Reverse Data Exchange: Coping with Nulls

Ronald Fagin
IBM Almaden

Phokion G. Kolaitis
UC Santa Cruz & IBM Almaden

Lucian Popa
IBM Almaden

Wang-Chiew Tan
UC Santa Cruz

ABSTRACT

An inverse of a schema mapping $\mathcal{M}$ is intended to "undo" what $\mathcal{M}$ does, thus providing a way to perform "reverse" data exchange. In recent years, three different formalizations of this concept have been introduced and studied, namely, the notions of an inverse of a schema mapping, a quasi-inverse of a schema mapping, and a maximum recovery of a schema mapping. The study of these notions has been carried out in the context in which source instances are restricted to consist entirely of constants, while target instances may contain both constants and labeled nulls. This restriction on source instances is crucial for obtaining some of the main technical results about these three notions, but, at the same time, limits their usefulness, since reverse data exchange naturally leads to source instances that may contain both constants and labeled nulls.

We develop a new framework for reverse data exchange that supports source instances that may contain nulls, thus overcoming the semantic mismatch between source and target instances of the previous formalizations. The development of this new framework requires a careful reformulation of all the important notions, including the notions of the identity schema mapping, inverse, and maximum recovery. To this effect, we introduce the notions of extended identity schema mapping, extended inverse, and maximum extended recovery, by making systematic use of the homomorphism relation on instances. We give results concerning the existence of extended inverses and of maximum extended recoveries, and results concerning their applications to reverse data exchange and query answering. Moreover, we show that maximum extended recoveries can be used to capture in a quantitative way the amount of *information loss* embodied in a schema mapping specified by source-to-target tuple-generating dependencies.

Categories and Subject Descriptors

H.2.5 [**Database Management**]: Heterogeneous Databases – Data translation; H.2.4 [**Database Management**]: Systems – Relational Databases

General Terms

Algorithms, Theory

Keywords

Schema mapping, data exchange, data integration, model management, inverse, quasi-inverse, maximum recovery, chase

1. Introduction

Background and Motivation. Schema mappings are high-level specifications of how data from a source schema is to be transformed to data in a different schema, called the target schema. More formally, a schema mapping is a triple $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where $\mathbf{S}$ is a source schema, $\mathbf{T}$ is a target schema, and Σ is a set of database dependencies that specify the relationship between $\mathbf{S}$ and $\mathbf{T}$. In recent years, an extensive investigation of schema mappings and of their uses in data exchange and data integration has been carried out. One particular direction of this investigation has focused on the study of operators on schema mappings. Among all such operators originally introduced in [4], the composition operator and the inverse operator have been recognized as two fundamental ones. The intuition behind these two operators is as follows. Given two schema mappings $\mathcal{M}_{12}$ and $\mathcal{M}_{23}$ such that the target schema of $\mathcal{M}_{12}$ is the same as the source schema of $\mathcal{M}_{23}$, the composition operator yields a schema mapping $\mathcal{M}_{13}$ that is "equivalent" to the successive application of $\mathcal{M}_{12}$ and $\mathcal{M}_{23}$, thus providing a way to perform data exchange directly between the source schema of $\mathcal{M}_{12}$ and the target schema of $\mathcal{M}_{23}$. Given a schema mapping $\mathcal{M}$, the inverse operator yields a schema mapping $\mathcal{M}'$ that "undoes" what $\mathcal{M}$ did, thus providing a way to do "reverse" data exchange. Clearly, these two operators are of interest in their own right; furthermore, when combined together, they attain even greater power since, in combination, they can be used to analyze schema evolution.

By now, the composition operator has been investigated in depth, and consensus has been achieved on what the definitive semantics for composition is [5, 9, 12, 14]. The state of affairs concerning the inverse operator, however, is more complicated and by far less definitive. In [7], rigorous semantics for the inverse operator was given for the first time. The notion of inverse introduced in [7] turned out to be rather restrictive, as most schema mappings specified by source-to-target tuple generating dependencies (s-t tgds) are not invertible. For this reason, the notion of a *quasi-inverse* of a schema mapping was introduced and studied in [10]. After this, a competing notion of a *maximum recovery* of a schema mapping was introduced and studied in [2]. For invertible schema mappings, the notions of inverse, quasi-inverse, and maximum recovery coincide. In contrast, for non-invertible schema mappings, the notions of quasi-inverse and maximum recovery differ in general. Moreover, every schema mapping specified by a set of s-t tgds has a maximum recovery, whereas there are such schema mappings that are not quasi-invertible.

Their differences notwithstanding, all previous studies of "inverse" operators (including inverse, quasi-inverse, and maximum recovery) have the following basic assumption in common: the source instances are *ground*, i.e., they consist entirely of constants, while, on the contrary, target instances may contain both constants

and labeled nulls (variables). In particular, some of the key technical results in [2, 7, 10] very much depend on the assumption that source instances do not contain labeled nulls. However, applications of "inverse" operators naturally lead to source instances that may contain labeled nulls, in addition to constants. The following example illustrates this scenario.

EXAMPLE 1.1. Let $\mathcal{M}$ be the schema mapping given by the tuple-generating dependency

$$P(x,y,z) \rightarrow Q(x,y) \wedge R(y,z),$$

which describes a decomposition of a source relation P into two target relations Q and R. It was shown in [10] that $\mathcal{M}$ is not invertible but is quasi-invertible. Furthermore, a natural "inverse" of $\mathcal{M}$, which is both a quasi-inverse of $\mathcal{M}$ and a maximum recovery for $\mathcal{M}$, is the schema mapping $\mathcal{M}'$ given by the following set of "reverse" tgds:

$$\Sigma' = \{\ Q(x,y) \rightarrow \exists z P(x,y,z),\ R(y,z) \rightarrow \exists x P(x,y,z)\ \}.$$

Consider the ground source instance $I = \{P(a,b,c)\}$, where a, b, c are constants. The result of chasing I with $\mathcal{M}$ (i.e., the result of performing data exchange with $\mathcal{M}$) is the instance $U = \{Q(a,b), R(b,c)\}$. If we now chase U with $\mathcal{M}'$ (i.e., perform the "reverse" data exchange with $\mathcal{M}'$), we obtain the source instance $V = \{P(a,b,Z), P(X,b,c)\}$, where Z and X are nulls. Note that V, which is the canonical result of "reverse" data exchange, is no longer a ground instance and, thus, it is ruled out from the semantics. □

Another limitation of restricting source instances to be ground is that data exchange cannot be performed on source instances that are the result of a prior data exchange with s-t tgds, since, in general, labeled nulls may be generated by chasing source instances with tgds to produce universal solutions [8]. Thus, the preceding considerations suggest that previous studies of inverse, quasi-inverse, and maximum recovery suffer from a *semantic mismatch* that stems from the assumption that only ground instances are allowed.

Summary of Contributions. Our goal in this paper is to develop a new framework for "reverse" data exchange that overcomes this semantic mismatch. This new framework supports source instances with nulls, and makes it possible to recover source instances using reverse data exchange and to permit target instances that result from one data exchange to be used as source instances of another data exchange (thus, in the long run, this framework will enable the analysis of schema evolution using composition and inverse).

The development of this new framework requires a careful reformulation of all the important notions, including the notions of the identity schema mapping (which was used in the definition of inverse [7]), inverse, and maximum recovery. This is so because these earlier notions, which were studied under the assumption that only ground source instances are allowed, lose their desirable properties when schema mappings are used to perform data exchange between source and target instances that may both contain nulls. For example, using the concept of maximum recovery in [2], it can be shown (see Proposition 4.2) that the schema mapping specified by the tgd $P(x,y) \rightarrow \exists z(Q(x,z) \wedge Q(z,y))$ has a maximum recovery when only ground source instances are allowed, but has no maximum recovery when source instances may contain nulls.

The key to finding the "right" notions in the new framework is to replace the *containment* relation $I_1 \subseteq I_2$ between instances by the *homomorphism* relation $I_1 \rightarrow I_2$, which holds if there is a homomorphism from I_1 to I_2 that maps constants to themselves (note that if I_1 is a ground instance, then $I_1 \rightarrow I_2$ if and only if $I_1 \subseteq I_2$). We use the notation $\rightarrow$ to denote the binary relation $\{(I_1, I_2) : I_1 \rightarrow I_2\}$ between instances that may contain nulls. The next step is to replace the identity schema mapping $\mathrm{Id} = \{(I_1, I_2) : I_1, I_2$ are ground instances such that $I_1 \subseteq I_2\}$ with the *extended identity* schema mapping $e(\mathrm{Id}) = \rightarrow$. Thus,

$$e(\mathrm{Id}) = \{(I_1, I_2) : I_1, I_2 \text{ are instances such that } I_1 \rightarrow I_2\}.$$

In fact, even the notion of a *solution* needs to be reformulated in the new framework. Specifically, we say that a target instance J is an *extended solution* for a source instance I w.r.t. a schema mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ if there are a source instance I' and a target instance J' such that $I \rightarrow I'$, $(I', J') \models \Sigma$, and $J' \rightarrow J$. In effect, an extended solution J for I w.r.t. $\mathcal{M}$ is a solution for I w.r.t. the schema mapping $e(\mathcal{M}) = \rightarrow \circ \mathcal{M} \circ \rightarrow$, which we call the *homomorphic extension of $\mathcal{M}$*. The intuition behind extended solutions is that, since labeled nulls represent unknown information, it is "legal" to homomorphically map them to other values before or after taking the standard notion of solution. In a sense, the notion of an extended solution is a relaxation of the notion of solution where the exact values of the labeled nulls (in both the source instance and the target instance) do not matter. Further, our definition of extended solutions relaxes the notion of solution even more by allowing the possibility that additional facts may be present (in the spirit of the "open-world assumption").[1] With these new notions at hand, we can then define the notions of *extended inverse*, *extended recovery*, and *maximum extended recovery* by systematically replacing the identity schema mapping Id by the extended identity schema mapping $e(\mathrm{Id})$, and the composition $\mathcal{M} \circ \mathcal{M}'$ by the composition $e(\mathcal{M}) \circ e(\mathcal{M}')$ of the homomorphic extensions of $\mathcal{M}$ and $\mathcal{M}'$. Specifically, we say that $\mathcal{M}'$ is an *extended inverse* of $\mathcal{M}$ if $e(\mathcal{M}) \circ e(\mathcal{M}') = e(\mathrm{Id})$. We say that $\mathcal{M}'$ is a *recovery* of $\mathcal{M}$ if for every source instance I, we have that $(I, I) \in e(\mathcal{M}) \circ e(\mathcal{M}')$; finally, we say that $\mathcal{M}'$ is a *maximum extended recovery* of $\mathcal{M}$ if $\mathcal{M}'$ is an extended recovery of $\mathcal{M}$ and for every extended recovery $\mathcal{M}''$ of $\mathcal{M}$ we have that $e(\mathcal{M}) \circ e(\mathcal{M}') \subseteq e(\mathcal{M}) \circ e(\mathcal{M}'')$.

The first group of our technical results concerns the properties of the extended inverse and its relation to the inverse. We give a necessary and sufficient condition for an arbitrary schema mapping to have an extended inverse and then focus on schema mappings specified by s-t tgds. We show that if $\mathcal{M}$ is specified by s-t tgds and is extended invertible, then it is invertible, but not vice-versa. An extended inverse, however, need not be an inverse (and, of course, an inverse need not be an extended inverse, since invertibility does not imply extended invertibility). We also show that if $\mathcal{M}$ and $\mathcal{M}'$ are schema mappings such that both are specified by tgds, then $\mathcal{M}'$ is an extended inverse of $\mathcal{M}$ if and only if $\mathcal{M}'$ is a *chase-inverse* of $\mathcal{M}$, that is, if every source instance I is homomorphically equivalent to $chase_{\mathcal{M}'}(chase_{\mathcal{M}}(I))$, which is the source instance obtained from I by first chasing with $\mathcal{M}$ and then with $\mathcal{M}'$. This result reveals that extended inverses specified by tgds are ideally suited for "reverse" data exchange, since if the original source instance I is no longer available, we can recover a homomorphically equivalent one by chasing a universal solution for I with an extended inverse specified by tgds. It should be noted that this result does not hold for (non-extended) inverses.

After this, we focus on maximum extended recoveries and obtain the main technical results of this paper. We show that every schema mapping $\mathcal{M}$ specified by s-t tgds has a maximum extended recovery $\mathcal{M}'$. Note that if both $\mathcal{M}'$ and $\mathcal{M}''$ are maximum extended

---

[1]This is automatic in the case of schema mappings specified by dependencies, such as s-t tgds.

recoveries of $\mathcal{M}$, then

$$e(\mathcal{M}) \circ e(\mathcal{M}') \;=\; e(\mathcal{M}) \circ e(\mathcal{M}'').$$

For schema mappings $\mathcal{M}$ specified by s-t tgds, we characterize the quantity $e(\mathcal{M}) \circ e(\mathcal{M}')$ in terms of $\mathcal{M}$ alone by showing that $e(\mathcal{M}) \circ e(\mathcal{M}') \;=\; \rightarrow_{\mathcal{M}}$, where

$$\rightarrow_{\mathcal{M}} \;=\; \{(I_1, I_2) : chase_{\mathcal{M}}(I_1) \rightarrow chase_{\mathcal{M}}(I_2)\}.$$

This result makes it possible to capture in a quantitative way the *information loss* embodied in a schema mapping. Specifically, we can define the information loss of a schema mapping $\mathcal{M}$ specified by s-t tgds as the set-theoretic difference $\rightarrow_{\mathcal{M}} \setminus e(\mathrm{Id})$; note also that $e(\mathrm{Id}) \subseteq \rightarrow_{\mathcal{M}}$. In effect, the information loss of a schema mapping $\mathcal{M}$ specified by s-t tgds measures the amount by which $\mathcal{M}$ deviates from being extended invertible, since we show that $\mathcal{M}$ is extended invertible if and only if $\rightarrow_{\mathcal{M}} \;=\; e(\mathrm{Id})$. The concept of information loss can also be used to *compare* two schema mappings in a quantitative way and determine which of the two is "more invertible".

We also embark on an investigation of the language needed to express maximum extended recoveries of schema mappings specified by s-t tgds. To this effect, we show that if $\mathcal{M}$ is a schema mapping specified by full s-t tgds, then it has a maximum extended recovery specified by disjunctive tgds with inequalities; this turns out to be an optimal result, since we show the necessity of both disjunctions and inequalities. We leave it as an open problem, however, to pinpoint the exact language needed to express maximum extended recoveries of schema mappings specified by arbitrary s-t tgds.

Finally, we show that maximum extended recoveries specified by disjunctive tgds can be used to perform "reverse" data exchange and "reverse" query answering, where the latter means answering queries over the *source* schema when the source instance is no longer available. A key notion that is central to all these applications (reverse data exchange, reverse query answering, and also comparing schema mappings) is the notion of a *universal-faithful* schema mapping, which provides a procedural counterpart to the notion of maximum extended recovery of a schema mapping.

2. Background, Basic Notions, and Notation

A *schema* $\mathbf{R}$ is a finite sequence $\langle R_1, \ldots, R_k \rangle$ of relation symbols, each of a fixed arity. An *instance* I over $\mathbf{R}$ is a sequence $(R_1^I, \ldots, R_k^I)$, where each R_i^I is a finite relation of the same arity as R_i. We shall often use R_i to denote both the relation symbol and the relation R_i^I that interprets it.

We assume that we have a fixed infinite set $\underline{\mathrm{Const}}$ of *constants* and an infinite set $\underline{\mathrm{Var}}$ of *labeled nulls* that is disjoint from $\underline{\mathrm{Const}}$. A *ground* instance over some schema is an instance such that all values occurring in its relations are constants. We will also assume that $\mathbf{S}$ is a fixed *source* schema and $\mathbf{T}$ is a fixed *target* schema. Starting with [8], earlier work on data exchange was carried out under the assumption that instances over the source schema $\mathbf{S}$ are ground, while instances over the target schema $\mathbf{T}$ may contain both constants and labeled nulls. This models the situation in which we perform data exchange from $\mathbf{S}$ to $\mathbf{T}$ under the assumption that the individual values of source instances are known, while the under-specification of a data exchange setting may give rise to null values in the target instances. In this paper, we will drop the assumption that source instances are ground and, instead, we will assume that instances over both the source and the target schema may have individual values from $\underline{\mathrm{Const}} \cup \underline{\mathrm{Var}}$. Allowing source instances to contain both constants and labeled nulls models the situation where

source instances may also contain incomplete information. In particular, under this assumption, the target instance of one data exchange can be used as source instances of another data exchange.

Schema mappings A *schema mapping* is a triple $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where Σ is a set of constraints (typically, formulas in some logic) that describe the relationship between $\mathbf{S}$ and $\mathbf{T}$. We say that Σ *specifies* $\mathcal{M}$. This is the syntactic view of schema mappings. For all practical purposes, however, a schema mapping can be identified with the binary relation

$$\{(I, J) : I \text{ is a } \mathbf{S}\text{-instance}, J \text{ is a } \mathbf{T}\text{-instance}, (I, J) \models \Sigma\}.$$

This is the semantic view of schema mappings. In what follows, we will switch freely between the syntactic view and the semantic view of schema mappings. In particular, we will use the notation $(I, J) \in \mathcal{M}$ to denote that the ordered pair (I, J) satisfies the constraints of $\mathcal{M}$; furthermore, we will sometimes define schema mappings by simply defining the set of ordered pairs (I, J) that constitute $\mathcal{M}$ (instead of giving a set of constraints that specify $\mathcal{M}$). We will also say that J is a *solution for I* w.r.t. $\mathcal{M}$ if $(I, J) \in \mathcal{M}$. We use $sol_{\mathcal{M}}(I)$ to denote the set of all solutions of I w.r.t. $\mathcal{M}$.

Dependencies and schema mappings A *source-to-target tuple-generating dependency*, in short, an *s-t tgd*, is a first-order formula of the form $\forall \mathbf{x}(\varphi(\mathbf{x}) \rightarrow \exists \mathbf{y} \psi(\mathbf{x}, \mathbf{y}))$, where $\varphi(\mathbf{x})$ is a conjunction of atomic formulas over $\mathbf{S}$, $\psi(\mathbf{x}, \mathbf{y})$ is a conjunction of atomic formulas over $\mathbf{T}$, and every variable in $\mathbf{x}$ occurs in an atomic formula in $\varphi(\mathbf{x})$. A *full* s-t tgd is a tgd with no existential quantifiers $\exists \mathbf{y}$. In this paper, we will focus on *schema mappings* $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ with Σ a set of s-t tgds. Our goal is to study *extended inverses* and *maximum extended recoveries* of such schema mappings. In particular, we will also investigate the language needed to express maximum extended recoveries. This will require bringing into the picture a richer class of dependencies that was first considered in the study of quasi-inverses of schema mappings [10] and then in the study of maximum recoveries [2].

Let *Constant* be a relation symbol that is different from all relation symbols in $\mathbf{S}$ and $\mathbf{T}$. A *disjunctive tgd with constants and inequalities from $\mathbf{T}$ to $\mathbf{S}$* is a first-order formula of the form

$$\forall \mathbf{x}(\varphi(\mathbf{x}) \rightarrow \bigvee_{i=1}^{n} \exists \mathbf{y}_i \psi_i(\mathbf{x}, \mathbf{y}_i)), \text{ where:}$$

- The formula $\varphi(\mathbf{x})$ is a conjunction of

  (1) atoms over $\mathbf{T}$, such that every variable in $\mathbf{x}$ occurs in at least one of them;

  (2) formulas of the form *Constant*(x) with x a variable in $\mathbf{x}$;

  (3) inequalities $x \neq x'$ with x and x' variables in $\mathbf{x}$.

- Each formula $\psi_i(\mathbf{x}, \mathbf{y}_i)$ is a conjunction of atoms over $\mathbf{S}$.

Naturally, a formula *Constant*(x) evaluates to true if and only if x is interpreted by a value in $\underline{\mathrm{Const}}$.

In this paper, we also make use of *disjunctive tgds with inequalities*, which are obtained by not allowing condition (2) in the above definition. Moreover, if neither condition (2) nor (3) is allowed, then we refer to these formulas simply as *disjunctive tgds*. We shall also make use of *tgds with constants*, which are obtained by disallowing both disjunction and condition (3).

Composing and inverting schema mappings Next, we recall the concept of the *composition* of two schema mappings, introduced in [9, 13], and the concept of an *inverse* of a schema mapping, introduced in [7].

If $\mathcal{M}_{12} = (\mathbf{S}_1, \mathbf{S}_2, \Sigma_{12})$ and $\mathcal{M}_{23} = (\mathbf{S}_2, \mathbf{S}_3, \Sigma_{23})$ are two schema mappings, their *composition* $\mathcal{M}_{12} \circ \mathcal{M}_{23}$ is a schema mapping $(\mathbf{S}_1, \mathbf{S}_3, \Sigma_{13})$ such that for every $\mathbf{S}_1$-instance I and every $\mathbf{S}_3$-instance K, we have that $(I, K) \models \Sigma_{13}$ if and only if there is a $\mathbf{S}_2$-instance J such that $(I, J) \models \Sigma_{12}$ and $(J, K) \models \Sigma_{23}$. When

the schemas involved are understood from the context, we will often write $\Sigma_{12} \circ \Sigma_{23}$ to denote the composition $\mathcal{M}_{12} \circ \mathcal{M}_{23}$.

The study of inverses of schema mappings in [7] assumes that source instances are ground. Let $\hat{\mathbf{S}}$ be a replica of the source schema $\mathbf{S}$, i.e., for every relation symbol R of $\mathbf{S}$, the schema $\hat{\mathbf{S}}$ contains a relation symbol $\hat{R}$ that is not in $\mathbf{S}$ and has the same arity as R. Thus, every source instance I has a replica instance $\hat{I}$ over $\hat{\mathbf{S}}$.

The *identity schema mapping* is $\mathrm{Id} = (\mathbf{S}, \hat{\mathbf{S}}, \Sigma_{\mathrm{Id}})$, where Σ_{Id} consists of the dependencies $R(\mathbf{x}) \to \hat{R}(\mathbf{x})$ as R ranges over the relation symbols in $\mathbf{S}$. Thus, from the semantic point of view, Id is the set of all pairs (I_1, I_2) such that I_1 is a ground $\mathbf{S}$-instance, I_2 is a ground $\hat{\mathbf{S}}$-instance, and $\hat{I}_1 \subseteq I_2$.

Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ be a schema mapping. We say that a schema mapping $\mathcal{M}' = (\mathbf{T}, \hat{\mathbf{S}}, \Sigma')$ is an *inverse* of $\mathcal{M}$ if $\mathcal{M} \circ \mathcal{M}' = \mathrm{Id}$ (as sets of pairs of instances). This means that, for every pair (I_1, I_2) of a ground $\mathbf{S}$-instance I_1 and a ground $\hat{\mathbf{S}}$-instance I_2, we have that $\hat{I}_1 \subseteq I_2$ if and only if there is a target instance J such that $(I_1, J) \models \Sigma$ and $(J, I_2) \models \Sigma'$.

From now on and for notational simplicity, we will write $\mathbf{S}$ to also denote its replica $\hat{\mathbf{S}}$; it will be clear from the context if we refer to $\mathbf{S}$ or to its replica. Moreover, we will use the same symbol to denote both a ground $\mathbf{S}$-instance I and its replica $\hat{\mathbf{S}}$-instance $\hat{I}$.

3. Extended Inverses

In this section, we introduce and study the notion of an extended inverse of a schema mapping. For this, we first introduce the notions of an extended solution and of the extended identity mapping.

We begin by giving the standard definitions of homomorphisms and homomorphic equivalence.

DEFINITION 3.1. Let I_1 and I_2 be instances over a schema $\mathbf{R}$, with values in $\underline{\mathrm{Const}} \cup \underline{\mathrm{Var}}$. A function h from $\underline{\mathrm{Const}} \cup \underline{\mathrm{Var}}$ to $\underline{\mathrm{Const}} \cup \underline{\mathrm{Var}}$ is a *homomorphism* from I_1 to I_2 if for every c in $\underline{\mathrm{Const}}$, we have that $h(c) = c$, and for every relation symbol R in $\mathbf{R}$ and every tuple $(a_1, \ldots, a_n) \in R^{I_1}$, we have that $(h(a_1), \ldots, h(a_n)) \in R^{I_2}$.

We use the notation $I_1 \to I_2$ to denote that there is a homomorphism from I_1 to I_2. We say that I_1 is *homomorphically equivalent* to I_2 if $I_1 \to I_2$ and $I_2 \to I_1$. We shall also use the notation $\to$ for the binary relation $\{(I_1, I_2) \mid I_1 \to I_2\}$. □

Note that $\to$ is itself a schema mapping in which the source schema is the same as the target schema. An important property of $\to$ that we shall often use is that $\to \circ \to \ = \ \to$.

DEFINITION 3.2. Let $\mathcal{M}$ be a schema mapping. We say that J is an *extended solution* of I w.r.t. $\mathcal{M}$ if there exist instances I' and J' such $I \to I'$, $(I', J') \in \mathcal{M}$, and $J' \to J$. This is the same as saying that $(I, J) \in \ \to \circ \mathcal{M} \circ \to$. We use $\mathrm{eSol}_{\mathcal{M}}(I)$ to denote the set of all extended solutions of I w.r.t. $\mathcal{M}$. □

EXAMPLE 3.3. Recall the earlier Example 1.1. The target instance U is not a solution for the source instance V w.r.t. schema mapping $\mathcal{M}$ because every solution for V w.r.t. $\mathcal{M}$ must contain $R(b, Z)$ and $Q(X, b)$. However, U is an extended solution for V w.r.t. $\mathcal{M}$. To see this, consider the target instance

$$U' = \{Q(a, b), Q(X, b), R(b, c), R(b, Z)\},$$

which is a solution for V w.r.t. $\mathcal{M}$. Furthermore, there is a homomorphism from U' to U (where X is mapped to a, and Z is mapped to c). Thus, $(V, U') \in \mathcal{M}$, and $U' \to U$. Hence, U is an extended solution for V w.r.t. $\mathcal{M}$.

Another way to see that U is an extended solution for V w.r.t. $\mathcal{M}$ is to observe that if I is as in Example 1.1, then $V \to I$, and U itself is a solution for I w.r.t. $\mathcal{M}$.

As this example illustrates, the main idea behind extended solutions is that, since nulls represent unknown information, it is "legal" to homomorphically map them to other values either before or after taking the standard notion of solution. □

The following proposition shows that extended solutions coincide with solutions in an important special case.

PROPOSITION 3.4. *If I is a ground source instance and $\mathcal{M}$ is a schema mapping specified by s-t tgds, then $\mathrm{eSol}_{\mathcal{M}}(I) = \mathrm{Sol}_{\mathcal{M}}(I)$.*

Based on extended solutions, we define extended universal solutions by mimicking the definition of universal solutions in [8].

DEFINITION 3.5. Let $\mathcal{M}$ be a schema mapping. We say that J is an *extended universal solution* for I w.r.t. $\mathcal{M}$ if $J \in \mathrm{eSol}_{\mathcal{M}}(I)$ and, for every $J' \in \mathrm{eSol}_{\mathcal{M}}(I)$, we have that $J \to J'$. □

We now define the notion of a homomorphic extension of a schema mapping. This plays a central role in what follows.

DEFINITION 3.6. Let $\mathcal{M}$ be a schema mapping. The *homomorphic extension of $\mathcal{M}$*, denoted by $e(\mathcal{M})$, is the schema mapping $\to \circ \mathcal{M} \circ \to$. □

Note that for every source instance I, the extended solutions for I w.r.t. $\mathcal{M}$ are exactly the (standard) solutions of I w.r.t. $e(\mathcal{M})$.

In the same spirit as the extended notion of solution, we now consider an extended notion of the identity schema mapping. This is obtained by applying the homomorphic extension operator e on the standard identity schema mapping Id.

DEFINITION 3.7. The *extended identity schema mapping* is the schema mapping $e(\mathrm{Id})$. □

Note that, by definition, $e(\mathrm{Id}) = \ \to \circ \mathrm{Id} \circ \to$. It is easy to see that $\to \circ \mathrm{Id} \circ \to$ is the same as $\to$ and, therefore, $e(\mathrm{Id}) = \ \to$. Thus, the key difference from the standard notion of the identity schema mapping is that $e(\mathrm{Id})$ considers pairs (I_1, I_2) of instances such that I_1 is not necessarily a subset of I_2 but, instead, I_1 can be homomorphically mapped into I_2. Intuitively, I_1 is a subset of I_2 up to homomorphic mapping of nulls. Note that when I_1 and I_2 are ground, we have that $I_1 \to I_2$ if and only if $I_1 \subseteq I_2$. Thus, for ground instances, $e(\mathrm{Id})$ coincides with Id.

We are now ready to give the definition of an extended inverse of a schema mapping.

DEFINITION 3.8. Let $\mathcal{M}$ be a schema mapping.

- A schema mapping $\mathcal{M}'$ is an *extended inverse* of $\mathcal{M}$ if

$$e(\mathcal{M}) \circ e(\mathcal{M}') \ = \ e(\mathrm{Id}).$$

  Since $\to \circ \to \ = \ \to$, the above equation is the same as

$$\to \circ \mathcal{M} \circ \to \circ \mathcal{M}' \to \ = \ \to .$$

- $\mathcal{M}$ is *extended-invertible* if it has an extended inverse. □

We next introduce the notion of a *capturing function*, which will be used to characterize extended invertibility.

DEFINITION 3.9. Let $\mathcal{M}$ be a schema mapping.

- We say that a target instance J *captures* a source instance I (for $\mathcal{M}$) if the following two conditions hold: (a) $J \in \mathrm{eSol}_{\mathcal{M}}(I)$; and (b) if K is a source instance such that $J \in \mathrm{eSol}_{\mathcal{M}}(K)$, then $K \to I$.

- A *capturing function* for $\mathcal{M}$ is a (total) function F from source instances to target instances such that for every source instance I, we have that $F(I)$ captures I. □

The definitions imply that if J captures both I_1 and I_2, then I_1 and I_2 are homomorphically equivalent. Thus, the source instance that is captured by J is unique up to homomorphic equivalence. In general, for a given I there may not exist a J that captures I. If such J exists for every I, then a capturing function for $\mathcal{M}$ exists.

We note that the notion of a target instance capturing a source instance is an extended version of the notion of a strong witness solution in [2]. The next theorem shows that extended invertibility is equivalent to the existence of a capturing function.

THEOREM 3.10. *Let $\mathcal{M}$ be a schema mapping. The following statements are equivalent:*

(1) $\mathcal{M}$ is extended-invertible.

(2) There exists a capturing function for $\mathcal{M}$.

Furthermore, if $\mathcal{M}$ is extended-invertible and F is a capturing function for $\mathcal{M}$, then $\mathcal{M}' = \{(J, I) \mid J = F(I)\}$ is an extended inverse of $\mathcal{M}$.

3.1 Extended Inverses of Mappings Specified by s-t tgds

We now address the important case in which the schema mapping $\mathcal{M}$ is specified by a finite set of s-t tgds. First we relate extended-invertibility of such schema mappings to the existence of a special capturing function given by the chase, and also to a homomorphism-based property, which we define shortly. We consider here the standard chase with tgds as introduced in [3] (see also [1]) but applied to data exchange [8]. In particular, given a source instance I, we write $chase_{\mathcal{M}}(I)$ to denote a target instance J such that (I, J) is the result of chasing $(I, \emptyset)$ with the dependencies in $\mathcal{M}$. Note here that $(I, \emptyset)$ and (I, J) are instances over the combined source and target schema.

The following proposition is analogous to (and follows easily from) the result in [8] that $chase_{\mathcal{M}}(I)$ is a universal solution for I.

PROPOSITION 3.11. *Let $\mathcal{M}$ be a schema mapping specified by a finite set of s-t tgds. If J is a universal solution for I w.r.t. $\mathcal{M}$, then J is an extended universal solution for I w.r.t. $\mathcal{M}$. In particular, $chase_{\mathcal{M}}(I)$ is an extended universal solution for I.*

DEFINITION 3.12. *Assume that $\mathcal{M}$ is a schema mapping specified by a finite set of s-t tgds. We say that $\mathcal{M}$ has the homomorphism property if for all source instances I_1 and I_2, the following holds: if $chase_{\mathcal{M}}(I_1) \to chase_{\mathcal{M}}(I_2)$, then $I_1 \to I_2$.*

THEOREM 3.13. *Let $\mathcal{M}$ be a schema mapping specified by a finite set of s-t tgds. The following statements are equivalent:*

(1) $\mathcal{M}$ is extended-invertible.

(2) There exists a capturing function for $\mathcal{M}$.

(3) The function F with $F(I) = chase_{\mathcal{M}}(I)$ is a capturing function for $\mathcal{M}$.

(4) $\mathcal{M}$ has the homomorphism property.

Moreover, if $\mathcal{M}$ is extended-invertible, then the schema mapping $\mathcal{M}^* = \{(J, I) \mid J = chase_{\mathcal{M}}(I)\}$ is an extended inverse of $\mathcal{M}$.

Theorems 3.10 and 3.13 provide useful tools for verifying whether a schema mapping is extended invertible or not. Next, we give an example that illustrates how Theorem 3.13 can be applied to show that a schema mapping given by s-t tgds is *not* extended invertible.

EXAMPLE 3.14. Consider the "union" schema mapping $\mathcal{M}$ specified by the s-t tgds $P(x) \to R(x)$ and $Q(x) \to R(x)$. We

now prove that $\mathcal{M}$ is not extended-invertible by showing that $\mathcal{M}$ does not have the homomorphism property. Let $I_1 = \{P(0)\}$ and $I_2 = \{Q(0)\}$. It is obvious that $chase_{\mathcal{M}}(I_1) \to chase_{\mathcal{M}}(I_2)$; however, it is not true that $I_1 \to I_2$. $\square$

We next relate the notions of extended invertibility and extended inverses to the notions of invertibility and inverses.

THEOREM 3.15. *The following hold:*

(1) If $\mathcal{M}$ is a schema mapping specified by a finite set of s-t tgds and $\mathcal{M}$ is extended invertible, then $\mathcal{M}$ is invertible.

(2) There is a schema mapping $\mathcal{M}$ specified by a finite set of s-t tgds that is invertible but not extended-invertible.

(3) There is a schema mapping $\mathcal{M}$ specified by a finite set of s-t tgds that is extended-invertible and such that:

    (a) $\mathcal{M}$ has an extended inverse $\mathcal{M}'$ that is not an inverse of $\mathcal{M}$.

    (b) $\mathcal{M}$ has an inverse $\mathcal{M}''$ that is not an extended inverse of $\mathcal{M}$.

PROOF. Part (1) follows from the fact that the homomorphism property (which is equivalent to extended invertibility) implies the "subset property" [10] (which characterizes invertibility).

For part (2), consider the schema mapping $\mathcal{M}$ that is specified by the following s-t tgds:

$$P(x) \to \exists y R(x, y) \qquad Q(y) \to \exists x R(x, y)$$

We now show that $\mathcal{M}$ is invertible but not extended-invertible. First, it can be easily verified that the schema mapping $\mathcal{M}'$ given by the following s-t tgds with constants is an inverse of $\mathcal{M}$:

$$R(x, y) \wedge Constant(x) \to P(x)$$
$$R(x, y) \wedge Constant(y) \to Q(y)$$

Thus, $\mathcal{M}$ is invertible. We now show that $\mathcal{M}$ is not extended-invertible by showing it fails to satisfy the homomorphism property. Consider the source instances $I_1 = \{P(n_1)\}$ and $I_2 = \{Q(n_2)\}$ where n_1 and n_2 are nulls. Then $chase_{\mathcal{M}}(I_1)$ and $chase_{\mathcal{M}}(I_2)$ are homomorphically equivalent. In particular, $chase_{\mathcal{M}}(I_1) \to chase_{\mathcal{M}}(I_2)$. However, it is not the case that $I_1 \to I_2$.

For part (3), consider the schema mapping $\mathcal{M}$ given by the following s-t tgd:

$$P(x, y) \to \exists z (Q(x, z) \wedge Q(z, y))$$

Moreover, consider the following "reverse" schema mappings $\mathcal{M}'$ and $\mathcal{M}''$ given, respectively, by the following dependencies:

$$\mathcal{M}': \qquad Q(x, z) \wedge Q(z, y) \to P(x, y)$$
$$\mathcal{M}'': \quad Q(x, z) \wedge Q(z, y) \wedge Constant(x) \wedge Constant(y)$$
$$\to P(x, y)$$

We shall show in the next subsection (Example 3.18) that $\mathcal{M}'$ is an extended inverse of $\mathcal{M}$. At the same time, it was shown in [10] that there is no inverse of $\mathcal{M}$ that is specified by s-t tgds without the *Constant* predicate. Hence, $\mathcal{M}'$ cannot be an inverse of $\mathcal{M}$. Thus, $\mathcal{M}'$ is an extended inverse of $\mathcal{M}$ that is not an inverse of $\mathcal{M}$. As for condition (b), it was shown in [10] that $\mathcal{M}''$ is an inverse of $\mathcal{M}$. We show in the next subsection (Example 3.19) that $\mathcal{M}''$ is not an extended inverse of $\mathcal{M}$. $\square$

Theorem 3.15 tells us that the notion of extended invertibility is stronger than invertibility. Intuitively, it is harder for a schema mapping to be extended invertible, since there are more instances (i.e., non-ground source instances) to consider. Furthermore, extended inverses and inverses do not necessarily coincide, even for schema mappings that are extended-invertible (hence, also invertible).

3.2 The Goodness of Extended Inverses

In this section, we show that extended inverses that are given by s-t tgds have desirable properties that make them ideally suited for "reverse" data exchange. The data exchange problem associated with a schema mapping $\mathcal{M}$ is to materialize a "good" solution J from a source instance I, based on $\mathcal{M}$. The canonical procedure for data exchange [8] uses the chase of I with $\mathcal{M}$ to produce a (canonical) universal solution. The *reverse data exchange problem* is then to materialize a source instance I' from a target instance J according to a reverse schema mapping $\mathcal{M}'$ (from the target schema to the source schema). Reverse data exchange is typically performed after an initial data exchange with $\mathcal{M}$ was performed; in such a case, the goal of reverse data exchange is to recover a source instance that is as "close as possible" to the *original* source instance I.

DEFINITION 3.16. Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ be a schema mapping specified by a finite set of s-t tgds. Let $\mathcal{M}' = (\mathbf{T}, \mathbf{S}, \Sigma')$ be a schema mapping specified by a finite set of s-t tgds. We say that $\mathcal{M}'$ is a *chase-inverse* of $\mathcal{M}$ if I and $chase_{\mathcal{M}'}(chase_{\mathcal{M}}(I))$ are homomorphically equivalent, for every source instance I. □

Note that a chase inverse makes it possible to recover the original source instance up to homomorphic equivalence.

The next theorem shows that extended inverses that are given by s-t tgds have an equivalent characterization as chase-inverses. This characterization is a precise measure of the goodness of extended inverses for reverse data exchange.

THEOREM 3.17. *Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ be a schema mapping specified by a finite set of s-t tgds, and let $\mathcal{M}' = (\mathbf{T}, \mathbf{S}, \Sigma')$ be a "reverse" schema mapping specified by a finite set of s-t tgds. The following statements are equivalent:*

(1) $\mathcal{M}'$ is an extended inverse of $\mathcal{M}$.

(2) $\mathcal{M}'$ is a chase-inverse of $\mathcal{M}$.

Theorem 3.17 easily extends to the case when $\mathcal{M}'$ is given by tgds with constants, and we also allow these as chase-inverses. Also, Theorem 3.17 gives us another tool to verify whether a schema mapping $\mathcal{M}'$ is an extended inverse of a schema mapping $\mathcal{M}$.

EXAMPLE 3.18. As in the proof of part (3) of Theorem 3.15, consider the schema mapping $\mathcal{M}$ specified by the s-t tgd

$$P(x, y) \rightarrow \exists z(Q(x, z) \wedge Q(z, y))$$

Let $\mathcal{M}'$ be the schema mapping specified by the target-to-source tgd $Q(x, z) \wedge Q(z, y) \rightarrow P(x, y)$. We can show that $\mathcal{M}'$ is an extended inverse of $\mathcal{M}$ by showing that $\mathcal{M}'$ is a chase-inverse of $\mathcal{M}$. Indeed, let I be a source instance, let $U = chase_{\mathcal{M}}(I)$ and let $V = chase_{\mathcal{M}'}(U)$. We shall show that $I \subseteq V$ and that $V \rightarrow I$, which imply that V and I are homomorphically equivalent.

First, we observe that every fact $P(a, b)$ in I generates (via the chase with $\mathcal{M}$) two facts in U, namely $Q(a, Z_{ab})$ and $Q(Z_{ab}, b)$, where Z_{ab} is a fresh new null, distinct for every choice of a and b. (Moreover, these are the only types of facts that are generated in V.) Then V must contain every fact $P(a, b)$ (in order to satisfy $\mathcal{M}'$ for $Q(a, Z_{ab})$ and $Q(Z_{ab}, b)$). Thus, $I \subseteq V$.

We now observe that every extra fact of V (i.e., not in I) can only be of the form $P(Z_{ab}, Z_{bc})$, arising via $\mathcal{M}'$ from two facts of U of the form $Q(Z_{ab}, b)$ and $Q(b, Z_{bc})$. But then U must contain two additional facts, $Q(a, Z_{ab})$ and $Q(Z_{bc}, c)$. At the same time, the only way to have $Q(a, Z_{ab})$ and $Q(Z_{ab}, b)$ in U is to have the fact $P(a, b)$ in I. Consider now the mapping h where $h(x) = x$ for members x of I, and where $h(Z_{ab}) = a$. Then h is a homomorphism from V to I. In particular, for every extra fact $P(Z_{ab}, Z_{bc})$ in V, we have that $P(h(Z_{ab}), h(Z_{bc}))$ is $P(a, b)$, which we have shown must exist in I. □

We have already noted that the schema mapping $\mathcal{M}'$ in the above example cannot be an inverse of $\mathcal{M}$ (since, as shown in [10], such an inverse would have to make use of the *Constant* predicate, for this particular $\mathcal{M}$). Thus, this example shows in particular that the characterization via chase-inverses does not hold for inverses (since $\mathcal{M}'$ is a chase-inverse of $\mathcal{M}$ but not an inverse of $\mathcal{M}$).

The next example uses Theorem 3.17 to prove that a schema mapping $\mathcal{M}''$ is *not* an extended inverse of a schema mapping $\mathcal{M}$.

EXAMPLE 3.19. As in the proof of part (3) of Theorem 3.15 and Example 3.18, consider the schema mapping $\mathcal{M}$ specified by

$$P(x, y) \rightarrow \exists z(Q(x, z) \wedge Q(z, y))$$

Moreover, let $\mathcal{M}''$ be the schema mapping specified by

$$Q(x, z) \wedge Q(z, y) \wedge Constant(x) \wedge Constant(y) \rightarrow P(x, y).$$

It was shown in [10] that $\mathcal{M}''$ is an inverse of $\mathcal{M}$. We now show that $\mathcal{M}''$ is not an extended inverse of $\mathcal{M}$ by showing that $\mathcal{M}''$ fails to be a chase-inverse of $\mathcal{M}$. (Here we allow chase-inverses to be specified by tgds with constants, as discussed after Theorem 3.17.)

To show that $\mathcal{M}''$ is not a chase-inverse of $\mathcal{M}$, consider the source instance $I = \{P(W, Z)\}$, where W and Z are nulls. Chasing I with $\mathcal{M}$ gives the target instance $U = \{Q(W, Y), Q(Y, Z)\}$ where Y is a null. Chasing U with $\mathcal{M}''$ gives the empty instance, since there are no constants in U. Hence, $chase_{\mathcal{M}''}(chase_{\mathcal{M}}(I))$ and I are not homomorphically equivalent. □

The above two examples point out problems with the notion of inverse, which do not arise for the new notion of extended inverse. Example 3.18 shows a natural "inverse" (the chase-inverse) that is not captured by the notion of inverse. Conversely, Example 3.19 shows an inverse that fails to be a chase-inverse.

Most schema mappings occurring in practice do not possess extended inverses, since they do not even possess inverses. Invertibility and extended invertibility are strong notions that, intuitively, cover the case of "no information loss" in a schema mapping. To address schema mappings with information loss, which is the frequent case, we will go beyond extended invertibility and study extended recoveries in Section 4.

4. Extended Recoveries

Arenas, Pérez, and Riveros [2] introduced the notions of recovery and maximum recovery.

DEFINITION 4.1. ([2]) Let $\mathcal{M}$ be a schema mapping from a source schema $\mathbf{S}$ to a target schema $\mathbf{T}$. A schema mapping $\mathcal{M}'$ from $\mathbf{T}$ to $\mathbf{S}$ is a *recovery* of $\mathcal{M}$ if for every source instance I, the pair (I, I) is in $\mathcal{M} \circ \mathcal{M}'$. The schema mapping $\mathcal{M}'$ is a *maximum recovery* of $\mathcal{M}$ if (1) $\mathcal{M}'$ is a recovery for $\mathcal{M}$, and (2) for every recovery $\mathcal{M}''$ of $\mathcal{M}$, we have that $\mathcal{M} \circ \mathcal{M}' \subseteq \mathcal{M} \circ \mathcal{M}''$. □

Thus, a maximum recovery is a recovery that is optimal among all recoveries in the sense that the composition of $\mathcal{M}$ with $\mathcal{M}'$ is the smallest among the compositions of $\mathcal{M}$ with every other recovery. Similar to the framework in [7, 10], the study of recoveries in [2] is carried out in the context in which source instances are ground. While some of the results in [2] continue to hold even when source instances are not restricted to be ground, certain other results do not. In particular, one of the important results in [2] is that every schema mapping specified by s-t tgds has a maximum recovery. However, we show in the next proposition that there is a schema mapping specified by s-t tgds with no maximum recovery when source instances can be non-ground.

PROPOSITION 4.2. *Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ be a schema mapping where $\Sigma = \{P(x, y) \rightarrow \exists z (Q(x, z) \wedge Q(z, y))\}$. Then $\mathcal{M}$ has no maximum recovery when source instances can be non-ground.*

PROOF. *(Sketch)* Our proof makes use of the notion of a *witness solution* from [2]. An instance J over $\mathbf{T}$ is a *witness* for a ground instance I over $\mathbf{S}$ under the schema mapping $\mathcal{M}$ if for every ground source instance I' over $\mathbf{S}$, if $J \in \mathrm{Sol}(\mathcal{M}, I')$, then $\mathrm{Sol}(\mathcal{M}, I) \subseteq \mathrm{Sol}(\mathcal{M}, I')$. In addition, if $J \in \mathrm{Sol}(\mathcal{M}, I)$, then J is called a *witness solution* for I under $\mathcal{M}$.

We will show that $\mathcal{M}$ has no witness solution for a specific instance I, when source instances may be non-ground. Using a straightforward generalization of Theorem 3.5 of [2] to non-ground instances, we will then infer that $\mathcal{M}$ has no maximum recovery.

Let $I = \{P(0, 1), P(1, 0)\}$, and let J_I denote $\mathit{chase}_{\mathcal{M}}(I)$, which is $\{Q(0, U), Q(U, 1), Q(1, V), Q(V, 0)\}$, where U and V are nulls created by the chase. It is easy to see that any (witness) solution J for I under $\mathcal{M}$ must contain the tuples $Q(0, X)$, $Q(X, 1)$, $Q(1, Y)$, $Q(Y, 0)$, for some values X and Y, which may be nulls or constants. We consider four cases that cover all possibilities of values for X and Y and show that in each of the cases, J cannot be a witness solution. The cases we consider are: (1) $X = Y$; (2) $X \neq Y$ and at least one of X or Y is not 0 or 1; (3) $X \neq Y$ and $X = 0$ and $Y = 1$; and (4) $X \neq Y$ and $X = 1$ and $Y = 0$. $\square$

Next, we define the notions of extended recovery and maximum extended recovery.

DEFINITION 4.3. A schema mapping $\mathcal{M}'$ is an *extended recovery* of a schema mapping $\mathcal{M}$ if $(I, I) \in e(\mathcal{M}) \circ e(\mathcal{M}')$, for every source instance I. $\square$

We note that it is straightforward to verify that the condition $(I, I) \in e(\mathcal{M}) \circ e(\mathcal{M}')$ is equivalent to $e(\mathrm{Id}) \subseteq e(\mathcal{M}) \circ e(\mathcal{M}')$.

DEFINITION 4.4. A schema mapping $\mathcal{M}'$ is a *maximum extended recovery* of a schema mapping $\mathcal{M}$ if (1) $\mathcal{M}'$ is an extended recovery for $\mathcal{M}$, and (2) for every extended recovery $\mathcal{M}''$ of $\mathcal{M}$, we have $e(\mathcal{M}) \circ e(\mathcal{M}') \subseteq e(\mathcal{M}) \circ e(\mathcal{M}'')$. $\square$

Clearly, if $\mathcal{M}_1$ and $\mathcal{M}_2$ are two maximum extended recoveries of a schema mapping $\mathcal{M}$, then $e(\mathcal{M}) \circ e(\mathcal{M}_1) = e(\mathcal{M}) \circ e(\mathcal{M}_2)$. Thus, the quantity $e(\mathcal{M}) \circ e(\mathcal{M}')$, where $\mathcal{M}'$ is a maximum extended recovery of $\mathcal{M}$, is a constant $\mathcal{C}_{\mathcal{M}}$ that depends only on $\mathcal{M}$. (in particular, it is independent of the choice of $\mathcal{M}'$.) Furthermore, by Definition 4.3, $\mathcal{C}_{\mathcal{M}}$ is the smallest superset of $e(\mathrm{Id})$ among all sets $e(\mathcal{M}) \circ e(\mathcal{M}'')$, as $\mathcal{M}''$ ranges over the extended recoveries of $\mathcal{M}$. In a precise sense, $\mathcal{C}_{\mathcal{M}}$ is the closest we can get to extended identity schema mapping $e(\mathrm{Id})$ via extended recoveries. This discussion suggests the following definition.

DEFINITION 4.5. Let $\mathcal{M}$ be a schema mapping that admits a maximum extended recovery $\mathcal{M}'$. Then the *information loss* of $\mathcal{M}$ is defined as the set difference $(e(\mathcal{M}) \circ e(\mathcal{M}')) \setminus \rightarrow$. $\square$

In what follows, we will show that the information loss of a schema mapping $\mathcal{M}$ specified by a finite set of s-t tgds can be characterized solely in terms of $\mathcal{M}$.

4.1 Characterization of Maximum Extended Recoveries

DEFINITION 4.6. Let $\mathcal{M}$ be a schema mapping. We say that $I_1 \rightarrow_{\mathcal{M}} I_2$ if $\mathrm{eSol}_{\mathcal{M}}(I_2) \subseteq \mathrm{eSol}_{\mathcal{M}}(I_1)$. $\square$

The next proposition characterizes $\rightarrow_{\mathcal{M}}$ when $\mathcal{M}$ is specified by a finite set of s-t tgds.

PROPOSITION 4.7. *Let $\mathcal{M}$ be a schema mapping specified by a finite set of s-t tgds. For all source instances I_1, I_2, we have $I_1 \rightarrow_{\mathcal{M}} I_2$ if and only if $\mathit{chase}_{\mathcal{M}}(I_1) \rightarrow \mathit{chase}_{\mathcal{M}}(I_2)$.*

We will show that maximum extended recoveries for schema mappings specified by s-t tgds always exist. We shall actually prove something stronger, for which we need yet another definition.

DEFINITION 4.8. A schema mapping $\mathcal{M}'$ is a *strong maximum extended recovery* of a schema mapping $\mathcal{M}$ if (1) $\mathcal{M}'$ is an extended recovery for $\mathcal{M}$, and (2) for every extended recovery $\mathcal{M}''$ of $\mathcal{M}$, we have $e(\mathcal{M}') \subseteq e(\mathcal{M}'')$. $\square$

Note that, by monotonicity of composition, every strong maximum extended recovery of $\mathcal{M}$ is a maximum extended recovery of $\mathcal{M}$. Thus, a strong maximum extended recovery $\mathcal{M}'$ not only minimizes $e(\mathcal{M}) \circ e(\mathcal{M}'')$ among all extended recoveries $\mathcal{M}''$, but, even more, minimizes $e(\mathcal{M}'')$ among all extended recoveries $\mathcal{M}''$.

LEMMA 4.9. *Let $\mathcal{M}$ be a schema mapping specified by a finite set of s-t tgds. Put*

$$\mathcal{M}^* = \{ (\mathit{chase}_{\mathcal{M}}(I), I) \mid I \text{ is a source instance } \}.$$

If $\mathcal{M}'$ is an extended recovery of $\mathcal{M}$, then $\mathcal{M}^* \subseteq e(\mathcal{M}')$; equivalently, $e(\mathcal{M}^*) \subseteq e(\mathcal{M}')$

PROOF. Let I be a source instance. Since $\mathcal{M}'$ is a extended recovery, we have that $(I, I) \in e(\mathcal{M}) \circ e(\mathcal{M}')$. Hence, there is a J such that $(I, J) \in e(\mathcal{M})$ and $(J, I) \in e(\mathcal{M}')$. In particular, J is an extended solution for I under $\mathcal{M}$. Since $\mathit{chase}_{\mathcal{M}}(I)$ is an extended universal solution for I under $\mathcal{M}$, we have that $\mathit{chase}_{\mathcal{M}}(I) \rightarrow J$. Thus, $(\mathit{chase}_{\mathcal{M}}(I), I) \in \rightarrow \circ e(\mathcal{M}') = e(\mathcal{M}')$. Finally, it is easy to see that $\mathcal{M}^* \subseteq e(\mathcal{M}')$ if and only if $e(\mathcal{M}^*) \subseteq e(\mathcal{M}')$. $\square$

THEOREM 4.10. *Every schema mapping $\mathcal{M}$ specified by a finite set of s-t tgds has a strong maximum extended recovery. Specifically, the schema mapping*

$$\mathcal{M}^* = \{(\mathit{chase}_{\mathcal{M}}(I), I) \mid I \text{ is a source instance}\}$$

is a strong maximum extended recovery of $\mathcal{M}$..

PROOF. $\mathcal{M}^*$ is an extended recovery of $\mathcal{M}$ because, for every source instance I, we have that $(I, I) \in \mathcal{M} \circ \mathcal{M}^* \subseteq e(\mathcal{M}) \circ e(\mathcal{M}^*)$. Now, assume that $\mathcal{M}'$ is an extended recovery of $\mathcal{M}$. By Lemma 4.9, we have that $e(\mathcal{M}^*) \subseteq e(\mathcal{M}')$. $\square$

We note that in the ground case (where we restrict attention to ground source instances), there is no analog to Theorem 4.10 for maximum recoveries. For example, let $\mathcal{M}$ be the schema mapping specified by $P(x) \rightarrow Q(x, x)$. It is not hard to show that there is no recovery $\mathcal{M}'$ such that $\mathcal{M}' \subseteq \mathcal{M}''$, for every recovery $\mathcal{M}''$ of $\mathcal{M}$. This is another advantage of extended recoveries over recoveries.

Our next main result is Theorem 4.13, which shows that the quantity $\mathcal{C}_{\mathcal{M}} = e(\mathcal{M}) \circ e(\mathcal{M}')$, where $\mathcal{M}'$ is a maximum extended recovery, coincides with $\rightarrow_{\mathcal{M}}$. The proof of Theorem 4.13 makes use of Proposition 4.11 and Lemma 4.12 below.

PROPOSITION 4.11. *Let $\mathcal{M}$ be a schema mapping specified by a finite set of s-t tgds. Then $\rightarrow_{\mathcal{M}} = \rightarrow \circ \rightarrow_{\mathcal{M}} \circ \rightarrow$.*

PROOF. The containment $\rightarrow_{\mathcal{M}} \subseteq \rightarrow \circ \rightarrow_{\mathcal{M}} \circ \rightarrow$ is obvious. The reverse inclusion can be shown by observing that $\rightarrow \subseteq \rightarrow_{\mathcal{M}}$. $\square$

LEMMA 4.12. *Let $\mathcal{M}$ be a schema mapping specified by a finite set of s-t tgds. If $\mathcal{M}^* = \{(\mathit{chase}_{\mathcal{M}}(I), I) : I \text{ is a source instance } \}$, then $e(\mathcal{M}) \circ e(\mathcal{M}^*) = \rightarrow_{\mathcal{M}}$.*

PROOF. We first show that $\rightarrow_{\mathcal{M}} \subseteq \mathcal{M} \circ \rightarrow \circ \mathcal{M}^*$. Assume that $\mathit{chase}_{\mathcal{M}}(I_1) \rightarrow \mathit{chase}_{\mathcal{M}}(I_2)$. Then $(I_1, \mathit{chase}_{\mathcal{M}}(I_1)) \in \mathcal{M}$, $\mathit{chase}_{\mathcal{M}}(I_1) \rightarrow \mathit{chase}_{\mathcal{M}}(I_2)$, and $(\mathit{chase}_{\mathcal{M}}(I_2), I_2) \in \mathcal{M}^*$; hence, $(I_1, I_2) \in \mathcal{M} \circ \rightarrow \circ \mathcal{M}^*$.

29

We now show that $\mathcal{M} \circ \rightarrow \circ \mathcal{M}^* \subseteq \rightarrow_{\mathcal{M}}$. If $(I_1, I_2) \in \mathcal{M} \circ \rightarrow \circ \mathcal{M}^*$, then there is a target instance J such that $(I_1, J) \in \mathcal{M}$ and $J \rightarrow chase_M(I_2)$. Since $chase_{\mathcal{M}}(I_1)$ is a universal solution for I_1, we have that $chase_{\mathcal{M}}(I_1) \rightarrow J$; hence, $chase_{\mathcal{M}}(I_1) \rightarrow chase_{\mathcal{M}}(I_2)$.

We have shown that $\mathcal{M} \circ \rightarrow \circ \mathcal{M}^* = \rightarrow_{\mathcal{M}}$. So by Proposition 4.11, we have $e(\mathcal{M}) \circ e(\mathcal{M}^*) = \rightarrow_{\mathcal{M}}$. □

THEOREM 4.13. *Let $\mathcal{M}$ be a schema mapping specified by a finite set of s-t tgds. The following statements are equivalent:*

(1) $\mathcal{M}'$ is a maximum extended recovery of $\mathcal{M}$.

(2) $e(\mathcal{M}) \circ e(\mathcal{M}') = \rightarrow_{\mathcal{M}}$.

PROOF. Assume first that $\mathcal{M}'$ is a maximum extended recovery of $\mathcal{M}$. By Theorem 4.10, we know that $\mathcal{M}^*$ is a maximum extended recovery of $\mathcal{M}$, and so $e(\mathcal{M}) \circ e(\mathcal{M}') = e(\mathcal{M}) \circ e(\mathcal{M}^*)$. But by Lemma 4.12, we have that $e(\mathcal{M}) \circ e(\mathcal{M}^*) = \rightarrow_{\mathcal{M}}$. So $e(\mathcal{M}) \circ e(\mathcal{M}') = \rightarrow_{\mathcal{M}}$, as desired.

Assume now that $e(\mathcal{M}) \circ e(\mathcal{M}') = \rightarrow_{\mathcal{M}}$. Then $\mathcal{M}'$ is an extended recovery of $\mathcal{M}$, since $(I, I) \in \rightarrow_{\mathcal{M}}$, and so $(I, I) \in e(\mathcal{M}) \circ e(\mathcal{M}')$. Let $\mathcal{M}''$ be an arbitrary extended recovery. Since $\mathcal{M}^*$ is a maximum extended recovery, we have $e(\mathcal{M}) \circ e(\mathcal{M}^*) \subseteq e(\mathcal{M}) \circ e(\mathcal{M}'')$. So by Lemma 4.12, we have $\rightarrow_{\mathcal{M}} \subseteq e(\mathcal{M}) \circ e(\mathcal{M}'')$. Therefore, $e(\mathcal{M}) \circ e(\mathcal{M}') \subseteq e(\mathcal{M}) \circ e(\mathcal{M}'')$. Since $\mathcal{M}''$ is an arbitrary extended recovery, this implies that $\mathcal{M}'$ is a maximum extended recovery. □

The preceding results yield a characterization of the information loss of a schema mapping specified by s-t tgds.

COROLLARY 4.14. *If $\mathcal{M}$ is a schema mapping specified by a finite set of s-t tgds, then the information loss of $\mathcal{M}$ is $\rightarrow_{\mathcal{M}} \setminus \rightarrow$. In other words, for every maximum extended recovery $\mathcal{M}'$ of $\mathcal{M}$, we have that $(e(\mathcal{M}) \circ e(\mathcal{M}')) \setminus \rightarrow = \rightarrow_{\mathcal{M}} \setminus \rightarrow$.*

Note that if schema mapping $\mathcal{M}$ specified by s-t tgds is extended invertible, then $\rightarrow_{\mathcal{M}} = e(\mathrm{Id}) = \rightarrow$, and hence the information loss is empty.

COROLLARY 4.15. *Let $\mathcal{M}$ be a schema mapping specified by a finite set of s-t tgds. The following are equivalent:*

(1) $\mathcal{M}$ is extended invertible.

(2) $\rightarrow_{\mathcal{M}} = \rightarrow$.

(3) $\mathcal{M}$ has no information loss, that is, $\rightarrow_{\mathcal{M}} \setminus \rightarrow = \emptyset$.

The final result of this section relates extended inverses to maximum extended recoveries.

PROPOSITION 4.16. *Let $\mathcal{M}$ be a schema mapping specified by a finite set of s-t tgds and let $\mathcal{M}'$ be an arbitrary schema mapping. If $\mathcal{M}$ is extended invertible, then $\mathcal{M}'$ is a maximum extended recovery of $\mathcal{M}$ if and only if $\mathcal{M}'$ is an extended inverse of $\mathcal{M}$.*

4.2 Information Loss on Ground Instances

We now consider schema mappings specified by s-t tgds but restrict the source instances to be ground. As shown in [2], every such schema mapping $\mathcal{M}$ has a maximum recovery $\mathcal{M}'$. By the definition of a maximum recovery, the quantity $\mathcal{M} \circ \mathcal{M}'$ is a constant that depends only on $\mathcal{M}$. This motivates the following notion.

DEFINITION 4.17. *Let $\mathcal{M}$ be a schema mapping specified by a finite set of s-t tgds. The information loss of $\mathcal{M}$ on ground instances is the set difference $(\mathcal{M} \circ \mathcal{M}') \setminus \mathrm{Id}$, where $\mathcal{M}'$ is a maximum recovery of $\mathcal{M}$ and Id is the identity schema mapping on ground instances.* □

Intuitively, the information loss of $\mathcal{M}$ "measures" how much $\mathcal{M}$ deviates from being an invertible mapping. Next, we introduce a notion that is an adaptation to ground instances of the earlier notion of $\rightarrow_{\mathcal{M}}$, and use it to characterize the information loss of $\mathcal{M}$ on ground instances.

DEFINITION 4.18. *Let $\mathcal{M}$ be a schema mapping specified by a finite set of s-t tgds. We write $\rightarrow_{\mathcal{M},g}$ to denote the following schema mapping:*

$$\{(I_1, I_2) : I_1, I_2 \text{ are ground instances and } \mathrm{Sol}_{\mathcal{M}}(I_2) \subseteq \mathrm{Sol}_{\mathcal{M}}(I_1)\}.$$

PROPOSITION 4.19. *Let $\mathcal{M}$ be a schema mapping specified by a finite set of s-t tgds. If $\mathcal{M}'$ is a maximum recovery of $\mathcal{M}$, then*

$$\mathcal{M} \circ \mathcal{M}' = \rightarrow_{\mathcal{M},g}.$$

Consequently, the information loss of $\mathcal{M}$ on ground instances is equal to $\rightarrow_{\mathcal{M},g} \setminus \mathrm{Id}$.

We note that the preceding proposition can be obtained from results in the full version of [2], which, however, does not have the notion of information loss.

5. Maximum Extended Recoveries: Language

In this section, we show that the quasi-inverse algorithm for full tgds (See Section 4.2 of [10]) can be used to compute a maximum extended recovery of a schema mapping that is specified by a finite set of full s-t tgds.

The algorithm takes as input a schema mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where Σ is specified by a finite set of full s-t tgds. The quasi-inverse algorithm for full tgds on $\mathcal{M}$ returns a reverse schema mapping $\mathcal{M}' = (\mathbf{T}, \mathbf{S}, \Sigma')$, where Σ' is a set of disjunctive tgds with inequalities. Our next result shows that the quasi-inverse algorithm for full tgds on $\mathcal{M}$ returns a maximum extended recovery of $\mathcal{M}$.

THEOREM 5.1. *Let $\mathcal{M}$ be a schema mapping specified by a finite set of full s-t tgds. The quasi-inverse algorithm for full tgds on $\mathcal{M}$ produces a maximum extended recovery of $\mathcal{M}$ given by disjunctive tgds with inequalities.*

The language of disjunctive tgds with inequalities turns out to be the "smallest" possible language needed to define maximum extended recoveries of schema mappings specified by full s-t tgds.

THEOREM 5.2. *There is a schema mapping $\mathcal{M}$ specified by a finite set of full s-t tgds such that:*

(1) $\mathcal{M}$ has a maximum extended recovery specified by a finite set of disjunctive tgds with inequalities.

(2) $\mathcal{M}$ has no maximum extended recovery specified by a set of disjunctive tgds.

(3) $\mathcal{M}$ has no maximum extended recovery specified by tgds with inequalities.

PROOF. *(Hint)* The schema mapping having the desired properties is $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where Σ consist of $P(x, y) \rightarrow P'(x, y)$ and $T(x) \rightarrow P'(x, x)$. A maximum extended recovery of $\mathcal{M}$ is the schema mapping $\mathcal{M}^* = (\mathbf{S}, \mathbf{T}, \Sigma^*)$, where Σ^* consists of $P'(x, y) \wedge x \neq y \rightarrow P(x, y)$ and $P'(x, x) \rightarrow T(x) \vee P(x, x)$. □

It is an open problem to identify the exact language needed to express maximum recoveries of schema mappings specified by arbitrary (not necessarily full) s-t tgds.

6. Applications to Data Exchange and Beyond

In this section, we study three applications of maximum extended recoveries: reverse data exchange, reverse query answering, and comparing schema mappings. Central to all these applications is the the notion of a universal-faithful schema mapping, which generalizes the notion of a chase-inverse and provides a procedural counterpart to the notion of maximum extended recovery.

6.1 Reverse Data Exchange: Universal-Faithful

We have seen that extended inverses specified by s-t tgds have an equivalent characterization as chase-inverses. This characterization shows the usefulness of extended inverses for reverse data exchange. However, for schema mappings that are not extended invertible, chase-inverses do not exist.

We shall define the notion of a *universal-faithful* schema mapping, which is a relaxation of the notion of chase-inverse. We argue that such a schema mapping has the desired properties for reverse data exchange (even when no extended inverses exist). Furthermore, we show that maximum extended recoveries that are given by disjunctive tgds coincide with universal-faithful mappings. This characterization gives, in a precise way, the goodness of maximum extended recoveries for reverse data exchange.

In the next definition and subsequent results, we make use of the *disjunctive chase* with disjunctive tgds. Chasing with disjunctive dependencies has been considered before in various contexts [6, 8, 10]. Intuitively, the disjunctive chase is an extension of the standard chase where each step "branches" out several instances, each satisfying one of the disjuncts in the dependency that is applied. Thus, the result of the disjunctive chase is, in general, a set of instances.

DEFINITION 6.1. Take $\mathcal{M}$ to be a schema mapping specified by a finite set of s-t tgds, and $\mathcal{M}'$ to be a schema mapping specified by a finite set of disjunctive tgds. Then $\mathcal{M}'$ is *universal-faithful* for $\mathcal{M}$ if for every source instance I, the following hold for $chase_{\mathcal{M}'}(chase_{\mathcal{M}}(I)) = \{V_1, ..., V_k\}$:

(1) For every $V_l \in \{V_1, ..., V_k\}$, we have $I \rightarrow_{\mathcal{M}} V_l$.

(2) There is some $V_i \in \{V_1, ..., V_k\}$ such that $V_i \rightarrow_{\mathcal{M}} I$.

(3) For every I' such that $I \rightarrow_{\mathcal{M}} I'$, there is some $V_j \in \{V_1, ..., V_k\}$ such that $V_j \rightarrow I'$.

In general, the condition $I_1 \rightarrow_{\mathcal{M}} I_2$ can be interpreted as saying that I_2 exports, via data exchange with $\mathcal{M}$, at least as much information (and possibly more) as I_1. This is apparent, since $I_1 \rightarrow_{\mathcal{M}} I_2$ is the same as $chase_{\mathcal{M}}(I_1) \rightarrow chase_{\mathcal{M}}(I_2)$, for the case we are considering (where $\mathcal{M}$ is specified by s-t tgds). Intuitively, the data that is in $chase_{\mathcal{M}}(I_2)$ "dominates" the data in $chase_{\mathcal{M}}(I_1)$. Then we can interpret the above definition as follows.

The first condition states that every instance in the set $\{V_1, ..., V_k\}$ exports at least as much information as the original source instance I. The second condition is a converse: the information exported by I is at least as much as that exported by some member of $\{V_1, ..., V_k\}$. Thus, conditions (1) and (2) together imply that there is some V_i that exports exactly the same information as I. (The existence of such V_i is what constitutes a *faithful* schema mapping, as defined in [10].) The last condition states the "universality" of the set $\{V_1, ..., V_k\}$ in the following sense: for every instance I' that exports at least as much information as I (note that I' can be I), there is some instance in the set $\{V_1, ..., V_k\}$ that has a homomorphism into I'. Note that condition (2) is actually implied by conditions (1) and (3), but we write it down because of its significance.

If $\mathcal{M}'$ has no disjunctions, then $chase_{\mathcal{M}'}(chase_{\mathcal{M}}(I))$ is a single instance V that exports the same information as I; also, V is universal w.r.t. instances I' such that $I \rightarrow_{\mathcal{M}} I'$.

Note also that if a schema mapping $\mathcal{M}$ is extended invertible (this means $\rightarrow_{\mathcal{M}}=\rightarrow$ by Theorem 3.13) and $\mathcal{M}'$ is a schema mapping that is specified by s-t tgds (with no disjunction) such that $\mathcal{M}'$ is universal-faithful for $\mathcal{M}$, then $\mathcal{M}'$ is a chase-inverse of $\mathcal{M}$.

The next theorem gives a situation where maximum extended recoveries and universal-faithful schema mappings coincide.

THEOREM 6.2. *Let $\mathcal{M}$ be a schema mapping specified by a finite set of s-t tgds and $\mathcal{M}'$ a schema mapping specified by finite set of disjunctive tgds. The following statements are equivalent:*

(1) *$\mathcal{M}'$ is a maximum extended recovery of $\mathcal{M}$.*

(2) *$\mathcal{M}'$ is universal-faithful for $\mathcal{M}$.*

6.2 Reverse Query Answering

Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ be a schema mapping, let q be a query over $\mathbf{T}$, and let I be an instance over $\mathbf{S}$. Since there can be more than one solution for I w.r.t. $\mathcal{M}$, the widely adopted semantics for answering q for I w.r.t. $\mathcal{M}$ is the *certain-answers* semantics.

DEFINITION 6.3. *The* certain-answers of q for I w.r.t. $\mathcal{M}$, *denoted as $certain_{\mathcal{M}}(q, I)$, is $\bigcap_{(I,J) \in \mathcal{M}} q(J)$.* □

In *reverse query answering*, we assume that we have performed data exchange with $\mathcal{M}$ from a source instance I, and the query q is posed against $\mathbf{S}$ instead. This problem arises in schema evolution, for example, when "old" data is migrated to a "new" schema, but there are still queries that need to access the "old" data. We note that most research on query answering has focused on the "direct" query answering, where the query is over the target.

The reverse query answering problem is trivial if I is still available. In other words, simply evaluate $q(I)$ to answer the query q against I. However, if I is no longer available, a natural question is whether $q(I)$ can be answered by using a maximum extended recovery. Since there are many possible source instances I' that can be "returned" by a maximum extended recovery $\mathcal{M}'$, a natural way of defining the semantics of reverse query answering is to consider *all* such possible I'. Thus, we can use $certain_{e(\mathcal{M}) \circ e(\mathcal{M}')}(q, I)$ to define the semantics of reverse query answering, provided that $\mathcal{M}'$ is a maximum extended recovery of $\mathcal{M}$.

The next theorem shows that if $\mathcal{M}'$ is an extended inverse of $\mathcal{M}$, then $certain_{e(\mathcal{M}) \circ e(\mathcal{M}')}(q, I)$, where q is a conjunctive query, coincides with $q(I)_{\downarrow}$. Here, $q(I)_{\downarrow}$ denotes the result of evaluating $q(I)$ and discarding all tuples that contain at least one null. Furthermore, if $\mathcal{M}'$ is an extended recovery, the converse also holds.

THEOREM 6.4. *Let $\mathcal{M}$ and $\mathcal{M}'$ be two schema mappings.*

(1) *If $\mathcal{M}'$ is an extended inverse of $\mathcal{M}$, then $certain_{e(\mathcal{M}) \circ e(\mathcal{M}')}(q, I) = q(I)_{\downarrow}$, for every source instance I and every conjunctive query q over the source schema.*

(2) *If $\mathcal{M}'$ is an extended recovery of $\mathcal{M}$ with the property that $certain_{e(\mathcal{M}) \circ e(\mathcal{M}')}(q, I) = q(I)_{\downarrow}$, for every source instance I and every conjunctive query q over the source schema, then $\mathcal{M}'$ is an extended inverse of $\mathcal{M}$.*

The above result is an indication of the goodness of the certain-answer semantics that we adopted for reverse query answering, since it shows that in the particular case of an extended invertible schema mapping $\mathcal{M}$ (where the source instance I can be recovered up to homomorphic equivalence), $certain_{e(\mathcal{M}) \circ e(\mathcal{M}')}(q, I)$ coincides with $q(I)_{\downarrow}$ (which is the best we can do).

The next result shows that in the case when $\mathcal{M}$ is specified by s-t tgds and there exists a maximum extended recovery $\mathcal{M}'$ of $\mathcal{M}$ that is specified by disjunctive s-t tgds, we can use the *chase* to

compute $certain_{e(\mathcal{M}) \circ e(\mathcal{M}')}(q, I)$. In particular, assume that we are given a target instance J that is the result of the original data exchange (chase) of I with $\mathcal{M}$. We can then employ the reverse chase of J with $\mathcal{M}'$ as follows: compute the set of source instances that form the result of the reverse (disjunctive) chase, evaluate the original query over these instances, and then take the intersection of all null-free tuples. This gives us precisely the certain answers. We note that the proof makes essential use of the fact that a maximum extended recovery given by disjunctive tgds is universal-faithful.

THEOREM 6.5. *Let $\mathcal{M}$ be a schema mapping specified by a finite set of s-t tgds, let $\mathcal{M}'$ be a maximum extended recovery of $\mathcal{M}$ specified by disjunctive tgds, and let q be a conjunctive query over the source schema. Then*

$$certain_{e(\mathcal{M}) \circ e(\mathcal{M}')}(q, I) = (\bigcap_{K \in \mathcal{K}} q(K))_{\downarrow},$$

where $\mathcal{K} = chase_{\mathcal{M}'}(chase_{\mathcal{M}}(I))$.

6.3 Comparing Schema Mappings

DEFINITION 6.6. A schema mapping $\mathcal{M}_1$ is *less lossy* than another schema mapping $\mathcal{M}_2$ if $\rightarrow_{\mathcal{M}_1} \subseteq \rightarrow_{\mathcal{M}_2}$. We say $\mathcal{M}_1$ is *strictly less lossy* than $\mathcal{M}_2$ if $\rightarrow_{\mathcal{M}_1} \subsetneq \rightarrow_{\mathcal{M}_2}$. $\square$

If $\mathcal{M}_1$ is less lossy than $\mathcal{M}_2$, then $\mathcal{M}_1$ has a smaller information loss than $\mathcal{M}_2$ (according to Definition 4.5). In other words, $\mathcal{M}_1$ is "more invertible" than $\mathcal{M}_2$.

EXAMPLE 6.7. Consider the schema mappings $\mathcal{M}_1 = (\mathbf{S}, \mathbf{T}, \Sigma_1)$ and $\mathcal{M}_2 = (\mathbf{S}, \mathbf{T}, \Sigma_2)$, where Σ_1 and Σ_2 are as follows:

$$\Sigma_1 = \{P(x, y) \rightarrow P'(x, y)\}$$
$$\Sigma_2 = \{P(x, y) \rightarrow \exists z P'(x, z), \ P(x, y) \rightarrow \exists u P'(u, y)\}$$

The first schema mapping $\mathcal{M}_1$ copies the binary relation P to the target relation P', while the second copies each component of P separately into the same target relation P'. We now show that $\mathcal{M}_1$ is less lossy than $\mathcal{M}_2$, that is, $\rightarrow_{\mathcal{M}_1} \subseteq \rightarrow_{\mathcal{M}_2}$. Indeed, if $(I_1, I_2) \in \rightarrow_{\mathcal{M}_1}$, then it follows immediately that $I_1 \rightarrow I_2$, since $\mathcal{M}_1$ is a "copying" schema mapping. Hence, $I_1 \rightarrow_{\mathcal{M}_2} I_2$. In fact, $\mathcal{M}_1$ is a schema mapping that has no information loss (i.e., $\rightarrow_{\mathcal{M}_1} = e(\mathrm{Id})$), since it is also the case that if $I_1 \rightarrow I_2$, then $(I_1, I_2) \in \rightarrow_{\mathcal{M}_1}$. Moreover, $\mathcal{M}_1$ is strictly less lossy than $\mathcal{M}_2$; let $I = \{P(1, 0)\}$ and let $I' = \{P(1, 1), P(0, 0)\}$. It is easy to see that $(I, I') \in \rightarrow_{\mathcal{M}_2}$ but $(I, I') \notin \rightarrow_{\mathcal{M}_1}$. $\square$

A theoretical framework for comparing schema mappings can be quite useful towards the justification of the design of algorithms that generate schema mappings, such as those of [11, 15]. Each of these algorithms generates schema mappings from a visual specification of the relationship between two schemas. There are multiple ways to interpret a visual specification in general. As a simple example, the schema mappings $\mathcal{M}_1$ and $\mathcal{M}_2$ in Example 6.7 are two possible interpretations of a visual specification that relates (via arrows) the first and, respectively, second component of P to the first and, respectively, second component of P'. We note that both schema mapping generation algorithms of [11, 15] generate $\mathcal{M}_1$, which is the less lossy schema mapping of the two.

Finally, we characterize the property of being "less lossy", provided the schema mappings compared are specified by s-t tgds and have maximum extended recoveries specified by disjunctive tgds.

THEOREM 6.8. *Let $\mathcal{M}_1$ and $\mathcal{M}_2$ be schema mappings specified by finite sets of s-t tgds and having the same source schema. Let $\mathcal{M}'_1$ and $\mathcal{M}'_2$ be schema mappings that are specified by disjunctive tgds and are maximum extended recoveries of $\mathcal{M}_1$ and $\mathcal{M}_2$, respectively. The following statements are equivalent:*

(1) $\rightarrow_{\mathcal{M}_1} \subseteq \rightarrow_{\mathcal{M}_2}$

(2) For every source instance I and for every member V_1 of the set $chase_{\mathcal{M}'_1}(chase_{\mathcal{M}_1}(I))$, there is a member V_2 of the set $chase_{\mathcal{M}'_2}(chase_{\mathcal{M}_2}(I))$ such that $V_2 \rightarrow V_1$.

To see the theorem in action, consider $\mathcal{M}_1$ and $\mathcal{M}_2$ of Example 6.7. The reverse schema mapping $\mathcal{M}'$ specified by $\{P'(x, y) \rightarrow P(x, y)\}$ is a maximum extended recovery for both $\mathcal{M}_1$ and $\mathcal{M}_2$. It is easy to see that for every source instance I, there is a homomorphism from $chase_{\mathcal{M}'}(chase_{\mathcal{M}_2}(I))$ to $chase_{\mathcal{M}'}(chase_{\mathcal{M}_1}(I))$. Hence, the schema mapping $\mathcal{M}_1$ is less lossy than $\mathcal{M}_2$.

7. Concluding Remarks

We developed a new framework for reverse data exchange that allows source instances to contain not only constants but also nulls. In the process, we introduced and studied the notions of maximum extended recovery and information loss of a schema mapping. We believe that the results presented here may, in the long run, lead to novel applications in the design and optimization of schema mappings. An immediate problem that is left open is to identify the language needed to express maximum extended recoveries of schema mappings specified by s-t tgds.

8. References

[1] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

[2] M. Arenas, J. Pérez, and C. Riveros. The Recovery of a Schema Mapping: Bringing Exchanged Data Back. In *PODS*, pages 13–22, 2008.

[3] C. Beeri and M. Y. Vardi. A Proof Procedure for Data Dependencies. In *JACM*, 31(4):718–741, 1984.

[4] P. A. Bernstein. Applying Model Management to Classical Meta-Data Problems. In *CIDR*, pages 209–220, 2003.

[5] P. A. Bernstein, T. J. Green, S. Melnik, and A. Nash. Implementing Mapping Composition. In *VLDB Journal*, 17(2):333–353, 2008.

[6] A. Deutsch and V. Tannen. Optimization Properties for Classes of Conjunctive Regular Path Queries. In *DBPL*, pages 21–39, 2001.

[7] R. Fagin. Inverting schema mappings. In *ACM TODS*, 32(4), 2007.

[8] R. Fagin, P. G. Kolaitis, R. J. Miller, and L. Popa. Data Exchange: Semantics and Query Answering. In *TCS*, 336(1):89–124, 2005.

[9] R. Fagin, P. G. Kolaitis, L. Popa, and W.-C. Tan. Composing Schema Mappings: Second-order Dependencies to the Rescue. In *ACM TODS*, 30(4):994–1055, 2005.

[10] R. Fagin, P. G. Kolaitis, L. Popa, and W. C. Tan. Quasi-inverses of schema mappings. In *ACM TODS*, 33(2), 2008.

[11] A. Fuxman, M. A. Hernández, C. T. H. Ho, R. J. Miller, P. Papotti, and L. Popa. Nested Mappings: Schema Mapping Reloaded. In *VLDB*, pages 67–78, 2006.

[12] J. Madhavan and A. Y. Halevy. Composing Mappings Among Data Sources. In *VLDB*, pages 572–583, 2003.

[13] S. Melnik. *Generic Model Management: Concepts and Algorithms*, volume 2967 of *Lecture Notes in Computer Science*. Springer, 2004.

[14] A. Nash, P. A. Bernstein, and S. Melnik. Composition of Mappings Given by Embedded Dependencies. In *ACM PODS*, pages 172–183, 2005.

[15] L. Popa, Y. Velegrakis, R. J. Miller, M. A. Hernández, and R. Fagin. Translating Web Data. In *VLDB*, pages 598–609, 2002.