Unique Solutions in Data Exchange under *sts* Mappings

Nhung Ngo and Enrico Franconi

KRDB Research Centre
Free University of Bozen-Bolzano, Italy
lastname@inf.unibz.it
http://www.inf.unibz.it/krdb/

Abstract In classical data exchange, multiple solutions may appear and inherently cause many problems. To tackle the problem, one may use a richer language for schema mapping to have a unique solution. Therefore, in the paper, we consider a data exchange setting in which schema mapping contains a set of source-to-target dependencies and a set of target-to-source dependencies (sts mappings). Under the setting, we first study the problem of deciding whether a data exchange setting has a unique solution with respect to a source instance. We show that the problem is as hard as Unique SAT problem and provide some restricted cases where the problem is tractable. Besides, we consider a more general problem that aims to check if a data exchange setting guarantees unique solutions for arbitrary source instances. While the problem is undecidable in general, we still can characterise some fragments where the problem is decidable and complete for some complexity class.

1 Introduction

Data exchange as a theoretical problem was introduced a decade ago in [10] and has been one of the most active research topics in foundation of databases due to the need for the exchange of data in many business applications. This is the problem of transforming data structured under a source schema into data structured under a target schema. Given a source instance, the purpose of data exchange is to materialise a valid target instance (called a solution) respecting the schema mapping, specifications that describe the relationship between data in the two heterogeneous schemas – the source and the target.

Classically, schema mappings in data exchange are written as source-to-target tuple generating dependencies (s-t tgds) to specify that if some positive patterns hold in the source, then some corresponding positive patterns must hold in the target as well. Source-to-target tuple generating dependencies are existential rules with positive conjunctive body and positive conjunctive head; the head may contain existentially quantified variables. Under a s-t tgds mapping, there might be more than one solution corresponding to a given source database because the target instances might contain additional facts or unknown facts. That is, the target database is actually an *incomplete database*, in the sense of classical database theory, namely it is a *set* of possible databases. This leads to a mismatch between the purpose of data exchange-materialising a valid target database and its specifications-generating multiple valid instances.

After data is exchanged and a target instance is generated, one may want to do query answering the target data. As a consequence of incomplete database, the problem is complex and non-intuitive for general (non-positive) relational or aggregate queries, since it is basically comparable to entailment with open-world semantics (namely the computation of *certain answers*), and standard relational database technologies can not be used. Anomalies caused by certain answer semantics are mentioned in [2,10,1,12], and nicely summarised by [14] thought a set of examples.

In order to solve the query answering problem, the classical data exchange framework restricts the target query language to just monotone queries (i.e., positive queries or union of conjunctive queries). It turns out that the *certain* answer to monotone queries over the incomplete target database are the same as the answers of the same query over a representative specific database (one of the so called *universal solutions* – the *core* being a minimal among them) [10]. With this restriction on the query language, query answering over the target databases becomes meaningful and efficient.

To give meaning to more expressive queries (i.e queries with negation or aggregation), various interesting extensions have been proposed to restrict the uncertainty of the target instance. Extensions that are based on semantic restriction include Close World Assumption (CWA) semantics by Libkin [13] and GCWA* semantics by Hernich [12]. Regarding to restrictions on syntax, a mapping language which is more expressive than s-t tgds, namely, bidirectional tgds were considered by Arenas et. al. [3]. However, these syntax and semantic restrictions are not strong enough in general to eliminate completely the uncertainty of target instances. As a matter of fact, they can not rule out all anomalies in query answering and also do not satisfy the ultimate goal of data exchange - materialising a valid target instance. Another way to deal with the problem is to use the definability abduction approach [15] that aims at finding extensions (including $t-s$ tgds) to the initial schema mappings to guarantee the uniqueness of the materialised target instance. The results mentioned in this paper can be considered as the complexity analysis of the $t-s$ tgds extension in the approach.

Contribution. In this paper, we are interested in data exchange settings where there is no ambiguity in selecting a target instance to be materialised and consequently, query answering can be done properly through the target instance. In other words, given a data exchange setting we would like to check if schema mapping rules are strong enough to guarantee the uniqueness of valid target instance. Obviously, we always get a negative answer if the language of schema mapping is s-t tgds. Therefore, in the paper, we consider both source-to-target tgds and target-to-source tgds (t-s tgds) for the mapping rules. The mapping is called *sts* mapping in this paper.

The mapping language was first considered in peer data exchange [11] in which a source peer may contribute data for a target peer through s-t tgds and a target peer may use t-s tgds to restrict the data it receives. The mapping language is more expressive than the bidirectional one in [3] since it contains arbitrary t-s tgds, not only the inverses of s-t tgds.

Given a data exchange setting in which schema mapping is in sts, we study the following decision questions.

1. Does the data exchange setting have an unique solution w.r.t a specific source instance?
2. Does the data exchange setting guarantee a unique solution for any source instance?

As an example for the first problem, consider the source schema with two relations {*Employee(EMPid), Phone(EMPid, PhNum)*}, and a target schema with one relation *Contact(CONid, PhNum)*. Given a source database {*Employee(1), Phone(1, 123)*} and the following mappings:

$$\begin{aligned} \mathit{Employee}(x) &\rightarrow \exists y\, \mathit{Contact}(x, y) \\ \mathit{Contact}(x, y) &\rightarrow \mathit{Phone}(x, y) \end{aligned}$$

Obviously, there is only one target instance {*Contact(1, 123)* } that together with the source instance forms a logical model of the mappings and therefore the decision procedure should give a positive answer. If we change the source database to {*Employee(1), Phone(1, 123), Phone(2, 234)*}, we do not have any more a unique solution and receive a negative answer.

To illustrate for the second problem, consider the source database with two relations with the schema {*Employee(EMPid), Manager(MANid)*}, and a target database with one relation with the schema *Staff(STid)*. In order to move the data from each of the source relation to the target relation we could state the following mappings:

$$\begin{aligned} \mathit{Employee}(x) &\rightarrow \mathit{Staff}(x) \\ \mathit{Manager}(x) &\rightarrow \mathit{Staff}(x) \\ \mathit{Staff}(x) &\rightarrow \mathit{Employee}(x) \end{aligned}$$

These mappings guarantee that for any source instance, there is at most one solution for it under these mappings and therefore the decision procedure should give a positive answer. Note that the mapping rules do not accept every source instance, i.e source databases in which there is some manager which is not an employee have no solution.

Note that our notion of unique solution is not related to the unique-solution property mentioned in [9] about inverting schema mapping, since the latter requires distinct source instances to have distinct sets of solutions.

Organisation. We start with some preliminary notions and definitions related to unique solution. In Section 3, we present the complexity results of checking the existence of unique solution w.r.t a source instance. Next section is devoted to the problem of guarantee unique solution. As usual, at the end is conclusion and outlook.

2 Formal Preliminaries

A *schema* is a finite set of predicate names with associated arities. Let $\mathbf{R}$ be a schema, $\mathbf{R} = \{R_1, ..., R_k\}$, an *instance* I over $\mathbf{R}$ is an union of $R_1^I, R_2^I, ..., R_k^I$ such that each R_i^I is a finite set of tuples having the same arity as R_i. If I is an instance of schema $\mathbf{S}_1$ and J is an instance of schema $\mathbf{S}_2$, we use (I, J) to denote an instance of schema $\mathbf{S}_1 \cup \mathbf{S}_2$.

We consider a classical first-order logic setting to define the semantics of the framework. If I is an instance and ϕ is a logic formula, we write $I \models \phi$ if I satisfies ϕ in the first-order logic sense. If Σ is a set of formulas, we write $I \models \Sigma$ to mean $I \models \phi$ for every $\phi \in \Sigma$. Given an instance I, we use $Adom(I)$ to describe the set of all constants appearing in I.

Given a set of sentences Φ, we use $\sigma(\Phi)$ to denote the signature of Φ, i.e. the set of all non-logical symbols in Φ.

Dependencies. A *tuple generating dependency* (tgd) is a sentence of the form $\forall \bar{x}, \bar{z}(\varphi(\bar{x}, \bar{z}) \rightarrow \exists \bar{y} \psi(\bar{x}, \bar{y}))$, where φ and ψ are conjunctions of atoms. For the sake of readability, we write $\varphi(\bar{x}, \bar{z}) \rightarrow \exists \bar{y} \psi(\bar{x}, \bar{y})$ instead of the full formula. We focus on simple tgds, i.e., tgds in which every atom does not contain occurrences of constants and repeated occurrences of variables.

A tgd is a *source-to-target tgd* (s-t tgd) if φ and ψ are formulas over $\mathbf{S}$ and $\mathbf{T}$ respectively, and vice versa, a tgd is a *target-to-source tgd* (t-s tgd) if φ and ψ are formulas over $\mathbf{T}$ and $\mathbf{S}$ respectively. A tgd is *full* if $\bar{y} = \emptyset$, otherwise, it is *embedded*. A tgd is a local-as-view (LAV) dependency if φ is an atom. A LAV dependency is *complete* if $\bar{z} = \emptyset$.

Data Exchange. A data exchange setting is a tuple $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where the set Σ is referred to *schema mapping*, $\mathbf{S}$ and $\mathbf{T}$ denote the source and target schemas, respectively. In the following we consider only data exchange settings $\mathcal{M}$ in which $\Sigma = \Sigma_{st} \cup \Sigma_{ts}$ and Σ_{st} is a set of s-t tgds and Σ_{ts} is a set of t-s tgds. We also assume that all the predicates from $\mathbf{T}$ appear in the schema mapping, i.e. the mappings tell us some information about each target predicate.

By a *solution* to the data exchange setting $\mathcal{M}$ for a source instance I, we mean a target instance J such that $(I, J) \models \Sigma$. We use the notation $Sol(\mathcal{M}, I)$ to denote the set of all solutions to the data exchange setting $\mathcal{M}$ for a source instance I.

Chase. Given an instance I and a set of tgds Σ, if the chase of I with Σ is finite, we denote by $chase_{\Sigma}(I)$ the result of the finite chase. Please refer to [10] for the detailed definition of chase.

Unique solution. We now define some notions related to unique solution that will be used later to define our main problems.

Definition 1. *Given a data exchange setting $\mathcal{M}$ and a source instance I, $(\mathcal{M}, I)$* has a unique solution *iff there is only one J such that J is a solution of $\mathcal{M}$ w.r.t the source instance I.*

Without a specific source instance, we are interested in the following property of data exchange setting.

Definition 2. *Given a data exchange setting $\mathcal{M}$, we say that $\mathcal{M}$* guarantees unique solutions *iff for every source instance I, if there is a solution J of $\mathcal{M}$ w.r.t I then $(\mathcal{M}, I)$ has a unique solution.*

3 Existence of Unique Solution

In this section, we study the complexity of checking if $(\mathcal{M}, I)$ has a unique solution. Formally, given a data exchange setting $\mathcal{M}$, the problem is defined as follows.

```
Problem: $ExistenceOfUniqueSolution(\mathcal{M})$
Input: source instance I
Output: Is there a unique target instance $J$ such
that $J \in Sol(\mathcal{M}, I)$?
```

We start our complexity analysis with a general setting in which mapping rules are arbitrary tgds. As we mentioned in the introduction, by considering also t-s tgds in the mapping, our data exchange setting is similar to a peer data exchange setting [11]. Since the data complexity of existent-of-solution problem in peer data exchange setting is NP-complete, we do not expect a lower complexity for $ExistenceOfUniqueSolution(\mathcal{M})$. Indeed, we show that the complexity connection between the two problems is analogous as the connection between SAT and Unique SAT [4].

Theorem 1. *Unique-solution-checking problem is in NP $\cap$ co-NP for data complexity.*

Proof. Let us start the proof with a simple observation about the unique solution of a given data exchange setting $\mathcal{M}$ and a source instance I. Intuitively, the observation implies the fact that the number of tuples in the unique solution is polynomially bounded by the input.

Lemma 1. *If $J \in Sol(\mathcal{M}, I)$ and J is unique then $Adom(J) \subseteq Adom(I)$.*

Proof. Assume that there is an element $a \in Adom(J)$ and $a \notin Adom(I)$. Let J' be a target instance which is the same as J but a is replaced by a' for some new constants $a' \notin Adom(I) \cup Adom(J)$. J' is isomorphic to J, therefore $J' \in Sol(\mathcal{M}, I)$. This leads to a contradiction to the fact that J is unique.

Because the schema is fixed, based on the lemma, it's straightforward to see that target instances which could be the unique solution of data exchange setting $\mathcal{M}$ for a source instance I have at most $|\mathbf{T}| \times |Adom(I)|^m$ tuples where $|\mathbf{T}|$ is the number of target predicates and m is the largest arity in $\mathbf{T}$.

As a consequence, we have the following naive algorithm to decide the problem.

1. Verify in NP if there is a solution by guessing a target instance J using only constants in $Adom(I)$ and checking if $J \in Sol(\mathcal{M}, I)$

2. Verify in co-NP if such the solution is unique as follows: For every pair of target instances (J, J') in which J contains only constants in $Adom(I)$ and J' contains constants in $Adom(I) \cup \{c_n\}$ ($c_n \notin Adom(I)$)
   (a) Check if J is a solution.
   (b) Check if $J \neq J'$.
   (c) Check if J' is also a solution.

□

We show the problem is complete for the class of NP ∩ co-NP using the following theorem.

Theorem 2. *Unique-solution-checking problem is NP ∩ co-NP hard.*

Proof. We prove the theorem by providing a reduction from Unique SAT to our problem.

Let ϕ be a propositional formula in CNF, $\phi = C_1 \wedge ... \wedge C_m$ where each C_i is a disjunction of literals among n variables $\{p_1, ..., p_n\}$.

We form $\mathcal{M}$ and I from ϕ as follows.

$$
\begin{array}{ll}
\mathbf{S} & = \{L(.,.), C(.), P(.), V(.), T(.), F(.), NotP(.), I(.,.)\}; \\
\mathbf{T} & = \{A(.,.), L'(.,.), T'(.), F'(.), I'(.,.)\}; \\
\Sigma_{st} & = \{L(i,p) \rightarrow L'(i,p), \\
& \quad T(x) \rightarrow T'(x),\ F(x) \rightarrow F'(x), \\
& \quad I(x,y) \rightarrow I'(x,y),\ P(x) \rightarrow \exists y A(x,y), \\
& \quad C(x) \rightarrow \exists yz.L'(x,y) \wedge A(y,z) \wedge T'(z)\} \\
\Sigma_{ts} & = \{L'(i,p) \rightarrow L(i,p), \\
& \quad T'(x) \rightarrow T(x),\ F'(x) \rightarrow F(x), \\
& \quad I'(x,y) \rightarrow I(x,y),\ A(x,y) \rightarrow V(y) \wedge P(x), \\
& \quad A(x,y) \wedge A(x,z) \wedge T(y) \wedge F(z) \rightarrow NotP(x), \\
& \quad A(x,y) \wedge A(x',y) \wedge I'(x,y') \rightarrow NotP(x)\} \\
I & = \{P(p_i), P(\bar{(p_i)}) | i = 1, n\} \cup \\
& \quad \{V(true), V(false)\} \cup \\
& \quad \{T(true), F(false)\} \cup \\
& \quad \{I(p_i, \bar{p_i}), I(\bar{p_i}, p_i))\} \cup \\
& \quad \{C(i) | i = 1, n\} \cup \\
& \quad \{L(i,p) | p \in C_i, i = 1, n)\} \cup \\
& \quad \{L(i,\bar{p}) | \neg p \in C_i, i = 1, n)\} \cup \\
& \quad \{NotP(-1)\}
\end{array}
$$

Intuitively, $\Sigma_{ts} \cup \Sigma_{st}$ guarantees A is a correct assignment which assigns: (i) truth values to propositions in ϕ and their negations; (ii) one truth value to one literals and (iii) opposite values to opposite literals. Besides, the last dependency in Σ_{st} implies that A form a model of ϕ.

Now we show that ϕ is unique SAT if and only if $(\mathcal{M}, I)$ has a unique solution.

If ϕ is unique SAT, let M be the unique model of ϕ. We consider target instance J such that $T'^J = T^I, L'^J = L^I, F'^J = F^I$ and $A^J = \{(p, true), (\bar{p}, false)|$

$p \in M\} \cup \{(\bar{p}, true), (p, false) | p \notin M\}$. Obviously $J \in Sol(\mathcal{M}, I)$ because $M \models \phi$ then for every clause C_i, there is some literal in C_i is assigned to *true*. If J is not unique then there is $J' \in Sol(\mathcal{M}, I)$ such that $A^{J'} \neq A^{J}$. Let $M' \neq M$ be the interpretation in which $p_i \in M'$ iff $(p_i, true) \in A^{J'}$, then for every clause C_i, there is some literal $l \in C_i$, $l \in M'$ if l is positive and $l \notin M'$ if l is negative. Consequently, $M' \models \phi$. This leads to a contradiction.

The inverse direction can be proved analogously.

Since Σ_{st} and Σ_{ts} do not depend in ϕ, we can conclude the problem is NP $\cap$ co-NP hard. $\square$

Now, let us identify some cases where the problem can be solved by a polynomial time algorithm.

Theorem 3. *Suppose we consider data exchange setting $\mathcal{M}$ in which dependencies from target to source are full tgds. Then $ExistenceOfUniqueSolution(\mathcal{M})$ is tractable.*

Proof. In the algorithm mentioned in the proof of Theorem 1, one does not know which could be the unique solution therefore we need to guess and check an arbitrary one. This step is not necessary in case Σ_{st} contains only full tgds because we know what could be the candidate for the unique solution.

Lemma 2. *Σ_{st} contains only full tgds. If $Sol(\mathcal{M}, I) \neq \emptyset$ then $J = chase_{\Sigma_{st}}(I) \in Sol(\mathcal{M}, I)$.*

Proof. Since $J = chase_{\Sigma_{st}}(I)$, $(I, J) \models \Sigma_{st}$. Assume that $J \notin Sol(\mathcal{M}, I)$, then $(I, J) \not\models \Sigma_{ts}$. Let J' be a target instance such that $J' \in Sol(\mathcal{M}, I)$. Because Σ_{st} contains only full tgds, J is also a core of the data exchange setting (Σ_{st}, S, T). Therefore, $J \subset J'$. Besides, since $(J', I) \models \Sigma_{ts}$, $(J, I) \models \Sigma$. This leads to a contradiction.

Assume $J = chase_{\Sigma_{st}}(I) \in Sol(\mathcal{M}, I)$ and there is another $J_1 \in Sol(\mathcal{M}, I)$. It holds that $J \subset J_1$ because Σ_{st} contains only full tgds. Let P be a target predicate and $\bar{a}$ is a tuple such that $P(\bar{a}) \in J_1 \setminus J$. Consider the target instance $J_2 = J \cup \{P(\bar{a})\}$. Since $J \subset J_2$, $(I, J_2) \models \Sigma_{st}$. Besides, since $J_2 \subset J_1$, $(J_2, I) \models \Sigma_{ts}$. Therefore $J_2 \in Sol(\mathcal{M}, I)$. As a matter of fact, to verify if J is the unique of $(\mathcal{M}, I)$, one can use the following PTIME algorithm:

– Verify if J is a solution.
– If J is a solution, for each target predicate P and each tuple $\bar{a}$, check if $J_2 = J \cup P(\bar{a})$ is in $Sol(\mathcal{M}, I)$. If there is no such $P(\bar{a}$ then conclude J is the unique solution.

$\square$

A similar situation happens if we require Σ_{ts} to contain only complete *LAV* tgds.

Theorem 4. *If Σ_{ts} contains only complete LAV dependencies then the unique-solution-checking problem is tractable .*

Proof. By restricting target to source dependencies to be complete LAV tgds, we also know what could be the candidate for the unique solution. Assume that we have T contains k target predicates $T_1, ..., T_k$. W.l.o.g we can assume that $\Sigma_{ts} = \{T_i(\bar{x}) \rightarrow \phi_{s_i}(\bar{x}, \bar{y}) | i = 1, k\}$ where $\phi_{s_i}(\bar{x}, \bar{y})$ is some conjunctive query over source schema $\mathbf{S}$. Let $\Sigma_{ts}^{-1} = \{T_i(\bar{x}) \leftarrow \phi_{s_i}(\bar{x}, \bar{y}) | i = 1, k\}$, then the following lemma holds.

Lemma 3. *Σ_{ts} contains only complete LAV tgds. If $Sol(\mathcal{M}, I) \neq \emptyset$ then $J = chase_{\Sigma_{ts}^{-1}}(I) \in Sol(\mathcal{M}, I)$.*

Proof. Since $J = chase_{\Sigma_{ts}^{-1}}(I) \in Sol(\mathcal{M}, I)$, $(I, J) \models \Sigma_{ts}$ and $T_i^J = \phi_{s_i}^I$ for $i = 1, k$. Assume that $J \notin Sol(\mathcal{M}, I)$, then $(I, J) \not\models \Sigma_{st}$. Let J' be a target instance such that $J' \in Sol(\mathcal{M}, I)$, $(I, J') \models \Sigma_{st}$ then for each $T_i \in \mathbf{T}$, $T_i^{J'} \subseteq \phi_{s_i}^I$. Consequently, $J' \subseteq J$. Besides, since $(I, J') \models \Sigma_{st}$, together with the fact $J' \subseteq J$, we can imply $(I, J) \models \Sigma_{st}$. This leads to a contradiction.

Based on Lemma 3, we have an analogous polynomial algorithm as in the proof of Theorem 3. $\square$

4 Guaranteeing Unique Solutions

In this section, we show results on complexity of deciding if a data exchange setting guarantees a unique solution for any source instance. The problem is formalised as follows.

Problem: $UniqueSolutionGuarantee()$
Input: a data exchange setting $\mathcal{M}$
Output: Does $\mathcal{M}$ guarantee unique solutions?

Without any restriction in the syntax of mapping rules, we show that the problem is undecidable as follows.

Theorem 5. *The problem of deciding if a data exchange setting guarantees unique solutions is undecidable.*

Proof. We prove the theorem by reducing the problem of checking conjunctive query containment under a set of tgds [6] to our problem.

Assume that we have a set of tgds Θ and two conjunctive queries $q_1(\bar{x})$ and $q_2(\bar{x})$ written over $\sigma(\Theta)$, Θ contains n dependencies $\theta_i : \phi_i(\bar{x}) \rightarrow \exists \bar{y}. \varphi_i(\bar{x}, \bar{y})$, $i = 1, n$. Take the data exchange setting $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ in which:

- $\mathbf{S} = \sigma(\Theta)$
- $\mathbf{T} = \{p(.)\} \cup \{T_i | i = 1, n\}$ such that each T_i is a new predicate which has the same arity as the arity of $\varphi_i(\bar{x}, \bar{y})$.
- $\Sigma_{st} = \{q_1(\bar{x}) \wedge q_2(\bar{x}) \rightarrow p(\bar{x})\} \cup \{\phi_i(\bar{x}) \rightarrow \exists \bar{y}. T_i(\bar{x}, \bar{y}) | i = 1, n\} \cup \{\varphi_i(\bar{x}, \bar{y}) \rightarrow T_i(\bar{x}, \bar{y}) | i = 1, n\}$.
- $\Sigma_{ts} = \{p(\bar{x}) \rightarrow q_1(\bar{x})\} \cup \{T_i(\bar{x}, \bar{y}) \rightarrow \varphi_i(\bar{x}, \bar{y}) | i = 1, n\}$.

We prove that $\Theta \models q_1 \subseteq q_2$ if and only if $\mathcal{M}$ guarantees unique solutions.

Assume that $\Theta \models q_1 \subseteq q_2$, then $\Theta \models \forall \bar{x}.q_1(\bar{x}) \leftrightarrow q_1(\bar{x}) \wedge q_2(\bar{x})$. Assume $\mathcal{M}$ does not guarantee unique solutions, i.e there is a source instance I, there are at least two different target instance J_1, J_2 such that both (I, J_1) and (I, J_2) satisfy Σ. Based on the construction of Σ, we have for each target predicate T_i, $T_i^{J_1} = T_i^{J_2} = \varphi_i^I$. Besides, since Σ implies Θ, $\Sigma \models q_1(\bar{x}) \leftrightarrow q_1(\bar{x}) \wedge q_2(\bar{x})$ as well. Therefore, $p^{J_1} = p^{J_2} = q_1^I = q_1^I \cap q_2^I$. This leads to a contradiction to the fact that J_1, J_2 are different target instances.

In case $\Theta \not\models q_1 \subseteq q_2$, then $\Theta \not\models \forall \bar{x}.q_1(\bar{x}) \rightarrow q_2(\bar{x})$. Therefore, there is a source instance I which is a model of Θ but $q_1^I \neq q_1^I \cap q_2^I$. Consider the two following different target instances J_1 and J_2 where $T_i^{J_1} = T_i^{J_1} = \varphi_i^I, p^{J_1} = q_1^I, p^{J_2} = q_1^I \cap q_2^I$. Based on the definition of Σ, J_1, J_2 are solutions of $\mathcal{M}$. This means $\mathcal{M}$ does not guarantee unique solutions.

Note that the above reduction also yields a lower bound for the complexity of the problem where the syntax of tgds is restricted and then it is decidable to check conjunctive query containment under a set of tgds. In order to introduce decidable algorithms for these cases, let us characterise necessary and sufficient conditions of a data exchange setting that guarantees unique solutions. This can be done by using the notion of Beth's definability [7].

Definition 3 (Beth's definability). *Let Σ be a set of sentences in FOL. A predicate P is implicitly definable from the set of predicates $\mathbf{P}$ under Σ if for every two interpretations $\mathcal{I} = \langle D^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ and $\mathcal{J} = \langle D^{\mathcal{J}}, \cdot^{\mathcal{J}} \rangle$ such that they are models of Σ, it holds that $\mathbf{P}^{\mathcal{I}} = \mathbf{P}^{\mathcal{J}}$ implies $P^{\mathcal{I}} = P^{\mathcal{J}}$.*

Based on the correspondence between the above definition and the definition of a data exchange that guarantees unique solutions, we have the following lemma.

Lemma 4. *A data exchange setting $\mathcal{M}$ guarantees unique solutions if and only if for any predicate $T \in \mathbf{T}$, T is implicitly definable from $\mathbf{S}$ under Σ.*

Since Beth's definability can be verified by a logical entailment, we also can reduce the problem of checking the guarantee to the problem of checking atomic query containment under a set of tgds as follows.

Theorem 6. *A data exchange setting $\mathcal{M}$ guarantees unique solutions if and only if for any predicate $T \in \mathbf{T}$, $\Sigma \cup \widetilde{\Sigma} \models \forall \bar{x}.T(\bar{x}) \leftrightarrow \widetilde{T}(\bar{x})$ where $\widetilde{\Sigma}$ is obtained from Σ by replacing every target predicate P_t with a new predicate with the same arity $\widetilde{P_t}$ and $\widetilde{T}$ is a new predicate having the same arity as T.*

Together with the results in deciding query containment under tgds mentioned in [6], the theorem and the reduction in the proof of Theorem 5 allow us to obtain tight bounds of *UniqueSolutionGuarantee*() for the following fragments of tgds.

Corollary 1. *Given data exchange setting $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$. Deciding if $\mathcal{M}$ guarantees unique solutions has the following complexity:*

1. 2EXPTIME-complete if Σ is weakly guarded.
2. 2EXPTIME-complete if Σ contains only guarded tgds.

Note that, by applying Beth's and Craig's theorems about definability in the case a data exchange setting guarantees unique solutions, one can actually constructively rewrite target predicates as views of source predicates. Consequently, given a source instance, a unique target instance can be materialised easily by using SQL to compute the views.

5 Conclusion and Outlook

We have considered the problem of checking if a data exchange setting under sts mappings has a unique solution and therefore it satisfies the purpose of exchanging data. We have studied two decision questions of the problem, one w.r.t a specific source instance and one w.r.t any source instance. While the former is decidable and can be solved using a Unique SAT solver, the latter is undecidable in general and is 2EXPTIME-complete in some fragments of tgds.

Besides the results and our on-going works in combined complexity analysis of the former, there are some issues that are deserved for further investigation. First, in [11] there is a syntactical class of tgds in which the existence-of-solution of peer data exchange is tractable. Besides, the class covers the case of full tgds and complete LAVs. Therefore, it is worth to study if the existence-of-unique-solution problem is also tractable for this class. Second, more tgd-based mapping languages should be considered for a complete complexity analysis such as disjunctive tgds [8] and its guarded fragments [5]. Last but not least, in the question about guarantee unique solution we have considered only the case that solution-existence implies unique -solution-existence. It is still unknown if the problem of checking a data exchange setting under sts mappings always admitting unique solutions is decidable or undecidable.

References

1. Foto N. Afrati and Phokion G. Kolaitis. Answering aggregate queries in data exchange. In *PODS*, PODS '08, pages 129–138. ACM, 2008.
2. Marcelo Arenas, Pablo Barceló, Ronald Fagin, and Leonid Libkin. Locally consistent transformations and query answering in data exchange. In *PODS*, PODS '04, pages 229–240, 2004.
3. Marcelo Arenas, Gabriel Diéguez, and Jorge Pérez. Bidirectional constraints for exchanging data: Beyond monotone queries. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 2698–2705, 2015.
4. Andreas Blass and Yuri Gurevich. On the unique satisfiability problem. *Information and Control*, 55(1-3):80–88, 1982.
5. Pierre Bourhis, Michael Morak, and Andreas Pieris. The impact of disjunction on query answering under guarded-based existential rules. In *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*, 2013.

6. Andrea Calì, Georg Gottlob, and Michael Kifer. Taming the infinite chase: Query answering under expressive relational constraints. *J. Artif. Intell. Res. (JAIR)*, 48:115–174, 2013.
7. William Craig. Three uses of the Herbrand-Gentzen theorem in relating model theory and proof theory. *J. Symb. Log.*, 22(3):269–285, 1957.
8. Alin Deutsch, Alan Nash, and Jeff Remmel. The chase revisited. In *PODS*, PODS '08, pages 149–158, New York, NY, USA, 2008. ACM.
9. Ronald Fagin. Inverting schema mappings. In *PODS*, PODS '06, pages 50–59, New York, NY, USA, 2006. ACM.
10. 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.
11. Ariel Fuxman, Phokion G. Kolaitis, Renee J. Miller, and Wang Chiew Tan. Peer data exchange. *ACM Trans. Database Syst.*, 31(4):1454–1498, 2006.
12. André Hernich. Answering non-monotonic queries in relational data exchange. In *ICDT*, pages 143–154, 2010.
13. Leonid Libkin. Data exchange and incomplete information. In *PODS*, PODS '06, pages 60–69, New York, NY, USA, 2006. ACM.
14. Leonid Libkin and Cristina Sirangelo. Open and closed world assumptions in data exchange. In *Proceedings of the 2009 Description Logics workshop*, 2009.
15. Nhung Ngo and Enrico Franconi. Unique solutions in data exchange. In *Database and Expert Systems Applications - 25th International Conference, DEXA 2014, Munich, Germany, September 1-4, 2014. Proceedings, Part II*, pages 281–294, 2014.