependencies. *TODS*, 4(4):455–469, 1979.

[Melnik *et al.*, 2008] Sergey Melnik, Atul Adya, and Philip A. Bernstein. Compiling mappings to bridge applications and databases. *TODS*, 33(4), 2008.

[Papadimitriou, 1994] Christos H. Papadimitriou. *Computational complexity*. Addison-Wesley, 1994.

[Reiter, 1977] Raymond Reiter. On closed world data bases. In *Logic and Data Bases*, pages 55–76, 1977.

[Stockmeyer, 1976] Larry J. Stockmeyer. The polynomial-time hierarchy. *Theor. Comput. Sci.*, 3(1):1–22, 1976.

[Vardi, 1982] Moshe Y. Vardi. The complexity of relational query languages (extended abstract). In *STOC*, pages 137–146, 1982.

[Wrathall, 1976] Celia Wrathall. Complete sets and the polynomial-time hierarchy. *Theor. Comput. Sci.*, 3(1):23–33, 1976.