PONTIFICIA UNIVERSIDAD CATOLICA DE CHILE
SCHOOL OF ENGINEERING

SCHEMA MAPPING MANAGEMENT IN DATA EXCHANGE SYSTEMS

JORGE PÉREZ

Thesis submitted to the Office of Research and Graduate Studies
in partial fulfillment of the requirements for the degree of
Doctor in Engineering Sciences

Advisor:
MARCELO ARENAS

Santiago de Chile, May 2011

PONTIFICIA UNIVERSIDAD CATOLICA DE CHILE
SCHOOL OF ENGINEERING

SCHEMA MAPPING MANAGEMENT IN DATA EXCHANGE SYSTEMS

JORGE PÉREZ

Members of the Committee:
MARCELO ARENAS
CLAUDIO GUTIERREZ
PABLO BARCELÓ
YADRAN ETEROVIC
MARIANO P. CONSENS
CRISTIÁN VIAL

Thesis submitted to the Office of Research and Graduate Studies
in partial fulfillment of the requirements for the degree of
Doctor in Engineering Sciences

Santiago de Chile, May 2011

Dedicado a Constanza,
María José y Sebastián

PONTIFICIA UNIVERSIDAD CATOLICA DE CHILE
ESCUELA DE INGENIERIA

SCHEMA MAPPING MANAGEMENT IN

DATA EXCHANGE SYSTEMS

Thesis submitted to the Office of Research and Graduate Studies in partial fulfillment of the requirements for the Degree of Doctor in Engineering Sciences by

JORGE PÉREZ

ABSTRACT

A schema mapping is a high-level specification that describes how data from a source schema is to be mapped to a target schema. In the last few years, a lot of attention has been paid to the specification and subsequent manipulation of schema mappings, a problem which is of fundamental importance in metadata management. In the metadata management context, schema mappings are first class citizens, and high-level algebraic operators are used to manipulate them.

In this dissertation, we present several contributions in the formalization and the theoretical study of schema-mapping algebraic operators. We begin our study by considering the *inverse* operator that has been identified as one of the most fundamental operators in schema mappings. We propose a new semantics for inverting mappings, present algorithms for computing inverses and study expressiveness issues. We also generalize the previous

notion to a family of inverses that allows us to solve the problem of finding a mapping-specification language that is *closed* under inversion. Inversion together with the *composition* operator for schema mappings play a fundamental role in several data-interoperability tasks. Thus, we also explore the issues that arise by combining these operators, and we propose a mapping language that has good properties for inverting and composing schema mappings. We then abstract away from studying particular operators, and we embark in the analysis of the fundamental notions that all the proposals for the semantics of different schema-mapping operators share. In this respect, we propose the notions of *information* and *redundancy* in schema mappings and prove that they can provide a powerful unifying framework for schema mapping management. In particular, we show that they are fundamental in the study of the inverse, the *extract* and the *merge* operators.

Keywords: Schema mappings, data exchange, inverse, composition, merge, extract, metadata management

Members of the Doctoral Thesis Committee:
Marcelo Arenas
Claudio Gutierrez
Pablo Barceló
Yadran Eterovic
Mariano P. Consens
Cristián Vial
Santiago, May, 2011

PONTIFICIA UNIVERSIDAD CATOLICA DE CHILE
ESCUELA DE INGENIERIA

MANEJO DE CORRESPONDENCIAS ENTRE ESQUEMAS EN SISTEMAS DE INTERCAMBIO DE DATOS

Tesis enviada a la Dirección de Investigación y Postgrado en cumplimiento parcial de los requisitos para el grado de Doctor en Ciencias de la Ingeniería.

JORGE ADRIÁN PÉREZ ROJAS

RESUMEN

Las correspondencias entre esquemas de bases de datos son especificaciones de alto nivel que describen como mapear datos desde un esquema fuente a un esquema de destino. En los últimos años, mucha atención se ha puesto en la especificación y subsiguiente manipulación de correspondencias entre esquemas, problema que ha provado ser fundamental en el área de manejo de metadatos. En esta área, las correspondencias entre esquemas son ciudadanos de primera categoría, y operadores algebráicos de alto nivel son usados para manipularlas.

En este documento de tesis presentamos contribuciones en la formalización y estudio teórico de operadores sobre correspondencias entre esquemas. Comenzamos con el operador de *inverso*, proponiendo una nueva semántica para invertir correspondencias entre esquemas, presentando algoritmos para computar inversas y estudiando problemas de expresividad. También generalizamos la anterior noción a una familia de inversas que nos

permite solucionar el problema de encontrar un lenguaje *cerrado* bajo inversión. Inversión en conjunto con *composición* son fundamentales en la solución de variados problemas de interoperabilidad. Por esto, estudiamos también las problemáticas que surgen al considerar ambos operadores en conjunto, y proponemos un lenguaje con buenas propiedades para inversión y composición. Luego nos abstraemos del estudio de operadores particulares y nos embarcamos en el estudio de las propiedades fundamentales que los operadores comparten. En este punto proponemos las nociones de *información* y *redundancia* y mostramos como estas pueden ser aplicadas para proveer una infraestructura unificadora para el problema de manejo de correspondencias entre esquemas. En paticular, mostramos que estas nociones son fundamentales en el estudio de la inversa y los operadores de *extracción* y *mezcla*.

Palabras Claves: Correspondencias entre esquemas, intercambio de datos, inversa, composición, mezcla, extracción, manejo de metadatos

Miembros de la Comisión de Tesis Doctoral:
Marcelo Arenas
Claudio Gutierrez
Pablo Barceló
Yadran Eterovic
Mariano P. Consens
Cristián Vial
Santiago, mayo, 2011

ACKNOWLEDGEMENTS

I want to thank my advisor Marcelo Arenas for his immense support and close collaboration. I also want to thank Juan Reutter and Cristian Riveros. Marcelo, Juan, and Cristian, participated in several discussions that shaped many of the contributions presented in this thesis work. This work was supported by:

- CONICYT Ph.D. Grant (2007-2011)
- Microsoft Research Ph.D. Fellowship (2009/2010)
- Dirección de Investigación y Postgrado, Escuela de Ingeniería, PUC
- Vicerrectoría de Investigación, PUC

TABLE OF CONTENTS

1. INTRODUCTION

A schema mapping is a specification that describes how data from a source schema is to be mapped to a target schema. Schema mappings are of fundamental importance in data management today. In particular, they have proved to be the essential building block for several data-interoperability tasks such as data exchange, data integration and peer data management.

In the relational-database context, schema mappings are usually specified by using a logical language considering the set of relation names (or table names) of the database schemas as vocabulary. For example, consider two independent database schemas $\mathbf{A}$ and $\mathbf{B}$ containing relations $\texttt{Emp}(\texttt{name}, \texttt{lives\_in}, \texttt{works\_in})$ and $\texttt{Shuttle}(\texttt{name}, \texttt{dest})$, respectively. Relation Emp in schema $\mathbf{A}$ is used to store employees names and the places where they live in and work in. Relation Shuttle in schema $\mathbf{B}$ is intended to store names of employees that must take the shuttle bus to reach the places where they work in (destination). A possible way of relating schemas $\mathbf{A}$ and $\mathbf{B}$ is by using the following first-order logic formula:

$$\forall x \forall z \big(\exists y (\texttt{Emp}(x, y, z) \wedge y \neq z) \ \rightarrow \ \texttt{Shuttle}(x, z) \big). \tag{1.1}$$

The above formula essentially states that if relation Emp stores an employee that lives in a place different from which she/he works in, then the employee name and the place where she/he works in should be stored in relation Shuttle. Formula (1.1) describes a mapping between schemas $\mathbf{A}$ and $\mathbf{B}$ and states how data should be transformed or exchanged from one schema to the other. The left-hand side of the formula (left of the implication symbol $\rightarrow$) is called the *premise*, and the right-hand side is called the *conclusion*. Formula (1.1) is an example of what is called a *source to target tuple-generating dependency* (st-tgd) with inequalities in the premise. The language of st-tgds is one of the most popular mapping languages.

The research on the schema mapping area has mainly focused on performing data-interoperability tasks using schema mappings. However, as Bernstein (2003) pointed out, many information-system problems involve not only the design and integration of complex

application artifacts, but also their subsequent manipulation. Notice that the creation of a schema mapping may imply a considerable work by an *expert* who needs to know the semantics of the schema components. Only an expert can establish a meaningful high-level correspondence between those components. Thus, a schema mapping reflects the knowledge of the expert about the relationship between the schemas. This knowledge could, in principle, be reused beyond the interoperability tasks for which the mapping was initially created. Driven by these considerations, Bernstein (2003) proposed a general framework for managing schema mappings. In this framework, schema mappings are first class citizens, and high-level algebraic operators are used to manipulate and reuse them.

An example of mapping reuse that is conceptually easy to understand, is the *composition* of schema mappings. Consider three independent schemas $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{E}$, and the schema mappings $\mathcal{M}_{\mathbf{AB}}$ and $\mathcal{M}_{\mathbf{BE}}$ that describe how data from $\mathbf{A}$ should be mapped to $\mathbf{B}$, and how data from $\mathbf{B}$ should be mapped to $\mathbf{E}$, respectively. Assume that a new application needs to exchange data between $\mathbf{A}$ and $\mathbf{E}$. Creating a mapping between $\mathbf{A}$ and $\mathbf{E}$ could imply a considerable work since, among other requirements, one needs to know the meaning of every component of both schemas and how these components are related. Then we would like to use $\mathcal{M}_{\mathbf{AB}}$ and $\mathcal{M}_{\mathbf{BE}}$ to automatically build a mapping $\mathcal{M}_{\mathbf{AE}}$ between $\mathbf{A}$ and $\mathbf{E}$. Intuitivelly, $\mathcal{M}_{\mathbf{AE}}$ should be the result of a *composition operation* between $\mathcal{M}_{\mathbf{AB}}$ and $\mathcal{M}_{\mathbf{BE}}$, and the new mapping $\mathcal{M}_{\mathbf{AE}}$ should be *semantically consistent* with the relationships previously established by $\mathcal{M}_{\mathbf{AB}}$ and $\mathcal{M}_{\mathbf{BE}}$.

There are several questions that immediately arise. What does it mean to compose in this context? That is, what is the semantics of composing schema mappings? Notice that schema mappings are given by logical formulas thus, it is no immediately clear what is the meaning of composing this kind of specifications. Another important question is what kind of mappings are needed to specify the composition, that is, how expressive a mapping language should be in order to define the composition? Once a semantics for composition is proposed, one would also want to device algorithms (ideally, efficient algorithms) to automatically compute the composition of two mappings.

The composition of schema mapping has received a lot of attention (Madhavan & Halevy, 2003; Fagin, Kolaitis, Popa, & Tan, 2005; Nash, Bernstein, & Melnik, 2005; Arenas, Pérez, Reutter, & Riveros, 2009a; Arocena, Fuxman, & Miller, 2010), and almost all the questiones discussed above have been answered for this operator. Nevertheless, several other operations between schema mappings have been identified as important, among them the *inverse*, the *merge* and the *extract* operators (Bernstein, 2003; Fagin, 2007; Bernstein & Melnik, 2007), and for every one of these operations, the same aforementioned questions arise. This dissertation presents several contributions on the formalization and the theoretical study of schema mappings operators. In particular, we propose a new semantics for inverting mappings, present algorithms for computing inverses and study expressiveness issues. We also generalize the previous notion of inverse to a family of inverses that allows us to solve the problem of finding a mapping-specification language that is *closed* under inversion. We also explore the issues that arise by combining inversion and composition, and we propose a mapping language that has good properties for inverting and composing schema mappings. We then abstract away from studying particular operators, and we embark in the analysis of the fundamental notions that all the proposals for the semantics of different schema-mapping operators share. In this respect, we propose the notions of *information* and *redundancy* in schema mappings and prove that they are essential to build a general framework to study some fundamental properties of existing mapping operators as well as to formalize new operators. In particular, we show that they are fundamental in the study of the inverse, the *extract* and the *merge* operators.

Before going into the details of our contributions, let us exemplify some application scenarios for the notions studied in this thesis. Regarding the inverse operator, in a data exchange context (Fagin, Kolaitis, Miller, & Popa, 2005), if a mapping $\mathcal{M}$ is used to exchange data from a source to a target schema, an inverse of $\mathcal{M}$ can be used to exchange the data back to the source, thus *reversing* the application of $\mathcal{M}$. Another application is schema evolution, where the inverse together with the composition play a crucial role (Bernstein & Melnik, 2007; Fagin, Kolaitis, Popa, & Tan, 2011). Consider a mapping $\mathcal{M}$ between schemas $\mathbf{A}$ and $\mathbf{B}$, and assume that schema $\mathbf{A}$ evolves into a schema $\mathbf{A}'$. This evolution

can be expressed as a mapping $\mathcal{M}'$ between $\mathbf{A}$ and $\mathbf{A}'$. Thus, the relationship between the new schema $\mathbf{A}'$ and schema $\mathbf{B}$ can be obtained by inverting mapping $\mathcal{M}'$ and then composing the result with mapping $\mathcal{M}$.

This dissertation also formalizes the notions of *information* and *redundancy* of schema mappings. The former notion essentially measures the amount of information that can be transferred by a schema mapping, while the latter measures how *efficient* is the mapping in terms of the resources used to store the information being transferred. In practical scenarios, schema mappings may be generated automatically, either by applying mapping operators, or by automatic tools performing schema matching techniques. Thus, it would be desirable to have some tools to compare schema mappings in order to choose the one that better satisfies certain criteria. The amount of information and redundancy of a mapping can be useful criteria to discriminate between competing mappings. As an application scenario, consider the *extract* operator that intuitively captures the idea of upgrading a legacy database. Consider a database with schema $\mathbf{S}$ that stores *legacy* data, and an application that consumes data from $\mathbf{S}$ by means of a mapping $\mathcal{M}$. In general, not all the information of $\mathbf{S}$ *participates* in the mapping and, thus, it is natural to ask whether one can *upgrade* the legacy schema $\mathbf{S}$ into a new schema $\mathbf{S}'$ that stores only the information that is being mapped by $\mathcal{M}$. Thus, in $\mathbf{S}'$ one should store the same amount of information that is transfered by $\mathcal{M}$. Moreover, we would also want to be non redundant in the way that we store the information in $\mathbf{S}'$ since we do not want to store information from $\mathbf{S}$ that is not needed by the application. One can readily see that the abstract notions of information and redundancy are in the core of the extract operator.

1.1. Contributions of this Dissertation

This dissertation presents contributions on schema mapping management, in particular, in the formalization and the theoretical study of schema mappings operators. Below we describe the contributions of this dissertation.

1.1.1. Inverting mappings

One of the most fundamental operators in schema mapping management is the *inversion* of schema mappings. Given a mapping $\mathcal{M}$ from a schema $\mathbf{A}$ to a schema $\mathbf{B}$, *an inverse* of $\mathcal{M}$ is a new mapping that describes the *reverse* relationship from $\mathbf{B}$ to $\mathbf{A}$.

As our first contribution in this dissertation, we study the semantics of the inverse operator. Fagin (2007) proposes a first formal definition for what it means for a schema mapping $\mathcal{M}'$ to be an inverse of a schema mapping $\mathcal{M}$. Roughly speaking, Fagin's definition is based on the idea that a mapping composed with its inverse should be equal to the *identity schema mapping*. More formally, Fagin (2007) introduces an identity schema mapping $\overline{\mathrm{Id}}$, suitably adapted for the case of mappings specified by source-to-target tuple-generating dependencies (st-tgds). Then he says that $\mathcal{M}'$ is an inverse of $\mathcal{M}$ if $\mathcal{M}$ composed with $\mathcal{M}'$ coincides with the mapping $\overline{\mathrm{Id}}$. This notion turns out to be rather restrictive, as it is rare that a schema mapping possesses an inverse. In view of this limitation, in a subsequent work, Fagin, Kolaitis, Popa, and Tan (2008) introduce the notion of a *quasi-inverse* of a schema mapping. The idea of the quasi-inverse is to relax the notion of inverse by not differentiating between source instances that are equivalent for data exchange purposes. Although numerous non-invertible schema mappings possess natural and useful quasi-inverses (Fagin, Kolaitis, Popa, & Tan, 2008), there are still simple mappings specified by st-tgds that have no quasi-inverse. Moreover, the notions of inverse and quasi-inverse are defined by considering identity mapping $\overline{\mathrm{Id}}$, that is only appropriate for mappings that are *closed down on the left* (Fagin, 2007) and, in particular, for mappings specified by st-tgds. This leaves out numerous mappings of practical interest.

This dissertation revisits the problem of inverting schema mappings. Although our motivation is similar to that of previous work, we follow a different approach to study this problem. In fact, our main goal is to give an intuition for what it means for a schema mapping $\mathcal{M}'$ to recover *sound* information with respect to a schema mapping $\mathcal{M}$. We call such an $\mathcal{M}'$ a *recovery* of $\mathcal{M}$. Given that, in general, there may exist many possible recoveries for a mapping, we introduce an order relation on recoveries. This naturally gives

rise to the notion of maximum recovery, which is a mapping that brings back the maximum amount of sound information.

As a motivating example, let $\mathcal{M}_{\text{E-S}}$ be the mapping specified by the dependency (1.1). An example of a reverse mapping $\mathcal{M}_1$ that recovers sound information w.r.t. $\mathcal{M}_{\text{E-S}}$ is

$$\forall x \forall z \big(\texttt{Shuttle}(x, z) \rightarrow \exists u \exists v \, \texttt{Emp}(x, u, v) \big). \tag{1.2}$$

The idea is that it is *correct* to bring back to relation Emp every employee name in relation Shuttle. Notice that variables u and v are existentially quantified, and thus, this mapping is only giving information about the names of the employees but not about the places where they work in and live in. As another example, it is also correct to assume that if an employee name and a destination appears in relation Shuttle, then the destination place is exactly the place where the employee works in. Thus, mapping $\mathcal{M}_2$ defined by

$$\forall x \forall z \big(\texttt{Shuttle}(x, z) \rightarrow \exists u \, \texttt{Emp}(x, u, z) \big) \tag{1.3}$$

is also a correct way of recovering information w.r.t. $\mathcal{M}_{\text{E-S}}$. In this dissertation we propose the notion of *recovery* of a schema mapping $\mathcal{M}$ that captures the intuition of bringing back correct information that have been exchanged by using $\mathcal{M}$. Under our definition, both $\mathcal{M}_1$ and $\mathcal{M}_2$ are recoveries of $\mathcal{M}_{\text{E-S}}$

Being a recovery is a sound but mild requirement. Then it would be desirable to have some criteria to compare alternative recoveries. In our motivating example, if one has to choose between $\mathcal{M}_1$ and $\mathcal{M}_2$ as a recovery of $\mathcal{M}$, then one would probably choose $\mathcal{M}_2$, since this mapping says not only that every employee that takes a shuttle bus works and lives in some place, but also that the place where the employee works in is the destination place of the shuttle bus. Intuitively, $\mathcal{M}_2$ is *more informative than* $\mathcal{M}_1$ w.r.t. $\mathcal{M}_{\text{E-S}}$. Is there a mapping that is better than $\mathcal{M}_2$ for recovering correct information w.r.t. $\mathcal{M}_{\text{E-S}}$? If we consider the mapping $\mathcal{M}_4$ defined by dependency:

$$\forall x \forall z \big(\texttt{Shuttle}(x, z) \rightarrow \exists u \, (\texttt{Emp}(x, u, z) \wedge u \neq z) \big) \tag{1.4}$$

then $\mathcal{M}_4$ is a recovery of $\mathcal{M}_{\text{E-S}}$ that is more informative than $\mathcal{M}_2$; $\mathcal{M}_4$ additionally states that although we do not know exactly the place where the employee lives in, if that employee was brought back from table `Shuttle` then the place where the employee lives in must be different from the place where the employee works in. We formalize these intuitions by defining an *order relation* on recoveries that compares when a recovery is more informative than another recovery. This *order* on recoveries naturally gives rise to the notion of maximum recovery, which is the best way of bringing correct information back to the source schema. In our example, it can be shown that mapping $\mathcal{M}_4$ is a maximum recovery of $\mathcal{M}_{\text{E-S}}$.

In Chapter 3, we study in detail the notions of recovery and maximum recovery, and some of our main contributions include the following. We first formalize the notions of recovery and maximum recovery in Section 3.1, and in Section 3.2 we show how the notion of maximum recovery can be applied to study the schema evolution problem. In Section 3.3, we present algorithms to compute maximum recoveries for mappings specified in a language that extends the language of st-tgds, and we study several related problems. In particular, in Section 3.3.1 we present an algorithm that given a mapping $\mathcal{M}$ specified by st-tgds, returns a set of tgds that use disjunctions and equalities in the conclusion of dependencies, and a special predicate $\mathbf{C}(\cdot)$ in the premises of dependencies. In Section 3.3.2, we show that the language used in the output of the algorithm is optimal as all its features are necessary to specify maximum recoveries of mappings given by st-tgds. The set obtained as output of our algorithm is of size exponential in the size of the input mappings. In Section 3.3.2, we also show that this exponential blow-up is unavoidable. Furthermore, in Section 3.3.3 we present a special algorithm to compute maximum recoveries of full dependencies which are dependencies that do not use existential quantification in the conclusions. For this case our algorithm works in polynomial time.

In Section 3.4 we study the complexity of some decision problems related to the notion of recovery. In particular, we settle the complexity of the problem of verifying, given mappings $\mathcal{M}$ and $\mathcal{M}'$, whether $\mathcal{M}'$ is a recovery of $\mathcal{M}$, for the cases in which $\mathcal{M}$ is specified by full st-tgds and $\mathcal{M}'$ is specified by either full or non-full st-tgds. We show that

if $\mathcal{M}'$ is specified by full tgds then the problem is coNP-complete, and if $\mathcal{M}'$ is specified by (general) st-tgds then the problem is Π_2^P-complete.

Finally, in Section 3.5 we present a relaxed notion of maximal recovery. We prove that the notion maximal recovery is a strict relaxation of the notion of maximum recovery by showing that the existence of maximal recoveries is guaranteed for a wider class of mappings.

Some of the result presented in Chapter 3 were published in PODS (Arenas, Pérez, & Riveros, 2008) and in TODS (Arenas, Pérez, & Riveros, 2009).

1.1.2. Query language-based inverses and closure properties

In the framework proposed by Bernstein (2003) schema mappings are first class citizens, and high-level algebraic operators are used to manipulate and reuse them. In this algebraic context, a natural question is whether a logical language for specifying mappings is *closed* under the application of some operator; given schema mappings specified in a language $\mathcal{L}$ and an algebraic operator, can the result of the operator be also specified in $\mathcal{L}$? Furthermore, complex transformations of schema mappings can be obtained by combining several operators. Thus, one may wonder whether a closure property holds for a set of operators. Such a closure property would ensure that the output of some operator can be used as the input for subsequent operators. This has been raised as a "prominent issue" in metadata management (Kolaitis, 2005).

The main goal of Chapter 4, is to find a mapping-specification language that is closed under inversion. This goal amounts to (1) first choose a particular semantics for the inverse operator, and then (2) prove that under this semantics, there exists a mapping language $\mathcal{L}$ such that every schema mapping specified in $\mathcal{L}$ has an inverse also specified in $\mathcal{L}$. Thus, we have to deal with two parameters: the semantics for inverting mappings, and the language used for specifying mappings. As a desiderata, we would like to have a *natural* and *useful* semantics, and a mapping-specification language expressive enough to contain the class of

st-tgds. We show in Chapter 4 that the notions of Fagin-inverse (Fagin, 2007) and quasi-inverse (Fagin, Kolaitis, Popa, & Tan, 2008) could not meet our requirements. We also explain why the notion of maximum recovery that we introduce in Chapter 3 does not admit a closed mapping language. Thus, necessarily, we need to look for a weaker notion of inverse to obtain our desired closure result.

To this end, we first introduce in Section 4.1 a unifying framework that gives us a range of natural and useful notions of inverse, that characterizes previous notions and allows us to reach our goal. Our framework is based on the following idea. Intuitively, any natural notion of inverse should capture the intuition that, if $\mathcal{M}$ describes how to exchange data, its inverse must describe how to *recover* the initial data (or, at least, part of it). Moreover, there is a soundness requirement; we would like to recover only *sound information*, that is, information that was already present before the exchange. But, what does it mean to recover sound information? By answering this simple yet fundamental question, we uncover a rich theory. We measure the amount of information that can be recovered by considering classes of queries. This gives rise to the notion of $\mathcal{C}$-recovery of a mapping $\mathcal{M}$, that is a mapping that recovers sound information for $\mathcal{M}$ under a query language $\mathcal{C}$. We further introduce an order relation on $\mathcal{C}$-recoveries, that leads to the notion of $\mathcal{C}$-maximum recovery, which is a mapping that recovers the maximum amount of information according to $\mathcal{C}$. In fact, we prove that there is a bound on the amount of information that can be recovered for a given mapping $\mathcal{M}$, that depends only on $\mathcal{M}$ and the query language $\mathcal{C}$. We then prove that a $\mathcal{C}$-maximum recovery is a mapping that reaches that bound.

We study several other properties about $\mathcal{C}$-maximum recoveries, and develop a set of tools that play a central role in finding mapping languages closed under inversion. Among others, in Section 4.1.1 we provide necessary and sufficient conditions for the existence of $\mathcal{C}$-maximum recoveries, for any given query language $\mathcal{C}$. In Section 4.2 we also show that this new notion is general enough to include the previously proposed notions of inverse (Fagin, 2007; Fagin, Kolaitis, Popa, & Tan, 2008), and that it is related to the recently raised topic of schema-mapping optimization (Fagin, Kolaitis, Nash, & Popa, 2008).

Let CQ be the class of conjunctive queries. The main result of Chapter 4 is that, when we consider the notion of CQ-maximum recovery as our semantics for inversion of mappings, there exists a language that contains the class of st-tgds and is closed under inversion. More specifically, in Section 4.3 we show that the language of $\text{CQ}^{\mathbf{C},\neq}$-TO-CQ dependencies, that is, st-tgds extended with inequalities and predicate $\mathbf{C}(\cdot)$ in the premises, satisfies this property. Moreover, we provide an algorithm that, given a mapping $\mathcal{M}$ specified by a set of $\text{CQ}^{\mathbf{C},\neq}$-TO-CQ dependencies, returns a mapping specified by $\text{CQ}^{\mathbf{C},\neq}$-TO-CQ dependencies that is a CQ-maximum recovery of $\mathcal{M}$. In Sections 4.3.2 and 4.3.3 we prove several results that show that our choice of CQ-maximum recovery as the semantics for inversion and $\text{CQ}^{\mathbf{C},\neq}$-TO-CQ as the mapping-specification language is, in a precise technical sense, optimal for obtaining a mapping language closed under inversion.

Some of the result presented in Chapter 4 were published in VLDB 2009 (Arenas, Pérez, Reutter, & Riveros, 2009b). Some results in Chapter 4 are reported for the first time in this dissertation, in particular, the result about the closure property for CQ-maximum recovery and the language of $\text{CQ}^{\mathbf{C},\neq}$-TO-CQ dependencies.

1.1.3. On the relationship between composition and inversion

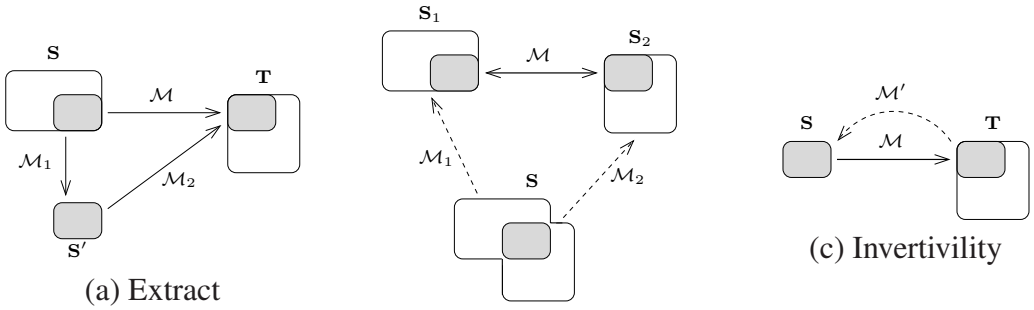
Fagin, Kolaitis, Popa, and Tan (2005) show that the language of st-tgds does not have good properties regarding composition. In view of this, Fagin, Kolaitis, Popa, and Tan (2005) propose an extension of st-tgds with second-order existential quantification (the language of SO-tgds) with good properties for composition. In particular, they show that this language is closed under composition, and they also prove several results that show that the language of SO-tgds is the right language to compose mappings. However, none of the notions of inverse proposed for schema mappings (Fagin, 2007; Fagin, Kolaitis, Popa, & Tan, 2008) have been applied to the case of SO-tgds. Studying the invertibility of mappings specified by SO-tgds is a first step to understand the difficulties in combining inversion and composition of schema mappings.

In Chapter 5 we show that, unfortunately, SO-tgds are not appropriate for our study; we show that there exist mappings specified by SO-tgds that have no Fagin-inverse, quasi-inverse and maximum recovery. In fact, we show that there exist mappings that do not even admit CQ-maximum recoveries. Thus, although the language of SO-tgds is the right language for composition, it has a bad behavior regarding inversion.

To overcome this limitation, we borrow the notion of composition w.r.t. conjunctive queries (CQ-composition), introduced by Madhavan and Halevy (2003). Then in Section 5.1 we propose a language called *plain SO-tgds* (which is a restriction of SO-tgds) such that: the language of plain SO-tgds is closed under CQ-composition, and every mapping given by plain SO-tgds has a CQ-maximum recovery. In fact, we prove something stronger, namely that every mapping specified by plain SO-tgds has a maximum recovery. To prove this last property we provide a polynomial-time algorithm that given a mapping $\mathcal{M}$ specified by a set of plain SO-tgds, returns a maximum recovery of $\mathcal{M}$ specified in a language that extends the class of plain SO-tgds. This result is interesting in its own since our algorithm is the first general polynomial-time algorithm to compute inverses of schema mappings specified by st-tgds. However, the gain in efficiency comes with the price of a stronger and less manageable mapping language for expressing inverses.

Besides the mentioned results, we prove some other interesting properties of mappings specified by plain SO-tgds. More importantly, we show in Section 5.2.1 that plain SO-tgds can be used to provide a negative answer to an open question posed by ten Cate and Kolaitis (2009, 2010).

Some of the results presented in Chapter 5 were published in VLDB 2009 (Arenas, Pérez, Reutter, & Riveros, 2009b) and in SIGMOD Record (Arenas, Pérez, Reutter, & Riveros, 2009a). Some results in Chapter 5 are reported for the first time in this dissertation, in particular, the relationship with the work by ten Cate and Kolaitis (2009, 2010).



(a) Extract

(b) Merge

(c) Invertivility

FIGURE 1.1. Examples of schema mapping operators

1.1.4. Information and redundancy of schema mappings

Schema mapping management is an area of active research, where there had been many achievements in the recent years. In fact, different proposals for several schema mapping management operators are being studied and implemented. Nevertheless, little research has being pursued towards understanding the fundamental notions that all these proposals seem to share. In particular, abstract notions of *information*, *redundancy* and *minimality* are part of almost every proposal for the semantics of schema mapping operators (Bernstein, 2003; Pottinger & Bernstein, 2003; Melnik, 2004; Fagin, 2007; Arenas, Pérez, & Riveros, 2009; Pottinger & Bernstein, 2008). Let us exemplify this with three operators: *extract*, *merge*, and *inverse*.

– *Extract* (Melnik, 2004; Melnik, Bernstein, Halevy, & Rahm, 2005): Consider a database with schema $\mathbf{S}$ that stores *legacy* data, and an application with schema $\mathbf{T}$ that consumes data from $\mathbf{S}$ by means of a mapping $\mathcal{M}$ from $\mathbf{S}$ to $\mathbf{T}$. In general, not all the information of $\mathbf{S}$ *participates* in the mapping and, thus, it is natural to ask whether one can *upgrade* the legacy schema $\mathbf{S}$ into a new schema $\mathbf{S}'$ that stores only the information that is being mapped by $\mathcal{M}$. This is the intended meaning of the *extract* operator over $\mathcal{M}$. A solution for extract is composed of two mappings, $\mathcal{M}_1$ from $\mathbf{S}$ to $\mathbf{S}'$ to drive the migration from the old to the new schema, and $\mathcal{M}_2$ from $\mathbf{S}'$ to $\mathbf{T}$ to relate the new schema with $\mathbf{T}$. A diagram of the complete process is shown in Figure 1.1(a) (the gray part represents the portion

of the schema that participates in the mappings). Intuitively, $\mathcal{M}_1$ should transfer to $\mathbf{S}'$ **the same amount of information** that is transfered by $\mathcal{M}$. Moreover, mapping $\mathcal{M}_1$ should also be **non redundant** in the way that it stores information in $\mathbf{S}'$ since we do not want to store information from $\mathbf{S}$ that is not needed by $\mathbf{T}$.

– *Merge* (Melnik, 2004; Melnik et al., 2005): Consider two independent schemas $\mathbf{S}_1$ and $\mathbf{S}_2$ and a mapping $\mathcal{M}$ between them, and assume that both schemas have materialized data that is being queried by several applications. Mapping $\mathcal{M}$ describes how data in these schemas is related, and, thus, the relationship stated by $\mathcal{M}$ leads to some **redundancy of storage**: there are corresponding pieces of data stored twice in these schemas. Thus, it is natural to ask whether one can have a single *global schema* $\mathbf{S}$ that simultaneously stores the data of $\mathbf{S}_1$ and $\mathbf{S}_2$ (and no more than that), but that is **not redundant** in the storage of the shared information. This is the intended meaning of the *merge* operation over $\mathcal{M}$. In a solution for merge one also needs two mappings $\mathcal{M}_1$ and $\mathcal{M}_2$ that describe the relationship between the new global schema and the initial schemas. These mappings ensure that an application that has used the initial schemas independently, would also be able to obtain the required data from the new global schema. A diagram of the complete process is shown in Figure 1.1(b).

– *Inverse* (Fagin, 2007): Consider a mapping $\mathcal{M}$ that is used to exchange data from $\mathbf{S}$ to $\mathbf{T}$ and assume that after exchanging data one needs to *undo* the process. That is, one needs to recover from $\mathbf{T}$ the initial information stored in $\mathbf{S}$. An inverse of $\mathcal{M}$ is a new mapping $\mathcal{M}'$ that describes how to exchange data back from $\mathbf{T}$ to $\mathbf{S}$. A diagram of the inverse is shown in Figure 1.1(c). As described by Fagin (2007), invertibility for a mapping $\mathcal{M}$ should intuitively coincide with **no loss of information**.

In Chapter 6, we address the problem of providing foundations for schema mapping management by focusing on the abstract notions of **information** and **redundancy** that, as the previous examples show, lie in the core of schema mapping operators. The formalization of these notions in a general setting would play an essential role in providing a unifying

framework for metadata management. The work by Melnik (2004) could be considered a first effort towards the development of a general framework for the area. In Chapter 6, we go a step further in this direction.

We develop theoretical tools to compare schema mappings in terms of the abstract notions of information and redundancy, providing characterizations to deal with these criteria, and showing their usefulness in defining and studying complex metadata management operators.

As an example to motivate our notions, consider the following two mappings given by st-tgds:

$$\mathcal{M}_1: \quad \forall x \forall y \forall z \big(\, A(x,y,z) \quad \rightarrow \quad \exists u \, P(x,u) \, \big)$$
$$\mathcal{M}_2: \quad \forall x \forall y \forall z \big(\, A(x,y,z) \quad \rightarrow \quad R(x) \wedge S(x,y) \, \big)$$

Intuitively, $\mathcal{M}_2$ *transfers more information* than $\mathcal{M}_1$ since the first and second components of tuples in A are being transferred to the target under $\mathcal{M}_2$, while only the first component is being transferred under $\mathcal{M}_1$. In fact, notice that the information transferred by $\mathcal{M}_1$ can be obtained from the target of $\mathcal{M}_2$ by means of the mapping $\forall x(R(x) \rightarrow \exists u \, P(x,u))$. However, the opposite is not true; we cannot obtain the information transferred by $\mathcal{M}_2$ from the target of $\mathcal{M}_1$. Consider now the mapping $\mathcal{M}_3$ given by:

$$\mathcal{M}_3: \quad \forall x \forall y \forall z \big(\, A(x,y,z) \quad \rightarrow \quad T(x,y) \, \big)$$

Intuitively, mapping $\mathcal{M}_3$ transfers the same amount of information as $\mathcal{M}_2$. Nevertheless, $\mathcal{M}_3$ is more *efficient* in the way that it structures the target data, as $\mathcal{M}_2$ stores *redundant information* in table R. In this paper, we formalize the previous notions, that is, we develop notions to compare mappings in terms of their ability to transfer source data and avoid storing redundant information in its target schema, as well as the symmetric notions of *covering* target data, and storing redundant information in the source schema. In fact, we prove the usefulness of the proposed notions by showing that they can play a central role in the study of complex metadata management operators.

More precisely, we start our investigation in Section 6.1 by defining a set of natural conditions that an order on the amount of information transferred by a schema mapping should satisfy. We then propose the order $\preceq_{\mathrm{S}}$, that is provably the strictest relation satisfying these conditions. Under our definition, the mappings in the above example satisfy that $\mathcal{M}_1 \preceq_{\mathrm{S}} \mathcal{M}_2$ but $\mathcal{M}_2 \not\preceq_{\mathrm{S}} \mathcal{M}_1$. We study some fundamental properties of $\preceq_{\mathrm{S}}$ and, in particular, we provide a characterization based on query rewriting that gives evidence of the naturalness of our notion. We also contrast our proposal with previous work on comparing mappings. Fagin, Kolaitis, Popa, and Tan (2009) propose a notion of *information loss* for schema mappings specified by st-tgds, which gives rise to an order on this type of mappings. In Section 6.1.1, we show that our notion coincides with the proposal of Fagin et al. for the class of mappings defined by st-tgds. Moreover, we also prove that beyond st-tgds, Fagin et al.'s notion does not satisfy the natural conditions that we impose over an order to compare the amount of information transferred by mappings.

As shown in the previous example, there may exist multiple ways to transfer the same information from a source schema. Thus, one also needs a way to distinguish between different alternatives. In particular, if schemas are designed together with mappings, it is desirable to use schemas that are *optimal* in the way they handle data. To deal with this requirement, we introduce in Section 6.4.2 the notion of *target redundancy*, and show that it captures the intuition of using the *exact amount of resources* needed to transfer information using a schema mapping. In fact, our notion formally captures the intuition in the previous example, as $\mathcal{M}_2$ is *target redundant* while $\mathcal{M}_3$ is not.

Furthermore, to complement our information framework, we devise two additional concepts that allow us to compare mappings that share the same target schema. Symmetrically to the definition of $\preceq_{\mathrm{S}}$, we introduce in Section 6.4.1 the order $\preceq_{\mathrm{T}}$, that intuitively measures the amount of information *covered* by a mapping, as well as a notion of *source redundancy* in Section 6.4.3. We provide characterizations and tools for all the proposed notions, and show that together they can be used as a powerful framework to study metadata management operators.

As a proof of concept, in Section 6.3 we show how the machinery developed can be used to study some metadata management problems in the context of data exchange. In particular, we provide simpler proofs for some fundamental results regarding the inverse operator proposed by Fagin (2007), and we also study the well-known schema evolution problem (Melnik, 2004; Kolaitis, 2005; Fagin et al., 2011).

Some of the results presented in Chapter 6 were published in PODS 2010 (Arenas, Pérez, Reutter, & Riveros, 2010).

1.1.5. The extract and merge operators

The *extract* and *merge* operators are considered as two of the most fundamental operators to develop a general framework for schema mapping management (Melnik, 2004; Melnik et al., 2005; Bernstein & Melnik, 2007).

In Section 7.1, we use all the machinery for the concepts of information and redundancy developed in Chapter 6 to revisit the semantics of the *extract* operator (Melnik, 2004; Melnik et al., 2005), that intuitively captures the idea of *upgrading* a legacy schema. We formalize this operator in terms of the notions developed in Chapter 6. We also provide in Section 7.1.1 an algorithm for computing it for a class of mappings that includes the mappings specified by st-tgds, and we compare our proposal with previous proposals for the same operator in Section 7.1.2.

Moreover, in Section 7.2 we also study the *merge* operator, that intuitively captures the idea of constructing a *global schema* that simultaneously stores the data that is replicated among several schemas but that it is not redundant in the storage of the shared information. Finally, in Section 7.2.1 we provide an algorithm to compute the merge operator for mappings specified by full st-tgds.

Some of the result presented in Chapter 7 were published in PODS 2010 (Arenas, Pérez, Reutter, & Riveros, 2010).

2. NOTATION AND PRELIMINARIES

In this Chapter we present the notation that is used in this dissertation. A *schema* $\mathbf{R}$ is a finite set $\{R_1, \ldots, R_k\}$ of relation symbols, with each R_i having a fixed arity n_i. Let $\mathbf{D}$ be a countably infinite domain. An instance I of $\mathbf{R}$ assigns to each n-ary relation symbol R of $\mathbf{R}$ a finite n-ary relation $R^I \subseteq \mathbf{D}^n$. The *domain* $\mathrm{dom}(I)$ of instance I is the set of all elements that occur in any of the relations R^I. If a tuple $\bar{a}$ belongs to R^I we say that $R(\bar{a})$ is a *fact* in I. We sometimes describe an instance as a set of facts. The set $\mathrm{Inst}(\mathbf{R})$ is defined to be the set of all instances of $\mathbf{R}$. Given instances $I, J \in \mathrm{Inst}(\mathbf{R})$, we write $I \subseteq J$ to denote that, for every relation symbol R of $\mathbf{R}$, it holds that $R^I \subseteq R^J$.

As is customary in the data exchange literature, we consider instances with two types of values: constants and nulls (Fagin, Kolaitis, Miller, & Popa, 2005; Fagin, 2007; Fagin, Kolaitis, Popa, & Tan, 2008). More precisely, let $\mathbf{C}$ and $\mathbf{N}$ be infinite and disjoint sets of constants and nulls, respectively, and assume that $\mathbf{D} = \mathbf{C} \cup \mathbf{N}$. An instance I is *ground* if it is composed only by constant values (elements from $\mathbf{C}$), and is *non-ground* if it is composed by constant and null values (elements from $\mathbf{C} \cup \mathbf{N}$). If we refer to a schema $\mathbf{R}$ as a ground schema, then we assume that $\mathrm{Inst}(\mathbf{R})$ is the set of all ground instances of $\mathbf{R}$. Moreover, if we refer to a schema $\mathbf{S}$ as a *source* schema, then we assume that $\mathbf{S}$ is a ground schema, and if we refer to a schema $\mathbf{T}$ as a *target* schema, then we assume that $\mathbf{T}$ is a general schema, that is, $\mathrm{Inst}(\mathbf{T})$ contains instances with constants and null values. In this paper, we usually use $\mathbf{S}$ to refer to a generic source schema and $\mathbf{T}$ to refer to a generic target schema.

2.1. Schema mappings

Given schemas $\mathbf{R}_1$ and $\mathbf{R}_2$, a *schema mapping* (or just *mapping*) from $\mathbf{R}_1$ to $\mathbf{R}_2$ is a nonempty subset of $\mathrm{Inst}(\mathbf{R}_1) \times \mathrm{Inst}(\mathbf{R}_2)$. If $\mathcal{M}$ is a mapping and $(I, J) \in \mathcal{M}$, then we say that J is a *solution for I under $\mathcal{M}$*. The set of solutions for I under $\mathcal{M}$ is denoted by $\mathrm{Sol}_{\mathcal{M}}(I)$. The domain of $\mathcal{M}$, denoted by $\mathrm{dom}(\mathcal{M})$, is defined as the set of instances I

such that $\mathrm{Sol}_{\mathcal{M}}(I) \neq \emptyset$, and the range of $\mathcal{M}$, denoted by $\mathrm{range}(\mathcal{M})$ is defined as the set of instances J that are a solution for some instance $I \in \mathrm{dom}(\mathcal{M})$. Furthermore, given a mapping $\mathcal{M}$, we denote by $\mathcal{M}^{-1}$ the mapping $\{(J, I) \mid (I, J) \in \mathcal{M}\}$.

Some of our results are focused on a special class of mappings that we call *source-to-target* mappings (st-mappings). Recall that we assume that a schema $\mathbf{R}$ is a source schema if its instances are constructed by using only constants (elements from $\mathbf{C}$), and a target schema if its instances are constructed by constants and nulls (elements from $\mathbf{C} \cup \mathbf{N}$). Thus if we refer to $\mathcal{M} = (\mathbf{R}_1, \mathbf{R}_2, \Sigma)$ as an *st-mapping*, then we assume that $\mathbf{R}_1$ is a source schema (that is, instances of $\mathbf{R}_1$ are constructed using only elements from $\mathbf{C}$) and $\mathbf{R}_2$ is a target schema (that is, instances of $\mathbf{R}_2$ are constructed by using elements from $\mathbf{C}$ and $\mathbf{N}$).

Composition of mappings

Since general mappings are simply binary relations on sets of instances, the composition of mappings can be defined by considering the classical definition of composition of binary relations. Given mappings $\mathcal{M}_{12}$ from $\mathbf{R}_1$ to $\mathbf{R}_2$ and $\mathcal{M}_{23}$ from $\mathbf{R}_2$ to $\mathbf{R}_3$, the composition of $\mathcal{M}_{12}$ and $\mathcal{M}_{23}$, denoted by $\mathcal{M}_{12} \circ \mathcal{M}_{23}$ is defined as $\mathcal{M}_{12} \circ \mathcal{M}_{23} = \{(I_1, I_3) \mid \exists I_2 : (I_1, I_2) \in \mathcal{M}_{12}$ and $(I_2, I_3) \in \mathcal{M}_{23}\}$ (Melnik, 2004; Fagin, Kolaitis, Popa, & Tan, 2005).

2.2. Queries and definability of mappings

A k-ary query Q over a schema $\mathbf{R}$, with $k \geq 0$, is a function that maps every instance $I \in \mathrm{Inst}(\mathbf{R})$ into a k-relation $Q(I) \subseteq \mathrm{dom}(I)^k$. Notice that if $k = 0$ (Q is a Boolean query), then the answer to Q is either the set with one 0-ary tuple (denoted by true), or the empty set (false). Thus, if Q is a Boolean query, then $Q(I)$ is either true or false. As is customary, we assume that queries are closed under isomorphisms.

We use CQ to denote the class of conjunctive queries and UCQ to denote the class of unions of conjunctive queries. If we extend these classes by allowing equalities or inequalities, then we use superscripts $=$ and $\neq$, respectively. Thus, for example, $\mathrm{CQ}^{=}$ is the

class of conjunctive queries with equalities and $\text{UCQ}^{\neq}$ is the class of union of conjunctive queries with inequalities. FO is the class of all first-order formulas with equality. Slightly abusing notation, we use $\mathbf{C}(\cdot)$ to denote a built-in unary predicate such that $\mathbf{C}(a)$ holds if and only if a is a constant, that is, $a \in \mathbf{C}$. If $\mathcal{L}$ is any of the previous query languages, then $\mathcal{L}^{\mathbf{C}}$ is the extension of $\mathcal{L}$ allowing predicate $\mathbf{C}(\cdot)$. For example, $\text{CQ}^{\mathbf{C},\neq}$ is the class of conjunctive queries with inequalities and predicate $\mathbf{C}(\cdot)$.

Dependencies

Let $\mathcal{L}_1$, $\mathcal{L}_2$ be query languages and $\mathbf{R}_1$, $\mathbf{R}_2$ be schemas with no relation symbols in common. A sentence Φ over $\mathbf{R}_1 \cup \mathbf{R}_2 \cup \{\mathbf{C}(\cdot)\}$ is an $\mathcal{L}_1$-TO-$\mathcal{L}_2$ *dependency from* $\mathbf{R}_1$ *to* $\mathbf{R}_2$ if Φ is of the form

$$\forall \bar{x}\,(\varphi(\bar{x}) \rightarrow \psi(\bar{x})),$$

where

(1) $\bar{x}$ is the tuple of free variables in both $\varphi(\bar{x})$ and $\psi(\bar{x})$;
(2) $\varphi(\bar{x})$ is an $\mathcal{L}_1$-formula over $\mathbf{R}_1 \cup \{\mathbf{C}(\cdot)\}$ if $\mathbf{C}(\cdot)$ is allowed in $\mathcal{L}_1$, and over $\mathbf{R}_1$ otherwise, and
(3) $\psi(\bar{x})$ is an $\mathcal{L}_2$-formula over $\mathbf{R}_2 \cup \{\mathbf{C}(\cdot)\}$ if $\mathbf{C}(\cdot)$ is allowed in $\mathcal{L}_2$, and over $\mathbf{R}_2$ otherwise.

We call $\varphi(\bar{x})$ the *premise* of Φ, and $\psi(\bar{x})$ the *conclusion* of Φ. If $\mathbf{S}$ is a source schema and $\mathbf{T}$ is a target schema, an $\mathcal{L}_1$-TO-$\mathcal{L}_2$ dependency from $\mathbf{S}$ to $\mathbf{T}$ is called an $\mathcal{L}_1$-TO-$\mathcal{L}_2$ *source-to-target dependency* ($\mathcal{L}_1$-TO-$\mathcal{L}_2$ st-dependency), and an $\mathcal{L}_1$-TO-$\mathcal{L}_2$ dependency from $\mathbf{T}$ to $\mathbf{S}$ is called an $\mathcal{L}_1$-TO-$\mathcal{L}_2$ *target-to-source dependency* ($\mathcal{L}_1$-TO-$\mathcal{L}_2$ ts-dependency).

Three fundamental classes of dependencies for data exchange, and in particular for inverting schema mappings, are source-to-target tuple-generating dependencies (st-tgds), full source-to-target tuple-generating dependencies (full st-tgds) and target-to-source disjunctive tuple-generating dependencies with inequalities and predicate $\mathbf{C}(\cdot)$ (Fagin, Kolaitis, Miller, & Popa, 2005; Fagin, Kolaitis, Popa, & Tan, 2008). The former corresponds to the class of CQ-TO-CQ st-dependencies, and the latter to the class of $\text{CQ}^{\neq,\mathbf{C}}$-TO-UCQ ts-dependencies. An FO-TO-CQ dependency is full if its conclusion does not include

existential quantifiers and, thus, the class of full st-tgds corresponds to the class of full CQ-TO-CQ st-dependencies.

Semantics of dependencies, safeness

Let I be an instance of a schema $\mathbf{R} = \{R_1, \ldots, R_m\}$. Instance I can be represented as an $(\mathbf{R} \cup \{\mathbf{C}(\cdot)\})$-structure $\mathfrak{A}_I = \langle A, R_1^A, \ldots, R_m^A, \mathbf{C}^A \rangle$, where $A = \mathrm{dom}(I)$ is the universe of $\mathfrak{A}_I$, $R_i^A = R_i^I$ for $i \in [1, m]$ and $\mathbf{C}^A = A \cap \mathbf{C}$. This representation is used to define the semantics of FO over source and target instances (here we assume familiarity with some basic notions of first-order logic).

Let $\mathbf{R}_1 = \{S_1, \ldots, S_m\}$ be a schema and I an instance of $\mathbf{R}_1$. If $\varphi(\bar{x})$ is an FO-formula over $\mathbf{R}_1 \cup \{\mathbf{C}(\cdot)\}$ and $\bar{a}$ is a tuple of elements from $\mathrm{dom}(I)$, then we say that I satisfies $\varphi(\bar{a})$, denoted by $I \models \varphi(\bar{a})$, if and only if $\mathfrak{A}_I \models \varphi(\bar{a})$. Whenever it holds that $I \models \varphi(\bar{a})$, we say that $\bar{a}$ is an answer for φ over instance I. Furthermore, let $\mathbf{R}_2 = \{T_1, \ldots, T_n\}$ be a schema with no relation symbols in common with $\mathbf{R}_1$, and J an instance of $\mathbf{R}_2$. Then $K = (I, J)$ is an instance of $\mathbf{R}_1 \cup \mathbf{R}_2$ defined as $S_i^K = S_i^I$ and $T_j^K = T_j^J$, for $i \in [1, m]$ and $j \in [1, n]$. Notice that $\mathrm{dom}(K) = \mathrm{dom}(I) \cup \mathrm{dom}(J)$. If $\varphi(\bar{x})$ is an FO-formula over $\mathbf{R}_1 \cup \mathbf{R}_2 \cup \{\mathbf{C}(\cdot)\}$ and $\bar{a}$ is a tuple of elements from $\mathrm{dom}(I) \cup \mathrm{dom}(J)$, then we say that (I, J) satisfies $\varphi(\bar{a})$, denoted by $(I, J) \models \varphi(\bar{a})$, if and only if $\mathfrak{A}_K \models \varphi(\bar{a})$. As usual, we say that an instance satisfies a set Σ of dependencies if the instance satisfies each dependency in Σ.

We impose the following safety condition on $\mathcal{L}_1$-TO-$\mathcal{L}_2$ dependencies. Recall that an FO-formula $\varphi(\bar{x})$ is *domain-independent* if its answer depends only on the database instance but not on the underlying domain (Fagin, 1982). Let $\mathbf{R}_1$ and $\mathbf{R}_2$ be schemas with no relation symbols in common and $\Phi = \forall \bar{x} \, (\varphi(\bar{x}) \rightarrow \psi(\bar{x}))$ an $\mathcal{L}_1$-TO-$\mathcal{L}_2$ dependency from $\mathbf{R}_1$ to $\mathbf{R}_2$. Then we say that Φ is *domain-independent* if both $\varphi(\bar{x})$ and $\psi(\bar{x})$ are domain-independent. The following strategy can be used to evaluate Φ: Given instances I, J of $\mathbf{R}_1$ and $\mathbf{R}_2$, respectively, we have that $(I, J) \models \Phi$ if and only if for every tuple $\bar{a}$ of elements from $\mathrm{dom}(I)$, if $I \models \varphi(\bar{a})$, then every component of tuple $\bar{a}$ is in $\mathrm{dom}(J)$

and $J \models \psi(\bar{a})$. We note that this strategy cannot be used for non domain-independent $\mathcal{L}_1$-TO-$\mathcal{L}_2$ dependencies.

Definability of mappings

Let $\mathbf{R}_1$ and $\mathbf{R}_2$ be schemas with no relation symbols in common and Σ a set of FO-sentences over $\mathbf{R}_1 \cup \mathbf{R}_2 \cup \{\mathbf{C}(\cdot)\}$. We say that a mapping $\mathcal{M}$ from $\mathbf{R}_1$ to $\mathbf{R}_2$ is *specified* by Σ, denoted by $\mathcal{M} = (\mathbf{R}_1, \mathbf{R}_2, \Sigma)$, if for every $(I, J) \in \mathrm{Inst}(\mathbf{R}_1) \times \mathrm{Inst}(\mathbf{R}_2)$, we have that $(I, J) \in \mathcal{M}$ if and only if $(I, J) \models \Sigma$.

In this dissertation we assume that every set Σ of dependencies is finite, and if Σ is a set of $\mathcal{L}_1$-TO-$\mathcal{L}_2$ dependencies, then we assume that every dependency in Σ is domain-independent (as defined above). Furthermore, we usually omit the outermost universal quantifiers from $\mathcal{L}_1$-TO-$\mathcal{L}_2$ dependencies and, thus, we write $\varphi(\bar{x}) \rightarrow \psi(\bar{x})$ instead of $\forall \bar{x}\, (\varphi(\bar{x}) \rightarrow \psi(\bar{x}))$. Finally, for the sake of readability, we usually write $\varphi(\bar{x}, \bar{y}) \rightarrow \psi(\bar{x})$ instead of $(\exists \bar{y}\, \varphi(\bar{x}, \bar{y})) \rightarrow \psi(\bar{x})$ in some examples, as these two formulas are equivalent.

2.3. Homomorphisms and universal solutions

The class of *universal solutions* for st-mappings was identified as a class of solutions that have good properties for data exchange (Fagin, Kolaitis, Miller, & Popa, 2005). To formally introduce this concept, we first review the notion of homomorphism that will be used in several proofs of this dissertation and that is also used to define universal solutions.

Let J_1 and J_2 be instances of the same schema $\mathbf{R}$. A *homomorphism* h from J_1 to J_2 is a function $h : \mathrm{dom}(J_1) \rightarrow \mathrm{dom}(J_2)$ such that,

(1) h is the identity on $\mathbf{C}$, that is, for every constant value $a \in \mathrm{dom}(J_1) \cap \mathbf{C}$ we have that $h(a) = a$, and

(2) for every $R \in \mathbf{R}$ and every tuple $(a_1, \ldots, a_k) \in R^{J_1}$, it holds $(h(a_1), \ldots, h(a_k)) \in R^{J_2}$.

Notice that a homomorphism may map a null value to either a constant or another null value, but preserves all the constant values.

Let $\mathcal{M}$ be an st-mapping, I a source instance and J a solution for I under $\mathcal{M}$. Then J is a *universal solution* for I under $\mathcal{M}$, if for every solution J' for I under $\mathcal{M}$, there exists a homomorphism from J to J'. One of the important properties of universal solutions proved by Fagin, Kolaitis, Miller, and Popa (2005) is that for every st-mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ with Σ a set of st-tgds and for every instance I of $\mathbf{S}$, there always exists a universal solution for I under $\mathcal{M}$. Moreover, it can be shown that the same property holds for st-mappings specified by FO-TO-CQ dependencies. In both cases, the *chase* procedure (Maier, Mendelzon, & Sagiv, 1979) can be used to construct a universal solution (see the next section for the definition of the chase for mappings specified by FO-TO-CQ dependencies).

2.4. Certain answers, query rewriting and the chase

Let $\mathcal{M}$ be a mapping from $\mathbf{R}_1$ to $\mathbf{R}_2$ and Q a query over schema $\mathbf{R}_2$. Given an instance I of $\mathbf{R}_1$, the set of *certain answers of Q over I under $\mathcal{M}$* is the set of tuples that belong to the evaluation of Q over every possible solution for I under $\mathcal{M}$. We denote this set by $\mathrm{certain}_{\mathcal{M}}(Q, I)$. Thus,

$$\mathrm{certain}_{\mathcal{M}}(Q, I) = \bigcap_{J \in \mathrm{Sol}_{\mathcal{M}}(I)} Q(J).$$

Given a mapping $\mathcal{M}$ from $\mathbf{S}$ to $\mathbf{T}$ and a query Q over $\mathbf{T}$, we say that a query Q' is a *rewriting of Q over the source* if Q' is a query over $\mathbf{S}$ such that for every $I \in \mathrm{Inst}(\mathbf{S})$, it holds that $Q'(I) = \mathrm{certain}_{\mathcal{M}}(Q, I)$. That is, to obtain the set of certain answers of Q over I under $\mathcal{M}$, we just have to evaluate its rewriting Q' over instance I. We usually call Q' a *source rewriting of Q* under $\mathcal{M}$.

Similarly, given a mapping $\mathcal{M}$ from $\mathbf{S}$ to $\mathbf{T}$ and a query Q' over $\mathbf{S}$, we say that a query Q is a *rewriting of Q' over the target* if Q is a query over $\mathbf{T}$ such that for every $I \in \mathrm{Inst}(\mathbf{S})$, it holds that $Q'(I) = \mathrm{certain}_{\mathcal{M}}(Q, I)$. We call Q a *target rewriting of Q'* under $\mathcal{M}$.

As it is evident by the definitions, Q' is a source rewriting of Q under $\mathcal{M}$ if and only if Q is a target rewriting of Q' under $\mathcal{M}$.

The chase

Another notion that would be used in this document is the notion of *chase* (Maier et al., 1979). This notion is tightly related with certain answers and rewriting of queries (Fagin, Kolaitis, Miller, & Popa, 2005; Arenas, Barceló, Fagin, & Libkin, 2004).

Assume that $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ is an st-mapping, where Σ is a set of FO-TO-CQ dependencies. Let I be an instance of $\mathbf{S}$, and let J_I be an instance of $\mathbf{T}$ constructed as follows. For every dependency $\sigma \in \Sigma$ of the form $\varphi(\bar{x}) \rightarrow \exists \bar{y}\, \psi(\bar{x}, \bar{y})$, with $\bar{x} = (x_1, \ldots, x_m)$, $\bar{y} = (y_1, \ldots, y_\ell)$ tuples of distinct variables, and for every m-tuple $\bar{a}$ of elements from $\mathrm{dom}(I)$ such that $I \models \varphi(\bar{a})$, do the following. Choose an ℓ-tuple $\bar{n}$ of distinct fresh values from $\mathbf{N}$, and include all the conjuncts of $\psi(\bar{a}, \bar{n})$ as facts in J_I. We call instance J_I *the chase of I with Σ*, and write $J_I = \mathrm{chase}_\Sigma(I)$. It is easy to see that $\mathrm{chase}_\Sigma(I)$ is unique up-to renaming of null values.

The instance $\mathrm{chase}_\Sigma(I)$ has several desirable properties (Fagin, Kolaitis, Miller, & Popa, 2005; Arenas et al., 2004). In particular, if $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ is an st-mapping with Σ a set of FO-TO-CQ dependencies, then for every I it holds that $\mathrm{chase}_\Sigma(I)$ is a universal solution for I under $\mathcal{M}$. We call $\mathrm{chase}_\Sigma(I)$ the *canonical universal solution* of I under $\mathcal{M}$. Fagin, Kolaitis, Miller, and Popa (2005) showed that if J is a universal solution for I under a mapping $\mathcal{M}$, then for every conjunctive query Q it holds that $\mathrm{certain}_\mathcal{M}(Q, I) = Q(J)_\downarrow$ where $Q(J)_\downarrow$ denotes the set of tuples obtained from $Q(J)$ by eliminating all the tuples that mention null values. In particular, if $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ is an st-mapping with Σ a set of FO-TO-CQ dependencies, and Q is a conjunctive query over $\mathbf{T}$, then for every instance I we have that $\mathrm{certain}_\mathcal{M}(Q, I) = Q(\mathrm{chase}_\Sigma(I))_\downarrow$. This gives us an effective procedure to compute the certain answers for conjunctive queries under an st-mapping specified by FO-TO-CQ dependencies (Fagin, Kolaitis, Miller, & Popa, 2005; Arenas et al., 2004). Moreover, notice that if Q' is a rewriting over the source of a conjunctive query Q, then it holds that $Q'(I) = Q(\mathrm{chase}_\Sigma(I))_\downarrow$. This last property is used in several proofs of this dissertation.

Disjunctive chase

We next introduce the notion of *disjunctive chase* defined by Fagin, Kolaitis, Popa, and Tan (2008). Let Σ be a set of FO$^{\mathbf{C}}$-TO-UCQ dependencies from schema $\mathbf{R}_1$ to schema $\mathbf{R}_2$. For every $I \in \mathrm{Inst}(\mathbf{R}_1)$, let J_I be an instance of $\mathbf{R}_2$ constructed with the following procedure. For every dependency $\sigma \in \Sigma$ of the form

$$\varphi(\bar{x}) \rightarrow \exists \bar{y}_1 \, \beta_1(\bar{x}, \bar{y}_1) \vee \cdots \vee \exists \bar{y}_k \, \beta_1(\bar{x}, \bar{y}_k)$$

with $\bar{x} = (x_1, \ldots, x_m)$ a tuple of distinct variables, and for every m-tuple $\bar{a}$ of elements from $\mathrm{dom}(I)$ such that $I \models \varphi(\bar{a})$, do the following. Choose an index $i \in \{1, \ldots, k\}$. Assume that $\bar{y}_i = (y_1, \ldots, y_\ell)$, then choose an ℓ-tuple $\bar{n}$ of distinct fresh values from $\mathbf{N}$, and include all the conjuncts of $\beta_i(\bar{a}, \bar{n})$ in J_I. We call J_I *a chase of* I *with* Σ. Notice that different instances are obtained by different choices of indexes in the process. Consider the set $\mathcal{J}_I = \{J_I^1, \ldots, J_I^p\}$ of all the instances that correspond to a chase of I. Then we say that $\mathcal{J}_I$ is *the (disjunctive) chase of* I *with* Σ, and write $\mathcal{J}_I = \mathrm{chase}_\Sigma(I)$ (Fagin, Kolaitis, Popa, & Tan, 2008). It is easy to see that $\mathcal{J}_I = \mathrm{chase}_\Sigma(I)$ is finite and unique up-to renaming of null values. As for the non-disjunctive chase, the disjunctive chase satisfies several desirable properties. In particular, it can be shown (Fagin, Kolaitis, Popa, & Tan, 2008), that for every pair of instances I, J, if $(I, J) \models \Sigma$, then there exists an instance $K \in \mathrm{chase}_\Sigma(I)$ and a homomorphism from K to J. By using the results in (Fagin, Kolaitis, Miller, & Popa, 2005), it is also straightforward to prove that, if I is an instance composed only by constant values and Q is a union of conjunctive queries, then the set of certain answers of I under Σ, equals the set of tuples that belongs to $Q(K)_\downarrow$, for all $K \in \mathrm{chase}_\Sigma(I)$.

Chase of the chase

Fagin, Kolaitis, Popa, and Tan (2008) extend some of the above mentioned results of the chase procedure. Let Σ be a set of FO$^{\mathbf{C}}$-TO-CQ dependencies from $\mathbf{R}_1$ to $\mathbf{R}_2$, and Σ' a set of CQ$^{\mathbf{C},\neq}$-TO-UCQ dependencies from $\mathbf{R}_2$ to $\mathbf{R}_3$. Further assume that for every inequality $x \neq x'$ that occurs in the premise of a dependency in Σ', predicates $\mathbf{C}(x)$ and $\mathbf{C}(x')$ also occurs in the premise of the same dependency (these kind of dependencies are

called *disjunctive tuple-generating dependencies with constants and inequalities among constants*). Fagin, Kolaitis, Popa, and Tan (2008) showed the following property of the successive application of the chase procedure with Σ and Σ'. Let I be an instance composed only by constant values, $J = \mathrm{chase}_{\Sigma}(I)$ and $\mathcal{V} = \mathrm{chase}_{\Sigma'}(J)$. Let K' be an instance such that there exists J' with $(I, J') \models \Sigma$ and $(J', K') \models \Sigma'$. Then there exists an instance $K \in \mathcal{V}$ and a homomorphism from K to K'.

Considering mappings, let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ be a source-to-target mappings with Σ a set of FO$^{\mathbf{C}}$-TO-CQ dependencies, and $\mathcal{M}' = (\mathbf{T}, \mathbf{S}, \Sigma')$ a target-to-source mapping with Σ' a set of CQ$^{\mathbf{C},\neq}$-TO-UCQ dependencies (fulfilling the above mentioned restriction). Let I be a source instance, $J = \mathrm{chase}_{\Sigma}(I)$ and $\mathcal{V} = \mathrm{chase}'_{\Sigma}(J)$. Notice that, the instances in $\mathcal{V}$ are not necessarily valid source instances since they may contain null values. That is, although $(I, J) \models \Sigma$ and $(J, K) \models \Sigma'$ for every $K \in \mathcal{V}$, we have that (I, K) not necessarily belongs to $\mathcal{M} \circ \mathcal{M}'$. Nevertheless, we know that for every pair of source instances I, I' such that $(I, I') \in \mathcal{M} \circ \mathcal{M}'$, there exists an instance $K \in \mathcal{V}$ and a homomorphism from K to I'.

2.5. Previous notions of inverse of schema mappings

In this section we introduce the two notions of inverse previously proposed in the literature. These notions will be used in several parts of this dissertation.

We start by recalling the definition of inverse proposed by Fagin (2007), that we call here *Fagin-inverse*[1]. A mapping $\mathcal{M}$ is *closed-down on the left* if whenever $(I, J) \in \mathcal{M}$ and $I' \subseteq I$, it holds that $(I', J) \in \mathcal{M}$. Fagin (2007) defines a notion of inverse focusing on mappings that satisfy this condition. More precisely, let $\mathbf{S}$ be a source schema. Fagin first defines an identity mapping $\overline{\mathrm{Id}}$ as

$$\overline{\mathrm{Id}} = \{(I_1, I_2) \mid (I_1, I_2) \in \mathrm{Inst}(\mathbf{S}) \times \mathrm{Inst}(\mathbf{S}) \text{ and } I_1 \subseteq I_2\},$$

[1]Fagin (2007) named his notion just as *inverse* of a schema mapping. In this dissertation we reserve the term *inverse* to refer to this operator in general, and use the name *Fagin-inverse* for the notion proposed by Fagin (2007).

which is appropriate for closed-down on the left mappings (Fagin, 2007). Then Fagin propose the following definition for an inverse of a mapping.

DEFINITION 2.5.1 (Fagin, 2007). *Let $\mathcal{M}$ be a mapping from* $\mathbf{S}$ *to* $\mathbf{T}$. *A mapping $\mathcal{M}'$ from* $\mathbf{T}$ *to* $\mathbf{S}$ *is a* Fagin-inverse *of $\mathcal{M}$ if and only if $\mathcal{M} \circ \mathcal{M}' = \overline{\mathrm{Id}}$.*

Since it is rare that a schema mapping possesses a Fagin-inverse (Fagin, 2007; Fagin, Kolaitis, Popa, & Tan, 2008), Fagin, Kolaitis, Popa, and Tan (2008) introduce the notion of a quasi-inverse of a schema mapping. The idea behind quasi-inverses is to relax the notion of inverse of a mapping by not differentiating between source instances that are data-exchange equivalent. Let $\mathcal{M}$ be a mapping from a source schema $\mathbf{S}$ to a target schema $\mathbf{T}$. Instances I_1 and I_2 of $\mathbf{S}$ are *data-exchange equivalent* w.r.t. $\mathcal{M}$, denoted by $I_1 \sim_{\mathcal{M}} I_2$, if $\mathrm{Sol}_{\mathcal{M}}(I_1) = \mathrm{Sol}_{\mathcal{M}}(I_2)$. Furthermore, given a mapping $\mathcal{M}_1$ from $\mathbf{S}$ to $\mathbf{S}$, mapping $\mathcal{M}_1[\sim_{\mathcal{M}}, \sim_{\mathcal{M}}]$ is defined as $\{(I_1, I_2) \in \mathrm{Inst}(\mathbf{S}) \times \mathrm{Inst}(\mathbf{S}) \mid \exists (I_1', I_2') : I_1 \sim_{\mathcal{M}} I_1', I_2 \sim_{\mathcal{M}} I_2'$ and $(I_1', I_2') \in \mathcal{M}_1\}$.

DEFINITION 2.5.2 (Fagin, Kolaitis, Popa, & Tan, 2008). *Let $\mathcal{M}$ be a mapping from* $\mathbf{S}$ *to* $\mathbf{T}$. *A mapping $\mathcal{M}'$ is a* quasi-inverse *of $\mathcal{M}$ if $(\mathcal{M} \circ \mathcal{M}')[\sim_{\mathcal{M}}, \sim_{\mathcal{M}}] = \overline{\mathrm{Id}}[\sim_{\mathcal{M}}, \sim_{\mathcal{M}}]$.*

3. THE MAXIMUM RECOVERY OF A SCHEMA MAPPING

In this chapter we present the notions of recovery and maximum recovery for schema mappings, and study several related problems including how to compute maximum recoveries, the language needed to express maximum recoveries and the complexity of some associated decision problems. We also introduce the relaxed notion of maximal recovery and report our initial results about this notion.

3.1. Recoveries and Maximum Recoveries

Let $\mathcal{M}$ be a mapping from a schema $\mathbf{R}_1$ to a schema $\mathbf{R}_2$, and Id the *identity schema mapping* over $\mathbf{R}_1$, that is, $\mathrm{Id} = \{(I, I) \mid I \in \mathrm{Inst}(\mathbf{R}_1)\}$. When trying to invert $\mathcal{M}$, the ideal would be to find a mapping $\mathcal{M}'$ from $\mathbf{R}_2$ to $\mathbf{R}_1$ such that, $\mathcal{M} \circ \mathcal{M}' = \mathrm{Id}$. If such a mapping exists, we know that if we use $\mathcal{M}$ to exchange data, the application of $\mathcal{M}'$ gives as result exactly the initial source instance. Unfortunately, in most cases this ideal is impossible to reach. For example, it is impossible to obtain such an inverse if $\mathcal{M}$ is specified by a set of st-tgds (Fagin, 2007). The main problem with such an ideal definition of inverse is that, in general, no matter what $\mathcal{M}'$ we choose, we will have not one but many solutions for a source instance under $\mathcal{M} \circ \mathcal{M}'$.

If for a mapping $\mathcal{M}$, there is no mapping $\mathcal{M}_1$ such that $\mathcal{M} \circ \mathcal{M}_1 = \mathrm{Id}$, at least we would like to find a schema mapping $\mathcal{M}_2$ that *does not forbid* the possibility of recovering the initial source data. That is, we would like that for every instance $I \in \mathrm{dom}(\mathcal{M})$, the space of solutions for I under $\mathcal{M} \circ \mathcal{M}_2$ contains I itself. Such a schema mapping $\mathcal{M}_2$ is called a *recovery* of $\mathcal{M}$.

DEFINITION 3.1.1. *Let $\mathbf{R}_1$ and $\mathbf{R}_2$ be two schemas, $\mathcal{M}$ a mapping from $\mathbf{R}_1$ to $\mathbf{R}_2$ and $\mathcal{M}'$ a mapping from $\mathbf{R}_2$ to $\mathbf{R}_1$. Then $\mathcal{M}'$ is a* recovery *of $\mathcal{M}$ iff $(I, I) \in \mathcal{M} \circ \mathcal{M}'$ for every instance $I \in \mathrm{dom}(\mathcal{M})$.*

Being a recovery is a sound but mild requirement. Indeed, a schema mapping $\mathcal{M}$ from $\mathbf{R}_1$ to $\mathbf{R}_2$ always has as recoveries, for example, mappings $\mathcal{M}_1 = \mathrm{Inst}(\mathbf{R}_2) \times \mathrm{Inst}(\mathbf{R}_1)$

and $\mathcal{M}_2 = \mathcal{M}^{-1} = \{(J, I) \mid (I, J) \in \mathcal{M}\}$. If one has to choose between $\mathcal{M}_1$ and $\mathcal{M}_2$ as a recovery of $\mathcal{M}$, then one would probably choose $\mathcal{M}_2$ since the space of possible solutions for a source instance I under $\mathcal{M} \circ \mathcal{M}_2$ is smaller than under $\mathcal{M} \circ \mathcal{M}_1$. In fact, if there exists a mapping $\mathcal{M}_3$ such that $\mathcal{M} \circ \mathcal{M}_3 = \mathrm{Id}$, then one would definitely prefer $\mathcal{M}_3$ over $\mathcal{M}_1$ and $\mathcal{M}_2$. In general, if $\mathcal{M}'$ is a recovery of $\mathcal{M}$, then the smaller the space of solutions generated by $\mathcal{M} \circ \mathcal{M}'$, the more informative $\mathcal{M}'$ is about the initial source instances. This notion induces an *order* among recoveries:

DEFINITION 3.1.2. *Let $\mathcal{M}$ be a mapping and $\mathcal{M}'$, $\mathcal{M}''$ recoveries of $\mathcal{M}$. We say that $\mathcal{M}'$ is* at least as informative as *$\mathcal{M}''$ for $\mathcal{M}$, and write $\mathcal{M}'' \preceq_{\mathcal{M}} \mathcal{M}'$, iff $\mathcal{M} \circ \mathcal{M}' \subseteq \mathcal{M} \circ \mathcal{M}''$.*

Moreover, we say that $\mathcal{M}'$ and $\mathcal{M}''$ are *equally informative for* $\mathcal{M}$, denoted by $\mathcal{M}' \equiv_{\mathcal{M}} \mathcal{M}''$, if $\mathcal{M}'' \preceq_{\mathcal{M}} \mathcal{M}'$ and $\mathcal{M}' \preceq_{\mathcal{M}} \mathcal{M}''$.

Example 3.1.3. Let $\mathcal{M}$ be an st-mapping specified by st-tgd:

$$P(x, y) \wedge R(y, z, u) \rightarrow T(x, y, z).$$

Then the ts-mapping $\mathcal{M}_1$ specified by $T(x, y, z) \rightarrow \exists v P(x, v)$ is a recovery of $\mathcal{M}$, as well as the ts-mapping $\mathcal{M}_2$ specified by $T(x, y, z) \rightarrow P(x, y) \wedge \exists u\, R(y, z, u)$. Intuitively, both $\mathcal{M}_1$ and $\mathcal{M}_2$ recover *sound* information given the definition of $\mathcal{M}$. Furthermore, it can be shown that $\mathcal{M}_1 \preceq_{\mathcal{M}} \mathcal{M}_2$, which agrees with the intuition that $\mathcal{M}_2$ recovers more information than $\mathcal{M}_1$. □

If for a mapping $\mathcal{M}$, there exists a recovery $\mathcal{M}'$ that is at least as informative as any other recovery of $\mathcal{M}$, then $\mathcal{M}'$ is the best alternative to bring exchanged data back, among all the recoveries. Intuitively, such a mapping $\mathcal{M}'$ recovers the maximum amount of sound information. Such a mapping $\mathcal{M}'$ is called a maximum recovery of $\mathcal{M}$.

DEFINITION 3.1.4. *Let $\mathcal{M}'$ be a recovery of a mapping $\mathcal{M}$. We say that $\mathcal{M}'$ is a* maximum recovery *of $\mathcal{M}$ if for every recovery $\mathcal{M}''$ of $\mathcal{M}$, it is the case that $\mathcal{M}'' \preceq_{\mathcal{M}} \mathcal{M}'$.*

Notice that if $\mathcal{M}_1$ and $\mathcal{M}_2$ are maximum recoveries of a mapping $\mathcal{M}$, then they are equally informative for $\mathcal{M}$, that is, $\mathcal{M}_1 \equiv_{\mathcal{M}} \mathcal{M}_2$.

Example 3.1.5. Consider st-mapping $\mathcal{M}$ and ts-mapping $\mathcal{M}_2$ from Example 3.1.3. Intuitively, $\mathcal{M}_2$ is doing the best effort to recover the information exchanged by $\mathcal{M}$. In fact, it can be shown that $\mathcal{M}_2$ is a maximum recovery of $\mathcal{M}$. □

3.1.1. Tools for studying recoveries and maximum recoveries

In this section, we present some theoretical tools for studying maximum recoveries. These tools include characterizations of when a mapping is a maximum recovery of another mapping and a general necessary and sufficient condition for the existence of maximum recoveries. These results were previously reported by Riveros (2008) and we state them here for completeness of this dissertation. The proofs of these results can be found in (Riveros, 2008; Arenas, Pérez, & Riveros, 2009).

Characterizing maximum recoveries

We first present characterizations of when a mapping $\mathcal{M}'$ is a maximum recovery of a mapping $\mathcal{M}$. For doing this, we need the notion of *reduced recovery*. A mapping $\mathcal{M}'$ is a *reduced recovery* of $\mathcal{M}$ if $\mathcal{M}'$ is a recovery of $\mathcal{M}$ and for every $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$, it holds that $I_2 \in \mathrm{dom}(\mathcal{M})$. It is easy to see that whenever $\mathcal{M}'$ is a recovery of $\mathcal{M}$, one can *extract* from $\mathcal{M}'$ a reduced recovery $\mathcal{M}''$ of $\mathcal{M}$ by discarding all the pairs of instances (J, I) of $\mathcal{M}'$ such that $I \notin \mathrm{dom}(\mathcal{M})$. The obtained reduced recovery $\mathcal{M}''$ is at least as informative as $\mathcal{M}'$ for $\mathcal{M}$ since $\mathcal{M} \circ \mathcal{M}'' \subseteq \mathcal{M} \circ \mathcal{M}'$.

The following proposition shows two alternative conditions for checking whether a mapping $\mathcal{M}'$ is a maximum recovery of a mapping $\mathcal{M}$, and that only depend on the structure of mappings $\mathcal{M}$ and $\mathcal{M}'$. The proof of the proposition can be found in (Riveros, 2008; Arenas, Pérez, & Riveros, 2009)

PROPOSITION 3.1.6 (Arenas et al., 2008; Riveros, 2008). *Let $\mathcal{M}$ and $\mathcal{M}'$ be mappings. Then the following conditions are equivalent:*

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

(2) *$\mathcal{M}'$ is a reduced recovery of $\mathcal{M}$ and $\mathcal{M} = \mathcal{M} \circ \mathcal{M}' \circ \mathcal{M}$.*

(3) *$\mathcal{M}'$ is a recovery of $\mathcal{M}$ and for every $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$, it is the case that $\emptyset \subsetneq \mathrm{Sol}_{\mathcal{M}}(I_2) \subseteq \mathrm{Sol}_{\mathcal{M}}(I_1)$.*

On the existence of maximum recoveries

The following definition introduces the notion of *witness*, that is used to provide a necessary and sufficient condition for the existence of a maximum recovery for a mapping $\mathcal{M}$.

DEFINITION 3.1.7 (Arenas et al., 2008; Riveros, 2008). *Let $\mathcal{M}$ be a mapping from a schema $\mathbf{R}_1$ to a schema $\mathbf{R}_2$ and $I \in \mathrm{Inst}(\mathbf{R}_1)$. Then instance $J \in \mathrm{Inst}(\mathbf{R}_2)$ is a* witness for I under $\mathcal{M}$ *if for every $I' \in \mathrm{Inst}(\mathbf{R}_1)$, if $J \in \mathrm{Sol}_{\mathcal{M}}(I')$, then $\mathrm{Sol}_{\mathcal{M}}(I) \subseteq \mathrm{Sol}_{\mathcal{M}}(I')$.*

A witness for an instance I under a mapping $\mathcal{M}$ is not necessarily a solution for I under $\mathcal{M}$. An instance J is a *witness solution* for I if J is both a witness and a solution for I. A witness solution can be considered as an *identifier* for a space of solutions as if J is a witness solution for instances I_1 and I_2, then $\mathrm{Sol}_{\mathcal{M}}(I_1) = \mathrm{Sol}_{\mathcal{M}}(I_2)$.

The following theorem provides a necessary and sufficient condition for the existence of maximum recoveries. The proof of the theorem can be found in (Riveros, 2008; Arenas, Pérez, & Riveros, 2009)

THEOREM 3.1.8 (Arenas et al., 2008; Riveros, 2008). *A mapping $\mathcal{M}$ has a maximum recovery iff for every $I \in \mathrm{dom}(\mathcal{M})$, there exists a witness solution for I under $\mathcal{M}$.*

Fagin, Kolaitis, Miller, and Popa (2005) identifies the class of *universal solutions* for st-mappings as a class of solutions that has good properties for data exchange. Recall that J is a *universal solution* for I under a mapping $\mathcal{M}$, if $J \in \mathrm{Sol}_{\mathcal{M}}(I)$ and for every solution J' for I under $\mathcal{M}$, there exists a homomorphism from J to J'.

It is known that for st-mappings specified by FO-TO-CQ st-dependencies, universal solutions exist for every source instance (Fagin, Kolaitis, Miller, & Popa, 2005; Arenas

et al., 2004). Moreover, st-mappings specified by FO-TO-CQ st-dependencies are *closed under target homomorphisms* (ten Cate & Kolaitis, 2009). That is, if $\mathcal{M}$ is an st-mapping specified by a set FO-TO-CQ dependencies, $(I, J) \in \mathcal{M}$ and there is a homomorphism from J to J', then $(I, J') \in \mathcal{M}$. With this last property it is straightforward to prove that if $\mathcal{M}$ is an st-mapping specified by a set of FO-TO-CQ dependencies, then every universal solution for I under $\mathcal{M}$ is a witness solution for I under $\mathcal{M}$. Thus, we obtain the following important result regarding the existence of maximum recoveries.

THEOREM 3.1.9 (Arenas et al., 2008; Riveros, 2008). *If $\mathcal{M}$ is an st-mapping specified by a set of* FO-TO-CQ *st-dependencies, then $\mathcal{M}$ has a maximum recovery.*

Notice that the previous result implies that every st-mapping specified by st-tgds has a maximum recovery.

Comparison with previous notions

In this section we report some results on the comparison of maximum recoveries and the previous notions of Fagin-inverse and quasi-inverse (see Section 2.5 for the formalization of Fagin-inverses and quasi-inverses).

Recall that a mapping $\mathcal{M}$ is total if $\mathrm{dom}(\mathcal{M})$ is the set of all source instances. The definitions of Fagin-inverse and quasi-inverse are only appropriate for total mappings, since if a mapping $\mathcal{M}$ is not total, then $\mathcal{M}$ is neither Fagin-invertible nor quasi-invertible. Moreover, the definitions of Fagin-inverse and quasi-inverse are appropriate for closed-down on the left mappings (see Section 2.5). In fact, some counterintuitive results are obtained if one removes this restriction. For example, let $\mathbf{S} = \{P(\cdot)\}$, $\mathbf{T} = \{R(\cdot)\}$ and $\mathcal{M}$ be a mapping from $\mathbf{S}$ to $\mathbf{T}$ specified by dependency $\forall x\,(P(x) \leftrightarrow R(x))$. In this case, mapping $\mathcal{M}'$ specified by $\forall x\,(R(x) \leftrightarrow P(x))$ is an *ideal* inverse of $\mathcal{M}$ since $\mathcal{M} \circ \mathcal{M}' = \mathrm{Id} = \{(I, I) \mid I \in \mathrm{Inst}(\mathbf{S})\}$. However, $\mathcal{M}'$ is neither a Fagin-inverse nor a quasi-inverse of $\mathcal{M}$ (although it is a maximum recovery of $\mathcal{M}$).

From the discussion in the previous paragraph, to compare the notions of maximum recovery, Fagin-inverse and quasi-inverse, we need to focus on the class of total st-mappings

that are closed-down on the left. This class includes, for example, the st-mappings specified by UCQ$^{\neq}$-TO-CQ st-dependencies. The following result establishes the relationship between Fagin-inverses, quasi-inverses and maximum recoveries. The proof can be found in (Riveros, 2008; Arenas, Pérez, & Riveros, 2009).

THEOREM 3.1.10 (Arenas et al., 2008; Riveros, 2008).

(1) *Let $\mathcal{M}$ be a total st-mapping that is closed-down on the left, and assume that $\mathcal{M}$ is Fagin-invertible. Then $\mathcal{M}'$ is a Fagin-inverse of $\mathcal{M}$ iff $\mathcal{M}'$ is a maximum recovery of $\mathcal{M}$.*

(2) *Let $\mathcal{M}$ be a total st-mapping that is closed-down on the left, and assume that $\mathcal{M}$ is quasi-invertible. Then $\mathcal{M}$ has a maximum recovery and, furthermore, $\mathcal{M}'$ is a maximum recovery of $\mathcal{M}$ iff $\mathcal{M}'$ is a quasi-inverse and a recovery of $\mathcal{M}$.*

It was shown in (Fagin, Kolaitis, Popa, & Tan, 2008) that there are mappings specified by st-tgds (even by full st-tgds) that has no quasi-inverse. Thus, the notion of maximum recovery has a clear advantage over quasi-inverses as a relaxation of the notion of Fagin-inverse, since maximum recoveries coincide with Fagin-inverses for Fagin-invertible mappings, and every mappings specified by st-tgds has a maximum recovery.

Example 3.1.11. Fagin, Kolaitis, Popa, and Tan (2008) showed that the schema mapping $\mathcal{M}$ specified by full st-tgd $E(x,z) \wedge E(z,y) \rightarrow F(x,y) \wedge M(z)$ has neither a quasi-inverse nor a Fagin-inverse. It is possible to show that the schema mapping $\mathcal{M}'$ specified by:

$$\begin{aligned} F(x,y) &\rightarrow \exists u(E(x,u) \wedge E(u,y)), \\ M(z) &\rightarrow \exists v \exists w(E(v,z) \wedge E(z,w)), \end{aligned}$$

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

3.1.2. Comparison with inverses for the extended solutions semantics

Fagin et al. (2009) made the observation that almost all the literature about data exchange and, in particular, the literature about inverses of schema mappings, assume that

source instances do not have null values. Although our definition of recovery and maximum recovery do not need this assumption, most of our results regarding inverses are proved for the case of st-mappings, that is, mappings in which the source instances contain only constant values while target instances may contain constant and null values. Fagin et al. (2009) go a step further and propose new refined notions for inverting mappings that consider nulls in the source. In particular, they propose the notions of *extended inverse*, and of *extended recovery* and *maximum extended recovery*. In this section, we review the definitions of the latter two notions and compare them with the previously proposed notions of recovery and maximum recovery.

The first observation to make is that since null values are intended to represent *missing* or *unknown* information, they should not be treated naively as constants (Imielinski & Lipski, 1984). In fact, as shown by Fagin et al. (2009), if one treats nulls in that way, the existence of a maximum recovery for mappings given by st-tgds is no longer guaranteed.

Example 3.1.12. Consider a source schema $\{S(\cdot,\cdot)\}$ where instances may contain null values, and let $\mathcal{M}$ be a mapping specified by st-tgd $S(x,y) \rightarrow \exists z\,(T(x,z) \wedge T(z,y))$. Then $\mathcal{M}$ has no maximum recovery if one considers a naïve semantics where null elements are used as constants in the source (Fagin et al., 2009). □

Since nulls should not be treated naively when exchanging data, Fagin et al. (2009) proposed a new way to deal with null values. Intuitively, the idea is to *close* mappings under homomorphisms. This idea is supported by the fact that nulls are intended to represent unknown data, thus, it should be possible to replace them by arbitrary values. Formaly, the authors introduce the following concept.

DEFINITION 3.1.13 (Fagin et al., 2009). *Let $\mathcal{M}$ be a mapping. The* homomorphic extension *of $\mathcal{M}$, denoted by $e(\mathcal{M})$, is the mapping*

$$e(\mathcal{M}) = \{(I,J) \mid \exists (I',J') \,:\, (I',J') \in \mathcal{M} \textit{ and there exist homomorphisms from } I \textit{ to } I' \textit{ and from } J' \textit{ to } J\}.$$

The idea is that for a mapping $\mathcal{M}$ that has nulls in source and target instances, one does not have to consider $\mathcal{M}$ but $e(\mathcal{M})$ as the mapping to deal with for exchanging data and computing mapping operators since $e(\mathcal{M})$ treats nulls in a meaningful way (Fagin et al., 2009). The following result shows that with this new semantics one can avoid anomalies as the one shown in Example 3.1.12.

THEOREM 3.1.14 (Fagin et al., 2009). *For every mapping $\mathcal{M}$ specified by a set of st-tgds and with nulls in source and target instances, $e(\mathcal{M})$ has a maximum recovery.*

As mentioned above, Fagin et al. (2009) go a step further by introducing new notions of inverse for mappings that consider nulls in the source. More specifically, the authors introduce the following definitions

DEFINITION 3.1.15 (Fagin et al., 2009). *Let $\mathcal{M}$ be a mapping from $\mathbf{R}_1$ to $\mathbf{R}_2$. Mapping $\mathcal{M}'$ is an* extended recovery *of $\mathcal{M}$ if $(I, I) \in e(\mathcal{M}) \circ e(\mathcal{M}')$, for every instance I of $\mathbf{R}_1$. Then given an extended recovery $\mathcal{M}'$ of $\mathcal{M}$, the mapping $\mathcal{M}'$ is a* maximum extended recovery *of $\mathcal{M}$ if for every extended recovery $\mathcal{M}''$ of $\mathcal{M}$, it holds that $e(\mathcal{M}) \circ e(\mathcal{M}') \subseteq e(\mathcal{M}) \circ e(\mathcal{M}'')$.*

At a first glance, one may think that the notions of maximum recovery and maximum extended recovery are incomparable. Nevertheless, the next result shows that there is a tight connection between these two notions. In particular, it shows that the notion proposed by Fagin et al. (2009) can be defined in terms of our notion of maximum recovery. In the theorem we focus on mappings $\mathcal{M}$ from $\mathbf{R}_1$ to $\mathbf{R}_2$ such that $e(\mathcal{M})$ is a *total mapping*, that is, $\mathrm{dom}(e(\mathcal{M})) = \mathrm{Inst}(\mathbf{R}_1)$. Notice that the notion of extended recovery of a mapping $\mathcal{M}$ from $\mathbf{R}_1$ to $\mathbf{R}_2$ is defined only for the case when in which $e(\mathcal{M})$ is total, since if there exists an instance I of $\mathbf{R}_1$ such that $I \notin \mathrm{dom}(e(\mathcal{M}))$, then for every mapping $\mathcal{M}'$ we have that $(I, I) \notin e(\mathcal{M}) \circ e(\mathcal{M}')$ and thus, $\mathcal{M}$ does not have an extended recovery.

THEOREM 3.1.16. *Let $\mathcal{M}$ be a mapping such that $e(\mathcal{M})$ is total. Then $\mathcal{M}$ has a maximum extended recovery if and only if $e(\mathcal{M})$ has a maximum recovery. Moreover, $\mathcal{M}'$*

is a maximum extended recovery of $\mathcal{M}$ if and only if $e(\mathcal{M}')$ is a maximum recovery of $e(\mathcal{M})$.

PROOF. We first introduce some notation to simplify the exposition. Let I_1 and I_2 be instances of the same schema $\mathbf{R}$ with values in $\mathbf{C} \cup \mathbf{N}$. Recall that a *homomorphism* from I_1 to I_2 is a function $h : \mathrm{dom}(I_1) \to \mathrm{dom}(I_2)$ such that, for every constant value $a \in \mathbf{C}$, it holds that $h(a) = a$, and for every $R \in \mathbf{R}$ and every tuple $(a_1, \ldots, a_k) \in R^{I_1}$, it holds $(h(a_1), \ldots, h(a_k)) \in R^{I_2}$. Consider a binary relation $\to$ defined as follows:

$$\to \;=\; \{(I_1, I_2) \mid \text{there exists a homomorphism from } I_1 \text{ to } I_2\}.$$

Fagin et al. (2009) introduced relation $\to$ to simplify the definition of the extended semantics of a mapping. In fact, given a mapping $\mathcal{M}$, we have that

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

Notice that the relation $\to$ is *idempotent*, that is, it holds that $(\to \circ \to) = \to$. In particular, we have that

$$\to \circ\, e(\mathcal{M}) \;=\; e(\mathcal{M}), \tag{3.1}$$

$$e(\mathcal{M}) \circ \to \;=\; e(\mathcal{M}). \tag{3.2}$$

Thus, if I_1, I_2, J are instances such that $(I_1, I_2) \in \to$ and $(I_2, J) \in e(\mathcal{M})$, then $(I_1, J) \in e(\mathcal{M})$. Hence, if $(I_1, I_2) \in \to$, then it holds that $\mathrm{Sol}_{e(\mathcal{M})}(I_2) \subseteq \mathrm{Sol}_{e(\mathcal{M})}(I_1)$. We use this property in this proof.

Before proving the theorem, we make an additional observation. Let $\mathcal{M}$ be the mapping in the statement of the theorem. Recall that we are assuming that $e(\mathcal{M})$ is a total mapping, thus from Proposition 3.1.6 we have that $\mathcal{M}'$ is a maximum recovery of $e(\mathcal{M})$ if and only if $\mathcal{M}'$ is a recovery of $e(\mathcal{M})$ and for every $(I_1, I_2) \in e(\mathcal{M}) \circ \mathcal{M}'$, it holds that $\mathrm{Sol}_{e(\mathcal{M})}(I_2) \subseteq \mathrm{Sol}_{e(\mathcal{M})}(I_1)$. We extensively use this property in this proof.

Now we are ready to prove the theorem. Let $\mathcal{M}$ be a mapping from a schema $\mathbf{S}$ to a schema $\mathbf{T}$, and assume that source instances are composed by null and constant values. We

first show that $e(\mathcal{M})$ has a maximum recovery if and only if $\mathcal{M}$ has a maximum extended recovery.

We show now that if $e(\mathcal{M})$ has a maximum recovery then $\mathcal{M}$ has a maximum extended recovery. Thus, assume that $e(\mathcal{M})$ has a maximum recovery, and let $\mathcal{M}'$ be a maximum recovery of $e(\mathcal{M})$. We show next that $\mathcal{M}'$ is also a maximum extended recovery of $\mathcal{M}$. Since $\mathcal{M}'$ is a recovery of $e(\mathcal{M})$, we have that $(I, I) \in e(\mathcal{M}) \circ \mathcal{M}'$ for every instance I of $\mathbf{S}$. Moreover, from (3.2) we have that $e(\mathcal{M}) \circ \mathcal{M}' = e(\mathcal{M}) \circ \rightarrow \circ \mathcal{M}'$ and, thus, $(I, I) \in e(\mathcal{M}) \circ \rightarrow \circ \mathcal{M}'$ for every instance I of $\mathbf{S}$. Thus, given that $(I, I) \in \rightarrow$ for every instance I of $\mathbf{S}$, we obtain that $(I, I) \in e(\mathcal{M}) \circ \rightarrow \circ \mathcal{M}' \circ \rightarrow = e(\mathcal{M}) \circ e(\mathcal{M}')$ for every instance I of $\mathbf{S}$, which implies that $e(\mathcal{M}')$ is a extended recovery of $e(\mathcal{M})$.

Now, let $\mathcal{M}''$ be an extended recovery of $\mathcal{M}$. Then, as above, we obtain that $(I, I) \in e(\mathcal{M}) \circ e(\mathcal{M}'')$ for every instance I of $\mathbf{S}$. Thus, we have that $e(\mathcal{M}'')$ is a recovery of $e(\mathcal{M})$. Recall that $\mathcal{M}'$ is a maximum recovery of $e(\mathcal{M})$ and, hence, we have that $e(\mathcal{M}) \circ \mathcal{M}' \subseteq e(\mathcal{M}) \circ e(\mathcal{M}'')$, which implies that $e(\mathcal{M}) \circ \mathcal{M}' \circ \rightarrow \subseteq e(\mathcal{M}) \circ e(\mathcal{M}'') \circ \rightarrow$. Therefore, given that $e(\mathcal{M}) = e(\mathcal{M}) \circ \rightarrow$ and $e(\mathcal{M}'') \circ \rightarrow = e(\mathcal{M}'')$ by (3.2), we have that $e(\mathcal{M}) \circ \rightarrow \circ \mathcal{M}' \circ \rightarrow \subseteq e(\mathcal{M}) \circ e(\mathcal{M}'')$, which implies that $e(\mathcal{M}) \circ e(\mathcal{M}') \subseteq e(\mathcal{M}) \circ e(\mathcal{M}'')$. Thus, we have shown that $\mathcal{M}'$ is an extended recovery of $\mathcal{M}$, and that for every other extended recovery $\mathcal{M}''$ of $\mathcal{M}$, it holds that $e(\mathcal{M}) \circ e(\mathcal{M}') \subseteq e(\mathcal{M}) \circ e(\mathcal{M}'')$, which implies that $\mathcal{M}'$ is a maximum extended recovery of $\mathcal{M}$, and thus, $\mathcal{M}$ has a maximum extended recovery.

We prove now the opposite direction, that is, we prove that if $\mathcal{M}$ has a maximum extended recovery then $e(\mathcal{M})$ has a maximum recovery. Thus, assume that $\mathcal{M}$ has a maximum extended recovery, and let $\mathcal{M}'$ be a maximum extended recovery of $\mathcal{M}$. Next we show that $e(\mathcal{M}')$ is a maximum recovery of $e(\mathcal{M})$. Given that $\mathcal{M}'$ is an extended recovery of $\mathcal{M}$, we have that $(I, I) \in e(\mathcal{M}) \circ e(\mathcal{M}')$ for every instance I of $\mathbf{S}$, which implies that $e(\mathcal{M}')$ is a recovery of $e(\mathcal{M})$. Thus, by Proposition 3.1.6, to prove that $e(\mathcal{M}')$ is a maximum recovery of $e(\mathcal{M})$, it is enough to show that $\mathrm{Sol}_{e(\mathcal{M})}(I_2) \subseteq \mathrm{Sol}_{e(\mathcal{M})}(I_1)$ for every $(I_1, I_2) \in e(\mathcal{M}) \circ e(\mathcal{M}')$. Let $(I_1, I_2) \in e(\mathcal{M}) \circ e(\mathcal{M}')$. To prove that $\mathrm{Sol}_{e(\mathcal{M})}(I_2) \subseteq$

$\mathrm{Sol}_{e(\mathcal{M})}(I_1)$, we make use of the following mapping $\mathcal{M}^\star$ from $\mathbf{T}$ to $\mathbf{S}$:

$$\begin{aligned}\mathcal{M}^\star \;=\; & \{(J, I) \mid I \text{ is an instance of } \mathbf{S} \text{ and } (I_1, J) \notin e(\mathcal{M})\} \;\cup \\ & \{(J, I) \mid (I_1, J) \in e(\mathcal{M}) \text{ and } \mathrm{Sol}_{e(\mathcal{M})}(I) \subseteq \mathrm{Sol}_{e(\mathcal{M})}(I_1)\}.\end{aligned}$$

We show first that $\mathcal{M}^\star$ is an extended recovery of $\mathcal{M}$, that is, we show that for every instance I of $\mathbf{S}$, it holds that $(I, I) \in e(\mathcal{M}) \circ e(\mathcal{M}^\star)$. First, assume that $\mathrm{Sol}_{e(\mathcal{M})}(I) \subseteq \mathrm{Sol}_{e(\mathcal{M})}(I_1)$, and consider an arbitrary instance $J^\star$ such that $(I, J^\star) \in e(\mathcal{M})$. Notice that $(I_1, J^\star) \in e(\mathcal{M})$ since $\mathrm{Sol}_{e(\mathcal{M})}(I) \subseteq \mathrm{Sol}_{e(\mathcal{M})}(I_1)$. Thus, we have that $(J^\star, I) \in \mathcal{M}^\star$ and, hence, $(J^\star, I) \in e(\mathcal{M}^\star)$. Therefore, given that $(I, J^\star) \in e(\mathcal{M})$ and $(J^\star, I) \in e(\mathcal{M}^\star)$, we conclude that $(I, I) \in e(\mathcal{M}) \circ e(\mathcal{M}^\star)$. Second, assume that $\mathrm{Sol}_{e(\mathcal{M})}(I) \not\subseteq \mathrm{Sol}_{e(\mathcal{M})}(I_1)$. Then there exists an instance $J^\star$ such that $(I, J^\star) \in e(\mathcal{M})$ and $(I_1, J^\star) \notin e(\mathcal{M})$. By definition of $\mathcal{M}^\star$, we have that $(J^\star, I) \in \mathcal{M}^\star$ and, thus, $(J^\star, I) \in e(\mathcal{M}^\star)$. Thus, we also conclude that $(I, I) \in e(\mathcal{M}) \circ e(\mathcal{M}^\star)$ in this case.

We are now ready to prove that for every $(I_1, I_2) \in e(\mathcal{M}) \circ e(\mathcal{M}')$, it holds that $\mathrm{Sol}_{e(\mathcal{M})}(I_2) \subseteq \mathrm{Sol}_{e(\mathcal{M})}(I_1)$. Let $(I_1, I_2) \in e(\mathcal{M}) \circ e(\mathcal{M}')$. Given that $\mathcal{M}'$ is a maximum extended recovery of $\mathcal{M}$ and $\mathcal{M}^\star$ is an extended recovery of $\mathcal{M}$, we have that $e(\mathcal{M}) \circ e(\mathcal{M}') \subseteq e(\mathcal{M}) \circ e(\mathcal{M}^\star)$ and, therefore, $(I_1, I_2) \in e(\mathcal{M}) \circ e(\mathcal{M}^\star)$. Thus, given that $e(\mathcal{M}) \circ e(\mathcal{M}^\star) = e(\mathcal{M}) \circ \mathcal{M}^\star \circ \rightarrow$ by (3.2), we conclude that there exist instances J of $\mathbf{T}$ and I_2' of $\mathbf{S}$ such that $(I_1, J) \in e(\mathcal{M})$, $(J, I_2') \in \mathcal{M}^\star$ and $(I_2', I_2) \in \rightarrow$. Hence, by definition of $\mathcal{M}^\star$, we have that $\mathrm{Sol}_{e(\mathcal{M})}(I_2') \subseteq \mathrm{Sol}_{e(\mathcal{M})}(I_1)$ (since $(I_1, J) \in e(\mathcal{M})$). But we also have that $\mathrm{Sol}_{e(\mathcal{M})}(I_2) \subseteq \mathrm{Sol}_{e(\mathcal{M})}(I_2')$ since $(I_2', I_2) \in \rightarrow$, and, therefore, we conclude that $\mathrm{Sol}_{e(\mathcal{M})}(I_2) \subseteq \mathrm{Sol}_{e(\mathcal{M})}(I_1)$, which was to be shown.

Up to this point, we have shown that $e(\mathcal{M})$ has a maximum recovery if and only if $\mathcal{M}$ has a maximum extended recovery. In fact, from the preceding proof, we conclude that:

(a) if $e(\mathcal{M})$ has a maximum recovery $\mathcal{M}'$, then $\mathcal{M}'$ is a maximum extended recovery of $\mathcal{M}$, and

(b) if $\mathcal{M}$ has a maximum extended recovery $\mathcal{M}'$, then $e(\mathcal{M}')$ is a maximum recovery of $e(\mathcal{M})$.

Next we prove the second part of the theorem, that is, we prove that a mapping $\mathcal{M}'$ is a maximum extended recovery of $\mathcal{M}$ if and only if $e(\mathcal{M}')$ is a maximum recovery of $e(\mathcal{M})$. It should be noticed that the "only if" direction corresponds to property (b) above and, thus, we only need to show that if $e(\mathcal{M}')$ is a maximum recovery of $e(\mathcal{M})$, then $\mathcal{M}'$ is a maximum extended recovery of $\mathcal{M}$.

Assume that $e(\mathcal{M}')$ is a maximum recovery of $e(\mathcal{M})$. Then we have that $e(\mathcal{M}')$ is a recovery of $e(\mathcal{M})$ and, thus, $\mathcal{M}'$ is an extended recovery of $\mathcal{M}$. Now let $\mathcal{M}''$ be an extended recovery of $\mathcal{M}$. Then we have that $e(\mathcal{M}'')$ is a recovery of $e(\mathcal{M})$ and, hence, $e(\mathcal{M}) \circ e(\mathcal{M}') \subseteq e(\mathcal{M}) \circ e(\mathcal{M}'')$ since $e(\mathcal{M}')$ is a maximum recovery of $e(\mathcal{M})$. Therefore, we conclude that $\mathcal{M}'$ is an extended recovery of $\mathcal{M}$, and for every extended recovery $\mathcal{M}''$ of $\mathcal{M}$, it holds that $e(\mathcal{M}) \circ e(\mathcal{M}') \subseteq e(\mathcal{M}) \circ e(\mathcal{M}'')$, which means that $\mathcal{M}'$ is a maximum extended recovery of $\mathcal{M}$. This completes the proof of the proposition. □

It was proved by Fagin et al. (2009) that every mapping specified by a set of st-tgds and considering nulls in the source has a maximum extended recovery. It should be noticed that this result is also implied by Theorems 3.1.14 and 3.1.16. Finally, another conclusion that can be drawn from the above result is that all the machinery presented in this section for the notion of maximum recovery, in particular, Proposition 3.1.6 and Theorem 3.1.8, can be applied over maximum extended recoveries, and the extended semantics for mappings, thus giving a new insight about inverses of mappings with null values in the source.

3.2. An Application of Maximum Recoveries: Schema Evolution

One of the main reasons for the study of the issues of composing and inverting schema mappings is to solve the schema evolution problem (Bernstein, 2003; Bernstein & Melnik, 2007; Kolaitis, 2005; Fagin et al., 2011). Two main scenarios have been identified for this problem, which are shown in Figure 3.1. In scenario (a), a mapping $\mathcal{M}$ from a schema $\mathbf{S}$ to a schema $\mathbf{T}$ has already been constructed, and it has been decided that target schema $\mathbf{T}$ will be replaced by a new schema $\mathbf{U}$. In particular, the relationship between schemas $\mathbf{T}$ and $\mathbf{U}$ has been given through a mapping $\mathcal{M}'$. The schema evolution problem is then to provide a

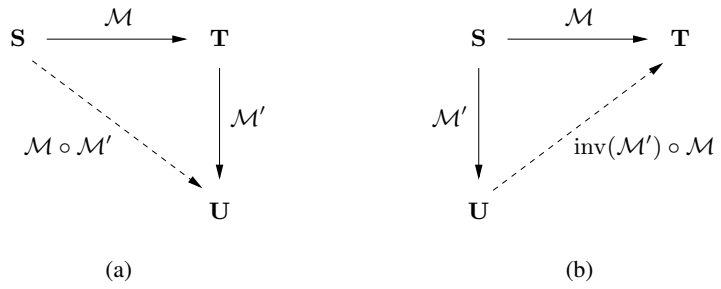


FIGURE 3.1. The schema evolution problem.

mapping from $\mathbf{S}$ to $\mathbf{U}$, considering the metadata provided by $\mathcal{M}$ and $\mathcal{M}'$. As pointed out by Kolaitis (2005), the process of constructing a schema mapping is time consuming and, thus, one would like to solve the schema evolution problem by automatically reusing the metadata that is given. In scenario (a), it is possible to do this by using the composition operator (Fagin, Kolaitis, Popa, & Tan, 2005; Kolaitis, 2005); the mapping $\mathcal{M} \circ \mathcal{M}'$ correctly represents the relationship between schemas $\mathbf{S}$ and $\mathbf{U}$.

Scenario (b) in Figure 3.1 is similar to scenario (a), but in this case it has been decided to replace source schema $\mathbf{S}$ by $\mathbf{U}$. As in (a), the relationship between $\mathbf{S}$ and $\mathbf{U}$ is given by a mapping, that is again called $\mathcal{M}'$. The natural question at this point is whether a combination of mappings $\mathcal{M}$ and $\mathcal{M}'$ could be used to provide the *right* mapping, or at least a *good* mapping, from $\mathbf{U}$ to $\mathbf{T}$ according to the metadata provided by $\mathcal{M}$ and $\mathcal{M}'$. It has been argued that the combination of the inverse and composition operators can be used for this purpose, and the mapping $\mathrm{inv}(\mathcal{M}') \circ \mathcal{M}$ has been proposed as a solution for the schema evolution problem (Fagin, 2007), where $\mathrm{inv}(\mathcal{M}')$ represents an inverse of mapping $\mathcal{M}'$. But, unfortunately, it has not been formally studied to what extend $\mathrm{inv}(\mathcal{M}') \circ \mathcal{M}$ is the right solution for the schema evolution problem. In this section, we address this issue for the common case of mappings given by st-tgds, and show that if $\mathrm{inv}(\mathcal{M}')$ is the maximum recovery of $\mathcal{M}'$, then $\mathrm{inv}(\mathcal{M}') \circ \mathcal{M}$ is the best solution in a precise sense for the schema evolution problem.

For the rest of this section, let $\mathbf{S}$ be a source schema, $\mathbf{T}$, $\mathbf{U}$ target schemas, $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ and $\mathcal{M}' = (\mathbf{S}, \mathbf{U}, \Sigma')$, where Σ and Σ' are sets of st-tgds. If $\mathcal{M}^\star$ is a mapping from $\mathbf{U}$ to $\mathbf{T}$, what properties should it satisfy in order to be considered a good solution

for the schema evolution problem? Or, in other words, what properties should $\mathcal{M}^\star$ satisfy to be considered a good representation of the metadata provide by $\mathcal{M}$ and $\mathcal{M}'$? Assume that I is an instance of $\mathbf{S}$, and let J be a solution for I under $\mathcal{M}'$. If J properly represents the information in I, then one would consider $\mathcal{M}^\star$ a good representation of the metadata provided by $\mathcal{M}$ and $\mathcal{M}'$ if the space of solutions for I under $\mathcal{M}$ is the same as the space of solutions for J under $\mathcal{M}^\star$, that is, $\mathrm{Sol}_{\mathcal{M}}(I) = \mathrm{Sol}_{\mathcal{M}^\star}(J)$. Or, at least, one would expect that none of the instances in $\mathrm{Sol}_{\mathcal{M}}(I)$ is ruled out by $\mathcal{M}^\star$ when mapping data from J, that is, $\mathrm{Sol}_{\mathcal{M}}(I) \subseteq \mathrm{Sol}_{\mathcal{M}^\star}(J)$. In this section, we use this simple criterion to compare different solutions for the schema evolution problem.

To formalize the criterion described above, for every instance I of $\mathbf{S}$, we first need to choose a particular solution J under $\mathcal{M}'$. A natural candidate for this is the canonical universal solution $\mathrm{chase}_{\Sigma'}(I)$ which has been identified in the database literature as a solution with several desirable properties (Fagin, Kolaitis, Miller, & Popa, 2005). Thus, the criterion mentioned above is formalized as follows: A mapping $\mathcal{M}^\star$ from $\mathbf{U}$ to $\mathbf{T}$ is said to be a *solution for the schema evolution problem for $\mathcal{M}$ and $\mathcal{M}'$* if for every instance I of $\mathbf{S}$, it holds that:

$$\mathrm{Sol}_{\mathcal{M}}(I) \quad \subseteq \quad \mathrm{Sol}_{\mathcal{M}^\star}(\mathrm{chase}_{\Sigma'}(I)).$$

The previous criterion also suggests a way to compare alternative solutions for the schema evolution problem; the closer the space of solutions $\mathrm{Sol}_{\mathcal{M}^\star}(\mathrm{chase}_{\Sigma'}(I))$ is to $\mathrm{Sol}_{\mathcal{M}}(I)$ the better is $\mathcal{M}^\star$ as a solution for the schema evolution problem. In the following proposition, we show that under this criterion, the notion of maximum recovery can be used to obtain the best solution for the schema evolution problem.

PROPOSITION 3.2.1. *Let $\mathbf{S}$ be a source schema, $\mathbf{T}$, $\mathbf{U}$ target schemas, $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ and $\mathcal{M}' = (\mathbf{S}, \mathbf{U}, \Sigma')$, where Σ and Σ' are sets of st-tgds. Then there exists a maximum recovery $\mathcal{N}$ of $\mathcal{M}'$ such that:*

(1) *$\mathcal{N} \circ \mathcal{M}$ is a solution for the schema evolution problem for $\mathcal{M}$ and $\mathcal{M}'$, and*

(2) *for every solution $\mathcal{M}^\star$ for the schema evolution problem for $\mathcal{M}$ and $\mathcal{M}'$, and for every instance I of* $\mathbf{S}$*, it holds that:*

$$\mathrm{Sol}_{\mathcal{N}\circ\mathcal{M}}(\mathrm{chase}_{\Sigma'}(I)) \;\subseteq\; \mathrm{Sol}_{\mathcal{M}^\star}(\mathrm{chase}_{\Sigma'}(I)).$$

PROOF. Let $\mathcal{N} = \{(\mathrm{chase}_{\Sigma'}(I), I) \mid I \in \mathrm{Inst}(\mathbf{S})\}$. It is straightforward to see that $\mathcal{N}$ is a recovery of $\mathcal{M}'$. Moreover, given that $\mathrm{dom}(\mathcal{M}') = \mathbf{S}$, we have that $\mathcal{N}$ is a reduced recovery of $\mathcal{M}'$. Assume that $(I_1, I_2) \in \mathcal{M}' \circ \mathcal{N}$. Thus, we have that $\mathrm{chase}_{\Sigma'}(I_2) \in \mathrm{Sol}_{\mathcal{M}'}(I_1)$. Now, we know that $\mathrm{chase}_{\Sigma'}(I_2)$ is a universal solution for I_2 under $\mathcal{M}'$ (Fagin, Kolaitis, Miller, & Popa, 2005; Arenas et al., 2004). Moreover, since $\mathcal{M}'$ is specified by st-tgds we know that $\mathcal{M}'$ is closed under target homomorphisms. From these two properties and the fact that $\mathrm{chase}_{\Sigma'}(I_2) \in \mathrm{Sol}_{\mathcal{M}'}(I_1)$, we obtain that for every $J \in \mathrm{Sol}_{\mathcal{M}'}(I_2)$ it holds that $J \in \mathrm{Sol}_{\mathcal{M}'}(I_1)$. We have shown that for every $(I_1, I_2) \in \mathcal{M}' \circ \mathcal{N}$ we have that $\mathrm{Sol}_{\mathcal{M}'}(I_2) \subseteq \mathrm{Sol}_{\mathcal{M}'}(I_1)$ which by Proposition 3.1.6 implies that $\mathcal{N}$ is a maximum recovery of $\mathcal{M}'$. Next we show that $\mathcal{N}$ satisfies the two conditions of the proposition.

(1) For every instance I of $\mathbf{S}$, we have that $(\mathrm{chase}_{\Sigma'}(I), I) \in \mathcal{N}$ and, thus, we conclude that $\mathrm{Sol}_{\mathcal{M}}(I) \subseteq \mathrm{Sol}_{\mathcal{N}\circ\mathcal{M}}(\mathrm{chase}_{\Sigma'}(I))$. Thus, we have that $\mathcal{N} \circ \mathcal{M}$ is a solution for the schema evolution problem for $\mathcal{M}$ and $\mathcal{M}'$.

(2) Let $\mathcal{M}^\star$ be a solution for the schema evolution problem for $\mathcal{M}$ and $\mathcal{M}'$, and I an instance of $\mathbf{S}$. We need to show that $\mathrm{Sol}_{\mathcal{N}\circ\mathcal{M}}(\mathrm{chase}_{\Sigma'}(I)) \subseteq \mathrm{Sol}_{\mathcal{M}^\star}(\mathrm{chase}_{\Sigma'}(I))$.

Assume that $J \in \mathrm{Sol}_{\mathcal{N}\circ\mathcal{M}}(\mathrm{chase}_{\Sigma'}(I))$. Then there exists an instance I' of $\mathbf{S}$ such that $(\mathrm{chase}_{\Sigma'}(I), I') \in \mathcal{N}$ and $(I', J) \in \mathcal{M}$. Given that $\mathcal{M}^\star$ is a solution for the schema evolution problem for $\mathcal{M}$ and $\mathcal{M}'$, we have that $\mathrm{Sol}_{\mathcal{M}}(I') \subseteq \mathrm{Sol}_{\mathcal{M}^\star}(\mathrm{chase}_{\Sigma'}(I'))$ and, hence, $J \in \mathrm{Sol}_{\mathcal{M}^\star}(\mathrm{chase}_{\Sigma'}(I'))$. But, by definition of $\mathcal{N}$, we have that $\mathrm{chase}_{\Sigma'}(I') = \mathrm{chase}_{\Sigma'}(I)$ since $(\mathrm{chase}_{\Sigma'}(I), I') \in \mathcal{N}$. Thus, we have that $J \in \mathrm{Sol}_{\mathcal{M}^\star}(\mathrm{chase}_{\Sigma'}(I))$. This concludes the proof of the proposition. □

Notice that an *ideal* solution for the schema evolution problem for mappings $\mathcal{M}$ and $\mathcal{M}'$ is a mapping $\mathcal{M}^\star$ such that $\mathrm{Sol}_{\mathcal{M}}(I) = \mathrm{Sol}_{\mathcal{M}^\star}(\mathrm{chase}_{\Sigma'}(I))$, for every source instance I. The following corollary of Proposition 3.2.1 shows that if such a solution exists, then

one can focus on the solutions constructed by using maximum recoveries in order to find an ideal solution.

COROLLARY 3.2.2. *Let* $\mathbf{S}$ *be a source schema,* $\mathbf{T}$*,* $\mathbf{U}$ *target schemas,* $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ *and* $\mathcal{M}' = (\mathbf{S}, \mathbf{U}, \Sigma')$*, with* Σ*,* Σ' *sets of st-tgds. If there exists an ideal solution for the schema evolution problem for* $\mathcal{M}$ *and* $\mathcal{M}'$*, then there exists a maximum recovery* $\mathcal{N}$ *of* $\mathcal{M}'$ *such that* $\mathcal{N} \circ \mathcal{M}$ *is an ideal solution for the schema evolution problem for* $\mathcal{M}$ *and* $\mathcal{M}'$*.*

From Proposition 3.2.1 and the previous corollary, we conclude that the combination of the maximum recovery and the composition operator is appropriate to provide a solution for the schema evolution problem shown in Figure 3.1 (b). We also note that maximum recovery can be replaced neither by Fagin-inverse nor by quasi-inverse in Proposition 3.2.1, as it is known that even for full st-tgds, Fagin-inverses and quasi-inverses are not guaranteed to exist (Fagin, 2007; Fagin, Kolaitis, Popa, & Tan, 2008).

3.3. Computing Maximum Recoveries

Up to this point we know that every st-mapping specified by a set of FO-TO-CQ dependencies has a maximum recovery, but we have not said anything about the language needed to express it. In this section, we show that every st-mapping specified by a set of FO-TO-CQ dependencies has a maximum recovery specified by a set of CQ^{C}-TO-FO dependencies. In fact, we provide an algorithm that computes maximum recoveries for st-mappings specified by FO-TO-CQ dependencies. Our algorithm runs in exponential time when mappings are given by sets of FO-TO-CQ dependencies, and can be adapted to run in quadratic time when the input is a mapping specified by a set of full FO-TO-CQ dependencies.

Before presenting our algorithm we introduce some basic terminology, and we also present some results that are important in the formulation of the algorithm

Our algorithm is based on *query rewriting*. Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ be an st-mapping such that Σ is a set of FO-TO-CQ dependencies, and let Q be a query over schema $\mathbf{T}$. Recall that a Q' is said to be a rewriting of Q over the source if Q' is a query over $\mathbf{S}$ such that

for every $I \in \mathrm{Inst}(\mathbf{S})$, it holds that $Q'(I) = \mathrm{certain}_{\mathcal{M}}(Q, I)$. That is, to obtain the set of certain answers of Q over I under $\mathcal{M}$, we just have to evaluate its rewriting Q' over instance I (see Section 2.4).

The computation of a rewriting of a conjunctive query is a basic step in the algorithm presented in this section. This problem has been extensively studied in the database area (Levy, Mendelzon, Sagiv, & Srivastava, 1995; Abiteboul & Duschka, 1998) and, in particular, in the data integration context (Halevy, 2000, 2001; Lenzerini, 2002). In particular, the class of CQ-TO-CQ dependencies corresponds to the class of GLAV mappings in the data integration context (Lenzerini, 2002), and, as such, the techniques developed to solved the query rewriting problem for GLAV mappings can be reused in our context.

For the sake of completeness, in this dissertation we present an exponential-time algorithm that given a mapping $\mathcal{M}$ specified by a set of FO-TO-CQ st-dependencies and a conjunctive query Q over the target schema, produces a rewriting of Q over the source of $\mathcal{M}$. This algorithm is presented in Appendix A. For computing maximum recoveries we only need the following lemma (the proof of the lemma can be found in Appendix A.1.1).

LEMMA 3.3.1. *There exists an algorithm* QUERYREWRITING *that given an st-mapping* $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$*, with* Σ *a set of* FO-TO-CQ *dependencies, and a conjunctive query* Q *over schema* $\mathbf{T}$*, computes a domain-independent* FO *query* Q' *that is a rewriting of* Q *over the source. The algorithm runs in exponential time and its output is of exponential size in the size of* Σ *and* Q*.*

Another notion that would be used in the proof of correctness of our algorithm for computing maximum recoveries is the notion of *chase* that we introduced in Section 2.4.

3.3.1. Computing maximum recoveries in the general case

In this section, we propose an algorithm that given a mapping $\mathcal{M}$ specified by a set of FO-TO-CQ dependencies, returns a maximum recovery of $\mathcal{M}$.

It is known that the simple process of "reversing the arrows" of source-to-target dependencies does not necessarily produce inverses as conclusions of different dependencies

may be related (Fagin, 2007); a conclusion of a dependency may be implied by the conclusions of other dependencies. The algorithm presented in this section first *searches* for these relations among conclusions of dependencies, and then suitably *composes* the premises of related dependencies and "reverses the arrows" to obtain a maximum recovery. Let us give some intuition with an example. Consider a mapping $\mathcal{M}$ specified by the FO-TO-CQ dependencies:

$$\varphi_1(x_1, x_2) \rightarrow \exists v(P(x_1, v) \wedge R(v, x_2)), \tag{3.3}$$

$$\varphi_2(y_1, y_2) \rightarrow P(y_1, y_2), \tag{3.4}$$

$$\varphi_3(z_1, z_2) \rightarrow R(z_1, z_2), \tag{3.5}$$

where φ_1, φ_2, and φ_3 are arbitrary FO formulas with two free variables. In this case, the conjunction of the conclusions of (3.4) and (3.5) implies the conclusion of (3.3) when y_2 is equal to z_1 and both are existentially quantified. The idea behind the algorithm is to make explicit these types of relationships. For instance, we could replace (3.3) by the dependency:

$$\varphi_1(u_1, u_2) \vee \exists y_2 \exists z_1 \big(\varphi_2(u_1, y_2) \wedge \varphi_3(z_1, u_2) \wedge y_2 = z_1\big) \rightarrow \exists v(P(u_1, v) \wedge R(v, u_2)). \tag{3.6}$$

It can be proved that the set of dependencies obtained by replacing formula (3.3) by (3.6) is logically equivalent to the initial set of dependencies. After making explicit these types of relationships between dependencies, the algorithm "reverses the arrows" to obtain a maximum recovery. When "reversing the arrows", we also need to impose an additional constraint. In the above example, given that (3.3) is a non-full dependency, when reversing (3.4) the algorithm needs to force variable y_2 in $P(y_1, y_2)$ to take values only from the set $\mathbf{C}$, that is, we have to use dependency $P(y_1, y_2) \wedge \mathbf{C}(y_2) \rightarrow \varphi_2(y_1, y_2)$ instead of $P(y_1, y_2) \rightarrow \varphi_2(y_1, y_2)$. This is because, given a source instance I such that $I \models \varphi_1(a, b)$, dependency (3.3) could be satisfied by including a tuple of the form $P(a, n)$ in a target

instance, where $n \in \mathbf{N}$, and value n should not be passed to a source instance by a recovery (see Theorem 3.3.4 for a formal justification for the use of predicate $\mathbf{C}(\cdot)$). In fact, as a safety condition, the algorithm presented in this section uses predicate $\mathbf{C}(\cdot)$ over each variable that passes values from the target to the source. Summing up, the following set of dependencies defines a maximum recovery of the mapping $\mathcal{M}$ above:

$$\begin{aligned} P(y_1, y_2) \wedge \mathbf{C}(y_1) \wedge \mathbf{C}(y_2) &\rightarrow \varphi_2(y_1, y_2), \\ R(z_1, z_2) \wedge \mathbf{C}(z_1) \wedge \mathbf{C}(z_2) &\rightarrow \varphi_3(z_1, z_2), \\ \exists v(P(u_1, v) \wedge R(v, u_2)) \wedge \mathbf{C}(u_1) \wedge \mathbf{C}(u_2) &\rightarrow \varphi_1(u_1, u_2) \ \vee \\ & \quad \exists y_2 \exists z_1 \big(\varphi_2(u_1, y_2) \wedge \varphi_3(z_1, u_2) \wedge y_2 = z_1\big). \end{aligned}$$

The following algorithm uses a query rewriting procedure to find the types of relationships between dependencies described above. In fact, in the above example, the formula:

$$\varphi_1(u_1, u_2) \ \vee \ \exists y_2 \exists z_1 \big(\varphi_2(u_1, y_2) \wedge \varphi_3(z_1, u_2) \wedge y_2 = z_1\big), \tag{3.7}$$

that appears as the premise of (3.6), makes explicit the relationship between the conclusion $\exists v(P(u_1, v) \wedge R(v, u_2))$ of FO-TO-CQ dependency (3.3) and dependencies (3.3), (3.4) and (3.5). But not only that, it can be shown that (3.7) is a rewriting of $\exists v(P(u_1, v) \wedge R(v, u_2))$ over the source schema (according to dependencies (3.3), (3.4) and (3.5)).

In the algorithm, if $\bar{x} = (x_1, \ldots, x_k)$, then $\mathbf{C}(\bar{x})$ is a shorthand for $\mathbf{C}(x_1) \wedge \cdots \wedge \mathbf{C}(x_k)$.

Algorithm MAXIMUMRECOVERY($\mathcal{M}$)

Input: An st-mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where Σ is a set of FO-TO-CQ dependencies.

Output: A ts-mapping $\mathcal{M}' = (\mathbf{T}, \mathbf{S}, \Sigma')$, where Σ' is a set of $\text{CQ}^{\mathbf{C}}$-TO-FO dependencies and $\mathcal{M}'$ is a maximum recovery of $\mathcal{M}$.

(1) Start with Σ' as the empty set.

(2) For every dependency $\sigma \in \Sigma$ of the form $\varphi(\bar{x}) \rightarrow \exists \bar{y} \psi(\bar{x}, \bar{y})$, do the following:

(a) Let Q be the conjunctive query defined by $\exists \bar{y}\psi(\bar{x}, \bar{y})$.

(b) Use QUERYREWRITING($\mathcal{M}, Q$) to compute an FO formula $\alpha(\bar{x})$ that is a rewriting of $\exists \bar{y}\psi(\bar{x}, \bar{y})$ over the source.

(c) Add dependency $\exists \bar{y}\psi(\bar{x}, \bar{y}) \wedge \mathbf{C}(\bar{x}) \rightarrow \alpha(\bar{x})$ to Σ'.

(3) Return $\mathcal{M}' = (\mathbf{T}, \mathbf{S}, \Sigma')$. □

THEOREM 3.3.2. *Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ be an st-mapping, where Σ is a set of* FO-TO-CQ *dependencies. Then* MAXIMUMRECOVERY($\mathcal{M}$) *computes a maximum recovery of $\mathcal{M}$ in exponential time in the size of Σ, which is specified by a set of* $\mathrm{CQ}^{\mathbf{C}}$-TO-FO *dependencies.*

PROOF. From Lemma 3.3.1, it is straightforward to conclude that the algorithm runs in exponential time. Assume that $\mathcal{M}' = (\mathbf{T}, \mathbf{S}, \Sigma')$ is the output of MAXIMUMRECOVERY($\mathcal{M}$). We first show that $\mathcal{M}'$ is a recovery of $\mathcal{M}$, that is, we show that for every instance I of $\mathbf{S}$, it holds that $(I, I) \in \mathcal{M} \circ \mathcal{M}'$.

We show now that $(\mathrm{chase}_\Sigma(I), I) \in \mathcal{M}'$ and, thus, since $(I, \mathrm{chase}_\Sigma(I)) \in \mathcal{M}$, we obtain that $(I, I) \in \mathcal{M} \circ \mathcal{M}'$. Let $\sigma' \in \Sigma'$, we need to show that $(\mathrm{chase}_\Sigma(I), I) \models \sigma'$. Assume that σ' is of the form $\exists \bar{y}\psi(\bar{x}, \bar{y}) \wedge \mathbf{C}(\bar{x}) \rightarrow \alpha(\bar{x})$, and that $\bar{a}$ is a tuple of values such that $\mathrm{chase}_\Sigma(I) \models \exists \bar{y}\psi(\bar{a}, \bar{y}) \wedge \mathbf{C}(\bar{a})$. We have to show that $I \models \alpha(\bar{a})$. Now, consider the conjunctive query Q_ψ defined by formula $\exists \bar{y}\psi(\bar{x}, \bar{y})$. Since $\mathbf{C}(\bar{a})$ holds and $\mathrm{chase}_\Sigma(I) \models \exists \bar{y}\psi(\bar{a}, \bar{y})$, we obtain that $\bar{a} \in Q_\psi(\mathrm{chase}_\Sigma(I))_\downarrow$. Thus, by the properties of the chase, we know that $\bar{a} \in \mathrm{certain}_\mathcal{M}(Q_\psi, I)$. Consider now the query Q_α defined by formula $\alpha(\bar{x})$. By the definition of MAXIMUMRECOVERY, we know that Q_α is a rewriting of Q_ψ over schema $\mathbf{S}$, and then $\mathrm{certain}_\mathcal{M}(Q_\psi, I) = Q_\alpha(I)$. Thus, we have that $\bar{a} \in Q_\alpha(I)$, and then $I \models \alpha(\bar{a})$ which was to be shown.

To complete the proof, we show that if $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$, then $\emptyset \subsetneq \mathrm{Sol}_\mathcal{M}(I_2) \subseteq \mathrm{Sol}_\mathcal{M}(I_1)$. Thus, by Proposition 3.1.6 and since $\mathcal{M}'$ is a recovery of $\mathcal{M}$, we obtain that $\mathcal{M}'$ is a maximum recovery of $\mathcal{M}$. Let $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$, and $J^\star$ an instance of $\mathbf{T}$ such that $(I_1, J^\star) \in \mathcal{M}$ and $(J^\star, I_2) \in \mathcal{M}'$. Notice first that $\mathrm{dom}(\mathcal{M}) = \mathrm{Inst}(\mathbf{S})$, and then $\emptyset \subsetneq \mathrm{Sol}_\mathcal{M}(I_2)$. Therefore, we only have to prove that $\mathrm{Sol}_\mathcal{M}(I_2) \subseteq \mathrm{Sol}_\mathcal{M}(I_1)$. Let

$J \in \mathrm{Sol}_{\mathcal{M}}(I_2)$, we need to show that $J \in \mathrm{Sol}_{\mathcal{M}}(I_1)$. Let $\sigma \in \Sigma$ be a dependency of the form $\varphi(\bar{x}) \rightarrow \exists \bar{y}\psi(\bar{x}, \bar{y})$, and assume that $I_1 \models \varphi(\bar{a})$ for some tuple $\bar{a}$ of constant values. We show next that $J \models \exists \bar{y}\psi(\bar{a}, \bar{y})$. Since $I_1 \models \varphi(\bar{a})$ we know that for every $J' \in \mathrm{Sol}_{\mathcal{M}}(I_1)$, it holds that $J' \models \exists \bar{y}\psi(\bar{a}, \bar{y})$. In particular, it holds that $J^\star \models \exists \bar{y}\psi(\bar{a}, \bar{y})$. By the definition of the algorithm, we know that there exists a dependency $\exists \bar{y}\psi(\bar{x}, \bar{y}) \wedge \mathbf{C}(\bar{x}) \rightarrow \alpha(\bar{x})$ in Σ', such that $\alpha(\bar{x})$ is a rewriting of $\exists \bar{y}\psi(\bar{x}, \bar{y})$ over $\mathbf{S}$. Then since $J^\star \models \exists \bar{y}\psi(\bar{a}, \bar{y})$, $\bar{a}$ is a tuple of constant values, and $(J^\star, I_2) \models \Sigma'$, we know that $I_2 \models \alpha(\bar{a})$. Now consider the queries Q_ψ and Q_α defined by formulas $\exists \bar{y}\psi(\bar{x}, \bar{y})$ and $\alpha(\bar{x})$, respectively. Since $I_2 \models \alpha(\bar{a})$, we know that $\bar{a} \in Q_\alpha(I_2)$. Furthermore, we know that $Q_\alpha(I_2) = \mathrm{certain}_{\mathcal{M}}(Q_\psi, I_2)$, and then $\bar{a} \in \mathrm{certain}_{\mathcal{M}}(Q_\psi, I_2)$. In particular, since $J \in \mathrm{Sol}_{\mathcal{M}}(I_2)$, we know that $\bar{a} \in Q_\psi(J)$, from which we conclude that $J \models \exists \bar{y}\psi(\bar{a}, \bar{y})$. We have shown that for every $\sigma \in \Sigma$ of the form $\varphi(\bar{x}) \rightarrow \exists \bar{y}\psi(\bar{x}, \bar{y})$, if $I_1 \models \varphi(\bar{a})$ for some tuple $\bar{a}$, then $J \models \exists \bar{y}\psi(\bar{a}, \bar{y})$. Thus, we have that $(I_1, J) \models \Sigma$ and therefore $J \in \mathrm{Sol}_{\mathcal{M}}(I_1)$. This concludes the proof of the theorem. □

From Theorem 3.1.10, we have that if Σ is a Fagin-invertible (quasi-invertible) set of st-tgds, then MAXIMUMRECOVERY computes a Fagin-inverse (quasi-inverse) of Σ. Fagin, Kolaitis, Popa, and Tan (2008), proposed algorithms for computing Fagin-inverses and quasi-inverses for the case of mappings given by st-tgds. It is important to notice that our algorithm works not only for st-tgds but also for the larger class of FO-TO-CQ dependencies. For the latter class, it is not clear how to extend the algorithms from (Fagin, Kolaitis, Popa, & Tan, 2008) to produce Fagin-inverses and quasi-inverses, as the notion of generator used in these algorithms (Fagin, Kolaitis, Popa, & Tan, 2008, Definition 4.2) becomes undecidable for FO-TO-CQ dependencies.

The next lemma shows that when the input of algorithm MAXIMUMRECOVERY is a mapping $\mathcal{M}$ specified by a set of st-tgds, then its output is a maximum recovery of $\mathcal{M}$ specified by a set of $\mathrm{CQ}^{\mathbf{C}}$-TO-$\mathrm{UCQ}^{=}$ dependencies. The proof of the lemma follows directly from the proof of Lemma 3.3.1.

LEMMA 3.3.3. *Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ be an st-mapping such that Σ is a set of st-tgds, and Q a conjunctive query over schema $\mathbf{T}$. Then algorithm* QUERYREWRITING$(\mathcal{M}, Q)$ *in Lemma 3.3.1 has as output a query Q' in* $\text{UCQ}^{=}$ *that is a rewriting of Q over the source.*

Thus, if the input of our algorithm is a mapping given by a set Σ of st-tgds, it computes a maximum recovery given by a set Σ' of $\text{CQ}^{\mathbf{C}}$-TO-$\text{UCQ}^{=}$ dependencies.

3.3.2. Justification for the output of the algorithm

In this section we prove provide justification for both the size of the output of algorithm MAXIMUMRECOVERY and the mapping language used when the input is a set of st-tgds. More precisely, we have shown that when the input mapping $\mathcal{M}$ is specified by a set of st-tgds, algorithm MAXIMUMRECOVERY produces as output a mapping specified by $\text{CQ}^{\mathbf{C}}$-TO-$\text{UCQ}^{=}$ dependencies that is of size exponential with respect to the size of the input mapping. We show in this section that the output of the algorithm is *optimal* in the sense that the language used in the output of MAXIMUMRECOVERY is, in a precise sense, minimal for expressing maximum recoveries of st-tgds, and that the exponential blow-up in the size of the output cannot be avoided.

We show first that the three distinctive features of the language of $\text{CQ}^{\mathbf{C}}$-TO-$\text{UCQ}^{=}$ dependencies, namely, predicate $\mathbf{C}(\cdot)$ in the premises, disjunctions in the conclusions, and equalities in the conclusions, are needed to specify maximum recoveries of mappings given by st-tgds.

Necessity of predicate $\mathbf{C}(\cdot)$

A first question about the output of MAXIMUMRECOVERY is whether predicate $\mathbf{C}(\cdot)$ is really needed. Fagin, Kolaitis, Popa, and Tan (2008) proved that $\mathbf{C}(\cdot)$ is needed when computing quasi-inverses of st-mappings specified by st-tgds, if quasi-inverses are expressed using st-tgds with inequalities in the premises and disjunction in the conclusions. Here we show that $\mathbf{C}(\cdot)$ is needed when computing maximum recoveries for st-mappings specified by st-tgds, even if we allow the full power of FO to express maximum recoveries.

THEOREM 3.3.4. *There exists an st-mapping* $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ *specified by a set* Σ *of st-tgds that has no maximum recovery specified by a set of* FO*-sentences over* $\mathbf{S} \cup \mathbf{T}$ *not using predicate* $\mathbf{C}(\cdot)$.

PROOF. Let $\mathbf{S} = \{P(\cdot), R(\cdot)\}$, $\mathbf{T} = \{T(\cdot)\}$ and Σ be the following set of st-tgds:

$$\begin{aligned} P(x) &\rightarrow \exists y T(y), \\ R(x) &\rightarrow T(x). \end{aligned}$$

Assume that $\mathcal{M}'$ is a recovery of $\mathcal{M}$ that is specified by a set of FO-sentences over $\mathbf{S} \cup \mathbf{T}$. Next we show that $\mathcal{M}'$ is not a maximum recovery of $\mathcal{M}$.

On the contrary, assume that $\mathcal{M}'$ is a maximum recovery of $\mathcal{M}$. Let I be an instance of $\mathbf{S}$ such that $P^I = \{a\}$ and $R^I = \emptyset$, where a is an arbitrary element of $\mathbf{C}$. Since $\mathcal{M}'$ is a recovery of $\mathcal{M}$, there exists an instance J of $\mathbf{T}$ such that $(I, J) \in \mathcal{M}$ and $(J, I) \in \mathcal{M}'$. We consider two cases.

- First, assume that J mentions an element $b \in \mathbf{C}$, that is not necessarily distinct from a. Then we have that $(I', J) \in \mathcal{M}$, where I' is an instance of $\mathbf{S}$ such that $P^{I'} = \emptyset$ and $R^{I'} = \{b\}$. Thus, given that $(J, I) \in \mathcal{M}'$, we have that $(I', I) \in \mathcal{M} \circ \mathcal{M}'$, which implies that $\emptyset \subsetneq \mathrm{Sol}_{\mathcal{M}}(I) \subseteq \mathrm{Sol}_{\mathcal{M}}(I')$ by Proposition 3.1.6. Let J' be an instance of $\mathbf{T}$ defined as $T^{J'} = \{n\}$, where n is an arbitrary element of $\mathbf{N}$. We have that $(I, J') \in \mathcal{M}$ and $(I', J') \notin \mathcal{M}$, which contradicts the fact that $\mathrm{Sol}_{\mathcal{M}}(I) \subseteq \mathrm{Sol}_{\mathcal{M}}(I')$.
- Second, assume that J does not mention any element from $\mathbf{C}$. Assume that $\mathrm{dom}(J) = \{n_1, \ldots, n_k\}$, and let f be a function defined as $f(n_i) = b_i$, where each b_i is an element of $\mathbf{C}$ that is distinct from a and $b_i \neq b_j$ for $i \neq j$. Let $J^\star$ be the target instance that results from replacing every value n_i by b_i. It is easy to see that $(I, J^\star) \in \mathcal{M}$. Let g be a function with domain $\{a, n_1, \ldots, n_k\}$ defined as $g(a) = a$ and $g(n_i) = f(n_i)$. We have that g is an isomorphism from (J, I)

to $(J^\star, I)$ when we consider these instances as structures over $\mathbf{S} \cup \mathbf{T}$[1]. Thus, given that $\mathcal{M}'$ is specified by a set of FO-sentences over $\mathbf{S} \cup \mathbf{T}$, we conclude that $(J^\star, I) \in \mathcal{M}'$. Therefore, there exists an instance $J^\star$ of $\mathbf{T}$ such that $(I, J^\star) \in \mathcal{M}$, $(J^\star, I) \in \mathcal{M}'$ and $J^\star$ mentions elements of $\mathbf{C}$. This leads to a contradiction, as we show in the previous case. This concludes the proof of the theorem.

□

As a corollary we obtain our desired result.

COROLLARY 3.3.5. *There exists an st-mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ specified by a set Σ of st-tgds that has no maximum recovery specified by a set of* CQ-TO-UCQ$^=$ *dependencies.*

Necessity of disjunctions

The following result shows that disjunctions in the conclusion of dependencies are strictly needed to specify maximum recoveries of mappings given by st-tgds.

THEOREM 3.3.6. *There exists an st-mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ specified by a set of st-tgds that has no maximum recovery specified by a set of* CQ$^\mathbf{C}$-TO-CQ$^=$ *dependencies.*

PROOF. Let $\mathbf{S} = \{A(\cdot), B(\cdot)\}$, $\mathbf{T} = \{P(\cdot)\}$ and Σ be the following set of st-tgds:

$$\begin{aligned} A(x) &\rightarrow P(x), \\ B(x) &\rightarrow P(x). \end{aligned}$$

To obtain a contradiction assume that $\mathcal{M}'$ is a maximum recovery of $\mathcal{M}$ that is specified by a set Σ' of CQ$^\mathbf{C}$-TO-CQ$^=$ dependencies. We first make a simple observation about the equalities in Σ'. Every dependency in Σ' is of the form $\varphi(\bar{x}) \rightarrow \exists \bar{y}(\psi(\bar{x}, \bar{y}) \wedge \theta(\bar{x}, \bar{y}))$, where $\varphi(\bar{x})$ is a query in CQ$^\mathbf{C}$ over $\mathbf{T}$, $\psi(\bar{x}, \bar{y})$ is a conjunction of relational atoms over $\mathbf{S}$, and $\theta(\bar{x}, \bar{y})$ is a conjunction of equalities over variables in $(\bar{x}, \bar{y})$. It is not difficult to see that we can assume that the equalities in $\theta(\bar{x}, \bar{y})$ are all of the form $x = x'$ with x and x'

[1]Notice that if we consider (J, I) and $(J^\star, I)$ as structures over $\mathbf{S} \cup \mathbf{T} \cup \{\mathbf{C}(\cdot)\}$, then g is not an isomorphism from (J, I) to $(J^\star, I)$.

variables in $\bar{x}$. To see this, notice that an equality of the form $y = y'$ with y and y' variables in $\bar{y}$ can be eliminated by replacing every occurrence of y in $\psi(\bar{x}, \bar{y})$ by y'. Similarly, an equality of the form $x = y$ with x a variable in $\bar{x}$ and y a variable in $\bar{y}$ can be eliminated by replacing every occurrence of y in $\psi(\bar{x}, \bar{y})$ by x. Thus, from now on we assume that dependencies in Σ' are of the form $\varphi(\bar{x}) \rightarrow \exists \bar{y} \psi(\bar{x}, \bar{y}) \wedge \theta(\bar{x})$, with $\theta(\bar{x})$ a conjuntion of equalities over the variables in $\bar{x}$.

To continue with the proof, let a be an element in $\mathbf{C}$ and consider the instance I_1 of $\mathbf{S}$ such that $A^{I_1} = \{a\}$ and $B^{I_1} = \emptyset$, and the instance I_2 of $\mathbf{S}$ such that $A^{I_2} = \emptyset$ and $B^{I_2} = \{a\}$. Let J be the instance of $\mathbf{T}$ such that $P^J = \{a\}$.

We show now that $\mathrm{Sol}_{\mathcal{M} \circ \mathcal{M}'}(I_1) = \mathrm{Sol}_{\mathcal{M}'}(J)$. Since $J \in \mathrm{Sol}_{\mathcal{M}}(I_1)$ then it is straightforward that $\mathrm{Sol}_{\mathcal{M}'}(J) \subseteq \mathrm{Sol}_{\mathcal{M} \circ \mathcal{M}'}(I_1)$. To show the opposite containment, let $K \in \mathrm{Sol}_{\mathcal{M} \circ \mathcal{M}'}(I_1)$. Then there exists an instance $J' \in \mathrm{Sol}_{\mathcal{M}}(I_1)$ such that $K \in \mathrm{Sol}_{\mathcal{M}'}(J')$. We prove now that $K \in \mathrm{Sol}_{\mathcal{M}'}(J)$. First notice that by the construction of $\mathcal{M}$ and Σ it holds that $J \subseteq J'$. Now, let σ be a dependency of the form $\varphi(\bar{x}) \rightarrow \exists \bar{y} \psi(\bar{x}, \bar{y}) \wedge \theta(\bar{x})$ in Σ'. We need to prove that $(J, K) \models \sigma$. Thus, assume that $J \models \varphi(\bar{a})$ for some tuple $\bar{a}$. Since $\mathrm{dom}(J) = \{a\}$ we have that all the values in $\bar{a}$ are equal to a, and thus, $\theta(\bar{a})$ holds. Now, since $J \subseteq J'$, the formula $\varphi(\bar{x})$ is in $\mathrm{CQ}^{\mathbf{C}}$ and $\bar{a}$ is a tuple of elements in $\mathbf{C}$, we have that $J' \models \varphi(\bar{a})$. From the fact that $(J', K) \models \sigma$ we have that $K \models \exists \bar{y} \psi(\bar{a}, \bar{y})$. Thus we obtain that if $J \models \varphi(\bar{a})$ then $\theta(\bar{a})$ holds and $K \models \exists \bar{y} \psi(\bar{a}, \bar{y})$, which implies that $(J, K) \models \sigma$. We have shown that $(J, K) \models \sigma$ for every $\sigma \in \Sigma'$ which implies that $K \in \mathrm{Sol}_{\mathcal{M}'}(J)$. This concludes the proof that $\mathrm{Sol}_{\mathcal{M} \circ \mathcal{M}'}(I_1) \subseteq \mathrm{Sol}_{\mathcal{M}'}(J)$. Similarly as we proved that $\mathrm{Sol}_{\mathcal{M}'}(J) = \mathrm{Sol}_{\mathcal{M} \circ \mathcal{M}'}(I_1)$, it can be proved that $\mathrm{Sol}_{\mathcal{M}'}(J) = \mathrm{Sol}_{\mathcal{M} \circ \mathcal{M}'}(I_2)$.

Now, since $\mathcal{M}'$ is a recovery of $\mathcal{M}$ we know that $I_1 \in \mathrm{Sol}_{\mathcal{M} \circ \mathcal{M}'}(I_1)$ and $I_2 \in \mathrm{Sol}_{\mathcal{M} \circ \mathcal{M}'}(I_2)$, which implies that $I_1, I_2 \in \mathrm{Sol}_{\mathcal{M}'}(J)$. Let σ be a dependency in Σ' of the form $\varphi(\bar{x}) \rightarrow \exists \bar{y} \psi(\bar{x}, \bar{y}) \wedge \theta(\bar{x})$. It is not difficult to see that since $\varphi(\bar{x})$ is a query in $\mathrm{CQ}^{\mathbf{C}}$ over the schema $\{P(\cdot)\}$ and J only contains the fact $P(a)$ with $a \in \mathbf{C}$, then $J \models \varphi(\bar{a})$ where $\bar{a}$ is a tuple of values a. Moreover, since I_1 belongs to $\mathrm{Sol}_{\mathcal{M}'}(J)$, we have that $I_1 \models \exists \bar{y} \psi(\bar{a}, \bar{y})$ which, since $B^{I_1} = \emptyset$, implies that $\psi(\bar{a}, \bar{y})$ does not mention the relational symbol $B(\cdot)$. On the

other hand, since $I_2 \in \mathrm{Sol}_{\mathcal{M}'}(J)$ we have that $I_2 \models \exists \bar{y} \psi(\bar{a}, \bar{y})$ which cannot be the case since $\psi(\bar{a}, \bar{y})$ does not mention the relational symbol $B(\cdot)$ and $B^{I_2} = \{a\} \neq \emptyset$. This is our desired contradiction. □

Necessity of equalities

The necessity of equalities follows directly from a result by Fagin, Kolaitis, Popa, and Tan (2008). Fagin, Kolaitis, Popa, and Tan (2008, Theorem 4.15) proved that there exists a mapping specified by st-tgds that is Fagin-invertible but that does not have a Fagin-inverse specified by a set of $\mathrm{CQ}^{\mathbf{C}}$-TO-UCQ dependencies. Thus from Theorem 3.1.10 we directly obtain the following.

COROLLARY 3.3.7. *There exists an st-mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ specified by a set of st-tgds that has no maximum recovery specified by a set of* $\mathrm{CQ}^{\mathbf{C}}$-TO-UCQ *dependencies.*

The size of the output

Notice that in general, the set Σ' computed by our algorithm could be of exponential size in the size of Σ even if Σ is a set of st-tgds. The following result shows that this exponential blow-up could not be avoided.

THEOREM 3.3.8. *There exists a family of st-mappings $\{\mathcal{M}_n = (\mathbf{S}_n, \mathbf{T}_n, \Sigma_n)\}_{n \geq 1}$, such that Σ_n is a set of st-tgds of size linear in n, and every set Σ' of* $\mathrm{CQ}^{\mathbf{C}}$-TO-$\mathrm{UCQ}^{=}$ *ts-dependencies that specifies a maximum recovery of $\mathcal{M}_n$ is of size $\Omega(2^n)$.*

PROOF. Let $\mathbf{S}_n = \{R(\cdot), A_1(\cdot), B_1(\cdot), \ldots, A_n(\cdot), B_n(\cdot)\}$, $\mathbf{T}_n = \{P_1(\cdot), \ldots, P_n(\cdot)\}$, and Σ_n the set of st-tgds:

$$\begin{aligned} R(x) &\rightarrow \exists y(P_1(y) \wedge \cdots \wedge P_n(y)), \\ A_1(x) &\rightarrow P_1(x), \\ B_1(x) &\rightarrow P_1(x), \\ &\vdots \\ A_n(x) &\rightarrow P_n(x), \\ B_n(x) &\rightarrow P_n(x). \end{aligned}$$

Let $\mathcal{M}_n = (\mathbf{S}_n, \mathbf{T}_n, \Sigma_n)$ and assume that $\mathcal{M}' = (\mathbf{T}_n, \mathbf{S}_n, \Sigma')$ is a maximum recovery of $\mathcal{M}_n$, where Σ' is a set of $\text{CQ}^{\mathbf{C}}$-TO-$\text{UCQ}^{=}$ ts-dependencies. We first prove some facts about Σ'. Through the proof, we let a be a fixed element in $\mathbf{C}$, and I_R a source instance such that $R^{I_R} = \{a\}$ and $A_i^{I_R} = B_i^{I_R} = \emptyset$ for every $i \in \{1, \ldots, n\}$. Since $\mathcal{M}'$ is a recovery of $\mathcal{M}_n$, we have that $(I_R, I_R) \in \mathcal{M}_n \circ \mathcal{M}'$. Thus, there exists an instance $J^\star$ such that $(I_R, J^\star) \models \Sigma_n$ and $(J^\star, I_R) \models \Sigma'$. We show first that the domain of $J^\star$ is composed only by null values. On the contrary, assume that there exists a constant element $b \in \mathbf{C}$ such that $b \in \text{dom}(J^\star)$. Then it holds that $b \in P_k^{J^\star}$ for some $k \in \{1, \ldots, n\}$. Consider a source instance I' such that $A_k^{I'} = B_k^{I'} = \{b\}$, $R^{I'} = \emptyset$, and $A_i^{I'} = B_i^{I'} = \emptyset$ for every $i \in \{1, \ldots, n\}$ with $i \neq k$. The target instance J' where $P_k^{J'} = \{b\}$ and $P_i^{J'} = \emptyset$ for every $i \in \{1, \ldots, n\}$ with $i \neq k$, is such that $(I', J') \in \mathcal{M}_n$. Notice that $J' \subseteq J^\star$. Now since Σ_n is a set of st-tgds, we know that $\mathcal{M}_n$ is closed-up on the right, obtaining that $(I', J^\star) \in \mathcal{M}_n$. Thus, given that $(J^\star, I_R) \in \mathcal{M}'$ we have that $(I', I_R) \in \mathcal{M}_n \circ \mathcal{M}'$. This last fact contradicts Proposition 3.1.6 since $\mathcal{M}'$ is a maximum recovery of $\mathcal{M}_n$ and $\text{Sol}_{\mathcal{M}_n}(I_R) \not\subseteq \text{Sol}_{\mathcal{M}_n}(I')$.

We claim now that it must exist a dependency $\sigma \in \Sigma'$ such that $J^\star$ satisfies the premise of σ. Assume that this is not the case. Then since Σ' is a set of $\text{CQ}^{\mathbf{C}}$-TO-$\text{UCQ}^{=}$ formulas, it would be the case that $(J^\star, I_\emptyset) \models \Sigma'$, where $I_\emptyset$ is the empty source instance. Thus, we have that $(I_R, I_\emptyset) \in \mathcal{M}_n \circ \mathcal{M}'$ which, by Proposition 3.1.6, contradicts the fact that $\mathcal{M}'$

is a maximum recovery of $\mathcal{M}_n$ since $\mathrm{Sol}_{\mathcal{M}_n}(I_\emptyset) \not\subseteq \mathrm{Sol}_{\mathcal{M}_n}(I_R)$. Assume now that σ is a dependency in Σ' whose premise is satisfied by $J^\star$. We show next that the premise and the conclusion of σ must be Boolean formulas. On the contrary, assume that σ is of the form $\varphi(\bar{x}) \rightarrow \psi(\bar{x})$, where $\bar{x}$ is a tuple of m variables with $m > 0$. Since we are assuming that $J^\star$ satisfies the premise of σ, there exists an m-tuple $\bar{b}$ such that $J^\star \models \varphi(\bar{b})$. We know that $\varphi(\bar{x})$ is a domain independent formula, then it holds that every component of $\bar{b}$ is in $\mathrm{dom}(J^\star)$. We have shown before that $\mathrm{dom}(J^\star)$ is composed only by nulls and, thus, every component of $\bar{b}$ is a null value. Now, since $(J^\star, I_R) \models \sigma$ and $J^\star \models \varphi(\bar{b})$, it must be the case that $I_R \models \psi(\bar{b})$. We also know that $\psi(\bar{x})$ is domain independent, then every component of $\bar{b}$ must be in $\mathrm{dom}(I_R)$, which leads to a contradiction since $\mathrm{dom}(I_R) = \{a\}$ and $a \in \mathbf{C}$. In the rest of the proof, we let $\Sigma'' \subseteq \Sigma'$ to be the set of all the dependencies σ of the form $\varphi \rightarrow \psi$ such that $J^\star \models \varphi$, where φ and ψ are Boolean formulas. Notice that $\Sigma'' \neq \emptyset$.

We have the necessary ingredients to show that Σ' is of size $\Omega(2^n)$. Consider for every n-tuple $\bar{d} = (d_1, \ldots, d_n) \in \{0,1\}^n$, the set of source relation symbols $\mathbf{S}_{\bar{d}} = \{U_1(\cdot), \ldots, U_n(\cdot)\}$ such that $U_i = A_i$ if $d_i = 0$ and $U_i = B_i$ if $d_i = 1$. We now show that for each of the 2^n tuples $\bar{d}$, there must exist a dependency $\sigma \in \Sigma''$ of the form $\varphi \rightarrow \psi$ such that ψ has a disjunct that mentions exactly the relation symbols in $\mathbf{S}_{\bar{d}}$. This is enough to show that Σ' is of size $\Omega(2^n)$. Fix a tuple $\bar{d}$ and consider a source instance $I_{\bar{d}}$ such that for every $U \in \mathbf{S}_n$, if $U \in \mathbf{S}_{\bar{d}}$ then $U^{I_{\bar{d}}} = \{a\}$, otherwise $U^{I_{\bar{d}}} = \emptyset$. Since $\mathcal{M}'$ is a maximum recovery of $\mathcal{M}_n$, there exists a target instance $J_{\bar{d}}$ such that $(I_{\bar{d}}, J_{\bar{d}}) \models \Sigma_n$ and $(J_{\bar{d}}, I_{\bar{d}}) \models \Sigma'$. Let J_P be a target instance such that $P_i^{J_P} = \{a\}$ for every $i \in \{1, \ldots, n\}$. It is straightforward to see that $J_P \subseteq J_{\bar{d}}$. It is also easy to see that, if θ is a boolean query in $\mathrm{CQ}^{\mathbf{C}}$ over $\mathbf{T}_n$, then $J_P \models \theta$. To see this just take a homomorphism h from the conjunctions of θ to the facts in J_P such that $h(x) = a$ for every existential variable in θ, and note that $\mathbf{C}(h(x))$ holds for every variable since $a \in \mathbf{C}$. Thus, given that queries in $\mathrm{CQ}^{\mathbf{C}}$ are monotone and $J_P \subseteq J_{\bar{d}}$, we have that $J_{\bar{d}} \models \theta$ for every $\mathrm{CQ}^{\mathbf{C}}$ boolean query θ over $\mathbf{T}_n$. In particular, we have that for every $\varphi \rightarrow \psi \in \Sigma''$, it holds that $J_{\bar{d}} \models \varphi$. Then it must hold that $I_{\bar{d}} \models \psi$ for every $\varphi \rightarrow \psi \in \Sigma''$. This last fact implies that for every $\varphi \rightarrow \psi \in \Sigma''$, there exists a formula α such that α is one of the disjunctions of ψ and $I_{\bar{d}} \models \alpha$ (recall that ψ is a query in $\mathrm{UCQ}^{=}$). Let Γ be a set

containing all such formulas α, that is, α is a formula in Γ if and only if there exists a dependency $\varphi \rightarrow \psi \in \Sigma''$ such that α is a disjunction in ψ and $I_{\bar{d}} \models \alpha$. Note that every $\alpha \in \Gamma$ is a $\text{CQ}^{=}$ Boolean query, and since $I_{\bar{d}} \models \alpha$, it could not be the case that α mentions relation symbols of $\mathbf{S}_n$ outside $\mathbf{S}_{\bar{d}}$. We now show that one of the queries in Γ mentions exactly the relation symbols in $\mathbf{S}_{\bar{d}}$. On the contrary, assume that for every $\alpha \in \Gamma$, it is the case that α mentions a proper subset of the relation symbols of $\mathbf{S}_{\bar{d}}$. Consider for every $\alpha \in \Gamma$ a fresh constant value c_α, and a source instance I_α such that for every $U \in \mathbf{S}_n$, we have $U^{I_\alpha} = \{c_\alpha\}$ if the relation symbol U is mentioned in α, and $U^{I_\alpha} = \emptyset$ otherwise. It is clear that $I_\alpha \models \alpha$ for every $\alpha \in \Gamma$. Let $I_\Gamma = \bigcup_{\alpha \in \Gamma} I_\alpha$. Notice that for every $\alpha \in \Gamma$, it holds that $I_\Gamma \models \alpha$. Recall that for every $\varphi \rightarrow \psi \in \Sigma''$, there exists a formula $\alpha \in \Gamma$ such that α is one of the disjunctions of ψ. Hence, we obtain that $I_\Gamma \models \psi$ for every $\varphi \rightarrow \psi \in \Sigma''$. We also know that $J^\star \models \varphi$ for every $\varphi \rightarrow \psi \in \Sigma''$, obtaining that $(J^\star, I_\Gamma) \models \Sigma''$. Notice that Σ'' contains all the dependencies of Σ' such that $J^\star$ satisfies their premises and, thus, $(J^\star, I_\Gamma) \models \Sigma'$. Then since $(I_R, J^\star) \models \Sigma_n$ and $(J^\star, I_\Gamma) \models \Sigma'$, we have that $(I_R, I_\Gamma) \in \mathcal{M}_n \circ \mathcal{M}'$. We show now that $\text{Sol}_{\mathcal{M}_n}(I_\Gamma) \not\subseteq \text{Sol}_{\mathcal{M}_n}(I_R)$, which contradicts Proposition 3.1.6. Notice first that for every target instance $J \in \text{Sol}_{\mathcal{M}_n}(I_R)$, there exists an element $c \in \text{dom}(J)$ such that $c \in P_i^J$ for every $i \in \{1, \ldots, n\}$. We prove that there exists an instance in $\text{Sol}_{\mathcal{M}_n}(I_\Gamma)$ that does not satisfy this last property. Consider for every $\alpha \in \Gamma$ the target instance $J_\alpha = \text{chase}_{\Sigma_n}(I_\alpha)$, and let $J_\Gamma = \bigcup_{\alpha \in \Gamma} J_\alpha$. It is easy to see that $J_\Gamma \in \text{Sol}_{\mathcal{M}_n}(I_\Gamma)$. Notice that since every $\alpha \in \Gamma$ mentions a proper subset of the relation symbols of $\mathbf{S}_{\bar{d}}$, there exists an index $i \in \{1, \ldots, n\}$ such that $A_i^{I_\alpha} = B_i^{I_\alpha} = \emptyset$, and then there exists an index $i \in \{1, \ldots, n\}$ such that $P_i^{J_\alpha} = \emptyset$. Moreover, since $\text{dom}(J_\alpha) \cap \text{dom}(J_{\alpha'}) = \emptyset$ for every pair of distinct elements α, α' of Γ, we obtain that there is no element $c \in \text{dom}(J_\Gamma)$ such that $c \in P_i^{J_\Gamma}$ for every $i \in \{1, \ldots, n\}$. Thus, $J_\Gamma \notin \text{Sol}_{\mathcal{M}_n}(I_R)$ implying that $\text{Sol}_{\mathcal{M}_n}(I_\Gamma) \not\subseteq \text{Sol}_{\mathcal{M}_n}(I_R)$ which leads to the contradiction mentioned above. We have shown that there exists a formula $\alpha \in \Gamma$ such that α mentions exactly the relation symbols in $\mathbf{S}_{\bar{d}}$. Thus, there exists a dependency $\sigma \in \Sigma'' \subseteq \Sigma'$ such that the conclusion of σ has a disjunct that mentions exactly the relation symbols in $\mathbf{S}_{\bar{d}}$. This last property holds for every one of the 2^n distinct tuples $\bar{d}$, which implies that Σ' is of size exponential in the size of Σ_n. $\square$

3.3.3. Computing maximum recoveries in the full case

Recall that a full FO-TO-CQ dependency does not include any existential quantifiers in its conclusion. In this section, we show that for mappings given by full FO-TO-CQ dependencies, maximum recoveries can be computed in polynomial time. This result is based in the fact that given a query composed by a single atom and with no existentially quantified variables, one can compute a rewriting of that query in quadratic time. This is formalized in the following lemma, where $\|\Sigma\|$ denotes the size of Σ. The proof of the lemma is given in Appendix A.1.2.

LEMMA 3.3.9. *There exists an algorithm* QUERYREWRITINGATOM *that given an st-mapping* $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$*, with* Σ *a set of* FO-TO-CQ *dependencies, and a conjunctive query* Q *over schema* $\mathbf{T}$ *composed by a single atom and with no existential quantifiers, computes in time* $O(\|\Sigma\|^2)$ *a domain-independent* FO *query* Q' *that is a rewriting of* Q *over the source. Moreover, if* Σ *is a set of full* FO-TO-CQ *st-dependencies where each dependency has a single atom in its conclusion, then the algorithm runs in time* $O(\|\Sigma\|)$.

By using algorithm QUERYREWRITINGATOM, we can compute in quadratic time a maximum recovery for mappings given by full dependencies.

Algorithm MAXIMUMRECOVERYFULL($\mathcal{M}$)

Input: An st-mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where Σ is a set of full FO-TO-CQ dependencies, each dependency with a single atom in its conclusion.

Output: A ts-mapping $\mathcal{M}' = (\mathbf{T}, \mathbf{S}, \Sigma')$, where Σ' is a set of CQ-TO-FO dependencies and $\mathcal{M}'$ is a maximum recovery of $\mathcal{M}$.

(1) Start with Σ' as the empty set.

(2) For every atom $R(\bar{x})$ that is the conclusion of a dependency in Σ, do the following:

  (a) Let Q be the conjunctive query defined by $R(\bar{x})$.

  (b) Use QUERYREWRITINGATOM($\mathcal{M}, Q$) to compute an FO formula $\alpha(\bar{x})$ that is a rewriting of $R(\bar{x})$ over the source.

(c) Add dependency $R(\bar{x}) \rightarrow \alpha(\bar{x})$ to Σ'.

(3) Return $\mathcal{M}' = (\mathbf{T}, \mathbf{S}, \Sigma')$. □

THEOREM 3.3.10. *Let $\mathcal{M}$ be an st-mapping specified by a set Σ of full* FO-TO-CQ *st-dependencies in which each dependency has a single atom in its conclusion. Then algorithm* MAXIMUMRECOVERYFULL($\mathcal{M}$) *computes a maximum recovery of $\mathcal{M}$ in time $O(\|\Sigma\|^2)$, which is specified by a set of* CQ-TO-FO *dependencies.*

PROOF. Since Σ is a set of full FO-TO-CQ st-dependencies, each dependency with a single atom in its conclusion, algorithm QUERYREWRITINGATOM($\mathcal{M}, Q$) runs in linear time. Thus, it is straightforward to see that algorithm MAXIMUMRECOVERYFULL runs in quadratic time. The correctness of the algorithm follows from the proof of Theorem 3.3.2. We only notice here that the output of algorithm MAXIMUMRECOVERYFULL does not include predicate $\mathbf{C}(\cdot)$. Since Σ is a set of full dependencies, $\mathrm{chase}_\Sigma(I)$ is composed only by constant values and, thus, $\mathbf{C}(\cdot)$ is not needed in the proof of Theorem 3.3.2. □

Notice that in Theorem 3.3.10, we assume that every dependency has a single atom in its conclusion. Nevertheless, this theorem can be extended to the general case; from a set Σ of arbitrary full FO-TO-CQ st-dependencies, one can obtain as follows an equivalent set Σ' of full FO-TO-CQ st-dependencies having a single atom in the conclusion of each constraint. For every dependency $\varphi(\bar{x}) \rightarrow \psi(\bar{x})$ in Σ and atom $R(\bar{y})$ in $\psi(\bar{x})$, where $\bar{y} \subseteq \bar{x}$, the dependency $\varphi(\bar{x}) \rightarrow R(\bar{y})$ is included in Σ'. Thus, to apply Theorem 3.3.10 to Σ, we first construct Σ' from Σ and then apply procedure MAXIMUMRECOVERYFULL. It is important to notice that Σ' could be of quadratic size in the size of Σ and, hence, by the fact that algorithm QUERYREWRITINGATOM runs in linear time and the definition of procedure MAXIMUMRECOVERYFULL, it follows that a maximum recovery for a mapping specified by an arbitrary set of full FO-TO-CQ st-dependencies can be computed in cubic-time.

As for the general case, from Theorem 3.1.10, we know that this algorithm computes a Fagin-inverse (quasi-inverse) if Σ is a Fagin-invertible (quasi-invertible) set of full st-tgds. The algorithm presented by Fagin, Kolaitis, Popa, and Tan (2008) for computing

a Fagin-inverse of a set Σ of full st-tgds returns a set Σ' of $\text{CQ}^{\neq}$-TO-CQ dependencies of exponential size in $\|\Sigma\|$. The algorithm in (Fagin, Kolaitis, Popa, & Tan, 2008) for computing a quasi-inverse of a set Σ of full st-tgds returns a set Σ' of $\text{CQ}^{\neq}$-TO-UCQ dependencies which is also of exponential size in $\|\Sigma\|$. In both cases, our algorithm works in quadratic time and returns a set Σ' of CQ-TO-$\text{UCQ}^{=}$ dependencies which is of quadratic size in $\|\Sigma\|$.

Notice that in Theorem 3.3.6 we use a mapping given by a set of full st-tgds to show that disjunctions in the conclusion of dependencies are needed to specify maximum recoveries of st-tgds. Similarly, Fagin, Kolaitis, Popa, and Tan (2008, Theorem 4.15) also use a mapping specified by a set of full st-tgds, and thus, from Corollary 3.3.7 we know that equalities in the conclusion are needed to specify maximum recoveries of mappings given by full st-tgds. Thus we obtain the following corollary that justify the language used in the output of MAXIMUMRECOVERYFULL.

COROLLARY 3.3.11.

(1) *There exists an st-mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ specified by a set of full st-tgds that has no maximum recovery specified by a set of* CQ-TO-$\text{CQ}^{=}$ *dependencies.*

(2) *There exists an st-mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ specified by a set of full st-tgds that has no maximum recovery specified by a set of* CQ^{C}-TO-UCQ *dependencies.*

3.4. Complexity Results

Fagin (2007) identified two problems as important decision problems for the notion of Fagin-inverse: (1) to check whether a mapping $\mathcal{M}$ is Fagin-invertible, and (2) to check whether a mapping $\mathcal{M}_2$ is a Fagin-inverse of a mapping $\mathcal{M}_1$. These questions are considered in the context of st-tgds in (Fagin, 2007), where they are also relevant for the notion of quasi-inverse (Fagin, Kolaitis, Popa, & Tan, 2008). In this context, the problem of verifying whether a mapping $\mathcal{M}$ has a maximum recovery becomes trivial, as every mapping specified by this type of dependencies admits a maximum recovery. In fact, this question is also trivial for the larger class of mappings specified by FO-TO-CQ dependencies. In this

section we study the complexity of verifying, given mappings $\mathcal{M}$ and $\mathcal{M}'$, whether $\mathcal{M}'$ is a recovery of $\mathcal{M}$. Notice that we only study the problem of verifying whether $\mathcal{M}'$ is a recovery but not a maximum recovery of $\mathcal{M}$. We leave the study of the complexity of this last problem for future work.

We note that the problem of checking wether $\mathcal{M}'$ is a recovery of $\mathcal{M}$ becomes undecidable if $\mathcal{M}$ is specified by a set Σ of full FO-TO-CQ dependencies (this is a straightforward consequence of the undecidability of the problem of verifying whether an FO sentence is finitely satisfiable (Libkin, 2004)). Thus, in this section we focus on the case in which the mapping $\mathcal{M}$ is specified by st-tgds.

We begin with a simple technical lemma that will allow us to use some of the complexity results proved by Fagin (2007) for Fagin-inverses (see Definition 2.5.1 for the definition of Fagin-inverses).

LEMMA 3.4.1. *Let $\mathcal{M}$ be an st-mapping specified by a set of st-tgds. Assume that $\mathcal{M}'$ is a ts-mapping such that whenever $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$, it holds that $I_1 \subseteq I_2$. Then $\mathcal{M}'$ is a Fagin-inverse of $\mathcal{M}$ iff $\mathcal{M}'$ is a recovery of $\mathcal{M}$*

PROOF. The direction $(\Rightarrow)$ is trivial by the definition of Fagin-inverse and recovery. To prove the other direction, assume that $\mathcal{M}'$ is a recovery of $\mathcal{M}$. We must show that $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$ if and only if $I_1 \subseteq I_2$. By hypothesis, it holds that if $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$ then $I_1 \subseteq I_2$. Now, assume that $I_1 \subseteq I_2$. Since $\mathcal{M}'$ is a recovery of $\mathcal{M}$, we know that $(I_2, I_2) \in \mathcal{M} \circ \mathcal{M}'$ and then, there exists a target instance J such that $(I_2, J) \in \mathcal{M}$ and $(J, I_2) \in \mathcal{M}'$. Now, given that $\mathcal{M}$ is specified by a set of st-tgds, $\mathcal{M}$ is closed-down on the left and then $(I_1, J) \in \mathcal{M}$. We have that $(I_1, J) \in \mathcal{M}$ and $(J, I_2) \in \mathcal{M}'$, which implies that $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$. This was to be shown. □

Before stating our complexity results we also need to introduce some terminology and prove a technical lemma. Let Σ be a set of CQ-TO-CQ dependencies from a schema $\mathbf{R}_1$ to a schema $\mathbf{R}_2$ and I an instance of $\mathbf{R}_1$. We denote by k_Σ the maximum, over all members $\varphi \in \Sigma$, of the number of conjuncts that appear in the premise of φ, and by $|I|$ the total

number of tuples in I, that is, $|I| = \sum_{R \in \mathbf{R}_1} |R^I|$, where $|R^I|$ is the number of tuples in R^I. Moreover, we define the notion of I being $\mathbf{N}$-*connected* as follows. Let $G_I = (V_I, E_I)$ be a graph such that: (1) V_I is the set of all tuples $t \in R^I$, for some $R \in \mathbf{R}_1$, and (2) a tuple $(t_1, t_2) \in E_I$ if and only if there exists a null value $n \in \mathbf{N}$ that is mentioned both in t_1 and t_2. Then I is $\mathbf{N}$-connected if the graph G_I is connected. An instance I_1 is an $\mathbf{N}$-*connected sub-instance* of I, if I_1 is a sub-instance of I and I_1 is $\mathbf{N}$-connected. Finally, I_1 is an $\mathbf{N}$-*connected component* of I, if I_1 is an $\mathbf{N}$-connected sub-instance of I and there is no $\mathbf{N}$-connected sub-instance I_2 of I such that I_1 is a proper sub-instance of I_2.

LEMMA 3.4.2. *Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ and $\mathcal{M}' = (\mathbf{T}, \mathbf{S}, \Sigma')$ be schema mappings, where Σ is a set of full st-tgds and Σ' is a set of ts-tgds. Then $\mathcal{M}'$ is a recovery of $\mathcal{M}$ if and only if for every source instance I such that $|I| \leq k_\Sigma \cdot k_{\Sigma'}$ and $\mathbf{N}$-connected component K of $\mathrm{chase}_{\Sigma'}(\mathrm{chase}_\Sigma(I))$, there exists a homomorphism from K to I that is the identity on $\mathbf{C}$.*

PROOF. We first prove direction ($\Rightarrow$). From (Fagin, Kolaitis, Popa, & Tan, 2005), we know that $\mathcal{M} \circ \mathcal{M}'$ can be specified by a set of st-tgds. Now, from (Fagin, 2007) (Proposition 7.2) we know that $\mathrm{chase}_{\Sigma'}(\mathrm{chase}_\Sigma(I))$ is a universal solution for I under $\mathcal{M} \circ \mathcal{M}'$, and then $(I, I) \in \mathcal{M} \circ \mathcal{M}'$ if and only if there exists a homomorphism from $\mathrm{chase}_{\Sigma'}(\mathrm{chase}_\Sigma(I))$ to I that is the identity on $\mathbf{C}$. The ($\Rightarrow$) direction of the proposition follows from the latter condition.

We now prove ($\Leftarrow$). Without loss of generality, assume that each st-tgd in Σ has a single atom in its right-hand side. For the sake of contradiction, suppose that $\mathcal{M}'$ is not a recovery of $\mathcal{M}$ and for every source instance I such that $|I| \leq k_\Sigma \cdot k_{\Sigma'}$ and $\mathbf{N}$-connected component K of $\mathrm{chase}_{\Sigma'}(\mathrm{chase}_\Sigma(I))$, there exists a homomorphism from K to I that is the identity on $\mathbf{C}$.

Given that $\mathcal{M}'$ is not a recovery of $\mathcal{M}$, there exists an instance I_1 of $\mathbf{S}$ such that $(I_1, I_1) \notin \mathcal{M} \circ \mathcal{M}'$. Let I be an instance of $\mathbf{S}$. Given that $\mathrm{chase}_\Sigma(I)$ is a universal solution for I under $\mathcal{M}$ and $\mathrm{chase}_{\Sigma'}(\mathrm{chase}_\Sigma(I))$ is a universal solution for $\mathrm{chase}_\Sigma(I)$ under $\mathcal{M}'$, it is straightforward to prove that if $(I, I') \in \mathcal{M} \circ \mathcal{M}'$, then there exists a homomorphism from $\mathrm{chase}_{\Sigma'}(\mathrm{chase}_\Sigma(I))$ to I' that is the identity on $\mathbf{C}$. Furthermore, if there exists a

homomorphism from $\mathrm{chase}_{\Sigma'}(\mathrm{chase}_{\Sigma}(I))$ to an instance I', then one can conclude that $(\mathrm{chase}_{\Sigma}(I), I') \in \mathcal{M}'$ since $(\mathrm{chase}_{\Sigma}(I), \mathrm{chase}_{\Sigma'}(\mathrm{chase}_{\Sigma}(I))) \in \mathcal{M}'$ and $\mathrm{chase}_{\Sigma}(I)$ does not mention any null values as Σ is a set of full st-tgds. Thus, we have that if there exists a homomorphism from $\mathrm{chase}_{\Sigma'}(\mathrm{chase}_{\Sigma}(I))$ to an instance I', then $(I, I') \in \mathcal{M} \circ \mathcal{M}'$. In particular, from the previous properties, we conclude that $(I, I) \in \mathcal{M} \circ \mathcal{M}'$ if and only if there exists a homomorphism from $\mathrm{chase}_{\Sigma'}(\mathrm{chase}_{\Sigma}(I))$ to I that is the identity on $\mathbf{C}$. Thus, given that $(I_1, I_1) \notin \mathcal{M} \circ \mathcal{M}'$, there is no homomorphism from $\mathrm{chase}_{\Sigma'}(\mathrm{chase}_{\Sigma}(I_1))$ to I_1 that is the identity on $\mathbf{C}$, which implies that there exists an $\mathbf{N}$-connected component K_1 of $\mathrm{chase}_{\Sigma'}(\mathrm{chase}_{\Sigma}(I_1))$ such that there is no homomorphism from K_1 to I_1 that is the identity on $\mathbf{C}$. Given that K_1 is an $\mathbf{N}$-connected component and Σ is a set of full st-tgds, there exists a dependency $\alpha(\bar{x}) \rightarrow \exists \bar{y}\, \beta(\bar{x}, \bar{y})$ in Σ' and a tuple $\bar{a}$ of elements from $\mathbf{C}$ such that $\mathrm{chase}_{\Sigma}(I_1) \models \alpha(\bar{a})$ and K_1 is generated from $\exists \bar{y}\, \beta(\bar{a}, \bar{y})$ when computing $\mathrm{chase}_{\Sigma'}(\mathrm{chase}_{\Sigma}(I_1))$. Assume that $\alpha(\bar{a})$ is equal to $T_1(\bar{a}_1) \wedge \cdots \wedge T_n(\bar{a}_n)$. Then for every $i \in \{1, \ldots, n\}$, there exists a full st-tgd $\gamma_i(\bar{x}_i) \rightarrow T_i(\bar{x}_i)$ such that $I_1 \models \gamma_i(\bar{a}_i)$. Let I_2 be a sub-instance of I_1 given by the union of all the tuples in the formulas $\gamma_i(\bar{a}_i)$ ($i \in \{1, \ldots, n\}$). Then we have that K_1 is an $\mathbf{N}$-connected component of $\mathrm{chase}_{\Sigma'}(\mathrm{chase}_{\Sigma}(I_2))$ and there is no homomorphism from K_1 to I_2 that is the identity on $\mathbf{C}$. But by definition of I_2, we know that $|I_2| \leq k_{\Sigma} \cdot k_{\Sigma'}$, which contradicts our initial assumption. □

We are now ready to state our first complexity result.

THEOREM 3.4.3. *The problem of verifying, given mappings $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ and $\mathcal{M}' = (\mathbf{T}, \mathbf{S}, \Sigma')$, where Σ is a set of full st-tgds and Σ' is a set of full ts-tgds, whether $\mathcal{M}'$ is a recovery of $\mathcal{M}$ is coNP-complete.*

PROOF. First, we assume that Σ' is a set of full ts-tgds, and we show that the problem of verifying whether $\mathcal{M}'$ is not a recovery of $\mathcal{M}$ is NP-complete. From Lemma 3.4.2 and the fact that Σ' is a set of full ts-tgds, we have that $\mathcal{M}'$ is not a recovery of $\mathcal{M}$ if and only if there exists a source instance I such that $|I| \leq k_{\Sigma} \cdot k_{\Sigma'}$ and there exists a tuple in $\mathrm{chase}_{\Sigma'}(\mathrm{chase}_{\Sigma}(I))$ which is not in I. The latter is an NP property; to check whether it holds, it is enough to guess an instance I such that $|I| \leq k_{\Sigma} \cdot k_{\Sigma'}$, and then guess the chase

steps that produce a tuple which is not in I. Thus, we have that the problem of verifying whether $\mathcal{M}'$ is not a recovery of $\mathcal{M}$ is in NP.

To show that the problem is coNP-hard we use a result by Fagin (2007). Fagin (2007, proof of Theorem 14.9) showed that given a propositional formula φ, one can construct two mappings $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ and $\mathcal{M}' = (\mathbf{T}, \mathbf{S}, \Sigma')$ with Σ and Σ' sets of full st-tgds and full ts-tgds, respectively, such that $\mathcal{M}'$ is an inverse of $\mathcal{M}$ if and only if φ is not satisfiable. Moreover, the mappings constructed in that proof were such that if $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$ then $I_1 \subseteq I_2$. Then from Lemma 3.4.1, we know that $\mathcal{M}'$ is an inverse of $\mathcal{M}$ if and only if $\mathcal{M}'$ is a recovery of $\mathcal{M}$. We have that, $\mathcal{M}'$ is a recovery of $\mathcal{M}$ if and only if φ is not satisfiable. Thus, the hardness results follows then from the well known fact that, testing whether a propositional formula is satisfiable is an NP-complete problem. □

Theorem 3.4.3 is in sharp contrast with the results presented by Fagin (2007), where it is shown that the problem of verifying, given schema mappings $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ and $\mathcal{M}' = (\mathbf{T}, \mathbf{S}, \Sigma')$, with Σ a set of full st-tgds and Σ' a set of full ts-tgds, whether $\mathcal{M}'$ is a Fagin-inverse of $\mathcal{M}$ is DP-complete[2]. The lower complexity for the case of the recovery is not surprising as the notion of recovery is much weaker than the notion of inverse.

Our next result settles the complexity for the case in which $\mathcal{M}$ is specified by a set of full st-tgds while the dependencies specifying $\mathcal{M}'$ are not necessarily full.

THEOREM 3.4.4. *The problem of verifying, given mappings $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ and $\mathcal{M}' = (\mathbf{T}, \mathbf{S}, \Sigma')$, where Σ is a set of full st-tgds and Σ' is a set of ts-tgds, whether $\mathcal{M}'$ is a recovery of $\mathcal{M}$ is Π_2^P-complete.*

PROOF. From Lemma 3.4.2, we have that $\mathcal{M}'$ is a recovery of $\mathcal{M}$ if and only if for every source instance I such that $|I| \leq k_\Sigma \cdot k_{\Sigma'}$ and $\mathbf{N}$-connected component K of $\mathrm{chase}_{\Sigma'}(\mathrm{chase}_\Sigma(I))$, there exists a homomorphism from K to I that is the identity on $\mathbf{C}$. Given that the size of I, as well as the size of K, is polynomial in the size of $\mathcal{M}$ and $\mathcal{M}'$, and that the homomorphism problem is in NP, we have that the problem of verifying whether

[2]A problem is in DP if it is the intersection of an NP problem and a coNP problem (Papadimitriou, 1994).

$\mathcal{M}'$ is a recovery of $\mathcal{M}$ is in Π_2^P. To prove that this problem is indeed Π_2^P-complete, we give a reduction from the problem of verifying whether a quantified propositional formula:

$$\varphi \;=\; \forall u_1 \cdots \forall u_\ell \exists v_1 \cdots \exists v_m \, \psi, \tag{3.8}$$

is valid, where ψ is a 3-CNF propositional formula. This problem is known to be Π_2^P-complete (Du & Ko, 2000).

Let $\mathbf{S} = \{TV(\cdot,\cdot), R_0(\cdot,\cdot,\cdot), R_1(\cdot,\cdot,\cdot), R_2(\cdot,\cdot,\cdot), R_3(\cdot,\cdot,\cdot)\}$ and $\mathbf{T} = \{U_1(\cdot,\cdot,\cdot), \ldots, U_\ell(\cdot,\cdot,\cdot)\}$. Next we define schema mappings $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ and $\mathcal{M}' = (\mathbf{T}, \mathbf{S}, \Sigma')$ such that, φ is valid if and only if $\mathcal{M}'$ is a recovery of $\mathcal{M}$. The first argument of predicate TV is used to store the truth value *true*, while its second argument is used to store the truth value *false*. Predicate R_0 is used to store the truth assignments that satisfy the clauses of the form $u \vee v \vee w$ (clauses without negative literals). Assuming that variables x, y store values *true* and *false*, respectively, the following formula is used to define R_0:

$$\varphi_0(x,y) \;=\; R_0(x,x,x) \wedge R_0(x,x,y) \wedge R_0(x,y,x) \wedge R_0(y,x,x) \wedge R_0(x,y,y) \wedge R_0(y,x,y) \wedge R_0(y,y,x).$$

Similarly, predicate R_1 is used to store the truth assignments that satisfy the clauses of the form $u \vee v \vee \neg w$, predicate R_2 is used to store the truth assignments that satisfy the clauses of the form $u \vee \neg v \vee \neg w$, and predicate R_3 is used to store the truth assignments that satisfy the clauses of the form $\neg u \vee \neg v \vee \neg w$. Again assuming that variables x, y store values *true*

and *false*, respectively, the following formulas are used to define R_1, R_2 and R_3:

$$\begin{aligned}
\varphi_1(x,y) \;=\; & R_1(x,x,x) \wedge R_1(x,x,y) \wedge R_1(x,y,x) \wedge \\
& R_1(y,x,x) \wedge R_1(x,y,y) \wedge R_1(y,x,y) \wedge R_1(y,y,y), \\
\varphi_2(x,y) \;=\; & R_2(x,x,x) \wedge R_2(x,x,y) \wedge R_2(x,y,x) \wedge \\
& R_2(x,y,y) \wedge R_2(y,x,y) \wedge R_2(y,y,x) \wedge R_2(y,y,y), \\
\varphi_3(x,y) \;=\; & R_3(x,x,y) \wedge R_3(x,y,x) \wedge R_3(y,x,x) \wedge \\
& R_3(x,y,y) \wedge R_3(y,x,y) \wedge R_3(y,y,x) \wedge R_3(y,y,y).
\end{aligned}$$

Finally, the first argument of predicate U_i is used to store the truth value of propositional variable u_i, for every $i \in \{1, \ldots, \ell\}$. We include two extra arguments in U_i for a technical reason that will become clear when we prove that the reduction is correct.

Set Σ of full st-tgds is given by the following dependency:

$$\begin{aligned}
T(x,y) \wedge \varphi_0(x,y) \wedge \varphi_1(x,y) \wedge \varphi_2(x,y) \wedge \varphi_3(x,y) \;\rightarrow \\
U_1(x,x,y) \wedge U_1(y,x,y) \wedge \cdots \wedge U_\ell(x,x,y) \wedge U_\ell(y,x,y). \quad (3.9)
\end{aligned}$$

Set Σ' of ts-tgds is given by the following dependency:

$$U_1(u_1,x,y) \wedge \cdots \wedge U_\ell(u_\ell,x,y) \;\rightarrow\; \exists v_1 \cdots \exists v_m \, \theta(u_1,\ldots,u_\ell,v_1,\ldots,v_m), \quad (3.10)$$

where $\theta(u_1,\ldots,u_\ell,v_1,\ldots,v_m)$ is defined as follows. If 3-CNF formula ψ in (3.8) is equal to $C_1 \wedge \cdots \wedge C_k$, where each C_i is a clause, then $\theta = \theta_1 \wedge \cdots \wedge \theta_k$, where θ_i is obtained from C_i as follows. Without loss of generality, we assume that in C_i, the positive literals appear before the negative literals (if C_i has at least one positive literal). Then if $C_i = u \vee v \vee w$, we have that $\theta_i = R_0(u,v,w)$, if $C_i = u \vee v \vee \neg w$, we have that $\theta_i = R_1(u,v,w)$, if $C_i = u \vee \neg v \vee \neg w$, we have that $\theta_i = R_2(u,v,w)$, and if $C_i = \neg u \vee \neg v \vee \neg w$, we have that $\theta_i = R_3(u,v,w)$. For example, if $\varphi = \forall u_1 \forall u_2 \exists v_1 \, ((u_1 \vee v_1 \vee \neg u_2) \wedge (u_1 \vee u_2 \vee v_1))$,

then

$$\begin{aligned}\Sigma &= \{TV(x,y) \wedge \varphi_0(x,y) \wedge \varphi_1(x,y) \wedge \varphi_2(x,y) \wedge \varphi_3(x,y) \rightarrow \\ &\qquad U_1(x,x,y) \wedge U_1(y,x,y) \wedge U_2(x,x,y) \wedge U_2(y,x,y)\}, \\ \Sigma' &= \{U_1(u_1,x,y) \wedge U_2(u_2,x,y) \rightarrow \exists v_1\,(R_1(u_1,v_1,u_2) \wedge R_0(u_1,u_2,v_1))\}.\end{aligned}$$

Next we show that φ is valid if and only if $\mathcal{M}'$ is a recovery of $\mathcal{M}$.

($\Rightarrow$) Assume that φ is valid. From Lemma 3.4.2, to show that $\mathcal{M}'$ is a recovery of $\mathcal{M}$, it is enough to prove that for every instance I of $\mathbf{S}$ and $\mathbf{N}$-connected component K of $\mathrm{chase}_{\Sigma'}(\mathrm{chase}_{\Sigma}(I))$, there exists a homomorphism from K to I that is the identity on $\mathbf{C}$. Next we show that this is the case.

Let I_1 be an instance of $\mathbf{S}$. By definition of Σ and Σ', and in particular because of the inclusion of the two extra arguments in each U_i, we have that if K is an $\mathbf{N}$-connected component of $\mathrm{chase}_{\Sigma'}(\mathrm{chase}_{\Sigma}(I_1))$, then there exists a pair of values a, b in $\mathrm{dom}(I_1)$ such that: (1) $I_1 \models T(a,b) \wedge \varphi_0(a,b) \wedge \varphi_1(a,b) \wedge \varphi_2(a,b) \wedge \varphi_3(a,b)$, (2) $\mathrm{chase}_{\Sigma}(I_1) \models U_1(c_1,a,b) \wedge \cdots \wedge U_\ell(c_\ell,a,b)$, where each c_i is either a or b, and (3) K is generated from $\exists v_1 \cdots \exists v_m\, \theta(c_1,\ldots,c_\ell,v_1,\ldots,v_m)$ when computing $\mathrm{chase}_{\Sigma'}(\mathrm{chase}_{\Sigma}(I_1))$. Assume that in the construction of K, variable v_i is replaced by value $n_i \in \mathbf{N}$, for every $i \in \{1,\ldots,m\}$. Given that φ is valid, we know that for the truth assignment σ_1 such that $\sigma_1(u_i) = c_i$, for every $i \in \{1,\ldots,\ell\}$, there exists a truth assignment σ_2 such that $\sigma_1 \cup \sigma_2$ satisfies propositional formula ψ in (3.8). From this we conclude that function h defined as $h(n_i) = \sigma_2(v_i)$ ($i \in \{1,\ldots,m\}$) and $h(c) = c$ ($c \in \mathbf{C}$) is a homomorphism from K to I that is the identity on $\mathbf{C}$.

($\Leftarrow$) Assume that $\mathcal{M}'$ is a recovery of $\mathcal{M}$, and let I be an instance of $\mathbf{S}$ such that $T^I = \{(a, b)\}$, where a and b are two distinct elements from $\mathbf{C}$, and

$$\begin{aligned}
R_0^I &= \{(a,a,a),(a,a,b),(a,b,a),(b,a,a),(a,b,b),(b,a,b),(b,b,a)\},\\
R_1^I &= \{(a,a,a),(a,a,b),(a,b,a),(b,a,a),(a,b,b),(b,a,b),(b,b,b)\},\\
R_2^I &= \{(a,a,a),(a,a,b),(a,b,a),(a,b,b),(b,a,b),(b,b,a),(b,b,b)\},\\
R_3^I &= \{(a,a,b),(a,b,a),(b,a,a),(a,b,b),(b,a,b),(b,b,a),(b,b,b)\}.
\end{aligned}$$

Given that $\mathcal{M}'$ is a recovery of $\mathcal{M}$, we have that $(I, I) \in \mathcal{M} \circ \mathcal{M}'$. Thus, for every tuple $(c_1, \ldots, c_\ell) \in \{a, b\}^\ell$, there exists a tuple $(d_1, \ldots, d_m) \in \{a, b\}^m$ such that $I \models \theta(c_1, \ldots, c_\ell, d_1, \ldots, d_m)$. Hence, by the definitions of θ, R_0^I, R_1^I, R_2^I and R_3^I, we conclude that φ is a valid formula. This concludes the proof of the theorem. □

In an unpublished manuscript, Arenas (2006) showed that the problem of verifying, given mappings $\mathcal{M}$ and $\mathcal{M}'$ specified by st-tgds and ts-tgds, respectively, whether $\mathcal{M}'$ is a Fagin-inverse of $\mathcal{M}$, is undecidable. Arenas et al. (2008) adapted this result to show that for the notion of recovery this problem is also undecidable. The following theorem formalizes this result. The proof can be found in (Arenas, Pérez, & Riveros, 2009).

THEOREM 3.4.5 (Arenas, 2006; Arenas et al., 2008). *The problem of verifying, given mappings $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ and $\mathcal{M}' = (\mathbf{T}, \mathbf{S}, \Sigma')$, where Σ is a set of st-tgds and Σ' is a set of ts-tgds, whether $\mathcal{M}'$ is a recovery of $\mathcal{M}$ is undecidable.*

Interestingly, Arenas et al. (2008) obtain as a corollary of the proof of the previous result that the problem is also undecidable for the notion of maximum recovery. The proof of this result can be found in (Arenas, Pérez, & Riveros, 2009).

COROLLARY 3.4.6 (Arenas et al., 2008). *The problems of verifying, given mappings $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ and $\mathcal{M}' = (\mathbf{T}, \mathbf{S}, \Sigma')$, where Σ is a set of st-tgds and Σ' is a set of ts-tgds, whether $\mathcal{M}'$ is a maximum recovery of $\mathcal{M}$ is undecidable.*

It is an interesting open problem whether one can obtain decidability results for the case of maximum recovery when $\mathcal{M}$ is given by a set of full st-tgds.

3.5. Maximal Recovery

Although maximum recoveries exist for a large class of mappings, as the following result shows there are classes of practical interest for which the existence of maximum recoveries is not guaranteed. This result was previously reported by Riveros (2008) and the proof can be found in (Riveros, 2008; Arenas, Pérez, & Riveros, 2009)

PROPOSITION 3.5.1 (Arenas et al., 2008; Riveros, 2008). *There exist st-mappings specified by (1)* CQ-TO-CQ$^{\neq}$ *and (2)* CQ-TO-UCQ *st-dependencies, that have no maximum recoveries.*

To overcome this limitation, one has to look for a weaker notion. A straightforward relaxation is to consider not maximum but maximal recoveries. In this section, we report our initial results about maximal recoveries, providing a necessary and sufficient condition for the existence of maximal recoveries, and showing that the notion of maximal recovery strictly generalizes the notion of maximum recovery. In fact, we show that maximal recoveries exist for the larger class of st-mappings specified by FO-TO-UCQ$^{\neq}$ dependencies. This result shows that the notion of maximal recovery is a promising direction for further research.

Recall that for two recoveries $\mathcal{M}'$ and $\mathcal{M}''$ of a mapping $\mathcal{M}$, we say that $\mathcal{M}'$ is at least as informative as $\mathcal{M}''$ for $\mathcal{M}$, and write $\mathcal{M}'' \preceq_{\mathcal{M}} \mathcal{M}'$, if $\mathcal{M} \circ \mathcal{M}' \subseteq \mathcal{M} \circ \mathcal{M}''$. If $\mathcal{M}'' \preceq_{\mathcal{M}} \mathcal{M}'$ and $\mathcal{M}' \not\preceq_{\mathcal{M}} \mathcal{M}''$, then we say that $\mathcal{M}'$ is *more informative than* $\mathcal{M}''$ for $\mathcal{M}$, and we write $\mathcal{M}'' \prec_{\mathcal{M}} \mathcal{M}'$.

DEFINITION 3.5.2. *Let $\mathcal{M}'$ be a recovery of a mapping $\mathcal{M}$. We say that $\mathcal{M}'$ is a* maximal recovery *of $\mathcal{M}$, if there is no recovery $\mathcal{M}''$ of $\mathcal{M}$ such that $\mathcal{M}' \prec_{\mathcal{M}} \mathcal{M}''$.*

That is, $\mathcal{M}'$ is a maximal recovery of $\mathcal{M}$ if there is no other recovery that is more informative for $\mathcal{M}$ than $\mathcal{M}'$. In Section 3.1, we show that the notion of witness can be

used to characterize the existence of maximum recoveries. In the following lemma, we provide an alternative description of the notion of witness for an instance. We later relax this description to characterize the existence of maximal recoveries (see Definition 3.5.4). Recall that $\mathcal{M}^{-1}$ denotes the mapping $\{(J, I) \mid (I, J) \in \mathcal{M}\}$.

LEMMA 3.5.3. *Let $\mathcal{M}$ be a mapping from $\mathbf{R}_1$ to $\mathbf{R}_2$ and $I \in \mathrm{Inst}(\mathbf{R}_1)$. Then $J \in \mathrm{Inst}(\mathbf{R}_2)$ is a witness for I under $\mathcal{M}$ iff for every $J' \in \mathrm{Sol}_{\mathcal{M}}(I)$, it is the case that $\mathrm{Sol}_{\mathcal{M}^{-1}}(J) \subseteq \mathrm{Sol}_{\mathcal{M}^{-1}}(J')$.*

PROOF. ($\Rightarrow$) Assume that J is a witness for instance I under $\mathcal{M}$. We need to prove that for every $J' \in \mathrm{Sol}_{\mathcal{M}}(I)$, it is the case that $\mathrm{Sol}_{\mathcal{M}^{-1}}(J) \subseteq \mathrm{Sol}_{\mathcal{M}^{-1}}(J')$. Let $I' \in \mathrm{Sol}_{\mathcal{M}^{-1}}(J)$. Then we have that $J \in \mathrm{Sol}_{\mathcal{M}}(I')$ and, thus, $\mathrm{Sol}_{\mathcal{M}}(I) \subseteq \mathrm{Sol}_{\mathcal{M}}(I')$ since J is a witness for I under $\mathcal{M}$. Given that $J' \in \mathrm{Sol}_{\mathcal{M}}(I)$, we conclude that $J' \in \mathrm{Sol}_{\mathcal{M}}(I')$ and, hence, $I' \in \mathrm{Sol}_{\mathcal{M}^{-1}}(J')$.

($\Leftarrow$) Assume that for every instance $J' \in \mathrm{Sol}_{\mathcal{M}}(I)$, it is the case that $\mathrm{Sol}_{\mathcal{M}^{-1}}(J) \subseteq \mathrm{Sol}_{\mathcal{M}^{-1}}(J')$. We need to prove that J is a witness for I under $\mathcal{M}$, that is, given an instance I' of $\mathbf{R}_1$ such that $J \in \mathrm{Sol}_{\mathcal{M}}(I')$, we need to prove that $\mathrm{Sol}_{\mathcal{M}}(I) \subseteq \mathrm{Sol}_{\mathcal{M}}(I')$. Let $J' \in \mathrm{Sol}_{\mathcal{M}}(I)$. Then we have that $\mathrm{Sol}_{\mathcal{M}^{-1}}(J) \subseteq \mathrm{Sol}_{\mathcal{M}^{-1}}(J')$. Thus, given that $I' \in \mathrm{Sol}_{\mathcal{M}^{-1}}(J)$, we conclude that $I' \in \mathrm{Sol}_{\mathcal{M}^{-1}}(J')$, that is, $J' \in \mathrm{Sol}_{\mathcal{M}}(I')$. □

3.5.1. Characterizing Maximal Recoveries

In the following definition, we introduce a relaxation of the notion of witness based on the alternative description in Lemma 3.5.3. We called this relaxed notion *partial-witness*.

DEFINITION 3.5.4. *Let $\mathcal{M}$ be a mapping. Instance J is a* partial-witness for I under $\mathcal{M}$ *iff for every $J' \in \mathrm{Sol}_{\mathcal{M}}(I)$, it is not the case that $\mathrm{Sol}_{\mathcal{M}^{-1}}(J') \subsetneq \mathrm{Sol}_{\mathcal{M}^{-1}}(J)$.*

Notice that a partial-witness for an instance I is not necessarily a solution for I. If J is both a partial-witness and a solution for I under $\mathcal{M}$, then we say that J is a *partial-witness solution* for I under $\mathcal{M}$. The notion of partial-witness solution can be used to provide a necessary and sufficient condition for the existence of maximal recoveries (see

Theorem 3.5.7). It can also be used to provide a characterization of when a mapping $\mathcal{M}'$ is a maximal recovery of a mapping $\mathcal{M}$ (see Theorem 3.5.6). But before stating these results, we nee to introduce some notation and prove a lemma.

Recall that a recovery $\mathcal{M}'$ of a mapping $\mathcal{M}$ is reduced if for every $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$, it is the case that $I_2 \in \mathrm{dom}(\mathcal{M})$.

LEMMA 3.5.5. *If $\mathcal{M}'$ is a maximal recovery of $\mathcal{M}$, then $\mathcal{M}'$ is a reduced recovery of $\mathcal{M}$.*

PROOF. By contradiction, assume that $\mathcal{M}'$ is a maximal recovery of $\mathcal{M}$ and $\mathcal{M}'$ is not a reduced recovery of $\mathcal{M}$. Then there exists $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$ such that $I_2 \notin \mathrm{dom}(\mathcal{M})$.

Define mapping $\mathcal{M}'' \subseteq \mathcal{M}'$ as $\mathcal{M}'' = \{(J, I) \in \mathcal{M}' \mid I \in \mathrm{dom}(\mathcal{M})\}$. Given that $\mathcal{M}'$ is a recovery of $\mathcal{M}$, we have that $\mathcal{M}''$ is a recovery of $\mathcal{M}$. Moreover, $\mathcal{M} \circ \mathcal{M}'' \subsetneq \mathcal{M} \circ \mathcal{M}'$ since $\mathcal{M}'' \subseteq \mathcal{M}'$ and $(I_1, I_2) \notin \mathcal{M} \circ \mathcal{M}''$. Thus, we have that $\mathcal{M}' \prec_{\mathcal{M}} \mathcal{M}''$, which contradicts the fact that $\mathcal{M}'$ is a maximal recovery of $\mathcal{M}$. □

Let $\mathcal{M}$ be a mapping from a schema $\mathbf{R}_1$ to a schema $\mathbf{R}_2$ and $\mathcal{M}'$ a recovery of $\mathcal{M}$. Recall that $\mathrm{range}(\mathcal{M}')$ denotes the set $\{I \in \mathrm{Inst}(\mathbf{R}_1) \mid \text{there exists } J \in \mathrm{Inst}(\mathbf{R}_2) \text{ such that } (J, I) \in \mathcal{M}'\}$. We say that a mapping $\mathcal{M}^\star$ is a *contraction* of $\mathcal{M}'$ if $\mathcal{M}^\star \subseteq \mathcal{M}'$ and for every instance $I \in \mathrm{range}(\mathcal{M}')$, there exists a unique instance J of $\mathbf{R}_2$ such that $(J, I) \in \mathcal{M}^\star$. We note that $\mathcal{M}^\star$ is a recovery of $\mathcal{M}$, since $\mathcal{M}'$ is a recovery of $\mathcal{M}$, and $\mathcal{M}' \preceq_{\mathcal{M}} \mathcal{M}^\star$. The notion of contraction together with the notion of partial-witness is used in the following theorem to characterize when a mapping $\mathcal{M}'$ is a maximal recovery of a mapping $\mathcal{M}$.

THEOREM 3.5.6. *Let $\mathcal{M}$ be a mapping from a schema $\mathbf{R}_1$ to a schema $\mathbf{R}_2$, and $\mathcal{M}'$ a recovery of $\mathcal{M}$. Then the following conditions are equivalent:*

(1) *$\mathcal{M}'$ is a maximal recovery of $\mathcal{M}$,*
(2) *$\mathcal{M}'$ is a reduced recovery of $\mathcal{M}$, and there exists a contraction $\mathcal{M}^\star$ of $\mathcal{M}'$ such that $\mathcal{M}' \equiv_{\mathcal{M}} \mathcal{M}^\star$ and for every $(I_1, J) \in \mathcal{M}$ and $(J, I_2) \in \mathcal{M}^\star$, J is a partial-witness for I_2 under $\mathcal{M}$.*

PROOF. $(1) \Rightarrow (2)$ Assume that $\mathcal{M}'$ is a maximal recovery of $\mathcal{M}$. Then by Lemma 3.5.5, we know that $\mathcal{M}'$ is a reduced recovery of $\mathcal{M}$ and, thus, we only need to show that the second part of (2) holds. Let $\mathcal{M}^\star$ be an arbitrary contraction of $\mathcal{M}$. Since $\mathcal{M}'$ is a maximal recovery of $\mathcal{M}$ and $\mathcal{M}' \preceq_{\mathcal{M}} \mathcal{M}^\star$, we have that $\mathcal{M}' \equiv_{\mathcal{M}} \mathcal{M}^\star$. Next we show that for every $(I_1, J) \in \mathcal{M}$ and $(J, I_2) \in \mathcal{M}^\star$, J is a partial-witness for I_2 under $\mathcal{M}$.

For the sake of contradiction, assume that there exist $(I_1, J) \in \mathcal{M}$ and $(J, I_2) \in \mathcal{M}^\star$ such that J is not a partial-witness for I_2 under $\mathcal{M}$. Then there exists an instance $J^\star$ of $\mathbf{R}_2$ such that $J^\star \in \mathrm{Sol}_{\mathcal{M}}(I_2)$ and $\mathrm{Sol}_{\mathcal{M}^{-1}}(J^\star) \subsetneq \mathrm{Sol}_{\mathcal{M}^{-1}}(J)$. Let $\mathcal{M}''$ be a mapping defined as follows. Let $(J^\star, I_2)$ be the only tuple in $\mathcal{M}''$ where I_2 appears as the second argument. Moreover, for every instance I of $\mathbf{R}_1$ such that $I \neq I_2$, choose an arbitrary $J \in \mathrm{Sol}_{\mathcal{M}}(I)$ such that $(J, I) \in \mathcal{M}^\star$, and let (J, I) be the only tuple in $\mathcal{M}''$ where I appears as the second argument. Given that $I_2 \in \mathrm{Sol}_{\mathcal{M}^{-1}}(J^\star)$ and $\mathcal{M}^\star$ is a recovery of $\mathcal{M}$, we have that $\mathcal{M}''$ is a recovery of $\mathcal{M}$. Next we show that $\mathcal{M} \circ \mathcal{M}'' \subsetneq \mathcal{M} \circ \mathcal{M}^\star$.

First, we show that $\mathcal{M} \circ \mathcal{M}'' \subseteq \mathcal{M} \circ \mathcal{M}^\star$. Assume that $(I_3, I_4) \in \mathcal{M} \circ \mathcal{M}''$. Then there exists an instance J_1 of $\mathbf{R}_2$ such that $(I_3, J_1) \in \mathcal{M}$ and $(J_1, I_4) \in \mathcal{M}''$. If $I_4 \neq I_2$, then by definition of $\mathcal{M}''$ we have that $(J_1, I_4) \in \mathcal{M}^\star$ and, hence, $(I_3, I_4) \in \mathcal{M} \circ \mathcal{M}^\star$. If $I_4 = I_2$, then by definition of $\mathcal{M}''$ we have that $J_1 = J^\star$. Thus, we have that $(I_3, J^\star) \in \mathcal{M}$ and, hence, $I_3 \in \mathrm{Sol}_{\mathcal{M}^{-1}}(J^\star)$. Thus, given that $\mathrm{Sol}_{\mathcal{M}^{-1}}(J^\star) \subsetneq \mathrm{Sol}_{\mathcal{M}^{-1}}(J)$, we have that $I_3 \in \mathrm{Sol}_{\mathcal{M}^{-1}}(J)$. We conclude that $(I_3, I_4) \in \mathcal{M} \circ \mathcal{M}^\star$ since $I_4 = I_2$, $(I_3, J) \in \mathcal{M}$ and $(J, I_2) \in \mathcal{M}^\star$. Second, we show that $\mathcal{M} \circ \mathcal{M}^\star \not\subseteq \mathcal{M} \circ \mathcal{M}''$. Given that $\mathrm{Sol}_{\mathcal{M}^{-1}}(J^\star) \subsetneq \mathrm{Sol}_{\mathcal{M}^{-1}}(J)$, there exists $I_5 \in \mathrm{Sol}_{\mathcal{M}^{-1}}(J)$ such that $I_5 \notin \mathrm{Sol}_{\mathcal{M}^{-1}}(J^\star)$. Then we conclude that $(I_5, I_2) \in \mathcal{M} \circ \mathcal{M}^\star$ since $(I_5, J) \in \mathcal{M}$ and $(J, I_2) \in \mathcal{M}^\star$. Furthermore, we also conclude that $(I_5, I_2) \notin \mathcal{M} \circ \mathcal{M}''$ since $(I_5, J^\star) \notin \mathcal{M}$ and $(J^\star, I_2)$ is the only tuple in $\mathcal{M}''$ where I_2 appears as the second argument.

From the previous paragraph, we conclude that $\mathcal{M} \circ \mathcal{M}'' \subsetneq \mathcal{M} \circ \mathcal{M}^\star$ and, thus, $\mathcal{M}' \prec_{\mathcal{M}} \mathcal{M}''$ since $\mathcal{M}^\star \prec_{\mathcal{M}} \mathcal{M}''$ and $\mathcal{M}' \equiv_{\mathcal{M}} \mathcal{M}^\star$. Given that this contradicts the maximality of $\mathcal{M}'$, we conclude that our initial assumption does not hold and, therefore,

for every $(I_1, J) \in \mathcal{M}$ and $(J, I_2) \in \mathcal{M}^\star$, it should be the case that J is a partial-witness for I_2 under $\mathcal{M}$.

$(2) \Rightarrow (1)$ Assume that $\mathcal{M}'$ is a reduced recovery of $\mathcal{M}$ and $\mathcal{M}^\star$ is a contraction of $\mathcal{M}'$ such that, for every $(I_1, J) \in \mathcal{M}$ and $(J, I_2) \in \mathcal{M}^\star$, J is a partial-witness for I_2 under $\mathcal{M}$. Next we show that $\mathcal{M}^\star$ is a maximal recovery of $\mathcal{M}$. Given that $\mathcal{M}'$ is a recovery of $\mathcal{M}$, we have that $\mathcal{M}^\star$ is a recovery of $\mathcal{M}$. Thus, to prove that $\mathcal{M}^\star$ is a maximal recovery of $\mathcal{M}$, we only need to show that for every recovery $\mathcal{M}''$ of $\mathcal{M}$, if $\mathcal{M}^\star \preceq_{\mathcal{M}} \mathcal{M}''$, then $\mathcal{M}^\star \equiv_{\mathcal{M}} \mathcal{M}''$. This condition is equivalent to:

$$\text{if } \mathcal{M}'' \not\preceq_{\mathcal{M}} \mathcal{M}^\star, \text{ then } \mathcal{M}^\star \not\preceq_{\mathcal{M}} \mathcal{M}''.$$

Thus, we assume that $\mathcal{M} \circ \mathcal{M}^\star \not\subseteq \mathcal{M} \circ \mathcal{M}''$, and we show that this implies $\mathcal{M} \circ \mathcal{M}'' \not\subseteq \mathcal{M} \circ \mathcal{M}^\star$. Let $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}^\star$ such that $(I_1, I_2) \notin \mathcal{M} \circ \mathcal{M}''$. Then we have that there exists a partial-witness $J^\star$ for I_2 under $\mathcal{M}$ such that $(I_1, J^\star) \in \mathcal{M}$ and $(J^\star, I_2) \in \mathcal{M}^\star$. Given that $\mathcal{M}'$ is a reduced recovery of $\mathcal{M}$ and $\mathcal{M}' \equiv_{\mathcal{M}} \mathcal{M}^\star$, we have that $\mathcal{M}^\star$ is a reduced recovery of $\mathcal{M}$ and, therefore, $I_2 \in \mathrm{dom}(\mathcal{M})$. Thus, given that $\mathcal{M}''$ is a recovery of $\mathcal{M}$, there exists an instance $J \in \mathrm{Sol}_{\mathcal{M}}(I_2)$ such that $(J, I_2) \in \mathcal{M}''$. Thus, given that $J^\star$ is a partial-witness for I_2 under $\mathcal{M}$, it is not the case that $\mathrm{Sol}_{\mathcal{M}^{-1}}(J) \subsetneq \mathrm{Sol}_{\mathcal{M}^{-1}}(J^\star)$.

Given that $(I_1, I_2) \notin \mathcal{M} \circ \mathcal{M}''$ and $(J, I_2) \in \mathcal{M}''$, we have that $I_1 \notin \mathrm{Sol}_{\mathcal{M}^{-1}}(J)$ and, therefore, $\mathrm{Sol}_{\mathcal{M}^{-1}}(J^\star) \not\subseteq \mathrm{Sol}_{\mathcal{M}^{-1}}(J)$ since $I_1 \in \mathrm{Sol}_{\mathcal{M}^{-1}}(J^\star)$. Thus, from the fact that $\mathrm{Sol}_{\mathcal{M}^{-1}}(J) \subsetneq \mathrm{Sol}_{\mathcal{M}^{-1}}(J^\star)$ does not hold, we conclude that $\mathrm{Sol}_{\mathcal{M}^{-1}}(J) \not\subseteq \mathrm{Sol}_{\mathcal{M}^{-1}}(J^\star)$ and, hence, there exists $I_3 \in \mathrm{Sol}_{\mathcal{M}^{-1}}(J)$ such that $I_3 \notin \mathrm{Sol}_{\mathcal{M}^{-1}}(J^\star)$. We have that $(I_3, I_2) \in \mathcal{M} \circ \mathcal{M}''$, since $(J, I_2) \in \mathcal{M}''$, and $(I_3, I_2) \notin \mathcal{M} \circ \mathcal{M}^\star$, since $(I_3, J^\star) \notin \mathcal{M}$ and $(J^\star, I_2)$ is the only tuple in $\mathcal{M}^\star$ where I_2 appears as the second argument (since $\mathcal{M}^\star$ is a contraction of $\mathcal{M}'$). We conclude that $\mathcal{M} \circ \mathcal{M}'' \not\subseteq \mathcal{M} \circ \mathcal{M}^\star$ and, hence, $\mathcal{M}^\star \not\preceq_{\mathcal{M}} \mathcal{M}''$.

From the previous paragraph, we have that $\mathcal{M}^\star$ is a maximal recovery of $\mathcal{M}$. Thus, given that $\mathcal{M}^\star \equiv_{\mathcal{M}} \mathcal{M}'$, we conclude that $\mathcal{M}'$ is a maximal recovery of $\mathcal{M}$. □

We can now state our characterization of the existence of maximal recoveries.

THEOREM 3.5.7. *$\mathcal{M}$ has a maximal recovery iff for every $I \in \mathrm{dom}(\mathcal{M})$, there exists a partial-witness solution for I under $\mathcal{M}$.*

PROOF. ($\Rightarrow$) Assume that $\mathcal{M}'$ is a maximal recovery of $\mathcal{M}$. Then by Theorem 3.5.6, there exists a contraction $\mathcal{M}^\star$ of $\mathcal{M}$ such that $\mathcal{M}' \equiv_{\mathcal{M}} \mathcal{M}^\star$ and for every $(I_1, J) \in \mathcal{M}$ and $(J, I_2) \in \mathcal{M}^\star$, J is a partial-witness for I under $\mathcal{M}$.

Let $I \in \mathrm{dom}(\mathcal{M})$. Given that $\mathcal{M}^\star$ is a contraction of $\mathcal{M}$, we have that $\mathcal{M}^\star$ is a recovery of $\mathcal{M}$ and, hence, $(I, I) \in \mathcal{M} \circ \mathcal{M}^\star$. Thus, there exists an instance J such that $(I, J) \in \mathcal{M}$, $(J, I) \in \mathcal{M}^\star$ and J is a partial-witness for I under $\mathcal{M}$. We conclude that there exists $J \in \mathrm{Sol}_{\mathcal{M}}(I)$ such that J is a partial-witness for I under $\mathcal{M}$.

($\Leftarrow$) Assume that for every instance $I \in \mathrm{dom}(\mathcal{M})$, there exists $J_I \in \mathrm{Sol}_{\mathcal{M}}(I)$ such that J_I is a partial-witness for I under $\mathcal{M}$, and let $\mathcal{M}^\star$ be a mapping defined as $\{(J_I, I) \mid I \in \mathrm{dom}(\mathcal{M})\}$. It is easy to see that $\mathcal{M}^\star$ is a reduced recovery of $\mathcal{M}$. Furthermore, given that the only contraction of $\mathcal{M}^\star$ is $\mathcal{M}^\star$ itself, and that for every $(J, I) \in \mathcal{M}^\star$, J is a partial-witness for I under $\mathcal{M}$, we conclude from Theorem 3.5.6 that $\mathcal{M}^\star$ is a maximal recovery of $\mathcal{M}$. □

3.5.2. Existence of maximal recoveries beyond FO-TO-CQ

The following theorem identifies an important class of st-mappings for which the existence of maximal recoveries is guaranteed, and also shows that the notion of maximal recovery strictly generalizes the notion of maximum recovery (see Proposition 4.3.10).

THEOREM 3.5.8. *If $\mathcal{M}$ is an st-mapping specified by a set of* FO-TO-UCQ$^{\neq}$ *st-dependencies, then $\mathcal{M}$ has a maximal recovery.*

To prove theorem, we need some technical results. We start with two simple lemmas.

LEMMA 3.5.9. *Let $\mathcal{M}$ be an st-mapping specified by a set of* FO-TO-FO *dependencies. Let J be a target instance and h a bijection that has as domain the set $\mathrm{dom}(J)$ and is the identity over $\mathrm{dom}(J) \cap \mathbf{C}$. If for a source instance I we have $(I, J) \in \mathcal{M}$, then $(I, h(J)) \in \mathcal{M}$.*

PROOF. Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ with Σ a set of FO-TO-FO dependencies, and assume that $(I, J) \models \Sigma$. By the safety condition imposed on FO-TO-FO dependencies, to prove the lemma we have to show that, for every dependency $\varphi(x_1, \ldots, x_k) \rightarrow \psi(x_1, \ldots, x_k)$ in Σ and tuple $\bar{a} \in \mathrm{dom}(I)^k$, if $I \models \varphi(\bar{a})$ then $\bar{a} \in \mathrm{dom}(h(J))^k$ and $h(J) \models \psi(\bar{a})$. Assume that $I \models \varphi(\bar{a})$. Since $\bar{a} \in \mathrm{dom}(I)^k$ we know that $\bar{a}$ is a tuple of elements from $\mathbf{C}$, and then, since $\bar{a} \in \mathrm{dom}(J)^k$ and h is the identity over $\mathrm{dom}(J) \cap \mathbf{C}$, we obtain that $\bar{a} \in \mathrm{dom}(h(J))^k$. Note that h is an isomorphism from J to $h(J)$ such that $h(\bar{a}) = \bar{a}$. Thus, given that $J \models \psi(\bar{a})$, we obtain that $h(J) \models \psi(\bar{a})$. □

LEMMA 3.5.10. *Let $\mathcal{M}$ be an st-mapping specified by a set of* FO-TO-UCQ$^{\neq}$ *dependencies. Let J be a target instance and h a bijection that has as domain the set* $\mathrm{dom}(J)$ *and is the identity over* $\mathrm{dom}(J) \cap \mathbf{C}$. *If $h(J) \subseteq J'$, then* $\mathrm{Sol}_{\mathcal{M}^{-1}}(J) \subseteq \mathrm{Sol}_{\mathcal{M}^{-1}}(J')$.

PROOF. Let $(I, J) \in \mathcal{M}$, we have to show that $(I, J') \in \mathcal{M}$. From Lemma 3.5.9 and since $(I, J) \in \mathcal{M}$, we obtain that $(I, h(J)) \in \mathcal{M}$. Then since $\mathcal{M}$ is closed-up on the right and $h(J) \subseteq J'$, we have that $(I, J') \in \mathcal{M}$. We have shown that, if $I \in \mathrm{Sol}_{\mathcal{M}^{-1}}(J)$ then $I \in \mathrm{Sol}_{\mathcal{M}^{-1}}(J')$, which proves that $\mathrm{Sol}_{\mathcal{M}^{-1}}(J) \subseteq \mathrm{Sol}_{\mathcal{M}^{-1}}(J')$. □

We now introduce some terminology. Let $\mathcal{M}$ be an st-mapping and I a source instance. Then

$$\mathrm{MinSol}_{\mathcal{M}}(I) = \{J \in \mathrm{Sol}_{\mathcal{M}}(I) \mid \text{ there is no } J' \in \mathrm{Sol}_{\mathcal{M}}(I) \text{ such that } J' \subsetneq J\},$$

that is, $\mathrm{MinSol}_{\mathcal{M}}(I)$ is the set of *minimal solutions* of I under $\mathcal{M}$. Note that for every $J' \in \mathrm{Sol}_{\mathcal{M}}(I)$, there exists $J \in \mathrm{MinSol}_{\mathcal{M}}(I)$ such that $J \subseteq J'$. Let $\sim_{\mathrm{dom}(I)}$ be an equivalence relation on $\mathrm{MinSol}_{\mathcal{M}}(I)$ such that, $J_1 \sim_{\mathrm{dom}(I)} J_2$ if and only if there exists a bijection $h : \mathrm{dom}(J_1) \rightarrow \mathrm{dom}(J_2)$ such that h is the identity on $\mathrm{dom}(I)$ and $h(J_1) = J_2$. Let $\mathrm{MinSol}^{\star}_{\mathcal{M}}(I)$ be the set that results from $\mathrm{MinSol}_{\mathcal{M}}(I)$ by selecting a single representative of each equivalence class defined by $\sim_{\mathrm{dom}(I)}$.

We now show that if a mapping $\mathcal{M}$ is specified by a set of FO-TO-FO dependencies, then in every equivalence class defined by $\sim_{\mathrm{dom}(I)}$ on $\mathrm{MinSol}_{\mathcal{M}}(I)$, there exists an instance

J such that $\mathrm{dom}(J) \cap \mathbf{C} \subseteq \mathrm{dom}(I)$. Let $J_1 \in \mathrm{MinSol}_{\mathcal{M}}(I)$ and h a function that assigns to every element in $(\mathrm{dom}(J_1) \cap \mathbf{C}) \setminus \mathrm{dom}(I)$ a fresh element in $\mathbf{N}$ and is the identity elsewhere. Note that function $h : \mathrm{dom}(J_1) \to \mathrm{dom}(h(J_1))$ is a bijection that is the identity on $\mathrm{dom}(I)$ and, hence, $J_1 \sim_{\mathrm{dom}(I)} h(J_1)$ and $\mathrm{dom}(h(J_1)) \cap \mathbf{C} \subseteq \mathrm{dom}(I)$. Next we show that $h(J_1) \in \mathrm{MinSol}_{\mathcal{M}}(I)$, which proves that in every equivalence class defined by $\sim_{\mathrm{dom}(I)}$ on $\mathrm{MinSol}_{\mathcal{M}}(I)$, there exists an instance J such that $\mathrm{dom}(J) \cap \mathbf{C} \subseteq \mathrm{dom}(I)$. Given that h is a bijection that is the identity on $\mathrm{dom}(I)$, $(I, J_1) \in \mathcal{M}$ and $\mathcal{M}$ is specified by a set of FO-TO-FO dependencies, we have that $(h(I), h(J_1)) = (I, h(J_1)) \in \mathcal{M}$ and, therefore, $h(J_1) \in \mathrm{Sol}_{\mathcal{M}}(I)$. Next we show that $h(J_1) \in \mathrm{MinSol}_{\mathcal{M}}(I)$. On the contrary, assume that $h(J_1)$ is not minimal, and let $J_2 \in \mathrm{Sol}_{\mathcal{M}}(I)$ be an instance such that $J_2 \subsetneq h(J_1)$. Consider then bijection h^{-1} and instance $h^{-1}(J_2)$. It is immediate that $h^{-1}(J_2) \subsetneq J_1$. Given that h^{-1} is a bijection that is the identity on $\mathrm{dom}(I)$, $(I, J_2) \in \mathcal{M}$ and $\mathcal{M}$ is specified by a set of FO-TO-FO dependencies, we conclude that $(h^{-1}(I), h^{-1}(J_2)) = (I, h^{-1}(J_2)) \in \mathcal{M}$ and, thus, $h^{-1}(J_2) \in \mathrm{Sol}_{\mathcal{M}}(I)$. This contradicts the fact that J_1 is minimal in $\mathrm{Sol}_{\mathcal{M}}(I)$. We have shown that, in every equivalence class defined by $\sim_{\mathrm{dom}(I)}$, there exists a J such that $\mathrm{dom}(J) \cap \mathbf{C} \subseteq \mathrm{dom}(I)$, and then, we can safely assume that for every instance J in $\mathrm{MinSol}^{\star}_{\mathcal{M}}(I)$, it holds that $\mathrm{dom}(J) \cap \mathbf{C} \subseteq \mathrm{dom}(I)$.

Recall that, by definition of $\mathrm{MinSol}_{\mathcal{M}}(I)$, we have that for every $J \in \mathrm{Sol}_{\mathcal{M}}(I)$, there exists $J_1 \in \mathrm{MinSol}_{\mathcal{M}}(I)$ such that $J_1 \subseteq J$. Thus, given that for every $J_1 \in \mathrm{MinSol}_{\mathcal{M}}(I)$, there exists $J^{\star} \in \mathrm{MinSol}^{\star}_{\mathcal{M}}(I)$ such that $J_1 \sim_{\mathrm{dom}(I)} J^{\star}$, and we have assumed that for every instance $J_2 \in \mathrm{MinSol}^{\star}_{\mathcal{M}}(I)$, it holds that $\mathrm{dom}(J_2) \cap \mathbf{C} \subseteq \mathrm{dom}(I)$ (by the previous paragraph), we conclude that for every $J \in \mathrm{Sol}_{\mathcal{M}}(I)$, there exists $J^{\star} \in \mathrm{MinSol}^{\star}_{\mathcal{M}}(I)$ and a bijection h that is the identity on $\mathrm{dom}(J^{\star}) \cap \mathbf{C} \subseteq \mathrm{dom}(I)$, such that $h(J^{\star}) \subseteq J$. Through the rest of this proof, we write $J \to J'$ to denote the fact that there exists a bijection h that is the identity on $\mathrm{dom}(J) \cap \mathbf{C}$ such that $h(J) \subseteq J'$. Notice that, we know by Lemma 3.5.10 that if $\mathcal{M}$ is specified by a set of FO-TO-UCQ$^{\neq}$ dependencies and $J \to J'$, then $\mathrm{Sol}_{\mathcal{M}^{-1}}(J) \subseteq \mathrm{Sol}_{\mathcal{M}^{-1}}(J')$.

We need an additional lemma in order to prove Theorem 3.5.8.

LEMMA 3.5.11. *Let $\mathcal{M}$ be an st-mapping specified by a set of* FO-TO-UCQ$^{\neq}$ *dependencies. Then for every source instance I, the set* $\mathrm{MinSol}^{\star}_{\mathcal{M}}(I)$ *is finite.*

PROOF. Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where Σ a set of FO-TO-UCQ$^{\neq}$ dependencies. To prove the lemma, we show by using a simple combinatorial argument that for every instance I, there exists an integer k such that every element in $\mathrm{MinSol}_{\mathcal{M}}(I)$ has at most k tuples. From this fact, it is straightforward to conclude that $\mathrm{MinSol}^{\star}_{\mathcal{M}}(I)$ is a finite set.

In this proof, we use $\mathrm{na}(\psi(\bar{x}))$ to denote the number of atoms that appear in UCQ$^{\neq}$-formula $\psi(\bar{x})$, $|\bar{x}|$ to denote the length of $\bar{x}$ and $|K|$ to denote the total number of tuples in an instance K of a schema $\mathbf{R}$, that is, $|K| = \sum_{R \in \mathbf{R}} |R^K|$, where $|R^K|$ is the number of tuples in R^K. Furthermore, let $m = \max\{|\bar{x}| \mid \varphi(\bar{x}) \rightarrow \psi(\bar{x}) \in \Sigma\}$, and $\ell = \max\{\mathrm{na}(\psi(\bar{x})) \mid \varphi(\bar{x}) \rightarrow \psi(\bar{x}) \in \Sigma\}$.

Let I be a source instance and $J \in \mathrm{Sol}_{\mathcal{M}}(I)$. We now show that, if $|J| > (|\Sigma| \cdot |\mathrm{dom}(I)|^m \cdot \ell)$, then $J \notin \mathrm{MinSol}_{\mathcal{M}}(I)$. In what follows, we describe a procedure to construct a target instance J' such that $(I, J) \in \mathcal{M}$ and $J' \subsetneq J$. Start with $J' = \emptyset$. For every dependency $\varphi(\bar{x}) \rightarrow \psi(\bar{x}) \in \Sigma$, and for every tuple $\bar{a}$ such that $I \models \varphi(\bar{a})$, choose a sub-instance J_1 of J such that $|J_1| \leq \ell$ and $J_1 \models \psi(\bar{a})$, and add the tuples of J_1 to J'. It is clear that $(I, J') \models \Sigma$ and $J' \subseteq J$. Note that, for every $\varphi(\bar{x}) \rightarrow \psi(\bar{x}) \in \Sigma$, there are at most $|\mathrm{dom}(I)|^m$ tuples $\bar{a}$ such that $I \models \varphi(\bar{a})$, and that $\psi(\bar{a})$ is composed by at most ℓ atoms. Then $|J'| \leq (|\Sigma| \cdot |\mathrm{dom}(I)|^m \cdot \ell)$, which implies that $J' \subsetneq J$. We conclude that J is not a minimal solution for I. □

We are ready to prove Theorem 3.5.8.

PROOF OF THEOREM 3.5.8. Let $\mathcal{M}$ be an st-mapping specified by FO-TO-UCQ$^{\neq}$ dependencies. By Theorem 3.5.7, to prove that $\mathcal{M}$ has a maximal recovery, we have to show that every instance $I \in \mathrm{dom}(\mathcal{M})$ has a partial-witness solution under $\mathcal{M}$. Recall that instance J is a partial-witness solution for I if for every $J' \in \mathrm{Sol}_{\mathcal{M}}(I)$, it is not the case that $\mathrm{Sol}_{\mathcal{M}^{-1}}(J') \subsetneq \mathrm{Sol}_{\mathcal{M}^{-1}}(J)$. Let I be an instance in $\mathrm{dom}(\mathcal{M})$ and assume that

$\mathrm{MinSol}^{\star}_{\mathcal{M}}(I) = \{J_1, J_2, \ldots, J_n\}$ ($\mathrm{MinSol}^{\star}_{\mathcal{M}}(I)$ is finite by Lemma 3.5.11). We claim that an instance in $\mathrm{MinSol}^{\star}_{\mathcal{M}}(I)$ is a partial-witness solution for I.

For the sake of contradiction, assume that for every $i \in [1, n]$, $J_i \in \mathrm{MinSol}^{\star}_{\mathcal{M}}(I)$ is not a partial-witness solution for I. Given that J_1 is not a partial-witness solution for I, there exists $J'_1 \in \mathrm{Sol}_{\mathcal{M}}(I)$ such that $\mathrm{Sol}_{\mathcal{M}^{-1}}(J'_1) \subsetneq \mathrm{Sol}_{\mathcal{M}^{-1}}(J_1)$. Given that $J'_1 \in \mathrm{Sol}_{\mathcal{M}}(I)$, there exists $J_i \in \mathrm{MinSol}^{\star}_{\mathcal{M}}(I)$ such that $J_i \rightarrow J'_1$. Note that, by Lemma 3.5.10 we have $\mathrm{Sol}_{\mathcal{M}^{-1}}(J_i) \subseteq \mathrm{Sol}_{\mathcal{M}^{-1}}(J'_1)$ and, therefore, $J_i \neq J_1$ (otherwise it could not be the case that $\mathrm{Sol}_{\mathcal{M}^{-1}}(J'_1) \subsetneq \mathrm{Sol}_{\mathcal{M}^{-1}}(J_1)$). Then there exists an instance $J_i \in \mathrm{MinSol}^{\star}_{\mathcal{M}}(I)$, with $i \in [2, n]$, such that $J_i \rightarrow J'_1$. Without loss of generality, we assume that $i = 2$, and then $J_2 \rightarrow J'_1$. By Lemma 3.5.10, we have that $\mathrm{Sol}_{\mathcal{M}^{-1}}(J_2) \subseteq \mathrm{Sol}_{\mathcal{M}^{-1}}(J'_1)$, which implies that $\mathrm{Sol}_{\mathcal{M}^{-1}}(J_2) \subsetneq \mathrm{Sol}_{\mathcal{M}^{-1}}(J_1)$ since $\mathrm{Sol}_{\mathcal{M}^{-1}}(J'_1) \subsetneq \mathrm{Sol}_{\mathcal{M}^{-1}}(J_1)$. Now, since we are assuming that J_2 is not a partial-witness solution for I, there exists $J'_2 \in \mathrm{Sol}_{\mathcal{M}}(I)$ such that $\mathrm{Sol}_{\mathcal{M}^{-1}}(J'_2) \subsetneq \mathrm{Sol}_{\mathcal{M}^{-1}}(J_2)$. As above, it could not be the case that $J_2 \rightarrow J'_2$ since this implies that $\mathrm{Sol}_{\mathcal{M}^{-1}}(J_2) \subseteq \mathrm{Sol}_{\mathcal{M}^{-1}}(J'_2)$. Furthermore, it cannot be the case that $J_1 \rightarrow J'_2$, otherwise we would conclude that $\mathrm{Sol}_{\mathcal{M}^{-1}}(J_1) \subsetneq \mathrm{Sol}_{\mathcal{M}^{-1}}(J_1)$ since $\mathrm{Sol}_{\mathcal{M}^{-1}}(J'_2) \subsetneq \mathrm{Sol}_{\mathcal{M}^{-1}}(J_2) \subsetneq \mathrm{Sol}_{\mathcal{M}^{-1}}(J_1)$. Then there exists an instance $J_i \in \mathrm{MinSol}^{\star}_{\mathcal{M}}(I)$, with $i \in [3, n]$, such that $J_i \rightarrow J'_2$. Again, without loss of generality, we assume that $i = 3$. By Lemma 3.5.10, we conclude that $\mathrm{Sol}_{\mathcal{M}^{-1}}(J_3) \subseteq \mathrm{Sol}_{\mathcal{M}^{-1}}(J'_2)$ and, thus, we have that $\mathrm{Sol}_{\mathcal{M}^{-1}}(J_3) \subsetneq \mathrm{Sol}_{\mathcal{M}^{-1}}(J_2) \subsetneq \mathrm{Sol}_{\mathcal{M}^{-1}}(J_1)$ since $\mathrm{Sol}_{\mathcal{M}^{-1}}(J'_2) \subsetneq \mathrm{Sol}_{\mathcal{M}^{-1}}(J_2)$ and $\mathrm{Sol}_{\mathcal{M}^{-1}}(J_2) \subsetneq \mathrm{Sol}_{\mathcal{M}^{-1}}(J_1)$. If we continue with this process, we conclude that:

$$\mathrm{Sol}_{\mathcal{M}^{-1}}(J_n) \subsetneq \mathrm{Sol}_{\mathcal{M}^{-1}}(J_{n-1}) \subsetneq \cdots \subsetneq \mathrm{Sol}_{\mathcal{M}^{-1}}(J_3) \subsetneq \mathrm{Sol}_{\mathcal{M}^{-1}}(J_2) \subsetneq \mathrm{Sol}_{\mathcal{M}^{-1}}(J_1).$$

But since we are assuming that J_n is not a partial-witness solution for I, there exists $J'_n \in \mathrm{Sol}_{\mathcal{M}}(I)$ such that $\mathrm{Sol}_{\mathcal{M}^{-1}}(J'_n) \subsetneq \mathrm{Sol}_{\mathcal{M}^{-1}}(J_n)$. Given that $J'_n \in \mathrm{Sol}_{\mathcal{M}}(I)$, it must be the case that $J_i \rightarrow J'_n$ for an instance $J_i \in \mathrm{MinSol}^{\star}_{\mathcal{M}}(I)$. No matter what J_i we choose this time, we obtain a contradiction. This concludes the proof of the theorem. □

The last result of this section shows that there exist mappings that do not have maximal recoveries.

PROPOSITION 3.5.12. *There exists an st-mapping $\mathcal{M}$ specified by an* FO*-formula that has no maximal recovery.*

PROOF. Let $\mathbf{R}_1 = \{P(\cdot)\}$, $\mathbf{R}_2 = \{R(\cdot)\}$ and $\mathcal{M}$ be the mapping from $\mathbf{R}_1$ to $\mathbf{R}_2$ specified by FO-sentence:

$$\forall x\,(P(x) \rightarrow \neg R(x)).$$

Next we show that if I is an instance of $\mathbf{R}_1$ such that $P^I = \{a\}$, there is no $J \in \mathrm{Sol}_{\mathcal{M}}(I)$ such that J is a partial-witness for I under $\mathcal{M}$. Let J be an arbitrary solution for I under $\mathcal{M}$. Assume that b is an element of $\mathbf{C}$ such that $b \notin \mathrm{dom}(J)$ and $b \neq a$. Furthermore, assume and that J' is an instance of $\mathbf{R}_2$ such that $R^{J'} = R^J \cup \{b\}$. It is easy to see that $J' \in \mathrm{Sol}_{\mathcal{M}}(I)$ since $J \in \mathrm{Sol}_{\mathcal{M}}(I)$ and $b \neq a$. Next we show that $\mathrm{Sol}_{\mathcal{M}^{-1}}(J') \subsetneq \mathrm{Sol}_{\mathcal{M}^{-1}}(J)$.

First, assume that $I_1 \in \mathrm{Sol}_{\mathcal{M}^{-1}}(J')$. Then $(I_1, J') \models \forall x\,(P(x) \rightarrow \neg R(x))$, from which we conclude that $(I_1, J) \models \forall x\,(P(x) \rightarrow \neg R(x))$ since $R^J \subsetneq R^{J'}$. Thus, we have that $I_1 \in \mathrm{Sol}_{\mathcal{M}^{-1}}(J)$. Second, we show that $\mathrm{Sol}_{\mathcal{M}^{-1}}(J) \not\subseteq \mathrm{Sol}_{\mathcal{M}^{-1}}(J')$. Let I_2 be an instance of $\mathbf{R}_1$ such that $P^I = \{b\}$. It is easy to see that $(I_2, J) \models \forall x\,(P(x) \rightarrow \neg R(x))$, since $b \notin \mathrm{dom}(J)$, and $(I_2, J') \not\models \forall x\,(P(x) \rightarrow \neg R(x))$, since $b \in (P^{I_2} \cap R^{J'})$. We conclude that $I_2 \in \mathrm{Sol}_{\mathcal{M}^{-1}}(J)$ and $I_2 \notin \mathrm{Sol}_{\mathcal{M}^{-1}}(J')$.

From the previous paragraphs, we have that there is no $J \in \mathrm{Sol}_{\mathcal{M}}(I)$ such that J is a partial-witness for I under $\mathcal{M}$. Thus, from Theorem 3.5.7 we conclude that $\mathcal{M}$ does not have a maximal recovery. □

In the above proof we use an st-mapping $\mathcal{M}$ specified by FO-dependency $P(x) \rightarrow \neg R(x)$. We note that formula $P(x) \rightarrow \neg R(x)$ does not satisfy the safety condition imposed on FO-TO-FO dependencies since $\neg R(x)$ is not domain-independent. It is left as a topic for further research, whether maximal recoveries exist for every st-mapping specified by a set of FO-TO-FO dependencies satisfying the safety condition.

4. QUERY LANGUAGE-BASED INVERSES OF SCHEMA MAPPINGS

The main goal of this chapter is to develop mapping languages with good properties for inverting schema mappings. To this end, in Section 4.1 we propose a query language-based formalization of the notion of recovering sound information. We use this formalization in the following section to study the previous notions of inverse (Fagin, 2007; Fagin, Kolaitis, Popa, & Tan, 2008) as well as the notion of maximum recovery proposed in Chapter 3, in terms of their capacity to retrieve sound information. This study gives a new perspective to these notions and, in particular, it allows to define notions of inverse that depend on the query language used to retrieve sound information. But more importantly, this idea of having a query language as parameter is used in Section 4.3 to find a mapping language that is *closed* under inversion.

4.1. Recovering Sound Information w.r.t. a Query Language: The Notions of $\mathcal{C}$-Recovery and $\mathcal{C}$-Maximum Recovery

One of the motivations for the notion of recovery that we propose in Chapter 3 was to identify mappings that recover sound information. However, strictly speaking, we did not formalize the concept of retrieving correct information. A simple way to formulate this notion is in terms of a query language. Let $\mathcal{M}$ be a mapping from a schema $\mathbf{R}_1$ to a schema $\mathbf{R}_2$, $\mathcal{M}'$ a mapping from $\mathbf{R}_2$ to $\mathbf{R}_1$ and Q a query over $\mathbf{R}_1$. Then we say that $\mathcal{M}'$ *recovers sound information for* $\mathcal{M}$ *under* Q if for every instance $I \in \mathrm{dom}(\mathcal{M})$, it holds that:

$$\mathrm{certain}_{\mathcal{M} \circ \mathcal{M}'}(Q, I) \subseteq Q(I).$$

Thus, we have that $\mathcal{M}'$ recovers sound information for $\mathcal{M}$ under Q if for every instance $I \in \mathrm{dom}(\mathcal{M})$, by posing query Q against the space of solutions for I under $\mathcal{M} \circ \mathcal{M}'$, one can only recover tuples that are already in the evaluation of Q over I. In the following definition, we extend the notion of recovering sound information to the case of a class of queries.

DEFINITION 4.1.1. *Let $\mathcal{C}$ be a class of queries, $\mathbf{R}_1$ and $\mathbf{R}_2$ schemas, $\mathcal{M}$ a mapping from $\mathbf{R}_1$ to $\mathbf{R}_2$ and $\mathcal{M}'$ a mapping from $\mathbf{R}_2$ to $\mathbf{R}_1$. Then $\mathcal{M}'$ is a* $\mathcal{C}$-recovery *of $\mathcal{M}$ if for every query $Q \in \mathcal{C}$ over $\mathbf{R}_1$ and instance $I \in \mathrm{dom}(\mathcal{M})$, it holds that*

$$\mathrm{certain}_{\mathcal{M} \circ \mathcal{M}'}(Q, I) \subseteq Q(I).$$

A natural question at this point is whether one can compare mappings according to their ability to recover sound information. It turns out that there is simple and natural way to do this. Assume that $\mathcal{M}_1$ and $\mathcal{M}_2$ are mappings that recover sound information for $\mathcal{M}$ under a query Q. Then we say that $\mathcal{M}_2$ *recovers as much information as $\mathcal{M}_1$ does for $\mathcal{M}$ under Q*, denoted by $\mathcal{M}_1 \preceq^Q_{\mathcal{M}} \mathcal{M}_2$, if for every instance $I \in \mathrm{dom}(\mathcal{M})$, it holds that:

$$\mathrm{certain}_{\mathcal{M} \circ \mathcal{M}_1}(Q, I) \subseteq \mathrm{certain}_{\mathcal{M} \circ \mathcal{M}_2}(Q, I).$$

Thus, we have that $\mathcal{M}_1 \preceq^Q_{\mathcal{M}} \mathcal{M}_2$ if for every instance $I \in \mathrm{dom}(\mathcal{M})$, every tuple that is retrieved by posing query Q against the space of solutions for I under $\mathcal{M} \circ \mathcal{M}_1$ is also retrieved by posing this query over the space of solutions for I under $\mathcal{M} \circ \mathcal{M}_2$. In the following definition, we extend the pre-order $\preceq^Q_{\mathcal{M}}$ to the case of a class of queries.

DEFINITION 4.1.2. *Let $\mathcal{C}$ be a class of queries, $\mathbf{R}_1$ and $\mathbf{R}_2$ schemas, $\mathcal{M}$ a mapping from $\mathbf{R}_1$ to $\mathbf{R}_2$ and $\mathcal{M}'$, $\mathcal{M}''$ $\mathcal{C}$-recoveries of $\mathcal{M}$. Then $\mathcal{M}''$* recovers as much information as $\mathcal{M}'$ does for $\mathcal{M}$ under $\mathcal{C}$, *denoted by $\mathcal{M}' \preceq^{\mathcal{C}}_{\mathcal{M}} \mathcal{M}''$, if $\mathcal{M}' \preceq^Q_{\mathcal{M}} \mathcal{M}''$ for every query $Q \in \mathcal{C}$ over $\mathbf{R}_1$.*

Given a class of queries $\mathcal{C}$, the notions introduced in this section can be used to define a mapping that is as good as any other mapping for retrieving sound information according to $\mathcal{C}$.

DEFINITION 4.1.3. *Let $\mathcal{C}$ be a class of queries and $\mathcal{M}'$ a $\mathcal{C}$-recovery of a mapping $\mathcal{M}$. Then $\mathcal{M}'$ is a* $\mathcal{C}$-maximum recovery *of $\mathcal{M}$ if for every $\mathcal{C}$-recovery $\mathcal{M}''$ of $\mathcal{M}$, it is the case that $\mathcal{M}'' \preceq^{\mathcal{C}}_{\mathcal{M}} \mathcal{M}'$.*

4.1.1. On the existence of $\mathcal{C}$-maximum recoveries

In this section, we provide a necessary and sufficient condition for the existence of $\mathcal{C}$-maximum recoveries, given any class of queries $\mathcal{C}$. The tools provided in this section will play a central role in the development of a good mapping language for inverting schema mappings. Let $\mathcal{M}$ be a mapping, $I \in \mathrm{dom}(\mathcal{M})$, and Q a query. We start our study by defining a set of tuples $\mathrm{Inf}_{\mathcal{M}}(Q, I)$ that captures the information that can be recovered for I under $\mathcal{M}$ by using query Q:

$$\mathrm{Inf}_{\mathcal{M}}(Q, I) = \bigcap \{Q(K) \mid \mathrm{Sol}_{\mathcal{M}}(K) \subseteq \mathrm{Sol}_{\mathcal{M}}(I)\}.$$

It can be easily proved that $\mathrm{Inf}_{\mathcal{M}}(Q, I)$ defines an upper bound on the amount of sound information that can be recovered for an instance I under a query Q. Before stating the lemma we introduce a notation that simplifies the exposition of the proof of this and subsequent results. Given a mapping $\mathcal{M}$ be a mapping and $I \in \mathrm{dom}(\mathcal{M})$, we define $\mathrm{Sub}_{\mathcal{M}}(I)$ as the set of instances $K \in \mathrm{dom}(\mathcal{M})$ such that $\mathrm{Sol}_{\mathcal{M}}(K) \subseteq \mathrm{Sol}_{\mathcal{M}}(I)$. Notice that, by using $\mathrm{Sub}_{\mathcal{M}}(I)$ the set $\mathrm{Inf}_{\mathcal{M}}(Q, I)$ can be defined as

$$\mathrm{Inf}_{\mathcal{M}}(Q, I) \;\; = \bigcap_{K \in \mathrm{Sub}_{\mathcal{M}}(I)} Q(K).$$

LEMMA 4.1.4. *Let $\mathcal{M}$ be a mapping and $\mathcal{M}'$ a $\mathcal{C}$-recovery of $\mathcal{M}$. Then for every $I \in \mathrm{dom}(\mathcal{M})$ and $Q \in \mathcal{C}$, it holds that:*

$$\mathrm{certain}_{\mathcal{M} \circ \mathcal{M}'}(Q, I) \;\subseteq\; \mathrm{Inf}_{\mathcal{M}}(Q, I) \subseteq\; Q(I).$$

PROOF. Let I and I' be instances in $\mathrm{dom}(\mathcal{M})$, and assume that $\mathrm{Sol}_{\mathcal{M}}(I') \subseteq \mathrm{Sol}_{\mathcal{M}}(I)$, that is $I' \in \mathrm{Sub}_{\mathcal{M}}(I)$. Then it holds that $\mathrm{Sol}_{\mathcal{M} \circ \mathcal{M}'}(I') \subseteq \mathrm{Sol}_{\mathcal{M} \circ \mathcal{M}'}(I)$ and, thus, we have that $\mathrm{certain}_{\mathcal{M} \circ \mathcal{M}'}(Q, I) \subseteq \mathrm{certain}_{\mathcal{M} \circ \mathcal{M}'}(Q, I')$ for every query $Q \in \mathcal{C}$. Given that $\mathcal{M}'$ is a $\mathcal{C}$-recovery of $\mathcal{M}$, it is the case that $\mathrm{certain}_{\mathcal{M} \circ \mathcal{M}'}(Q, I') \subseteq Q(I')$. Thus, for every $I' \in \mathrm{Sub}_{\mathcal{M}}(I)$, we have that $\mathrm{certain}_{\mathcal{M} \circ \mathcal{M}'}(Q, I) \subseteq \mathrm{certain}_{\mathcal{M} \circ \mathcal{M}'}(Q, I') \subseteq Q(I')$. Hence, by the definition of $\mathrm{Inf}_{\mathcal{M}}(Q, I)$ and given that $I \in \mathrm{Sub}_{\mathcal{M}}(I)$, we conclude that:

$$\mathrm{certain}_{\mathcal{M} \circ \mathcal{M}'}(Q, I) \;\subseteq\; \mathrm{Inf}_{\mathcal{M}}(Q, I) \subseteq\; Q(I).$$

□

Lemma 4.1.4 shows that the amount of sound information that a $\mathcal{C}$-recovery $\mathcal{M}'$ can retrieve for a source instance I by using a query Q, is bounded by the amount of information contained in the set $\mathrm{Inf}_{\mathcal{M}}(Q, I)$. Notice that this bound does not depend on the $\mathcal{C}$-recovery $\mathcal{M}'$. Moreover, given that $\mathrm{Inf}_{\mathcal{M}}(Q, I) \subseteq Q(I)$ and the set $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I)$ is bounded by $\mathrm{Inf}_{\mathcal{M}}(Q, I)$, we directly obtain that if $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I) = \mathrm{Inf}_{\mathcal{M}}(Q, I)$ for every query $Q \in \mathcal{C}$, then $\mathcal{M}'$ is a $\mathcal{C}$-maximum recovery of $\mathcal{M}$. This provides us with a sufficient condition for testing whether $\mathcal{M}'$ is a $\mathcal{C}$-maximum recovery of $\mathcal{M}$. In the following theorem we show that this property indeed characterizes when $\mathcal{M}'$ is a $\mathcal{C}$-maximum recovery of $\mathcal{M}$, that is, it is not only a sufficient but also a necessary condition. This is a strong justification of our initial claim that $\mathrm{Inf}_{\mathcal{M}}$ exactly captures the information that can be recovered for a mapping. To prove the result, we use an alternative characterization of when $\mathcal{M}'$ is a $\mathcal{C}$-maximum recovery of $\mathcal{M}$ (part (2) of the theorem) that is interesting in its own.

THEOREM 4.1.5. *Let $\mathcal{M}$ be a mapping from a schema $\mathbf{R}_1$ to a schema $\mathbf{R}_2$. The following are equivalent.*

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

(2) *$\mathcal{M}'$ is a $\mathcal{C}$-recovery of $\mathcal{M}$ and for every $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$ and every query $Q \in \mathcal{C}$ over $\mathbf{R}_1$, it holds that:*

$$\mathrm{Inf}_{\mathcal{M}}(Q, I_1) \quad \subseteq \quad Q(I_2)$$

(3) *For every instance $I \in \mathrm{dom}(\mathcal{M})$ and query $Q \in \mathcal{C}$, it holds that:*

$$\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I) \quad = \quad \mathrm{Inf}_{\mathcal{M}}(Q, I).$$

PROOF. We first prove the implication (1) $\Rightarrow$ (2). For the sake of contradiction, let $\mathcal{M}'$ be a $\mathcal{C}$-maximum recovery of $\mathcal{M}$ and suppose that there exists a pair $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$ and a query $Q \in \mathcal{C}$ such that:

$$\mathrm{Inf}_{\mathcal{M}}(Q, I_1) \quad \not\subseteq \quad Q(I_2).$$

Then there exist a tuple $\bar{t} \in \mathrm{Inf}_{\mathcal{M}}(Q, I_1)$ such that $\bar{t} \notin Q(I_2)$. This implies that $\bar{t} \notin \mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I_1)$ since $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$. Now, let $\mathcal{M}''$ be the following mapping from $\mathbf{R}_2$ to $\mathbf{R}_1$:

$$\mathcal{M}'' = \{(J, I) \mid J \in \mathrm{Sol}_{\mathcal{M}}(I_1) \text{ and } I \in \mathrm{Sub}_{\mathcal{M}}(I_1)\} \cup \{(J, I) \mid J \notin \mathrm{Sol}_{\mathcal{M}}(I_1) \text{ and } I \in \mathrm{Inst}(\mathbf{R}_1)\},$$

Next we show that $\mathcal{M}''$ is a $\mathcal{C}$-recovery of $\mathcal{M}$. Let I be an instance in $\mathrm{dom}(\mathcal{M})$. If $\mathrm{Sol}_{\mathcal{M}}(I) \subseteq \mathrm{Sol}_{\mathcal{M}}(I_1)$, then we have that $\mathrm{Sol}_{\mathcal{M}\circ\mathcal{M}''}(I) = \mathrm{Sub}_{\mathcal{M}}(I_1)$. Thus, we obtain that $(I, I) \in \mathcal{M} \circ \mathcal{M}''$ and, hence, $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q', I) \subseteq Q'(I)$ for every query $Q' \in \mathcal{C}$. On the other hand, if $\mathrm{Sol}_{\mathcal{M}}(I) \not\subseteq \mathrm{Sol}_{\mathcal{M}}(I_1)$, then it holds that $\mathrm{Sol}_{\mathcal{M}\circ\mathcal{M}''}(I) = \mathrm{Inst}(\mathbf{R}_1)$. Thus, we obtain again that $(I, I) \in \mathcal{M} \circ \mathcal{M}''$. We conclude that for every query $Q' \in \mathcal{C}$, it holds that $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q', I) \subseteq Q'(I)$, and thus $\mathcal{M}''$ is a $\mathcal{C}$-recovery of $\mathcal{M}$.

Now, for instance I_1, it holds that $\mathrm{Sol}_{\mathcal{M}\circ\mathcal{M}''}(I_1) = \mathrm{Sub}_{\mathcal{M}}(I_1)$. Thus, by the definition of $\mathrm{Inf}_{\mathcal{M}}(Q, I_1)$ we obtain that $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I_1) = \mathrm{Inf}_{\mathcal{M}}(Q, I_1)$. Recall that $\bar{t} \in \mathrm{Inf}_{\mathcal{M}}(Q, I_1)$, thus we have that $\bar{t} \in \mathrm{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I_1)$. Thus, given that $\bar{t} \notin \mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I_1)$, we have that:

$$\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I_1) \not\subseteq \mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I_1),$$

which contradicts our assumption that $\mathcal{M}'$ is a $\mathcal{C}$-maximum recovery of $\mathcal{M}$. This concludes the proof of (1) $\Rightarrow$ (2).

We prove now that (2) $\Rightarrow$ (3). Let I be an instance in $\mathrm{dom}(\mathcal{M})$. From (2) we know that for every $(I, I') \in \mathcal{M} \circ \mathcal{M}'$ and every query $Q \in \mathcal{C}$, it holds that $\mathrm{Inf}_{\mathcal{M}}(Q, I) \subseteq Q(I')$. Thus, we have that for every $I' \in \mathrm{Sol}_{\mathcal{M}\circ\mathcal{M}'}(I)$, it holds that $\mathrm{Inf}_{\mathcal{M}}(Q, I) \subseteq Q(I')$ and, therefore, $\mathrm{Inf}_{\mathcal{M}}(Q, I) \subseteq \mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I)$. Then since since $\mathcal{M}'$ is a $\mathcal{C}$-recovery of $\mathcal{M}$, by Lemma 4.1.4 we obtain that $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I) \subseteq \mathrm{Inf}_{\mathcal{M}}(Q, I)$. Hence, we conclude that $\mathrm{Inf}_{\mathcal{M}}(Q, I) = \mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I)$ for every $I \in \mathrm{dom}(\mathcal{M})$ and query $Q \in \mathcal{C}$.

Finally implication (3) $\Rightarrow$ (1) is straightforward (see the discussion in the paragraph before the theorem). $\square$

In Definition 3.1.7 in Section 3.1.1, we introduced the notion of *witness* which plays a crucial role for the notion of maximum recovery. We use $\mathrm{Inf}_{\mathcal{M}}$ to define the notion of $\mathcal{C}$-witness, which plays this role for $\mathcal{C}$-maximum recoveries. In particular, we show that the notion of $\mathcal{C}$-witness can be used to provide a necessary and sufficient condition for the existence of a $\mathcal{C}$-maximum recovery.

DEFINITION 4.1.6. *Let $\mathcal{M}$ be a mapping and $I_1 \in \mathrm{dom}(\mathcal{M})$. Then J is a* $\mathcal{C}$-witness for I_1 under $\mathcal{M}$ *if for every instance I_2 and query $Q \in \mathcal{C}$, if $J \in \mathrm{Sol}_{\mathcal{M}}(I_2)$, then:*

$$\mathrm{Inf}_{\mathcal{M}}(Q, I_2) \quad \subseteq \quad Q(I_1).$$

In the next lemma we show that the notion of $\mathcal{C}$-witness is a relaxation of the notion of witness of Definition 3.1.7.

LEMMA 4.1.7. *Let $\mathcal{M}$ be a mapping from a schema $\mathbf{R}_1$ to a schema $\mathbf{R}_2$ and $I \in \mathrm{Inst}(\mathbf{R}_1)$. If J is a witness for I under $\mathcal{M}$, then J is a $\mathcal{C}$-witness for I under $\mathcal{M}$.*

PROOF. Let $\mathcal{M}$ be a mapping from a schema $\mathbf{R}_1$ to a schema $\mathbf{R}_2$, $I_1 \in \mathrm{Inst}(\mathbf{R}_1)$ and $J \in \mathrm{Inst}(\mathbf{R}_2)$ a witness for I_1 under $\mathcal{M}$. Next we show that J is a $\mathcal{C}$-witness for I_1 under $\mathcal{M}$. Assume that there exists an instance $I_2 \in \mathrm{Inst}(\mathbf{R}_1)$ such that $J \in \mathrm{Sol}_{\mathcal{M}}(I_2)$. By the definition of witness, we have that $\mathrm{Sol}_{\mathcal{M}}(I_1) \subseteq \mathrm{Sol}_{\mathcal{M}}(I_2)$. This imply that $\mathrm{Sub}_{\mathcal{M}}(I_1) \subseteq \mathrm{Sub}_{\mathcal{M}}(I_2)$. Thus, for every query Q in $\mathcal{C}$, it holds that:

$$\mathrm{Inf}_{\mathcal{M}}(Q, I_2) \quad \subseteq \quad \mathrm{Inf}_{\mathcal{M}}(Q, I_1).$$

Since $I_1 \in \mathrm{Sub}_{\mathcal{M}}(I_1)$, we have that $\mathrm{Inf}_{\mathcal{M}}(Q, I_1) \subseteq Q(I_1)$ for every query Q in $\mathcal{C}$. Therefore, we have that for every query $Q \in \mathcal{C}$, it holds that:

$$\mathrm{Inf}_{\mathcal{M}}(Q, I_2) \quad \subseteq \quad Q(I_1).$$

We conclude that J is a $\mathcal{C}$-witness for I_1 under $\mathcal{M}$, which was to be shown. □

The notion of $\mathcal{C}$-witness can be used to characterize when a mapping $\mathcal{M}'$ is a $\mathcal{C}$-maximum recovery of a mapping $\mathcal{M}$. In fact, the following theorem shows that $\mathcal{C}$-witness instances are the building blocks of $\mathcal{C}$-maximum recoveries.

THEOREM 4.1.8. *$\mathcal{M}'$ is a $\mathcal{C}$-maximum recovery of $\mathcal{M}$ iff, $\mathcal{M}'$ is a $\mathcal{C}$-recovery of $\mathcal{M}$ and for every $(I_1, J) \in \mathcal{M}$ and $(J, I_2) \in \mathcal{M}'$, it holds that J is a $\mathcal{C}$-witness for I_2 under $\mathcal{M}$.*

PROOF. ($\Rightarrow$) Let $\mathcal{M}$ be a mapping from a schema $\mathbf{R}_1$ to a schema $\mathbf{R}_2$, and assume that $\mathcal{M}'$ is a $\mathcal{C}$-maximum recovery of $\mathcal{M}$. By hypothesis, $\mathcal{M}'$ is a $\mathcal{C}$-recovery of $\mathcal{M}$. So, it only remains to show that for every $(I_1, J) \in \mathcal{M}$ and $(J, I_2) \in \mathcal{M}'$, it holds that J is a $\mathcal{C}$-witness for I_2 under $\mathcal{M}$. For the sake of contradiction, assume that there exist tuples $(I_1, J) \in \mathcal{M}$ and $(J, I_2) \in \mathcal{M}'$ such that J is not a $\mathcal{C}$-witness for I_2 under $\mathcal{M}$. Then there exists an instance $I \in \mathrm{dom}(\mathcal{M})$ and a query $Q \in \mathcal{C}$ such that $J \in \mathrm{Sol}_{\mathcal{M}}(I)$ and

$$\mathrm{Inf}_{\mathcal{M}}(Q, I) \quad \not\subseteq \quad Q(I_2).$$

Given that $(I, J) \in \mathcal{M}$ and $(J, I_2) \in \mathcal{M}'$, it holds that $(I, I_2) \in \mathcal{M} \circ \mathcal{M}'$. Thus, we have a tuple $(I, I_2) \in \mathcal{M} \circ \mathcal{M}'$ and a query $Q \in \mathcal{C}$ such that $\mathrm{Inf}_{\mathcal{M}}(Q, I) \not\subseteq Q(I_2)$. By Theorem 4.1.5, this implies that $\mathcal{M}'$ is not a $\mathcal{C}$-maximum recovery of $\mathcal{M}$, which leads to a contradiction.

($\Leftarrow$) Now, assume that $\mathcal{M}'$ is a $\mathcal{C}$-recovery of $\mathcal{M}$ and that for every $(I_1, J) \in \mathcal{M}$ and $(J, I_2) \in \mathcal{M}'$, it holds that J is a $\mathcal{C}$-witness for I_2 under $\mathcal{M}$. Next we use Theorem 4.1.5 to show that $\mathcal{M}'$ is a $\mathcal{C}$-maximum recovery of $\mathcal{M}$.

Let $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$. Then there exists an instance J such that $(I_1, J) \in \mathcal{M}$ and $(J, I_2) \in \mathcal{M}'$. Thus, given that J is a $\mathcal{C}$-witness for I_2 under $\mathcal{M}$ and $J \in \mathrm{Sol}_{\mathcal{M}}(I_1)$, we have that $\mathrm{Inf}_{\mathcal{M}}(Q, I_1) \subseteq Q(I_2)$ for every query $Q \in \mathcal{C}$. Therefore, we conclude from Theorem 4.1.5 that $\mathcal{M}'$ is a $\mathcal{C}$-maximum recovery of $\mathcal{M}$. This concludes the proof of the theorem. □

Next we show that the notion of $\mathcal{C}$-witness can be used to provide a necessary and sufficient condition for the existence of $\mathcal{C}$-maximum recoveries. Given a mapping $\mathcal{M}$, the *$\mathcal{C}$-witness mapping* of $\mathcal{M}$, denoted by $\mathcal{M}^{\mathcal{C}}$, is defined as:

$$\mathcal{M}^{\mathcal{C}} \quad = \quad \{(J, I) \mid J \text{ is a } \mathcal{C}\text{-witness for } I \text{ under } \mathcal{M}\}.$$

Thus, $\mathcal{M}^{\mathcal{C}}$ is composed by all the $\mathcal{C}$-witness instances under a given mapping $\mathcal{M}$.

THEOREM 4.1.9. *A mapping $\mathcal{M}$ has a $\mathcal{C}$-maximum recovery iff the mapping $\mathcal{M}^{\mathcal{C}}$ is a $\mathcal{C}$-maximum recovery of $\mathcal{M}$.*

PROOF. The "if" part of the theorem follows directly from the hypothesis. Thus, we just have to show that if $\mathcal{M}$ has a $\mathcal{C}$-maximum recovery then the mapping $\mathcal{M}^{\mathcal{C}}$ is a $\mathcal{C}$-maximum recovery of $\mathcal{M}$.

Assume that there exists a mapping $\mathcal{M}'$ that is a $\mathcal{C}$-maximum recovery of $\mathcal{M}$. By Theorem 4.1.8, for every $(I_1, J) \in \mathcal{M}$ and $(J, I_2) \in \mathcal{M}'$, it holds that J is a $\mathcal{C}$-witness for I_2 under $\mathcal{M}$. Thus, by definition of $\mathcal{M}^{\mathcal{C}}$, we conclude that for every tuple $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$, it holds that $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}^{\mathcal{C}}$. Therefore, $\mathcal{M} \circ \mathcal{M}' \subseteq \mathcal{M} \circ \mathcal{M}^{\mathcal{C}}$, which implies that for every $I \in \mathrm{dom}(\mathcal{M})$ and query $Q \in \mathcal{C}$, it holds that $\mathrm{certain}_{\mathcal{M} \circ \mathcal{M}^{\mathcal{C}}}(Q, I) \subseteq \mathrm{certain}_{\mathcal{M} \circ \mathcal{M}'}(Q, I)$. Hence, given that $\mathcal{M}'$ is a $\mathcal{C}$-recovery of $\mathcal{M}$, we conclude that $\mathcal{M}^{\mathcal{C}}$ is a $\mathcal{C}$-recovery of $\mathcal{M}$. To show that the mapping $\mathcal{M}^{\mathcal{C}}$ is a $\mathcal{C}$-maximum recovery of $\mathcal{M}$, we use theorem 4.1.8. Let $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}^{\mathcal{C}}$. Then there exists an instance J such that $(I_1, J) \in \mathcal{M}$ and $(J, I_2) \in \mathcal{M}^{\mathcal{C}}$. By the definition of the mapping $\mathcal{M}^{\mathcal{C}}$, we have that J is a $\mathcal{C}$-witness for I_2. Given that $\mathcal{M}^{\mathcal{C}}$ is $\mathcal{C}$-recovery of $\mathcal{M}$, we conclude by Theorem 4.1.8 that $\mathcal{M}^{\mathcal{C}}$ is a $\mathcal{C}$-maximum recovery of $\mathcal{M}$. This was to be shown. □

4.1.2. On the choice of a query language

Up to this point, a natural question is what is the influence of the parameter $\mathcal{C}$ on the notion of $\mathcal{C}$-maximum recovery. In the following sections, we show that this parameter is essential to obtain a good mapping language for inversion. Thus, the goal of this section is to shed light on this issue. Let us start with an example.

Example 4.1.10. Let $\mathcal{M}$ be specified by these two st-tgds:

$$A(x,y) \rightarrow R(x,y), \qquad B(x) \rightarrow R(x,x).$$

It can be shown that mapping $\mathcal{M}_1$ specified by dependency:

$$R(x,y) \quad \rightarrow \quad A(x,y) \vee \big(B(x) \wedge x = y\big)$$

is a UCQ-maximum recovery of $\mathcal{M}$. To specify $\mathcal{M}_1$, we have used a disjunction in the conclusion of the dependency. This disjunction is unavoidable if we use UCQ to retrieve information. On the other hand, if we focus on CQ to retrieve information, then, intuitively, there is no need for disjunctions in the right-hand side of the rules as conjunctive queries cannot extract disjunctive information. In fact, it can be shown that a CQ-maximum recovery of $\mathcal{M}$ is specified by dependency:

$$R(x,y) \wedge x \neq y \quad \rightarrow \quad A(x,y).$$

□

The example suggests that the notion of CQ-maximum recovery is a strict generalization of the notion of UCQ-maximum recovery. The following proposition provides a complete picture of the relationship of the notions of $\mathcal{C}$-maximum recovery, when one focuses on mappings specified by st-tgds and the most common extensions of CQ.

PROPOSITION 4.1.11.

(1) *There exist mappings $\mathcal{M}$ and $\mathcal{M}'$ such that $\mathcal{M}'$ is a* UCQ*-maximum recovery of $\mathcal{M}$ but not a* $\text{CQ}^{\neq}$*-maximum recovery of $\mathcal{M}$.*

(2) *There exist mappings $\mathcal{M}$ and $\mathcal{M}'$ such that $\mathcal{M}'$ is a* $\text{CQ}^{\neq}$*-maximum recovery of $\mathcal{M}$ but not a* UCQ*-maximum recovery of $\mathcal{M}$.*

PROOF. To show part (1), let $\mathbf{S} = \{P(\cdot), R(\cdot)\}$, $\mathbf{T} = \{T(\cdot,\cdot), S(\cdot)\}$ and Σ, Σ' be the following sets of dependencies:

$$\begin{aligned}\Sigma &= \{P(x) \rightarrow \exists y\, T(x,y),\ R(x) \rightarrow S(x)\},\\ \Sigma' &= \{T(x,y) \rightarrow P(x) \wedge P(y),\ T(x,x) \wedge S(y) \rightarrow P(y),\ S(x) \rightarrow R(x)\}.\end{aligned}$$

First, we show that $\mathcal{M}' = (\mathbf{T}, \mathbf{S}, \Sigma')$ is not a CQ$^{\neq}$-recovery of $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, from which we conclude that $\mathcal{M}'$ is not a CQ$^{\neq}$-maximum recovery of $\mathcal{M}$. Consider instance I of $\mathbf{S}$ such that $P^I = \{a\}$ and $R^I = \{b\}$, with $a \neq b$, and Boolean query Q in CQ$^{\neq}$ defined as:

$$\exists x \exists y (P(x) \wedge P(y) \wedge x \neq y).$$

Clearly $Q(I) = \underline{\text{false}}$. Let J be an instance such that $(I, J) \in \mathcal{M} \circ \mathcal{M}'$. Then there exists an instance K of $\mathbf{T}$ such that $(I, K) \in \mathcal{M}$ and $(K, J) \in \mathcal{M}'$. Given that $(I, K) \models \Sigma$, we have that $b \in S^K$ and $(a, c) \in T^K$ for some value c. Given that $(K, J) \models \Sigma'$, we conclude by considering ts-tgd $T(x, y) \rightarrow P(x) \wedge P(y)$ that $a, c \in P^J$. Thus, if $a \neq c$, we have $Q(J) = \underline{\text{true}}$. Now, if $a = c$, then by considering ts-tgd $T(x, x) \wedge S(y) \rightarrow P(y)$ we conclude that $b \in P^J$. Therefore, $Q(J)$ holds since $a \in P^J$ and $a \neq b$. We have shown that for every $J \in \text{Sol}_{\mathcal{M}\circ\mathcal{M}'}(I)$, it is the case that $Q(J) = \underline{\text{true}}$. Hence, we have that $\text{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I) = \underline{\text{true}}$, which shows that $\mathcal{M}'$ is not a CQ$^{\neq}$-recovery of $\mathcal{M}$ since $Q(I) = \underline{\text{false}}$.

Now we show that $\mathcal{M}'$ is a UCQ-maximum recovery of $\mathcal{M}$. In fact, we show a stronger result, namely that for every instance I of $\mathbf{S}$ and query Q in UCQ over $\mathbf{S}$, it holds that $\text{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I) = Q(I)$. Let I be an instance of $\mathbf{S}$ and Q a query in UCQ over $\mathbf{S}$. Furthermore, let $J = \text{chase}_{\Sigma}(I)$ and $K = \text{chase}_{\Sigma'}(J)$. By the definitions of Σ and Σ', it is straightforward to prove that K is homomorphically equivalent to I. Thus, given that Q is a query in UCQ and homomorphisms are assumed to be the identity on the constants, we have that for every tuple $\bar{a}$ of constants, $\bar{a} \in Q(I)$ if and only if $\bar{a} \in Q(K)$. Therefore, we have that $Q(I) = Q(K)_{\downarrow}$, where $Q(K)_{\downarrow}$ is defined as the set of tuples of constants that belong to $Q(K)$ (Fagin, Kolaitis, Miller, & Popa, 2005). But from (Fagin, Kolaitis, Miller,

& Popa, 2005; Fagin, Kolaitis, Popa, & Tan, 2005), we know that $\text{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I) = Q(K)_{\downarrow}$ and, thus, $\text{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I) = Q(I)$.

To show part (2), let $\mathbf{S} = \{D(\cdot), E(\cdot), F(\cdot)\}$, $\mathbf{T} = \{P(\cdot), R(\cdot)\}$ and Σ, Σ' be the following sets of dependencies:

$$\begin{aligned}\Sigma &= \{D(x) \rightarrow P(x),\ E(x) \rightarrow P(x),\ F(x) \rightarrow R(x)\},\\ \Sigma' &= \{R(x) \rightarrow F(x)\}.\end{aligned}$$

First, we show that $\mathcal{M}' = (\mathbf{T}, \mathbf{S}, \Sigma')$ is not a UCQ-maximum recovery of $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$. For the sake of contradiction, assume that $\mathcal{M}'$ is a UCQ-maximum recovery of $\mathcal{M}$, and consider instance I of $\mathbf{S}$ such that $I = \{D(a)\}$, and Boolean query Q in UCQ defined as:

$$\exists x D(x) \vee \exists y E(y).$$

It is straightforward to prove that $(I, I_\emptyset) \in \mathcal{M} \circ \mathcal{M}'$, where $I_\emptyset$ is the empty instance. Thus, we have that $\text{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I) = \underline{\text{false}}$ since $Q(I_\emptyset) = \underline{\text{false}}$. Therefore, we have from Theorem 4.1.5 that $\text{Inf}_{\mathcal{M}}(Q, I) = \underline{\text{false}}$ since $\mathcal{M}'$ is a UCQ-maximum recovery of $\mathcal{M}$. Hence, there exists an instance I' of $\mathbf{S}$ such that $\text{Sol}_{\mathcal{M}}(I') \subseteq \text{Sol}_{\mathcal{M}}(I)$ and $Q(I') = \underline{\text{false}}$. We conclude that $D^{I'} = E^{I'} = \emptyset$, which implies that the instance J of $\mathbf{T}$ such that $P^J = \emptyset$ and $R^J = F^{I'}$ is a solution for I' under $\mathcal{M}$. But J is not a solution for I under $\mathcal{M}$ (since $a \notin P^J$), which contradicts the fact that $\text{Sol}_{\mathcal{M}}(I') \subseteq \text{Sol}_{\mathcal{M}}(I)$.

Second, we show that $\mathcal{M}'$ is a $\text{CQ}^{\neq}$-maximum recovery of $\mathcal{M}$. Notice that for every instance I of $\mathbf{S}$, it holds that $I' = \text{chase}_{\Sigma'}(\text{chase}_{\Sigma}(I))$ is a solution for I under $\mathcal{M} \circ \mathcal{M}'$ such that $I' \subseteq I$. Thus, given that every query in $\text{CQ}^{\neq}$ is monotone, we conclude that $\text{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I) \subseteq Q(I') \subseteq Q(I)$ for every query Q in $\text{CQ}^{\neq}$ over $\mathbf{S}$. Therefore, we have that $\mathcal{M}'$ is a $\text{CQ}^{\neq}$-recovery of $\mathcal{M}$. To conclude the proof, we have to show that $\mathcal{M}'' \preceq_{\mathcal{M}}^{\text{CQ}^{\neq}} \mathcal{M}'$ for every $\text{CQ}^{\neq}$-recovery $\mathcal{M}''$ of $\mathcal{M}$. Let $\mathcal{M}''$ be a $\text{CQ}^{\neq}$-recovery of $\mathcal{M}$, Q a query in $\text{CQ}^{\neq}$ over $\mathbf{S}$ and I an instance of $\mathbf{S}$. We consider two cases to show that $\text{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I) \subseteq \text{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I)$.

(I) Assume that one of the conjuncts of Q is of the form either $D(x)$ or $E(x)$. Next we show that $\text{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I) = \emptyset$, from which we conclude that $\text{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I) \subseteq \text{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I)$.

For the sake of contradiction, assume that $\text{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I) \neq \emptyset$. Without loss of generality, assume that $D(x)$ is one of the conjunct of Q, and let I' be an instance of $\mathbf{S}$ such that $D^{I'} = \emptyset$, $E^{I'} = (D^I \cup E^I)$ and $F^{I'} = F^I$. Then we have that $\text{Sol}_{\mathcal{M}}(I) = \text{Sol}_{\mathcal{M}}(I')$, which implies that $\text{Sol}_{\mathcal{M}\circ\mathcal{M}''}(I) = \text{Sol}_{\mathcal{M}\circ\mathcal{M}''}(I')$. Thus, we have that $\text{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I) = \text{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I')$. Given that $D(x)$ is one of the conjuncts of Q, Q is a query in $\text{CQ}^{\neq}$ and $D^{I'} = \emptyset$, we have that $Q(I') = \emptyset$. Thus, given that $\mathcal{M}''$ is a $\text{CQ}^{\neq}$-recovery of $\mathcal{M}$ and $I' \in \text{dom}(\mathcal{M})$, we have that $\text{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I') = \emptyset$, which leads to a contradiction (since $\text{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I) = \text{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I')$ and we assume that $\text{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I) \neq \emptyset$).

(II) Assume that all of the conjuncts of Q are of the form $R(x)$. Next we show that $Q(I) \subseteq \text{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I)$, from which we conclude that $\text{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I) \subseteq \text{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I)$ since $\mathcal{M}''$ is a $\text{CQ}^{\neq}$-recovery of $\mathcal{M}$.

It is straightforward to prove that for every $I' \in \text{Sol}_{\mathcal{M}\circ\mathcal{M}'}(I)$, it holds that $R^I \subseteq R^{I'}$. Thus, given that all of the conjuncts of Q are of the form $R(x)$ and Q is a query in $\text{CQ}^{\neq}$, we conclude that $Q(I) \subseteq Q(I')$ for every $I' \in \text{Sol}_{\mathcal{M}\circ\mathcal{M}'}(I)$. Therefore, we have that $Q(I) \subseteq \text{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I)$.

□

Proposition 4.1.11 tells that the notions of UCQ- and $\text{CQ}^{\neq}$-maximum recovery are incomparable, even in the case of st-tgds. From this proposition, we can also conclude the following. Assume that $\mathcal{C}_1$ and $\mathcal{C}_2$ are any of the query languages CQ, UCQ, $\text{CQ}^{\neq}$ or $\text{UCQ}^{\neq}$. Then from Proposition 4.1.11, one can conclude that if $\mathcal{C}_1 \subsetneq \mathcal{C}_2$, then there exist mappings $\mathcal{M}$ and $\mathcal{M}'$ specified by tgds such that, $\mathcal{M}'$ is a $\mathcal{C}_1$-maximum recovery of $\mathcal{M}$ but not a $\mathcal{C}_2$-maximum recovery of $\mathcal{M}$.

4.2. $\mathcal{C}$-Maximum Recovery and Previous Notions

In this section we compare the notion of $\mathcal{C}$-maximum recovery with previous notions proposed in the literature about inverses and the notion of $\mathcal{C}$-equivalence for schema mappings (Madhavan & Halevy, 2003; Fagin, Kolaitis, Nash, & Popa, 2008).

4.2.1. $\mathcal{C}$-Maximum Recovery and other notions of inverse

In this section, we study the notions of Fagin-inverse (Fagin, 2007) and quasi-inverse (Fagin, Kolaitis, Popa, & Tan, 2008), as well as the notion of maximum recovery by using the formal notion of retrieving sound information introduced in Section 4.1 (see Section 2.5 for the definitions of Fagin-inverses and quasi-inverses). In particular, we show that these notions appear as specific points in the range of $\mathcal{C}$-maximum recoveries, for different choices of the class $\mathcal{C}$. In this study, we also use an additional concept to highlight some properties of these notions of inverse. This concept measures the ability of an inverse operator to recover all the sound data. Formally, let $\mathcal{M}$ be a mapping from a schema $\mathbf{R}_1$ to a schema $\mathbf{R}_2$ and Q a query over $\mathbf{R}_1$. Then we say that $\mathcal{M}'$ *fully recovers* Q *for* $\mathcal{M}$ if for every instance $I \in \mathrm{dom}(\mathcal{M})$, it holds that

$$\mathrm{certain}_{\mathcal{M} \circ \mathcal{M}'}(Q, I) = Q(I).$$

Moreover, given a class $\mathcal{C}$ of queries, we say that $\mathcal{M}'$ fully recovers $\mathcal{C}$ for $\mathcal{M}$ if for every query $Q \in \mathcal{C}$ over $\mathbf{R}_1$, it holds that $\mathcal{M}'$ fully recovers Q for $\mathcal{M}$.

We start our study by considering the notion of Fagin-inverse (Fagin, 2007). Recall that a mapping $\mathcal{M}$ from a schema $\mathbf{R}_1$ to a schema $\mathbf{R}_2$ is closed-down on the left if whenever $(I, J) \in \mathcal{M}$ an $I' \subseteq I$, it holds that $(I', J) \in \mathcal{M}$, and is total if $\mathrm{dom}(\mathcal{M}) = \mathrm{Inst}(\mathbf{R}_1)$. As we explained in Section 3.1.1 the notion of Fagin-inverse is appropriate for closed-down on the left and total mappings. The following theorem establishes the relationship between Fagin-inverses and $\mathcal{C}$-maximum recoveries.

THEOREM 4.2.1. *Let $\mathcal{M}$ be a total and closed-down on the left mapping, that has a Fagin-inverse. Then the following statements are equivalent:*

(1) *$\mathcal{M}'$ is a Fagin-inverse of $\mathcal{M}$,*

(2) *$\mathcal{M}'$ is a* $\mathrm{UCQ}^{\neq}$*-maximum recovery of $\mathcal{M}$,*

(3) *$\mathcal{M}'$ fully recovers* $\mathrm{UCQ}^{\neq}$ *for $\mathcal{M}$.*

PROOF. We first show (1) $\Leftrightarrow$ (3). First, we show that if $\mathcal{M}'$ is a Fagin-inverse of $\mathcal{M}$, then for every query Q in $\mathrm{UCQ}^{\neq}$ over $\mathbf{S}$, it holds that $\mathcal{M}'$ fully recovers Q for $\mathcal{M}$.

Let Q be a query in $\mathrm{UCQ}^{\neq}$ over $\mathbf{S}$ and I an instance of $\mathbf{S}$. We have to show that $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I) = Q(I)$. Given that $\mathcal{M}'$ is a Fagin-inverse of $\mathcal{M}$, we have that $I \subseteq J$ for every $J \in \mathrm{Sol}_{\mathcal{M}\circ\mathcal{M}'}(I)$. Thus, given that Q is a monotone query, we have that $Q(I) \subseteq Q(J)$ for every $J \in \mathrm{Sol}_{\mathcal{M}\circ\mathcal{M}'}(I)$. It follows that $Q(I) \subseteq \mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I)$ and, thus, $Q(I) = \mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I)$ since $I \in \mathrm{Sol}_{\mathcal{M}\circ\mathcal{M}'}(I)$.

We show now that if for every query Q in $\mathrm{UCQ}^{\neq}$ over $\mathbf{S}$, it holds that $\mathcal{M}'$ fully recovers Q for $\mathcal{M}$, then $\mathcal{M}'$ is a Fagin-inverse of $\mathcal{M}$. That is, we show that $(I, J) \in \mathcal{M} \circ \mathcal{M}'$ if and only if $I \subseteq J$.

(I) Assume that $(I, J) \in \mathcal{M} \circ \mathcal{M}'$, and for every $R \in \mathbf{S}$, let Q_R be the identity query for table R, that is, $Q_R(\bar{x}) = R(\bar{x})$. Given that $\mathcal{M}'$ fully recovers each of these queries for $\mathcal{M}$, we conclude that for every $J \in \mathrm{Sol}_{\mathcal{M}\circ\mathcal{M}'}(I)$ and $R \in \mathbf{S}$, it holds that $Q_R(I) \subseteq Q_R(J)$, that is, $R^I \subseteq R^J$. Thus, we have that $I \subseteq J$.

(II) Assume that $I \subseteq J$. To prove that $(I, J) \in \mathcal{M} \circ \mathcal{M}'$, we first show that $(J, J) \in \mathcal{M} \circ \mathcal{M}'$.

For the sake of contradiction, assume that $(J, J) \notin \mathcal{M} \circ \mathcal{M}'$. Then for every relation $R \in \mathbf{S}$, define a Boolean query Q_R as follows. Assuming that the arity of R is k and R^J contains n tuples, Q_R is the following query in $\mathrm{UCQ}^{\neq}$.

$$\exists \bar{x}_1 \cdots \exists \bar{x}_n \exists \bar{x}_{n+1} \left(\left(\bigwedge_{1 \leq i \leq n+1} R(\bar{x}_i) \right) \wedge \left(\bigwedge_{1 \leq i < j \leq n+1} \bar{x}_i \neq \bar{x}_j \right) \right),$$

where $\bar{u} \neq \bar{v}$ stands for the formula $\bigvee_{\ell=1}^{k} u_i \neq v_i$, for k-tuples $\bar{u} = (u_1, \ldots, u_k)$ and $\bar{v} = (v_1, \ldots, v_k)$. Thus, Q_R says that relation R contains at least $n+1$ tuples.

Let Q be the following query in UCQ$^{\neq}$.

$$Q \;=\; \bigvee_{R \in \mathbf{S}} Q_R.$$

Then we have that $Q(J) = \underline{\text{false}}$.

By (I), we know that if $(J, J') \in \mathcal{M} \circ \mathcal{M}'$, then $J \subseteq J'$ (notice that the proof in (I) was done for an arbitrary pair of instances in $\mathcal{M} \circ \mathcal{M}'$). Thus, given that $(J, J) \notin \mathcal{M} \circ \mathcal{M}'$, we have that for every $J' \in \mathrm{Sol}_{\mathcal{M} \circ \mathcal{M}'}(J)$, there exists $R \in \mathbf{S}$ such that $R^J \subsetneq R^{J'}$. We conclude that $Q(J') = \underline{\text{true}}$ for every $J' \in \mathrm{Sol}_{\mathcal{M} \circ \mathcal{M}'}(J)$. But this contradicts the fact that $\mathcal{M}'$ fully recovers Q for $\mathcal{M}$ since $Q(J) = \underline{\text{false}}$ and $J \in \mathrm{dom}(\mathcal{M})$ (since $\mathcal{M}$ is a total mapping).

Given that $(J, J) \in \mathcal{M} \circ \mathcal{M}'$, we have that there exists an instance K of $\mathbf{T}$ such that $(J, K) \in \mathcal{M}$ and $(K, J) \in \mathcal{M}'$. Thus, given that $\mathcal{M}$ is closed-down on the left and $I \subseteq J$, we conclude that $(I, K) \in \mathcal{M}$. Hence, given that $(K, J) \in \mathcal{M}'$, we have that $(I, J) \in \mathcal{M} \circ \mathcal{M}'$.

This concludes the proof of (1) $\Leftrightarrow$ (3).

We now show (1) $\Leftrightarrow$ (2). First, notice that if $\mathcal{M}'$ is a Fagin-inverse of $\mathcal{M}$, then we have that $\mathcal{M}'$ fully recovers UCQ$^{\neq}$ for $\mathcal{M}$, which implies that $\mathcal{M}'$ is a UCQ$^{\neq}$-maximum recovery of $\mathcal{M}$. Second, assume that $\mathcal{M}'$ is a UCQ$^{\neq}$-maximum recovery of $\mathcal{M}$. Given that $\mathcal{M}$ is Fagin-invertible, there exists a Fagin-inverse $\mathcal{M}^\star$ of $\mathcal{M}$. Thus, we have that $\mathcal{M}^\star$ fully recovers UCQ$^{\neq}$ for $\mathcal{M}$. Therefore, given that $\mathcal{M}'$ is a UCQ$^{\neq}$-maximum recovery of $\mathcal{M}$, we have that $\mathcal{M}^\star \preceq_{\mathcal{M}}^{\mathrm{UCQ}^{\neq}} \mathcal{M}'$ and, hence, $\mathcal{M}'$ fully recovers UCQ$^{\neq}$ for $\mathcal{M}$. Again using the result above, we deduce that $\mathcal{M}'$ is a Fagin-inverse of $\mathcal{M}$, which concludes the proof of the theorem. □

Theorem 4.2.1 implies that a Fagin-inverse exists only if every query in UCQ$^{\neq}$ can be fully recovered, which rarely occurs. It also states that, when a Fagin-inverse exists, it coincides with the notion of UCQ$^{\neq}$-maximum recovery.

We continue our study by considering the notion of quasi-inverse (Fagin, Kolaitis, Popa, & Tan, 2008). (See Definition 2.5.2 for the formalization of the notion of quasi-inverse). In the following theorem, we characterize the notion of quasi-inverse for the case of st-tgds in terms of its ability to recover sound information.

THEOREM 4.2.2. *Let $\mathcal{M}$ be a mapping specified by a set of st-tgds, that has a quasi-inverse. Then there exists a subclass $\mathcal{C}$ of* $\text{UCQ}^{\neq}$ *such that the following statements are equivalent.*

(1) *$\mathcal{M}'$ is a quasi-inverse of $\mathcal{M}$,*

(2) *$\mathcal{M}'$ is a $\mathcal{C}$-maximum recovery of $\mathcal{M}$,*

(3) *$\mathcal{M}'$ fully recovers $\mathcal{C}$ for $\mathcal{M}$.*

Before proving the Theorem we recall some concepts regarding query rewriting and prove a technical lemma that we later use to construct the class of queries $\mathcal{C}$ in the statement of the theorem. Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ be an st-mapping where Σ is a set of FO-TO-CQ dependencies. Recall that, given a query Q over $\mathbf{T}$, a rewriting of Q over the source, is a query Q' over $\mathbf{S}$ such that $\text{certain}_{\mathcal{M}}(Q, I) = Q'(I)$ for every source instance I. As we show in Lemma 3.3.1, when Q is a conjunctive query such a rewriting Q' always exists and can be specified in FO over $\mathbf{S}$. (Moreover, from Lemma 3.3.3 we know that if Σ is a set of st-tgds, then the rewriting Q' is a query in $\text{UCQ}^{=}$.) In the proof we also use the following observation. Let I be a source instance and J the result of chasing I with Σ. It is known that $\text{certain}_{\mathcal{M}}(Q, I) = Q(J)_{\downarrow}$ for every Q that is a conjunctive query (Fagin, Kolaitis, Miller, & Popa, 2005; Arenas et al., 2004). These two concepts imply that, if Q' is a rewriting of a conjunctive query Q over the source and J is the result of chasing I with Σ, then $Q'(I) = Q(J)_{\downarrow}$.

In the following technical lemma we use query rewriting to introduce a set of queries $\mathcal{C}_{\mathcal{M}}$ that defines the space of solutions associated to a mapping $\mathcal{M}$. This set will be used to construct the set of queries in the statement of Theorem 4.2.2.

LEMMA 4.2.3. *Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ be an st-mapping where Σ is a set of* FO-TO-CQ *dependencies, and consider the following set of queries over $\mathbf{S}$:*

$$\mathcal{C}_{\mathcal{M}} = \{\chi(\bar{x}) \mid \varphi(\bar{x}) \rightarrow \psi(\bar{x}) \in \Sigma \text{ and } \chi(\bar{x}) \text{ is a source rewriting of } \psi(\bar{x}) \text{ w.r.t. } \mathcal{M}\}.$$

Then for every pair of instances I_1, I_2, it holds that $\mathrm{Sol}_{\mathcal{M}}(I_2) \subseteq \mathrm{Sol}_{\mathcal{M}}(I_1)$ if and only if, for every query $Q \in \mathcal{C}_{\mathcal{M}}$ we have that $Q(I_1) \subseteq Q(I_2)$.

PROOF. Assume first that $\mathrm{Sol}_{\mathcal{M}}(I_2) \subseteq \mathrm{Sol}_{\mathcal{M}}(I_1)$, and let $Q \in \mathcal{C}_{\mathcal{M}}$. We show now that $Q(I_1) \subseteq Q(I_2)$. Let σ in Σ be a formula of the form $\varphi(\bar{x}) \rightarrow \psi(\bar{x})$, such that Q is the rewriting of $\psi(\bar{x})$ over $\mathbf{S}$. Now, since $\mathrm{Sol}_{\mathcal{M}}(I_2) \subseteq \mathrm{Sol}_{\mathcal{M}}(I_1)$, we have that $\mathrm{certain}_{\mathcal{M}}(Q', I_1) \subseteq \mathrm{certain}_{\mathcal{M}}(Q', I_2)$ for every query Q'. In particular, this last property holds for the query defined by formula $\psi(\bar{x})$. Then since Q is a rewriting of $\psi(\bar{x})$ over $\mathbf{S}$, we conclude that $Q(I_1) \subseteq Q(I_2)$.

For the other direction, let I_1, I_2 be source instances and assume that for every $Q \in \mathcal{C}_{\mathcal{M}}$, it holds that $Q(I_1) \subseteq Q(I_2)$. We show now that $\mathrm{Sol}_{\mathcal{M}}(I_2) \subseteq \mathrm{Sol}_{\mathcal{M}}(I_1)$. Let $(I_2, J) \models \Sigma$, we must show that $(I_1, J) \models \Sigma$. Let $\sigma \in \Sigma$ be a dependency of the form $\varphi(\bar{x}) \rightarrow \psi(\bar{x})$, and assume that $I_1 \models \varphi(\bar{a})$ for some tuple $\bar{a}$ of constant values. We need to show that $J \models \psi(\bar{a})$. Since $I_1 \models \varphi(\bar{a})$ we know that for every $J' \in \mathrm{Sol}_{\mathcal{M}}(I_1)$ it holds that $J' \models \psi(\bar{a})$. Now, let Q_ψ be the conjunctive query defined by $\psi(\bar{x})$, and consider the query $Q' \in \mathcal{C}_{\mathcal{M}}$ obtained by rewriting $\psi(\bar{x})$. Since $Q'(I_1) = \mathrm{certain}_{\mathcal{M}}(Q_\psi, I_1)$ and since for every $J' \in \mathrm{Sol}_{\mathcal{M}}(I_1)$ we have that $\bar{a} \in Q_\psi(J')$, we obtain that $\bar{a} \in \mathrm{certain}_{\mathcal{M}}(Q_\psi, I_1)$ and then $\bar{a} \in Q'(I_1)$. Thus, since we are assuming that $Q'(I_1) \subseteq Q'(I_2)$, we conclude that $\bar{a} \in Q'(I_2)$. Recall that Q' is a rewriting of $\psi(\bar{x})$, and then since $\bar{a} \in Q'(I_2)$ we have that $\bar{a} \in \mathrm{certain}_{\mathcal{M}}(Q_\psi, I_2)$. That is, for every $K \in \mathrm{Sol}_{\mathcal{M}}(I_2)$ we have that $\bar{a} \in Q_\psi(K)$. In particular, for J we have that $\bar{a} \in Q_\psi(J)$, and then $J \models \psi(\bar{a})$. This was to be shown. □

We now have all the ingredients to prove Theorem 4.2.2.

PROOF OF THEOREM 4.2.2. We describe first how to construct the set of queries in the statement of the theorem. Given the mapping $\mathcal{M}$, let $\mathcal{C}_{\mathcal{M}}$ be the set of queries constructed in Lemma 4.2.3. Notice that since $\mathcal{M}$ is specified by st-tgds, we know that $\mathcal{C}_{\mathcal{M}}$ is a set of queries in $\mathrm{UCQ}^{=}$ over $\mathbf{S}$ (see Lemma 3.3.3). Consider now the set of queries $\mathcal{C}^{\star}_{\mathcal{M}}$ obtained from $\mathcal{C}_{\mathcal{M}}$ by closing the set under conjunction, disjunction, existential quantification, variable substitution, and addition of inequalities between free variables. Since $\mathcal{C}_{\mathcal{M}}$ is a set of queries in $\mathrm{UCQ}^{=}$, we have that $\mathcal{C}^{\star}_{\mathcal{M}}$ is a set of queries in $\mathrm{UCQ}^{=,\neq}$.

We first show that the set $\mathcal{C}^{\star}_{\mathcal{M}}$ satisfies the same property of Lemma 4.2.3. That is, for every pair of instances I_1, I_2, it holds that $\mathrm{Sol}_{\mathcal{M}}(I_2) \subseteq \mathrm{Sol}_{\mathcal{M}}(I_1)$ if and only if $Q(I_1) \subseteq Q(I_2)$ for every $Q \in \mathcal{C}^{\star}_{\mathcal{M}}$. Since $\mathcal{C}_{\mathcal{M}} \subseteq \mathcal{C}^{\star}_{\mathcal{M}}$, the "only if" part follows immediately. For the other direction we use an inductive argument. Assume that $\mathrm{Sol}_{\mathcal{M}}(I_2) \subseteq \mathrm{Sol}_{\mathcal{M}}(I_1)$ and let Q be a query in $\mathcal{C}^{\star}_{\mathcal{M}}$. There are several cases. Suppose that Q is the conjunction of two queries Q_1 and Q_2 in $\mathcal{C}^{\star}_{\mathcal{M}}$. Then $Q(I_1) = Q_1(I_1) \cap Q_2(I_1)$. By induction hypothesis we have that $Q_1(I_1) \subseteq Q_1(I_2)$ and $Q_2(I_1) \subseteq Q_2(I_2)$ and then $Q(I_1) = Q_1(I_1) \cap Q_2(I_1) \subseteq Q_1(I_2) \cap Q_2(I_2) = Q(I_2)$. If Q is the disjunction of two queries the argument is similar. Assume Q is obtained from $Q' \in \mathcal{C}^{\star}_{\mathcal{M}}$ by existentially quantifying some of the free variables of Q'. From $Q'(I_1) \subseteq Q'(I_2)$ it is straightforward to conclude that $Q(I_1) \subseteq Q(I_2)$. It is also straightforward to conclude that $Q(I_1) \subseteq Q(I_2)$ if Q has been obtained by substituting some variables in a query $Q' \in \mathcal{C}^{\star}_{\mathcal{M}}$. Finally, let Q' be a query in $\mathcal{C}^{\star}_{\mathcal{M}}$ and $(x_1, \ldots, x_k)$ the tuple of free variables of Q' (with $k \geq 2$), and assume that Q is obtained from Q' by adding the inequality $x_i \neq x_j$ with $i \neq j$. Notice that if a tuple $\bar{a} = (a_1, \ldots, a_k)$ is in $Q(I_1)$, then $a_i \neq a_j$ and $\bar{a} \in Q'(I_1)$. Since $Q'(I_1) \subseteq Q'(I_2)$ we obtain that, if $\bar{a} \in Q(I_1)$ then $a_i \neq a_j$ and $\bar{a} \in Q'(I_2)$, which implies that $\bar{a} \in Q(I_2)$.

We show now that for the set of queries $\mathcal{C}^{\star}_{\mathcal{M}}$, statements (1), (2), and (3) are equivalent. Thus, assume first that $\mathcal{M}$ is quasi-invertible. We show now that $\mathcal{M}'$ is a quasi-inverse of $\mathcal{M}$ if and only if $\mathcal{M}'$ fully recovers $\mathcal{C}^{\star}_{\mathcal{M}}$ which proves the equivalence (1) $\Leftrightarrow$ (3). To prove the "only if" part, let $\mathcal{M}'$ be a quasi-inverse of $\mathcal{M}$. We need to show that $\mathcal{M}'$ fully recovers $\mathcal{C}^{\star}_{\mathcal{M}}$. That is, we need to show that $\mathrm{certain}_{\mathcal{M} \circ \mathcal{M}'}(Q, I) = Q(I)$ for all $Q \in \mathcal{C}^{\star}_{\mathcal{M}}$. Let Q be a query in $\mathcal{C}^{\star}_{\mathcal{M}}$ and let I be a source instance. Recall that, if $\mathrm{Sol}_{\mathcal{M}}(I_2) \subseteq \mathrm{Sol}_{\mathcal{M}}(I_1)$,

then $Q(I_1) \subseteq Q(I_2)$. This fact implies that if $\mathrm{Sol}_{\mathcal{M}}(I_2) = \mathrm{Sol}_{\mathcal{M}}(I_1)$, then $Q(I_1) = Q(I_2)$. We use this last property several times in this part of the proof. Now, since $\mathcal{M}'$ is a quasi-inverse of $\mathcal{M}$, there exists instances I' and I'' such that $\mathrm{Sol}_{\mathcal{M}}(I) = \mathrm{Sol}_{\mathcal{M}}(I') = \mathrm{Sol}_{\mathcal{M}}(I'')$ and $(I', I'') \in \mathcal{M} \circ \mathcal{M}'$. Since $\mathrm{Sol}_{\mathcal{M}}(I) = \mathrm{Sol}_{\mathcal{M}}(I')$ and $(I', I'') \in \mathcal{M} \circ \mathcal{M}'$, we know that $(I, I'') \in \mathcal{M} \circ \mathcal{M}'$. Now, since $\mathrm{Sol}_{\mathcal{M}}(I) = \mathrm{Sol}_{\mathcal{M}}(I'')$, we know that $Q(I) = Q(I'')$, then we have that $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I) \subseteq Q(I'') = Q(I)$. Let $(I, I') \in \mathcal{M} \circ \mathcal{M}'$. There exists instances I_2, I_2' such that $\mathrm{Sol}_{\mathcal{M}}(I) = \mathrm{Sol}_{\mathcal{M}}(I_2)$, $\mathrm{Sol}_{\mathcal{M}}(I') = \mathrm{Sol}_{\mathcal{M}}(I_2')$ and $I_2 \subseteq I_2'$. Since Q is a monotone query, we obtain that $Q(I_2) \subseteq Q(I_2')$. Finally, since $\mathrm{Sol}_{\mathcal{M}}(I) = \mathrm{Sol}_{\mathcal{M}}(I_2)$, $\mathrm{Sol}_{\mathcal{M}}(I') = \mathrm{Sol}_{\mathcal{M}}(I_2')$, we obtain that $Q(I_2) = Q(I')$, $Q(I_2') = Q(I')$ and $Q(I) \subseteq Q(I')$. We have shown that, if $(I, I') \in \mathcal{M} \circ \mathcal{M}'$, then $Q(I) \subseteq Q(I')$, and then $Q(I) \subseteq \mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I)$. Thus it holds that for every $Q \in \mathcal{C}_{\mathcal{M}}$ and for every instance I, it holds that $Q(I) = \mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I)$. Then we conclude that $\mathcal{M}'$ fully recovers $\mathcal{C}_{\mathcal{M}}$ for $\mathcal{M}$.

To prove the opposite direction, assume that $\mathcal{M}'$ fully recovers $\mathcal{C}^{\star}_{\mathcal{M}}$. We need to show that $\mathcal{M}'$ is a quasi-inverse of $\mathcal{M}$. We show two properties:

(a) If $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$ then there exists instances I_1', I_2' such that $\mathrm{Sol}_{\mathcal{M}}(I_1) = \mathrm{Sol}_{\mathcal{M}}(I_1')$ and $\mathrm{Sol}_{\mathcal{M}}(I_2) = \mathrm{Sol}_{\mathcal{M}}(I_2')$, and $I_1' \subseteq I_2'$.

(b) For every I there exists an instance I', such that $\mathrm{Sol}_{\mathcal{M}}(I) = \mathrm{Sol}_{\mathcal{M}}(I')$ and $(I, I') \in \mathcal{M} \circ \mathcal{M}'$.

Properties (a) and (b) are enough to conclude that $\mathcal{M}'$ is a quasi-inverse of $\mathcal{M}$ (see Definition 2.5.2). To prove (a), let $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$. Since $\mathcal{M}'$ fully recovers $\mathcal{C}^{\star}_{\mathcal{M}}$, we know that for every $Q \in \mathcal{C}^{\star}_{\mathcal{M}}$ it holds that $Q(I_1) = \mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I_1)$. From $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$ we know that $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I_1) \subseteq Q(I_2)$. We have shown that, for every $Q \in \mathcal{C}^{\star}_{\mathcal{M}}$, it holds that $Q(I_1) \subseteq Q(I_2)$ which by the properties of $\mathcal{C}^{\star}_{\mathcal{M}}$ implies that $\mathrm{Sol}_{\mathcal{M}}(I_2) \subseteq \mathrm{Sol}_{\mathcal{M}}(I_1)$. Finally, since $\mathcal{M}$ is quasi-invertible, we know that $\mathcal{M}$ satisfies the $(\sim_{\mathcal{M}}, \sim_{\mathcal{M}})$-subset property (Fagin, Kolaitis, Popa, & Tan, 2008). Then from $\mathrm{Sol}_{\mathcal{M}}(I_2) \subseteq \mathrm{Sol}_{\mathcal{M}}(I_1)$ we conclude that there exists I_1' and I_2' such that $\mathrm{Sol}_{\mathcal{M}}(I_1) = \mathrm{Sol}_{\mathcal{M}}(I_1')$, $\mathrm{Sol}_{\mathcal{M}}(I_2) = \mathrm{Sol}_{\mathcal{M}}(I_2')$, and $I_1' \subseteq I_2'$. This completes the proof of (1).

Before proving (b), notice that, while proving (a) we have shown that if $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$ then $Q(I_1) \subseteq Q(I_2)$ for every $Q \in \mathcal{C}^{\star}_{\mathcal{M}}$. When proving (b) we use this last property.

Now we show that (b) holds. Let I be an arbitrary instance. For every query $Q \in \mathcal{C}_{\mathcal{M}}$ we construct a query Q^I as follows. If Q is a k-ary query and $m = |Q(I)|$, then we define the Boolean query

$$Q^I \ = \ \exists \bar{x}_1 \cdots \exists \bar{x}_{m+1} \Big(Q(\bar{x}_1) \wedge \cdots \wedge Q(\bar{x}_{m+1}) \wedge \bigwedge_{1 \leq i < j \leq m+1} \bar{x}_i \neq \bar{x}_j \Big)$$

where $\bar{x}_i$ a k-ary tuple. It is straightforward to see that $Q^I \in \mathcal{C}^{\star}_{\mathcal{M}}$. Notice that, if for an instance I' it holds that $Q^I(I') = \underline{\text{true}}$ then $|Q(I)| < |Q(I')|$. And conversely, if for an instance I' it holds that $Q^I(I') = \underline{\text{false}}$ then $|Q(I')| \leq |Q(I)|$. If Q is a Boolean query, we consider two cases. If $Q(I) = \underline{\text{false}}$ then $Q^I = Q$. If $Q(I) = \underline{\text{true}}$ we let $Q^I = \mathit{false}$ (that is, a query such that for every instance I', $Q(I') = \underline{\text{false}}$). Consider now the query

$$Q^{\star} \ = \ \bigvee_{Q \in \mathcal{C}_{\mathcal{M}}} Q^I.$$

Notice that $Q^{\star} \in \mathcal{C}^{\star}_{\mathcal{M}}$, and then $\mathcal{M}'$ fully recovers $Q^{\star}$. Also notice that $Q^{\star}(I) = \underline{\text{false}}$, and then $\text{certain}_{\mathcal{M} \circ \mathcal{M}'}(Q^{\star}, I) = \underline{\text{false}}$. This implies that there exists an instance $I^{\star}$ such that $(I, I^{\star}) \in \mathcal{M} \circ \mathcal{M}'$ and $Q^{\star}(I^{\star}) = \underline{\text{false}}$. That is, $Q^I(I^{\star}) = \underline{\text{false}}$ for every $Q \in \mathcal{C}_{\mathcal{M}}$. Now, let $Q \in \mathcal{C}_{\mathcal{M}}$ and assume that Q is k-ary. Since $Q^I(I^{\star}) = \underline{\text{false}}$ we obtain that $|Q(I^{\star})| \leq |Q(I)|$. Notice that from $(I, I^{\star}) \in \mathcal{M} \circ \mathcal{M}'$ and since $\mathcal{M}'$ fully recovers Q, we know that $Q(I) \subseteq Q(I^{\star})$. Then by using $|Q(I^{\star})| \leq |Q(I)|$ we conclude that $Q(I) = Q(I^{\star})$. Now, let $Q \in \mathcal{C}_{\mathcal{M}}$ and assume that Q is a Boolean query. If $Q(I) = \underline{\text{true}}$, then from $Q(I) \subseteq Q(I^{\star})$ we conclude that $Q(I^{\star}) = \underline{\text{true}}$. If $Q(I) = \underline{\text{false}}$ then $Q^{\star}(I^{\star}) = \underline{\text{false}}$ and since $Q^I = Q$ is a disjunction in $Q^{\star}$, we obtain that $Q(I^{\star}) = \underline{\text{false}}$. In any case we obtain that $Q(I) = Q(I^{\star})$. Thus we have shown that, for every $Q \in \mathcal{C}_{\mathcal{M}}$ it holds that $Q(I) = Q(I^{\star})$ which implies that $\text{Sol}_{\mathcal{M}}(I) = \text{Sol}_{\mathcal{M}}(I^{\star})$. Finally, we have that there exists an instance $I^{\star}$ such that $(I, I^{\star}) \in \mathcal{M} \circ \mathcal{M}'$, which is exactly what was to be shown.

We have shown that $\mathcal{M}'$ is a quasi-inverse of $\mathcal{M}$ if and only if $\mathcal{M}'$ fully recovers $\mathcal{C}^{\star}_{\mathcal{M}}$ for $\mathcal{M}$. To conclude the proof, we show that $\mathcal{M}'$ is a quasi-inverse of $\mathcal{M}$ if and only if $\mathcal{M}'$ is a $\mathcal{C}^{\star}_{\mathcal{M}}$-maximum recovery of $\mathcal{M}$ (that is, we show equivalence (1) $\Leftrightarrow$ (2). First, notice that if $\mathcal{M}'$ is a quasi-inverse of $\mathcal{M}$, then we have that $\mathcal{M}'$ fully recovers $\mathcal{C}^{\star}_{\mathcal{M}}$ for $\mathcal{M}$, which implies that $\mathcal{M}'$ is a $\mathcal{C}^{\star}_{\mathcal{M}}$-maximum recovery of $\mathcal{M}$. Second, assume that $\mathcal{M}'$ is a $\mathcal{C}^{\star}_{\mathcal{M}}$-maximum recovery of $\mathcal{M}$. Given that $\mathcal{M}$ is quasi-invertible, there exists a quasi-inverse $\mathcal{M}^{\star}$ of $\mathcal{M}$. Thus, we have that $\mathcal{M}^{\star}$ fully recovers $\mathcal{C}^{\star}_{\mathcal{M}}$ for $\mathcal{M}$. Therefore, given that $\mathcal{M}'$ is a $\mathcal{C}^{\star}_{\mathcal{M}}$-maximum recovery of $\mathcal{M}$, we have that $\mathcal{M}^{\star} \preceq^{\mathcal{C}^{\star}_{\mathcal{M}}}_{\mathcal{M}} \mathcal{M}'$ and, hence, $\mathcal{M}'$ fully recovers $\mathcal{C}^{\star}_{\mathcal{M}}$ for $\mathcal{M}$. Again using the result above, we deduce that $\mathcal{M}'$ is a quasi-inverse of $\mathcal{M}$, which concludes the proof of the theorem. □

We consider now the notion of maximum recovery introduced in Section 3.1. The following theorem presents our main result for this section regarding the notion of maximum recovery. Part (1) of Theorem 4.2.4 shows that for every class of queries $\mathcal{C}$, if a mapping $\mathcal{M}'$ is a maximum recovery of $\mathcal{M}$, then $\mathcal{M}'$ is also a $\mathcal{C}$-maximum recovery of $\mathcal{M}$. Thus, a maximum recovery is the best possible alternative to retrieve sound information.

THEOREM 4.2.4. *Let $\mathcal{M}$ be a mapping from a schema $\mathbf{R}_1$ to a schema $\mathbf{R}_2$, $\mathcal{M}'$ a maximum recovery of $\mathcal{M}$ and Q an arbitrary query over $\mathbf{R}_1$.*

(1) *If $\mathcal{M}''$ recovers sound information for $\mathcal{M}$ under Q, then $\mathcal{M}'' \preceq^{Q}_{\mathcal{M}} \mathcal{M}'$.*

(2) *If some mapping fully recovers Q for $\mathcal{M}$, then $\mathcal{M}'$ fully recovers Q for $\mathcal{M}$.*

PROOF. To prove part (1), assume that a mapping $\mathcal{M}''$ recovers sounds information for $\mathcal{M}$ under Q. Next we show that $\mathcal{M}'' \preceq^{Q}_{\mathcal{M}} \mathcal{M}'$. Let $I \in \mathrm{dom}(\mathcal{M})$. We have to show that $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I) \subseteq \mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I)$. Given that $\mathcal{M}'$ is a maximum recovery of $\mathcal{M}$, we have that $\mathrm{Sol}_{\mathcal{M}\circ\mathcal{M}'}(I) \neq \emptyset$ since $(I, I) \in \mathcal{M} \circ \mathcal{M}'$. Let I' be an arbitrary element of $\mathrm{Sol}_{\mathcal{M}\circ\mathcal{M}'}(I)$. Then given that $\mathcal{M}'$ is a maximum recovery of $\mathcal{M}$, we have that $I' \in \mathrm{dom}(\mathcal{M})$ and $\mathrm{Sol}_{\mathcal{M}}(I') \subseteq \mathrm{Sol}_{\mathcal{M}}(I)$ (see Proposition 3.1.6). Thus, we have that $\mathrm{Sol}_{\mathcal{M}\circ\mathcal{M}''}(I') \subseteq \mathrm{Sol}_{\mathcal{M}\circ\mathcal{M}''}(I)$, which implies that $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I) \subseteq$

$\text{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I')$. Therefore, given that $I' \in \text{dom}(\mathcal{M})$ and $\mathcal{M}''$ recovers sound information for $\mathcal{M}$ under Q, we have that $\text{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I) \subseteq \text{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I') \subseteq Q(I')$. Given that I' is an arbitrary instance, we conclude that for every instance $J \in \text{Sol}_{\mathcal{M}\circ\mathcal{M}'}(I)$, it is the case that $\text{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I) \subseteq Q(J)$, which implies that $\text{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I) \subseteq \text{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I)$ since $\text{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I) = \bigcap_{J\in\text{Sol}_{\mathcal{M}\circ\mathcal{M}'}(I)} Q(J)$.

We prove now part (2). Assume that there exists a mapping $\mathcal{M}''$ that fully recovers Q for $\mathcal{M}$. Notice that this implies that $\mathcal{M}''$ recovers sound information for $\mathcal{M}$ under Q. Next we show that these two facts imply that $\mathcal{M}'$ fully recovers Q for $\mathcal{M}$. Let $I \in \text{dom}(\mathcal{M})$. We need to show that $Q(I) = \text{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I)$. Given that $\mathcal{M}'$ is a maximum recovery, it holds that $\text{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I) \subseteq Q(I)$ and, hence, we only need to show that $Q(I) \subseteq \text{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I)$. But given that $\mathcal{M}''$ recovers sound information for $\mathcal{M}$ under Q, we have by (a) that $\mathcal{M}'' \preceq^{Q}_{\mathcal{M}} \mathcal{M}'$. Therefore, we have that $\text{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I) \subseteq \text{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I)$. Thus, we conclude that $Q(I) \subseteq \text{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I)$ since $Q(I) = \text{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I)$ (given that $\mathcal{M}''$ fully recovers Q for $\mathcal{M}$). This concludes the proof of the theorem. □

A natural question at this point is whether there exists a characterization of the notion of maximum recovery similar to the ones given for the notions of Fagin-inverse and quasi-inverse. In particular, since a maximum recovery is an ALL-maximum recovery, where ALL is the class of all queries, one may wonder whether the notions of maximum recovery and ALL-maximum recovery coincide. Somewhat surprisingly, the following result shows that this does not hold, even for the case of st-tgds.

PROPOSITION 4.2.5. *There exist mappings $\mathcal{M}$ and $\mathcal{M}'$ such that $\mathcal{M}$ is specified by a set of st-tgds (and thus $\mathcal{M}$ has a maximum recovery), $\mathcal{M}'$ is an* ALL*-maximum recovery of $\mathcal{M}$ and $\mathcal{M}'$ is not a maximum recovery of $\mathcal{M}$.*

PROOF. Let $\mathbf{S} = \{S(\cdot)\}$, $\mathbf{T} = \{T(\cdot)\}$ and $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where $\Sigma = \{\exists x S(x) \rightarrow \exists y T(y)\}$. Furthermore, assume that a is an arbitrary element of $\mathbf{C}$, and let $\mathcal{M}^\star$ be a mapping from $\mathbf{T}$ to $\mathbf{S}$ defined as follows:

$$\mathcal{M}^\star \;=\; \{(J_\emptyset, I_\emptyset)\} \cup \{(J, I) \mid J \in \mathrm{Inst}(\mathbf{T}),\ I \in \mathrm{Inst}(\mathbf{S}),\ J \neq J_\emptyset,\ I \neq I_\emptyset \text{ and } a \notin \mathrm{dom}(I)\},$$

where $I_\emptyset$ and $J_\emptyset$ are the empty instances of schemas $\mathbf{S}$ and $\mathbf{T}$, respectively.

Given that $\mathcal{M}$ is specified by a set of st-tgds, we know that $\mathcal{M}$ has a maximum recovery. In fact, by using the tools in Section 3.1.1 it is easy to show that the ts-mapping $\mathcal{M}'$ specified by dependency $\exists y T(y) \rightarrow \exists x S(x)$ is a maximum recovery of $\mathcal{M}$. By the definition of $\mathcal{M}^\star$, we know that $\mathcal{M}^\star$ is not a maximum recovery of $\mathcal{M}$. In fact, $\mathcal{M}^\star$ is not even a recovery of $\mathcal{M}$ since for the instance $I = \{S(a)\}$, we have that $(I, I) \notin \mathcal{M} \circ \mathcal{M}^\star$. Thus, to conclude the proof, we only need to show that $\mathcal{M}^\star$ is an ALL-maximum recovery of $\mathcal{M}$.

Assume that $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}^\star$. Then there exists an instance J of $\mathbf{T}$ such that $(I_1, J) \in \mathcal{M}$ and $(J, I_2) \in \mathcal{M}^\star$. If $J = J_\emptyset$, then we trivially have that $(J, I_2) \models \exists y T(y) \rightarrow \exists x S(x)$, from which we conclude that $(J, I_2) \in \mathcal{M}'$ and $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$. If $J \neq J_\emptyset$, then we have that $I_2 \neq I_\emptyset$ by definition of $\mathcal{M}^\star$. Thus, again we have that $(J, I_2) \models \exists y T(y) \rightarrow \exists x S(x)$, from which we conclude that $(J, I_2) \in \mathcal{M}'$ and $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$. Hence, we have that for every $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}^\star$, it holds that $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$, and we conclude that $\mathcal{M} \circ \mathcal{M}^\star \subseteq \mathcal{M} \circ \mathcal{M}'$.

Given that $\mathcal{M}'$ is a maximum recovery of $\mathcal{M}$ and $\mathcal{M} \circ \mathcal{M}^\star \subseteq \mathcal{M} \circ \mathcal{M}'$, to prove that $\mathcal{M}^\star$ is an ALL-maximum recovery of $\mathcal{M}$, we only need to show that $\mathcal{M}^\star$ is an ALL-recovery of $\mathcal{M}$. Let Q be a query over $\mathbf{S}$ and $I \in \mathrm{dom}(\mathcal{M})$. If the arity of Q is $k \geq 1$, then it is easy to see that $\mathrm{certain}_{\mathcal{M} \circ \mathcal{M}^\star}(Q, I) = \emptyset \subseteq Q(I)$. If Q is a Boolean query, then we need to show that $\mathrm{certain}_{\mathcal{M} \circ \mathcal{M}^\star}(Q, I) = \underline{\text{false}}$ whenever $Q(I) = \underline{\text{false}}$. Thus, assume that $Q(I) = \underline{\text{false}}$. By definition of $\mathcal{M}^\star$, there exists an instance I' isomorphic to I such that $(I, I') \in \mathcal{M} \circ \mathcal{M}^\star$. Therefore, given that Q is closed under isomorphisms, we conclude that

$\text{certain}_{\mathcal{M}\circ\mathcal{M}^\star}(Q, I) = \underline{\text{false}}$ since $Q(I') = \underline{\text{false}}$ and $I' \in \text{Sol}_{\mathcal{M}\circ\mathcal{M}^\star}(I)$. This concludes the proof of the proposition. □

4.2.2. $\mathcal{C}$-maximum recoveries and $\mathcal{C}$-equivalence

The idea of parameterizing by a class of queries a problem related to the management of mapping languages is not new. In fact, this idea was developed by Madhavan and Halevy (2003) to study the composition operator, and was also used by Fagin, Kolaitis, Nash, and Popa (2008) to develop a theory of schema-mapping optimization, where the authors introduced the notion of *certain-answers equivalence* of mappings (Madhavan & Halevy, 2003; Fagin, Kolaitis, Nash, & Popa, 2008). Let $\mathcal{C}$ be a class of queries. Then two mappings $\mathcal{M}$ and $\mathcal{M}'$ from $\mathbf{R}_1$ to $\mathbf{R}_2$ are $\mathcal{C}$-equivalent, denoted by $\mathcal{M} \equiv_{\mathcal{C}} \mathcal{M}'$, if for every query $Q \in \mathcal{C}$ over $\mathbf{R}_2$ and every instance I in $\mathbf{R}_1$, it holds that: (1) $I \in \text{dom}(\mathcal{M})$ if and only if $I \in \text{dom}(\mathcal{M}')$, and (2) $\text{certain}_{\mathcal{M}}(Q, I) = \text{certain}_{\mathcal{M}'}(Q, I)$, if $I \in \text{dom}(\mathcal{M}) \cap \text{dom}(\mathcal{M}')$.

If $\mathcal{M}_1$ and $\mathcal{M}_2$ are $\mathcal{C}$-equivalent, then we know that they behave in the same way with respect to $\mathcal{C}$. Thus, if one is going to retrieve information by using only queries from $\mathcal{C}$, a mapping $\mathcal{M}$ can be replaced by any other $\mathcal{C}$-equivalent mapping. In particular, it could be replaced by a mapping that can be handled more efficiently, thus optimizing the initial schema mapping. In the notion of $\mathcal{C}$-maximum recovery, this idea of only considering a particular query language to retrieve information is also present. The following result shows that the notions of maximum recovery and $\mathcal{C}$-maximum recovery can be related through the notion of $\mathcal{C}$-equivalence (Fagin, Kolaitis, Nash, & Popa, 2008).

PROPOSITION 4.2.6. *Let $\mathcal{M}'$ be a maximum recovery of $\mathcal{M}$, and $\mathcal{C}$ a class of queries.*

(1) *$\mathcal{M}''$ is a $\mathcal{C}$-maximum recovery of $\mathcal{M}$ iff $(\mathcal{M} \circ \mathcal{M}'') \equiv_{\mathcal{C}} (\mathcal{M} \circ \mathcal{M}')$.*

(2) *If $\mathcal{M}''$ is such that $\mathcal{M}'' \equiv_{\mathcal{C}} \mathcal{M}'$, then $\mathcal{M}''$ is a $\mathcal{C}$-maximum recovery of $\mathcal{M}$.*

PROOF. Notice first that, since $\mathcal{M}'$ is a maximum recovery of $\mathcal{M}$, from part (1) of Theorem 4.2.4 we know that $\mathcal{M}'$ is a $\mathcal{C}$-maximum recovery of $\mathcal{M}$.

We show now part (1). Assume that $(\mathcal{M} \circ \mathcal{M}'') \equiv_{\mathcal{C}} (\mathcal{M} \circ \mathcal{M}')$. It is easy to see that $\text{dom}(\mathcal{M}) = \text{dom}(\mathcal{M} \circ \mathcal{M}')$ (since $\mathcal{M}'$ is a recovery of $\mathcal{M}$) and then from the definition of

$\mathcal{C}$-equivalence we obtain that $\mathrm{dom}(\mathcal{M}) = \mathrm{dom}(\mathcal{M} \circ \mathcal{M}'')$. Thus, we have that for every $I \in \mathrm{dom}(\mathcal{M})$ and $Q \in \mathcal{C}$, it holds that $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I) = \mathrm{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I)$. This last property together with the fact that $\mathcal{M}'$ is a $\mathcal{C}$-maximum recovery of $\mathcal{M}$ implies that $\mathcal{M}''$ is a $\mathcal{C}$-maximum recovery of $\mathcal{M}$.

To prove the opposite direction, assume that $\mathcal{M}''$ is a $\mathcal{C}$-maximum recovery of $\mathcal{M}$. Since $\mathcal{M}'$ is also a $\mathcal{C}$-maximum recovery of $\mathcal{M}$, we obtain that for every $I \in \mathrm{dom}(\mathcal{M})$ and $Q \in \mathcal{C}$, it holds that $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I) = \mathrm{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I) \subseteq Q(I)$. From this last property, we obtain that if $I \in \mathrm{dom}(\mathcal{M})$ then $I \in \mathrm{dom}(\mathcal{M} \circ \mathcal{M}'')$, otherwise it could not be the case that $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I) \subseteq Q(I)$, and then we have that $\mathrm{dom}(\mathcal{M}) \subseteq \mathrm{dom}(\mathcal{M} \circ \mathcal{M}'')$. It is obvious that $\mathrm{dom}(\mathcal{M} \circ \mathcal{M}'') \subseteq \mathrm{dom}(\mathcal{M})$, and then we obtain that $\mathrm{dom}(\mathcal{M}) = \mathrm{dom}(\mathcal{M} \circ \mathcal{M}'')$. Since $\mathcal{M}'$ is a recovery of $\mathcal{M}$, we have that $\mathrm{dom}(\mathcal{M}) = \mathrm{dom}(\mathcal{M} \circ \mathcal{M}')$ and then we obtain that $\mathrm{dom}(\mathcal{M} \circ \mathcal{M}') = \mathrm{dom}(\mathcal{M} \circ \mathcal{M}'')$. Summing up, we have shown that $\mathrm{dom}(\mathcal{M} \circ \mathcal{M}') = \mathrm{dom}(\mathcal{M} \circ \mathcal{M}'')$, and for every $I \in \mathrm{dom}(\mathcal{M} \circ \mathcal{M}')$ and $Q \in \mathcal{C}$, it holds that $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I) = \mathrm{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I)$. Hence, $(\mathcal{M} \circ \mathcal{M}'') \equiv_{\mathcal{C}} (\mathcal{M} \circ \mathcal{M}')$.

We prove now part (2). Assume that $\mathcal{M}'' \equiv_{\mathcal{C}} \mathcal{M}'$. Next we show that $(\mathcal{M} \circ \mathcal{M}'') \equiv_{\mathcal{C}} (\mathcal{M} \circ \mathcal{M}')$, from which we conclude from (a) that $\mathcal{M}''$ is a $\mathcal{C}$-maximum recovery of $\mathcal{M}$. As in the previous proof, using the fact that $\mathcal{M}'$ is a recovery of $\mathcal{M}$ and $\mathcal{M}'' \equiv_{\mathcal{C}} \mathcal{M}'$, we can conclude that $\mathrm{dom}(\mathcal{M}) = \mathrm{dom}(\mathcal{M} \circ \mathcal{M}') = \mathrm{dom}(\mathcal{M} \circ \mathcal{M}'')$. Then it only remains to show that $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I) = \mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I)$ for every $I \in \mathrm{dom}(\mathcal{M})$ and $Q \in \mathcal{C}$. Let $I \in \mathrm{dom}(\mathcal{M})$ and $Q \in \mathcal{C}$. We first show that $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I) \subseteq \mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I)$. Let $\bar{a} \in \mathrm{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I)$ and consider an instance K such that $(I, K) \in \mathcal{M} \circ \mathcal{M}'$. Then there exists J such that $(I, J) \in \mathcal{M}$ and $(J, K) \in \mathcal{M}'$. Since $\mathcal{M}' \equiv_{\mathcal{C}} \mathcal{M}''$, it holds that $\mathrm{dom}(\mathcal{M}') = \mathrm{dom}(\mathcal{M}'')$ and then $J \in \mathrm{dom}(\mathcal{M}'')$. From the fact that $\bar{a} \in \mathrm{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I)$ and $(I, J) \in \mathcal{M}$, we obtain that for all L such that $(J, L) \in \mathcal{M}''$, it must be the case that $\bar{a} \in Q(L)$. This last property implies that $\bar{a} \in \mathrm{certain}_{\mathcal{M}''}(Q, J)$ and then, given that $\mathrm{certain}_{\mathcal{M}'}(Q, J) = \mathrm{certain}_{\mathcal{M}''}(Q, J)$, we have that $\bar{a} \in \mathrm{certain}_{\mathcal{M}'}(Q, J)$. Finally, given that $(J, K) \in \mathcal{M}'$, we have that $\bar{a} \in Q(K)$. We have proved that if $\bar{a} \in \mathrm{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I)$, then for every K such that $(I, K) \in \mathcal{M} \circ \mathcal{M}'$, it holds that $\bar{a} \in Q(K)$, from which we conclude that $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I) \subseteq \mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I)$. To prove that for

every $I \in \mathrm{dom}(\mathcal{M})$ and $Q \in \mathcal{C}$ it holds that $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I) \subseteq \mathrm{certain}_{\mathcal{M}\circ\mathcal{M}''}(Q, I)$, a symmetric argument can be used. □

4.3. A Schema Mapping Language Closed Under Inversion

One of our main goals in this chapter is to find a mapping-specification language that is closed under inversion. This goal amounts to (1) first choose a particular semantics for the inverse operator, and then (2) prove that under this semantics, there exists a mapping language $\mathcal{L}$ such that every schema mapping specified in $\mathcal{L}$ has an inverse also specified in $\mathcal{L}$. Thus, we have to deal with two parameters: the semantics for inverting mappings, and the language used for specifying mappings. As a desiderata, we would like to have a *natural* and *useful* semantics, and a mapping-specification language expressive enough to contain the class of st-tgds. It is important to notice that the notions of Fagin-inverse and quasi-inverse could not meet our requirements, as there exist mappings specified by st-tgds that admit neither Fagin-inverses nor quasi-inverses. The notion of $\mathcal{C}$-maximum recovery, developed in the previous sections, provides a range of natural and useful semantics for the inverse operator that will allow us to reach our goal.

The main result of this section is that, when we consider the notion of CQ-maximum recovery as our semantics for inversion of st-mappings, there exists a language that is closed under inversion and contains the class of st-tgds. More specifically, we we prove that every st-mapping specified by a set of $\mathrm{CQ}^{\mathrm{C},\neq}$-TO-CQ dependencies has a CQ-maximum recovery also specified by a set of $\mathrm{CQ}^{\mathrm{C},\neq}$-TO-CQ dependencies (Theorem 4.3.1). Although this language has appeared before in the literature about inverses of schema mappings (Fagin, Kolaitis, Popa, & Tan, 2008), it has not been used to study closure properties as the ones considered in this paper. It should be noticed that, with $\mathrm{CQ}^{\mathrm{C},\neq}$-TO-CQ dependencies, the standard *chase procedure* gives a single instance as output. Thus, every instance I has a solution that can be considered as a representative of the space of solutions for I, which is a desirable property for data exchange (Fagin, Kolaitis, Miller, & Popa, 2005). These results provide strong evidence that the language of $\mathrm{CQ}^{\mathrm{C},\neq}$-TO-CQ has good properties for inverting mappings.

Our closure result depends on both the mapping language and the class $\mathcal{C}$ used in the notion of $\mathcal{C}$-maximum recovery. Thus, a natural question is whether this result could be strengthened by considering other alternatives for these parameters. In Sections 4.3.2 and 4.3.3, we prove several negative results in this respect. These results show that, our choice of CQ-maximum recovery as the semantics for inversion and $\text{CQ}^{\mathbf{C},\neq}$-TO-CQ as the mapping language is, in a technical sense, optimal for obtaining a mapping language closed under inversion.

It should be noticed that our closure result (Theorem 4.3.1) is specific to the case of st-mappings, that is, mappings that consider only constant values in source instances. As we have seen, in this scenario inverses are ts-mappings which are mappings that transform instances with constant and null values into source instances that only contain constant values. This has been a common assumption on the literature on inverting mappings (Fagin, 2007; Fagin, Kolaitis, Popa, & Tan, 2008; Arenas, Pérez, & Riveros, 2009). Nevertheless, Fagin et al. (2009) have raised the issue of the asymmetry in the study of the inverse operator, and have proposed to study the inverse operator in a symmetric scenario in which both source and target schemas have constant and null values. We leave the study of closure properties in this symmetric scenario for future work.

4.3.1. $\text{CQ}^{\mathbf{C},\neq}$-TO-CQ is closed under inversion

The following is the main result of this section.

THEOREM 4.3.1. *Every st-mapping specified by a set of* $\text{CQ}^{\mathbf{C},\neq}$-TO-CQ *dependencies, has a* CQ*-maximum recovery specified by a set of* $\text{CQ}^{\mathbf{C},\neq}$-TO-CQ *dependencies.*

To prove the theorem we describe in this section an algorithm to compute a CQ-maximum recovery of an st-mapping given by $\text{CQ}^{\mathbf{C},\neq}$-TO-CQ dependencies, that gives as output a set of $\text{CQ}^{\mathbf{C},\neq}$-TO-CQ dependencies (see Theorem 4.3.7). For the rest of this section fix Σ as a finite set of $\text{CQ}^{\mathbf{C},\neq}$-TO-CQ dependencies, and $\mathcal{M}$ as the st-mapping specified by Σ.

We start with a simple observation. Consider the set $\bar{\Sigma}$ obtained from Σ by dropping all the atoms of the form $\mathbf{C}(x)$ that appear in the premises of the dependencies in Σ. Notice that, since $\mathcal{M}$ is a *source-to-target* mapping, every instance in the domain of $\mathcal{M}$ is composed only by elements in $\mathbf{C}$. This implies that the st-mapping specified by $\bar{\Sigma}$ is exactly $\mathcal{M}$. Thus, trough the rest of this section we work with $\bar{\Sigma}$ instead of Σ. We split the presentation of our main procedure in several sub-procedures.

Compute a maximum recovery for $\bar{\Sigma}$

We start by computing a set Σ' that specifies a maximum recovery of $\mathcal{M}$. We make use of the following lemma that is similar to Lemma 3.3.3 which refers to the output of procedure QUERYREWRITING of Lemma 3.3.1. The proof of the lemma follows from inspecting the proof of Lemma 3.3.1.

LEMMA 4.3.2. *Let $\bar{\mathcal{M}} = (\mathbf{S}, \mathbf{T}, \bar{\Sigma})$ be an st-mapping such that $\bar{\Sigma}$ is a set of* CQ$^{\neq}$-TO-CQ *dependencies, and let Q be an n-ary conjunctive query over schema $\mathbf{T}$. Then algorithm* QUERYREWRITING$(\bar{\mathcal{M}}, Q)$ *in Lemma 3.3.1 has as output a query Q' in* UCQ$^{\neq,=}$ *that is a rewriting of Q over the source. Moreover, the output query Q' is a formula of the form $\beta_1(\bar{x}) \vee \cdots \vee \beta_k(\bar{x})$ where $\bar{x}$ is an n-tuple of distinct variables, and for $1 \leq i \leq k$, the formula $\beta_i(\bar{x})$ is a* CQ$^{\neq,=}$ *query such that,*

- *$\bar{x}$ is exactly the tuple of free variables in $\beta_i(\bar{x})$,*
- *if the inequality $z \neq z'$ occurs in $\beta_i(\bar{x})$ then z and z' occur in some relational atom of $\beta_i(\bar{x})$, and*
- *if the equality $z = z'$ occurs in $\beta_i(\bar{x})$ then z or z' (but not necessarily both) occur in some relational atom of $\beta_i(\bar{x})$.*

Thus, let $\bar{\mathcal{M}}$ be the st-mapping specified by the set $\bar{\Sigma}$. Let $\mathcal{M}'$ be the mapping obtained as the output of MAXIMUMRECOVERY$(\bar{\mathcal{M}})$, and assume that Σ' is the set of dependencies that specify $\mathcal{M}'$. From Lemma 4.3.2 we have that Σ' is a set of CQ$^{\mathbf{C}}$-TO-UCQ$^{\neq,=}$ dependencies. Furthermore, by following algorithm MAXIMUMRECOVERY we can see that

every dependency in Σ' is of the form $\exists \bar{y}\psi(\bar{x}, \bar{y}) \wedge \mathbf{C}(\bar{x}) \rightarrow \beta_1(\bar{x}) \vee \cdots \vee \beta_k(\bar{x})$, with $k \geq 1$ and where

- $\exists \bar{y}\psi(\bar{x}, \bar{y})$ is the conclusion of some of the dependencies in $\bar{\Sigma}$,
- $\bar{x}$ is exactly the tuple of free variables of $\exists \bar{y}\psi(\bar{x}, \bar{y})$ and of $\beta_i(\bar{x})$ for $1 \leq i \leq k$,
- $\mathbf{C}(\bar{x})$ is a conjunction of formulas $\mathbf{C}(x)$ for every x in $\bar{x}$, and
- for $1 \leq i \leq k$, the formula $\beta_i(\bar{x})$ is a $\text{CQ}^{\neq,=}$ query such that,
  - if the inequality $z \neq z'$ occurs in $\beta_i(\bar{x})$ then z and z' occur in some relational atom of $\beta_i(\bar{x})$,
  - if the equality $z = z'$ occurs in $\beta_i(\bar{x})$ then z or z' (but not necessarily both) occur in some relational atom of $\beta_i(\bar{x})$.

Moreover, we can assume, without loss of generality, that for every $1 \leq i \leq k$, the equalities occurring in the formula $\beta_i(\bar{x})$ are only among free variables (equalities among existentially quantified variables, or among free variables and existentially quantified variables, can be eliminated by replacing the corresponding variables).

In the rest of the section we show how we can apply some operations to transform Σ' into a set $\Sigma^\star$ of $\text{CQ}^{\mathbf{C},\neq}$-TO-CQ dependencies such that, the ts-mapping $\mathcal{M}^\star$ specified by $\Sigma^\star$ is CQ-equivalent to $\mathcal{M}'$. Then since $\mathcal{M}'$ is a maximum recovery of $\mathcal{M}$, and $\mathcal{M}^\star \equiv_{\text{CQ}} \mathcal{M}'$, we conclude from Proposition 4.2.6 that $\mathcal{M}^\star$ is a CQ-maximum recovery of $\mathcal{M}$. The first step in our quest is to eliminate equalities and inequalities form the conclusions of Σ'.

Eliminate equalities and inequalities among free variables from the conclusions of Σ'

In this step we construct a set Σ'' that defines a maximum recovery of $\mathcal{M}$, such that the dependencies in Σ'' do not have equalities nor inequalities among free variables in their conclusions. We use a notion similar to what is called *complete description* in (Fagin, Kolaitis, Popa, & Tan, 2008). Let $\bar{x} = (x_1, \ldots, x_n)$ be a tuple of distinct variables. Consider a partition π of the set $\{x_1, \ldots, x_n\}$, and let $[x_i]_\pi$ be the equivalence class induced by π to which x_i belongs ($1 \leq i \leq n$). Let $f_\pi : \{x_1, \ldots, x_n\} \rightarrow \{x_1, \ldots, x_n\}$ be a function such that $f_\pi(x_i) = x_j$ if j is the minimum over all the index of the variables in $[x_i]_\pi$. That is, f_π is a function that selects a unique representative from every equivalence class induced by

π. For example, if $\bar{x} = (x_1, x_2, x_3, x_4, x_5)$ and π is the partition $\{\{x_1, x_4\}, \{x_2, x_5\}, \{x_3\}\}$, then $f_\pi(x_1) = x_1$, $f_\pi(x_2) = x_2$, $f_\pi(x_3) = x_3$, $f_\pi(x_4) = x_1$, $f_\pi(x_5) = x_2$. We also consider the formula δ_π that is constructed by taking the conjunction of the inequalities $f_\pi(x_i) \neq f_\pi(x_j)$ whenever $f_\pi(x_i)$ and $f_\pi(x_j)$ are different variables. In the above example we have that δ_π is the formula $x_1 \neq x_2 \wedge x_1 \neq x_3 \wedge x_2 \neq x_3$. Finally, given a conjunction of equalities and inequalities α and a conjunction of inequalities β, we say α is *consistent* with β if there is an assignment of values to the variables in α and β that satisfies all the equalities and inequalities in these formulas. For example, $x_1 = x_2$ is consistent with $x_1 \neq x_3$, while $x_1 = x_2 \wedge x_2 = x_3$ is not consistent with $x_1 \neq x_3$.

Recall that Σ' is a set of dependencies that specify mapping $\mathcal{M}'$ which is a maximum recovery of $\mathcal{M}$. We have the necessary ingredients to describe the procedure to construct a set Σ'' from Σ', such that dependencies in Σ'' has only inequalities among free variables in their conclusions. We call this procedure ELIMINATEEQINEQ.

Procedure ELIMINATEEQINEQ(Σ')

(1) Let Σ'' be empty.

(2) For every dependency σ in Σ' of the form $\exists \bar{y}\psi(\bar{x}, \bar{y}) \wedge \mathbf{C}(\bar{x}) \rightarrow \alpha(\bar{x})$ with $\bar{x} = (x_1, \ldots, x_n)$ a tuple of distinct variables, and for every partition π of $\{x_1, \ldots, x_n\}$ do the following:

- Let $\alpha(\bar{x}) = \beta_1(\bar{x}) \vee \cdots \vee \beta_k(\bar{x})$.
- Construct a formula γ from $\alpha(f_\pi(\bar{x}))$ as follows. For every $i \in \{1, \ldots, k\}$:
  - If the equalities and inequalities among free variables in $\beta_i(f_\pi(\bar{x}))$ are consistent with δ_π, then drop the equalities and inequalities among free variables in $\beta_i(f_\pi(\bar{x}))$ and add the resulting formula as a disjunct in γ.
- If γ has at least one disjunct then add to Σ'' the dependency σ_π given by $\exists \bar{y}\psi(f_\pi(\bar{x}), \bar{y}) \wedge \mathbf{C}(f_\pi(\bar{x})) \wedge \delta_\pi \rightarrow \gamma$.

(3) Return Σ'' □

For example, assume that $\sigma \in \Sigma'$ is the following dependency:

$$\exists y_1 A(x_1, x_2, y_1) \wedge B(x_3) \wedge \mathbf{C}(x_1) \wedge \mathbf{C}(x_2) \wedge \mathbf{C}(x_3) \rightarrow$$
$$[\, \exists u_1(P(x_1, x_2, x_3, u_1) \wedge u_1 \neq x_1) \wedge x_1 \neq x_2 \,] \vee [\, R(x_1, x_3) \wedge x_1 = x_2 \wedge x_1 \neq x_3 \,].$$

Consider the partition $\pi_1 = \{\{x_1, x_2\}, \{x_3\}\}$. Since $f_{\pi_1}(x_1) = f_{\pi_1}(x_2) = x_1$, the inequality $x_1 \neq x_2$ become $x_1 \neq x_1$ after applying f_{π_1}, and then the first disjunction in the conclusion of σ become unsatisfiable. Considering the second disjunction, the formula $x_1 = x_2 \wedge x_1 \neq x_3$ become $x_1 = x_1 \wedge x_1 \neq x_3$ after applying f_{π_1}, and since δ_{π_1} is formula $x_1 \neq x_3$, we obtain a satisfiable formula. Then we have that σ_{π_1} is the dependency

$$\exists y_1 A(x_1, x_1, y_1) \wedge B(x_3) \wedge \mathbf{C}(x_1) \wedge \mathbf{C}(x_3) \wedge x_1 \neq x_3 \rightarrow R(x_1, x_3),$$

and then σ_{π_1} is added to Σ'' in procedure ELIMINATEEQINEQ. If we consider the partition $\pi_2 = \{\{x_1\}, \{x_2, x_3\}\}$, the second disjunction in the conclusion of σ become unsatisfiable (since the inequality $x_1 \neq x_2$ is forced in δ_{π_2}). In this case we have that σ_{π_2} is the dependency

$$\exists y_1 A(x_1, x_2, y_1) \wedge B(x_2) \wedge \mathbf{C}(x_1) \wedge \mathbf{C}(x_2) \wedge x_1 \neq x_2 \rightarrow \exists u_1(P(x_1, x_2, x_3, u_1) \wedge u_1 \neq x_1),$$

and σ_{π_2} is added to Σ''. On the other hand, if we consider partition $\pi_3 = \{\{x_1, x_2, x_3\}\}$, that is, considering a single equivalence class for the whole set of variables, then σ_{π_3} is not added to Σ'' since both disjunctions in the conclusion of σ become unsatisfiable.

Notice that the set Σ'' obtained after the described process, is a set of $\mathrm{CQ}^{\mathbf{C},\neq}$-TO-$\mathrm{UCQ}^{\neq}$ dependencies. Moreover, Σ'' is such that for every disjunction $\beta(\bar{x})$ in the conclusion of a dependency, and for every inequality $x \neq x'$ occurring in $\beta(\bar{x})$, we have that x or x' are existentially quantified variables (that is, it is not the case that both x and x' belong to $\bar{x}$).

The following lemma shows the key property that the set Σ'' satisfies.

LEMMA 4.3.3. *Let $\mathcal{M}''$ be the ts-mapping specified by the set Σ'' constructed in procedure* ELIMINATEEQINEQ(Σ'). *Then $\mathcal{M}''$ is a maximum recovery of $\mathcal{M}$.*

PROOF. In what follows, we make use of the following technical result proved in (Fagin, Kolaitis, Popa, & Tan, 2008) (Proposition 6.7). (We have also included some of the details in Section 2.4.) Assume that Γ_1 is a set of FO-TO-CQ dependencies and Γ_2 a set of $\text{CQ}^{\mathbf{C},\neq}$-TO-UCQ dependencies. Let $\mathcal{M}_1$ be the st-mapping specified by Γ_1 and $\mathcal{M}_2$ the ts-mapping specified by Γ_2. In (Fagin, Kolaitis, Popa, & Tan, 2008) the authors proved that, if I is a source instance, J the result of chasing I with Γ_1, and $\mathcal{V} = \{K_1, K_2, \ldots, K_\ell\}$ the result of chasing J with Γ_2, then for every I' such that $(I, I') \in \mathcal{M}_1 \circ \mathcal{M}_2$ there exists a homomorphism from some $K \in \mathcal{V}$ to I'. By following the proof in (Fagin, Kolaitis, Popa, & Tan, 2008), one can see that the mentioned result also holds when Γ_2 is a set of $\text{CQ}^{\mathbf{C},\neq}$-TO-$\text{UCQ}^{\neq}$ dependencies, provided that the inequalities in the conclusions of the dependencies of Γ_2 always mention an existentially quantified variable (inequalities that mention existentially quantified variables do not affect the normal chase procedure). Thus, we can apply this result to $\bar{\Sigma}$ and Σ''.

We continue now with the proof of the lemma. Recall that $\mathcal{M}'$ the mapping specified by Σ', is a maximum recovery of $\mathcal{M}$. We show now that $\mathcal{M}''$ is also a maximum recovery of $\mathcal{M}$. First, it is straightforward to see that, if $(J, I) \models \Sigma'$ then $(J, I) \models \Sigma''$, from which we obtain that $\mathcal{M} \circ \mathcal{M}' \subseteq \mathcal{M} \circ \mathcal{M}''$. Then it only remains to prove that $\mathcal{M} \circ \mathcal{M}'' \subseteq \mathcal{M} \circ \mathcal{M}'$.

Before proving that $\mathcal{M} \circ \mathcal{M}'' \subseteq \mathcal{M} \circ \mathcal{M}'$, we make the following observation about Σ''. Notice that for every dependency σ in Σ'', and for every variable x that simultaneously occurs in the premise and the conclusion of σ, we have that $\mathbf{C}(x)$ occurs in the premise of σ. This property is enough to conclude that, if $(J, K) \models \Sigma''$ with J and K arbitrary instances composed by constants and null values, and there exists a homomorphism from K to K', then $(J, K') \models \Sigma''$. That is, Σ'' is closed under *target homomorphism* (see (Fagin, Kolaitis, Nash, & Popa, 2008)).

In order to prove that $\mathcal{M} \circ \mathcal{M}'' \subseteq \mathcal{M} \circ \mathcal{M}'$, let $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}''$. Assume that J is the result of chasing I_1 with $\bar{\Sigma}$, and $\mathcal{V}$ the result of chasing J with Σ''. From the discussion above, we know that there exists an instance $K \in \mathcal{V}$ and a homomorphism from K to I_2. We also know that $(J, K) \models \Sigma''$. Then given that Σ'' is closed under *target*

homomorphism, we have that $(J, I_2) \models \Sigma''$, and then $(J, I_2) \in \mathcal{M}''$ since I_2 is a valid source instance. We show now that $(J, I_2) \models \Sigma'$ and then $(J, I_2) \in \mathcal{M}'$. Let σ be a dependency of Σ' of the form $\exists \bar{y}\psi(\bar{x}, \bar{y}) \wedge \mathbf{C}(\bar{x}) \rightarrow \alpha(\bar{x})$ with $\bar{x} = (x_1, \ldots, x_n)$ a tuple of distinct variables. Assume that there exists an n-tuple $\bar{a} = (a_1, \ldots, a_n)$ of elements from $\mathrm{dom}(J)$ such that $J \models \exists \bar{y}\psi(\bar{a}, \bar{y}) \wedge \mathbf{C}(\bar{a})$. We have to show that $I_2 \models \alpha(\bar{a})$. Recall that Σ' is the output of the algorithm MAXIMUMRECOVERY of (Arenas et al., 2008) applied to $\bar{\Sigma}$. It was shown in (Arenas et al., 2008) that, if J is the result of chasing I_1 with $\bar{\Sigma}$, then $(J, I_1) \models \Sigma'$. Then since $J \models \exists \bar{y}\psi(\bar{a}, \bar{y}) \wedge \mathbf{C}(\bar{a})$, we have that $I_1 \models \alpha(\bar{a})$. Now, consider a partition $\pi_{\bar{a}}$ of $\{x_1, \ldots, x_n\}$ constructed by considering the equivalence classes $[x_i]_{\pi_{\bar{a}}} = \{x_j \mid a_j = a_i\}$ for $1 \leq i \leq n$. Notice that, by the construction of the partition $\pi_{\bar{a}}$, if in the tuple $f_{\pi_{\bar{a}}}(\bar{x}) = (f_{\pi_{\bar{a}}}(x_1), \ldots, f_{\pi_{\bar{a}}}(x_n))$ we assign to every variable x_j its corresponding value a_j for $1 \leq j \leq n$, we obtain exactly the tuple $\bar{a}$. Also notice that this same assignment satisfies the formula $\delta_{\pi_{\bar{a}}}$. Then since $I_1 \models \alpha(\bar{a}) = \beta_1(\bar{a}) \vee \cdots \vee \beta_k(\bar{a})$, we have that there exists an index i with $1 \leq i \leq k$ such that $\beta_i(f_{\pi_{\bar{a}}}(\bar{x})) \wedge \delta_{\pi_{\bar{a}}}$ is satisfiable by using the assignment $x_j \rightarrow a_j$ for $1 \leq j \leq n$. Then we know that dependency $\sigma_{\pi_{\bar{a}}}$ of the form $\exists \bar{y}\psi(f_{\pi_{\bar{a}}}(\bar{x}), \bar{y}) \wedge \mathbf{C}(f_{\pi_{\bar{a}}}(\bar{x})) \wedge \delta_{\pi_{\bar{a}}} \rightarrow \alpha'(f_{\pi_{\bar{a}}}(\bar{x}))$ is added to the set Σ''. Finally, since $(J, I_2) \models \Sigma''$ we know that $(J, I_2) \models \sigma_{\pi_{\bar{a}}}$. Then given that J satisfies the formula $\exists \bar{y}\psi(\bar{a}, \bar{y}) \wedge \mathbf{C}(\bar{a}) \wedge \delta_{\pi_{\bar{a}}}$ we know that $I_2 \models \alpha'(\bar{a})$, and from this is straightforward to see that $I_2 \models \alpha(\bar{a})$. This was to be shown. Then since $(I_1, J) \in \mathcal{M}$ and $(J, I_2) \in \mathcal{M}'$ we obtain that $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$.

We have shown that $\mathcal{M} \circ \mathcal{M}' \subseteq \mathcal{M} \circ \mathcal{M}''$ and that $\mathcal{M} \circ \mathcal{M}'' \subseteq \mathcal{M} \circ \mathcal{M}'$, which implies that $\mathcal{M} \circ \mathcal{M}' = \mathcal{M} \circ \mathcal{M}''$. Then since $\mathcal{M}'$ is a maximum recovery of $\mathcal{M}$ we obtain that $\mathcal{M}''$ is also a maximum recovery of $\mathcal{M}$. That is, we have that Σ'' specifies a maximum recovery of $\mathcal{M}$. In the next step of our procedure, we eliminate the remaining inequalities in the conclusions. □

We continue now with the procedure to compute a CQ-maximum recovery of $\mathcal{M}$. In the rest of this section we consider Σ'' as the set constructed from Σ' as described above, and $\mathcal{M}''$ as the ts-mapping specified by Σ''.

Eliminate the remaining inequalities in the conclusions of Σ''

In this step we just drop all the remaining inequalities in the disjunctions of the conclusions of the dependencies of Σ''. It turns out that, although the obtained set of dependencies may no longer define a maximum recovery of $\mathcal{M}$, it does define a CQ-maximum recovery of $\mathcal{M}$. In fact a stronger result holds, namely, that the obtained set of dependencies defines a UCQ-maximum recovery of $\mathcal{M}$. We formalize this fact in the following lemma.

LEMMA 4.3.4. *Let Σ''' be the set constructed from Σ'' by dropping all the inequalities in the conclusions of dependencies in Σ'', and let $\mathcal{M}'''$ be the ts-mapping specified by Σ'''. Then (1) $\mathcal{M}''' \equiv_{\text{UCQ}} \mathcal{M}''$ and (2) $\mathcal{M}'''$ is a* UCQ*-maximum recovery of $\mathcal{M}$.*

PROOF. Recall that all the inequalities in the disjunctions of the conclusions of the dependencies in Σ'' are of the form $x \neq x'$ where x or x' is an existentially quantified variable. Also notice that, when chasing an instance with Σ'' we select a fresh null value for every existentially quantified variable. These facts are enough to conclude that the result of chasing with Σ'', is the same as the result of chasing with Σ''' (up to isomorphic images of null values). By using this last property we can show that $\mathcal{M}'''$ is a UCQ-maximum recovery of $\mathcal{M}$.

We show first property (1), that is, we show that $\mathcal{M}''' \equiv_{\text{UCQ}} \mathcal{M}''$. Notice that from this last fact and since $\mathcal{M}''$ is a maximum recovery of $\mathcal{M}$, we obtain from Proposition 4.2.6 that $\mathcal{M}'''$ is a UCQ-maximum recovery of $\mathcal{M}$, thus showing property (2). First notice that the domain of $\mathcal{M}''$ as well as the domain of $\mathcal{M}'''$, is the set of all target instances (all the instances composed by constants and null values). Then in order to prove that $\mathcal{M}''' \equiv_{\text{UCQ}} \mathcal{M}''$, we need to show that for every target instance J and every query Q that is a union of conjunctive queries, it holds that $\text{certain}_{\mathcal{M}''}(Q, J) = \text{certain}_{\mathcal{M}'''}(Q, J)$.

Let J be a target instance, and $\mathcal{V} = \{K_1, \ldots, K_\ell\}$ the result of chasing J with Σ''. Notice that every dependency σ in Σ'' is such that, for every variable x that occurs simultaneously in the premise and the conclusion of σ, the atom $\mathbf{C}(x)$ also occurs in the premise of σ. From this last fact and the properties of $\mathcal{V}$, it follows directly from the results in (Fagin,

Kolaitis, Miller, & Popa, 2005) that $\text{certain}_{\mathcal{M}''}(Q, J) = Q(K_1)_{\downarrow} \cap \cdots \cap Q(K_\ell)_{\downarrow}$, for every query Q that is a union of conjunctive queries. Finally, since the result of chasing with Σ'' is the same as the result of chasing with Σ''', we have that $\text{certain}_{\mathcal{M}'''}(Q, J) = Q(K_1)_{\downarrow} \cap \cdots \cap Q(K_\ell)_{\downarrow} = \text{certain}_{\mathcal{M}''}(Q, J)$. Then we have that $\text{certain}_{\mathcal{M}'''}(Q, J) = \text{certain}_{\mathcal{M}''}(Q, J)$ for every target instance J and union of conjunctive queries Q, and then $\mathcal{M}''' \equiv_{\text{UCQ}} \mathcal{M}''$. $\square$

In the rest of this section we consider Σ''' as the set constructed from Σ'' as described above, and $\mathcal{M}'''$ as the ts-mapping specified by Σ'''. Before going to the last step of our procedure, it is worth recalling what is the form of the dependencies in Σ'''. Every $\sigma \in \Sigma'''$ is a dependency of the form

$$\varphi(\bar{x}) \wedge \mathbf{C}(\bar{x}) \wedge \delta(\bar{x}) \rightarrow \beta_1(\bar{x}) \vee \cdots \vee \beta_k(\bar{x}),$$

where (1) $\bar{x}$ is a tuple of distinct variables, (2) $\varphi(\bar{x})$ and $\beta_i(\bar{x})$ for $1 \leq i \leq k$, are conjunctive queries with exactly $\bar{x}$ as tuple of free variables, (3) $\mathbf{C}(\bar{x})$ is a conjunction of formulas $\mathbf{C}(x)$ for every x in $\bar{x}$, and (4) $\delta(\bar{x})$ is a conjunction of inequalities $x \neq x'$ for every pair of distinct variables x, x' in $\bar{x}$. In the last step in our procedure we eliminate the disjunctions from the conclusions in the dependencies of Σ''' to obtain a set of $\text{CQ}^{\mathbf{C},\neq}$-TO-CQ dependencies that specifies a CQ-maximum recovery of $\mathcal{M}$.

Eliminate the disjunctions in the conclusions of Σ'''

We explain first the machinery needed in this step of the procedure. We borrow some notions and tools from graph theory, in particular, properties about graph homomorphisms.

Given two instances J_1 and J_2 composed by constants and null values, we define the *product* of J_1 and J_2, denoted by $J_1 \times J_2$, as the instance constructed by the following procedure. Consider an injective function $f : (\mathbf{C} \cup \mathbf{N}) \times (\mathbf{C} \cup \mathbf{N}) \rightarrow (\mathbf{C} \cup \mathbf{N})$ such that (1) $f(a, a) = a$ for every constant value $a \in \mathbf{C}$, and (2) $f(a, b) = n_{(a,b)}$ is a null value, whenever a or b is a null value, or a and b are distinct constant values. Then for every n-ary relation symbol R and every pairs of n-tuples $\bar{a} = (a_1, \ldots, a_n) \in R^{J_1}$ and $\bar{b} = (b_1, \ldots, b_n) \in R^{J_2}$, the tuple $f(\bar{a}, \bar{b}) = (f(a_1, b_1), \ldots, f(a_n, b_n))$ is included in $R^{J_1 \times J_2}$.

For example, consider the instances

$$\begin{aligned} J_1 &= \{P(a,b), R(n_1,a), R(n_1,c)\} \\ J_2 &= \{P(a,c), R(n_2,a), R(n_2,c)\}, \end{aligned}$$

where a, b, c are distinct constant values, and n_1, n_2 are distinct null values. Then $J_1 \times J_2$ is the instance

$$J_1 \times J_2 = \{P(a,m_1), R(m_2,a), R(m_2,c), R(m_2,m_3), R(m_2,m_4)\},$$

where m_1, m_2, m_3, m_4 are distinct null values. In this case we have used a function f such that $f(b,c) = m_1$, $f(n_1,n_2) = m_2$, $f(a,c) = m_3$ and $f(c,a) = m_4$. Notice that the product of two instances could be the empty instance. For example, if we consider $J_1 = \{P(a,b)\}$ and $J_2 = \{R(a,b)\}$, then $J_1 \times J_2$ is the empty instance.

If we consider a schema with a single binary relation and two instances J_1 and J_2 composed only by null values, the product $J_1 \times J_2$ resembles exactly the graph-theoretical Cartesian product (Hell & Nešetřil, 2004). As for graph products, the operation $\times$ between instances satisfies several algebraic properties. To state them, let us introduce another useful operation among instances. The *null-disjoint union* of instances J_1 and J_2, denoted by $J_1 \uplus J_2$, is the instance constructed by first renaming the nulls in J_1 and J_2 such that they do not share null values, and then taking the set-theoretical union of the instances. The next lemma summarizes some of the algebraic properties of $\times$ and $\uplus$.

LEMMA 4.3.5 (c.f. Hell & Nešetřil, 2004). *Let J_1, J_2, and J_3 be instances composed by constant and null values.*

(1) *There exists a homomorphism from $J_1 \times J_2$ to J_1, and a homomorphism from $J_1 \times J_2$ to J_2.*

(2) *If there exists a homomorphism from J to J_1 and a homomorphism from J to J_2, then there exists a homomorphism from J to $J_1 \times J_2$.*

(3) *$J_1 \times J_1$ is homomorphically equivalent to J_1.*

(4) *$J_1 \times J_2$ is homomorphically equivalent to $J_2 \times J_1$.*

(5) $(J_1 \times J_2) \times J_3$ *is homomorphically equivalent to* $J_1 \times (J_2 \times J_3)$.

(6) $J_1 \times (J_2 \uplus J_3)$ *is homomorphically equivalent to* $(J_1 \times J_2) \uplus (J_1 \times J_3)$.

(7) $(J_1 \times J_2) \uplus J_3$ *is homomorphically equivalent to* $(J_1 \uplus J_3) \times (J_2 \uplus J_3)$.

These properties follows almost directly form the definitions. Just as an example we show property 7. By using property 6, we know that $(J_1 \uplus J_3) \times (J_2 \uplus J_3)$ is homomorphically equivalent to $((J_1 \uplus J_3) \times J_2) \uplus ((J_1 \uplus J_3) \times J_3)$ which is homomorphically equivalent to $(J_1 \times J_2) \uplus (J_1 \times J_3) \uplus (J_2 \times J_3) \uplus J_3$. Then we have that the identity is a homomorphism from $(J_1 \times J_2) \uplus J_3$ to $(J_1 \times J_2) \uplus (J_1 \times J_3) \uplus (J_2 \times J_3) \uplus J_3$. Now since there exists a homomorphism from $(J_1 \times J_3)$ to J_3 and from $(J_2 \times J_3)$ to J_3, we obtain that there exists a homomorphism from $(J_1 \times J_2) \uplus (J_1 \times J_3) \uplus (J_2 \times J_3) \uplus J_3$ to $(J_1 \times J_2) \uplus J_3$.

Let us highlight an important characteristic of the product of instances. Property 2 above says that, if we have any instance that has homomorphism to both J_1 and J_2, then it must also has a homomorphism to $J_1 \times J_2$. Intuitively, this property states that from all the space of possible instances, $J_1 \times J_2$ is the *closest instance* to both J_1 and J_2, taking homomorphisms as our proximity criterion. Since the answering process of conjunctive queries can be characterized in terms of homomorphism, this property gives us the following intuition. If some answer to a conjunctive query holds both in the evaluation of the query over J_1 and over J_2, then it holds in the evaluation of the query over $J_1 \times J_2$. And the opposite also holds. That is, if some answer holds in the evaluation over $J_1 \times J_2$, then we know that it also holds in both the evaluation over J_1 and over J_2. We will make extensively use of these properties.

The notion of product of instances can be also applied to conjunctive queries. Let Q_1 and Q_2 be two n-ary conjunctive queries, and assume that $\bar{x}$ is the tuple of free variables of Q_1 and Q_2. The *product* of Q_1 and Q_2, denoted by $Q_1 \times Q_2$, is defined as a k-ary conjunctive query (with $k \leq n$) constructed as follows. Let $f(\cdot, \cdot)$ be a one-to-one mapping from pairs of variables to variables such that: (1) $f(x, x) = x$ for every variable x in $\bar{x}$, and (2) $f(y, z)$ is a fresh variable (mentioned neither in Q_1 nor in Q_2) in any other case. Then for every pair of atoms $R(y_1, \ldots, y_m)$ in Q_1 and $R(z_1, \ldots, z_m)$ in Q_2, the atom

$R(f(y_1, z_1), \ldots, f(y_m, z_m))$ is included as a conjunct in the query $Q_1 \times Q_2$. Furthermore, the set of free variables of $Q_1 \times Q_2$ is the set of variables from $\bar{x}$ that are mentioned in $Q_1 \times Q_2$. For example, consider conjunctive queries: $Q_1(x_1, x_2) = P(x_1, x_2) \wedge R(x_2, x_1)$ and $Q_2(x_1, x_2) = \exists y\,(P(x_1, y) \wedge R(y, x_2))$. Then we have that:

$$(Q_1 \times Q_2)(x_1) \;\; = \;\; \exists z_1 \exists z_2\,(P(x_1, z_1) \wedge R(z_1, z_2)).$$

In this case, we used a mapping f such that $f(x_1, x_1) = x_1$, $f(x_2, y) = z_1$ and $f(x_1, x_2) = z_2$. As shown in the example, the free variables of $Q_1 \times Q_2$ do not necessarily coincide with the free variables of Q_1 and Q_2. Notice that the product of two queries may be empty. For example, if $Q_1 = \exists y_1 \exists y_2 P(y_1, y_2)$ and $Q_2 = \exists z_1 R(z_1, z_1)$, then $Q_1 \times Q_2$ is empty.

We have all the necessary ingredients to construct a set of dependencies $\Sigma^\star$ from Σ''', such that the dependencies in $\Sigma^\star$ do not have disjunctions in their conclusions, and such that $\Sigma^\star$ defines a CQ-maximum recovery of $\mathcal{M}$.

Procedure ELIMINATEDISJUNCTIONS(Σ''')

(1) Let $\Sigma^\star$ be empty.

(2) For every dependency in Σ''' of the form

$$\varphi(\bar{x}) \wedge \mathbf{C}(\bar{x}) \wedge \delta(\bar{x}) \rightarrow \beta_1(\bar{x}) \vee \cdots \vee \beta_k(\bar{x}),$$

do the following. If $\beta_1(\bar{x}) \times \cdots \times \beta_k(\bar{x})$ is not empty, then add the dependency

$$\varphi(\bar{x}) \wedge \mathbf{C}(\bar{x}) \wedge \delta(\bar{x}) \rightarrow \beta_1(\bar{x}) \times \cdots \times \beta_k(\bar{x})$$

to $\Sigma^\star$.

(3) Return $\Sigma^\star$. □

For example, assume that the following dependency is in Σ''':

$$A(x_1, x_2) \wedge \mathbf{C}(x_1) \wedge \mathbf{C}(x_2) \wedge x_1 \neq x_2 \ \rightarrow \\ [\ R(x_1, x_2) \wedge R(x_1, x_1)\ ] \ \vee \ [\ \exists y_1 (P(x_1, y_1) \wedge R(x_2, x_2))\ ]. \quad (4.1)$$

Then we add to $\Sigma^\star$ the dependency

$$A(x_1, x_2) \wedge \mathbf{C}(x_1) \wedge \mathbf{C}(x_2) \wedge x_1 \neq x_2 \ \rightarrow \ \exists z_1 \exists z_2 (R(z_1, x_2) \wedge R(z_2, z_2)) \quad (4.2)$$

since $\exists z_1 \exists z_2 (R(z_1, x_2) \wedge R(z_2, z_2))$ is the result of

$$[R(x_1, x_2) \wedge R(x_1, x_1)] \times [\exists y_1 (P(x_1, y_1) \wedge R(x_2, x_2))].$$

Notice that the obtained set of dependencies $\Sigma^\star$, is a set of $\text{CQ}^{\mathbf{C},\neq}$-TO-CQ dependencies. The following is the key property of $\Sigma^\star$.

LEMMA 4.3.6. *Let $\mathcal{M}^\star$ be the ts-mapping specified by the set $\Sigma^\star$ constructed in procedure* ELIMINATEDISJUNCTIONS(Σ'''). *Then $\mathcal{M}^\star$ is* CQ*-equivalent with $\mathcal{M}'''$.*

PROOF. Before proving that $\mathcal{M}^\star \equiv_{\text{CQ}} \mathcal{M}'''$, let us give some intuition of why this result holds. Let Γ_1 be the set containing the dependency of equation (4.1) above, and let Γ_2 be the set containing the dependency of equation (4.2). Consider the instance $J = \{A(a, b)\}$, with a, b distinct constant values. When we chase J with Γ_1, we obtain the set of instances $\mathcal{V} = \{K_1, K_2\}$ with $K_1 = \{R(a, b), R(a, a)\}$ and $K_2 = \{P(a, n), R(b, b)\}$, where n is a null value. Recall that every solution K of J under Γ_1 is such that, there exists a homomorphism from K_1 to K, or there exists a homomorphism from K_2 to K. Intuitively, when considering the *conjunctive information* contained in the space of solutions of J under Γ_1, the things that we know for sure are, that the value b appears in the second component of relation R, and that some element appears in a single tuple in both components of R. If we now chase J with Γ_2, we obtain the instance $K = \{R(n_1, b), R(n_2, n_2)\}$ with n_1, n_2 null values, that carries exactly the conjunctive information contained in $\mathcal{V}$. In fact, it can be formally proved that the certain answers under Γ_1 coincides with the certain

answers under Γ_2 for every conjunctive query. In this example, it is clear that this last property does not hold if we consider unions of conjunctive queries (consider for example the query $\exists u P(x_1, u) \vee \exists u R(x_1, u)$).

We show now that $\mathcal{M}^\star \equiv_{\text{CQ}} \mathcal{M}'''$. We prove first the following Let J be an instance composed by null and constant values. Assume that $\mathcal{V} = \{K_1, \ldots, K_\ell\}$ is the result of chasing J with Σ''', and K the result of chasing J with $\Sigma^\star$. Then we claim that K is homomorphically equivalent to $K_1 \times \cdots \times K_\ell$.

To prove the above mentioned property we first reformulate the process of chasing with Σ''' by using the operation $\uplus$ between instances. Recall that every dependency σ in Σ''' is of the form

$$\varphi(\bar{x}) \wedge \mathbf{C}(\bar{x}) \wedge \delta(\bar{x}) \rightarrow \beta_1(\bar{x}) \vee \cdots \vee \beta_k(\bar{x}).$$

Let J be an instance and consider the set A_J of all the pairs $(\bar{a}, \sigma)$ such that: (1) $\bar{a}$ is a tuple of elements in $\text{dom}(J)$, (2) $\sigma \in \Sigma'''$ is a dependency of the above form, and (3) $J \models \varphi(\bar{a}) \wedge \mathbf{C}(\bar{a}) \wedge \delta(\bar{a})$. The idea is that the set A_J contains all the possible *assignments* to the premises of the dependencies in Σ''', such that the premises hold in J. Notice that, if a pair $(\bar{a}, \sigma)$ belongs to A_J, then since $\mathbf{C}(\bar{a}) \wedge \delta(\bar{a})$ holds, we have that $\bar{a}$ is a tuple of distinct constant values. We define now the notion of *choice* function. Consider a function f from A_J to the natural numbers, such that for every $(\bar{a}, \sigma)$ it holds that $f(\bar{a}, \sigma) \in \{1, \ldots, k\}$, whenever σ has k disjuncts in its conclusion. Choice functions are used to select a particular disjunct when we apply a dependency to instance J while computing a disjunctive chase. Let F_J be the set of all choice functions with domain A_J. Notice that, since J is a finite instance and Σ''' is a finite set of dependencies, A_J and F_J are finite sets. We need an additional notion. Given a conjunctive query $\beta(\bar{x})$ and an assignment $\bar{a}$ for the variables $\bar{x}$, we denote by $I_{\beta(\bar{a})}$ the instance constructed by considering the atoms of $\beta(\bar{a})$, where every existentially quantified variable has been replaced by a fresh null value. We can formally define now the process of chasing with Σ''' in terms of A_J and F_J.

For every $f \in F_J$ we denote by K_f the instance:

$$\biguplus\{I_{\beta_i(\bar{a})} \mid (\bar{a}, \sigma) \in A_J,\ \beta_i(\bar{x}) \text{ is a disjunct in the conclusion of } \sigma, \text{ and } i = f(\bar{a}, \sigma)\}.$$

It is clear that K_f as defined above, is a chase of J with Σ'''. Then the (disjunctive) chase of J with Σ''' is the set $\mathcal{V} = \{K_f \mid f \in F_J\}$.

We now reformulate the chase of J with $\Sigma^\star$ by using the operations $\uplus$ and $\times$ between instances. Let σ be a dependency in $\Sigma^\star$ of the form

$$\varphi(\bar{x}) \wedge \mathbf{C}(\bar{x}) \wedge \delta(\bar{x}) \rightarrow \beta_1(\bar{x}) \times \cdots \times \beta_k(\bar{x}).$$

Notice that, if $\bar{a}$ is a tuple in $\mathrm{dom}(J)$ such that $J \models \varphi(\bar{a}) \wedge \mathbf{C}(\bar{a}) \wedge \delta(\bar{a})$, then the atoms of $\beta_1(\bar{a}) \times \cdots \times \beta_k(\bar{a})$, where every existentially quantified variable has been replaced by a fresh null value, are added to the chase of J with $\Sigma^\star$. That is, the instance $I_{\beta_1(\bar{a}) \times \cdots \times \beta_k(\bar{a})}$ is added to the chase of J with $\Sigma^\star$. The crucial observation here is that, since $\bar{a}$ is a tuple of distinct constant values, then the instance $I_{\beta_1(\bar{a}) \times \cdots \times \beta_k(\bar{a})}$, equals the product of instances $I_{\beta_1(\bar{a})} \times \cdots \times I_{\beta_k(\bar{a})}$ (up-to isomorphic image of null values). It should be noticed that this last property does not hold if $\bar{a}$ is a tuple where some values are repeated. For example, if $\beta_1(x_1, x_2) = R(x_1, x_1, x_2)$ and $\beta_2(x_1, x_2) = R(x_1, x_2, x_2)$ then $\beta_1(x_1, x_2) \times \beta_2(x_1, x_2)$ is the query $\exists u_1 R(x_1, u_1, x_2)$. Now if we consider tuple $\bar{a} = (a, a)$, then we have that $I_{\beta_1(a,a) \times \beta_2(a,a)}$ is the instance $\{R(a, n, a)\}$ with n a null value, while $I_{\beta_1(a,a)} \times I_{\beta_2(a,a)} = \{R(a, a, a)\}$.

Consider $A^\star_J$ the set of pairs defined as for Σ''' but considering the dependencies in $\Sigma^\star$. We can now reformulate the chase of J with $\Sigma^\star$ in terms of $A^\star_J$, $\uplus$ and $\times$ as:

$$K = \biguplus\{I_{\beta_1(\bar{a})} \times \cdots \times I_{\beta_k(\bar{a})} \mid (\bar{a}, \sigma) \in A^\star_J \text{ and } \beta_1(\bar{x}) \times \cdots \times \beta_k(\bar{x}) \text{ is the conclusion of } \sigma\}.$$

To conclude the proof of the claim, we must show that the product of the instances in $\mathcal{V} = \{K_f \mid f \in F_J\}$, is homomorphically equivalent to instance K above. Fix a pair $(\bar{a}', \sigma')$ in A_J and assume that $\beta'_1(\bar{x}') \vee \cdots \vee \beta'_{k'}(\bar{x}')$ is the conclusion of σ'. Notice that, if

for a particular function $f \in F_J$ it is the case that $f(\bar{a}', \sigma') = i'$, then we can write K_f as

$$I_{\beta'_{i'}(\bar{a}')} \uplus \biguplus\{I_{\beta_i(\bar{a})} \mid (\bar{a}, \sigma) \in A_J \setminus \{(\bar{a}', \sigma')\},\ \beta_i(\bar{x}) \text{ is a disjunct in } \sigma, \text{ and } i = f(\bar{a}, \sigma)\}$$

Using this observation, it is straightforward to see that we can write the product of the instances in $\mathcal{V}$ as

$$\bigtimes_{i'=1}^{k'} \Bigg(\bigtimes_{f \in F_J} \Bigg(I_{\beta'_{i'}(\bar{a}')} \uplus \biguplus\{I_{\beta_i(\bar{a})} \mid (\bar{a}, \sigma) \in A_J \setminus \{(\bar{a}', \sigma')\},\ \beta_i(\bar{x}) \text{ is a disjunct in } \sigma, \text{ and } i = f(\bar{a}, \sigma)\} \Bigg) \Bigg).$$

By applying property 7 of Lemma 4.3.5, we know that the above instance is homomorphically equivalent to

$$\bigtimes_{i'=1}^{k'} \Bigg(I_{\beta'_{i'}(\bar{a}')} \uplus \bigtimes_{f \in F_J} \Bigg(\biguplus\{I_{\beta_i(\bar{a})} \mid (\bar{a}, \sigma) \in A_J \setminus \{(\bar{a}', \sigma')\},\ \beta_i(\bar{x}) \text{ is a disjunct in } \sigma, \text{ and } i = f(\bar{a}, \sigma)\} \Bigg) \Bigg),$$

and by applying property 7 again, we obtain that this last instance is homomorphically equivalent to

$$\left(I_{\beta'_1(\bar{a}')} \times \cdots \times I_{\beta'_{k'}(\bar{a}')}\right) \uplus \bigtimes_{f \in F_J} \Bigg(\biguplus\{I_{\beta_i(\bar{a})} \mid (\bar{a}, \sigma) \in A_J \setminus \{(\bar{a}', \sigma')\},\ \beta_i(\bar{x}) \text{ is a disjunct in } \sigma, \text{ and } i = f(\bar{a}, \sigma)\} \Bigg) \Bigg).$$

If we continue separating one by one the elements in A_J, we finally obtain that the above instance is homomorphically equivalent to the instance

$$\biguplus\{I_{\beta_1(\bar{a})} \times \cdots \times I_{\beta_k(\bar{a})} \mid (\bar{a}, \sigma) \in A_J \text{ and } \beta_1(\bar{x}) \vee \cdots \vee \beta_k(\bar{x}) \text{ is the conclusion of } \sigma\}.$$

It is straightforward to see that this last instance is homomorphically equivalent to K. Just notice that, if for a dependency $\sigma \in \Sigma'''$ of the form $\varphi(\bar{x}) \wedge \mathbf{C}(\bar{x}) \wedge \delta(\bar{x}) \rightarrow \beta_1(\bar{x}) \vee \cdots \vee \beta_k(\bar{x})$, we do not include a corresponding dependency in $\Sigma^\star$, then $\beta_1(\bar{x}) \times \cdots \times \beta_k(\bar{x})$

is empty, which implies that $I_{\beta_1(\bar{a})} \times \cdots \times I_{\beta_k(\bar{a})}$ is the empty instance for every tuple $\bar{a}$ of distinct constant values. Thus we have shown that K homomorphically equivalent to $K_1 \times \cdots \times K_\ell$ with $\mathcal{V} = \{K_1, \ldots, K_\ell\}$ the result of chasing J with Σ'''. This completes the proof of the claim.

We are ready now to prove that $\mathcal{M}^\star \equiv_{\text{CQ}} \mathcal{M}'''$. Let J be a target instance (composed by constant and null values), and $\mathcal{V} = \{K_1, \ldots, K_\ell\}$ the result of chasing J with Σ''', and K the result of chasing J with $\Sigma^\star$. Notice that every dependency σ in Σ''' is such that, for every variable x that occurs simultaneously in the premise and the conclusion of σ, the atom $\mathbf{C}(x)$ also occurs in the premise of σ. From this last fact and the properties of $\mathcal{V}$, it follows directly that $\text{certain}_{\mathcal{M}'''}(Q, J) = Q(K_1)_\downarrow \cap \cdots \cap Q(K_\ell)_\downarrow$, for every conjunctive query Q. Similarly, for $\mathcal{M}^\star$ we have that, for every conjunctive query it holds that $\text{certain}_{\mathcal{M}^\star}(Q, J) = Q(K)_\downarrow$. Then we only have to prove that $Q(K)_\downarrow = Q(K_1)_\downarrow \cap \cdots \cap Q(K_\ell)_\downarrow$, for every conjunctive query Q. Let Q be an n-ary conjunctive query with free variables $\bar{x}$. We first show that $Q(K)_\downarrow \subseteq Q(K_1)_\downarrow \cap \cdots \cap Q(K_\ell)_\downarrow$. Let $\bar{a}$ be an n-ary tuple such that $\bar{a} \in Q(K)_\downarrow$. Since K is homomorphically equivalent to the product $K_1 \times \ldots \times K_\ell$ we know that there exists a homomorphism from K to every K_i with $1 \leq i \leq \ell$, and then $\bar{a} \in Q(K_i)_\downarrow$ for every $1 \leq i \leq \ell$. Now, to show that $Q(K_1)_\downarrow \cap \cdots \cap Q(K_\ell)_\downarrow \subseteq Q(K)_\downarrow$, assume that $\bar{a} \in Q(K_1)_\downarrow \cap \cdots \cap Q(K_\ell)_\downarrow$. Then we know that, for every K_i with $1 \leq i \leq \ell$ there exists a homomorphism h from the atoms in Q to K_i, such that $h(\bar{x}) = \bar{a}$. Then by the properties of the product of instances (property 2 of Lemma 4.3.5), we know that there exists a homomorphism h' from the atoms in Q to K, such that $h(\bar{x}) = \bar{a}$, and then $\bar{a} \in Q(K)$. We have shown that $Q(K)_\downarrow = Q(K_1)_\downarrow \cap \cdots \cap Q(K_\ell)_\downarrow$ for every conjunctive query Q, which implies that $\mathcal{M}^\star \equiv_{\text{CQ}} \mathcal{M}'''$. This concludes the proof of the lemma. □

Putting it all togehter

The following is the complete algorithm that uses all the previous procedures to compute CQ-maximum recoveries of st-mappings specified by $\text{CQ}^{\mathbf{C},\neq}$-TO-CQ.

Algorithm CQ-MAXIMUMRECOVERY($\mathcal{M}$)

Input: An st-mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where Σ is a set of CQ$^{\mathbf{C},\neq}$-TO-CQ dependencies.

Output: A ts-mapping $\mathcal{M}^\star = (\mathbf{T}, \mathbf{S}, \Sigma^\star)$, where $\Sigma^\star$ is a set of CQ$^{\mathbf{C},\neq}$-TO-CQ dependencies such that $\mathcal{M}^\star$ is a CQ-maximum recovery of $\mathcal{M}$.

(1) Let $\bar{\Sigma}$ the set of dependencies obtained from Σ by dropping all the atoms of the form $\mathbf{C}(x)$ that appear in the premises of Σ, and let $\bar{\mathcal{M}} = (\mathbf{S}, \mathbf{T}, \bar{\Sigma})$.
(2) Let $\mathcal{M}' = (\mathbf{T}, \mathbf{S}, \Sigma')$ be the output of algorithm MAXIMUMRECOVERY($\bar{\mathcal{M}}$).
(3) Let Σ'' be the output of procedure ELIMINATEEQINEQ(Σ').
(4) Let Σ''' be the set obtained from Σ'' by dropping all the inequalities that appear in the conclusion of the dependencies in Σ''
(5) Let $\Sigma^\star$ be the output of procedure ELIMINATEDISJUNCTIONS(Σ''').
(6) Return $\mathcal{M}^\star = (\mathbf{T}, \mathbf{S}, \Sigma^\star)$.

□

THEOREM 4.3.7. *Let $\mathcal{M}$ be an st-mapping specified by a set of* CQ$^{\mathbf{C},\neq}$-TO-CQ *dependencies. Then algorithm* CQ-MAXIMUMRECOVERY($\mathcal{M}$) *computes a* CQ*-maximum recovery of $\mathcal{M}$ specified by set of* CQ$^{\mathbf{C},\neq}$-TO-CQ *dependencies.*

PROOF. The proof follows from Lemmas 4.3.3, 4.3.4, and 4.3.6. From Lemma 4.3.3 we know that the mapping $\mathcal{M}''$ specified by the set Σ'' computed in Step 3, is a maximum recovery of $\mathcal{M}$. Moreover, we know that Σ'' is a set of CQ$^{\mathbf{C},\neq}$-TO-UCQ$^{\neq}$ dependencies. Now, from Lemma 4.3.4 we know that the mapping $\mathcal{M}'''$ specified by the set Σ''' computed in Step 4, is UCQ-equivalent with $\mathcal{M}''$. In this case we have that Σ''' is a set of CQ$^{\mathbf{C},\neq}$-TO-UCQ dependencies. By Lemma 4.3.6 we have that the mapping $\mathcal{M}^\star$ specified by the set $\Sigma^\star$ computed in Step 5, is CQ-equivalent with $\mathcal{M}'''$, and moreover, $\Sigma^\star$ is a set of CQ$^{\mathbf{C},\neq}$-TO-CQ dependencies. Thus, since $\mathcal{M}^\star$ is CQ-equivalent with $\mathcal{M}'''$ which is UCQ-equivalent with $\mathcal{M}''$, we obtain that $\mathcal{M}^\star$ is CQ-equivalent with $\mathcal{M}''$. Finally, since $\mathcal{M}''$ is a maximum recovery of $\mathcal{M}$, from Proposition 4.2.6 we conclude that $\mathcal{M}^\star$ is a CQ-maximum

recovery of $\mathcal{M}$. Thus we have shown that the output of CQ-MAXIMUMRECOVERY($\mathcal{M}$) is a CQ-maximum recovery of $\mathcal{M}$ specified by a set of $\text{CQ}^{\mathbf{C},\neq}$-TO-CQ dependencies. □

Theorem 4.3.1 follows directly from Theorem 4.3.7.

4.3.2. $\text{CQ}^{\mathbf{C},\neq}$-TO-CQ is the right language

Most of the dependencies considered in the data exchange literature (Fagin, Kolaitis, Miller, & Popa, 2005; Fagin, 2007; Fagin, Kolaitis, Popa, & Tan, 2008; Arenas et al., 2008) are $\mathcal{L}_1$-TO-$\mathcal{L}_2$ dependencies, where $\mathcal{L}_1$ and $\mathcal{L}_2$ are fragments of $\text{UCQ}^{\mathbf{C},\neq}$. In this section, we show that among all of them, $\text{CQ}^{\mathbf{C},\neq}$-TO-CQ is the *right* language for the notion of CQ-maximum recovery; if one adds or removes features from this class of dependencies, then closure under CQ-maximum recovery does not longer hold. More precisely, we start by showing that both inequalities and predicate $\mathbf{C}(\cdot)$ are necessary for the closure property of Theorem 4.3.1.

THEOREM 4.3.8.

(a) *There exists an st-mapping specified by a set of st-tgds that has no* CQ*-maximum recovery specified by a set of* $\text{CQ}^{\mathbf{C}}$-TO-CQ *dependencies.*

(b) *There exists an st-mapping specified by a set of st-tgds that has no* CQ*-maximum recovery specified by a set of* $\text{CQ}^{\neq}$-TO-CQ *dependencies.*

PROOF. (a) Let $\mathbf{S} = \{A(\cdot,\cdot), B(\cdot,\cdot)\}$, $\mathbf{T} = \{P(\cdot,\cdot)\}$ and $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where Σ is the following set of st-tgds:

$$\Sigma \;=\; \{A(x,y) \rightarrow P(x,y),\ B(x,x) \rightarrow P(x,x)\}.$$

Next we show that $\mathcal{M}$ does not have a CQ-maximum recovery specified by a set of $\text{CQ}^{\mathbf{C}}$-TO-CQ dependencies.

For the sake of contradiction, assume that there exists a mapping $\mathcal{M}'$ specified by a set Σ' of $\text{CQ}^{\mathbf{C}}$-TO-CQ dependencies such that $\mathcal{M}'$ is a CQ-maximum recovery of $\mathcal{M}$. Next we show that this leads to a contradiction by considering two cases.

(I) Assume that Σ' is empty. Then, for every instance J of $\mathbf{T}$ and K of $\mathbf{S}$, we have that $(J, K) \models \Sigma'$. Thus, by the definition of Σ, we conclude that for every pair of source instances I and I', it holds that $(I, I') \in \mathcal{M} \circ \mathcal{M}'$. Let $\mathcal{M}''$ be a ts-mapping specified by dependency $P(x, y) \wedge x \neq y \rightarrow A(x, y)$. It is straightforward to prove that $\mathcal{M}''$ is a recovery of $\mathcal{M}$, which implies that $\mathcal{M}''$ is a CQ-recovery of $\mathcal{M}$. Thus, given that we assume that $\mathcal{M}'$ is a CQ-maximum recovery of $\mathcal{M}$, we have that $\mathcal{M}'' \preceq^{\text{CQ}}_{\mathcal{M}} \mathcal{M}'$. But for a source instance I such that $A^I = B^I = \{(a, b)\}$, where $a \neq b$, and Boolean query $Q = \exists x \exists y\, A(x, y)$, we have that $Q(I) = \underline{\text{true}} = \text{certain}_{\mathcal{M} \circ \mathcal{M}''}(Q, I)$ but $\text{certain}_{\mathcal{M} \circ \mathcal{M}'}(Q, I) = \underline{\text{false}}$, a contradiction.

(II) Assume that Σ' is nonempty, and let I_1 and I_2 be instances of $\mathbf{S}$ such that:

$$\begin{array}{lcllcl} A^{I_1} & = & \{(a, a)\} & \qquad A^{I_2} & = & \emptyset \\ B^{I_1} & = & \emptyset & \qquad B^{I_2} & = & \{(a, a)\} \end{array}$$

Let J be the instance such that $P^J = \{(a, a)\}$. Notice that J is the canonical universal solution for both I_1 and I_2 under Σ. Assume that K is the result of chasing J with Σ'. Notice that K could not be a valid source instance (it may contain null values). Nevertheless, since J is composed only by constant values, and by using the properties of the chase (Fagin, Kolaitis, Miller, & Popa, 2005; Fagin, Kolaitis, Popa, & Tan, 2005, 2008), we know that for every Q that is a union of conjunctive queries, it holds that $\text{certain}_{\mathcal{M} \circ \mathcal{M}'}(Q, I_1) = \text{certain}_{\mathcal{M} \circ \mathcal{M}'}(Q, I_2) = Q(K)_{\downarrow}$. We use this last fact and the fact that Σ' is nonempty, to derive a contradiction.

Given that Σ' is nonempty, there exists a $\text{CQ}^{\mathbf{C}}$-TO-CQ dependency:

$$\varphi(x_1, \ldots x_m) \;\; \rightarrow \;\; \exists y_1 \cdots \exists y_n\, \psi(x_1, \ldots, x_m, y_1, \ldots, y_n)$$

that belongs to Σ'. Thus, we have that (J, K) satisfies this constraint. But this implies that K is nonempty since $P^J = \{(a, a)\}$ and $J \models \varphi(a, \ldots, a)$ (since a is a constant and φ is a query in $\text{CQ}^{\mathbf{C}}$ over $\mathbf{T}$).

Given that K is nonempty, we have that $A^K \neq \emptyset$ or $B^K \neq \emptyset$. If $A^K \neq \emptyset$, then let Q_A be Boolean query $\exists x \exists y A(x, y)$. We know that $\mathrm{certain}_{\mathcal{M} \circ \mathcal{M}'}(Q_A, I_2) = Q_A(K)$. Thus, given that $A^K \neq \emptyset$, we conclude that $\mathrm{certain}_{\mathcal{M} \circ \mathcal{M}'}(Q_A, I_2) =$ true. But this leads to a contradiction since we assume that $\mathcal{M}'$ is a CQ-recovery of $\mathcal{M}$ and $Q_A(I_2) =$ false. If $B^K \neq \emptyset$, then we obtain a similar contradiction by considering Boolean query $Q_B = \exists x \exists y B(x, y)$ and source instance I_1. This concludes the proof of the first part of the theorem.

(b) Let $\mathbf{S} = \{A(\cdot), B(\cdot)\}$, $\mathbf{T} = \{P(\cdot)\}$ and $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where Σ is the following set of st-tgds:

$$\Sigma \;=\; \{A(x) \rightarrow \exists y P(y),\ B(x) \rightarrow P(x)\}.$$

Next we show that $\mathcal{M}$ does not have a CQ-maximum recovery specified by a set of CQ$^{\neq}$-TO-CQ dependencies. For the sake of contradiction, assume that $\mathcal{M}'$ is a mapping specified by a set Σ' of CQ$^{\neq}$-TO-CQ dependencies such that $\mathcal{M}'$ is a CQ-maximum recovery of $\mathcal{M}$.

Let $\mathcal{M}^\star$ be a mapping $(\mathbf{T}, \mathbf{S}, \Sigma^\star)$, where $\Sigma^\star$ is the set of ts-tgds $\{P(x) \wedge \mathbf{C}(x) \rightarrow B(x)\}$. Next we show that $\mathcal{M}^\star$ is a CQ-recovery of $\mathcal{M}$. Let I be an instance of $\mathbf{S}$ and Q a conjunctive query over $\mathbf{S}$, and assume that $J = \mathrm{chase}_{\Sigma^\star}(\mathrm{chase}_{\Sigma}(I))$. It is straightforward to prove that $A^J = \emptyset$ and $B^J = B^I$. Thus, we have that $J \subseteq I$, which implies that $Q(J) \subseteq Q(I)$. Given that $J \in \mathrm{Sol}_{\mathcal{M} \circ \mathcal{M}^\star}(I)$ in this case, we conclude that $\mathrm{certain}_{\mathcal{M} \circ \mathcal{M}^\star}(Q, I) \subseteq Q(I)$, which implies that $\mathcal{M}^\star$ is a CQ-recovery of $\mathcal{M}$.

Let I_1 and I_2 be instances of $\mathbf{S}$ such that $I_1 = \{A(a)\}$ and $I_2 = \{B(a)\}$, where a is an arbitrary element of $\mathbf{C}$, and Q_B be Boolean query $\exists x B(x)$. It is straightforward to prove that $\mathrm{certain}_{\mathcal{M} \circ \mathcal{M}^\star}(Q_B, I_2) =$ true. Thus, we have that $\mathrm{certain}_{\mathcal{M} \circ \mathcal{M}'}(Q_B, I_2) =$ true since $\mathcal{M}^\star$ is a CQ-recovery of $\mathcal{M}$ and $\mathcal{M}'$ is a CQ-maximum recovery of $\mathcal{M}$. Next we use this fact to prove that $\mathrm{certain}_{\mathcal{M} \circ \mathcal{M}'}(Q_B, I_1) =$ true. Let $J \in \mathrm{Sol}_{\mathcal{M} \circ \mathcal{M}'}(I_1)$. Then there exists an instance K of $\mathbf{T}$ such that $(I_1, K) \models \Sigma$ and $(K, J) \models \Sigma'$. Let $f : \mathrm{dom}(K) \cup \mathrm{dom}(J) \rightarrow \mathbf{C}$ be a one-to-one mapping such that $f(u) = a$ for some $u \in \mathrm{dom}(K)$. We

note that such a function exists since K is not empty. By the definition of Σ, we have that $(I_2, f(K)) \models \Sigma$. Furthermore, given that Σ' is a set of CQ$^{\neq}$-TO-CQ dependencies, we have that $(f(K), f(J)) \models \Sigma'$. Thus, we have that $(I_2, f(J)) \in \mathcal{M} \circ \mathcal{M}'$, from which we conclude that $f(J) \models \exists x B(x)$ (since $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q_B, I_2) = \underline{\text{true}}$). Therefore, given that J and $f(J)$ are isomorphic instances (not considering predicate $\mathbf{C}$), we have that $J \models \exists x B(x)$. We conclude that for every $J \in \mathrm{Sol}_{\mathcal{M}\circ\mathcal{M}'}(I_1)$, it holds that $J \models \exists x B(x)$. Thus, we have that $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q_B, I_1) = \underline{\text{true}}$. But this leads to a contradiction since $Q_B(I_1) = \underline{\text{false}}$ and $\mathcal{M}'$ is assumed to be a CQ-maximum recovery of $\mathcal{M}$. This concludes the proof of the theorem. □

We have shown that both inequalities and predicate $\mathbf{C}(\cdot)$ are necessary for the closure property of Theorem 4.3.1. A natural question is whether a similar closure property can be obtained if one adds extra features to CQ$^{\mathbf{C},\neq}$-TO-CQ. For example, it could be the case that CQ$^{\mathbf{C},\neq}$-TO-CQ$^{\neq}$ is closed under CQ-maximum recovery. The next proposition shows that if one includes disjunctions, inequalities or predicate $\mathbf{C}$ in the conclusions of the dependencies, then mappings do not necessarily admit CQ-maximum recoveries, even if these features are added to st-tgds. Before stating the result we provide a necessary condition that can be used to show that some mappings do not admit $\mathcal{C}$-maximum recoveries. Recall that in Section 4.1.1 we defined given a mapping $\mathcal{M}$ and an instance I, the set $\mathrm{Sub}_{\mathcal{M}}(I)$ as the set of instances K such that $\mathrm{Sol}_{\mathcal{M}}(K) \subseteq \mathrm{Sol}_{\mathcal{M}}(I)$. Now given a set of instances $\mathcal{S}$, define $\mathrm{Sub}_{\mathcal{M}}(\mathcal{S})$ as

$$\mathrm{Sub}_{\mathcal{M}}(\mathcal{S}) = \bigcup_{I \in \mathcal{S}} \mathrm{Sub}_{\mathcal{M}}(I).$$

That is, $\mathrm{Sub}_{\mathcal{M}}(\mathcal{S})$ is the set of instances K such that there exists $I \in \mathcal{S}$ for which $\mathrm{Sol}_{\mathcal{M}}(K) \subseteq \mathrm{Sol}_{\mathcal{M}}(I)$. Moreover, similarly as we defined the set $\mathrm{Inf}_{\mathcal{M}}(Q, I)$, we define the set of tuples $\mathrm{Inf}_{\mathcal{M}}(Q, \mathcal{S})$ with Q a query and $\mathcal{S}$ a set of instances as:

$$\mathrm{Inf}_{\mathcal{M}}(Q, \mathcal{S}) \;=\; \bigcap \{Q(K) \mid K \in \mathrm{Sub}_{\mathcal{M}}(\mathcal{S})\}.$$

PROPOSITION 4.3.9. *If $\mathcal{M}$ has a $\mathcal{C}$-maximum recovery, then for every $I \in \mathrm{dom}(\mathcal{M})$, $Q \in \mathcal{C}$ and $\mathcal{S} \subseteq \mathrm{dom}(\mathcal{M})$ such that $\mathrm{Sol}_{\mathcal{M}}(I) \subseteq \bigcup_{I' \in \mathcal{S}} \mathrm{Sol}_{\mathcal{M}}(I')$, it holds that:*

$$\mathrm{Inf}_{\mathcal{M}}(Q, \mathcal{S}) \;\subseteq\; \mathrm{Inf}_{\mathcal{M}}(Q, I).$$

PROOF. Let $\mathcal{M}$ be a mapping from a schema $\mathbf{R}_1$ to a schema $\mathbf{R}_2$. For the sake of contradiction, assume that $\mathcal{M}$ has a $\mathcal{C}$-maximum recovery and there exists an instance $I \in \mathrm{dom}(\mathcal{M})$, a query $Q \in \mathcal{C}$ and a subset $\mathcal{S} \subseteq \mathrm{dom}(\mathcal{M})$ such that, $\mathrm{Sol}_{\mathcal{M}}(I) \subseteq \bigcup_{I' \in \mathcal{S}} \mathrm{Sol}_{\mathcal{M}}(I')$ but

$$\mathrm{Inf}_{\mathcal{M}}(Q, \mathcal{S}) \;\not\subseteq\; \mathrm{Inf}_{\mathcal{M}}(Q, I).$$

Thus, there exists a tuple $\bar{t} \in \mathrm{Inf}_{\mathcal{M}}(Q, \mathcal{S})$ such that $\bar{t} \notin \mathrm{Inf}_{\mathcal{M}}(Q, I)$. Since $\mathcal{M}$ has a $\mathcal{C}$-maximum recovery, we know by Theorem 4.1.9 that $\mathcal{M}^{\mathcal{C}}$ is a $\mathcal{C}$-recovery of $\mathcal{M}$, which implies that $\mathrm{Sol}_{\mathcal{M} \circ \mathcal{M}^{\mathcal{C}}}(I) \neq \emptyset$. Let $I_1 \in \mathrm{Sol}_{\mathcal{M} \circ \mathcal{M}^{\mathcal{C}}}(I)$. Then there exists $J \in \mathrm{Inst}(\mathbf{R}_2)$ such that $(I, J) \in \mathcal{M}$ and $(J, I_1) \in \mathcal{M}^{\mathcal{C}}$. By the definition of $\mathcal{M}^{\mathcal{C}}$, we have that J is a $\mathcal{C}$-witness of I_1 under $\mathcal{M}$. Given that $\mathrm{Sol}_{\mathcal{M}}(I) \subseteq \bigcup_{I' \in \mathcal{S}} \mathrm{Sol}_{\mathcal{M}}(I')$ and $J \in \mathrm{Sol}_{\mathcal{M}}(I)$, we know that there exists an instance $I_2 \in \mathcal{S}$ such that $J \in \mathrm{Sol}_{\mathcal{M}}(I_2)$. Therefore, given that J is a $\mathcal{C}$-witness of I_1 under $\mathcal{M}$ and $J \in \mathrm{Sol}_{\mathcal{M}}(I_2)$, it holds that:

$$\mathrm{Inf}_{\mathcal{M}}(Q, I_2) \;\subseteq\; Q(I_1).$$

But given that $I_2 \in \mathcal{S}$, we have that:

$$\mathrm{Inf}_{\mathcal{M}}(Q, \mathcal{S}) \;\subseteq\; \mathrm{Inf}_{\mathcal{M}}(Q, I_2) \;\subseteq\; Q(I_1).$$

Thus, given that $\bar{t} \in \mathrm{Inf}_{\mathcal{M}}(Q, \mathcal{S})$, we conclude that $\bar{t} \in Q(I_1)$. We have shown that for every $I_1 \in \mathrm{Sol}_{\mathcal{M} \circ \mathcal{M}^{\mathcal{C}}}(I)$, it holds that $\bar{t} \in Q(I_1)$. Therefore, we have that $\bar{t} \in \mathrm{certain}_{\mathcal{M} \circ \mathcal{M}^{\mathcal{C}}}(Q, I)$. But we know that $\bar{t} \notin \mathrm{Inf}_{\mathcal{M}}(Q, I)$, from which one concludes that:

$$\mathrm{certain}_{\mathcal{M} \circ \mathcal{M}^{\mathcal{C}}}(Q, I) \;\not\subseteq\; \mathrm{Inf}_{\mathcal{M}}(Q, I).$$

Hence, we conclude by Lemma 4.1.4 that $\mathcal{M}^{\mathcal{C}}$ is not a $\mathcal{C}$-recovery of $\mathcal{M}$, which contradicts out initial assumption. This concludes the proof of the proposition. □

We are now ready to prove that if one includes disjunctions, inequalities or predicate $\mathbf{C}$ in the conclusions of the dependencies, then mappings do not necessarily admit CQ-maximum recoveries

PROPOSITION 4.3.10. *There exist st-mappings specified by sets of (1)* CQ-TO-UCQ *dependencies, (2)* CQ-TO-$\text{CQ}^{\neq}$ *dependencies, (3)* CQ-TO-$\text{CQ}^{\mathbf{C}}$ *dependencies, that have no* CQ*-maximum recoveries.*

PROOF. In order to prove (1), let $\mathbf{S} = \{A(\cdot), B(\cdot), C_1(\cdot), C_2(\cdot)\}$, $\mathbf{T} = \{R_1(\cdot), R_2(\cdot)\}$, and $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ an st-mapping specified by the following set Σ of CQ-TO-UCQ st-dependencies:

$$\begin{aligned} B(x) \wedge C_1(x) &\rightarrow R_1(x), \\ B(x) \wedge C_2(x) &\rightarrow R_2(x), \\ A(x) &\rightarrow R_1(x) \vee R_2(x). \end{aligned}$$

Let $I_1 = \{B(a), C_1(a)\}$, $I_2 = \{B(a), C_2(a)\}$, and $I_3 = \{A(a)\}$ be instances of $\mathbf{S}$, where a is an arbitrary element of $\mathbf{C}$. By the definition of Σ, it is clear that $\text{Sol}_{\mathcal{M}}(I_3) \subseteq \text{Sol}_{\mathcal{M}}(I_1) \cup \text{Sol}_{\mathcal{M}}(I_2)$. Furthermore, we claim that for each instance $I \in \{I_1, I_2\}$, if $I' \in \text{Sub}_{\mathcal{M}}(I)$, then it holds that $I \subseteq I'$. Without loss of generality, we prove this claim for the instance I_1. By contradiction, assume that there exists an instance $I' \in \text{Sub}_{\mathcal{M}}(I_1)$ such that $I_1 \not\subseteq I'$. It is easy to see that for every $J \in \text{Sol}_{\mathcal{M}}(I_1)$, it holds that $a \in R_1^J$. Thus, given that $I' \in \text{Sub}_{\mathcal{M}}(I_1)$, we have that $a \in R_1^J$ for every $J \in \text{Sol}_{\mathcal{M}}(I')$. Notice that the space of solutions for a source instance under $\mathcal{M}$ is always nonempty, and assume that $J \in \text{Sol}_{\mathcal{M}}(I')$. Then define an instance J' of $\mathbf{T}$ as $R_1^{J'} = R_1^J \setminus \{a\}$ and $R_2^{J'} = R_2^J \cup \{a\}$. By the definition of J' and given that $I_1 \not\subseteq I'$, we have that $(I', J') \models \Sigma$. Thus, we have shown that $J' \in \text{Sol}_{\mathcal{M}}(I')$. But this leads to a contradiction since for every $J \in \text{Sol}_{\mathcal{M}}(I')$, it holds that $a \in R_1^J$. We conclude that $I_1 \subseteq I'$ for every instance $I' \in \text{Sub}_{\mathcal{M}}(I_1)$, and we have proved our claim. Now, let Q be conjunctive query $B(x)$. Then we have that $\text{Inf}_{\mathcal{M}}(Q, I_1) = \text{Inf}_{\mathcal{M}}(Q, I_2) = \{a\}$, but $\text{Inf}_{\mathcal{M}}(Q, I_3) = \emptyset$ since $B^{I_3} = \emptyset$, which implies

that:

$$\mathrm{Inf}_{\mathcal{M}}(Q, \{I_1, I_2\}) \quad \not\subseteq \quad \mathrm{Inf}_{\mathcal{M}}(Q, I_3).$$

Thus, we conclude from Proposition 4.3.9 that $\mathcal{M}$ does not have a CQ-maximum recovery since $\mathrm{Sol}_{\mathcal{M}}(I_3) \subseteq \mathrm{Sol}_{\mathcal{M}}(I_1) \cup \mathrm{Sol}_{\mathcal{M}}(I_2)$.

Now to prove (2), let $\mathbf{S} = \{A(\cdot), B(\cdot), C_1(\cdot), C_2(\cdot)\}$, $\mathbf{T} = \{R(\cdot, \cdot)\}$, and $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ an st-mapping specified by the following set Σ of CQ-TO-CQ$^{\neq}$ st-dependencies:

$$\begin{aligned} B(x) \wedge C_1(x) &\rightarrow R(x, x), \\ B(x) \wedge C_2(x) &\rightarrow \exists y(R(x, y) \wedge x \neq y), \\ A(x) &\rightarrow \exists y R(x, y). \end{aligned}$$

Let $I_1 = \{B(a), C_1(a)\}$, $I_2 = \{B(a), C_2(a)\}$, and $I_3 = \{A(a)\}$ be instances of $\mathbf{S}$, where a is an arbitrary element of $\mathbf{C}$. By the definition of Σ, it is clear that $\mathrm{Sol}_{\mathcal{M}}(I_3) \subseteq \mathrm{Sol}_{\mathcal{M}}(I_1) \cup \mathrm{Sol}_{\mathcal{M}}(I_2)$. Furthermore, we claim that for each instance $I \in \{I_1, I_2\}$, if $I' \in \mathrm{Sub}_{\mathcal{M}}(I)$, then it holds that $I \subseteq I'$. First, we prove this claim for the instance I_1. By contradiction, assume that there exists an instance $I' \in \mathrm{Sub}_{\mathcal{M}}(I_1)$ such that $I_1 \not\subseteq I'$. It is easy to see that for every $J \in \mathrm{Sol}_{\mathcal{M}}(I_1)$, it holds that $(a, a) \in R^J$. Thus, given that $I' \in \mathrm{Sub}_{\mathcal{M}}(I_1)$, we have that $(a, a) \in R^J$ for every $J \in \mathrm{Sol}_{\mathcal{M}}(I')$. Notice that the space of solutions for a source instance under $\mathcal{M}$ is always nonempty, and assume that $J \in \mathrm{Sol}_{\mathcal{M}}(I')$. Then define an instance J' as $R^{J'} = R^J \cup \{(a, b)\} \setminus \{(a, a)\}$, where b is an arbitrary element of $\mathbf{D}$ such that $a \neq b$. By the definition of J' and given that $I_1 \not\subseteq I'$, we have that $(I', J') \models \Sigma$. Thus, we have shown that $J' \in \mathrm{Sol}_{\mathcal{M}}(I')$. But this leads to a contradiction since for every $J \in \mathrm{Sol}_{\mathcal{M}}(I')$, it holds that $(a, a) \in R^J$. We conclude that $I_1 \subseteq I'$ for every instance $I' \in \mathrm{Sub}_{\mathcal{M}}(I_1)$.

Second, we prove the claim for the instance I_2, that is, we prove that $I_2 \subseteq I'$ for every $I' \in \mathrm{Sub}_{\mathcal{M}}(I_2)$. By contradiction, assume that there exists an instance $I' \in \mathrm{Sub}_{\mathcal{M}}(I_2)$ such that $I_2 \not\subseteq I'$. It is easy to see that for every $J \in \mathrm{Sol}_{\mathcal{M}}(I_2)$, it holds that $(a, b) \in R^J$ for some $b \neq a$. Thus, given that $I' \in \mathrm{Sub}_{\mathcal{M}}(I_2)$, we have that for every $J \in \mathrm{Sol}_{\mathcal{M}}(I')$,

$(a, b) \in R^J$ for some $b \neq a$. Notice that the space of solutions for a source instance under $\mathcal{M}$ is always nonempty, and assume that $J \in \mathrm{Sol}_{\mathcal{M}}(I')$. Then define an instance J' as $R^{J'} = (R^J \cup \{(a, a)\}) \setminus \{(a, b) \mid b \in \mathbf{D} \text{ and } b \neq a\}$. By the definition of J' and given that $I_2 \not\subseteq I'$, we have that $(I', J') \models \Sigma$. Thus, we have shown that $J' \in \mathrm{Sol}_{\mathcal{M}}(I')$. But this leads to a contradiction since for every $J \in \mathrm{Sol}_{\mathcal{M}}(I')$, it holds that $(a, b) \in R^J$ for some $b \neq a$. We conclude that $I_2 \subseteq I'$ for every instance $I' \in \mathrm{Sub}_{\mathcal{M}}(I_2)$, and we have proved our claim. Now, let Q be conjunctive query $B(x)$. Then we have that $\mathrm{Inf}_{\mathcal{M}}(Q, I_1) = \mathrm{Inf}_{\mathcal{M}}(Q, I_2) = \{a\}$, but $\mathrm{Inf}_{\mathcal{M}}(Q, I_3) = \emptyset$ since $B^{I_3} = \emptyset$, which implies that:

$$\mathrm{Inf}_{\mathcal{M}}(Q, \{I_1, I_2\}) \quad \not\subseteq \quad \mathrm{Inf}_{\mathcal{M}}(Q, I_3).$$

Therefore, we conclude from Proposition 4.3.9 that $\mathcal{M}$ does not have a CQ-maximum recovery since $\mathrm{Sol}_{\mathcal{M}}(I_3) \subseteq \mathrm{Sol}_{\mathcal{M}}(I_1) \cup \mathrm{Sol}_{\mathcal{M}}(I_2)$.

Finally, for case (3), let $\mathbf{S} = \{A(\cdot), B(\cdot)\}$, $\mathbf{T} = \{R(\cdot)\}$, and $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ an st-mapping specified by the following set Σ of CQ-TO-CQ$^{\mathbf{C}}$ st-dependencies:

$$\begin{aligned} A(x) &\rightarrow R(x), \\ B(x) &\rightarrow \exists y\,(R(y) \wedge \mathbf{C}(y)). \end{aligned}$$

Let $I = \{B(a)\}$ be an instance of $\mathbf{S}$, where a is an arbitrary element of $\mathbf{C}$, and

$$\mathcal{S} \quad = \quad \{I' \in \mathrm{Inst}(\mathbf{S}) \mid \text{there exists } b \in \mathbf{C} \text{ such that } A^{I'} = \{b\} \text{ and } B^{I'} = \emptyset\}.$$

By the definition of Σ, it is clear that $\mathrm{Sol}_{\mathcal{M}}(I) \subseteq \bigcup_{I' \in \mathcal{S}} \mathrm{Sol}_{\mathcal{M}}(I')$. Furthermore, we claim that if $I^\star \in \mathrm{Sub}_{\mathcal{M}}(\mathcal{S})$, then it holds that $A^{I^\star} \neq \emptyset$. Assume, for the sake of contradiction, that there exist instances $I', I^\star$ of $\mathbf{S}$ such that $I' \in \mathcal{S}$, $I^\star \in \mathrm{Sub}_{\mathcal{M}}(I')$ and $A^{I^\star} = \emptyset$. By the definition of $\mathcal{S}$, we have that there exists a constant b such that $A^{I'} = \{b\}$. Assume that J is an instance of $\mathbf{T}$ such that $R^J = \{c\}$, where c is a constant such that $c \neq b$. Then we have that $J \in \mathrm{Sol}_{\mathcal{M}}(I^\star)$ since $A^{I^\star} = \emptyset$, and that $J \notin \mathrm{Sol}_{\mathcal{M}}(I')$ since $b \notin R^J$. Thus, we obtain a contradiction with the fact that $I^\star \in \mathrm{Sub}_{\mathcal{M}}(I')$.

Now, let Q be conjunctive query $\exists x A(x)$. Then given that for every $I' \in \mathrm{Sub}_{\mathcal{M}}(\mathcal{S})$, it holds that $A^{I'} \neq \emptyset$, we conclude that $\mathrm{Inf}_{\mathcal{M}}(Q, \mathcal{S}) = \underline{\text{true}}$. Thus, given that $\mathrm{Inf}_{\mathcal{M}}(Q, I) = \emptyset$ since $A^I = \emptyset$ and $I \in \mathrm{Sub}_{\mathcal{M}}(I)$, we have that:

$$\mathrm{Inf}_{\mathcal{M}}(Q, \mathcal{S}) \quad \not\subseteq \quad \mathrm{Inf}_{\mathcal{M}}(Q, I).$$

Thus, we conclude from Proposition 4.3.9 that $\mathcal{M}$ does not have a CQ-maximum recovery since $\mathrm{Sol}_{\mathcal{M}}(I) \subseteq \bigcup_{I' \in \mathcal{S}} \mathrm{Sol}_{\mathcal{M}}(I')$. This concludes the proof of the proposition. □

4.3.3. CQ-maximum recovery is the right notion

In this section, we consider several alternatives to CQ for the semantics of inverse operators, and show that none of them is appropriate to obtain a closure property as in Theorem 4.3.1. We start with the notion of UCQ-maximum recovery. The following result shows that to express the UCQ-maximum recovery of a mapping given by a set of st-tgds, one needs dependencies with disjunctions in their conclusions (even if the full power of FO is allowed in the premises of the dependencies).

PROPOSITION 4.3.11. *There exists an st-mapping $\mathcal{M}$ specified by a set of st-tgds such that:*

(a) *$\mathcal{M}$ has a* UCQ*-maximum recovery specified by a set of* CQ-TO-UCQ *dependencies.*

(b) *$\mathcal{M}$ does not have a* UCQ*-maximum recovery specified by a set of* FO^{C}-TO-CQ *dependencies.*

PROOF. Let $\mathbf{S} = \{A(\cdot), B(\cdot)\}$, $\mathbf{T} = \{T(\cdot)\}$ and $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where Σ is the following set of st-tgds:

$$\Sigma \quad = \quad \{A(x) \rightarrow T(x),\ B(x) \rightarrow T(x)\}.$$

We first prove (a). Let $\mathcal{M}^\star = (\mathbf{T}, \mathbf{S}, \Sigma^\star)$ be a ts-mapping specified by the following set of CQ-TO-UCQ dependencies:

$$\Sigma^\star \;=\; \{T(x) \rightarrow A(x) \vee B(x)\}.$$

By using the tools in Section 3.1.1, it is easy to show that $\mathcal{M}^\star$ is a maximum recovery of $\mathcal{M}$, which implies that $\mathcal{M}^\star$ is a UCQ-maximum recovery of $\mathcal{M}$.

We now prove (b). For the sake of contradiction, assume that there exists a mapping $\mathcal{M}'$ specified by a set Σ' of FO$^{\mathbf{C}}$-TO-CQ dependencies such that $\mathcal{M}'$ is a UCQ-maximum recovery of $\mathcal{M}$. Moreover, let I_1 and I_2 be instances of $\mathbf{S}$ such that:

$$\begin{array}{lcllcl} A^{I_1} & = & \{a\} & \qquad A^{I_2} & = & \emptyset \\ B^{I_1} & = & \emptyset & \qquad B^{I_2} & = & \{a\} \end{array}$$

Let J be the instance such that $T^J = \{a\}$. Notice that J is the canonical universal solution for both I_1 and I_2 under Σ. Assume that K is the result of chasing J with Σ'. Notice that K could not be a valid source instance (it may contain null values). Nevertheless, since J is composed only by constant values, and by using the properties of the chase (Fagin, Kolaitis, Miller, & Popa, 2005; Fagin, Kolaitis, Popa, & Tan, 2005, 2008), we know that for every Q that is a union of conjunctive queries, it holds that $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I_1) = \mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I_2) = Q(K)_{\downarrow}$. We use this last fact to derive a contradiction.

We first show that, if we assume that K is not the empty instance we obtain a contradiction. Then assume that K is not empty. If $A^K \neq \emptyset$, then let Q_A be Boolean query $\exists x A(x)$. Thus, given that $A^K \neq \emptyset$, we conclude that $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q_A, I_2) = \underline{\text{true}}$. But this leads to a contradiction since we assume that $\mathcal{M}'$ is a UCQ-recovery of $\mathcal{M}$ and $Q_A(I_2) = \underline{\text{false}}$. If $B^K \neq \emptyset$, then we obtain a similar contradiction by considering Boolean query $Q_B = \exists y B(y)$ and source instance I_1.

Assume now that K is the empty instance. Let Q be the Boolean query $\exists x A(x) \vee \exists y B(y)$. Then we have that $Q(K) = \underline{\text{false}}$, which implies that $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(Q, I_1) = \underline{\text{false}}$. But the mapping $\mathcal{M}^\star$ defined in (a) is a UCQ-recovery of $\mathcal{M}$ and $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}^\star}(Q, I_1) = \underline{\text{true}} = Q(I_1)$, which implies that $\mathcal{M}^\star \not\preceq^{\text{UCQ}}_{\mathcal{M}} \mathcal{M}'$. This contradicts our assumption that

$\mathcal{M}'$ is a UCQ-maximum recovery of $\mathcal{M}$. In either case we obtain a contradiction. This concludes the proof of the proposition. □

Proposition 4.3.10 shows that there exist mappings specified by CQ-TO-UCQ dependencies that have no CQ-maximum recoveries and, thus, have no UCQ-maximum recoveries. Thus, this proposition together with Proposition 4.3.11 show that if we use UCQ-maximum recovery as our notion of inverse, then we are doomed to failure.

Now we consider the notion of $\text{CQ}^{\neq}$-maximum recovery. By Theorem 4.3.8, we have that inequalities in the premises of dependencies are needed to express CQ-maximum recoveries of mappings given by st-tgds. Thus, if a mapping language contains the class of st-tgds and is closed under $\text{CQ}^{\neq}$-maximum recovery, then it has to include inequalities in the premises of dependencies. Our next result shows that in order to express the $\text{CQ}^{\neq}$-maximum recovery of a mapping given by a set of $\text{CQ}^{\neq}$-TO-CQ dependencies, one needs dependencies with inequalities in their conclusions (even if the full power of FO is allowed in the premises of the dependencies).

PROPOSITION 4.3.12. *There exists an st-mapping $\mathcal{M}$ specified by a set of* $\text{CQ}^{\neq}$-TO-CQ *dependencies such that:*

(a) *$\mathcal{M}$ has a* $\text{CQ}^{\neq}$*-maximum recovery specified by a set of* CQ-TO-$\text{CQ}^{\neq}$ *dependencies.*

(b) *$\mathcal{M}$ does not have a* $\text{CQ}^{\neq}$*-maximum recovery specified by a set of* $\text{FO}^{\mathbf{C}}$-TO-CQ *dependencies.*

PROOF. Let $\mathbf{S} = \{P(\cdot, \cdot)\}$, $\mathbf{T} = \{T(\cdot)\}$ and $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where Σ is the following set of $\text{CQ}^{\neq}$-TO-CQ dependencies:

$$\Sigma \quad = \quad \{P(x, y) \wedge x \neq y \rightarrow T(x)\}.$$

In order to prove part (a), let $\mathcal{M}^\star = (\mathbf{T}, \mathbf{S}, \Sigma^\star)$ be a ts-mapping specified by the following set of CQ-TO-CQ$^{\neq}$ dependencies:

$$\Sigma^\star \;=\; \{T(x) \rightarrow \exists y\,(P(x,y) \wedge x \neq y)\}.$$

By using the tools in Section 3.1.1 it is easy to prove that $\mathcal{M}^\star$ is a maximum recovery of $\mathcal{M}$, which implies that $\mathcal{M}^\star$ is a CQ$^{\neq}$-maximum recovery of $\mathcal{M}$.

We now prove (b). For the sake of contradiction, assume that there exists a mapping $\mathcal{M}'$ specified by a set of FO$^{\mathbf{C}}$-TO-CQ dependencies such that $\mathcal{M}'$ is a CQ$^{\neq}$-maximum recovery of $\mathcal{M}$. Moreover, let I be an instance of $\mathbf{S}$ such that $P^I = \{(a,b)\}$, where $a \neq b$, and J a target instance such that $T^J = \{a\}$. Then we have that $(I,J) \models \Sigma$, and then $(I,J) \in \mathcal{M}$. Let K be the chase of J with Σ'. Thus, given that $\mathrm{dom}(J) = \{a\}$, we have that

$$\mathrm{dom}(K) \cap \mathbf{C} \;\subseteq\; \{a\}. \tag{4.3}$$

Then since J is composed only by constant values and Σ' is a set of FO$^{\mathbf{C}}$-TO-CQ, we have that every instance K' that is a homomorphic image of K, is such that $(J,K') \models \Sigma'$. Consider the homomorphism h that maps every null in K into a. Notice that $h(K)$ is composed only by constant values and that $(J,h(K)) \models \Sigma'$, then $(J,h(K)) \in \mathcal{M}'$. Thus, we have that $(I,h(K)) \in \mathcal{M} \circ \mathcal{M}'$. From (4.3), we have that $P^{h(K)} \subseteq \{(a,a)\}$ and, hence, $h(K) \not\models \exists x \exists y\,(P(x,y) \wedge x \neq y)$. We conclude that:

$$\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}'}(\exists x \exists y\,(P(x,y) \wedge x \neq y), I) \;=\; \underline{\text{false}}.$$

It is straightforward to prove that $\mathrm{certain}_{\mathcal{M}\circ\mathcal{M}^\star}(\exists x \exists y\,(P(x,y) \wedge x \neq y), I) = \underline{\text{true}}$, where $\mathcal{M}^\star$ is the ts-mapping defined in (a). Therefore, we have that $\mathcal{M}^\star \not\preceq_{\mathcal{M}}^{\mathrm{CQ}^{\neq}} \mathcal{M}'$, which contradicts the fact that $\mathcal{M}'$ is a CQ$^{\neq}$-maximum recovery of $\mathcal{M}$ since $\mathcal{M}^\star$ is a CQ$^{\neq}$-recovery of $\mathcal{M}$. This concludes the proof of the proposition. □

Proposition 4.3.10 shows that there exist mappings specified by CQ-TO-CQ$^{\neq}$ dependencies that have no CQ-maximum recoveries and, thus, have no CQ$^{\neq}$-maximum recoveries. Thus, this proposition together with Proposition 4.3.12 shows that, if we use CQ$^{\neq}$-maximum recovery as our notion of inverse, then we cannot hope for a closure result as in Theorem 4.3.1.

We conclude this section by pointing out that the negative results for UCQ- and CQ$^{\neq}$-maximum recoveries imply a negative result for the notion of $\mathcal{C}$-maximum recovery, for every class $\mathcal{C}$ of queries containing UCQ or CQ$^{\neq}$.

5. ON INVERTING AND COMPOSING SCHEMA MAPPINGS

In this chapter, we explore the relationship between good mapping languages for inversion and composition together. Fagin, Kolaitis, Popa, and Tan (2005) show that an extension of st-tgds with second-order existential quantification, the language of SO-tgds (Fagin, Kolaitis, Popa, & Tan, 2005), is closed under composition, and prove several results that show that the language of SO-tgds is the right language to compose mappings. However, none of the notions of inverse proposed for schema mappings (Fagin, 2007; Fagin, Kolaitis, Popa, & Tan, 2008) have been considered for the case of SO-tgds. In this chapter we show that, unfortunately, SO-tgds are not appropriate for our study; we show that there exist mappings specified by SO-tgds that have no CQ-maximum recoveries (and, thus, have no Fagin-inverse, quasi-inverse and maximum recovery). Thus, although the language of SO-tgds is the right language for composition, it has a bad behavior regarding inverting mappings.

To overcome this limitation, we borrow the notion of composition w.r.t. conjunctive queries (CQ-composition), introduced by Madhavan and Halevy (2003). Then we propose a language called *plain SO-tgds* such that: the language of plain SO-tgds is closed under CQ-composition, and every mapping given by plain SO-tgds has a maximum recovery. To prove this last property we provide a polynomial-time algorithm that given a mapping $\mathcal{M}$ specified by a set of plain SO-tgds, returns a maximum recovery of $\mathcal{M}$ specified in a language that extends the class of plain SO-tgds. This result is interesting in its own since our algorithm is the first polynomial-time algorithm to compute inverses of schema mappings.

Before giving the technical results of this chapter, we need to make an observation about the composition of st-tgds. Assume that $\mathbf{R}_1$, $\mathbf{R}_2$ and $\mathbf{R}_3$ are relational schemas with no symbols in common, and let $\mathcal{M}_{12} = (\mathbf{R}_1, \mathbf{R}_2, \Sigma_{12})$ and $\mathcal{M}_{23} = (\mathbf{R}_2, \mathbf{R}_3, \Sigma_{23})$, where Σ_{12} and Σ_{23} are set of st-tgds. As we mentioned in Section 2.1, the composition of $\mathcal{M}_{12}$ and $\mathcal{M}_{23}$ is simply defined as the mapping that contains all the pairs $(I_1, I_3) \in \mathrm{Inst}(\mathbf{R}_1) \times \mathrm{Inst}(\mathbf{R}_3)$, for which there exists an instance I_2 of $\mathbf{R}_2$ such that $(I_1, I_2) \in \mathcal{M}_{12}$

and $(I_2, I_3) \in \mathcal{M}_{23}$. But notice that there is a mismatch in the way that schema $\mathbf{R}_2$ is used in this composition; $\mathbf{R}_2$ is a target schema in $\mathcal{M}_{12}$ and a source schema in $\mathcal{M}_{23}$. In particular, instance I_2 above may include null values if $\mathbf{R}_2$ is used as a target schema, but it can only contain constants if $\mathbf{R}_2$ is considered to be a source schema. To overcome this issue, in the composition of $\mathcal{M}_{12}$ and $\mathcal{M}_{23}$, we allow instances of $\mathbf{R}_2$ to contain null values and, thus, we always use $\mathbf{R}_2$ as a target schema.

5.1. The Language of Plain SO-tgds

Our language is based on the notion of second-order tgds proposed by Fagin, Kolaitis, Popa, and Tan (2005), so we start by introducing this language. Given schemas $\mathbf{R}_1$ and $\mathbf{R}_2$ with no relation symbols in common, a *second-order tuple-generating dependency from* $\mathbf{R}_1$ *to* $\mathbf{R}_2$ (SO-tgd) is a formula of the form

$$\exists \bar{f} \, (\forall \bar{x}_1(\varphi_1 \rightarrow \psi_1) \wedge \cdots \wedge \forall \bar{x}_n(\varphi_n \rightarrow \psi_n)), \tag{5.1}$$

where

(1) each member of $\bar{f}$ is a function symbol,

(2) each formula φ_i $(1 \leq i \leq n)$ is a conjunction of relational atoms of the form $S(y_1, \ldots, y_k)$ and equality atoms of the form $t = t'$, where S is a k-ary relation symbol of $\mathbf{R}_1$ and y_1, ..., y_k are (not necessarily distinct) variables in $\bar{x}_i$, and t, t' are terms built from $\bar{x}_i$ and $\bar{f}$,

(3) each formula ψ_i $(1 \leq i \leq n)$ is a conjunction of relational atomic formulas over $\mathbf{R}_2$ mentioning terms built from $\bar{x}_i$ and $\bar{f}$, and

(4) each variable in $\bar{x}_i$ $(1 \leq i \leq n)$ appears in some relational atom of φ_i.

The semantics of SO-tgds is defined as follows. Let λ be an SO-tgd of the form (5.1) from $\mathbf{R}_1$ to $\mathbf{R}_2$, and assume that I is an instance of $\mathbf{R}_1$ and J and instance of $\mathbf{R}_2$. An interpretation of a k-ary function f in (I, J) is a function that maps every tuple $(a_1, \ldots, a_k)$ of elements in I to an element in J. Then a pair (I, J) of instances satisfies λ if there exists an interpretation in (I, J) of each function in $\bar{f}$ such that (I, J) satisfies each dependency

$\forall \bar{x}_i(\varphi_i \rightarrow \psi_i)$ with this interpretation. That is, the semantics of SO-tgds is inherited from the usual semantics of second order logic (Libkin, 2004).

Notice that SO-tgds are closed under conjunction. That is, if σ_1 and σ_2 are SO-tgds, then $\sigma_1 \wedge \sigma_2$ is logically equivalent to an SO-tgd. Thus, when specifying mappings, we talk about a mapping specified by an SO-tgd (instead of a set of SO-tgds).

Fagin, Kolaitis, Popa, and Tan (2005) show that the language of SO-tgds is the right language for expressing the composition of mappings given by st-tgds. First, it is not difficult to see that every set of st-tgds can be transformed into an SO-tgd by *Skolemizing* the existentially quantified variables in the conclusions of the st-tgds. For example, the set given by the st-tgds:

$$\begin{aligned} A(x,y) &\rightarrow \exists z(R(x,z) \wedge T(z,y)) \\ B(x,y) &\rightarrow R(x,y) \end{aligned}$$

is logically equivalent to the following SO-tgd:

$$\exists f \Big(\forall x \forall y \left(A(x,y) \rightarrow R(x, f(x,y))\right) \wedge \forall x \forall y \left(A(x,y) \rightarrow T(f(x,y),y)\right) \wedge \forall x \forall y \left(B(x,y) \rightarrow R(x,y)\right) \Big). \quad (5.2)$$

Second, Fagin, Kolaitis, Popa, and Tan (2005) show that SO-tgds are closed under composition.

THEOREM 5.1.1 (Fagin, Kolaitis, Popa, & Tan, 2005). *Let $\mathcal{M}_{12}$ and $\mathcal{M}_{23}$ be mappings specified by SO-tgds. Then the composition $\mathcal{M}_{12} \circ \mathcal{M}_{23}$ can also be specified by an SO-tgd.*

It is important to notice that Theorem 5.1.1 implies that the composition of a finite number of mappings specified by st-tgds can be defined by an SO-tgd, as every set of st-tgds can be expressed as an SO-tgd.

THEOREM 5.1.2 (Fagin, Kolaitis, Popa, & Tan, 2005). *The composition of a finite number of mappings, each defined by a set of st-tgds, is defined by an SO-tgd.*

Unfortunately, as we show in the next result, SO-tgds are not appropriate for inversion, as there exist mappings specified by SO-tgds that do not admit CQ-maximum recoveries (and, thus, do not admit Fagin-inverses, quasi-inverses and maximum recoveries).

PROPOSITION 5.1.3. *There exists an st-mapping $\mathcal{M}$ specified by an SO-tgd that has no* CQ*-maximum recovery.*

PROOF. Let $\mathbf{S} = \{A(\cdot), B(\cdot), C_1(\cdot), C_2(\cdot)\}$, $\mathbf{T} = \{R(\cdot,\cdot), S(\cdot)\}$, and $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ an st-mapping specified by the following SO-tgd:

$$\exists f \exists g \big[\forall x\,(B(x) \wedge C_1(x) \rightarrow R(x, f(x))) \wedge \forall x\,(B(x) \wedge C_1(x) \wedge x = f(x) \rightarrow S(x)) \wedge \forall x\,(B(x) \wedge C_2(x) \rightarrow R(x,x)) \wedge \forall x\,(A(x) \rightarrow R(x, g(x)))\big].$$

Let $I_1 = \{B(a), C_1(a)\}$, $I_2 = \{B(a), C_2(a)\}$, and $I_3 = \{A(a)\}$ be instances of $\mathbf{S}$, where a is an arbitrary element of $\mathbf{C}$. By definition of Σ, it holds that $\mathrm{Sol}_{\mathcal{M}}(I_3) \subseteq \mathrm{Sol}_{\mathcal{M}}(I_1) \cup \mathrm{Sol}_{\mathcal{M}}(I_2)$. Furthermore, we claim that for each instance $I \in \{I_1, I_2\}$, if $I' \in \mathrm{Sub}_{\mathcal{M}}(I)$, then it holds that $I \subseteq I'$. First, we prove this claim for the instance I_1. By contradiction, assume that there exists an instance $I' \in \mathrm{Sub}_{\mathcal{M}}(I_1)$ such that $I_1 \not\subseteq I'$. It is easy to see that for every $J \in \mathrm{Sol}_{\mathcal{M}}(I_1)$, it holds that $a \in S^J$ whenever (a,a) is the only tuple in R^J with a as its first component. Thus, given that $I' \in \mathrm{Sub}_{\mathcal{M}}(I_1)$, we have that for every $J \in \mathrm{Sol}_{\mathcal{M}}(I')$, $a \in S^J$ whenever (a,a) is the only tuple in R^J with a as its first component. Notice that the space of solutions for a source instance under $\mathcal{M}$ is always nonempty, and assume that $J \in \mathrm{Sol}_{\mathcal{M}}(I')$. Then define an instance J' of $\mathbf{T}$ as follows:

$$\begin{aligned} R^{J'} &= (R^J \cup \{(a,a)\}) \setminus \{(a,b) \mid b \in \mathbf{D}, \text{ and } a \neq b\}, \\ S^{J'} &= S^J \setminus \{a\}. \end{aligned}$$

By definition of J' and given that $I_1 \not\subseteq I'$, we have that $(I', J') \models \Sigma$. Thus, we have shown that $J' \in \mathrm{Sol}_{\mathcal{M}}(I')$. But this leads to a contradiction since for every $J \in \mathrm{Sol}_{\mathcal{M}}(I')$, it holds that if (a,a) is the only tuple in R^J with a as its first component, then $a \in S^J$. We conclude that $I_1 \subseteq I'$ for every instance $I' \in \mathrm{Sub}_{\mathcal{M}}(I_1)$. Second, we prove the claim for the instance

I_2, that is, we prove that for every $I' \in \mathrm{Sol}_{\mathcal{M}}(I_2)$, it holds that $I_2 \subseteq I'$. By contradiction, assume that there exists an instance $I' \in \mathrm{Sub}_{\mathcal{M}}(I_2)$ such that $I_2 \not\subseteq I'$. It is easy to see that for every $J \in \mathrm{Sol}_{\mathcal{M}}(I_2)$, it holds that $(a, a) \in R^J$. Thus, given that $I' \in \mathrm{Sub}_{\mathcal{M}}(I_2)$, we have that $(a, a) \in R^J$ for every $J \in \mathrm{Sol}_{\mathcal{M}}(I')$. Notice that the space of solutions for a source instance under $\mathcal{M}$ is always nonempty, and assume that $J \in \mathrm{Sol}_{\mathcal{M}}(I')$. Then let b be an arbitrary element of $\mathbf{C}$ such that $a \neq b$, and define an instance J' of $\mathbf{T}$ as follows:

$$\begin{aligned} R^{J'} &= (R^J \setminus \{(a, a)\}) \cup \{(a, b)\}, \\ S^{J'} &= S^J. \end{aligned}$$

By definition of J' and given that $I_2 \not\subseteq I'$, we have that $(I', J') \models \Sigma$. Thus, we have shown that $J' \in \mathrm{Sol}_{\mathcal{M}}(I')$. But this leads to a contradiction since for every $J \in \mathrm{Sol}_{\mathcal{M}}(I')$, it holds that $(a, a) \in R^J$. We conclude that $I_2 \subseteq I'$ for every instance $I' \in \mathrm{Sub}_{\mathcal{M}}(I_2)$, and we have proved our claim. Now, let Q be conjunctive query $B(x)$. Then we have that $\mathrm{Inf}_{\mathcal{M}}(Q, I_1) = \mathrm{Inf}_{\mathcal{M}}(Q, I_2) = \{a\}$, but $\mathrm{Inf}_{\mathcal{M}}(Q, I_3) = \emptyset$ since $B^{I_3} = \emptyset$, which implies that:

$$\mathrm{Inf}_{\mathcal{M}}(Q, \{I_1, I_2\}) \not\subseteq \mathrm{Inf}_{\mathcal{M}}(Q, I_3).$$

Hence, we conclude by Proposition 4.3.9 that $\mathcal{M}$ does not have a CQ-maximum recovery since $\mathrm{Sol}_{\mathcal{M}}(I_3) \subseteq \mathrm{Sol}_{\mathcal{M}}(I_1) \cup \mathrm{Sol}_{\mathcal{M}}(I_2)$. This concludes the proof of the proposition. □

Thus, by the previous proposition and the results of Section 4.2, we obtain that mappings specified by SO-tgds are not guaranteed to have maximum recoveries. Also notice that mappings specified by SO-tgds are total and closed-down on the left, thus, by Theorem 3.1.10 we have that SO-tgds are not guaranteed to have Fagin-inverses nor quasi-inverse.

By Proposition 5.1.3 (and the aforementioned results about the composition), in order to obtain a language that is closed under composition and has good properties regarding inversion, we have no choice but to relax the semantics of composition. More precisely, we borrow the notion of composition w.r.t. conjunctive queries (CQ-composition) introduced

by Madhavan and Halevy (2003), and we say that $\mathcal{M}_{13}$ is the CQ-composition of mappings $\mathcal{M}_{12}$ and $\mathcal{M}_{23}$ if $\mathcal{M}_{13} \equiv_{\text{CQ}} (\mathcal{M}_{12} \circ \mathcal{M}_{23})$. (Recall that $\mathcal{M}_{13} \equiv_{\text{CQ}} (\mathcal{M}_{12} \circ \mathcal{M}_{23})$ if and only if for every instance I and every query Q in CQ, it holds that $\text{certain}_{\mathcal{M}_{12}}(Q, I) = \text{certain}_{\mathcal{M}_{12} \circ \mathcal{M}_{23}}(Q, I)$.)

Using this notion, we show that there is a mapping language that is not only closed under CQ-composition, but also admits a maximum recovery for every st-mapping specified in it. We next introduce this mapping language. In the definition, we use the notion of plain term. Given a tuple of function symbols $\bar{f}$ and a tuple of variables $\bar{x}$, a *plain term* constructed over $\bar{f}$ and $\bar{x}$, is either a variable x in $\bar{x}$, or a term of the form $f(u_1, u_2, \ldots, u_k)$ where f is a function symbol in $\bar{f}$ and u_i is variables in $\bar{x}$ (for $1 \leq i \leq k$).

DEFINITION 5.1.4. *Given schemas $\mathbf{R}_1$ and $\mathbf{R}_2$ with no relation symbols in common, a* plain second-order tuple-generating dependency (plain SO-tgd) from $\mathbf{R}_1$ to $\mathbf{R}_2$ *is a formula of the form*

$$\exists \bar{f} \left(\forall \bar{x}_1 (\varphi_1 \rightarrow \psi_1) \wedge \cdots \wedge \forall \bar{x}_n (\varphi_n \rightarrow \psi_n) \right)$$

such that

(1) *each member of $\bar{f}$ is a function symbol,*

(2) *each formula φ_i ($1 \leq i \leq n$) is a conjunction of relational atoms of the form $S(y_1, \ldots, y_k)$ where S is a k-ary relation symbol of $\mathbf{R}_1$ and y_1, ..., y_k are (not necessarily distinct) variables in $\bar{x}_i$,*

(3) *each formula ψ_i ($1 \leq i \leq n$) is a conjunction of relational atomic formulas over $\mathbf{R}_2$ mentioning plain terms built from $\bar{x}_i$ and $\bar{f}$, and*

(4) *each variable in $\bar{x}_i$ ($1 \leq i \leq n$) appears in some relational atom of φ_i.*

That is, the language of plain SO-tgds is obtained from the language of SO-tgds by forbidding equality of terms and nesting of functions. For example, formula (5.2) is a plain SO-tgd, but the formula used in the proof of Proposition 5.1.3 is not a plain SO-tgd (since it has an equality of terms). The semantics of plain SO-tgds is inherited from the semantics of SO-tgds. As for the case of SO-tgds, plain SO-tgds are closed under conjunction and, thus,

we talk about a mapping specified by a plain SO-tgds (instead of a set of plain SO-tgds). Moreover, it is easy to see that every set of st-tgds is equivalent to a plain SO-tgd.

5.2. Plain SO-tgds are Closed under CQ-Composition

Fagin, Kolaitis, Popa, and Tan (2005) show that to obtain a closure result for composition SO-tgds need equality of terms. More precisely, Fagin, Kolaitis, Popa, and Tan (2005) show that equality of terms is necessary even to specify the composition of mappings given by st-tgds. Thus, we know that we cannot obtain a closure result for composition with the language of plain SO-tgds (as this language does not consider equality of terms). Nevertheless, we show in this section that, if we relax the notion of composition to CQ-composition, then we obtain a closure result for the language of plain SO-tgds.

This result is a consequence of the following lemma that shows that even though the language of plain SO-tgds is less expressive than the language of SO-tgds, in terms of CQ-equivalence they are equally expressive. (To recall the notion of CQ-equivalence, and more generally, the notion of $\mathcal{C}$-equivalence for a query language $\mathcal{C}$, see Section 4.2.2.)

LEMMA 5.2.1. *For every SO-tgd λ, there exists a plain SO-tgd λ' such that $\lambda \equiv_{\mathrm{CQ}} \lambda'$.*

PROOF. We first recall a result that will considerably simplify the proof. Arenas, Fagin, and Nash (2010) proved that every SO-tgd λ is logically equivalent to an SO-tgd $\lambda^\star$ that does not have nesting of functions. That is, the language of SO-tgds is equivalent to the language obtained from plain SO-tgds by adding equality of plain terms (Arenas, Fagin, & Nash, 2010). Thus we can assume that SO-tgds contain only plain terms.

We also make use of the chase procedure for SO-tgds that we next introduce. Given an SO-tgd λ from $\mathbf{S}$ to $\mathbf{T}$ of the form

$$\exists \bar{f}(\forall \bar{x}_1(\varphi_1 \rightarrow \psi_1) \wedge \cdots \wedge \forall \bar{x}_k(\varphi_n \rightarrow \psi_n)), \tag{5.3}$$

(where λ only considers plain terms) the chase of the source instance I with λ, denoted by $\mathrm{chase}_\lambda(I)$, is the instance J of $\mathbf{T}$ constructed as follows (Fagin, Kolaitis, Popa, & Tan,

2005). Fix an interpretation for the function symbols of $\bar{f}$ such that for every n-ary function symbol f in $\bar{f}$ and n-ary tuple $\bar{a}$ of elements in I, $f(\bar{a})$ is interpreted as a fresh null value. Notice that with this interpretation, the equality $f(\bar{a}) = g(\bar{b})$ holds if and only if f and g are the same function symbol and $\bar{a} = \bar{b}$. Thus we can denote a null of the form $f(\bar{a})$ with its own syntactic form (that is, simply as $f(\bar{a})$). We need an additional notation. Given a formula α that considers variables in a tuple $\bar{x}$ and plain terms constructed from $\bar{f}$ and $\bar{x}$, we denote by $\alpha[\bar{x} \rightarrow \bar{a}]$ the formula obtained by replacing every occurrence of a variable in $\bar{x}$ by the corresponding value in $\bar{a}$. We are now ready to introduce the process to construct instance J. For every $1 \leq i \leq n$ and every tuple of $\bar{a}$ of elements in I such that the arity of $\bar{a}$ is the same as the arity of $\bar{x}_i$, if I satisfies the formula $\varphi_i[\bar{x}_i \rightarrow \bar{a}]$, then add all the atoms in $\psi_i[\bar{x}_i \rightarrow \bar{a}]$ to instance J. Fagin, Kolaitis, Popa, and Tan (2005) show that $\mathrm{chase}_\lambda(I)$ is a universal solution for I under the mapping specified by λ.

We next show how to transform an SO-tgd λ into a plain SO-tgd λ' such that $\mathrm{chase}_\lambda(I) = \mathrm{chase}_{\lambda'}(I)$. Let λ be an SO-tgd of the form (5.3). We first construct a set of formulas Σ' by repeating the following for every i such that $1 \leq i \leq n$. Start with φ'_i as φ_i and ψ'_i as ψ_i.

(1) Replace every equality of the form $f(x_1, \ldots, x_k) = f(y_1, \ldots, y_k)$ in φ'_i by the conjunction of equalities $x_1 = y_1 \wedge \cdots \wedge x_k = y_k$.

(2) If there is an equality of the form $x = x'$ in φ'_i with x and x' variables, then eliminate the equality and replace in φ'_i and in ψ'_i every occurrence of x' by x.

(3) Repeat the previous step until φ'_i has no equality of the form $x = x'$ with x and x' variables.

If after the process φ'_i does not contain any equality between plain terms then add formula $\forall \bar{x}'_i(\varphi'_i \rightarrow \psi'_i)$ to Σ', where $\bar{x}'_i$ is the tuple of remaining variables in φ'_i. Finally we define λ' as the formula $\exists \bar{f} \bigwedge \Sigma'$ where $\bigwedge \Sigma'$ denotes the conjunction of all the formulas in Σ'. Notice that the formula λ' is a plain SO-tgd.

It is not difficult to see that for every instance I it holds that $\mathrm{chase}_\lambda(I) = \mathrm{chase}_{\lambda'}(I)$. Just notice that if I satisfies the formula $\varphi_i[\bar{x}_i \rightarrow \bar{a}]$, then (i) φ_i does not contain equalities of the form $f(\bar{u}) = g(\bar{v})$ with f and g distinct function symbols, (ii) if $f(x_1, \ldots, x_k) =$

$f(y_1, \ldots, y_k)$ is an equality in φ_i then the equalities $x_1 = y_1$, ..., $x_k = y_k$ should hold for the assignment $\bar{x}_i \rightarrow \bar{a}$ and (iii) if $x = x'$ is an equality in φ_i, then the values a and a' that correspond to x and x' in the assignment $\bar{x}_i \rightarrow \bar{a}$, should be equal. This implies that if $\forall \bar{x}'_i(\varphi'_i \rightarrow \psi'_i)$ is in Σ', and $\bar{x}'_i \rightarrow \bar{a}'$ is the assignment obtained from the assignment $\bar{x}_i \rightarrow \bar{a}$ by considering only the variables in $\bar{x}'_i$, then I satisfies $\varphi'_i[\bar{x}'_i \rightarrow \bar{a}']$. Similarly, it can be shown that if $\forall \bar{x}'_i(\varphi'_i \rightarrow \psi'_i)$ is in Σ' and I satisfies $\varphi'_i[\bar{x}'_i \rightarrow \bar{a}']$, then I satisfies $\varphi_i[\bar{x}_i \rightarrow \bar{a}]$. The property follows from the observation that if I satisfies $\varphi_i[\bar{x}_i \rightarrow \bar{a}]$, then $\psi_i[\bar{x}_i \rightarrow \bar{a}]$ is equal to $\psi'_i[\bar{x}'_i \rightarrow \bar{a}']$.

To conclude the proof we use a result proved by Fagin, Kolaitis, Nash, and Popa (2008). Fagin, Kolaitis, Nash, and Popa (2008) showed that if two mappings share universal solutions for every source instance, then the mappings are CQ equivalent. Since for every source instance I it holds that $\mathrm{chase}_\lambda(I)$ is a universal solution for I under the mapping specified by λ, and $\mathrm{chase}_{\lambda'}(I)$ is a universal solution for I under the mapping specified by λ', and given that $\mathrm{chase}_\lambda(I) = \mathrm{chase}_{\lambda'}(I)$, we obtain that λ and λ' are CQ-equivalent. This concludes the proof of the lemma. □

The following result is a direct consequence of Lemma 5.2.1 and the fact that SO-tgds are closed under composition.

THEOREM 5.2.2. *Let $\mathcal{M}_1$ and $\mathcal{M}_2$ be mappings specified by plain SO-tgds. Then the* CQ*-composition of $\mathcal{M}_1$ and $\mathcal{M}_2$ can be specified with a plain SO-tgd.*

As a corollary of the above theorem we obtain that the CQ-composition of a finite number of mappings each specified by a set of st-tgds, is definable by a plain SO-tgd.

COROLLARY 5.2.3. *The* CQ*-composition of a finite number of mappings, each defined by a set of st-tgds, is defined by a plain SO-tgd.*

5.2.1. More properties of plain SO-tgds

In this section we include some results on the structure of plain SO-tgds that are interesting on their own. More importantly, we show that plain SO-tgds can be used to provide a negative answer to an open question posed by ten Cate and Kolaitis (2009, 2010).

Recall that an st-mapping $\mathcal{M}$ is *closed under target homomorphisms* (ten Cate & Kolaitis, 2009) if whenever $(I, J) \in \mathcal{M}$ and there is a homomorphism from J to J', then $(I, J') \in \mathcal{M}$. The next proposition shows that mappings specified by plain SO-tgds are closed under target homomorphisms.

PROPOSITION 5.2.4. *Every mapping specified by a plain SO-tgd is closed under target homomorphisms.*

PROOF. Let λ be a plain SO-tgd from $\mathbf{S}$ to $\mathbf{T}$, and assume that $(I, J) \models \lambda$. Further assume that there is a homomorphism h from J to J'. We next prove that $(I, J') \models \lambda$. Thus, assume that λ is of the form $\exists \bar{f}(\forall x_1(\varphi_1 \rightarrow \psi_1) \wedge \cdots \wedge \forall x_n(\varphi_n \rightarrow \psi_n))$. As for the case of the proof of Lemma 5.2.1, given a formula α mentioning variables from $\bar{x}$ and plain terms constructed from $\bar{x}$ and $\bar{f}$, and a tuple of elements $\bar{a}$, $\alpha[\bar{x} \rightarrow \bar{a}]$ denotes the formula obtained from α by replacing every variable in $\bar{x}$ according to the assignment $\bar{x} \rightarrow \bar{a}$. Now, since $(I, J) \models \lambda$ we know that there exists an interpretation $\bar{f}^{(I,J)}$ over (I, J) for the function symbols in $\bar{f}$ such that for every i with $1 \leq i \leq n$, if I satisfies the formula $\varphi_i[\bar{x}_i \rightarrow \bar{a}]$ for a tuple of elements $\bar{a}$ from I with interpretation $\bar{f}^{(I,J)}$, then J satisfies formula $\psi_i[\bar{x}_i \rightarrow \bar{a}]$ with interpretation $\bar{f}^{(I,J)}$. Consider now the interpretation $\bar{f}^{(I,J')}$ of $\bar{f}$ over (I, J') constructed as follows. For every function symbol f in $\bar{f}$ and tuple $\bar{b}$ of elements from (I, J), we let $f^{(I,J')}(\bar{b}) = h(f^{(I,J)}(\bar{b}))$. We show now that (I, J') satisfies λ with interpretation $\bar{f}^{(I,J')}$.

Assume that I satisfies $\varphi_i[\bar{x}_i \rightarrow \bar{a}]$ with interpretation $\bar{f}^{(I,J')}$ for some i such that $1 \leq i \leq n$. Since λ is a plain SO-tgd, then φ_i does not mention any function symbol which implies that I satisfies $\varphi_i[\bar{x}_i \rightarrow \bar{a}]$ with interpretation $\bar{f}^{(I,J)}$. Then we know that $J \models \psi_i[\bar{x}_i \rightarrow \bar{a}]$ with interpretation $\bar{f}^{(I,J)}$. Let $R(\bar{a}')$ be an arbitrary conjunct in $\psi_i[\bar{x}_i \rightarrow \bar{a}]$ with

function symbols interpreted following $\bar{f}^{(I,J)}$. Then we know that $J \models R(\bar{a}')$. Moreover, since h is a homomorphism from J to J', we know that $J' \models R(h(\bar{a}'))$. This implies that J satisfies every conjunct of $\psi_i[\bar{x}_i \to \bar{a}]$ with interpretation $\bar{f}^{(I,J')}$ which implies that $J \models \psi_i[\bar{x}_i \to \bar{a}]$ with interpretation $\bar{f}^{(I,J')}$. This was to be shown. □

It is known that mappings specified by general SO-tgds are not closed under target homomorphisms (Fagin, Kolaitis, Popa, & Tan, 2005; Fagin, Kolaitis, Nash, & Popa, 2008). Nevertheless, in the next result we show that if a mapping specified by an SO-tgd is closed under target homomorphism, then this mapping is definable by a plain SO-tgd.

PROPOSITION 5.2.5. *Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \lambda)$ be such that λ is an SO-tgd and assume that $\mathcal{M}$ is closed under target homomorphisms. Then λ is logically equivalent to a plain SO-tgd.*

PROOF. As discussed in the proof of Lemma 5.2.1, for every SO-tgd λ one can construct a plain SO-tgd λ' such that $\mathrm{chase}_\lambda(I) = \mathrm{chase}_{\lambda'}(I)$. Thus, given that $\mathrm{chase}_\lambda(I)$ is a universal solution for I and we are assuming that $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \lambda)$ is closed under target homomorphisms, we know that $(I, J) \models \lambda$ if and only if there is a homomorphism from $\mathrm{chase}_\lambda(I)$ to J. On the other hand, we know that $\mathrm{chase}_{\lambda'}(I)$ is universal for I, and since λ' is a plain SO-tgds, we also know that λ' is closed under target homomorphisms. Thus $(I, J) \models \lambda'$ if and only if there is a homomorphism from $\mathrm{chase}_{\lambda'}(I)$ to J. Finally, since $\mathrm{chase}_\lambda(I) = \mathrm{chase}_{\lambda'}(I)$, we obtain that $(I, J) \models \lambda$ if and only if $(I, J) \models \lambda'$. This completes the proof of the proposition. □

ten Cate and Kolaitis (2009) define several structural properties of schema mappings, being one of them the closure under target homomorphisms. There are two more properties that are of special interest in this section: the properties of *admitting universal solutions* and *allowing for conjunctive query rewriting* (ten Cate & Kolaitis, 2009), that we next introduce.

We say that a mapping $\mathcal{M}$ *admits universal solutions* if for every source instance I there exists a universal solution for I under $\mathcal{M}$. In particular, every mapping specified by

an SO-tgd admits universal solutions. In fact, if $\mathcal{M}$ is specified by SO-tgd λ, we know that $\mathrm{chase}_\lambda(I)$ is a universal solution for I under $\mathcal{M}$.

We say that a mapping $\mathcal{M}$ from $\mathbf{S}$ to $\mathbf{T}$ allows for conjunctive query rewriting if for every conjunctive query Q over $\mathbf{T}$, there exists a conjunctive query Q' over $\mathbf{S}$ such that $Q'(I) = \mathrm{certain}_{\mathcal{M}}(Q, I)$. It was proved by ten Cate and Kolaitis (2009) that every mapping specified by an SO-tgd allows for conjunctive query rewriting.

ten Cate and Kolaitis (2009) were interested in characterizing schema-mapping languages in terms of the structural properties that the mappings satisfy. In particular, they were interested in a language that is capable of specifying all the mappings that are closed under target homomorphisms, admit universal solutions, and allow for conjunctive query rewriting. One candidate that the authors considered is the language of *nested st-tgds* proposed by Fuxman et al. (2006). For our purposes, it is not important to formally introduce the language of nested st-tgds (we refer the reader to (Fuxman et al., 2006) and (ten Cate & Kolaitis, 2009) for details). We only mention that the language of nested st-tgds is a fragment of First-Order logic. Regarding the language of nested st-tgds, ten Cate and Kolaitis (2009) show that mappings specified in this language satisfy the three structural properties mentioned above. Furthermore, the authors posed the following question.

QUESTION 5.2.6 (ten Cate & Kolaitis, 2009, 2010). *Is it the case that a schema mapping is definable by a set of nested st-tgds if and only if it is closed under target homomorphisms, admits universal solutions, and allows for conjunctive query rewriting?*

The following proposition gives a negative answer to the above question as mappings specified by plain SO-tgds are closed under target homomorphisms, admit universal solutions, and allow for conjunctive query rewriting, but there are plain SO-tgds that are not expressible in First-Order logic.

PROPOSITION 5.2.7. [1] *The language of plain SO-tgds satisfy the following properties:*

[1] This unpublished proposition was established by Marcelo Arenas, Juan L. Reutter and myself

(1) *Every mapping specified by a plain SO-tgd is closed under target homomorphisms, admits universal solutions, and allows for conjunctive query rewriting.*

(2) *There exists a plain SO-tgd that is not expressible in First-Order logic.*

PROOF. Part (1) follows from Proposition 5.2.4, and the results by Fagin, Kolaitis, Popa, and Tan (2005) stating that mappings specified by SO-tgds admit universal solutions, and by ten Cate and Kolaitis (2009) stating that mappings specified by SO-tgds allow for conjunctive query rewriting.

To show part (2), consider the plain SO-tgd λ from $\mathbf{R}_1 = \{E(\cdot,\cdot)\}$ to $\mathbf{R}_2 = \{T(\cdot,\cdot)\}$ given by

$$\exists f\Big(\forall x\forall y\big(\ E(x,y) \rightarrow T(f(x),f(y))\ \big)\Big).$$

We show now that λ is not expressible by a First-Order logic sentence over the vocabulary $\mathbf{R}_1 \cup \mathbf{R}_2$. To obtain a contradiction, assume that λ is expressible in FO, and let Φ be an FO sentence over $\mathbf{R}_1 \cup \mathbf{R}_2$ that is logically equivalent to λ. Let u and v be two variables not mentioned in Φ and consider the sentence Φ' obtained from Φ by replacing every relational atom $T(x,y)$ by formula $(x = u \wedge y = v) \vee (x = v \wedge y = u)$. Now let Ψ be the FO sentence over $\mathbf{R}_1$ defined by

$$\exists u\exists v(u \neq v \wedge \Phi').$$

It is not difficult to see that $I \models \Psi$ if and only if for the instance J such that $T^J = \{(1,2),(2,1)\}$, it holds that $(I,J) \models \Phi$. Given that we are assuming that Φ is logically equivalent to λ, we obtain that $I \models \Psi$ if and only if there exists an interpretation $f^\star$ for the function symbol f such that for every element c in I it holds that $f^\star(c)$ is either 1 or 2, and for every tuple (a,b) in E^I it holds that $f^\star(a) \neq f^\star(b)$. That is, we have that $I \models \Psi$ if and only if the graph represented by I is 2-colorable. This is our desired contradiction since 2-colorability is not expressible in FO (Libkin, 2004). □

Notice that by using a similar argument as the one used in the above proof, we can obtain a contradiction by using 3-colorability instead of 2-colorability. Thus, by a result by Dawar (1998), we obtain that λ cannot even be defined in the finite-variable infinitary

logic $\mathcal{L}^{\omega}_{\infty\omega}$, which is strictly more expressive than FO (see (Libkin, 2004) for a definition of $\mathcal{L}^{\omega}_{\infty\omega}$).

It is an open problem whether the language of plain SO-tgds exactly characterizes the class of mappings that are closed under target homomorphisms, admit universal solutions, and allow for conjunctive query rewriting. In fact, in view of Proposition 5.2.7 one can propose two questions that remain open. The first one is a refinement of Question 5.2.6 for the case of mappings specified in FO.

QUESTION 5.2.8. *Is it the case that a schema mapping is definable by a set of nested st-tgds if and only if it is definable in* FO*, is closed under target homomorphisms, admits universal solutions, and allows for conjunctive query rewriting?*

QUESTION 5.2.9. *Is it the case that a schema mapping is definable by a plain SO-tgd if and only if it is closed under target homomorphisms, admits universal solutions, and allows for conjunctive query rewriting?*

5.3. Inverting Plain SO-tgds (in Polynomial Time)

We show in this section that every st-mapping specified by a plain SO-tgd has a maximum recovery. To prove this we present in this section a polynomial time algorithm that, given a set of plain SO-tgds, returns a maximum recovery that is expressed in a language that extends plain SO-tgds with some extra features.

We start by giving some of the intuition behind the algorithm. Consider the following plain SO-tgd:

$$\exists f \exists g \Big(\forall x \forall y \forall z \big(R(x,y,z) \ \rightarrow \ T(x, f(y), f(y), g(x,z)) \big) \Big). \tag{5.4}$$

When exchanging data with an SO-tgd like (5.4), the standard assumption is that every application of a function symbol generates a fresh value (Fagin, Kolaitis, Popa, & Tan, 2005). For example, consider a source instance $\{R(1,2,3)\}$. When we exchange data with (5.4), we obtain a canonical target instance $\{T(1,a,a,b)\}$, where $a = f(2)$, $b = g(1,3)$, and $a \neq b$. The intuition behind our algorithm is to produce a reverse mapping that

focuses on this canonical target instance to recover as much source data as possible. Thus, in order to invert a dependency like (5.4), we consider three unary functions f_1, g_1 and g_2. The idea is that f_1 represents the inverse of function f, while (g_1, g_2) represents the inverse of g. Notice that since g has two arguments, we need to use two functions to represent its inverse. Thus, considering the above example, the intended meaning of the functions is $f_1(a) = 2$, $g_1(b) = 1$, and $g_2(b) = 3$. With this in mind, we can represent an inverse of the plain SO-tgd (5.4) with a dependency of the form:

$$\exists f_1 \exists g_1 \exists g_2 \Big(\forall u \forall v \forall w \big(T(u, v, v, w) \ \rightarrow \ R(u, f_1(v), g_2(w)) \ \wedge \ u = g_1(w) \big) \Big). \quad (5.5)$$

Notice that, if we use dependency (5.5) to exchange data back from instance $\{T(1, a, a, b)\}$, we obtain an instance $\{R(1, f_1(a), g_2(b))\}$. The equality $u = g_1(w)$ has been added in order to ensure the correct interpretation of g_1 as the inverse function of g. In the example, the equality ensures that $g_1(b)$ is 1.

In order to obtain a correct algorithm we need another technicality, that we describe next. We have mentioned that when exchanging data with SO-tgds, we assume that every application of a function produces a fresh value. In the above example, we have that value a is the result of applying f to 2, thus, we know that value a cannot be obtained with any other function. In particular, a cannot be obtained as an application of function g. Thus, when exchanging data back we should ensure that at most one inverse function is applied to every possible target value. We do that by using an additional unary function $f_\star$. In the above example, whenever we apply function f_1 to some value v, we require that $f_1(v) = f_\star(v)$ and that $g_1(v) \neq f_\star(v)$. Similarly, whenever we apply function g_1 to some value w, we require that $g_1(w) = f_\star(w)$ and that $f_1(w) \neq f_\star(w)$. Thus, for example, if we apply f_1 to value a, we require that $f_1(a) = f_\star(a)$ and that $g_1(a) \neq f_\star(a)$. Notice that this forbids the application of g_1 to a, since, if that were the case, we would require that $g_1(a) = f_\star(a)$, which contradicts the previous requirement $g_1(a) \neq f_\star(a)$. Our algorithm adds these equalities and inequalities as conjuncts in the conclusions of

dependencies. Considering the example, our algorithm adds

$$f_1(v) = f_\star(v) \ \wedge \ g_1(v) \neq f_\star(v) \ \wedge$$
$$g_1(w) = f_\star(w) \ \wedge \ f_1(w) \neq f_\star(w)$$

to the right-hand side of the implication of dependency (5.5), and also adding the existential quantification over function $f_\star$. The final SO dependency that specifies a maximum recovery of the plain SO-tgd (5.4) is[2]

$$\exists f_\star \exists f_1 \exists g_1 \exists g_2 \Big[\forall u \forall v \forall w \Big(T(u,v,v,w) \ \rightarrow \ R(u, f_1(v), g_2(w)) \ \wedge \ u = g_1(w) \wedge f_1(v) = f_\star(v) \ \wedge \ g_1(v) \neq f_\star(v) \wedge g_1(w) = f_\star(w) \ \wedge \ f_1(w) \neq f_\star(w) \Big) \Big]. \quad (5.6)$$

Before presenting our algorithm, we make some observations Although we have assumed in the above explanation that every application of a function generates a fresh value, we remark that this assumption has only been used as a guide in the design of our algorithm. In fact, it is shown in Theorem 5.3.1 that the algorithm presented in this section produces inverses for the general case, where no assumption about the function symbols is made. We now formalize our algorithm to compute maximum recoveries of plain SO-tgds

Preliminary procedures

We start by fixing some notation. Given a plain SO-tgd λ, we denote by F_λ the set of function symbols that occur in λ. We also consider a set of function symbols F'_λ constructed as follows. For every n-ary function symbol f in F_λ, the set F'_λ contains n unary function symbols $f_1, \ldots, f_n$. Additionally, F'_λ contains a unary function symbol $f_\star$. For example, for plain SO-tgd (5.4), $F_\lambda = \{f, g\}$ and $F'_\lambda = \{f_1, g_1, g_2, f_\star\}$.

[2]Our algorithm also uses predicate $\mathbf{C}(\cdot)$ over variables that are in the left and right-hand side of the implication formulas.

We describe the procedures CREATETUPLE, ENSUREINV, and SAFE. These procedures are the building blocks of the algorithm that computes an inverse of a plain SO-tgd.

Procedure CREATETUPLE($\bar{t}$) receives as input a tuple $\bar{t} = (t_1, \ldots, t_n)$ of plain terms. Then it builds an n-tuple of variables $\bar{u} = (u_1, \ldots, u_n)$ such that, if $t_i = t_j$ then u_i and u_j are the same variable, and they are distinct variables otherwise. For example, consider the right-hand side of the implication of dependency (5.4). In the argument of relation T we have the tuple of terms $\bar{t} = (x, f(y), f(y), g(x, z))$. In this case, we have that procedure CREATETUPLE($\bar{t}$) returns a tuple of the form (u, v, v, w). Notice that we have used this tuple as the argument of T in the left-hand side of the implication of dependency (5.5).

Tuple $\bar{u}$ created with CREATETUPLE is used as an input in the following two procedures. We now formalize the procedure to obtain a formula that guarantees the correct use of the inverse function symbols.

Procedure: ENSUREINV($\lambda, \bar{u}, \bar{s}$)

Input: A plain SO-tgd λ, an n-tuple $\bar{u} = (u_1, \ldots, u_n)$ of (not necessarily distinct) variables, and an n-tuple of plain terms $\bar{s}$ built from F_λ and a tuple of variables $\bar{y}$.

Output: A formula Q_e consisting of conjunctions of equalities between terms built from F'_λ and $\bar{u}$, $\bar{y}$.

(1) Let $\bar{s} = (s_1, \ldots, s_n)$.

(2) Construct formula Q_e as follows. For every $i \in \{1, \ldots, n\}$ do the following:

- If s_i is a variable y, then add equality $u_i = y$ as a conjunct in Q_e.
- If s_i is a term of the form $f(y_1, \ldots, y_k)$, then add the conjunction of equalities

$$f_1(u_i) = y_1 \wedge \cdots \wedge f_k(u_i) = y_k$$

to Q_e, where $f_1, \ldots, f_k$ are the k unary functions in F'_λ associated with f.

(3) Return Q_e. □

As an example, let λ be the dependency (5.4), $\bar{s} = (x, f(y), f(y), g(x, z))$ the tuple of terms in the right-hand side of the implication of λ, and $\bar{u} = (u, v, v, w)$. When running

the procedure ENSUREINV($\lambda, \bar{u}, \bar{s}$), we have that $u_4 = w$ and $s_4 = g(x, z)$. Thus, in the loop of Step 2, the conjunction $g_1(w) = x \ \wedge \ g_2(w) = z$ is added to formula Q_e. The final output of the procedure in this case is:

$$u = x \ \wedge \ f_1(v) = y \ \wedge \ g_1(w) = x \ \wedge \ g_2(w) = z. \tag{5.7}$$

We need to describe one more procedure, which guarantees that it could not be the case that a value in the target was generated by two distinct functions.

Procedure: SAFE($\lambda, \bar{u}, \bar{s}$)

Input: A plain SO-tgd λ, an n-tuple $\bar{u} = (u_1, \ldots, u_n)$ of not necessarily distinct variables, and an n-tuple of plain terms $\bar{s}$ built from F_λ and a tuple of variables $\bar{y}$.

Output: A formula Q_s consisting of equalities and inequalities between terms built from F'_λ and $\bar{u}$.

(1) Let $\bar{s} = (s_1, \ldots, s_n)$.

(2) Construct formula Q_s as follows. For every $i \in \{1, \ldots, n\}$ do the following:

- If s_i is a term of the form $f(y_1, \ldots, y_k)$, then add the following conjuncts to Q_s:
  - The equality $f_\star(u_i) = f_1(u_i)$.
  - The inequality $f_\star(u_i) \neq g_1(u_i)$, for every function symbol g in F_λ different from f.

(3) Return Q_s. □

Considering λ as the dependency (5.4), $\bar{s}$ the tuple of terms $(x, f(y), f(y), g(x, z))$ and $\bar{u} = (u, v, v, w)$, the algorithm SAFE($\lambda, \bar{u}, \bar{s}$) returns:

$$f_1(v) = f_\star(v) \ \wedge \ g_1(v) \neq f_\star(v) \ \wedge \ g_1(w) = f_\star(w) \ \wedge \ f_1(w) \neq f_\star(w). \tag{5.8}$$

Notice that all the procedures presented so far work in polynomial time with respect to the size of their inputs.

Building the inverse

We need some additional notation before we present the algorithm for computing inverses of plain SO-tgds. Let $\bar{t}$ and $\bar{s}$ be n-tuples of plain terms. Then we say that $\bar{t}$ *is subsumed by* $\bar{s}$ (or $\bar{s}$ *subsumes* $\bar{t}$) if, whenever the i-th component of $\bar{t}$ contains a variable, the i-th component of $\bar{s}$ also contains a variable. Notice that if $\bar{s}$ subsumes $\bar{t}$, then whenever the i-th component of $\bar{s}$ contains a non-atomic term, the i-th component of $\bar{t}$ also contains a non-atomic term. For example, the tuple of terms $(x, f(y), f(y), g(x, z))$ is subsumed by $(u, v, h(u), h(v))$.

The following algorithm computes a maximum recovery of a plain SO-tgd λ in polynomial time. As a consequence of the results in Section 3.1.1 and the discussion in Section 4.2, the algorithm can also be used to compute Fagin-inverses, quasi-inverses, as well as CQ-maximum recoveries.

Algorithm: POLYSOINVERSE($\mathcal{M}$)

Input: An st-mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \lambda)$ with λ a plain SO-tgd of the form $\exists \bar{f}(\sigma_1 \wedge \cdots \wedge \sigma_n)$.

Output: A ts-mapping $\mathcal{M}' = (\mathbf{T}, \mathbf{S}, \lambda')$ that specifies a maximum recovery of λ such that λ' is an SO dependency.

(1) Let $\Sigma = \{\sigma_1, \ldots, \sigma_n\}$, and Σ' be empty.

(2) Normalize Σ as follows. For every $i \in \{1, \ldots, n\}$ do:
   - If σ_i is of the form $\forall \bar{x}(\varphi(\bar{x}) \rightarrow R_1(\bar{t}_1) \wedge \ldots \wedge R_\ell(\bar{t}_\ell))$, then replace σ_i by ℓ dependencies $\forall \bar{x}_1(\varphi_1(\bar{x}_1) \rightarrow R_1(\bar{t}_1)), \ldots, \forall \bar{x}_\ell(\varphi_\ell(\bar{x}_\ell) \rightarrow R_\ell(\bar{t}_\ell))$ such that for every $1 \leq i \leq \ell$:
     - $\bar{x}_i$ is exactly the tuple of variables shared by $\bar{x}$ and $\bar{t}_i$
     - $\varphi_i(\bar{x}_i)$ is the formula obtained from $\varphi(\bar{x})$ by existentially quantifying the variables of $\bar{x}$ not mentioned in $\bar{x}_i$.

(3) For every σ of the form $\forall \bar{x}(\varphi(\bar{x}) \rightarrow R(\bar{t}))$ in the normalized set Σ, where $\bar{t} = (t_1, \ldots, t_m)$ is a tuple of plain terms, do the following:
   (a) Let $\bar{u} =$ CREATETUPLE($\bar{t}$).

(b) Let $prem_\sigma(\bar{u})$ be a formula defined as the conjunction of the atom $R(\bar{u})$ and the formulas $\mathbf{C}(u_i)$ for every i such that t_i is a variable.

(c) Create a set of formulas Γ_σ as follows. For every dependency $\forall\bar{y}(\psi(\bar{y}) \rightarrow R(\bar{s}))$ in Σ such that $\bar{s}$ subsumes $\bar{t}$, do the following:

- Let $Q_e = \textsc{EnsureInv}(\lambda, \bar{u}, \bar{s})$.
- Let $Q_s = \textsc{Safe}(\lambda, \bar{u}, \bar{s})$.
- Add to Γ_σ the formula $\exists\bar{y}\,\big(\,\psi(\bar{y}) \;\wedge\; Q_e \wedge\; Q_s\,\big)$

(d) Add to Σ' the dependency:

$$\forall\bar{u}\big(prem_\sigma(\bar{u}) \rightarrow \gamma_\sigma(\bar{u})\big)$$

where $\gamma_\sigma(\bar{u})$ is the disjunction of the formulas in Γ_σ.

(4) Let $\lambda' = \exists\bar{f}'\,(\bigwedge\Sigma')$, where $\bar{f}'$ is a tuple containing the function symbols in F'_λ. Return $\mathcal{M}' = (\mathbf{T}, \mathbf{S}, \lambda')$. □

As an example of the execution of the algorithm let λ be the plain SO-tgd (5.4), and assume that $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \lambda)$ is the input of algorithm PolySOInverse. In Step 3 of algorithm $\textsc{PolySOInverse}(\mathcal{M})$ we have to consider a single dependency $\sigma = R(x, y, z) \rightarrow T(x, f(y), f(y), g(x, z))$. Let $\bar{t}$ be the tuple of terms $(x, f(y), f(y), g(x, z))$. Recall that $\textsc{CreateTuple}(\bar{t})$ is a tuple of the form (u, v, v, w) and, thus,

$$prem_\sigma(\bar{u}) \;\; = \;\; T(u, v, v, w) \wedge \mathbf{C}(u)$$

is built in Step 3.b. Notice that $\mathbf{C}(u)$ has been added since the first component of $\bar{t}$ is the variable x. Then in Step 3.c, we need to consider just dependency σ. Notice that $\bar{t}$ subsumes itself and, hence, formula

$$\exists x \exists y \exists z\,(R(x, y, z) \wedge Q_e \wedge Q_s)$$

is added to the set Γ_σ, where Q_e is the formula (5.7) and Q_s is the formula (5.8). Notice that $F'_\lambda = \{f_\star, f_1, g_1, g_2\}$, and thus, the following formula λ' is created in the last step of

the algorithm:

$$\exists f_\star \exists f_1 \exists g_1 \exists g_2 \Big[\forall u \forall v \forall w \Big(T(u,v,v,w) \ \wedge \ \mathbf{C}(u) \ \rightarrow \ \exists x \exists y \exists z \ \big(\ R(x,y,z) \ \wedge \ u = x \ \wedge \ f_1(v) = y \ \wedge \ g_1(w) = x \ \wedge \ g_2(w) = z \ \wedge \ Q_s \big) \Big) \Big]. \quad (5.9)$$

Notice that the existentially quantified variables can be eliminated from dependency (5.9). Thus, replacing formula Q_s and eleminating the existential quantification in the right-hand side of the implication, we obtain that dependency (5.9) is equivalent to:

$$\begin{aligned} \exists f_\star \exists f_1 \exists g_1 \exists g_2 \Big[\forall u \forall v \forall w \Big(& T(u,v,v,w) \ \wedge \ \mathbf{C}(u) \ \rightarrow \ R(u, f_1(v), g_2(w)) \ \wedge \ u = g_1(w) \ \wedge \\ & f_1(v) = f_\star(v) \ \wedge \ g_1(v) \neq f_\star(v) \ \wedge \\ & g_1(w) = f_\star(w) \ \wedge \ f_1(w) \neq f_\star(w) \Big) \Big]. \end{aligned}$$

which specifies a maximum recovery of $\mathcal{M}$.

THEOREM 5.3.1. *Let $\mathcal{M}$ be a mapping specified by a plain SO-tgd λ. Algorithm* POLYSOINVERSE($\mathcal{M}$) *computes in polynomial time a mapping $\mathcal{M}'$ specified by an SO dependency such that $\mathcal{M}'$ is a maximum recovery of $\mathcal{M}$.*

PROOF. Let λ be a plain SO-tgd of the form $\exists \bar{f}(\sigma_1 \wedge \cdots \wedge \sigma_k)$, and $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \lambda)$. Let Σ be the set $\{\sigma_1, \ldots, \sigma_k\}$. Notice that after Step 2 of the algorithm, we have that every σ in the (normalized) set Σ, is a formula of the form $\forall \bar{x}(\varphi(\bar{x}) \rightarrow R(\bar{t}(\bar{x})))$ where:

- $\varphi(\bar{x})$ is a CQ formula over $\mathbf{S}$ with $\bar{x}$ as tuple of free variables,
- R is an n-ary relation symbol in $\mathbf{T}$,
- $\bar{t}(\bar{x})$ is an n-tuple of plain terms constructed by using functions from $\bar{f}$ and variables from $\bar{x}$, and
- $\bar{x}$ is exactly the tuple of (distinct) variables that $\varphi(\bar{x})$ and $\bar{t}(\bar{x})$ share.

Notice that in the above notation we have made explicit the variables mentioned in the tuple of terms $\bar{t}(\bar{x})$. Additionally, we assume that all the formulas in Σ have pair-wise disjoint sets of variables. It is straightforward to see that the formula $\exists \bar{f} \bigwedge \Sigma$ obtained after the normalization step is logically equivalent to the original plain SO-tgd provided as input for the algorithm.

Before continuing with the proof, recall that F_λ is the set of function symbols in $\bar{f}$. Also recall that F'_λ is the set of function symbols constructed as follows. For every n-ary function symbol f in $\bar{f}$, the set F'_λ contains n unary function symbols $f_1, \ldots, f_n$. Additionally, F'_λ contains a new unary function symbol $f_\star$.

We introduce some notation regarding the procedures CREATETUPLE, ENSUREINV and SAFE. First, given a tuple of terms $\bar{t}(\bar{x}) = (t_1(\bar{x}), \ldots, t_n(\bar{x}))$ we use $\bar{u}_{\bar{t}(\bar{x})}$ to denote the output of CREATETUPLE$(\bar{t}(x))$. That is, $\bar{u}_{\bar{t}(\bar{x})}$ is an n-tuple of variables $(u_1, \ldots, u_n)$ such that, for $i \neq j$, if the i-th component of $\bar{t}(\bar{x})$ is equal to the j-th component of $\bar{t}(\bar{x})$, then u_i and u_j are the same variable, and they are different variables otherwise. Consider for example the tuple of terms $\bar{t}(x_1, x_2) = (x_1, f(x_1), f(x_1), x_1, x_2)$. In this case we have that $t_1(x_1, x_2) = t_4(x_1, x_2) = x_1$, $t_2(x_1, x_2) = t_3(x_1, x_2) = f(x_1)$, and $t_5(x_1, x_2) = x_2$, and then $\bar{u}_{\bar{t}(x_1,x_2)}$ is a tuple of the form $(u_1, u_2, u_2, u_1, u_3)$. Second, given an n-tuple $\bar{u} = (u_1, \ldots, u_n)$ of not necessarily distinct variables, and an n-tuple $\bar{s}(\bar{y}) = (s_1(\bar{y}), \ldots, s_n(\bar{y}))$ of plain terms, we denote by $\nu(\bar{u}, \bar{s}(\bar{y}))$ the formula obtained as output of procedure ENSUREINV$(\lambda, \bar{u}, \bar{s}(\bar{y}))$. Thus, we have that for every i with $1 \leq i \leq n$ it holds that:

- If $s_i(\bar{y})$ is a variable y_m of $\bar{y}$, then $\nu(\bar{u}, \bar{s}(\bar{y}))$ contains the equality $u_i = y_m$ as a conjunction.
- If $s_i(\bar{y})$ is a non-atomic term $f(y_{m_1}, \ldots, y_{m_k})$, with f a k-ary function symbol of $\bar{f}$ and every y_{m_j} a variable in $\bar{y}$, then $\nu(\bar{u}, \bar{s}(\bar{y}))$ contains the conjunction of equalities $f_1(u_i) = y_{m_1} \wedge \cdots \wedge f_k(u_i) = y_{m_k}$, where $f_1, \ldots, f_k$ are the k unary functions in $\bar{f}'$ associated with f.

For example, let $\bar{u} = (u_1, u_2, u_1)$ and $\bar{s}(y_1, y_2, y_3) = (y_1, f(y_3, y_1, y_2), g(y_2))$. Notice that $s_1(y_1, y_2, y_3) = y_1$, $s_2(y_1, y_2, y_3) = f(y_3, y_1, y_2)$ and $s_3(y_1, y_2, y_3) = g(y_2)$. In this case we have that $\nu(\bar{u}, \bar{s}(y_1, y_2, y_3))$ is the formula

$$u_1 = y_1 \ \wedge \ f_1(u_2) = y_3 \ \wedge \ f_2(u_2) = y_1 \ \wedge \ f_3(u_2) = y_2 \ \wedge \ g_1(u_1) = y_2.$$

Third, we denote by $\omega(\bar{u}, \bar{s}(\bar{y}))$ the formula obtained as the output of SAFE$(\lambda, \bar{u}, \bar{s}(\bar{y}))$. That is, we have that for every i with $1 \leq i \leq n$, if $s_i(\bar{x})$ is a non-atomic term of the form $f(y_{m_1}, \ldots, y_{m_k})$ with f a k-ary function symbol of $\bar{f}$ and every y_{m_j} a variable in $\bar{y}$, then $\omega(\bar{u}, \bar{s}(\bar{y}))$ contains as conjunctions:

- the equality $f_\star(u_i) = f_1(u_i)$ and,
- the inequality $f_\star(u_i) \neq g_1(u_i)$, for every function symbol g in $\bar{f}$ different from f.

Notice that $\omega(\bar{u}, \bar{s}(\bar{y}))$ may be the empty formula if the tuple $\bar{s}(\bar{y})$ is composed only by variables. In that case, we assume that $\omega(\bar{u}, \bar{s}(\bar{y}))$ is $true$ (an arbitrary valid formula). We have introduced this notation only to make explicit the variables used in the tuple of terms $\bar{t}(\bar{x})$ and $\bar{s}(\bar{y})$ in the inputs of CREATETUPLE, ENSUREINV, and SAFE.

Finally, given a dependency σ in Σ of the form $\varphi(\bar{x}) \ \rightarrow \ R(\bar{t}(\bar{x}))$, in this proof we denote by $\mathcal{C}_{R(\bar{t}(\bar{x}))}$ the set Γ_σ constructed in Step 3.c. That is, for every dependency of the form $\psi(\bar{y}) \rightarrow R(\bar{s}(\bar{y}))$ in Σ such that $\bar{s}(\bar{y})$ subsumes $\bar{t}(\bar{x})$, then $\mathcal{C}_{R(\bar{t}(x))}$ includes the formula

$$\exists \bar{y} \big(\ \psi(\bar{y}) \ \wedge \ \nu(\bar{u}_{\bar{t}(\bar{x})}, \bar{s}(\bar{y})) \ \wedge \ \omega(\bar{u}_{\bar{t}(\bar{x})}, \bar{s}(\bar{y})) \ \big).$$

Notice that in the above formula we are using our new notation for the outputs of procedures CREATETUPLE$(\bar{t}(x))$, ENSUREINV$(\lambda, \bar{u}, \bar{s}(\bar{y}))$, and SAFE$(\lambda, \bar{u}, \bar{s}(\bar{y}))$.

With our new notation we have that the set Σ' constructed in POLYSOINVERSE contains, for every $\sigma \in \Sigma$ of the form $\forall \bar{x}(\varphi(\bar{x}) \rightarrow R(\bar{t}(\bar{x})))$, a dependency σ' such that

- the premise of σ' is composed of the atom $R(\bar{u}_{\bar{t}(\bar{x})})$ and formulas $\mathbf{C}(u_i)$ for every i such that $t_i(\bar{x})$ is a variable x of $\bar{x}$, and
- the conclusion of σ' is the disjunction of all the formulas in $\mathcal{C}_{R(\bar{t}(\bar{x}))}$.

We are now ready to continue with the proof. For simplicity, in what follows we omit the universal quantification in the formulas in Σ and Σ'. That is, if σ is a formula in Σ of the form $\forall\bar{x}(\varphi(\bar{x}) \rightarrow R(\bar{t}(\bar{x})))$, we just write $\varphi(\bar{x}) \rightarrow R(\bar{t}(\bar{x}))$ to denote σ. Let $\mathcal{M}$ be the st-mapping specified by the formula $\lambda = \exists\bar{f} \bigwedge \Sigma$, and $\mathcal{M}'$ the ts-mapping specified by the formula $\lambda' = \exists\bar{f}' \bigwedge \Sigma'$ constructed in the last step of algorithm POLYSOINVERSE. We need to show that $\mathcal{M}'$ is a maximum recovery of $\mathcal{M}$. We first show that $\mathcal{M}'$ is a recovery of $\mathcal{M}$. Let I be a source instance, and J the result of chasing I with λ. Recall that J is constructed as follows. For every σ in Σ of the form $\varphi(\bar{x}) \rightarrow R(\bar{t}(\bar{x}))$ and for every tuple $\bar{a}$ of constant values such that $I \models \varphi(\bar{a})$, we include in J the tuple $R(\bar{t}(\bar{a}))$. Notice that in this procedure, every ground term is viewed as a distinct value (for example, $f(a)$ and $g(a)$ are considered to be distinct values), and every ground non-atomic term is considered to be a null value. We claim that $(J, I) \in \mathcal{M}'$ which proves that $(I, I) \in \mathcal{M} \circ \mathcal{M}'$. In order to show that $(J, I) \models \lambda'$, we need to prove that there exists an *interpretation* for the functions of $\bar{f}'$ such that $(J, I) \models \bigwedge \Sigma'$. For every k-ary function f in $\bar{f}$ consider the interpretation of f_i with $1 \leq i \leq k$ as follows:

- for every k-tuple $(a_1, \ldots, a_k)$, we have that $f_i(f(a_1, \ldots, a_k)) = a_i$,
- f_i is an arbitrary value in every other case.

That is f_i is interpreted as the *projection* of f over component i. Additionally, we interpret function $f_\star$ as follows. For every k-ary function f in $\bar{f}$ we let $f_\star(f(a_1, \ldots, a_k)) = a_1$, and $f_\star$ is an arbitrary value in every other case. Notice that this interpretation is well defined since every ground term produced when chasing I, is viewed as a distinct value. We show now that with this interpretation, it holds that $(J, I) \models \bigwedge \Sigma'$.

Let σ' be a formula in Σ'. We need to show that (J, I) satisfies σ'. Assume that σ' was created from a formula in Σ of the form $\varphi(\bar{x}) \rightarrow R(\bar{t}(\bar{x}))$. Assume that R is an n-ary relation symbol. Then σ' is of the form $R(\bar{u}_{\bar{t}(\bar{x})}) \wedge \mathbf{C}(\bar{u}') \rightarrow \alpha(\bar{u}_{\bar{t}(\bar{x})})$ with $\bar{u}_{\bar{t}(\bar{x})}$ an n-tuple of (not necessarily distinct) variables, $\bar{u}' \subseteq \bar{u}_{\bar{t}(\bar{x})}$, and $\alpha(\bar{u}_{\bar{t}(\bar{x})})$ the disjunction of the formulas in $\mathcal{C}_{R(\bar{t}(\bar{x}))}$. Now, suppose that there exists an n-tuple $\bar{b}$ of ground terms such that $J \models R(\bar{b}) \wedge \mathbf{C}(\bar{b}')$ (with $\bar{b}'$ the corresponding assignment to $\bar{u}'$ that derives from the

assignment of $\bar{b}$ to $\bar{u}_{\bar{t}(\bar{x})}$). We must show that $I \models \alpha(\bar{b})$. Since J is the result of chasing I with λ, we know that there exists a formula $\psi(\bar{y}) \rightarrow R(\bar{s}(\bar{y}))$ that is used to generate $R(\bar{b})$ in J. Then there exists a tuple $\bar{a}$ of constants such that $I \models \psi(\bar{a})$ and $\bar{s}(\bar{a}) = \bar{b}$. Now, given that $\mathbf{C}(\bar{b}')$ holds and $\bar{s}(\bar{a}) = \bar{b}$, we conclude that $\bar{s}(\bar{y})$ subsumes $\bar{t}(\bar{x})$. Consequently, the formula

$$\beta(\bar{u}_{\bar{t}(\bar{x})}) = \exists \bar{y}\big(\ \psi(\bar{y})\ \wedge\ \nu(\bar{u}_{\bar{t}(\bar{x})}, \bar{s}(\bar{y}))\ \wedge\ \omega(\bar{u}_{\bar{t}(\bar{x})}, \bar{s}(\bar{y}))\ \big)$$

belongs to $\mathcal{C}_{R(\bar{t}(\bar{x}))}$, and then, it is a disjunct in $\alpha(\bar{u}_{\bar{t}(\bar{x})})$. We claim that $I \models \beta(\bar{b}) = \beta(\bar{s}(\bar{a}))$ and then $I \models \alpha(\bar{b})$. Notice that $I \models \psi(\bar{a})$ and by the interpretation of the functions of $\bar{f}'$ it is straightforward to see that $\nu(\bar{s}(\bar{a}), \bar{s}(\bar{a}))$ and $\omega(\bar{s}(\bar{a}), \bar{s}(\bar{a}))$ holds. Then we have that $I \models \beta(\bar{b})$ with the chosen interpretation for the functions in $\bar{f}'$ by using tuple $\bar{a}$ as the witness for the tuple $\bar{y}$ of existentially quantified variables, and then $I \models \alpha(\bar{b})$. We have shown that, with the chosen interpretation for the functions in $\bar{f}'$, it holds that (J, I) satisfies every formula in Σ', and then (J, I) satisfies $\bigwedge \Sigma'$, which was to be shown.

To complete the proof it is enough to show that, if $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$ then $\mathrm{Sol}_{\mathcal{M}}(I_2) \subseteq \mathrm{Sol}_{\mathcal{M}}(I_1)$ (see Proposition 3.1.6). To simplify the exposition we introduce some notation. A *ground plain term* is a term of the form $f(a_1, \ldots, a_k)$ where $a_1, \ldots, a_k$ are values from some domain. Let $\bar{p}$ be a tuple of groun plain terms constructed by using function symbols from a tuple $\bar{g}$, and let $\bar{g}^0$ be an interpretation for the function symbols in $\bar{g}$. We write $\bar{p}[\bar{g} \mapsto \bar{g}^0]$ to denote the tuple obtained by replacing every ground plain term using a function symbol in $\bar{g}$ by its corresponding interpretation in $\bar{g}^0$. Similarly, if γ is a conjunction of ground atoms we write $\gamma[\bar{g} \mapsto \bar{g}^0]$ to denote the conjunction obtained by replacing every ground plain term by its corresponding interpretation in $\bar{g}^0$. Abusing of the notation, for a first order formula α that mentions plain terms constructed from function symbols in $\bar{g}$ we write $I \models \alpha[\bar{g} \mapsto \bar{g}^0]$ to denote that I satisfies α with the interpretation $\bar{g}^0$ for the function symbols in $\bar{g}$. To continue with the proof, let I_1 and I_2 be source instances such that $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$, and assume that $(I_2, J^\star) \in \mathcal{M}$. We need to show that $(I_1, J^\star) \in \mathcal{M}$. Since $(I_1, I_2) \in \mathcal{M} \circ \mathcal{M}'$, there exists a target instance J such that $(I_1, J) \in \mathcal{M}$ and $(J, I_2) \in \mathcal{M}'$. Thus, we have that $(I_1, J) \models \exists \bar{f} \bigwedge \Sigma$, and $(J, I_2) \models \exists \bar{f}' \bigwedge \Sigma'$. Then we know that there exists an interpretation $\bar{f}^{(I_1, J)}$ for the functions in $\bar{f}$, and an

interpretation $\bar{f}'^{(J,I_2)}$ for the functions in $\bar{f}'$, such that $(I_1, J) \models (\bigwedge \Sigma)[\bar{f} \mapsto \bar{f}^{(I_1,J)}]$ and $(J, I_2) \models (\bigwedge \Sigma')[\bar{f}' \mapsto \bar{f}'^{(J,I_2)}]$. Moreover, since $(I_2, J^\star) \in \mathcal{M}$ we know that there exists an interpretation $\bar{f}^{(I_2,J^\star)}$ for the functions in $\bar{f}'$, such that $(I_2, J^\star) \models (\bigwedge \Sigma)[\bar{f} \mapsto \bar{f}^{(I_2,J^\star)}]$. We need to show that there exists an interpretation $\bar{f}^{(I_1,J^\star)}$ for $\bar{f}$, such that $(I_1, J^\star) \models (\bigwedge \Sigma)[\bar{f} \mapsto \bar{f}^{(I_1,J^\star)}]$. We describe now how to construct $\bar{f}^{(I_1,J^\star)}$ from $\bar{f}^{(I_1,J)}$, $\bar{f}'^{(J,I_2)}$ and $\bar{f}^{(I_2,J^\star)}$. Let f be a k-ary function symbol in $\bar{f}$, and let $\bar{a}$ be a k-tuple of constant values. Define $f^{(I_1,J^\star)}(\bar{a})$ as follows:

- Assume that there exists a unique function symbol g_1 in $\bar{f}'$, such that

$$f_\star^{(J,I_2)}(f^{(I_1,J)}(\bar{a})) = g_1^{(J,I_2)}(f^{(I_1,J)}(\bar{a})). \tag{5.10}$$

  Then, if g_1 is associated with the k-ary function symbol g from $\bar{f}$, we let

$$f^{(I_1,J^\star)}(\bar{a}) = g^{(I_2,J^\star)}\Big(g_1^{(J,I_2)}\big(f^{(I_1,J)}(\bar{a})\big),\ \ldots,\ g_k^{(J,I_2)}\big(f^{(I_1,J)}(\bar{a})\big)\Big).$$

- Otherwise, if there is no function symbol in $\bar{f}'$ satisfying equality (5.10), or there is more than one function symbol in $\bar{f}'$ satisfying (5.10), then $f^{(I_1,J^\star)}(\bar{a})$ is an arbitrary value.

We show next that, with $\bar{f}^{(I_1,J^\star)}$ as defined above, it holds that $(I_1, J^\star) \models (\bigwedge \Sigma)[\bar{f} \mapsto \bar{f}^{(I_1,J^\star)}]$.

Let σ be a formula in Σ of the form $\varphi(\bar{x}) \rightarrow R(\bar{t}(\bar{x}))$, with R an n-ary relation symbol, and assume that $I_1 \models \varphi(\bar{a})$ for some tuple $\bar{a}$ of constant values. We need to show that $J^\star \models R(\bar{t}(\bar{a}))[\bar{f} \mapsto \bar{f}^{(I_1,J^\star)}]$. Now, since $I_1 \models \varphi(\bar{a})$ and $(I_1, J) \models (\bigwedge \Sigma)[\bar{f} \mapsto \bar{f}^{(I_1,J)}]$, we know that $J \models R(\bar{t}(\bar{a}))[\bar{f} \mapsto \bar{f}^{(I_1,J)}]$. By construction of Σ', there exists a formula σ' in Σ' of the form

$$R(\bar{u}_{\bar{t}(\bar{x})}) \wedge \mathbf{C}(\bar{u}') \rightarrow \alpha(\bar{u}_{\bar{t}(\bar{x})}),$$

that has been constructed from σ, with $\alpha(\bar{u}_{\bar{t}(\bar{x})})$ the disjunction of the formulas in $\mathcal{C}_{R(\bar{t}(\bar{x}))}$. Notice that, by the construction of $\bar{u}_{\bar{t}(\bar{x})}$, we obtain that J satisfies $R(\bar{u}_{\bar{t}(\bar{x})})$ with the assignment $\bar{t}(\bar{a})[\bar{f} \mapsto \bar{f}^{(I_1,J)}]$ to $\bar{u}_{\bar{t}(\bar{x})}$. Let $\bar{a}'$ be the corresponding assignment to $\bar{u}'$ that derives from the assignment of $\bar{t}(\bar{a})[\bar{f} \mapsto \bar{f}^{(I_1,J)}]$ to $\bar{u}_{\bar{t}(\bar{x})}$. By the construction of σ' and since $\bar{a}$

is a tuple of constant values, it is straightforward to see that $\mathbf{C}(\bar{a}')$ holds. Then we have that $J \models R(\bar{t}(\bar{a}))[\bar{f} \mapsto \bar{f}^{(I_1,J)}] \wedge \mathbf{C}(\bar{a}')$. Moreover, since $(J, I_2) \models (\bigwedge \Sigma')[\bar{f}' \mapsto \bar{f}'^{(J,I_2)}]$, we obtain that $I_2 \models \alpha(\bar{t}(\bar{a})[\bar{f} \mapsto \bar{f}^{(I_1,J)}])[\bar{f}' \mapsto \bar{f}'^{(J,I_2)}]$. From this last fact we conclude that there exists a disjunction $\beta(\bar{u}_{\bar{t}(\bar{x})})$ of $\alpha(\bar{u}_{\bar{t}(\bar{x})})$ of the form $\exists \bar{y}\big(\ \psi(\bar{y}) \ \wedge\ \nu(\bar{u}_{\bar{t}(\bar{x})}, \bar{s}(\bar{y}))\ \wedge\ \omega(\bar{u}_{\bar{t}(\bar{x})}, \bar{s}(\bar{y}))\ \big)$, and a tuple $\bar{b}$ of constant values such that

$$I_2 \models \psi(\bar{b}) \ \wedge\ \Big(\ \nu(\bar{t}(\bar{a})[\bar{f} \mapsto \bar{f}^{(I_1,J)}], \bar{s}(\bar{b}))\ \wedge\ \omega(\bar{t}(\bar{a})[\bar{f} \mapsto \bar{f}^{(I_1,J)}], \bar{s}(\bar{b}))\ \Big)[\bar{f}' \mapsto \bar{f}'^{(J,I_2)}].$$

By the construction of the formula $\alpha(\bar{u}_{\bar{t}(\bar{x})})$, we know that $\beta(\bar{u}_{\bar{t}(\bar{x})})$ belongs to the set $\mathcal{C}_{R(\bar{t}(\bar{x}))}$. Thus, there exists a formula $\psi(\bar{y}) \ \rightarrow\ R(\bar{s}(\bar{y}))$ in Σ such that $\bar{s}(\bar{y})$ subsumes $\bar{t}(\bar{x})$. Notice that $I_2 \models \psi(\bar{b})$, and then since $(I_2, J^\star) \models (\bigwedge \Sigma)[\bar{f} \mapsto \bar{f}^{(I_2,J^\star)}]$, we know that $J^\star \models R(\bar{s}(\bar{b}))[\bar{f} \mapsto \bar{f}^{(I_2,J^\star)}]$. We show next that $R(\bar{s}(\bar{b}))[\bar{f} \mapsto \bar{f}^{(I_2,J^\star)}]$ is equal to $R(\bar{t}(\bar{a}))[\bar{f} \mapsto \bar{f}^{(I_1,J^\star)}]$, and then we obtain that $J^\star \models R(\bar{t}(\bar{a}))[\bar{f} \mapsto \bar{f}^{(I_1,J^\star)}]$, which is exactly what we want to prove.

Let $\bar{u}_{\bar{t}(\bar{x})} = (u_1, \ldots, u_n)$, $\bar{t}(\bar{x}) = (t_1(\bar{x}), \ldots, t_n(\bar{x}))$, and $\bar{s}(\bar{y}) = (s_1(\bar{y}), \ldots, s_n(\bar{y}))$. We show now that, for every i such that $1 \leq i \leq n$, it holds that $t_i(\bar{a})[\bar{f} \mapsto \bar{f}^{(I_1,J^\star)}] = s_i(\bar{b})[\bar{f} \mapsto \bar{f}^{(I_2,J^\star)}]$. First, assume that $s_i(\bar{y})$ is a variable y_m from $\bar{y}$, and b_m the (constant) value that corresponds to y_m in the assignment of $\bar{b}$ to $\bar{y}$. Notice that formula $\nu(\bar{u}_{\bar{t}(\bar{x})}, \bar{s}(\bar{y}))$ contains then equality $u_i = y_m$. Then since $\nu(\bar{t}(\bar{a})[\bar{f} \mapsto \bar{f}^{(I_1,J)}], \bar{s}(\bar{b}))[\bar{f}' \mapsto \bar{f}'^{(J,I_2)}]$ holds, we obtain that $t_i(\bar{a})[\bar{f} \mapsto \bar{f}^{(I_1,J^\star)}] = b_m$. Thus, we have that $t_i(\bar{a})[\bar{f} \mapsto \bar{f}^{(I_1,J^\star)}] = s_i(\bar{b})[\bar{f} \mapsto \bar{f}^{(I_2,J^\star)}]$. Now suppose that $\bar{s}_i(\bar{y})$ is a non-atomic term $g(y_{m_1}, \ldots, y_{m_k})$, with g a k-ary function symbol in $\bar{f}$ and $(y_{m_1}, \ldots, y_{m_k})$ a k-tuple of variables from $\bar{y}$. Then since $\bar{s}(\bar{y})$ subsumes $\bar{t}(\bar{x})$, it holds that $t_i(\bar{x})$ is a non-atomic term $f(x_{r_1}, \ldots, x_{r_\ell})$, with f an ℓ-ary function symbol in $\bar{f}$ and $(x_{r_1}, \ldots, x_{r_\ell})$ an ℓ-tuple of variables from $\bar{x}$. Notice that formula $\nu(\bar{u}_{\bar{t}(\bar{x})}, \bar{s}(\bar{y}))$ contains, for every j such that $1 \leq j \leq k$, the equality $y_{m_j} = g_j(u_i)$ as a conjunction, with g_j a unary function symbol in $\bar{f}'$. We also know that formula $\nu(\bar{t}(\bar{a})[\bar{f} \mapsto \bar{f}^{(I_1,J)}], \bar{s}(\bar{b}))[\bar{f}' \mapsto \bar{f}'^{(J,I_2)}]$ holds. Then for every j such that $1 \leq j \leq k$, we have that $b_{m_j} = \big(g_j(t_i(\bar{a})[\bar{f} \mapsto \bar{f}^{(I_1,J)}])\big)[\bar{f}' \mapsto \bar{f}'^{(J,I_2)}]$. Thus, since we are

assuming that $t_i(\bar{x}) = f(x_{r_1}, \ldots, x_{r_\ell})$, we know that the following equalities hold:

$$\begin{aligned} b_{m_1} &= g_1^{(J,I_2)}\big(f^{(I_1,J)}(a_{r_1}, \ldots, a_{r_\ell})\big), \\ &\vdots \\ b_{m_k} &= g_k^{(J,I_2)}\big(f^{(I_1,J)}(a_{r_1}, \ldots, a_{r_\ell})\big). \end{aligned} \tag{5.11}$$

Now, focus on the formula $\omega(\bar{u}_{\bar{t}(\bar{x})}, \bar{s}(\bar{y}))$. Since $\bar{s}_i(\bar{y}) = g(y_{m_1}, \ldots, y_{m_k})$, we know that $\omega(\bar{u}_{\bar{t}(\bar{x})}, \bar{s}(\bar{y}))$ contains the equality $f_\star(u_i) = g_1(u_i)$, and the inequalities $f_\star(u_i) \neq h_1(u_i)$, for every h in $\bar{f}$ different from g. Then since we know that formula $\omega(\bar{t}(\bar{a})[\bar{f} \mapsto \bar{f}^{(I_1,J)}], \bar{s}(\bar{b}))[\bar{f}' \mapsto \bar{f}'^{(J,I_2)}]$ holds, we obtain that

$$f_\star^{(J,I_2)}\big(f^{(I_1,J)}(a_{r_1}, \ldots, a_{r_\ell})\big) = g_1^{(J,I_2)}\big(f^{(I_1,J)}(a_{r_1}, \ldots, a_{r_\ell})\big), \tag{5.12}$$

and for every h in $\bar{f}$ different from g,

$$f_\star^{(J,I_2)}\big(f^{(I_1,J)}(a_{r_1}, \ldots, a_{r_\ell})\big) \neq h_1^{(J,I_2)}\big(f^{(I_1,J)}(a_{r_1}, \ldots, a_{r_\ell})\big).$$

Notice then that g_1 is the unique function in $\bar{f}$ that satisfies (5.12). Then by the construction of $\bar{f}^{(I_1,J^\star)}$ we know that

$$f^{(I_1,J^\star)}(a_{r_1}, \ldots, a_{r_\ell}) = g^{(I_2,J^\star)}\Big(g_1^{(J,I_2)}\big(f^{(I_1,J)}(a_{r_1}, \ldots, a_{r_\ell})\big), \ \ldots, \ g_k^{(J,I_2)}\big(f^{(I_1,J)}(a_{r_1}, \ldots, a_{r_\ell})\big)\Big).$$

By replacing the equalities in (5.11) in this last expression we obtain that

$$f^{(I_1,J^\star)}(a_{r_1}, \ldots, a_{r_\ell}) = g^{(I_2,J^\star)}(b_{m_1}, \ldots, b_{m_k}).$$

Notice $s_i(\bar{b}) = g(b_{m_1}, \ldots, b_{m_k})$, and $t_i(\bar{a}) = f(a_{r_1}, \ldots, a_{r_\ell})$, thus we have that $t_i(\bar{a})[\bar{f} \mapsto \bar{f}^{(I_1,J^\star)}] = s_i(\bar{b})[\bar{f} \mapsto \bar{f}^{(I_2,J^\star)}]$. We have shown that, for every i such that $1 \leq i \leq n$, it holds that $t_i(\bar{a})[\bar{f} \mapsto \bar{f}^{(I_1,J^\star)}] = s_i(\bar{b})[\bar{f} \mapsto \bar{f}^{(I_2,J^\star)}]$. Thus we have that $R(\bar{t}(\bar{a}))[\bar{f} \mapsto \bar{f}^{(I_1,J^\star)}]$ is equal to $R(\bar{s}(\bar{b}))[\bar{f} \mapsto \bar{f}^{(I_2,J^\star)}]$. Then since $J^\star \models R(\bar{s}(\bar{b}))[\bar{f} \mapsto \bar{f}^{(I_2,J^\star)}]$ we know that $J^\star \models R(\bar{t}(\bar{a}))[\bar{f} \mapsto \bar{f}^{(I_1,J^\star)}]$.

What we have proved is that, for every formula $\varphi(\bar{x}) \rightarrow R(\bar{t}(\bar{x}))$ in Σ, if $I_1 \models \varphi(\bar{a})$, then $J^\star \models R(\bar{t}(\bar{a}))[\bar{f} \mapsto \bar{f}^{(I_1,J^\star)}]$. Thus we have that $(I_1, J^\star) \models (\bigwedge \Sigma)[\bar{f} \mapsto \bar{f}^{(I_1,J^\star)}]$, and then $(I_1, J^\star) \models \exists \bar{f} \bigwedge \Sigma$. This concludes the proof of the theorem. □

Since mappings specified by plain SO-tgds are total and closed-down on the left, from the results in Section 3.1.1, we obtain the following corollary.

COROLLARY 5.3.2. *Let $\mathcal{M}$ be a mapping specified by a plain SO-tgd λ. If $\mathcal{M}$ has a Fagin-inverse (quasi-inverse), then algorithm* POLYSOINV($\mathcal{M}$) *computes in polynomial time a Fagin-inverse (quasi-inverse) of $\mathcal{M}$.*

It is important to notice that since every set of st-tgds can be transformed into a plain SO-tgd in linear time, our algorithm can be used to compute Fagin-inverses, quasi-inverses, and maximum recoveries for st-mappings specified by sets of st-tgds. This is the first polynomial time algorithm capable of doing this. However, the gain in time complexity comes with the price of a stronger and less manageable mapping language for expressing inverses.

6. INFORMATION AND REDUNDANCY IN SCHEMA MAPPINGS

Schema mapping management is an area of active research, where there had been many achievements in the recent years. In fact different proposals for several schema mapping management operators are being studied and implemented. Nevertheless, little research has being pursued towards understanding the fundamental notions that all these proposals seem to share. In particular, abstract notions of *information*, *redundancy* and *minimality* are part of almost every proposal for the semantics of schema mapping operators (Bernstein, 2003; Pottinger & Bernstein, 2003; Melnik, 2004; Fagin, 2007; Arenas, Pérez, & Riveros, 2009; Pottinger & Bernstein, 2008). The main goal of this chapter is to formalize and study abstract notions of information and redundancy for schema mappings.

6.1. Transferring Source Information: The Order $\preceq_{\mathrm{s}}$

In a data exchange scenario, a schema mapping is used to transfer information from a source schema to a target schema. Thus, it is natural to ask how much source information a mapping transfers and, in particular, if two mappings are used to transfer information from the same source schema, it is natural to ask whether one of them transfers *more* source information than the other. The latter question is the main motivation of this section.

The problem of measuring the amount of source information transferred by a mapping has been studied in the data exchange scenario (Fagin et al., 2009) and we have also implicitly discussed this issue in Chapters 3 and 4. In fact, the issue of developing an order for comparing the amount of source information transferred by two mappings has been explicitly considered by Fagin et al. (2009). However, we follow here a different approach to develop such an order, as we first identity five natural conditions that such an order should satisfy, and then we consider the strictest order according to these conditions.

From now on, we use symbol $\preceq$ to denote an order between mappings that transfer information from the same schema. That is, if $\mathcal{M}_1 \preceq \mathcal{M}_2$, then we assume that there exists a schema $\mathbf{R}$ such that both $\mathrm{dom}(\mathcal{M}_1)$ and $\mathrm{dom}(\mathcal{M}_2)$ are contained in $\mathrm{Inst}(\mathbf{R})$. The following is the first condition that we impose on $\preceq$.

(C1) $\mathcal{M}_1 \preceq \mathcal{M}_2$ implies $\mathrm{dom}(\mathcal{M}_1) \subseteq \mathrm{dom}(\mathcal{M}_2)$.

If I is an instance in the domain of $\mathcal{M}_1$, then $\mathcal{M}_1$ provides some information about I as it gives a collection of target instances that are considered to be valid translations of I. Thus, if $\mathcal{M}_2$ gives as much source information as $\mathcal{M}_1$, then I should also be in the domain of $\mathcal{M}_2$, as stated by condition (C1).

As usual for any notion of preference, $\preceq$ is also asked to be reflexive and transitive:

(C2) $\mathcal{M} \preceq \mathcal{M}$,

(C3) $\mathcal{M}_1 \preceq \mathcal{M}_2$ and $\mathcal{M}_2 \preceq \mathcal{M}_3$ implies $\mathcal{M}_1 \preceq \mathcal{M}_3$.

Notice that we do not ask relation $\preceq$ to be antisymmetric, as it is usually the case that the same information can be transferred in different ways. Thus, strictly speaking, $\preceq$ is not an order but a preorder.

Furthermore, let $\mathrm{Id}_{\mathbf{R}}$ be the identity schema mapping for a schema $\mathbf{R}$, that is, $\mathrm{Id}_{\mathbf{R}} = \{(I, I) \mid I \in \mathrm{Inst}(\mathbf{R})\}$. This mapping transfers exactly the information that is contained in the instances of $\mathbf{R}$ and, thus, any other mapping that transfers information from $\mathbf{R}$ could not be more informative than $\mathrm{Id}_{\mathbf{R}}$. This gives rise to the fourth condition for the desired order:

(C4) if $\mathcal{M}$ is a mapping from a schema $\mathbf{R}$ to a schema $\mathbf{R}_1$, then $\mathcal{M} \preceq \mathrm{Id}_{\mathbf{R}}$.

Finally, our last condition accounts for the information that is transferred through a composition of schema mappings. Assume that a mapping $\mathcal{M}$ transfers information from a schema $\mathbf{R}$ to a schema $\mathbf{R}_1$ and that $\mathcal{M}_1$, $\mathcal{M}_2$ are mappings that transfer information from $\mathbf{R}_1$. If $\mathcal{M}_2$ maps as much source information as $\mathcal{M}_1$, then given that $\mathcal{M}$ transfers information to schema $\mathbf{R}_1$, one would expect that $\mathcal{M} \circ \mathcal{M}_2$ transfers as much source information as $\mathcal{M} \circ \mathcal{M}_1$. This is stated in our last condition:

(C5) let $\mathcal{M}$ be a mapping from a schema $\mathbf{R}$ to a schema $\mathbf{R}_1$, and $\mathcal{M}_1$, $\mathcal{M}_2$ mappings from $\mathbf{R}_1$ to schemas $\mathbf{R}_2$ and $\mathbf{R}_3$, respectively. If $\mathcal{M}_1 \preceq \mathcal{M}_2$, then $\mathcal{M} \circ \mathcal{M}_1 \preceq \mathcal{M} \circ \mathcal{M}_2$.

Now that we have identified five conditions that the desired order should satisfy, the first question to answer is whether there exists any order that meet them. In the following paragraphs, we give a positive answer to this question by introducing a relation $\preceq_{\mathrm{s}}$. Moreover, we also show that $\preceq_{\mathrm{s}}$ is the strictest order that satisfy the above conditions.

DEFINITION 6.1.1. **(Order $\preceq_{\mathrm{s}}$)** *Let $\mathbf{R}$, $\mathbf{R}_1$, $\mathbf{R}_2$ be schemas, and $\mathcal{M}_1$, $\mathcal{M}_2$ mappings from $\mathbf{R}$ to $\mathbf{R}_1$ and $\mathbf{R}$ to $\mathbf{R}_2$, respectively. Then $\mathcal{M}_1 \preceq_{\mathrm{s}} \mathcal{M}_2$ if there exists a mapping $\mathcal{N}$ from $\mathbf{R}_2$ to $\mathbf{R}_1$ such that $\mathcal{M}_1 = \mathcal{M}_2 \circ \mathcal{N}$.*

Intuitively, the preceding definition says that $\mathcal{M}_1 \preceq_{\mathrm{s}} \mathcal{M}_2$ if $\mathcal{M}_2$ transfers enough information from $\mathbf{R}$ to be able to reconstruct the information transferred by $\mathcal{M}_1$.

Example 6.1.2. Let $\mathcal{M}_1$ and $\mathcal{M}_2$ be mappings specified by dependencies $S(x, y) \rightarrow T(x)$ and $S(x, y) \rightarrow U(y, x)$, respectively. Intuitively, $\mathcal{M}_2$ maps more information than $\mathcal{M}_1$ as all the source information is stored in the target according to mapping $\mathcal{M}_2$. In fact, in this case we have that $\mathcal{M}_1 \preceq_{\mathrm{s}} \mathcal{M}_2$ since $\mathcal{M}_1 = \mathcal{M}_2 \circ \mathcal{N}$, where $\mathcal{N}$ is a mapping specified by dependency $U(x, y) \rightarrow T(y)$. In this case, it is also possible to prove that $\mathcal{M}_2 \not\preceq_{\mathrm{s}} \mathcal{M}_1$.

On the other hand, if $\mathcal{M}_3$ is a mapping specified by dependency $S(x, y) \rightarrow V(y)$, then one would expect $\mathcal{M}_1$ and $\mathcal{M}_3$ to be incomparable, as these mappings extract information from different columns of table S. In fact, in this case it is possible to prove that $\mathcal{M}_1 \not\preceq_{\mathrm{s}} \mathcal{M}_3$ and $\mathcal{M}_3 \not\preceq_{\mathrm{s}} \mathcal{M}_1$. □

As a corollary of the fact that the composition is associative, we obtain that $\preceq_{\mathrm{s}}$ satisfies the above conditions:

PROPOSITION 6.1.3. *The order $\preceq_{\mathrm{s}}$ satisfies (C1), (C2), (C3), (C4) and (C5).*

The following proposition shows the somewhat surprising result that such a simple relation is the strictest order that satisfies the above conditions.

PROPOSITION 6.1.4. *Assume that an order $\preceq$ satisfies (C1), (C2), (C3), (C4) and (C5). Then for every pair of mappings $\mathcal{M}_1$ and $\mathcal{M}_2$, $\mathcal{M}_1 \preceq_{\mathrm{s}} \mathcal{M}_2$ implies that $\mathcal{M}_1 \preceq \mathcal{M}_2$.*

PROOF. Let $\preceq$ be an order that satisfies (C1), (C2), (C3), (C4) and (C5). Furthermore, assume that $\mathcal{M}_1$ and $\mathcal{M}_2$ are mappings from a schema $\mathbf{R}$ to schemas $\mathbf{R}_1$ and $\mathbf{R}_2$. respectively. By definition of $\preceq_{\mathrm{s}}$, we know that there exists a mapping $\mathcal{N}$ from $\mathbf{R}_2$ to $\mathbf{R}_1$ such that $\mathcal{M}_1 = \mathcal{M}_2 \circ \mathcal{N}$. Then by (C2) we have that:

$$\mathcal{M}_1 \quad \preceq \quad \mathcal{M}_2 \circ \mathcal{N}. \tag{6.1}$$

But not only that, by (C4) we have that $\mathcal{N} \preceq \mathrm{Id}_{\mathbf{R}_2}$ and, therefore, we conclude by (C5) that:

$$\mathcal{M}_2 \circ \mathcal{N} \quad \preceq \quad \mathcal{M}_2 \circ \mathrm{Id}_{\mathbf{R}_2}. \tag{6.2}$$

Thus, given that $\mathcal{M}_2 \circ \mathrm{Id}_{\mathbf{R}_2} = \mathcal{M}_2$, we have by (C2) that $\mathcal{M}_2 \circ \mathrm{Id}_{\mathbf{R}_2} \preceq \mathcal{M}_2$ and, hence, from (6.2) and (C3), it holds that:

$$\mathcal{M}_2 \circ \mathcal{N} \quad \preceq \quad \mathcal{M}_2. \tag{6.3}$$

Finally, from (6.1), (6.3) and (C3), we conclude that $\mathcal{M}_1 \preceq \mathcal{M}_2$, which was to be shown. □

In our investigation, we use $\preceq_{\mathrm{s}}$ to compare the amount of information transferred by two mappings from the same source schema. In particular, if $\mathcal{M}_1 \preceq_{\mathrm{s}} \mathcal{M}_2$ and $\mathcal{M}_2 \preceq_{\mathrm{s}} \mathcal{M}_1$, we say that $\mathcal{M}_1$ and $\mathcal{M}_2$ transfer the same amount of source information, which is denoted by $\mathcal{M}_1 \equiv_{\mathrm{s}} \mathcal{M}_2$.

6.1.1. Comparison with other notions of order

Fagin et al. (2009) propose to use some notions of inversion of schema mappings (Fagin, 2007; Arenas, Pérez, & Riveros, 2009; Fagin et al., 2009) to measure the *information loss* of a mapping. Loosely speaking, the more invertible a mapping is, the less information the mapping loses (Fagin et al., 2009). In this section, we contrast and compare Fagin et al's approach with the order $\preceq_{\mathrm{s}}$.

In order to give some intuition behind the definitions presented by Fagin et al. (2009), we first introduce an order $\preceq_{\text{R}}$ that is based on the notion of maximum recovery introduced in Chapter 3. By the definition of maximum recovery, it is easy to see that if $\mathcal{M}_1^\star$ and $\mathcal{M}_2^\star$ are both maximum recoveries of $\mathcal{M}$, then $\mathcal{M} \circ \mathcal{M}_1^\star = \mathcal{M} \circ \mathcal{M}_2^\star$. Thus, the composition of a mapping $\mathcal{M}$ with any of its maximum recoveries depends only on $\mathcal{M}$. In fact, from Proposition 3.1.6 (part (3)) we can conclude that if $\mathcal{M}^\star$ is a maximum recovery of $\mathcal{M}$, then the composition $\mathcal{M} \circ \mathcal{M}^\star$ is equal to the set $\{(I, K) \mid \text{Sol}_{\mathcal{M}}(K) \subseteq \text{Sol}_{\mathcal{M}}(I)\}$. The definition of order $\preceq_{\text{R}}$ is based on this property. More precisely, a mapping $\mathcal{M}_2$ is said to be *less lossy than* a mapping $\mathcal{M}_1$ if for every pair of instances I, K, it holds that $\text{Sol}_{\mathcal{M}_1}(I) \subseteq \text{Sol}_{\mathcal{M}_1}(K)$ whenever $\text{Sol}_{\mathcal{M}_2}(I) \subseteq \text{Sol}_{\mathcal{M}_2}(K)$ holds (Fagin et al., 2009). Let $\preceq_{\text{R}}$ denote the order induced by this notion, that is, $\mathcal{M}_1 \preceq_{\text{R}} \mathcal{M}_2$ if and only if $\mathcal{M}_2$ is less lossy than $\mathcal{M}_1$.

It is important to notice that if mappings $\mathcal{M}_1$ and $\mathcal{M}_2$ have maximum recoveries, say $\mathcal{M}_1^\star$ and $\mathcal{M}_2^\star$, respectively, then $\mathcal{M}_1 \preceq_{\text{R}} \mathcal{M}_2$ if and only if $\text{Id}_{\mathbf{R}} \subseteq \mathcal{M}_2 \circ \mathcal{M}_2^\star \subseteq \mathcal{M}_1 \circ \mathcal{M}_1^\star$. Thus, $\mathcal{M}_1 \preceq_{\text{R}} \mathcal{M}_2$ if the composition of $\mathcal{M}_2$ with its maximum recovery is more similar to the identity mapping than the composition of $\mathcal{M}_1$ with its maximum recovery.

As a first result, we prove that $\preceq_{\text{R}}$ does not satisfy all the conditions identified in this section for a natural order, thus showing that $\preceq_{\text{S}}$ can be considered as a better alternative than $\preceq_{\text{R}}$ to compare the information transferred by schema mappings. In particular, $\preceq_{\text{R}}$ does not satisfy (C5).

PROPOSITION 6.1.5. *There exist mappings $\mathcal{M}$, $\mathcal{M}_1$ and $\mathcal{M}_2$ such that $\mathcal{M}_1 \preceq_{\text{R}} \mathcal{M}_2$ and $\mathcal{M} \circ \mathcal{M}_1 \not\preceq_{\text{R}} \mathcal{M} \circ \mathcal{M}_2$.*

PROOF. Let $\mathbf{R} = \{A(\cdot), B(\cdot)\}$, $\mathbf{S} = \{F(\cdot), G(\cdot), H(\cdot)\}$, $\mathbf{T}_1 = \{F'(\cdot), G'(\cdot), H'(\cdot)\}$ and $\mathbf{T}_2 = \{R(\cdot), S(\cdot), T(\cdot)\}$. Consider the mapping $\mathcal{M}$ from $\mathbf{R}$ to $\mathbf{S}$ given by:

$$\begin{aligned} A(x) &\rightarrow F(x) \vee G(x) \\ B(x) &\rightarrow F(x) \vee H(x) \end{aligned}$$

Let $\mathcal{M}_1$ be the mapping from $\mathbf{S}$ to $\mathbf{T}_1$ given by:

$$\begin{aligned} F(x) &\rightarrow F'(x) \\ G(x) &\rightarrow G'(x) \\ H(x) &\rightarrow H'(x) \end{aligned}$$

Let $\mathcal{M}_2$ be the mapping from $\mathbf{S}$ to $\mathbf{T}_2$ given by:

$$\begin{aligned} F(x) &\rightarrow R(x) \vee S(x) \\ G(x) &\rightarrow S(x) \vee T(x) \\ H(x) &\rightarrow T(x) \vee R(x) \end{aligned}$$

It was shown by Arenas, Pérez, and Riveros (2009, Proposition 6.6 (2)) that $\mathcal{M}_2$ satisfies the following property. For every I, K if $\mathrm{Sol}_{\mathcal{M}_2}(I) \subseteq \mathrm{Sol}_{\mathcal{M}_2}(K)$ then $K \subseteq I$. From this, it is straightforward to prove that $\mathrm{Sol}_{\mathcal{M}_1}(I) \subseteq \mathrm{Sol}_{\mathcal{M}_1}(K)$, and thus $\mathcal{M}_1 \preceq_{\mathrm{R}} \mathcal{M}_2$. We prove now that $\mathcal{M} \circ \mathcal{M}_1 \not\preceq_{\mathrm{R}} \mathcal{M} \circ \mathcal{M}_2$.

Consider the instances I such that $A^I = \{1\}$ and $B^I = \emptyset$ and K such that $A^K = \emptyset$ and $B^K = \{1\}$. We show next that $\mathrm{Sol}_{\mathcal{M} \circ \mathcal{M}_2}(I) = \mathrm{Sol}_{\mathcal{M} \circ \mathcal{M}_2}(K) = \{J \in \mathrm{Inst}(\mathbf{T}_2) \mid 1 \in R^J$ or $1 \in S^J$ or $1 \in T^J\}$. First, it is clear that if $(I, J) \in \mathcal{M} \circ \mathcal{M}_2$ then $1 \in R^J$ or $1 \in S^J$ or $1 \in T^J$, and similarly if $(K, J) \in \mathcal{M} \circ \mathcal{M}_2$ then $1 \in R^J$ or $1 \in S^J$ or $1 \in T^J$. Now assume that $J \in \mathrm{Inst}(\mathbf{T}_2)$ and $1 \in R^J$ or $1 \in S^J$ or $1 \in T^J$. Assume first that $1 \in R^J$ then consider the instance L such that $F^L = \{1\}$, $G^L = H^L = \emptyset$. Then we have that $(I, L) \in \mathcal{M}$ and $(L, J) \in \mathcal{M}_2$ which implies that $(I, J) \in \mathcal{M} \circ \mathcal{M}_2$. Similarly, if $1 \in S^J$ then we consider the instance L such that $F^L = \{1\}$, $G^L = H^L = \emptyset$ to obtain that $(I, L) \in \mathcal{M}$ and $(L, J) \in \mathcal{M}_2$. If $1 \in T^J$ then we consider the instance L such that $G^L = \{1\}$ and $F^L = H^L = \emptyset$. Then, again we obtain that $(I, L) \in \mathcal{M}$ and $(L, J) \in \mathcal{M}_2$, and thus, $(I, J) \in \mathcal{M} \circ \mathcal{M}_2$. By symmetry we can show that if $J \in \mathrm{Inst}(\mathbf{T}_2)$ and $1 \in R^J$ or $1 \in S^J$ or $1 \in T^J$ then $(K, J) \in \mathcal{M} \circ \mathcal{M}_2$. We have shown that $\mathrm{Sol}_{\mathcal{M} \circ \mathcal{M}_2}(I) = \mathrm{Sol}_{\mathcal{M} \circ \mathcal{M}_2}(K)$. Now we consider $\mathcal{M}_1$. Notice that the instance J such that $G'^J = \{1\}$ and $F'^J = H'^J = \emptyset$ is such that $(I, J) \in \mathcal{M} \circ \mathcal{M}_1$ but $(K, J) \notin \mathcal{M} \circ \mathcal{M}_1$, and then $\mathrm{Sol}_{\mathcal{M} \circ \mathcal{M}_1}(I) \not\subseteq \mathrm{Sol}_{\mathcal{M} \circ \mathcal{M}_1}(K)$.

Thus we have that $\mathrm{Sol}_{\mathcal{M}\circ\mathcal{M}_2}(I) \subseteq \mathrm{Sol}_{\mathcal{M}\circ\mathcal{M}_2}(K)$ but $\mathrm{Sol}_{\mathcal{M}\circ\mathcal{M}_1}(I) \not\subseteq \mathrm{Sol}_{\mathcal{M}\circ\mathcal{M}_1}(K)$ which implies that $\mathcal{M} \circ \mathcal{M}_1 \not\preceq_{\mathrm{R}} \mathcal{M} \circ \mathcal{M}_2$. □

Fagin et al. (2009) were interested in studying mappings with null values in source and target instances. In particular, for a mapping $\mathcal{M}$ of this type, Fagin et al. define a mapping $e(\mathcal{M})$ that extends $\mathcal{M}$ by giving a semantics to the nulls that distinguish them from the constants (see Definition 3.1.13 for the formalization of $e(\mathcal{M})$). The authors then introduce a notion of information loss of a schema mapping $\mathcal{M}$ by considering the extension $e(\mathcal{M})$ of $\mathcal{M}$. More precisely, if $\mathcal{M}_1$ and $\mathcal{M}_2$ are two mappings containing null values in source and target instances, then $\mathcal{M}_2$ is said to be less lossy than $\mathcal{M}_1$ if for every pair of instances I, K, it holds that $\mathrm{Sol}_{e(\mathcal{M}_1)}(I) \subseteq \mathrm{Sol}_{e(\mathcal{M}_1)}(K)$ whenever $\mathrm{Sol}_{e(\mathcal{M}_2)}(I) \subseteq \mathrm{Sol}_{e(\mathcal{M}_2)}(K)$ holds (Fagin et al., 2009). Let $\preceq_{\mathrm{E}}$ be the order induced by this notion. Notice that $\preceq_{\mathrm{E}}$ is tightly connected with $\preceq_{\mathrm{R}}$; in fact, it holds that $\mathcal{M}_1 \preceq_{\mathrm{E}} \mathcal{M}_2$ if and only if $e(\mathcal{M}_1) \preceq_{\mathrm{R}} e(\mathcal{M}_2)$. The following proposition shows that as for the case of $\preceq_{\mathrm{R}}$, the order $\preceq_{\mathrm{E}}$ does not satisfy (C5).

PROPOSITION 6.1.6. *There exist mappings $\mathcal{M}$, $\mathcal{M}_1$ and $\mathcal{M}_2$ such that $\mathcal{M}$, $\mathcal{M}_1$ and $\mathcal{M}_2$ contain null values in source and target instances, $\mathcal{M}_1 \preceq_{\mathrm{E}} \mathcal{M}_2$ and $\mathcal{M} \circ \mathcal{M}_1 \not\preceq_{\mathrm{E}} \mathcal{M} \circ \mathcal{M}_2$.*

PROOF. In this proof we suppose that every mapping $\mathcal{M}$ contains null values in the source. In this scenario, recall that the extended semantics of a mapping $\mathcal{M}$, denoted by $e(\mathcal{M})$, is the mapping $e(\mathcal{M}) = \rightarrow\ \circ\mathcal{M}\circ\ \rightarrow$, where $\rightarrow = \{(I_1, I_2) \mid I_1 \rightarrow I_2\}$ (see Definition 3.1.13 for more details on the extended semantics for mappings).

Let $\mathbf{R} = \{A(\cdot), B(\cdot)\}$, $\mathbf{S} = \{F(\cdot), G(\cdot), H(\cdot)\}$, $\mathbf{T}_1 = \{F'(\cdot), G'(\cdot), H'(\cdot)\}$ and $\mathbf{T}_2 = \{R(\cdot), S(\cdot), T(\cdot)\}$. Consider the mapping $\mathcal{M}$ from $\mathbf{R}$ to $\mathbf{S}$ given by:

$$\begin{aligned} A(x) &\rightarrow F(x) \vee G(x) \\ B(x) &\rightarrow F(x) \vee H(x) \end{aligned}$$

Let $\mathcal{M}_1$ be the mapping from $\mathbf{S}$ to $\mathbf{T}_1$ given by:

$$\begin{aligned} F(x) &\rightarrow F'(x) \\ G(x) &\rightarrow G'(x) \\ H(x) &\rightarrow H'(x) \end{aligned}$$

Let $\mathcal{M}_2$ be the mapping from $\mathbf{S}$ to $\mathbf{T}_2$ given by:

$$\begin{aligned} F(x) &\rightarrow R(x) \vee S(x) \\ G(x) &\rightarrow S(x) \vee T(x) \\ H(x) &\rightarrow T(x) \vee R(x) \end{aligned}$$

First of all, we will show that for every I, K if $\mathrm{Sol}_{e(\mathcal{M}_2)}(I) \subseteq \mathrm{Sol}_{e(\mathcal{M}_2)}(K)$ then $K \rightarrow I$. From this, it is straightforward to prove that $\mathrm{Sol}_{e(\mathcal{M}_1)}(I) \subseteq \mathrm{Sol}_{e(\mathcal{M}_1)}(K)$, and thus $\mathcal{M}_1 \preceq_{\mathrm{E}} \mathcal{M}_2$. So, let I and K be source instances. For the sake of contradiction, assume $\mathrm{Sol}_{e(\mathcal{M}_2)}(I) \subseteq \mathrm{Sol}_{e(\mathcal{M}_2)}(K)$ and $K \not\rightarrow I$. Then either $F^K \not\rightarrow F^I$, or $G^K \not\rightarrow G^I$, or $H^K \not\rightarrow H^I$. Assume first that $F^K \not\rightarrow F^I$. Then there exists an element $a \in \mathbf{C}$ such that $a \in F^K$ but $a \notin F^I$. If not, then $F^I = \emptyset$ and we have that $\mathrm{Sol}_{e(\mathcal{M}_2)}(I) \not\subseteq \mathrm{Sol}_{e(\mathcal{M}_2)}(K)$ which is a contradiction. So, let $a \in \mathbf{C}$ be an element such that $a \in F^K$ but $a \notin F^I$ and J a solution for I such that $R^J = F^I$, $S^J = \emptyset$ and $T^J = G^I \cup H^I$. Now, for every solution $J' \in \mathrm{Sol}_{e(\mathcal{M}_2)}(K)$, we have that $a \in R^{J'}$ or $a \in S^{J'}$. Thus, given that $a \notin R^J$ and $S^J = \emptyset$, we obtain that $J \notin \mathrm{Sol}_{e(\mathcal{M}_2)}(K)$, and then $\mathrm{Sol}_{e(\mathcal{M}_2)}(I) \not\subseteq \mathrm{Sol}_{e(\mathcal{M}_2)}(K)$, which contradicts our initial assumption. By using a similar argument, we can show that if $G^K \not\rightarrow G^I$ then $\mathrm{Sol}_{e(\mathcal{M}_2)}(I) \not\subseteq \mathrm{Sol}_{e(\mathcal{M}_2)}(K)$, and if $H^K \not\rightarrow H^I$ then $\mathrm{Sol}_{e(\mathcal{M}_2)}(K) \not\subseteq \mathrm{Sol}_{e(\mathcal{M}_2)}(I)$, which also lead to a contradiction.

We prove now that $\mathcal{M} \circ \mathcal{M}_1 \not\preceq_{\mathrm{E}} \mathcal{M} \circ \mathcal{M}_2$. Consider the instances I such that $A^I = \{1\}$ and $B^I = \emptyset$ and K such that $A^K = \emptyset$ and $B^K = \{1\}$. We show next that $\mathrm{Sol}_{e(\mathcal{M} \circ \mathcal{M}_2)}(I) = \mathrm{Sol}_{e(\mathcal{M} \circ \mathcal{M}_2)}(K) = \{J \in \mathrm{Inst}(\mathbf{T}_2) \mid 1 \in R^J \text{ or } 1 \in S^J \text{ or } 1 \in T^J\}$. First, it is clear that if $(I, J) \in e(\mathcal{M} \circ \mathcal{M}_2)$ then $1 \in R^J$ or $1 \in S^J$ or $1 \in T^J$, and similarly if $(K, J) \in e(\mathcal{M} \circ \mathcal{M}_2)$ then $1 \in R^J$ or $1 \in S^J$ or $1 \in T^J$. Now assume that $J \in \mathrm{Inst}(\mathbf{T}_2)$ and

$1 \in R^J$ or $1 \in S^J$ or $1 \in T^J$. Assume first that $1 \in R^J$ then consider the instance L such that $F^L = \{1\}$, $G^L = H^L = \emptyset$. Then we have that $(I, L) \in \mathcal{M}$ and $(L, J) \in \mathcal{M}_2$ which implies that $(I, J) \in e(\mathcal{M} \circ \mathcal{M}_2)$. Similarly, if $1 \in S^J$ then we consider the instance L such that $F^L = \{1\}$, $G^L = H^L = \emptyset$ to obtain that $(I, L) \in \mathcal{M}$ and $(L, J) \in \mathcal{M}_2$. If $1 \in T^J$ then we consider the instance L such that $G^L = \{1\}$ and $F^L = H^L = \emptyset$. Then, again we obtain that $(I, L) \in \mathcal{M}$ and $(L, J) \in \mathcal{M}_2$, and thus, $(I, J) \in e(\mathcal{M} \circ \mathcal{M}_2)$. By symmetry we can show that if $J \in \mathrm{Inst}(\mathbf{T}_2)$ and $1 \in R^J$ or $1 \in S^J$ or $1 \in T^J$ then $(K, J) \in e(\mathcal{M} \circ \mathcal{M}_2)$. We have shown that $\mathrm{Sol}_{e(\mathcal{M} \circ \mathcal{M}_2)}(I) = \mathrm{Sol}_{e(\mathcal{M} \circ \mathcal{M}_2)}(K)$. Now we consider $\mathcal{M}_1$. Notice that the instance J such that $G'^J = \{1\}$ and $F'^J = H'^J = \emptyset$ is such that $(I, J) \in e(\mathcal{M} \circ \mathcal{M}_1)$ but $(K, J) \notin e(\mathcal{M} \circ \mathcal{M}_1)$, and then $\mathrm{Sol}_{e(\mathcal{M} \circ \mathcal{M}_1)}(I) \not\subseteq \mathrm{Sol}_{e(\mathcal{M} \circ \mathcal{M}_1)}(K)$. Thus we have that $\mathrm{Sol}_{e(\mathcal{M} \circ \mathcal{M}_2)}(I) \subseteq \mathrm{Sol}_{e(\mathcal{M} \circ \mathcal{M}_2)}(K)$ but $\mathrm{Sol}_{e(\mathcal{M} \circ \mathcal{M}_1)}(I) \not\subseteq \mathrm{Sol}_{e(\mathcal{M} \circ \mathcal{M}_1)}(K)$ which implies that $\mathcal{M} \circ \mathcal{M}_1 \not\preceq_{\mathrm{E}} \mathcal{M} \circ \mathcal{M}_2$. □

No restrictions on mappings were imposed when defining the order $\preceq_{\mathrm{S}}$. In particular, $\preceq_{\mathrm{S}}$ can be used to compare mappings containing null values in source and target instances. Thus, Proposition 6.1.6 gives evidence that $\preceq_{\mathrm{S}}$ is a better alternative than $\preceq_{\mathrm{E}}$ to compare the information transferred by schema mappings.

It should be pointed out that Fagin et al. (2009) introduce the order $\preceq_{\mathrm{E}}$ but do not study its fundamental properties. Interestingly, the following result shows that $\preceq_{\mathrm{S}}$, $\preceq_{\mathrm{R}}$ and $\preceq_{\mathrm{E}}$ coincide for the class of st-mappings (mappings not containing null values in source instances) that are specified by st-tgds. Thus, the machinery developed in this paper for $\preceq_{\mathrm{S}}$ can also be used for $\preceq_{\mathrm{E}}$ and $\preceq_{\mathrm{R}}$ in this case.

PROPOSITION 6.1.7. *Let $\mathcal{M}_1 = (\mathbf{S}, \mathbf{T}_1, \Sigma_1)$ and $\mathcal{M}_2 = (\mathbf{S}, \mathbf{T}_2, \Sigma_2)$ be st-mappings, where Σ_1, Σ_2 are sets of st-tgds. Then the following statements are equivalent:*

(1) $\mathcal{M}_1 \preceq_{\mathrm{S}} \mathcal{M}_2$.

(2) $\mathcal{M}_1 \preceq_{\mathrm{R}} \mathcal{M}_2$.

(3) $\mathcal{M}_1 \preceq_{\mathrm{E}} \mathcal{M}_2$.

Notice that by the results in Section 3.1.1 we know that every st-mappings specified by st-tgds has a maximum recovery. Moreover, it is easy to prove that if $\mathcal{M}$ is an st-mapping specified by a set of st-tgds, then $\mathrm{Sol}_{e(\mathcal{M})}(I) = \mathrm{Sol}_{\mathcal{M}}(I)$ for every I (Fagin et al., 2009). Thus, the proof of the above proposition follows from the next result.

PROPOSITION 6.1.8. *Let $\mathcal{M}_1$ and $\mathcal{M}_2$ be mappings that have maximum recovery such that $\mathrm{dom}(\mathcal{M}_1) \subseteq \mathrm{dom}(\mathcal{M}_2)$. The following statements are equivalent.*

(1) $\mathcal{M}_1 \preceq_{\mathrm{s}} \mathcal{M}_2$.

(2) *For every pair of instances I, K, if $\mathrm{Sol}_{\mathcal{M}_2}(I) \subseteq \mathrm{Sol}_{\mathcal{M}_2}(K)$ then $\mathrm{Sol}_{\mathcal{M}_1}(I) \subseteq \mathrm{Sol}_{\mathcal{M}_1}(K)$.*

PROOF. We prove first that (1) $\Rightarrow$ (2). Assume that there exists a mapping $\mathcal{N}$ such that $\mathcal{M}_1 = \mathcal{M}_2 \circ \mathcal{N}$. Then suppose that $\mathrm{Sol}_{\mathcal{M}_2}(I) \subseteq \mathrm{Sol}_{\mathcal{M}_2}(K)$ for some instance I, K. If we compose $\mathcal{M}_2$ and $\mathcal{N}$, then we have that $\mathrm{Sol}_{\mathcal{M}_2 \circ \mathcal{N}}(I) \subseteq \mathrm{Sol}_{\mathcal{M}_2 \circ \mathcal{N}}(K)$. Furthermore, given that $\mathcal{M}_1 = \mathcal{M}_2 \circ \mathcal{N}$, this implies that $\mathrm{Sol}_{\mathcal{M}_1}(I) \subseteq \mathrm{Sol}_{\mathcal{M}_1}(K)$.

We show now that (2) $\Rightarrow$ (1). Assume that for every pair of instances I, K, if $\mathrm{Sol}_{\mathcal{M}_2}(I) \subseteq \mathrm{Sol}_{\mathcal{M}_2}(K)$ then $\mathrm{Sol}_{\mathcal{M}_1}(I) \subseteq \mathrm{Sol}_{\mathcal{M}_1}(K)$. We need to show that there exists a mapping $\mathcal{N}$ such that $\mathcal{M}_1 = \mathcal{M}_2 \circ \mathcal{N}$. Given that $\mathcal{M}_2$ has maximum recovery, let $\mathcal{M}_2^\star$ be a maximum recovery of $\mathcal{M}_2$. We claim that $\mathcal{M}_1 = \mathcal{M}_2 \circ (\mathcal{M}_2^\star \circ \mathcal{M}_1)$.

Since $\mathcal{M}_2^\star$ is a recovery of $\mathcal{M}_2$, then it holds that $(I, I) \in \mathcal{M}_2 \circ \mathcal{M}_2^\star$ for every instance $I \in \mathrm{dom}(\mathcal{M}_2)$. By hypothesis, we know that $\mathrm{dom}(\mathcal{M}_1) \subseteq \mathrm{dom}(\mathcal{M}_2)$ and then, for every pair $(I, J) \in \mathcal{M}_1$ we have that $(I, J) \in \mathcal{M}_2 \circ \mathcal{M}_2^\star \circ \mathcal{M}_1$. This implies that $\mathcal{M}_1 \subseteq \mathcal{M}_2 \circ (\mathcal{M}_2^\star \circ \mathcal{M}_1)$.

So, it only left to show that $\mathcal{M}_2 \circ (\mathcal{M}_2^\star \circ \mathcal{M}_1) \subseteq \mathcal{M}_1$. Given that $\mathcal{M}_2^\star$ is a maximum recovery of $\mathcal{M}_2$, we know by Proposition 3.1.6 that for every pair of instances I and K, if $I \in \mathrm{Sol}_{\mathcal{M}_2 \circ \mathcal{M}_2^\star}(K)$ then $\mathrm{Sol}_{\mathcal{M}_2}(I) \subseteq \mathrm{Sol}_{\mathcal{M}_2}(K)$. By hypothesis, we know that if $\mathrm{Sol}_{\mathcal{M}_2}(I) \subseteq \mathrm{Sol}_{\mathcal{M}_2}(K)$ then $\mathrm{Sol}_{\mathcal{M}_1}(I) \subseteq \mathrm{Sol}_{\mathcal{M}_1}(K)$. Thus, for every instance $I \in \mathrm{Sol}_{\mathcal{M}_2 \circ \mathcal{M}_2^\star}(K)$ we have that $\mathrm{Sol}_{\mathcal{M}_1}(I) \subseteq \mathrm{Sol}_{\mathcal{M}_1}(K)$. We conclude that $\mathrm{Sol}_{\mathcal{M}_2 \circ \mathcal{M}_2^\star \circ \mathcal{M}_1}(K) \subseteq \mathrm{Sol}_{\mathcal{M}_1}(K)$ for every instance K, that is, $\mathcal{M}_2 \circ (\mathcal{M}_2^\star \circ \mathcal{M}_1) \subseteq \mathcal{M}_1$. This was to be shown. $\square$

6.2. Fundamental Properties of the Order $\preceq_{\mathrm{s}}$

In this section, we provide a characterization of the order $\preceq_{\mathrm{s}}$ that gives evidence of why it is appropriate to compare the amount of source information being transferred by two mappings. Furthermore, we use this characterization to provide an algorithm that given st-mappings $\mathcal{M}_1$ and $\mathcal{M}_2$ specified by CQ$^{\neq}$-TO-CQ dependencies and such that $\mathcal{M}_1 \preceq_{\mathrm{s}} \mathcal{M}_2$, computes a mapping $\mathcal{N}$ such that $\mathcal{M}_1 = \mathcal{M}_2 \circ \mathcal{N}$.

6.2.1. Characterizing the order $\preceq_{\mathrm{s}}$

We present a characterization of the order $\preceq_{\mathrm{s}}$ for mappings given by FO-TO-CQ dependencies, that is based on query rewriting. Given a mapping $\mathcal{M}$ from a schema $\mathbf{S}$ to a schema $\mathbf{T}$ and a query Q over $\mathbf{S}$, we say that Q is *target rewritable* under $\mathcal{M}$ if there exists a query Q' over $\mathbf{T}$ which is a target rewriting of Q under $\mathcal{M}$, that is, for every instance I of $\mathbf{S}$, it holds that $Q(I) = \mathrm{certain}_{\mathcal{M}}(Q', I)$.

Example 6.2.1. Let $\mathbf{S} = \{A(\cdot,\cdot), B(\cdot)\}$ and $\mathbf{T} = \{P(\cdot,\cdot), T(\cdot)\}$, and let $\mathcal{M}$ be the st-mapping from $\mathbf{S}$ to $\mathbf{T}$ specified by dependencies

$$\begin{aligned} A(x,y) &\rightarrow P(x,y), \\ B(x) &\rightarrow P(x,x), \\ A(x,x) &\rightarrow R(x). \end{aligned}$$

Consider the query $Q_1(x,y)$ over $\mathbf{S}$ given by formula $A(x,y)$, and consider the following query $Q_1'(x,y)$ in UCQ$^{=,\neq}$ over $\mathbf{T}$:

$$(P(x,y) \ \wedge \ x \neq y) \ \vee \ (R(x) \ \wedge \ x = y).$$

It can be shown that for every instance I of $\mathbf{S}$ it holds that $Q_1(I) = \mathrm{certain}_{\mathcal{M}}(Q_1', I)$, and thus Q_1' is a target rewriting of Q_1 under $\mathcal{M}$. (It can also be shown that equalities, inequalities and disjunctions are strictly necessary to specify a target rewriting of Q_1 under $\mathcal{M}$.) Thus, we have then that $Q_1(x,y)$ is target rewritable under $\mathcal{M}$. On the other hand, the Boolean query Q_2 given by $\exists x \ B(x)$, is not target rewritable under $\mathcal{M}$. One might be

tempted to consider the query Q'_2 given by $\exists x\, P(x, x)$ as candidate rewriting. Nevertheless, Q'_2 is not a target rewriting of Q_2 since, for example, for the instance $I = \{A(1, 1)\}$ we have that $Q_2(I) = \underline{\text{false}}$ while $Q'_2(I) = \underline{\text{true}}$. □

Notice that, intuitively, a query Q is target rewritable under a mapping $\mathcal{M}$ if $\mathcal{M}$ transfers enough source information to be able to answer Q by using only the target data. Thus, the amount of source information transferred by two mappings can be compared in terms of the queries that are target rewritable under them. In fact, as the following result shows, this idea can be used to characterize the order $\preceq_{\text{s}}$.

THEOREM 6.2.2. *Let* $\mathcal{M}_1 = (\mathbf{S}, \mathbf{T}_1, \Sigma_1)$ *and* $\mathcal{M}_2 = (\mathbf{S}, \mathbf{T}_2, \Sigma_2)$ *be st-mappings, where* Σ_1, Σ_2 *are sets of* FO-TO-CQ *dependencies. Then the following statements are equivalent:*

(1) $\mathcal{M}_1 \preceq_{\text{s}} \mathcal{M}_2$.

(2) *For every query* Q *over* $\mathbf{S}$*, if* Q *is target rewritable under* $\mathcal{M}_1$*, then* Q *is target rewritable under* $\mathcal{M}_2$.

It is important to notice that the preceding theorem considers the class of *all queries*, as defined in Section 2.2. Thus, Theorem 6.2.2 gives strong evidence in favor of the order $\preceq_{\text{s}}$. To show the theorem we need to introduce some terminology and state two intermediate results.

Given instances I_1 and I_2 over a schema $\mathbf{S}$ and a set $\mathcal{C} = \{Q_1, \ldots, Q_n\}$ of queries over $\mathbf{S}$, we use $\mathcal{C}(I_1) \subseteq \mathcal{C}(I_2)$ to denote that $Q_i(I_1) \subseteq Q_i(I_2)$ for every $i \in \{1, \ldots, n\}$. Let Q be an arbitrary query over $\mathbf{S}$. We say that $\mathcal{C}$ *strongly determines* Q, and write $\mathcal{C} \Rightarrow Q$, when for every pair of instances I_1 and I_2 of $\mathbf{S}$ if $\mathcal{C}(I_1) \subseteq \mathcal{C}(I_2)$ then $Q(I_1) \subseteq Q(I_2)$. The notion of strong determination is closely related with the notion of determination of a query given a set of views introduced by Segoufin and Vianu (2005) (see the proof of Lemma 6.2.3 where our notion of strong determinacy is formulated in terms of views and quey rewriting using views).

We now use strong determination to characterize the notion of target rewritability. Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ be an st-mapping where Σ is a set of FO-TO-CQ st-dependencies. Consider the set of queries $\mathcal{C}_{\mathcal{M}}$ constructed in Lemma 4.2.3. That is, $\mathcal{C}_{\mathcal{M}}$ is a set of queries such that for every dependency of the form $\varphi(\bar{x}) \rightarrow \psi(\bar{x})$ in Σ the set $\mathcal{C}_{\mathcal{M}}$ contains a query $\chi(\bar{x})$ that is a rewriting of $\psi(\bar{x})$ over the source schema $\mathbf{S}$. Notice that such a rewriting always exists and can be expressed as an FO query (see Lemma 3.3.1). Furthermore, if Σ is a set of st-tgds, then the rewriting of $\psi(\bar{x})$ over the source can be expressed as a query in $\text{UCQ}^{=}$ (see Lemma 3.3.3). The next result shows that strong determination can be used to characterize target rewritablity of queries.

LEMMA 6.2.3. *Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ be an st-mapping where Σ is a set of* FO-TO-CQ *st-dependencies, and Q an arbitrary query over $\mathbf{S}$. Then Q is target rewritable under $\mathcal{M}$ iff $\mathcal{C}_{\mathcal{M}} \Rightarrow Q$.*

The proof this lemma is very involved and uses some additional notions and techniques from query rewriting using views. This proof can be found in Appendix A.2. Before stating the next lemma we need to introduce some additional notation. For sets of queries $\mathcal{C}_1$ and $\mathcal{C}_2$ we say that $\mathcal{C}_1$ strongly determines $\mathcal{C}_2$, and write $\mathcal{C}_1 \Rightarrow \mathcal{C}_2$, when for every query $Q \in \mathcal{C}_2$ it holds that $\mathcal{C}_1 \Rightarrow Q$.

LEMMA 6.2.4. *Let $\mathcal{M}_1 = (\mathbf{S}, \mathbf{T}_1, \Sigma_1)$ and $\mathcal{M}_2 = (\mathbf{S}, \mathbf{T}_2, \Sigma_2)$ be st-mappings where Σ_1 and Σ_2 are sets of* FO-TO-CQ *st-dependencies. Then $\mathcal{M}_1 \preceq_{\text{s}} \mathcal{M}_2$ iff $\mathcal{C}_{\mathcal{M}_2} \Rightarrow \mathcal{C}_{\mathcal{M}_1}$.*

Lemma 6.2.4 follows from Lemma 4.2.3 and the characterization of the order $\preceq_{\text{s}}$ proved in Proposition 6.1.8. The proof can be found in Appendix A.2. By using Lemmas 6.2.3 and 6.2.4, we can provide a very simple proof for Theorem 6.2.2.

PROOF OF THEOREM 6.2.2. Assume first that $\mathcal{M}_1 \preceq_{\text{s}} \mathcal{M}_2$ and let Q be a target-rewritable query under $\mathcal{M}_1$. We need to prove that Q is target rewritable under $\mathcal{M}_2$. Since $\mathcal{M}_1 \preceq_{\text{s}} \mathcal{M}_2$ from Lemma 6.2.4 we know that $\mathcal{C}_{\mathcal{M}_2} \Rightarrow \mathcal{C}_{\mathcal{M}_1}$. Now, if Q is target-rewritable query under $\mathcal{M}_1$ from Lemma 6.2.3 we know that $\mathcal{C}_{\mathcal{M}_1} \Rightarrow Q$. Finally, since $\mathcal{C}_{\mathcal{M}_2} \Rightarrow \mathcal{C}_{\mathcal{M}_1}$

and $\mathcal{C}_{\mathcal{M}_1} \rightarrowtriangle Q$ we have that $\mathcal{C}_{\mathcal{M}_2} \rightarrowtriangle Q$ and then applying Lemma 6.2.3 again we obtain that Q is target rewritable under $\mathcal{M}_2$.

Assume now that every target-rewritable query under $\mathcal{M}_1$ is also target rewritable under $\mathcal{M}_2$. Applying Lemma 6.2.3 we obtain that for every source query Q, if $\mathcal{C}_{\mathcal{M}_1} \rightarrowtriangle Q$ then $\mathcal{C}_{\mathcal{M}_2} \rightarrowtriangle Q$. Finally, since $\mathcal{C}_{\mathcal{M}_1} \rightarrowtriangle Q$ for every query $Q \in \mathcal{C}_{\mathcal{M}_1}$, we have that $\mathcal{C}_{\mathcal{M}_2} \rightarrowtriangle Q$ for every $Q \in \mathcal{C}_{\mathcal{M}_1}$ which implies that $\mathcal{C}_{\mathcal{M}_2} \rightarrowtriangle \mathcal{C}_{\mathcal{M}_1}$ and then from Lemma 6.2.4 we obtain that $\mathcal{M}_1 \preceq_{\mathrm{s}} \mathcal{M}_2$. □

6.2.2. Fundamental algorithmic issues for the order $\preceq_{\mathrm{s}}$

Some algorithmic issues related to the order $\preceq_{\mathrm{s}}$ play a key role in the development of algorithms for some metadata management operators. The main algorithmic issue that we consider in this section is the problem of constructing a mapping $\mathcal{N}$ such that $\mathcal{M}_1 = \mathcal{M}_2 \circ \mathcal{N}$, whenever $\mathcal{M}_1 \preceq_{\mathrm{s}} \mathcal{M}_2$. Next, we present an algorithm that solves this problem for CQ$^{\neq}$-TO-CQ dependencies, which uses the following terminology.

Let $\mathcal{M}$ be a mapping from $\mathbf{S}$ to $\mathbf{T}$ and Q a query that is target rewritable under $\mathcal{M}$. Recall that if Q' is a query such that $Q(I) = \mathrm{certain}_{\mathcal{M}}(Q', I)$ holds for every instance I, then we say that Q' is a *target rewriting* for Q under $\mathcal{M}$. Correspondingly, we also say that Q is a *source rewriting* of Q'.

Let $\mathcal{M}$ be an st-mapping from $\mathbf{S}$ to $\mathbf{T}$ specified by FO-TO-CQ dependencies. We know from Lemma 3.3.1 that it is always possible to compute the *source rewriting* of a conjunctive query Q' over $\mathbf{T}$, that is, there exists a procedure QUERYREWRITING that, given such a mapping $\mathcal{M}$ and a query Q' in CQ over $\mathbf{T}$, computes a query Q in FO over $\mathbf{S}$ that is a source rewriting of Q'. In particular, if the input mapping is specified by CQ$^{\neq}$-TO-CQ dependencies, then the output of the procedure is a query Q in UCQ$^{=,\neq}$. We state this result in the following lemma (the proof follows directly from the proof of Lemma 3.3.1).

LEMMA 6.2.5. *Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ be an st-mapping with Σ a set of* CQ$^{\neq}$-TO-CQ *dependencies, and Q a conjunctive query over $\mathbf{T}$. Algorithm* QUERYREWRITING$(\mathcal{M}, Q)$ *in Lemma 3.3.1 has as output a query Q' in* UCQ$^{=,\neq}$ *that is a source rewriting of Q.*

For the class of mappings specified by $\text{CQ}^{\neq}$-TO-CQ dependencies, target rewritings can also be computed. More precisely, it follows from the proof of Theorem 6.2.2 that there exists a procedure TARGETREWRITING that, given a mapping $\mathcal{M}$ specified by a set of $\text{CQ}^{\neq}$-TO-CQ dependencies and a target rewritable query Q in $\text{UCQ}^{=,\neq}$ over $\mathbf{S}$, computes a query in $\text{UCQ}^{=,\neq,\mathbf{C}}$ that is a target rewriting of Q. We formalize this result in the following Lemma (the proof can be found in Appendix A.2).

LEMMA 6.2.6. *There exists an algorithm* TARGETREWRITING *that given an st-mapping* $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$*, with* Σ *a set of* $\text{CQ}^{\neq}$-TO-CQ *dependencies, and a query* Q *in* $\text{UCQ}^{=,\neq}$ *over* $\mathbf{S}$ *that is target rewritable under* $\mathcal{M}$*, computes a query* Q' *in* $\text{UCQ}^{=,\neq,\mathbf{C}}$ *over* $\mathbf{T}$ *that is a target rewriting of* Q*. The query* Q' *is such that every inequality occurring in* Q' *is between variables that are under predicate* $\mathbf{C}(\cdot)$*.*

The following algorithm uses the procedures mentioned in Lemmas 6.2.5 and 6.2.6 as black boxes.

Algorithm COMPUTEORDER($\mathcal{M}_1, \mathcal{M}_2$)

Input: st-mappings $\mathcal{M}_1 = (\mathbf{S}, \mathbf{T}_1, \Sigma_1)$ and $\mathcal{M}_2 = (\mathbf{S}, \mathbf{T}_2, \Sigma_2)$, where Σ_1, Σ_2 are sets of $\text{CQ}^{\neq}$-TO-CQ dependencies and $\mathcal{M}_1 \preceq_{\text{s}} \mathcal{M}_2$.

Output: A mapping $\mathcal{N} = (\mathbf{T}_2, \mathbf{T}_1, \Sigma)$ such that $\mathcal{M}_1 = \mathcal{M}_2 \circ \mathcal{N}$, where Σ is a set of $\text{CQ}^{\neq,\mathbf{C}}$-TO-CQ dependencies.

(1) Construct a set Σ' of dependencies as follows. Start with $\Sigma' = \emptyset$. Then, for every dependency $\varphi(\bar{x}) \rightarrow \psi(\bar{x}) \in \Sigma_1$ repeat the following:
   (a) Use algorithm QUERYREWRITING($\mathcal{M}_1, \psi(\bar{x})$) to compute a formula $\alpha(\bar{x})$ in $\text{UCQ}^{=,\neq}$ over $\mathbf{S}$ that is a source rewriting of $\psi(\bar{x})$ under $\mathcal{M}_1$.
   (b) Use algorithm TARGETREWRITING($\mathcal{M}_2, \alpha(\bar{x})$) to compute a formula $\beta(\bar{x})$ in $\text{UCQ}^{=,\neq,\mathbf{C}}$ over $\mathbf{T}_2$ that is a target rewriting of $\alpha(\bar{x})$ under $\mathcal{M}_2$.
   (c) For every disjunct $\gamma(\bar{x})$ of $\beta(\bar{x})$, add to Σ' the formula

$$\gamma(\bar{x}) \wedge \mathbf{C}(\bar{x}) \quad \rightarrow \quad \psi(\bar{x}).$$

(2) Let Σ be the set obtained from Σ' by eliminating the equalities by using variable replacements.

(3) Return the mapping $\mathcal{N} = (\mathbf{T}_2, \mathbf{T}_1, \Sigma)$

□

It is important to notice that we need $\mathcal{M}_1 \preceq_{\mathrm{s}} \mathcal{M}_2$ as precondition for the algorithm, otherwise we are not guaranteed to find a target rewriting of $\alpha(\bar{x})$ in Step 1 (b). Notice that $\alpha(\bar{x})$ is a source rewriting of $\psi(\bar{x})$ under $\mathcal{M}_1$, thus we know that $\alpha(\bar{x})$ is target rewritable under $\mathcal{M}_1$ (since $\psi(\bar{x})$ is a target rewriting of $\alpha(\bar{x})$). Then since $\mathcal{M}_1 \preceq_{\mathrm{s}} \mathcal{M}_2$, from Theorem 6.2.2 we know that $\alpha(\bar{x})$ is also target rewritable under $\mathcal{M}_2$.

The following proposition shows that the algorithm is correct.

THEOREM 6.2.7. COMPUTEORDER$(\mathcal{M}_1, \mathcal{M}_2)$ *returns a mapping* $\mathcal{N}$ *specified by a set of* $\mathrm{CQ}^{\neq,\mathbf{C}}$-TO-CQ *dependencies such that* $\mathcal{M}_1 = \mathcal{M}_2 \circ \mathcal{N}$.

PROOF. Let $\mathcal{M}_1 = (\mathbf{S}, \mathbf{T}_1, \Sigma_1)$ and $\mathcal{M}_2 = (\mathbf{S}, \mathbf{T}_2, \Sigma_2)$ st-mappings with Σ_1 and Σ_2 sets of $\mathrm{CQ}^{\neq}$-TO-CQ dependencies. Assume that $\mathcal{M}_1 \preceq_{\mathrm{s}} \mathcal{M}_2$ and let $\mathcal{N} = (\mathbf{T}_2, \mathbf{T}_1, \Sigma)$ be the output of COMPUTEORDER$(\mathcal{M}_1, \mathcal{M}_2)$. We need to prove that for every $I \in \mathrm{Inst}(\mathbf{S})$ and $J \in \mathrm{Inst}(\mathbf{T}_1)$ we have that $(I, J) \models \Sigma_1$ if and only if there exists $K \in \mathrm{Inst}(\mathbf{T}_2)$ such that $(I, K) \models \Sigma_2$ and $(K, J) \models \Sigma$.

($\Leftarrow$) Assume that there exists $K \in \mathrm{Inst}(\mathbf{T}_2)$ such that $(I, K) \models \Sigma_2$ and $(K, J) \models \Sigma$. Let $\varphi(\bar{x}) \rightarrow \psi(\bar{x})$ be a dependency in Σ and assume that $I \models \varphi(\bar{a})$ for some tuple $\bar{a}$ of constant values. We need to prove that $J \models \psi(\bar{a})$. Let $\alpha(\bar{x})$ be the source rewriting of $\psi(\bar{x})$ under $\mathcal{M}_1$ computed in Step 1 (a). Since $I \models \varphi(\bar{a})$ we know that for every $L \in \mathrm{Sol}_{\mathcal{M}_1}(I)$ it holds that $L \models \psi(\bar{a})$, and thus $I \models \alpha(\bar{a})$. Now, let $\beta(\bar{x})$ be the target rewriting of $\alpha(\bar{x})$ under $\mathcal{M}_2$ computed in Step 1 (b). Notice that we are assuming that $(I, K) \models \Sigma_2$. Therefore, since $I \models \alpha(\bar{a})$ and $\beta(\bar{x})$ is a rewriting of $\alpha(\bar{x})$, we obtain that $K \models \beta(\bar{a})$. Finally, we know that Σ contains a dependency of the form $\gamma(\bar{x}) \wedge \mathbf{C}(\bar{x}) \rightarrow \psi(\bar{x})$ where $\gamma(\bar{x})$ is a disjunct in $\beta(\bar{x})$. Thus, since $K \models \beta(\bar{a})$, $\bar{a}$ is a tuple of constants, and we are assuming that $(K, J) \models \Sigma$, we obtain that $J \models \psi(\bar{a})$, which was to be shown.

($\Rightarrow$) Assume that $(I, J) \models \Sigma_1$. We need to show that there exists an instance $K \in \mathrm{Inst}(\mathbf{T}_2)$ such that $(I, K) \models \Sigma_2$ and $(K, J) \models \Sigma$. We show next that $(I, \mathrm{chase}_{\Sigma_2}(I)) \models \Sigma_2$ and $(\mathrm{chase}_{\Sigma_2}(I), J) \models \Sigma$. The first property is trivial so we just need to prove that $(\mathrm{chase}_{\Sigma_2}(I), J) \models \Sigma$. Then let $\gamma(\bar{x}) \wedge \mathbf{C}(\bar{x}) \rightarrow \psi(\bar{x})$ be a dependency in Σ where $\gamma(\bar{x})$ is a disjunct in the query $\beta(\bar{x})$ computed in Step 1 (b). Assume that $\mathrm{chase}_{\Sigma_2}(I) \models \gamma(\bar{a}) \wedge \mathbf{C}(\bar{a})$ for some tuple $\bar{a}$. We need to show that $J \models \psi(\bar{a})$. Now, since $\gamma(\bar{x})$ is a conjunctive query with equalities and inequalities, every inequality is between variables under predicate $\mathbf{C}(\cdot)$, and $\bar{a}$ is a tuple of constants, by the properties of the chase (see (Arenas, Barceló, & Reutter, 2009)) we know that for every $L \in \mathrm{Sol}_{\mathcal{M}_2}(I)$ we have that $L \models \gamma(\bar{a})$ and thus $L \models \beta(\bar{a})$. Let $\alpha(\bar{x})$ be the query computed in Step 1 (a). Since $\beta(\bar{x})$ is a target rewriting of $\alpha(\bar{x})$ under $\mathcal{M}_2$ and $L \models \beta(\bar{a})$ for every $L \in \mathrm{Sol}_{\mathcal{M}_2}(I)$, we have that $I \models \alpha(\bar{a})$. Finally, since $\alpha(\bar{x})$ is a source rewriting of $\psi(\bar{x})$ under $\mathcal{M}_1$ and $(I, J) \models \Sigma_1$ we obtain that $J \models \psi(\bar{a})$, which was to be shown. □

6.3. Two Applications of $\preceq_{\mathrm{s}}$ in Data Exchange

The issue of providing foundations for metadata management has appeared in different contexts. In particular, in the data exchange context, the schema evolution problem has been a driving force for the development of the composition and inverse operators (Bernstein, 2003; Bernstein & Melnik, 2007; Kolaitis, 2005; Fagin et al., 2011). In this section, we show the potential of the order $\preceq_{\mathrm{s}}$ for providing foundations for metadata management, as the machinery developed in the previous sections can be used as a uniform framework to study the inverse operator and the schema evolution problem.

6.3.1. Inverting schema mappings

In this section, we focus on the definition of the inverse operator given by Fagin (2007), and we show that this operator can be defined in terms of the order $\preceq_{\mathrm{s}}$. Interestingly, this characterization can be used to extend, and provide simpler proofs of, some of the fundamental results about this operator.

We start by recalling the definition of inverse given by Fagin (2007). For a ground schema $\mathbf{R}$, let $\widehat{\mathbf{R}}$ be the schema $\{\widehat{R} \mid R \in \mathbf{R}\}$, and $\widehat{\mathrm{Id}}_{\mathbf{R}} = (\mathbf{R}, \widehat{\mathbf{R}}, \Sigma)$ be an *identity mapping*, where Σ contains a dependency of the form $R(x_1, \ldots, x_k) \rightarrow \widehat{R}(x_1, \ldots, x_k)$, for every k-ary predicate $R \in \mathbf{R}$. Then given $\mathcal{M}$ from $\mathbf{R}$ to $\mathbf{R}_1$ and $\mathcal{M}'$ from $\mathbf{R}_1$ to $\widehat{\mathbf{R}}$, mapping $\mathcal{M}'$ is said to be a Fagin-inverse of mapping $\mathcal{M}$ if $\mathcal{M} \circ \mathcal{M}' = \widehat{\mathrm{Id}}_{\mathbf{R}}$ (Fagin, 2007)[1].

The following theorem shows that the notion of Fagin-inverse can be defined in terms of the order $\preceq_{\mathrm{s}}$ for the class of mappings that are total and closed-down on the left.

THEOREM 6.3.1. *Let $\mathcal{M}$ be a mapping from a ground schema $\mathbf{R}$ to a schema $\mathbf{R}_1$ that is total and closed-down on the left. Then the following statements are equivalent.*

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

(2) *$\mathcal{M}$ is $\preceq_{\mathrm{s}}$-maximal in the class of total and closed-down on the left mappings.*

(3) $\widehat{\mathrm{Id}}_{\mathbf{R}} \preceq_{\mathrm{s}} \mathcal{M}$.

PROOF. (1) $\Rightarrow$ (3): Given that $\mathcal{M}$ is Fagin-invertible, there exists a mapping $\mathcal{N}$ from $\mathbf{R}_1$ to $\widehat{\mathbf{R}}$ such that $\mathcal{M} \circ \mathcal{N} = \widehat{\mathrm{Id}}_{\mathbf{R}}$. Thus, we have that $\widehat{\mathrm{Id}}_{\mathbf{R}} \preceq_{\mathrm{s}} \mathcal{M}$.

(3) $\Rightarrow$ (2): Assume that $\widehat{\mathrm{Id}}_{\mathbf{R}} \preceq_{\mathrm{s}} \mathcal{M}$. To prove that $\mathcal{M}$ is $\preceq_{\mathrm{s}}$-maximal in the class of total and closed-down on the left mappings, we show that for every $\mathcal{M}'$ in this class such that $\mathcal{M} \preceq_{\mathrm{s}} \mathcal{M}'$, it holds that $\mathcal{M}' \preceq_{\mathrm{s}} \mathcal{M}$. Given that $\mathcal{M} \preceq_{\mathrm{s}} \mathcal{M}'$, we know that $\mathcal{M}'$ is a mapping from $\mathbf{R}$ to some schema $\mathbf{R}_2$. Then define a mapping $\mathcal{N}$ from $\widehat{\mathbf{R}}$ to $\mathbf{R}$ as follows. For every instance I of $\mathbf{R}$, let $\widehat{I}$ be an instance of $\widehat{\mathbf{R}}$ defined as $\widehat{R}^{\widehat{I}} = R^I$, for every $R \in \mathbf{R}$, and then let $\mathcal{N}$ be defined as $\{(\widehat{I}, I) \mid I \in \mathrm{Inst}(\mathbf{R})\}$. It is straightforward to prove that $\widehat{\mathrm{Id}}_{\mathbf{R}} \circ \mathcal{N} \circ \mathcal{M}' = \mathcal{M}'$. Thus, we have that $\mathcal{M}' \preceq_{\mathrm{s}} \widehat{\mathrm{Id}}_{\mathbf{R}}$, from which we conclude that $\mathcal{M}' \preceq_{\mathrm{s}} \mathcal{M}$ since $\widehat{\mathrm{Id}}_{\mathbf{R}} \preceq_{\mathrm{s}} \mathcal{M}$.

(2) $\Rightarrow$ (1): Let $\mathcal{N}$ be a mapping defined as in the previous paragraph. Then given that $\mathcal{M}$ is closed-down on the left, we have that $\mathcal{M} = \widehat{\mathrm{Id}}_{\mathbf{R}} \circ \mathcal{N} \circ \mathcal{M}$ and, therefore,

[1]The definition of Fagin-inverse given in this section is a reformulation of the one given in Section 2.5 by using st-tgds to define the identity mapping $\overline{\mathrm{Id}}$ (used in Definition 2.5.1). Just notice that $\widehat{\mathrm{Id}}_{\mathbf{R}} = \{(I_1, \hat{I}_2) \mid (I_1, I_2) \in \mathrm{Inst}(\mathbf{R}) \times \mathrm{Inst}(\mathbf{R})$ and $I_1 \subseteq I_2\}$, where $\hat{I}_2$ is the instance of $\hat{\mathbf{R}}$ obtained from I_2 by replacing every relation name R by its copy $\hat{R}$.

$\mathcal{M} \preceq_{\mathrm{s}} \widehat{\mathrm{Id}}_{\mathbf{R}}$. Thus, given that $\mathcal{M}$ is $\preceq_{\mathrm{s}}$-maximal in the class of total and closed-down on the left mappings, we conclude that $\widehat{\mathrm{Id}}_{\mathbf{R}} \preceq_{\mathrm{s}} \mathcal{M}$. Hence, there exists a mapping $\mathcal{N}'$ such that $\mathcal{M} \circ \mathcal{N}' = \widehat{\mathrm{Id}}_{\mathbf{R}}$, which implies that $\mathcal{M}$ is Fagin-invertible by definition of the notion of Fagin-inverse. This concludes the proof of theorem. □

The preceding theorem can be used to extend some of the fundamental results that have been obtained for the Fagin-inverse operator. In particular, a fundamental question about any notion of inverse is how to compute it. Fagin, Kolaitis, Popa, and Tan (2008), gave an algorithm for computing Fagin-inverses of mappings specified by st-tgds. From Theorem 6.3.1, we know that algorithm COMPUTEORDER can also be used for this task, and not only for the case of st-tgds but also for the larger class of $\mathrm{CQ}^{\neq}$-TO-CQ dependencies.

PROPOSITION 6.3.2. *Let $\mathcal{M} = (\mathbf{R}, \mathbf{R}_1, \Sigma)$ be a Fagin-invertible st-mapping specified by a set Σ of* $\mathrm{CQ}^{\neq}$-TO-CQ *dependencies. Then the output of* COMPUTEORDER$(\widehat{\mathrm{Id}}_{\mathbf{R}}, \mathcal{M})$ *is a Fagin-inverse of $\mathcal{M}$.*

The use of query rewriting in COMPUTEORDER makes the above approach for computing Fagin-inverses more suitable for optimization compared to the approach proposed by Fagin, Kolaitis, Popa, and Tan (2008). In fact, one can reuse the large number of techniques developed for query rewriting (Levy et al., 1995; Duschka & Genesereth, 1997; Halevy, 2001; Pottinger & Halevy, 2001) when implementing procedure COMPUTEORDER.

As a direct corollary of Proposition 6.3.2, we obtain another fundamental result for the notion of inverse proposed by Fagin (2007).

COROLLARY 6.3.3. *For every Fagin-invertible st-mapping $\mathcal{M}$ specified by a set of* $\mathrm{CQ}^{\neq}$-TO-CQ *dependencies, there exists an inverse of $\mathcal{M}$ that is specified by a set of* $\mathrm{CQ}^{\neq,\mathrm{C}}$-TO-CQ *dependencies.*

It is important to notice that Fagin, Kolaitis, Popa, and Tan (2008) showed that if a mapping $\mathcal{M}$ specified by a set of st-tgds is Fagin-invertible, then it has a Fagin-inverse that

is given by a set of $\text{CQ}^{\neq,\text{C}}$-TO-CQ dependencies. The above corollary extends this results for the class of Fagin-invertible mappings specified by $\text{CQ}^{\neq}$-TO-CQ dependencies.

As we mentioned in Section 6.1.1, the notion of information loss presented by Fagin et al. (2009) is tightly connected with invertibility of mappings. In fact, Fagin et al. (2009) claim that if $\mathcal{M}_1$ and $\mathcal{M}_2$ are mappings specified by st-tgds, then $\mathcal{M}_2$ is less lossy than $\mathcal{M}_1$ when, intuitively, "$\mathcal{M}_2$ is *more invertible* than $\mathcal{M}_1$" (Fagin et al., 2009). We conclude this subsection by showing that if one goes beyond st-tgds (which is the class of mappings considered by Fagin et al. (2009)), the orders $\preceq_{\text{R}}$ and $\preceq_{\text{E}}$ fail to capture the idea of *being more invertible*. Notice that Theorem 6.3.1 shows that our order $\preceq_{\text{S}}$ captures exactly the intuition mentioned by Fagin et al. (2009) for a large class of mappings, which gives evidence of the usefulness of $\preceq_{\text{S}}$ to compare schema mappings. We begin by showing that $\preceq_{\text{R}}$ does not capture Fagin-invertibility beyond st-tgds.

PROPOSITION 6.3.4. *There exists an st-mapping $\mathcal{M}$ specified by a set of* CQ-TO-UCQ *dependencies such that $\mathcal{M}$ is $\preceq_{\text{R}}$-maximal on the class of total and closed-down on the left mappings and $\mathcal{M}$ is not invertible.*

PROOF. Consider the st-mapping $\mathcal{M}_2$ in the proof of Proposition 6.1.5. Notice that $\mathcal{M}_2$ is closed-down on the left and total. Moreover, we have said that $\mathcal{M}_2$ satisfies the following property. For every I, K if $\text{Sol}_{\mathcal{M}_2}(I) \subseteq \text{Sol}_{\mathcal{M}_2}(K)$ then $K \subseteq I$. Let $\mathcal{M}$ be an arbitrary mapping from schema $\mathbf{S}$ that is total and closed-down on the left. By definition we have that if $K \subseteq I$ then $\text{Sol}_{\mathcal{M}}(I) \subseteq \text{Sol}_{\mathcal{M}}(K)$. Thus, for every I, K in $\mathbf{S}$, if $\text{Sol}_{\mathcal{M}_2}(I) \subseteq \text{Sol}_{\mathcal{M}_2}(K)$ then $\text{Sol}_{\mathcal{M}}(I) \subseteq \text{Sol}_{\mathcal{M}}(K)$ which implies that $\mathcal{M} \preceq_{\text{R}} \mathcal{M}_2$. Then we have that $\mathcal{M}_2$ is $\preceq_{\text{R}}$-maximal on the class of total and closed-down on the left mappings. Finally, it was shown Arenas et al. (2009, Proposition 6.6 (2)) that $\mathcal{M}_2$ is not Fagin-invertible, which completes the proof of the proposition. □

Fagin et al. (2009) introduce an alternative notion of inversion for mappings with null values in source and target instances. They call this notion *extended invertibility*, and show that the order $\preceq_{\text{E}}$ is tightly connected with this extended notion of inversion. Recall that

$\rightarrow$ is defined as the mapping $\{(I, J) \mid$ there exists a homomorphism from I to $J\}$ (Fagin et al., 2009). Then, given a mapping $\mathcal{M}$ with nulls in source and target instances, $\mathcal{M}'$ is an *extended inverse* of $\mathcal{M}$ if $e(\mathcal{M}) \circ e(\mathcal{M}') = \rightarrow$ (Fagin et al., 2009). (See Definition 3.1.13 for the formalization of $e(\mathcal{M})$.) The following result shows that beyond st-tgds, $\preceq_{\text{E}}$ captures neither the notion of Fagin-invertibility nor the notion of extended invertibility.

PROPOSITION 6.3.5. *There exists a mapping $\mathcal{M}$ such that (1) $\mathcal{M}$ is specified by a set of* CQ-TO-UCQ *dependencies, (2) $\mathcal{M}$ has nulls in source and target instances, (3) $\mathcal{M}$ is $\preceq_{\text{E}}$-maximal in the class of all mappings, and (4) $\mathcal{M}$ is neither invertible nor extended invertible.*

PROOF. Recall that for a mapping $\mathcal{M}$ it holds that $e(\mathcal{M}) = \rightarrow \circ \mathcal{M} \circ \rightarrow$. Notice that if I, K are two instance such that $K \rightarrow I$, then it holds that $\text{Sol}_{e(\mathcal{M})}(I) \subseteq \text{Sol}_{e(\mathcal{M})}(K)$. Now, consider the mapping $\mathcal{M}$ given by the CQ-TO-UCQ dependencies:

$$\begin{aligned} F(x) &\rightarrow R(x) \vee S(x) \\ G(x) &\rightarrow S(x) \vee T(x) \\ H(x) &\rightarrow T(x) \vee R(x) \end{aligned}$$

and assume that source and target instances may contain null values.

We show first that $\mathcal{M}$ is $\preceq_{\text{E}}$-maximal in the class of all mappings. Let $\mathcal{M}'$ be an arbitrary mapping. In the proof of Proposition 6.1.6 it was shown that for every I, K if $\text{Sol}_{e(\mathcal{M})}(I) \subseteq \text{Sol}_{e(\mathcal{M})}(K)$ then $K \rightarrow I$. Notice that $K \rightarrow I$ implies that $\text{Sol}_{e(\mathcal{M}')}(I) \subseteq \text{Sol}_{e(\mathcal{M}')}(K)$. Thus, we have that if $\text{Sol}_{e(\mathcal{M})}(I) \subseteq \text{Sol}_{e(\mathcal{M})}(K)$ then $\text{Sol}_{e(\mathcal{M}')}(I) \subseteq \text{Sol}_{e(\mathcal{M}')}(K)$. This shows that $\mathcal{M}' \preceq_{\text{E}} \mathcal{M}$.

We prove now that $\mathcal{M}$ is not Fagin-invertible. Arenas et al. (2009) introduced the notion of *strong witness solutions* that characterizes Fagin-invertibility for general mappings. Given a mapping $\mathcal{N}$ and an instance I, instance J is a strong witness solution for I under $\mathcal{N}$ if $J \in \text{Sol}_{\mathcal{N}}(I)$ and for every instance I' if $J \in \text{Sol}_{\mathcal{N}}(I')$ then $I' \subseteq I$. It was shown by Arenas et al. (2009) that a mapping $\mathcal{N}$ is Fagin-invertible if and only if every instance

has a strong witness solution under $\mathcal{N}$. We use this result to show next that $\mathcal{M}$ as defined above does not have a Fagin-inverse. Let $I = \{F(a)\}$ with $a \in \mathbf{C}$, and assume that I has a strong witness solution, say J, under $\mathcal{M}$. Thus, J is such that $R(a) \in J$ or $S(a) \in J$. Assume first that $R(a) \in J$ and consider the instance $K = \{G(a)\}$. Then we have that $J \in \mathrm{Sol}_{\mathcal{M}}(K)$ but $K \not\subseteq I$ which contradicts the fact that J is a strong witness solution for I.

We show next that $\mathcal{M}$ is not extended invertible. For this we use the notion of *capturing instance* introduced by Fagin et al. (2009). Given a mapping $\mathcal{N}$ and an instance I, instance J is a capturing instance for I under $\mathcal{N}$ if $J \in \mathrm{Sol}_{e(\mathcal{N})}(I)$ and for every instance I' if $J \in \mathrm{Sol}_{e(\mathcal{N})}(I')$ then $I' \rightarrow I$. It was shown by Fagin et al. (2009) that a mapping $\mathcal{N}$ is extended invertible if and only if every instance has a capturing instance under $\mathcal{N}$. We use this to prove that $\mathcal{M}$ is not extended invertible. Let $I = \{F(a)\}$ with $a \in \mathbf{C}$, and assume that J is a capturing instance for I under $\mathcal{M}$. Thus, since $J \in \mathrm{Sol}_{e(\mathcal{M})}(I)$ we have that $R(a) \in J$ or $S(a) \in J$. Assume first that $R(a) \in J$ and consider the instance $K = \{G(a)\}$. Then we have that $J \in \mathrm{Sol}_{e(\mathcal{M})}(K)$ but $K \not\rightarrow I$ which contradicts the fact that J is a capturing instance for I. □

We conclude this subsection by pointing out that we have mainly focused here on the notion of inverse proposed by Fagin (2007). However it would be interesting to study whether the notions of quasi-inverse, maximum recovery, and $\mathcal{C}$-maximum recovery can also be characterized in term of the order $\preceq_{\mathrm{s}}$, and whether the machinery proposed in this paper can be used to improve our understanding of these notions. Although we have made a bit of progress in this direction, these questions remain unanswered.

6.3.2. Schema evolution

The schema evolution problem has been one of the driving forces behind the study of the composition and inverse operators (Bernstein, 2003; Bernstein & Melnik, 2007; Kolaitis, 2005; Fagin et al., 2011). As we explained in Section 3.2, two main scenarios have been identified for this problem. In the first scenario, one is given a mapping $\mathcal{M}$ from a schema

$\mathbf{S}$ to a schema $\mathbf{T}$, and a mapping $\mathcal{M}'$ that specifies how $\mathbf{T}$ evolves into a new schema $\mathbf{T}'$. The schema evolution problem is then to provide a mapping from $\mathbf{S}$ to $\mathbf{T}'$, that captures the metadata provided by $\mathcal{M}$ and $\mathcal{M}'$. In this scenario, it is always possible to find a solution for this problem by using the composition operator (Fagin, Kolaitis, Popa, & Tan, 2005; Kolaitis, 2005), as mapping $\mathcal{M} \circ \mathcal{M}'$ correctly represents the relationship between $\mathbf{S}$ and $\mathbf{T}'$. In the second scenario, one is also given a mapping $\mathcal{M}$ from a schema $\mathbf{S}$ to a schema $\mathbf{T}$, but in this case $\mathbf{S}$ evolves into a new schema $\mathbf{S}'$, and the relationship between $\mathbf{S}$ and $\mathbf{S}'$ is given by a mapping $\mathcal{M}'$. Then again the question is how to construct a mapping from $\mathbf{S}'$ to $\mathbf{T}$ that captures the metadata provided by $\mathcal{M}$ and $\mathcal{M}'$. In this section, we use the machinery developed in the previous sections to formally study this problem. It is important to notice that we focus on the second scenario, as the first one has been completely solved by using the composition operator (Fagin, Kolaitis, Popa, & Tan, 2005).

Let $\mathcal{M}_1$ be a mapping from a schema $\mathbf{R}$ to a schema $\mathbf{R}_1$, and $\mathcal{M}_2$ a mapping from $\mathbf{R}$ to a schema $\mathbf{R}_2$. Then a mapping $\mathcal{N}$ is *an exact solution for the schema evolution problem for* $(\mathcal{M}_1, \mathcal{M}_2)$ if $\mathcal{M}_1 = \mathcal{M}_2 \circ \mathcal{N}$. (Notice that if $\mathcal{N}$ is an exact solution for the schema evolution problem for $(\mathcal{M}_1, \mathcal{M}_2)$, then it is also an *ideal solution for the schema evolution problem* as defined in Section 3.2.) The following result shows that the schema evolution problem can be characterized in terms of the order $\preceq_{\mathrm{s}}$, as it is just a reformulation of the definition of $\preceq_{\mathrm{s}}$.

PROPOSITION 6.3.6. *There exists an exact solution for the schema evolution problem for* $(\mathcal{M}_1, \mathcal{M}_2)$ *iff* $\mathcal{M}_1 \preceq_{\mathrm{s}} \mathcal{M}_2$.

Thus, as a corollary of Theorem 6.2.7, we obtain a solution for the schema evolution problem for the class of mappings specified by $\mathrm{CQ}^{\neq}$-TO-CQ dependencies

COROLLARY 6.3.7. *For every st-mappings* $\mathcal{M}_1$, $\mathcal{M}_2$ *specified by* $\mathrm{CQ}^{\neq}$-TO-CQ *dependencies, if there exists an exact solution for the schema evolution problem for* $(\mathcal{M}_1, \mathcal{M}_2)$, *then* COMPUTEORDER$(\mathcal{M}_1, \mathcal{M}_2)$ *returns an exact solution for this problem specified by a set of* $\mathrm{CQ}^{\neq,\mathrm{C}}$-TO-CQ *dependencies.*

6.4. Target Information and Redundancy

In this section, we use the order $\preceq_{\text{S}}$ to define three additional concepts which, together with $\preceq_{\text{S}}$, provide a theoretical framework to study complex metadata management operators such as extract and merge (Melnik, 2004). More precisely, we introduce in Section 6.4.1 an order to compare mappings that possess the same target schema. This order, denoted by $\preceq_{\text{T}}$, intuitively measures the amount of target information *covered* by a mapping. As there may exist multiple ways to transfer the same information from a source schema, or to cover the same information of a target schema, one also needs a way of distinguishing between different alternatives. To deal with this requirement, in Sections 6.4.2 and 6.4.3, we use the orders $\preceq_{\text{S}}$ and $\preceq_{\text{T}}$ to introduce the notions of target redundancy and source redundancy, and show that they capture the intuition of using the *exact amount of resources* needed to transfer information between schemas.

6.4.1. Target information covered by a mapping

In some metadata management scenarios, it is important to measure the amount of target information *covered* by a mapping. When $\preceq_{\text{S}}$ was introduced, we said that $\mathcal{M}_1 \preceq_{\text{S}} \mathcal{M}_2$ if $\mathcal{M}_2$ transfers enough source information to be able to reconstruct the information transferred by $\mathcal{M}_1$. Similarly, we say that $\mathcal{M}_2$ *covers as much target information as* $\mathcal{M}_1$, denoted by $\mathcal{M}_1 \preceq_{\text{T}} \mathcal{M}_2$, if $\mathcal{M}_2$ covers enough target information to be able to reconstruct the information that is covered by $\mathcal{M}_1$. More precisely,

DEFINITION 6.4.1 (Order $\preceq_{\text{T}}$). *Let $\mathcal{M}_1$ and $\mathcal{M}_2$ be mappings that share the target schema. Then $\mathcal{M}_1 \preceq_{\text{T}} \mathcal{M}_2$ if there exists a mapping $\mathcal{N}$ such that $\mathcal{M}_1 = \mathcal{N} \circ \mathcal{M}_2$.*

Moreover, we say that $\mathcal{M}_1$ and $\mathcal{M}_2$ *cover the same target information*, and write $\mathcal{M}_1 \equiv_{\text{T}} \mathcal{M}_2$, if $\mathcal{M}_1 \preceq_{\text{T}} \mathcal{M}_2$ and $\mathcal{M}_2 \preceq_{\text{T}} \mathcal{M}_1$.

Example 6.4.2. Let $\mathcal{M}_1$ be the st-mapping specified by dependency $A(x) \rightarrow T(x, x)$, and $\mathcal{M}_2$ the st-mapping specified by $R(x, y, z) \rightarrow T(x, y)$. Then $\mathcal{M}_2$ covers more target information than $\mathcal{M}_1$. In fact, we have that $\mathcal{M}_1 \preceq_{\text{T}} \mathcal{M}_2$ since for the mapping $\mathcal{N}$ specified

by $A(x) \rightarrow \exists z R(x, x, z)$ we have that $\mathcal{M}_1 = \mathcal{N} \circ \mathcal{M}_2$. Moreover, it can be shown that $\mathcal{M}_2 \not\preceq_{\mathrm{T}} \mathcal{M}_1$. □

The following result shows that, as pointed out above, $\preceq_{\mathrm{T}}$ can be defined in terms of the order $\preceq_{\mathrm{S}}$.

PROPOSITION 6.4.3. $\mathcal{M}_1 \preceq_{\mathrm{T}} \mathcal{M}_2$ *iff* $(\mathcal{M}_1)^{-1} \preceq_{\mathrm{S}} (\mathcal{M}_2)^{-1}$.

PROOF. Assume that $\mathcal{M}_1 \preceq_{\mathrm{T}} \mathcal{M}_2$. Then there exists a mapping $\mathcal{N}$ such that $\mathcal{M}_1 = \mathcal{N} \circ \mathcal{M}_2$. This implies that $(\mathcal{M}_1)^{-1} = (\mathcal{M}_2)^{-1} \circ \mathcal{N}^{-1}$ and thus $(\mathcal{M}_1)^{-1} \preceq_{\mathrm{S}} (\mathcal{M}_2)^{-1}$. The other direction is similar. □

This relationship between the orders does not imply that we can directly apply to $\preceq_{\mathrm{T}}$ the results for $\preceq_{\mathrm{S}}$ that we have obtained in Section 6.2. For instance, notice that if $\mathcal{M}$ is specified by CQ-TO-CQ dependencies, then $\mathcal{M}^{-1}$ cannot be specified by CQ-TO-CQ dependencies ($\mathcal{M}^{-1}$ cannot even be specified by FO-TO-CQ dependencies). Thus we need to develop specific tools (algorithms, characterizations, etc.) for the order $\preceq_{\mathrm{T}}$. That is what we do in the next section.

Characterizing the order $\preceq_{\mathrm{T}}$.

We provide here a characterization of the order $\preceq_{\mathrm{T}}$ for mappings given by FO-TO-CQ dependencies that is based on the concept of universal solution (Fagin, Kolaitis, Miller, & Popa, 2005), and supports our claim that $\preceq_{\mathrm{T}}$ can be used to compare the amount of target information covered by two schema mappings.

THEOREM 6.4.4. *Let* $\mathcal{M}_1 = (\mathbf{S}_1, \mathbf{T}, \Sigma_1)$ *and* $\mathcal{M}_2 = (\mathbf{S}_2, \mathbf{T}, \Sigma_2)$ *be st-mappings, where* Σ_1, Σ_2 *are sets of* FO-TO-CQ *dependencies. Then the following statements are equivalent:*

(1) $\mathcal{M}_1 \preceq_{\mathrm{T}} \mathcal{M}_2$.

(2) *For every instance* J *of* $\mathbf{T}$*, if* J *is a universal solution for some instance under* $\mathcal{M}_1$*, then* J *is a universal solution for some instance under* $\mathcal{M}_2$.

The theorem is a particular case of the following general result. In the statement we use some terminology. We say that a mapping $\mathcal{M}$ *admits universal solutions* if for every instance of $I \in \mathrm{dom}(\mathcal{M})$ there exists an instance $J \in \mathrm{range}(\mathcal{M})$ that is a universal solution for I under $\mathcal{M}$. Moreover, $\mathcal{M}$ is *closed under homomorphism on* $\mathrm{range}(\mathcal{M})$ if every for every pair $(I, J) \in \mathcal{M}$ and every $J' \in \mathrm{range}(\mathcal{M})$, if there exists a homomorphism from J to J', then the pair (I, J') also belongs to $\mathcal{M}$.

LEMMA 6.4.5. *Let $\mathcal{M}_1$ from $\mathbf{R}_1$ to $\mathbf{R}$ and $\mathcal{M}_2$ from $\mathbf{R}_2$ to $\mathbf{R}$ be mappings that admit universal solutions, are closed under homomorphisms on $\mathrm{range}(\mathcal{M}_1)$ and $\mathrm{range}(\mathcal{M}_2)$, respectively, and are such that $\mathrm{range}(\mathcal{M}_1) = \mathrm{range}(\mathcal{M}_2)$. Then, the following are equivalent:*

(1) $\mathcal{M}_1 \preceq_{\mathrm{T}} \mathcal{M}_2$

(2) *For every instance $I \in \mathrm{dom}(\mathcal{M}_1)$ there exists an instance $K \in \mathrm{dom}(\mathcal{M}_2)$ such that the universal solutions for I under $\mathcal{M}_1$ and the universal solutions for K under $\mathcal{M}_2$ are homomorphically equivalent.*

PROOF. For the sake of readability, although there may be infinitely many universal solutions for a given instance I we use $\mathrm{Sol}^U_{\mathcal{M}}(I)$ to denote an arbitrary universal solution of I under $\mathcal{M}$. We do this without loss of generality, since we compare such solutions by using homomorphisms.

We first prove that (1) implies (2). Assume that $\mathcal{M}_1 \preceq_{\mathrm{T}} \mathcal{M}_2$. Then, there exists a mapping $\mathcal{N}$ from $\mathbf{R}_1$ to $\mathbf{R}_2$ such that $\mathcal{M}_1 = \mathcal{N} \circ \mathcal{M}_2$. Consider now an arbitrary instance $I \in \mathrm{dom}(\mathcal{M}_1)$. Then, $(I, \mathrm{Sol}^U_{\mathcal{M}_1}(I))$ clearly belongs to $\mathcal{M}_1$, and since $\mathcal{M}_1 = \mathcal{N} \circ \mathcal{M}_2$, there is an instance $K \in \mathrm{dom}(\mathcal{M}_2)$ such that $(I, K) \in \mathcal{N}$ and $(K, \mathrm{Sol}^U_{\mathcal{M}_1}(I)) \in \mathcal{M}_2$. We claim that $\mathrm{Sol}^U_{\mathcal{M}_1}(I)$ is universal for K under $\mathcal{M}_2$, which suffices for the proof, since every two universal solutions are homomorphically equivalent. For every pair $(K, J) \in \mathcal{M}_2$, since $(I, K) \in \mathcal{N}$, then $(I, J) \in \mathcal{M}_1$. Moreover, since $\mathrm{Sol}^U_{\mathcal{M}_1}(I)$ is universal for $\mathcal{M}_1$, there is a homomorphism from $\mathrm{Sol}^U_{\mathcal{M}_1}(I)$ to J. This proves our claim. Notice that we did not use the hypothesis $\mathrm{range}(\mathcal{M}_1) = \mathrm{range}(\mathcal{M}_2)$ in the proof of this direction.

We now prove that (2) implies (1). Assume that (2) holds, and define a mapping $\mathcal{N}$ from $\mathbf{R}_1$ to $\mathbf{R}_2$ as follows:

$$\mathcal{N} = \{(I, K) \mid I \in \mathrm{dom}(\mathcal{M}_1), K \in \mathrm{dom}(\mathcal{M}_2) \text{ and } \mathrm{Sol}^U_{\mathcal{M}_1}(I) \text{ is homomorphically equivalent to } \mathrm{Sol}^U_{\mathcal{M}_2}(K)\}.$$

We first prove that $\mathcal{M}_1 = \mathcal{N} \circ \mathcal{M}_2$. First, to prove that $\mathcal{M}_1 \subseteq \mathcal{N} \circ \mathcal{M}_2$, consider a pair $(I, J) \in \mathcal{M}_1$, and notice that $(I, \mathrm{Sol}^U_{\mathcal{M}_1}(I))$ also belongs to $\mathcal{M}_1$ (and that there is a homomorphism from $\mathrm{Sol}^U_{\mathcal{M}_1}(I)$ to J). Further, by the hypothesis (2), there exists an instance $K \in \mathrm{dom}(\mathcal{M}_2)$ such that $\mathrm{Sol}^U_{\mathcal{M}_1}(I)$ is homomorphically equivalent to $\mathrm{Sol}^U_{\mathcal{M}_2}(K)$. By the definition of $\mathcal{N}$, this in turn implies that (I, K) belongs to $\mathcal{N}$. Further, we know that $(K, \mathrm{Sol}^U_{\mathcal{M}_2}(K))$ belongs to $\mathcal{M}_2$, and since $\mathrm{Sol}^U_{\mathcal{M}_1}(I)$ and $\mathrm{Sol}^U_{\mathcal{M}_2}(K)$ are homomorphically equivalent, and we are assuming that $(I, J) \in \mathcal{M}_1$ we know there is a homomorphism from $\mathrm{Sol}^U_{\mathcal{M}_2}(K)$ to J. Finally, since $J \in \mathrm{range}(\mathcal{M}_1)$ and $\mathrm{range}(\mathcal{M}_1) = \mathrm{range}(\mathcal{M}_2)$, we have that $J \in \mathrm{range}(\mathcal{M}_2)$ and there exists a homomorphism from $\mathrm{Sol}^U_{\mathcal{M}_2}(K)$ to J, thus, by the closure property (on $\mathrm{range}(\mathcal{M}_2)$), we obtain that (K, J) belongs to $\mathcal{M}_2$. This proves that $\mathcal{M}_1 \subseteq \mathcal{N} \circ \mathcal{M}_2$.

Next, we prove that $\mathcal{N} \circ \mathcal{M}_2 \subseteq \mathcal{M}_1$. Assume that the pair (I, J) belongs to $\mathcal{N} \circ \mathcal{M}_2$. Then, there exists an instance $K \in \mathrm{dom}(\mathcal{M}_2)$ such that $(I, K) \in \mathcal{N}$ and $(K, J) \in \mathcal{M}_2$. The last assertion ensures that there exists a homomorphism from $\mathrm{Sol}^U_{\mathcal{M}_2}(K)$ to J. By the definition of $\mathcal{N}$, we have that $\mathrm{Sol}^U_{\mathcal{M}_1}(I)$ and $\mathrm{Sol}^U_{\mathcal{M}_2}(K)$ are homomorphically equivalent. Combining the last two facts, we obtain that there must be a homomorphism from $\mathrm{Sol}^U_{\mathcal{M}_1}(I)$ to J. Again, since $J \in \mathrm{range}(\mathcal{M}_2)$ and $\mathrm{range}(\mathcal{M}_1) = \mathrm{range}(\mathcal{M}_2)$, we have that $J \in \mathrm{range}(\mathcal{M}_1)$ and there exists a homomorphism from $\mathrm{Sol}^U_{\mathcal{M}_1}(I)$ to J, thus, by the closure property (on $\mathrm{range}(\mathcal{M}_1)$), we obtain that (I, J) belongs to $\mathcal{M}_1$. □

Theorem 6.4.4 follows directly from Lemma 6.4.5. Just notice that for mapping $\mathcal{M}_1$ and $\mathcal{M}_2$ in Theorem 6.4.4 we have that $\mathrm{range}(\mathcal{M}_1) = \mathrm{range}(\mathcal{M}_2) = \mathrm{Inst}(\mathbf{T})$, and both mappings are closed under target homomorphisms (ten Cate & Kolaitis, 2009).

The characterization in Theorem 6.4.4 supports our claim that $\preceq_{\mathrm{T}}$ measures the amount of information covered by a mapping. In fact, universal solutions have been identified as a fundamental class of solutions in data exchange, as they represent (in a precise sense) the entire space of solutions (Fagin, Kolaitis, Miller, & Popa, 2005; Fagin, Kolaitis, & Popa, 2005). Our characterization shows that if $\mathcal{M}_1 \preceq_{\mathrm{T}} \mathcal{M}_2$, then the space of possible universal solutions for $\mathcal{M}_1$ is contained in that of $\mathcal{M}_2$.

6.4.2. Target redundancy in schema mappings

There may exist many different ways to transfer the same information and, hence, metadata management systems should handle some criteria that help them in identifying the best alternatives, in terms of the resources they use. In this section, we introduce one such criteria, the notion of *target redundancy*. We use the following example to motivate our definition.

Example 6.4.6. Let $\mathcal{M}_1 = (\mathbf{R}, \mathbf{R}_1, \Sigma_1)$ and $\mathcal{M}_2 = (\mathbf{R}, \mathbf{R}_2, \Sigma_2)$ be mappings specified by dependencies $A(x) \rightarrow R(x)$ and $A(x) \rightarrow P(x, x)$, respectively, and where $\mathbf{R}$, $\mathbf{R}_1$ and $\mathbf{R}_2$ are ground schemas. It is easy to see that $\mathcal{M}_1 \equiv_{\mathrm{s}} \mathcal{M}_2$. However, $\mathcal{M}_1$ can be considered *better* than $\mathcal{M}_2$ in the sense that it does not *waste* resources when transferring information from $\mathbf{R}$. In fact, every instance in $\mathrm{range}(\mathcal{M}_1)$ is *essential* for $\mathcal{M}_1$, as it is a universal solution for an instance of $\mathbf{R}$ under $\mathcal{M}_1$. On the other hand, every universal solution of $\mathcal{M}_2$ can only contain tuples of the form $P(a, a)$, which implies that several instances in $\mathrm{range}(\mathcal{M}_2)$ are not universal for any source instance, and thus not essential for this mapping. □

As shown in Example 6.4.6, it would be advisable to design mappings for which every target instance is *essential* in transferring source information. In the following definition, we use the order $\preceq_{\mathrm{s}}$ to formalize this notion.

DEFINITION 6.4.7 (Target redundancy). *A mapping $\mathcal{M}$ is* target redundant *if there exists an instance $J^\star \in \mathrm{range}(\mathcal{M})$ such that $\mathcal{M}^\star = \{(I, J) \in \mathcal{M} \mid J \neq J^\star\}$ satisfies that $\mathcal{M}^\star \equiv_{\mathrm{s}} \mathcal{M}$.*

Thus, intuitively, we say that a mapping $\mathcal{M}$ is target redundant if we can remove a target instance from $\mathcal{M}$, and still be able to transfer the same amount of information. Correspondingly, we say that a mapping $\mathcal{M}$ is *target non-redundant* if we cannot remove any target instance from $\mathcal{M}$, and still be able to transfer the same amount of information.

Example 6.4.8. Consider mappings $\mathcal{M}_1$ and $\mathcal{M}_2$ in Example 6.4.6. $\mathcal{M}_1$ is target non-redundant, but $\mathcal{M}_2$ is target redundant as $\mathcal{M}_2 \equiv_{\mathrm{S}} \mathcal{M}^\star$, where $\mathcal{M}^\star$ is generated from $\mathcal{M}_2$ by removing from $\mathrm{range}(\mathcal{M}_2)$ an arbitrary instance that contains a tuple $P(a, b)$ with $a \neq b$. Notice that if we add $P(x, y) \rightarrow x = y$ as a target constraint to $\mathcal{M}_2$, the resulting mapping is target non-redundant. □

Characterizing target redundancy

We provide in this section a characterization of the notion of target redundancy for mappings specified by FO-TO-CQ dependencies. But first, we shed light on the issue of how the use of null values generate redundant information.

Null values are used in data exchange to deal with incomplete information. For example, assume that one needs to transfer data from a schema $\mathrm{Emp}_1(\cdot)$ storing a list of employee names, to a schema $\mathrm{Emp}_2(\cdot, \cdot)$ storing a list of employee names and the departments where they work. Given that the source schema does not contain any information about departments, one has to use a dependency of the form $\mathrm{Emp}_1(x) \rightarrow \exists y\, \mathrm{Emp}_2(x, y)$. Thus, when exchanging data, a null value n is included in a tuple $\mathrm{Emp}_2(a, n)$ if one does not know the department where employee a works. Null values introduce redundant information, as they allow one to represent the same data in many different ways. For example, a target instance $\mathrm{Emp}_2(a, n)$ contains exactly the same information as a target instance $\mathrm{Emp}_2(a, n')$ if n and n' are null values. Thus, instance $\mathrm{Emp}_2(a, n')$ is really not needed when transferring source data. In fact, the next result shows that every st-mapping specified by FO-TO-CQ dependencies that allows null values in the target schema is target redundant (recall that we use the term st-mapping for mappings that only have constants in their source instances, and constants and nulls in their target instances).

PROPOSITION 6.4.9. *Let $\mathcal{M}$ be an st-mapping from $\mathbf{S}$ to $\mathbf{T}$ specified by a set of* FO-TO-CQ *dependencies. Then $\mathcal{M}$ is target redundant.*

PROOF. Let $J^\star$ be an arbitrary instance of $\mathbf{T}$ containing at least one null value, and assume that $\mathcal{M}'$ is a mapping defined as $\{(I, J) \in \mathcal{M} \mid J \neq J^\star\}$. Next, we show that $\mathcal{M} \equiv_{\mathrm{s}} \mathcal{M}'$, which implies that $\mathcal{M}'$ is target redundant. First, it is straightforward to prove that $\mathcal{M}' \preceq_{\mathrm{s}} \mathcal{M}$, as $\mathcal{M} \circ \mathcal{N} = \mathcal{M}'$ with $\mathcal{N} = \{(J, J) \mid J \in \mathrm{Inst}(\mathbf{T})$ and $J \neq J^\star\}$. Thus, we only need to show that $\mathcal{M} \preceq_{\mathrm{s}} \mathcal{M}'$. Let $\mathcal{N}'$ be a mapping from $\mathbf{T}$ to $\mathbf{T}$ defined as follows:

$$\{(J_1, J_2) \in \mathrm{Inst}(\mathbf{T}) \times \mathrm{Inst}(\mathbf{T}) \mid \text{there exists an isomorphism } f \text{ from } J_1 \text{ to } J_2 \text{ that is the identity on the constants}\}.$$

Given that $\mathcal{M}$ is closed under isomorphism, no instance of $\mathbf{S}$ contains null values and $J^\star$ contains at least one null value, we have that $\mathcal{M}' \circ \mathcal{N}' = \mathcal{M}$. Therefore, we conclude that $\mathcal{M} \preceq_{\mathrm{s}} \mathcal{M}'$. □

It is important to notice that target redundancy does not mean that a mapping is poorly designed, as in some cases the redundancy, and in particular the use of null values, is unavoidable (like in the above mapping $\mathrm{Emp}_1(x) \rightarrow \exists y\, \mathrm{Emp}_2(x, y)$). Nevertheless, when mappings are specified by using dependencies without existential quantifiers in their conclusions, that is, full dependencies, there is no need to use null values as one does not need to deal with incomplete information. We provide in the following theorem a characterization of the notion of target redundancy for mappings specified by full dependencies that allow only constant values in source and target schemas.

THEOREM 6.4.10. *Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where $\mathbf{S}$ and $\mathbf{T}$ are ground schemas and Σ is a set of full* FO-TO-CQ *dependencies. Then the following properties are equivalent:*

(1) *$\mathcal{M}$ is target non-redundant.*

(2) *Every instance in $\mathrm{range}(\mathcal{M})$ is a universal solution for some instance in $\mathrm{dom}(\mathcal{M})$.*

Thus, our characterization shows that a mapping $\mathcal{M}$ defined by a set of full FO-TO-CQ dependencies is target non-redundant if and only if every instance in $\mathrm{range}(\mathcal{M})$ is essential for $\mathcal{M}$, as it is a universal solution for some instance in $\mathrm{dom}(\mathcal{M})$. Theorem 6.4.10 is a corollary of the following lemma, where we use the following terminology. Given a mapping $\mathcal{M}$ and an instance $I \in \mathrm{dom}(\mathcal{M})$, we say that $J \in \mathrm{Sol}_{\mathcal{M}}(I)$ is a *minimum* solution for I if for every instance $K \in \mathrm{Sol}_{\mathcal{M}}(I)$, it holds that $J \subseteq K$.

LEMMA 6.4.11. *Let $\mathcal{M}$ be a mapping such that every instance in $\mathrm{dom}(\mathcal{M})$ has a* minimum *solution. Then, the following properties are equivalent:*

(1) *Every instance in $\mathrm{range}(\mathcal{M})$ is a minimum solution for some instance in $\mathrm{dom}(\mathcal{M})$.*

(2) *$\mathcal{M}$ is target non-redundant.*

PROOF. (1) $\Rightarrow$ (2) Assume for the sake of contradiction that $\mathcal{M}$ is target redundant. Then, there exists an instance $J^\star \in \mathrm{range}(\mathcal{M})$ such that the mapping $\mathcal{M}^\star = \{(I, J) \in \mathcal{M} \mid J \neq J^\star\}$ satisfies $\mathcal{M}^\star \equiv_{\mathrm{s}} \mathcal{M}$. In particular, $\mathcal{M} \preceq_{\mathrm{s}} \mathcal{M}^\star$, and thus there exists a mapping $\mathcal{N}$ such that $\mathcal{M} = \mathcal{M}^\star \circ \mathcal{N}$.

Let now $I^\star \in \mathrm{dom}(\mathcal{M})$ be the instance such that $J^\star$ is a minimum solution for $I^\star$. Then, clearly, the pair $(I^\star, J^\star)$ belongs to $\mathcal{M}$. Since $\mathcal{M} = \mathcal{M}^\star \circ \mathcal{N}$, there must exist an instance $K \in \mathrm{range}(\mathcal{M}^\star)$ such that $(I^\star, K) \in \mathcal{M}^\star)$ and $(K, J^\star) \in \mathcal{N}$.

Notice that every instance present in $\mathrm{range}(\mathcal{M}^\star)$ belongs also to $\mathrm{range}(\mathcal{M})$, and thus K must be a minimum solution for some instance in $\mathrm{dom}(\mathcal{M})$ (since $K \in \mathrm{range}(\mathcal{M})$). Let I be the instance for which K is a minimum solution in $\mathcal{M}$. Then, $(I, K) \in \mathcal{M}$, and thus $(I, K) \in \mathcal{M}^\star$, since $K \neq J^\star$. We combine this with the fact that the pair $(K, J^\star)$ belongs to $\mathcal{N}$ and the fact that $\mathcal{M} = \mathcal{M}^\star \circ \mathcal{N}$ to conclude that $(I, J^\star)$ belongs to $\mathcal{M}$. Then, since K is minimum for I, it must be that $K \subseteq J^\star$.

On the other hand, since $(I^\star, K)$ belong to $\mathcal{M}^\star$, it must also be that $(I^\star, K)$ belong to $\mathcal{M}$, since $K \neq J^\star$. But since $J^\star$ is minimum for $I^\star$, we obtain that $J^\star \subseteq K$.

These two facts imply that $K = J$, which is a contradiction.

(2) $\Rightarrow$ (1) Assume that $\mathcal{M}$ is target non-redundant, and that every instance in $\mathrm{dom}(\mathcal{M})$ has a minimum solution. We now prove that every instance in $\mathrm{range}(\mathcal{M})$ is a minimum solution for some instance in $\mathrm{dom}(\mathcal{M})$.

Assume for the sake of contradiction that there is an instance $J^\star$ in $\mathcal{M}$ that is no minimum solution for any instance in $\mathrm{dom}(\mathcal{M})$. Construct a mapping $\mathcal{M}^\star$ as follows: $\mathcal{M}^\star = \{(I, J) \in \mathcal{M} \mid J \neq J^\star\}$. We now prove that $\mathcal{M}^\star \equiv_{\mathrm{s}} \mathcal{M}$, which contradicts the fact that $\mathcal{M}$ is non target redundant.

First we prove that $\mathcal{M}^\star \preceq_{\mathrm{s}} \mathcal{M}$ by constructing a mapping $\mathcal{N}_1$ such that $\mathcal{M}^\star = \mathcal{M} \circ \mathcal{N}_1$. Define then $\mathcal{N}_1$ as $\{(J, J) \mid J \in \mathrm{range}(\mathcal{M}^\star)\}$. The proof that $\mathcal{M}^\star = \mathcal{M} \circ \mathcal{N}_1$ follows easily.

Next, we prove that $\mathcal{M} \preceq_{\mathrm{s}} \mathcal{M}^\star$ by use of a mapping $\mathcal{N}_2$ such that $\mathcal{M} = \mathcal{M}^\star \circ \mathcal{N}_2$. Define $\mathcal{N}_2$ to contain all the pairs $\{(J, J) \mid J \in \mathrm{range}(\mathcal{M}^\star)\}$ plus the pairs $\{(J, J^\star) \mid (I, J^\star) \in \mathcal{M}$ and J is minimum for $I\}$.

To see that $\mathcal{M} \subseteq \mathcal{M}^\star \circ \mathcal{N}_2$, let $(I, J) \in \mathcal{M}$. there are two cases to consider:

- If $J \neq J^\star$, then $(I, J) \in \mathcal{M}^\star$, and $(J, J) \in \mathcal{N}_2$.
- If $J = J^\star$, then assume that K is minimum for I. It is clear that $(I, K) \in \mathcal{M}^\star$, and, then the pair (K, J) belongs to $\mathcal{N}_2$.

Next, let $(I, J) \in \mathcal{M}^\star \circ \mathcal{N}_2$. Then, there must exist an instance $K \in \mathrm{range}(\mathcal{M}^\star)$ such that $(I, K) \in \mathcal{M}^\star$, and $(K, J) \in \mathcal{N}_2$. We prove that $(I, J) \in \mathcal{M}$. From the definition of $\mathcal{N}_2$, there are two cases to consider:

- If $K = J$, then, from the construction of $\mathcal{N}_2$, $K \neq J^\star$, and thus if $(I, K) \in \mathcal{M}^\star$ then (I, K) must belong to $\mathcal{M}$.
- If $K \neq J$, then $J = J^\star$, and from the definition of $\mathcal{N}_2$ the pair $(I, J^\star)$ must belong to $\mathcal{M}$.

This completes the proof of the lemma. $\square$

Theorem 6.4.10 follows from the above lemma. Just notice that for a mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where $\mathbf{S}$ and $\mathbf{T}$ are ground schemas and Σ is a set of full FO-TO-CQ dependencies, and for every instance I of $\mathbf{S}$, the universal solution $\mathrm{chase}_{\Sigma}(I)$ is a minimum solution for I under $\mathcal{M}$.

6.4.3. Source redundancy

Just as there exists a symmetric definition for the order $\preceq_{\mathrm{S}}$, so is the case for the notion of target redundancy. In fact, we use the order $\preceq_{\mathrm{T}}$ in the following definition to introduce the notion of source redundancy, which also plays a fundamental role in providing foundations for metadata management.

DEFINITION 6.4.12 (Source redundancy). *A mapping $\mathcal{M}$ is* source redundant *if there exists an instance $I^{\star} \in \mathrm{dom}(\mathcal{M})$ such that $\mathcal{M}^{\star} = \{(I, J) \in \mathcal{M} \mid I \neq I^{\star}\}$ satisfies $\mathcal{M}^{\star} \equiv_{\mathrm{T}} \mathcal{M}$.*

That is, a mapping $\mathcal{M}$ is source redundant if one can eliminate an instance from $\mathrm{dom}(\mathcal{M})$ and still cover the same amount of target information. Not surprisingly, there is a tight relation between target and source redundancy.

PROPOSITION 6.4.13. *$\mathcal{M}$ is source redundant if and only if $\mathcal{M}^{-1}$ is target redundant.*

PROOF. Assume that $\mathcal{M}$ is source redundant. Then there exists an instance $I^{\star} \in \mathrm{dom}(\mathcal{M})$ such that $\mathcal{M}^{\star} = \{(I, J) \in \mathcal{M} \mid I \neq I^{\star}\}$ satisfies $\mathcal{M}^{\star} \equiv_{\mathrm{S}} \mathcal{M}$. Notice that $I^{\star} \in \mathrm{range}(\mathcal{M}^{-1})$. Moreover, from Proposition 6.4.3 we know that $(\mathcal{M}^{\star})^{-1} \equiv_{\mathrm{T}} \mathcal{M}^{-1}$. Thus we have that there exists an instance $I^{\star} \in \mathrm{range}(\mathcal{M}^{-1})$ such that the mapping $(\mathcal{M}^{\star})^{-1} = \{(J, I) \in \mathcal{M}^{-1} \mid I \neq I^{\star}\}$ satisfies $(\mathcal{M}^{\star})^{-1} \equiv_{\mathrm{T}} \mathcal{M}^{-1}$. This implies that $\mathcal{M}^{-1}$ is target redundant which was to be shown. The other direction is similar. □

Characterizing source redundancy

In this section we provide a characterization of source redundancy for the class of mappings specified by FO-TO-CQ dependencies. From the point of view of covering target

information, a non-redundant mapping should not assign the same space of solutions to two different source instances, as this means that one of them is not necessary. The following theorem shows that the notion of source redundancy captures this intuition.

THEOREM 6.4.14. *Let $\mathcal{M}$ be an st-mapping specified by a set of* FO-TO-CQ *dependencies. Then the following statements are equivalent:*

(1) *$\mathcal{M}$ is source non-redundant.*

(2) *For every pair of source instances I_1, I_2, if $I_1 \neq I_2$ then $\mathrm{Sol}_{\mathcal{M}}(I_1) \neq \mathrm{Sol}_{\mathcal{M}}(I_2)$.*

The Theorem is a corollary of the following result.

LEMMA 6.4.15. *Let $\mathcal{M}$ be a mapping that admits universal solutions, and that is closed under homomorphisms in $\mathrm{range}(\mathcal{M})$. Then, $\mathcal{M}$ is source redundant if an only if there exist two instances I and I' in $\mathrm{dom}(\mathcal{M})$ such that $I \neq I'$ and the universal solutions of I and I' under $\mathcal{M}$ are homomorphically equivalent.*

PROOF. In the proof we use $\mathrm{Sol}^U_{\mathcal{M}}(I)$ to denote an arbitrary universal solution for I under $\mathcal{M}$.

Only If. Assume that $\mathcal{M}$ is source redundant, and let $I^\star$ be an instance of $\mathrm{dom}(\mathcal{M})$ such that the mapping $\mathcal{M}^\star = \{(I, J) \in \mathcal{M} \mid I \neq I^\star\}$ satisfies $\mathcal{M}^\star \equiv_{\mathrm{T}} \mathcal{M}$. Notice that $\mathcal{M}^\star$ is closed under homomorphisms in $\mathrm{range}(\mathcal{M}^\star)$. Since $\mathcal{M}^\star \equiv_{\mathrm{T}} \mathcal{M}$, then in particular $\mathcal{M} \preceq_{\mathrm{T}} \mathcal{M}^\star$, and by Lemma 6.4.5, it must be the case that for every $I \in \mathrm{dom}(\mathcal{M})$ there exists an instance $K \in \mathrm{dom}(\mathcal{M}^\star)$ such that $\mathrm{Sol}^U_{\mathcal{M}}(I)$ is homomorphically equivalent to $\mathrm{Sol}^U_{\mathcal{M}^\star}(K)$. Then, there exists an instance $K^\star \in \mathrm{dom}(\mathcal{M}^\star)$ such that $\mathrm{Sol}^U_{\mathcal{M}}(I^\star)$ is homomorphically equivalent to $\mathrm{Sol}^U_{\mathcal{M}^\star}(K^\star)$. Notice that $I^\star \neq K^\star$ since $K^\star \in \mathrm{dom}(\mathcal{M}^\star)$ and $I^\star \notin \mathrm{dom}(\mathcal{M}^\star)$. Moreover, $\mathrm{Sol}_{\mathcal{M}^\star}(K^\star) = \mathrm{Sol}_{\mathcal{M}}(K^\star)$. Thus we have that $I^\star \neq K^\star$ and the universal solutions of $I^\star$ are homomorphically equivalent to the universal solutions of $K^\star$ under $\mathcal{M}$, which completes this part of the proof.

If. Assume that there exist two instances $I_1 \neq I_2$ such that $\mathrm{Sol}^U_{\mathcal{M}}(I_1)$ is homomorphically equivalent to $\mathrm{Sol}^U_{\mathcal{M}}(I_2)$. Notice that since $\mathcal{M}$ is closed under homomorphism in $\mathrm{range}(\mathcal{M})$ we have that $\mathrm{Sol}_{\mathcal{M}}(I_1) = \mathrm{Sol}_{\mathcal{M}}(I_2)$. We prove next that $\mathcal{M}$ is source redundant.

Let us define the mapping $\mathcal{M}^\star$ as $\mathcal{M}^\star = \{(I, J) \in \mathcal{M} \mid I \neq I_2\}$. We shall prove that $\mathcal{M} \equiv_{\mathrm{T}} \mathcal{M}^\star$, which proves that $\mathcal{M}$ is source redundant. First, it is easy to see that $\mathcal{M}^\star \preceq_{\mathrm{T}} \mathcal{M}$, thus we only need to prove that $\mathcal{M} \preceq_{\mathrm{T}} \mathcal{M}^\star$. Now, notice that $\mathcal{M}^\star$ is closed under homomorphisms in $\mathrm{range}(\mathcal{M}^\star)$ and, since $\mathrm{Sol}_{\mathcal{M}}(I_1) = \mathrm{Sol}_{\mathcal{M}}(I_2)$ we also have that $\mathrm{range}(\mathcal{M}^\star) = \mathrm{range}(\mathcal{M})$. Thus, we can use Lemma 6.4.5 to prove $\mathcal{M} \preceq_{\mathrm{S}} \mathcal{M}^\star$. Let $I \in \mathrm{dom}(\mathcal{M})$. Then, clearly, there exists $K \in \mathrm{dom}(\mathcal{M}^\star)$ such that $\mathrm{Sol}^U_{\mathcal{M}}(I)$ is homomorphically equivalent to $\mathrm{Sol}^U_{\mathcal{M}^\star}(K)$: it suffices to consider $K = I$ if $I \neq I_2$, and $K = I_1$ if $I = I_2$. This completes the proof of the Lemma. □

Property (2) above is called *unique-solutions* property in (Fagin, 2007), where it is shown to be a necessary condition for Fagin-invertibility.

6.5. Concluding Remarks

In this chapter, we have developed a theory to compare schema mappings in terms of notions of information and redundancy. In particular we have introduced the order $\preceq_{\mathrm{S}}$ as a measure of the amount of information transferred by a schema mapping, and studied some of its fundamental properties. From the order $\preceq_{\mathrm{S}}$ we have derived several other criteria to compare mappings and we provide tools to deal with these criteria. We introduced the notion of *target redundancy* and showed that it captures the intuition of using the *exact amount of resources* needed to transfer information using a schema mapping. Furthermore, to complement our information framework, we devise two additional concepts that allow us to compare mappings that share the same target schema. Symmetrically to the definition of $\preceq_{\mathrm{S}}$, we introduced the order $\preceq_{\mathrm{T}}$, that intuitively measures the amount of information *covered* by a mapping, as well as a notion of *source redundancy*. We provided characterizations for all the proposed notions, and show that together they can be used as a powerful framework to study metadata management operators. As a proof of concept, we showed how the machinery developed can be used to study some metadata management problems in the context of data exchange.

We use all the machinery developed in this chapter to study more complex metadata management operators in Chapter 7.

7. THE EXTRACT AND MERGE OPERATORS

In this chapter we use all the machinery for the concepts of information and redundancy developed in Chapter 6 to revisit the semantics of the *extract* operator (Melnik, 2004; Melnik et al., 2005), that intuitively captures the idea of *upgrading* a legacy schema. We formalize this operator in terms of the notions developed in Chapter 6, and we provide an algorithm for computing it for a class of mappings that includes the mappings specified by st-tgds. Moreover, we also study the *merge* operator, that as well as the extract operator, has been identified as fundamental for the development of a metadata management framework.

7.1. The Extract Operator

Consider a mapping $\mathcal{M}$ between schemas $\mathbf{S}$ and $\mathbf{T}$, and assume that $\mathbf{S}$ is the schema of a database that is only being used to map data through $\mathcal{M}$. In general, not all the information of $\mathbf{S}$ *participates* in the mapping and, thus, it is natural to ask whether one can *upgrade* $\mathbf{S}$ into a new schema that stores only the information being mapped by $\mathcal{M}$, that is, whether one can *extract* from $\mathbf{S}$ the portion of the schema that is actually participating in $\mathcal{M}$. This is the intended meaning of the *extract operator* (Melnik, 2004; Melnik et al., 2005), as shown in the following example.

Example 7.1.1. Let $\mathbf{S} = \{P(\cdot,\cdot), R(\cdot,\cdot), S(\cdot,\cdot)\}$ and $\mathbf{T} = \{T(\cdot,\cdot), U(\cdot,\cdot), V(\cdot,\cdot,\cdot)\}$, and assume that $\mathbf{S}$ is a ground schema. Consider a mapping $\mathcal{M}$ from $\mathbf{S}$ to $\mathbf{T}$ given by the following dependencies:

$$P(x,y) \rightarrow \exists u\, T(x,u) \wedge U(x,x) \tag{7.1}$$

$$P(x,y) \wedge R(y,z) \rightarrow \exists v\, V(x,y,v) \tag{7.2}$$

The first column of P is being transferred from the source by dependency (7.1), while all the tuples in P that can be joined with some tuples in R are being transferred by dependency (7.2). Moreover, notice that relation S is not participating at all in the mapping.

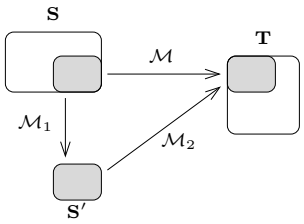


FIGURE 7.1. $(\mathcal{M}_1, \mathcal{M}_2)$ is an EXTRACT of $\mathcal{M}$.

A natural way to upgrade $\mathbf{S}$, and store only the data that is transferred by $\mathcal{M}$, is to have a new ground schema $\mathbf{S}' = \{P_1(\cdot), P_2(\cdot,\cdot)\}$, where relation $P_1(\cdot)$ is used to store the first component of P, and relation $P_2(\cdot,\cdot)$ is used to store the tuples in P that can be joined with some tuples in R. But we can do even better. Notice that by the intended meaning of relations P_1 and P_2, one knows that they must satisfy the inclusion dependency $P_2(x,y) \rightarrow P_1(x)$. Thus, schema $\mathbf{S}'$ plus this dependency still have enough capacity to store all the source information being transferred by $\mathcal{M}$. □

Given a mapping $\mathcal{M}$ from a schema $\mathbf{S}$ to a schema $\mathbf{T}$, the idea of the *extract* operator is to create a new source schema $\mathbf{S}'$ that captures *exactly* the information that is participating in $\mathcal{M}$ and *no other information* (Melnik, 2004; Melnik et al., 2005). As shown in Figure 7.1, a solution for the extract operator has two components, a mapping $\mathcal{M}_1$ from $\mathbf{S}$ to $\mathbf{S}'$ that drives the *migration* from the old to the new source schema, and a mapping $\mathcal{M}_2$ from $\mathbf{S}'$ to $\mathbf{T}$ that states how data should be mapped from the new source schema to the target schema. But what are the conditions that have to be imposed on mappings $\mathcal{M}_1$ and $\mathcal{M}_2$ (and schema $\mathbf{S}'$) to capture the intuition behind the extract operator? A set of such conditions was proposed by Melnik et al. in (Melnik, 2004; Melnik et al., 2005). In what follows, we show that the machinery developed in Chapter 6 can be used to provide a natural semantics for the extract operator. We compare our proposal with that of Melnik et al. (2005) in Section 7.1.2.

Assume that $\mathcal{M}$, $\mathcal{M}_1$ and $\mathcal{M}_2$ are the mappings shown in Figure 7.1. The first condition that we impose on $\mathcal{M}_1$ and $\mathcal{M}_2$, to consider them a valid extract of $\mathcal{M}$, is that the composition of $\mathcal{M}_1$ and $\mathcal{M}_2$ is equal to $\mathcal{M}$:

(E1) $\mathcal{M}_1 \circ \mathcal{M}_2 = \mathcal{M}$.

In this way, one ensures that for every instance I of $\mathbf{S}$, if one first *migrates* I from $\mathbf{S}$ to $\mathbf{S}'$, and then maps the result to $\mathbf{T}$, then one obtains exactly the same space of possible solutions as if I is being mapped by using the initial mapping $\mathcal{M}$. Notice that (E1) does not impose any restrictions over $\mathcal{M}_1$ and $\mathcal{M}_2$ alone. We do that with the next conditions.

The intended meaning of the extract operator is to store in a new schema exactly the information that is being transferred by the initial mapping. Thus, we require that $\mathcal{M}_1$ transfers from $\mathbf{S}$ to $\mathbf{S}'$ the same amount of source information as $\mathcal{M}$. Similarly, since $\mathcal{M}_2$ is used as the new way of mapping the information from $\mathbf{S}'$, we require that $\mathcal{M}_2$ covers exactly the same target information as $\mathcal{M}$. Thus, we impose the following condition on $\mathcal{M}_1$ and $\mathcal{M}_2$:

(E2) $\mathcal{M}_1 \equiv_{\mathrm{s}} \mathcal{M}$ and $\mathcal{M}_2 \equiv_{\mathrm{T}} \mathcal{M}$.

To complete the description of the extract operator, we only need a condition that captures the *optimality* of the new source schema. To do this, we do not impose an explicit condition on this schema, but instead we impose conditions over the range of $\mathcal{M}_1$ and the domain of $\mathcal{M}_2$. Notice that, although we require $\mathcal{M}_1$ to transfer exactly the same source information as $\mathcal{M}$, this mapping can be *redundant* and store the data in $\mathbf{S}'$ in a suboptimal way. Thus, we require $\mathcal{M}_1$ to be target non-redundant, as well as $\mathcal{M}_2$ to be source non-redundant. In that way, we force $\mathrm{range}(\mathcal{M}_1)$ and $\mathrm{dom}(\mathcal{M}_2)$ to be *minimal*, since one cannot lose an instance from $\mathrm{range}(\mathcal{M}_1)$ or $\mathrm{dom}(\mathcal{M}_2)$, and still obtain mappings that fulfill conditions (E1) and (E2). Thus, our last condition is:

(E3) $\mathcal{M}_1$ is target non-redundant and $\mathcal{M}_2$ is source non-redundant.

Notice that it is not difficult to show that under the above conditions, it holds that $\mathrm{range}(\mathcal{M}_1) = \mathrm{dom}(\mathcal{M}_2)$.

We finally have all the necessary ingredients to define the semantics of the extract operator.

DEFINITION 7.1.2 (Extract operator). $(\mathcal{M}_1, \mathcal{M}_2)$ *is an* extract *of* $\mathcal{M}$ *if* $\mathcal{M}_1$ *and* $\mathcal{M}_2$ *satisfy conditions (E1), (E2), and (E3).*

Example 7.1.3. Consider schemas $\mathbf{S}$, $\mathbf{S}'$, $\mathbf{T}$, and mapping $\mathcal{M}$ from Example 7.1.1. Let Σ_1 be the set that consists of dependencies:

$$\begin{aligned} P(x,y) &\rightarrow P_1(x), \\ P(x,y) \wedge R(y,z) &\rightarrow P_2(x,y), \end{aligned}$$

Σ_2 the set that consists of:

$$\begin{aligned} P_1(x) &\rightarrow \exists u\, T(x,u) \wedge U(x,x), \\ P_2(x,y) &\rightarrow \exists v\, V(x,y,v), \end{aligned}$$

and $\Gamma_{\mathbf{S}'}$ the set containing the inclusion dependency over $\mathbf{S}'$:

$$P_2(x,y) \rightarrow P_1(x).$$

Consider now the mappings $\mathcal{M}_1 = (\mathbf{S}, \mathbf{S}', \Sigma_1 \cup \Gamma_{\mathbf{S}'})$, and $\mathcal{M}_2 = (\mathbf{S}', \mathbf{T}, \Sigma_2 \cup \Gamma_{\mathbf{S}'})$. Then it can be shown that $(\mathcal{M}_1, \mathcal{M}_2)$ is an extract of $\mathcal{M}$. □

7.1.1. Computing the extract operator

Two fundamental questions about any schema-mapping management operator are for which classes of mappings is the operator defined, and how can it be computed. In this section, we provide answers to both questions for the class of mappings specified by FO-TO-CQ dependencies, as we provide an algorithm that, given a mapping $\mathcal{M}$ specified by a set of FO-TO-CQ dependencies, computes an extract $(\mathcal{M}_1, \mathcal{M}_2)$ of $\mathcal{M}$.

To present our algorithm, we need to introduce some terminology. In what follows, we use a procedure COMPOSE that given pairwise disjoint schemas $\mathbf{S}_1$, $\mathbf{S}_2$, $\mathbf{S}_3$, a set Σ_1 of dependencies from $\mathbf{S}_1$ to $\mathbf{S}_2$ and a set Σ_2 of dependencies from $\mathbf{S}_2$ to $\mathbf{S}_3$, computes a set Σ of dependencies from $\mathbf{S}_1$ to $\mathbf{S}_3$ such that $(I, J) \models \Sigma$ if and only if there exists K such that

$(I, K) \models \Sigma_1$ and $(K, J) \models \Sigma_2$. That is, COMPOSE(Σ_1, Σ_2) returns a set of dependencies Σ specifying a mapping that represents the composition of the mappings specified by Σ_1 and Σ_2. As pointed out by Nash et al. (2005) and by Melnik et al. (2005), there exists a straightforward implementation of COMPOSE when Σ_1 and Σ_2 are sets of FO sentences; if $\Sigma_1 = \{\sigma_1, \ldots, \sigma_n\}$, $\Sigma_2 = \{\gamma_1, \ldots, \gamma_m\}$ are set of FO-sentences, and $\mathbf{S}_2 = \{S_1, \ldots, S_k\}$, then a set Σ consisting of second-order dependency $\exists S_1 \cdots \exists S_k \, (\sigma_1 \wedge \cdots \wedge \sigma_n \wedge \gamma_1 \wedge \cdots \wedge \gamma_m)$ satisfies the above condition.

It should be noticed that second-order quantification is unavoidable to express the composition of mappings specified by FO dependencies, even for the case of st-tgds (Fagin, Kolaitis, Popa, & Tan, 2005). In what follows, we use COMPOSE as a black box, which could have been implemented by considering the idea shown above and the techniques presented in (Fagin, Kolaitis, Popa, & Tan, 2005; Nash et al., 2005; Melnik et al., 2005; Arenas, Fagin, & Nash, 2010). In particular, we use COMPOSE in Step 2 of the following algorithm to create constraints that eliminate the redundancy of mappings In the algorithm we also use procedure QUERYREWRITING that given a mapping $\mathcal{M}$ and a conjunctive query Q over the target, computes a source rewriting of Q under $\mathcal{M}$ (see Lemma 3.3.1 for the details of algorithm QUERYREWRITING).

Algorithm EXTRACT($\mathcal{M}$)

Input: An st-mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$, where Σ is a set of FO-TO-CQ dependencies.

Output: An extract $(\mathcal{M}_1, \mathcal{M}_2)$ of $\mathcal{M}$.

(1) Construct sets Σ_1, Σ_2 of dependencies, and a ground schema $\mathbf{R}$ as follows. For every $\varphi(\bar{x}) \rightarrow \psi(\bar{x}) \in \Sigma$, where $\bar{x}$ is an n-ary tuple of variables, repeat the following:
   (a) Include a fresh n-ary relational symbol R into $\mathbf{R}$.
   (b) Let $\alpha(\bar{x})$ be a formula in FO that is the output of QUERYREWRITING$(\mathcal{M}, \psi(\bar{x}))$.
   (c) Include dependency $\alpha(\bar{x}) \rightarrow R(\bar{x})$ into Σ_1 and dependency $R(\bar{x}) \rightarrow \psi(\bar{x})$ into Σ_2.

(2) Construct a set of formulas $\Gamma_{\mathbf{R}}$ over $\mathbf{R}$ as follows.

(a) Let $\widehat{\mathbf{R}} = \{\widehat{R} \mid R \in \mathbf{R}\}$ and Σ_1^- be the set of dependencies $\{\widehat{R}(\bar{x}) \rightarrow \beta(\bar{x}) \mid \beta(\bar{x}) \rightarrow R(\bar{x}) \in \Sigma_1\}$.

(b) Let Σ' be an SO-formula over $\mathbf{R} \cup \widehat{\mathbf{R}}$ that is the output of COMPOSE(Σ_1^-, Σ_1).

(c) Let $\Gamma_{\mathbf{R}}$ be the set of formulas over $\mathbf{R}$ obtained from Σ' by replacing every symbol $\widehat{R} \in \widehat{\mathbf{R}}$ by R.

(3) Let $\mathcal{M}_1 = (\mathbf{S}, \mathbf{R}, \Sigma_1 \cup \Gamma_{\mathbf{R}})$ and $\mathcal{M}_2 = (\mathbf{R}, \mathbf{T}, \Sigma_2 \cup \Gamma_{\mathbf{R}})$. Return $(\mathcal{M}_1, \mathcal{M}_2)$.

□

THEOREM 7.1.4. EXTRACT$(\mathcal{M})$ *returns an extract of* $\mathcal{M}$.

PROOF. Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ be an st-mapping with $\mathbf{S}$ a ground schema and Σ a set of FO-TO-CQ dependencies. Assume that $(\mathcal{M}_1, \mathcal{M}_2)$ is the output of EXTRACT$(\mathcal{M})$ with $\mathcal{M}_1 = (\mathbf{S}, \mathbf{R}, \Sigma_1 \cup \Gamma_{\mathbf{R}})$ and $\mathcal{M}_2 = (\mathbf{R}, \mathbf{T}, \Sigma_2 \cup \Gamma_{\mathbf{R}})$. We need to show that $(\mathcal{M}_1, \mathcal{M}_2)$ is an extract of $\mathcal{M}$. Thus, we need to show that $\mathcal{M}_1$ is non target redundant and $\mathcal{M}_1 \equiv_{\mathrm{s}} \mathcal{M}$, that $\mathcal{M}_2$ is non source redundant and $\mathcal{M}_2 \equiv_{\mathrm{T}} \mathcal{M}$, and that $\mathcal{M}_1 \circ \mathcal{M}_2 = \mathcal{M}$.

We show next that $\mathcal{M}_1$ is non target redundant by using Lemma 6.4.11. We show first that for every instance I in $\mathbf{S}$ it holds that $\mathrm{chase}_{\Sigma_1}(I)$ is a minimum solution for I. To show this we need to show the following two properties:

(a) $\mathrm{chase}_{\Sigma_1}(I) \in \mathrm{Sol}_{\mathcal{M}_1}(I)$, and that

(b) for every other solution $J \in \mathrm{Sol}_{\mathcal{M}_1}(I)$ we have that $\mathrm{chase}_{\Sigma_1}(I) \subseteq J$.

Notice that property (a) is non-trivial since $\mathcal{M}_1$ is defined using constraints over schema $\mathbf{R}$. Thus, although it is straightforward that $(I, \mathrm{chase}_{\Sigma_1}(I)) \models \Sigma_1$, we still need to show that $\mathrm{chase}_{\Sigma_1}(I) \models \Gamma_{\mathbf{R}}$. Recall that $\mathbf{R}$ is a ground schema. Moreover, since Σ_1 is a set of full dependencies we have that $\mathrm{chase}_{\Sigma_1}(I)$ is a ground instance. We show now that $\mathrm{chase}_{\Sigma_1}(I) \models \Gamma_{\mathbf{R}}$.

Let Σ_1^- and Σ' be the sets constructed in the algorithm. That is, Σ_1^- is obtained from Σ_1 by *reversing the arrows* and replacing every $R \in \mathbf{R}$ by $\widehat{R}$, and Σ' is the output of COMPOSE(Σ_1^-, Σ_1), that is, it is logically equivalent to $\Sigma_1^- \circ \Sigma_1$. Thus, $\Gamma_{\mathbf{R}}$ is the set obtained from Σ' replacing $\widehat{R}$ by R. Let $J^\star$ be the instance obtained from $\mathrm{chase}_{\Sigma_1}(I)$ by replacing

every $R \in \mathbf{R}$ by $\widehat{R}$. Notice that $\mathrm{chase}_{\Sigma_1}(I) \models \Gamma_{\mathbf{R}}$ if and only if $(J^\star, \mathrm{chase}_{\Sigma_1}(I)) \models \Sigma_1^- \circ \Sigma_1$. We show next this last property. For this we show that $(J^\star, I) \models \Sigma_1^-$, and thus, since $(I, \mathrm{chase}_{\Sigma_1}(I)) \models \Sigma_1$ we have that $(J^\star, \mathrm{chase}_{\Sigma_1}(I)) \models \Sigma_1^- \circ \Sigma_1$, and consequently $\mathrm{chase}_{\Sigma_1}(I) \models \Gamma_{\mathbf{R}}$. We need to show that $(J^\star, I) \models \sigma$ for every $\sigma \in \Sigma_1^-$. Let σ be a dependency of the form $\widehat{R}(\bar{x}) \rightarrow \alpha(\bar{x})$ and assume that $J^\star \models \widehat{R}(\bar{a})$ for some tuple $\bar{a}$. Then we have that $\mathrm{chase}_{\Sigma_1}(I) \models R(\bar{a})$. Notice that in Σ_1 there exists a dependency $\alpha(\bar{x}) \rightarrow R(\bar{x})$, and moreover the relation symbol R occurs only in this dependency. Thus, since $\mathrm{chase}_{\Sigma_1}(I) \models R(\bar{a})$ we know that $I \models \alpha(\bar{a})$. We have shown that $(J^\star, I) \models \sigma$ for every $\sigma \in \Sigma_1^-$ which was to be shown. The property (b) follows directly from the properties of the chase and the fact that Σ_1 is a set of full dependencies. Finally, we have shown that for every instance I the instance $\mathrm{chase}_{\Sigma_1}(I)$ is a minimum solution for I.

We show now that every solution $J \in \mathrm{range}(\mathcal{M}_1)$ is a minimum solution. In particular, we show that for every $J \in \mathrm{range}(\mathcal{M}_1)$ there exists an instance $I \in \mathrm{dom}(\mathcal{M}_1)$ such that $J = \mathrm{chase}_{\Sigma_1}(I)$. Thus, assume that $J \in \mathrm{range}(\mathcal{M}_1)$. Then we know that $J \models \Gamma_{\mathbf{R}}$. Let $J^\star$ be the instance obtained from J by replacing replacing every $R \in \mathbf{R}$ by $\widehat{R}$. Since $J \models \Gamma_{\mathbf{R}}$ we know that there exists an instance L such that $(J^\star, L) \models \Sigma_1^-$ and $(L, J) \models \Sigma_1$. We show now that $J = \mathrm{chase}_{\Sigma_1}(L)$. Let $R \in \mathbf{R}$ and assume that $\bar{a} \in R^{\mathrm{chase}_{\Sigma_1}(L)}$. We know that there exists a single dependency of the form $\alpha(\bar{x}) \rightarrow R(\bar{x})$ in Σ_1. Thus, since $\mathrm{chase}_{\Sigma_1}(L) \models R(\bar{a})$, necessarily $L \models \alpha(\bar{a})$. Moreover, since $(L, J) \models \Sigma_1$ we obtain that $J \models R(\bar{a})$ and thus, $\bar{a} \in R^J$. This shows that for every $R \in \mathbf{R}$ we have that $R^{\mathrm{chase}_{\Sigma_1}(L)} \subseteq R^J$. To show the opposite direction, let $R \in \mathbf{R}$ and assume that $\bar{a} \in R^J$. We know that there exists a dependency of the form $\widehat{R}(\bar{x}) \rightarrow \alpha(\bar{x})$ in Σ_1^-. Moreover, since $\bar{a} \in R^J$ we know that $\bar{a} \in \widehat{R}^{J^\star}$. Thus, since $(J^\star, L) \models \Sigma_1^-$ we obtain that $L \models \alpha(\bar{a})$. We also know that $\alpha(\bar{x}) \rightarrow R(\bar{x})$ is a dependency in Σ_1 then $\mathrm{chase}_{\Sigma_1}(L) \models R(\bar{a})$, and therefore $\bar{a} \in R^{\mathrm{chase}_{\Sigma_1}(L)}$. This shows that for every $R \in \mathbf{R}$ it holds that $R^J \subseteq R^{\mathrm{chase}_{\Sigma'}(L)}$.

We have shown that $\mathcal{M}_1$ is non target redundant. We show now that $\mathcal{M}_1 \equiv_{\mathrm{s}} \mathcal{M}$. First, recall that the set $\mathcal{C}_{\mathcal{M}}$ (see Lemmas 6.2.3 and 6.2.4) is constructed by including a source

rewriting for every conclusion of the dependencies defining $\mathcal{M}$. Consider now the non-ground schema $\widehat{\mathbf{R}} = \{\widehat{R} \mid R \in \mathbf{R}\}$ and the set $\widehat{\Sigma}_1$ obtained from Σ_1 replacing every $R \in \mathbf{R}$ by $\widehat{R}$. Let $\mathcal{M}' = (\mathbf{S}, \widehat{\mathbf{R}}, \widehat{\Sigma}_1)$. By the construction of Σ_1 and $\widehat{\Sigma}_1$ it is straightforward to prove that $\mathcal{C}_{\mathcal{M}} = \mathcal{C}_{\mathcal{M}'}$. Therefore, since $\widehat{\mathbf{R}}$ is non-ground we can use Lemma 6.2.4 to conclude that $\mathcal{M}' \equiv_{\mathrm{s}} \mathcal{M}$.

We show now that $\mathcal{M}' \equiv_{\mathrm{s}} \mathcal{M}_1$. Consider the set of formulas $\Lambda_1 = \{\widehat{R}(\bar{x}) \rightarrow R(\bar{x}) \mid R \in \mathbf{R}\}$ and the mapping $\mathcal{N}_1 = (\widehat{\mathbf{R}}, \mathbf{R}, \Lambda_1 \cup \Gamma_{\mathbf{R}})$. Notice that $\mathcal{N}_1$ is a mapping from a non-ground to a ground schema. Recall that $\mathcal{M}' = (\mathbf{S}, \widehat{\mathbf{R}}, \widehat{\Sigma}_1)$ where $\widehat{\Sigma}_1$ is a set of FO-TO-CQ dependencies. It is straightforward to see that Σ_1 is logically equivalent to $\widehat{\Sigma}_1 \circ \Lambda_1$, therefore, we obtain that $\mathcal{M}' \circ \mathcal{N}_1$ is a mapping from $\mathbf{S}$ to $\mathbf{R}$ specified by $\Sigma_1 \cup \Gamma_{\mathbf{R}}$, and thus $\mathcal{M}' \circ \mathcal{N}_1 = \mathcal{M}_1$ which implies that $\mathcal{M}_1 \preceq_{\mathrm{s}} \mathcal{M}'$. Consider now the set $\Lambda_2 = \{R(\bar{x}) \rightarrow \widehat{R}(\bar{x}) \mid R \in \mathbf{R}\}$ and the mapping $\mathcal{N}_2 = (\mathbf{R}, \widehat{\mathbf{R}}, \Lambda_2)$. Let I be an arbitrary instance in $\mathrm{Inst}(\mathbf{R})$ (thus I is ground). We have shown that $\mathrm{chase}_{\Sigma_1}(I)$ is a minimum solution for I under $\mathcal{M}_1$. Now let $J^\star$ be the instance of $\widehat{\mathbf{R}}$ obtained from $\mathrm{chase}_{\Sigma_1}(I)$ by replacing R by $\widehat{R}$. It is straightforward to see that $J^\star$ is a minimum solution for I under $\mathcal{M}_1 \circ \mathcal{N}_2$. Now notice that $\mathcal{N}_2$ is closed-up on the right, and thus $\mathcal{M}_1 \circ \mathcal{N}_2$ is also closed-up on the right. This implies that $\mathrm{Sol}_{\mathcal{M}_1 \circ \mathcal{N}_2}(I) = \{K \in \mathrm{Inst}\ \widehat{R} \mid J^\star \subseteq K\}$. Notice that by the construction of $J^\star$, we have that $K \in \mathrm{Sol}_{\mathcal{M}_1 \circ \mathcal{N}_2}(I)$ if and only if $(I, K) \models \widehat{\Sigma}_1$ where K is a non necessarily ground instance. We have shown that for every $I \in \mathrm{Inst}(\mathbf{R})$ it holds that $\mathrm{Sol}_{\mathcal{M}_1 \circ \mathcal{N}_2}(I) = \mathrm{Sol}_{\mathcal{M}'}(I)$ and thus, $\mathcal{M}_1 \circ \mathcal{N}_2 = \mathcal{M}'$ which implies that $\mathcal{M}' \preceq_{\mathrm{s}} \mathcal{M}_1$. Thus, we have that $\mathcal{M}_1 \preceq_{\mathrm{s}} \mathcal{M}'$ and $\mathcal{M}' \preceq_{\mathrm{s}} \mathcal{M}_1$, therefore, $\mathcal{M}_1 \equiv_{\mathrm{s}} \mathcal{M}'$, and since $\mathcal{M}' \equiv_{\mathrm{s}} \mathcal{M}$ we obtain that $\mathcal{M}_1 \equiv_{\mathrm{s}} \mathcal{M}$ completing this part of the proof.

Up to this point we have shown that $\mathcal{M}_1 \equiv_{\mathrm{s}} \mathcal{M}$ and that $\mathcal{M}_1$ is non target redundant. We show now that $\mathcal{M} = \mathcal{M}_1 \circ \mathcal{M}_2$. Consider the set of dependencies $\Sigma^\star$ created as follows. For every dependency $\varphi(\bar{x}) \rightarrow \psi(\bar{x})$ in Σ use QUERYREWRITING$(\mathcal{M}, \psi(\bar{x}))$ to obtain a formula $\alpha(\bar{x})$ that is a source rewriting of $\psi(\bar{x})$ under $\mathcal{M}$, and add the dependency $\alpha(\bar{x}) \rightarrow \psi(\bar{x})$ to $\Sigma^\star$. It is straightforward to show that Σ and $\Sigma^\star$ are logically equivalent, and thus, $(I, J) \in \mathcal{M}$ if and only id $(I, J) \models \Sigma^\star$. Also, by the construction of Σ_1 and Σ_2 it is easy to see that for every instance I of $\mathbf{S}$ we have that $\mathrm{chase}_{\Sigma^\star}(I)$ and

$\text{chase}_{\Sigma_2}(\text{chase}_{\Sigma_1}(I))$ are homomorphically equivalent. We have shown that $\text{chase}_{\Sigma_1}(I) \in \text{Sol}_{\mathcal{M}_1}(I)$ since $\text{chase}_{\Sigma_1}(I) \models \Gamma_{\mathbf{R}}$. Thus we have that $\text{chase}_{\Sigma_2}(\text{chase}_{\Sigma_1}(I)) \in \text{Sol}_{\mathcal{M}_1 \circ \mathcal{M}_2}(I)$, and moreover, $\text{chase}_{\Sigma_2}(\text{chase}_{\Sigma_1}(I))$ is a universal solution for I under $\mathcal{M}_1 \circ \mathcal{M}_2$. Notice that since Σ_1 is a set of full dependencies we have that $\mathcal{M}_1 \circ \mathcal{M}_2$ is closed under target homomorphisms. Thus, we have that $\text{chase}_{\Sigma^\star}(I)$ and $\text{chase}_{\Sigma_2}(\text{chase}_{\Sigma_1}(I))$ are homomorphically equivalent and are both universal solutions for I under $\mathcal{M}$ and under $\mathcal{M}_1 \circ \mathcal{M}_2$ respectively, and since $\mathcal{M}$ and $\mathcal{M}_1 \circ \mathcal{M}_2$ are closed under target homomorphism, we obtain that $\text{Sol}_{\mathcal{M}}(I) = \text{Sol}_{\mathcal{M}_1 \circ \mathcal{M}_2}(I)$ for every I. This implies that $\mathcal{M} = \mathcal{M}_1 \circ \mathcal{M}_2$ which was to be shown.

It only remains to prove that $\mathcal{M}_2 \equiv_{\text{T}} \mathcal{M}$ and that $\mathcal{M}_2$ is non source redundant. That is what we do next. We have already shown that $\mathcal{M}_1 \circ \mathcal{M}_2 = \mathcal{M}$ which implies that $\mathcal{M} \preceq_{\text{T}} \mathcal{M}_2$. Thus, to prove that $\mathcal{M}_2 \equiv_{\text{T}} \mathcal{M}$ we only have to show that there exists a mapping $\mathcal{N}$ such that $\mathcal{N} \circ \mathcal{M} = \mathcal{M}_2$. Consider the set of dependencies Σ_1' obtained from Σ_1 by *reversing the arrows*. Notice that the difference between Σ_1' and Σ_1^- is that in in Σ_1' we do not rename the relations in $\mathbf{R}$. Let $\mathcal{M}_1' = (\mathbf{R}, \mathbf{S}, \Sigma_1' \cup \Gamma_R)$. We prove first that $\mathcal{M}_2 \subseteq \mathcal{M}_1' \circ \mathcal{M}$. Assume that $(J, K) \in \mathcal{M}_2$, thus $J \models \Gamma_{\mathbf{R}}$ and $(J, K) \models \Sigma_2$. We need to prove that $(J, K) \in \mathcal{M}_1' \circ \mathcal{M}$. Notice that since Σ_2 is a set of st-tgds, we have that there exists a homomorphism from $\text{chase}_{\Sigma_2}(J)$ to K. Now, we have previously proved that if $J \models \Gamma_{\mathbf{R}}$ then there exists an instance I in $\mathbf{S}$ such that $J = \text{chase}_{\Sigma_1}(I)$. Notice that by the construction of Σ_1 and Σ_1' we have that $(\text{chase}_{\Sigma_1}(I), I) \models \Sigma_1'$, and then since $\text{chase}_{\Sigma_1}(I) \models \Gamma_{\mathbf{R}}$ we have that $(\text{chase}_{\Sigma_1}(I), I) \in \mathcal{M}_1'$. We prove next that $(I, \text{chase}_{\Sigma_2}(J)) \models \Sigma$. Notice that Σ is logically equivalent to the set $\Sigma^\star$ of the previous paragraph. Thus, it is enough to show that $(I, \text{chase}_{\Sigma_2}(J)) \models \Sigma^\star$. Assume that $I \models \alpha(\bar{a})$ for a dependency $\alpha(\bar{x}) \rightarrow \psi(\bar{x})$ in $\Sigma^\star$. We need to show that $\text{chase}_{\Sigma_2}(J) \models \psi(\bar{a})$. Thus, since $J = \text{chase}_{\Sigma_1}(I)$ we know that $J \models R(\bar{a})$ where $\alpha(\bar{x}) \rightarrow R(\bar{x})$ is the dependency in Σ_1 created from $\varphi(\bar{x}) \rightarrow \psi(\bar{x})$ in Σ. Moreover, we that that there is a dependency of the form $R(\bar{x}) \rightarrow \psi(\bar{x})$ in Σ_2, thus, since $J \models R(\bar{a})$ we obtain that $\text{chase}_{\Sigma_2}(J) \models \psi(\bar{a})$ which was to be shown. Thus we have that $(I, \text{chase}_{\Sigma_2}(J)) \models \Sigma^\star$ and then $(I, \text{chase}_{\Sigma_2}(J)) \models \Sigma$. Finally since Σ is closed under target homomorphism and there is a homomorphism from

$\mathrm{chase}_{\Sigma_2}(J))$ to K we have that $(I, K) \models \Sigma$ and thus $(I, K) \in \mathcal{M}$ We have shown that if $(J, K) \in \mathcal{M}_2$ then there exists an instance I such that $(J, I) \in \mathcal{M}_1'$ and $(I, K) \in \mathcal{M}$ which proves that $(J, K) \in \mathcal{M}_1' \circ \mathcal{M}$.

We show now that $\mathcal{M}_1' \circ \mathcal{M} \subseteq \mathcal{M}_2$. Assume that $(J, K) \in \mathcal{M}_1' \circ \mathcal{M}$, we need to prove that $(J, K) \in \mathcal{M}_2$. Notice that $(J, K) \in \mathcal{M}_2$ if and only if $J \models \Gamma_{\mathbf{R}}$ and $(J, K) \models \Sigma_2$. Since $(J, K) \in \mathcal{M}_1' \circ \mathcal{M}$ we know that $J \in \mathrm{dom}(\mathcal{M}_1')$ which implies that $J \models \Gamma_{\mathbf{R}}$. Thus, it only remains to prove that $(J, K) \models \Sigma_2$. Let $R(\bar{x}) \rightarrow \psi(\bar{x})$ in Σ_2 and assume that $J \models R(\bar{a})$. We need to prove that $K \models \psi(\bar{a})$. Since $R(\bar{x}) \rightarrow \psi(\bar{x}) \in \Sigma_2$ we know that there are dependencies $R(\bar{x}) \rightarrow \alpha(\bar{x}) \in \Sigma_1'$ and $\alpha(\bar{x}) \rightarrow \psi(\bar{x}) \in \Sigma^\star$. Now, since $(J, K) \in \mathcal{M}_1' \circ \mathcal{M}$ we know that there exists I such that $(J, I) \models \Sigma_1'$ and $(I, K) \models \Sigma^\star$. Thus, since $J \models R(\bar{a})$ we have that $I \models \alpha(\bar{a})$, and thus, $K \models \psi(\bar{a})$ which was to be shown. We have shown that if $(J, K) \in \mathcal{M}_1' \circ \mathcal{M}$ then $(J, K) \in \mathcal{M}_2$ and thus, $\mathcal{M}_1' \circ \mathcal{M} \subseteq \mathcal{M}_2$.

We have shown that $\mathcal{M}_2 \equiv_{\mathrm{T}} \mathcal{M}$. It only remains to prove that $\mathcal{M}_2$ is non source redundant. To prove this last property we make use of the characterization in Lemma 6.4.15. We show that for every pair of instances J and K in $\mathrm{dom}(\mathcal{M}_2)$, if $\mathrm{Sol}_{\mathcal{M}_2}(J) = \mathrm{Sol}_{\mathcal{M}_2}(K)$ then $J = K$. Notice that $\mathrm{Sol}_{\mathcal{M}_2}(J) = \mathrm{Sol}_{\mathcal{M}_2}(K)$ if and only if $\mathrm{chase}_{\Sigma_2}(J)$ is homomorphically equivalent to $\mathrm{chase}_{\Sigma_2}(K)$. We have shown (in the previous paragraph) that for every $J \in \mathrm{dom}(\mathcal{M}_2)$ there exists an instance I_J in $\mathbf{S}$ such that $\mathrm{chase}_{\Sigma_1}(I_J) = J$ and moreover $\mathrm{chase}_{\Sigma}(I_J)$ is homomorphically equivalent to $\mathrm{chase}_{\Sigma_2}(J)$. Similarly there exists an instance I_K such that $\mathrm{chase}_{\Sigma_1}(I_K) = K$ and $\mathrm{chase}_{\Sigma}(I_K)$ is homomorphically equivalent to $\mathrm{chase}_{\Sigma_2}(K)$. Thus, since $\mathrm{chase}_{\Sigma_2}(J)$ and $\mathrm{chase}_{\Sigma_2}(K)$ are homomorphically equivalent, we have that $\mathrm{chase}_{\Sigma}(I_J)$ is homomorphically equivalent with $\mathrm{chase}_{\Sigma}(I_K)$. Notice that this implies that for every conjunctive query Q over $\mathbf{T}$ it holds that $\mathrm{certain}_{\mathcal{M}}(Q, I_J) = \mathrm{certain}_{\mathcal{M}}(Q, I_K)$. Now let $\alpha(\bar{x}) \rightarrow R(\bar{x})$ be a dependency in Σ_1. We know that $\alpha(\bar{x})$ is a source rewriting of a formula $\varphi(\bar{x})$ under $\mathcal{M}$. Thus, $I_J \models \alpha(\bar{a})$ if and only if $I_K \models \alpha(\bar{a})$. This implies that $\mathrm{chase}_{\Sigma_1}(I_J) = \mathrm{chase}_{\Sigma_1}(I_K)$, obtaining that $J = K$ which is our desired property. This completes the proof of correctness of algorithm EXTRACT. □

7.1.2. On the semantics of the extract operator

The extract operator was considered by Melnik et al. (2004; 2005). Given a mapping $\mathcal{M}$ from a schema $\mathbf{S}$ to a schema $\mathbf{T}$, the output of this operator according to Melnik et al. is a mapping $\mathcal{M}_1$ from $\mathbf{S}$ to a schema $\mathbf{S}'$ together with the schema constraints $\Gamma_{\mathbf{S}'}$ that $\mathbf{S}'$ should satisfy. Moreover, the following two conditions should be satisfied by $\mathcal{M}_1$ (Melnik, 2004; Melnik et al., 2005): (1) $\mathcal{M}_1 \circ (\mathcal{M}_1)^{-1} \circ \mathcal{M}$ is equal to $\mathcal{M}$, and (2) $\mathrm{range}(\mathcal{M}_1)$ is the set of instances of $\mathbf{S}'$ that satisfy $\Gamma_{\mathbf{S}'}$. Notice that mapping $\mathcal{M}_2$ from $\mathbf{S}'$ to $\mathbf{T}$ is not part of the semantics by Melnik et al. (2004; 2005) since it can be obtained as $\mathcal{M}_2 = (\mathcal{M}_1)^{-1} \circ \mathcal{M}$.

Although our semantics for the extract operator was inspired by the work of Melnik et al. (2004; 2005), there are two features of Melnik et al.'s definition that limit its applicability, in particular if more expressive languages are used to specify mappings. First, if mapping $\mathcal{M}_1$ above is specified by a set of st-tgds (or, in general, by a set of FO-TO-CQ dependencies), then $\mathcal{M}_1 \circ (\mathcal{M}_1)^{-1}$ is a trivial mapping that contains all the pairs of instances from $\mathbf{S}$. Thus, $\mathcal{M}_1 \circ (\mathcal{M}_1)^{-1} \circ \mathcal{M}$ is also a trivial mapping in this case and, therefore, $\mathcal{M}_1 \circ (\mathcal{M}_1)^{-1} \circ \mathcal{M} = \mathcal{M}$ does not hold in general. This rules out the possibility of having natural solutions for the extract operator specified by st-tgds, as the one shown in Example 7.1.3. Second, in the semantics proposed by Melnik et al. (2005), no minimality restriction is imposed on the generated schema $\mathbf{S}'$, thus allowing redundant information. Moreover, in the semantics proposed by Melnik (2004), a minimality criterion based on *counting* the number of instances of a schema is imposed, which is only meaningful when instances are generated from a finite domain, and thus, not applicable in our context.

In view of the aforementioned limitations of the semantics of the extract operator proposed by Melnik et al. (2004; 2005), we have imposed some new conditions on this operator that try to capture the intuition behind it. In particular, we have used the notions of redundancy proposed in Section 6.4 to impose a minimality condition over the generated schemas. Moreover, we have imposed some conditions on the information transferred by the generated mappings, that ensure that condition (1) above holds when $(\cdot)^{-1}$ is replaced

by the notion of maximum recovery introduced in Chapter 3, which is a more natural notion of inverse when mappings are given by FO-TO-CQ.

PROPOSITION 7.1.5. *Let $\mathcal{M}$ be an st-mapping specified by a set of* FO-TO-CQ *dependencies, $(\mathcal{M}_1, \mathcal{M}_2)$ the output of* EXTRACT$(\mathcal{M})$ *and $\mathcal{M}_1^\star$ a maximum recovery of $\mathcal{M}_1$. Then it holds that $\mathcal{M}_1 \circ \mathcal{M}_1^\star \circ \mathcal{M} = \mathcal{M}$.*

PROOF. Consider the mapping $\mathcal{M}_1'$ constructed in the proof of Theorem 7.1.4 (we constructed that mapping when showing that $\mathcal{M}_2 \equiv_{\mathrm{T}} \mathcal{M}$ in the proof of Theorem 7.1.4). Recall that $\mathcal{M}_1' = (\mathbf{R}, \mathbf{S}, \Sigma_1' \cup \Gamma_{\mathbf{R}})$ where Σ_1' is the set obtained from Σ_1 by *reversing the arrows*. Notice that Σ_1' is exactly the set of dependencies obtained as output of the algorithm MAXIMUMRECOVERYFULL (presented in Section 3.3.3) if the input is the set Σ_1. Thus, by the properties of maximum recoveries proved in Proposition 3.1.6 we have that if $(I, J) \models \Sigma_1$ and $(J, K) \models \Sigma_1'$ then $\mathrm{Sol}_{\mathcal{M}_1}(K) \subseteq \mathrm{Sol}_{\mathcal{M}_1}(I)$. Thus, we have that if $(I, J) \models \Sigma_1$ and $(J, K) \models \Sigma_1'$ and $J \models \Gamma_{\mathbf{R}}$, then $\mathrm{Sol}_{\mathcal{M}_1}(K) \subseteq \mathrm{Sol}_{\mathcal{M}_1}(I)$. From the above discussion, we obtain that if $(I, K) \in \mathcal{M}_1 \circ \mathcal{M}_1'$ then $\mathrm{Sol}_{\mathcal{M}_1}(K) \subseteq \mathrm{Sol}_{\mathcal{M}_1}(I)$. Moreover, from the proof of correctness of algorithm MAXIMUMRECOVERYFULL we know that for every I in $\mathbf{S}$ it holds that $(\mathrm{chase}_{\Sigma_1}(I), I) \models \Sigma_1'$. Thus, since $\mathrm{chase}_{\Sigma_1}(I) \models \Gamma_{\mathbf{R}}$ for every I we obtain that $(I, \mathrm{chase}_{\Sigma_1}(I)) \in \mathcal{M}_1$ and $(\mathrm{chase}_{\Sigma_1}(I), I) \in \mathcal{M}_1'$ for every I in $\mathbf{S}$. We have shown that $(I, I) \in \mathcal{M}_1 \circ \mathcal{M}_1'$ and that if $(I, K) \in \mathcal{M}_1 \circ \mathcal{M}_1'$ it holds that $\mathrm{Sol}_{\mathcal{M}_1}(K) \subseteq \mathrm{Sol}_{\mathcal{M}_1}(I)$, which, by the properties of maximum recoveries proved in Proposition 3.1.6, implies that $\mathcal{M}_1'$ is a maximum recovery of $\mathcal{M}_1$.

In the proof of Theorem 7.1.4 we showed that $\mathcal{M}_1' \circ \mathcal{M} = \mathcal{M}_2$. We also showed that $\mathcal{M}_1 \circ \mathcal{M}_2 = \mathcal{M}$. These properties implies that $\mathcal{M}_1 \circ \mathcal{M}_1' \circ \mathcal{M} = \mathcal{M}$. Now, let $\mathcal{N}$ be an arbitrary maximum recovery of $\mathcal{M}_1$. Since $\mathcal{M}_1'$ is also a maximum recovery of $\mathcal{M}_1$ we have that $\mathcal{M}_1 \circ \mathcal{M}_1' = \mathcal{M}_1 \circ \mathcal{N}$. Finally, we obtain that for every $\mathcal{N}$ that is a maximum recovery of $\mathcal{M}_1$ it holds that $\mathcal{M}_1 \circ \mathcal{N} \circ \mathcal{M} = \mathcal{M}$. This completes the proof of the proposition. □

7.2. The Merge Operator

Consider two independent database schemas $\mathbf{S}_1$ and $\mathbf{S}_2$ and a mapping $\mathcal{M}$ between them, and assume that both schemas have materialized data that is being queried by several applications. Mapping $\mathcal{M}$ describes how data in these schemas is related, and, thus, the relationship stated by $\mathcal{M}$ leads to some *redundancy of storage*: there are *corresponding pieces of data* stored twice in these schemas. Although this corresponding data can be structured in different ways, and served different purposes in these schemas, it is natural to ask whether one can have a single *global schema* $\mathbf{S}$ that simultaneously stores the data of $\mathbf{S}_1$ and $\mathbf{S}_2$, (and no more than that), but that is not redundant in the storage of the shared information. This is the intuition behind the *merge operator* (Bernstein, Halevy, & Pottinger, 2000; Bernstein, 2003; Melnik, 2004), and, hence, we say that $\mathbf{S}$ is the result of *merging* $\mathbf{S}_1$ and $\mathbf{S}_2$ with respect to the relationship established by $\mathcal{M}$. A complete solution for the merge operator should also include mappings $\mathcal{M}_1$ and $\mathcal{M}_2$ from $\mathbf{S}$ to $\mathbf{S}_1$ and $\mathbf{S}_2$, respectively, that describe the relationship between the global schema and the initial schemas (Melnik, 2004; Melnik et al., 2005). These mappings ensure that an application that has used the initial schemas independently, would also be able to obtain the required data from the global schema. A diagram of the complete process is shown in Figure 7.2.

Example 7.2.1. Let $\mathbf{S}_1 = \{A(\cdot,\cdot)\}$ and $\mathbf{S}_2 = \{B(\cdot,\cdot)\}$ be ground schemas, and consider a mapping $\mathcal{M}$ given by dependency:

$$A(x,y) \quad \rightarrow \quad B(x,y).$$

This simple mapping states that all the tuples of relation A in $\mathbf{S}_1$ should also be part of relation B in $\mathbf{S}_2$. A natural way to store the information of both $\mathbf{S}_1$ and $\mathbf{S}_2$ in a non-redundant way is to consider a schema $\mathbf{S}$ with one relation $A'(\cdot,\cdot)$ storing all the information in A, and a new relation $D(\cdot,\cdot)$ storing the *difference* between B and A. By the intended meaning of relations A' and D, we know that they should satisfy the denial constraint:

$$\forall x \forall y \, \neg(A'(x,y) \wedge D(x,y)). \tag{7.3}$$

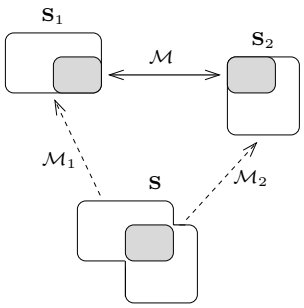


FIGURE 7.2. $(\mathcal{M}_1, \mathcal{M}_2)$ is a merge of $\mathcal{M}$.

In fact, schema $\mathbf{S}$ plus this constraint have enough capacity to store the information of both $\mathbf{S}_1$ and $\mathbf{S}_2$. Moreover, let $\mathcal{M}_1 = (\mathbf{S}, \mathbf{S}_1, \Sigma_1 \cup \Gamma_{\mathbf{S}})$ and $\mathcal{M}_2 = (\mathbf{S}, \mathbf{S}_2, \Sigma_2 \cup \Gamma_{\mathbf{S}})$, where Σ_1 consists of dependency:

$$A'(x, y) \quad \rightarrow \quad A(x, y),$$

Σ_2 consists of dependencies:

$$A'(x, y) \quad \rightarrow \quad B(x, y),$$
$$D(x, y) \quad \rightarrow \quad B(x, y),$$

and $\Gamma_{\mathbf{S}}$ is the set that consists of denial constraint (7.3). Then $\mathcal{M}_1$ and $\mathcal{M}_2$ can be used to relate the new schema $\mathbf{S}$ with schemas $\mathbf{S}_1$ and $\mathbf{S}_2$, respectively. □

In what follows, we propose a semantics for the merge operator using the machinery developed in Chapter 6. As for the case of the extract operator, we formalize the merge considering only the mappings $\mathcal{M}$, $\mathcal{M}_1$ and $\mathcal{M}_2$, as the schemas and schema constraints will be implicit in the mappings.

To define the semantics of the merge operator, we need to introduce the notion of $\mathcal{M}$-*confluence*, which is inspired by the notion of confluence proposed by Melnik et al. (2004; 2005). Let $\mathbf{S}$, $\mathbf{S}_1$, $\mathbf{S}_2$ be schemas, where $\mathbf{S}_1$ and $\mathbf{S}_2$ have no relation symbols in common, $\mathcal{M}_1$, $\mathcal{M}_2$ mappings from $\mathbf{S}$ to $\mathbf{S}_1$ and from $\mathbf{S}$ to $\mathbf{S}_2$, respectively, and $\mathcal{M}$ a mapping from $\mathbf{S}_1$ to $\mathbf{S}_2$. Then the $\mathcal{M}$-*confluence* of $\mathcal{M}_1$ and $\mathcal{M}_2$, denoted by $\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2$, is defined as

the following mapping from $\mathbf{S}$ to $\mathbf{S}_1 \cup \mathbf{S}_2$:

$$\{(I, J \cup K) \mid (J, K) \in \mathcal{M},\ (I, J) \in \mathcal{M}_1,\ (I, K) \in \mathcal{M}_2\},$$

where $J \cup K$ is the union of instances J and K, that is, $R^{J \cup K} = R^J$ for every $R \in \mathbf{S}_1$, and $S^{J \cup K} = S^K$ for every $S \in \mathbf{S}_2$. Intuitively, $\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2$ describes the *unified* instances $J \cup K$ that are valid pairs according to $\mathcal{M}$ and also simultaneously mapped from $\mathcal{M}_1$ and $\mathcal{M}_2$.

As pointed out before, $\mathbf{S}$ is a valid global schema for the merge of two schemas $\mathbf{S}_1$ and $\mathbf{S}_2$ related trough a mapping $\mathcal{M}$, if every instance I of $\mathbf{S}$ represents in a non-redundant way a valid unified instance of $\mathbf{S}_1$ and $\mathbf{S}_2$ according to $\mathcal{M}$. Then, if the pair $(\mathcal{M}_1, \mathcal{M}_2)$ of mappings from $\mathbf{S}$ to $\mathbf{S}_1$ and $\mathbf{S}$ to $\mathbf{S}_2$, respectively, is given as a solution for the merge operator, one can formalize this intuition by imposing conditions over $\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2$, and considering $\mathrm{dom}(\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2)$ as the new global schema. More precisely, we impose the following conditions for a solution of the merge of two schemas related by mapping $\mathcal{M}$:

(M1) $\mathrm{range}(\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2) = \{J \cup K \mid (J, K) \in \mathcal{M}\}$ and $\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2$ is target non-redundant.

(M2) $\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2$ is source non-redundant.

(M3) $\mathrm{dom}(\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2) = \mathrm{dom}(\mathcal{M}_1) = \mathrm{dom}(\mathcal{M}_2)$.

Condition (M1) indicates that every valid unified instance of $\mathcal{M}$ is covered in an *essential way* by $\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2$. Condition (M2) specifies that every instance in the global schema is necessary to cover the unified instances of $\mathcal{M}$. Finally, condition (M3) indicates that $\mathcal{M}_1$ and $\mathcal{M}_2$ do not consider instances that are outside the schema defined by $\mathrm{dom}(\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2)$. We use these conditions to define the merge operator.

DEFINITION 7.2.2 (Merge operator). *$(\mathcal{M}_1, \mathcal{M}_2)$ is a merge of $\mathcal{M}$ if $\mathcal{M}_1$ and $\mathcal{M}_2$ satisfy conditions (M1), (M2) and (M3).*

Example 7.2.3. Consider schemas $\mathbf{S}_1$, $\mathbf{S}_2$, $\mathbf{S}$ and mapping $\mathcal{M}$ in Example 7.2.1, and let $\Gamma_{\mathbf{S}} = \{\forall x \forall y\, \neg(A'(x, y) \wedge D(x, y))\}$. Moreover, consider mappings $\mathcal{M}_1 = (\mathbf{S}, \mathbf{S}_1, \Sigma_1 \cup \Gamma_{\mathbf{S}})$

and $\mathcal{M}_2 = (\mathbf{S}, \mathbf{S}_2, \Sigma_2 \cup \Gamma_{\mathbf{S}})$ with $\Sigma_1 = \{A'(x,y) \rightarrow A(x,y)\}$ and $\Sigma_2 = \{A'(x,y) \rightarrow B(x,y),\ D(x,y) \rightarrow B(x,y)\}$. Then it can be shown that $(\mathcal{M}_1, \mathcal{M}_2)$ is a merge of $\mathcal{M}$. $\square$

The merge operator has been studied in different contexts (Buneman, Davidson, & Kosky, 1992; Pottinger & Bernstein, 2003; Melnik, 2004; Melnik et al., 2005; Pottinger & Bernstein, 2008; Li, Quix, Kensche, & Geisler, 2010). Our definition is inspired by the definition of Melnik et al. (2004; 2005). However, as for the case of the extract operator, there are some features of their definition that limit its applicability, such as the use of $(\cdot)^{-1}$ that rules out the possibility of merging mappings specified by st-tgds (as shown in Section 7.1.2). This has leaded us to impose some new conditions to define this operator. Moreover, Pottinger and Bernstein (2003, 2008) define a semantics for the merge operator that is based on some *preservation of information* and *minimality* conditions. However, whereas this approach is similar to the one taken in this section, the semantics proposed by Pottinger and Bernstein (2008) is specific to a class of mappings that specify the overlap between schemas by means of conjunctive queries. Finally, the problem of merging different schemas was also considered by Buneman et al. (1992). In that paper, the authors rely on syntactic matchings when merging schemas, instead of considering (semantic) schema mappings to derive relationships between schemas. Thus, in a sense, our merge operator can be seen as a generalization, as schema mappings can specify more complex relationships than syntactic correspondences.

7.2.1. Computing the merge operator

Melnik (2004, Theorem 4.2.4) proposes a straightforward algorithm for the computation of the merge operator, which can also be used in our context to compute this operator for mappings specified by FO-TO-CQ dependencies, thus showing that the merge operator is defined for this class. The algorithm proposed by Melnik (2004, Theorem 4.2.4) for merging schemas $\mathbf{S}_1$ and $\mathbf{S}_2$ given a mapping $\mathcal{M} = (\mathbf{S}_1, \mathbf{S}_2, \Sigma)$, essentially considers a unified schema that simply stores the union of all the instances of $\mathbf{S}_1$ and $\mathbf{S}_2$ that are related by Σ. More precisely, assume that $\mathcal{M} = (\mathbf{S}_1, \mathbf{S}_2, \Sigma)$, where $\mathbf{S}_1$ and $\mathbf{S}_2$ are disjoint ground schemas and Σ is an arbitrary set of dependencies over $\mathbf{S}_1$ and $\mathbf{S}_2$. Moreover, let

$\mathcal{M}_1 = (\widehat{\mathbf{S}}_1 \cup \widehat{\mathbf{S}}_2, \mathbf{S}_1, \Sigma_1 \cup \Sigma)$ and $\mathcal{M}_2 = (\widehat{\mathbf{S}}_1 \cup \widehat{\mathbf{S}}_2, \mathbf{S}_2, \Sigma_2 \cup \Sigma)$, where $\widehat{\mathbf{S}}_1 = \{\widehat{R} \mid R \in \mathbf{S}_1\}$, $\widehat{\mathbf{S}}_2 = \{\widehat{S} \mid S \in \mathbf{S}_2\}$ and Σ_1, Σ_2 are *copying settings* for $\mathbf{S}_1$ and $\mathbf{S}_2$, respectively, that is, $\Sigma_1 = \{\widehat{R}(\bar{x}) \rightarrow R(\bar{x}) \mid R \in \mathbf{S}_1\}$ and $\Sigma_2 = \{\widehat{S}(\bar{x}) \rightarrow S(\bar{x}) \mid S \in \mathbf{S}_2\}$. Then it can be easily shown that $(\mathcal{M}_1, \mathcal{M}_2)$ is a merge of $\mathcal{M}$.

However, motivated by Example 7.2.1, we develop here an algorithm for the case of mappings given by full FO-TO-CQ dependencies, that outputs a merge that makes use of smaller instances in the global schema. In the algorithm we make use of the procedure QUERYREWRITINGATOM described in Lemma 3.3.9 that given a mapping $\mathcal{M}$ specified by FO-TO-CQ dependencies and an atomic conjunctive query Q of the form $A(\bar{x})$ with A a relational symbol in the target schema of $\mathcal{M}$, computes a source rewriting of Q under $\mathcal{M}$.

Algorithm MERGE($\mathcal{M}$)

Input: Mapping $\mathcal{M} = (\mathbf{S}_1, \mathbf{S}_2, \Sigma)$, where $\mathbf{S}_1$, $\mathbf{S}_2$ are disjoint ground schemas and Σ is a set of full FO-TO-CQ dependencies.

Output: A merge $(\mathcal{M}_1, \mathcal{M}_2)$ of $\mathcal{M}$.

(1) Let $\mathbf{S} = \{\widehat{P} \mid P \in \mathbf{S}_1\} \cup \{D_R \mid R \in \mathbf{S}_2\}$ be a ground schema.

(2) Construct a set Σ_1 as follows. For every n-ary relation symbol P in $\mathbf{S}_1$, include dependency $\widehat{P}(\bar{x}) \rightarrow P(\bar{x})$ into Σ_1, with $\bar{x}$ an n-ary tuple of distinct variables.

(3) Construct sets Σ_2 and $\Gamma_{\mathbf{S}}$ as follows. For every n-ary relation symbol R in $\mathbf{S}_2$ do the following. Let $\bar{x}$ be an n-ary tuple of distinct variables.

  (a) Include $D_R(\bar{x}) \rightarrow R(\bar{x})$ into Σ_2.

  (b) Let $\alpha(\bar{x})$ be the output of QUERYREWRITINGATOM($\mathcal{M}, R(\bar{x})$), and $\widehat{\alpha}(\bar{x})$ the formula obtained from $\alpha(\bar{x})$ by replacing every relation symbol $P \in \mathbf{S}_1$ by $\widehat{P}$.

  (c) Include $\widehat{\alpha}(\bar{x}) \rightarrow R(\bar{x})$ into Σ_2 and $\forall \bar{x}\, \neg(\widehat{\alpha}(\bar{x}) \wedge D_R(\bar{x}))$ into $\Gamma_{\mathbf{S}}$.

(4) Let $\mathcal{M}_1 = (\mathbf{S}, \mathbf{S}_1, \Sigma_1 \cup \Gamma_{\mathbf{S}})$ and $\mathcal{M}_2 = (\mathbf{S}, \mathbf{S}_2, \Sigma_2 \cup \Gamma_{\mathbf{S}})$. Return $(\mathcal{M}_1, \mathcal{M}_2)$.

□

THEOREM 7.2.4. MERGE($\mathcal{M}$) *returns a merge of* $\mathcal{M}$.

PROOF. Let $\mathcal{M} = (\mathbf{S}_1, \mathbf{S}_2, \Sigma)$ with $\mathbf{S}_1$ and $\mathbf{S}_2$ ground schemas and Σ a set of full FO-TO-CQ dependencies. Assume that $\mathcal{M}_1 = (\mathbf{S}, \mathbf{S}_1, \Sigma_1 \cup \Gamma_{\mathbf{S}})$ and $\mathcal{M}_2 = (\mathbf{S}, \mathbf{S}_2, \Sigma_2 \cup \Gamma_{\mathbf{S}})$ are the output of MERGE($\mathcal{M}$). Since $\mathbf{S}_1$ and $\mathbf{S}_2$ are disjoint, it is easy to see that $\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2$ is the mapping

$$\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2 = (\mathbf{S}, \mathbf{S}_1 \cup \mathbf{S}_2, \Sigma_1 \cup \Sigma_2 \cup \Sigma \cup \Gamma_{\mathbf{S}})$$

Let I be an arbitrary instance of $\mathbf{S}$ and assume that $I \models \Gamma_{\mathbf{S}}$. We show next that $\mathrm{chase}_{\Sigma_1}(I) \cup \mathrm{chase}_{\Sigma_2}(I)$ is a minimum solution for I under $\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2$. By the properties of the chase and since Σ_1 and Σ_2 are sets of full dependencies and $\mathbf{S}_1$ and $\mathbf{S}_2$ are disjoint, we only need to show that $\mathrm{chase}_{\Sigma_1}(I) \cup \mathrm{chase}_{\Sigma_2}(I)$ is a solution for I under $\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2$. That is what we do next.

Since $(I, \mathrm{chase}_{\Sigma_1}(I)) \models \Sigma_1$, $(I, \mathrm{chase}_{\Sigma_2}(I)) \models \Sigma_2$, and $I \models \Gamma_{\mathbf{S}}$ in order to prove that $\mathrm{chase}_{\Sigma_1}(I) \cup \mathrm{chase}_{\Sigma_2}(I)$ is a solution for I under $\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2$ we just need to show that $(\mathrm{chase}_{\Sigma_1}(I), \mathrm{chase}_{\Sigma_2}(I)) \models \Sigma$. Let $\varphi(\bar{y}) \rightarrow R(\bar{y})$ be a dependency in Σ with $\bar{y}$ a tuple of non necessarily distinct variables, and assume that $\mathrm{chase}_{\Sigma_1}(I) \models \varphi(\bar{a})$. We need to prove that $\mathrm{chase}_{\Sigma_2}(I) \models R(\bar{a})$. Notice that $\mathrm{chase}_{\Sigma_1}(I)$ is an instance such that $P^{\mathrm{chase}_{\Sigma_1}(I)} = \widehat{P}^I$ for every $P \in \mathbf{S}_1$. Thus, if $\mathrm{chase}_{\Sigma_1}(I) \models \varphi(\bar{a})$ then $I \models \widehat{\varphi}(\bar{a})$ where $\widehat{\varphi}(\bar{a})$ is obtained from $\varphi(\bar{a})$ replacing every $P \in \mathbf{S}_1$ by $\widehat{P}$. Now, $\mathrm{chase}_{\Sigma_2}(I)$ is an instance such that $R^{\mathrm{chase}_{\Sigma_2}(I)} = D_R^I \cup \{\bar{a} \mid I \models \widehat{\alpha}(\bar{a})\}$ where $\alpha(\bar{x})$ is a source rewriting of $R(\bar{x})$ under $\mathcal{M}$ and $\widehat{\alpha}(\bar{x})$ is obtained from $\alpha(x)$ replacing every $P \in \mathbf{S}_1$ by $\widehat{P}$. Moreover, notice that for every $J \in \mathrm{Sol}_{\mathcal{M}}(\mathrm{chase}_{\Sigma_1}(I))$ we have that $J \models R(\bar{a})$, thus $\bar{a} \in \mathrm{certain}_{\mathcal{M}}(R(\bar{x}), \mathrm{chase}_{\Sigma_1}(I))$. This implies that $\mathrm{chase}_{\Sigma_1}(I) \models \alpha(\bar{a})$, and thus $I \models \widehat{\alpha}(\bar{a})$ which implies that $\bar{a} \in R^{\mathrm{chase}_{\Sigma_2}(I)}$ and thus $\mathrm{chase}_{\Sigma_2}(I) \models R(\bar{a})$. This was to be shown.

Thus we have that if $I \models \Gamma_{\mathbf{S}}$ then we have that $\mathrm{chase}_{\Sigma_1}(I) \cup \mathrm{chase}_{\Sigma_2}(I)$ is a minimum solution for I under $\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2$. In particular, this shows that $I \in \mathrm{dom}(\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2)$ if and only if $I \models \Gamma_{\mathbf{S}}$. Moreover, since $\mathcal{M}_1$ and $\mathcal{M}_2$ are st-tgds, we have that $I \in \mathrm{dom}(\mathcal{M}_1)$ if and only if $I \in \mathrm{dom}(\mathcal{M}_2)$ if and only if $I \models \Gamma_{\mathbf{S}}$. Thus we have that $\mathrm{dom}(\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2) = \mathrm{dom}(\mathcal{M}_1) = \mathrm{dom}(\mathcal{M}_2)$. And thus (M3) holds.

We show now that for every $L = J \cup K$ such that $(J, K) \in \mathcal{M}$ there exists an instance I such that $J = \text{chase}_{\Sigma_1}(I)$, $K = \text{chase}_{\Sigma_2}(I)$ and $I \models \Gamma_{\mathbf{S}}$. In particular this implies two facts: (1) for every $(J, K) \in \mathcal{M}$ there exists $I \in \mathbf{S}$ such that $(I, J \cup K) \in \mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2$, and (2) every instance $L \in \text{range}(\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2)$ is a minimum solution of some instance $I \in \text{dom}(\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2)$. Now, given the instance $J \cup K$ such that $(J, K) \in \mathcal{M}$ consider the following instance I of $\mathbf{S}$. For every $P \in \mathbf{S}_1$ we have $\widehat{P}^I = P^J$, and for every $R \in \mathbf{S}_2$ we have that $D_R^I = \{\bar{a} \mid \bar{a} \in R^K \text{ and } J \not\models \alpha(\bar{a})\}$ where $\alpha(\bar{x})$ is a source rewriting of $R(\bar{x})$ under $\mathcal{M}$. First notice that every formula in $\Gamma_{\mathbf{S}}$ is of the form $\widehat{\alpha}(\bar{x}) \rightarrow \neg D_R(\bar{x})$ where $\alpha(\bar{x})$ is a source rewriting of $R(\bar{x})$ under $\mathcal{M}$ and $\widehat{\alpha}(\bar{x})$ is obtained from $\alpha(x)$ replacing every $P \in \mathbf{S}_1$ by $\widehat{P}$. Thus, by the construction of I we directly obtain that $I \models \Gamma_{\mathbf{S}}$ and thus I is in $\text{dom}(\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2)$. Now, it is straightforward to see that $J = \text{chase}_{\Sigma_1}(I)$, thus we just need to prove that $K = \text{chase}_{\Sigma_2}(I)$. Let $R \in \mathbf{S}_2$. Notice that $R^{\text{chase}_{\Sigma_2}(I)} = D_R^I \cup \{\bar{a} \mid I \models \widehat{\alpha}(\bar{a})\}$ where $\alpha(\bar{x})$ is a source rewriting of $R(\bar{x})$ under $\mathcal{M}$ and $\widehat{\alpha}(\bar{x})$ is obtained from $\alpha(x)$ replacing every $P \in \mathbf{S}_1$ by $\widehat{P}$. Thus, since $J = \text{chase}_{\Sigma_1}(I)$ we know that $I \models \widehat{\alpha}(\bar{a})$ if and only if $J \models \alpha(\bar{a})$. We have that $R^{\text{chase}_{\Sigma_2}(I)} = D_R^I \cup \{\bar{a} \mid J \models \alpha(\bar{a})\}$, and then from the definition of D_R^I, we obtain that $R^{\text{chase}_{\Sigma_2}(I)} = \{\bar{a} \mid \bar{a} \in R^K \text{ and } J \not\models \alpha(\bar{a})\} \cup \{\bar{a} \mid J \models \alpha(\bar{a})\}$. Now notice that $(J, K) \in \mathcal{M}$ thus if $J \models \alpha(\bar{a})$ then $K \models R(\bar{a})$ which implies that $\{\bar{a} \mid J \models \alpha(\bar{a})\} \subseteq R^K$. From this last property we obtain that $R^{\text{chase}_{\Sigma_2}(I)} = \{\bar{a} \mid \bar{a} \in R^K \text{ and } J \not\models \alpha(\bar{a})\} \cup \{\bar{a} \mid J \models \alpha(\bar{a})\}$ equals R^K. We have shown that for every $R \in \mathbf{S}_2$ it holds that $R^{\text{chase}_{\Sigma_2}(I)} = R^K$ which implies that $\text{chase}_{\Sigma_2}(I) = K$. This completes this part of the proof.

We have shown that if $(J, K) \in \mathcal{M}$ then there exists an instance $I \in \text{dom}(\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2)$ such that $(I, J \cup K) \in \mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2$. In particular, this shows that $\text{range}(\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2) = \{J \cup K \mid (J, K) \in \mathcal{M}\}$. Moreover, we have shown that every instance $I \in \text{dom}(\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2)$ has a minimum solution, and that every instance $L \in \text{range}(\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2)$ is minimum solution of some instance. Thus, by Lemma 6.4.11 we conclude that $\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2$ is target non redundant. This proves condition (M1).

Now we show that $\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2$ is source non-redundant. Notice that $\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2$ has universal solutions and is closed under homomorphisms in $\text{range}(\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2)$. In

fact, since the dependencies defining $\mathcal{M}_1$ and $\mathcal{M}_2$ are full, we have that for every $I \in \mathrm{dom}(\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2)$ the instance $\mathrm{chase}_{\Sigma_1}(I) \cup \mathrm{chase}_{\Sigma_2}(I)$ is a minimum solution, and for every instance K such that $K \in \mathrm{range}(\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2)$ and $\mathrm{chase}_{\Sigma_1}(I) \cup \mathrm{chase}_{\Sigma_2}(I) \subseteq K$ we have that K is a solution for I. Thus, we can make use of Theorem 6.4.14 to show that $\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2$ is source non-redundant. Thus let I and L be two instances with the same space of solutions under $\mathcal{M}_1 \oplus_{\mathcal{M}} \mathcal{M}_2$. We need to show that $I = L$. Notice that $\mathrm{chase}_{\Sigma_1}(I) \cup \mathrm{chase}_{\Sigma_2}(I)$ is a minimum solution for I, and similarly for L. Thus, if I and L have the same space of solutions, we have that $\mathrm{chase}_{\Sigma_1}(I) = \mathrm{chase}_{\Sigma_1}(L)$ and that $\mathrm{chase}_{\Sigma_2}(I) = \mathrm{chase}_{\Sigma_2}(L)$. From $\mathrm{chase}_{\Sigma_1}(I) = \mathrm{chase}_{\Sigma_1}(L)$ we obtain that for every $P \in \mathbf{S}_1$ it holds that $P^I = P^L$ and thus, $\widehat{P}^I = \widehat{P}^L$. Now, by the construction of Σ_2, since $\widehat{P}^I = \widehat{P}^L$ for every $P \in \mathbf{S}_1$, and since $\mathrm{chase}_{\Sigma_2}(I) = \mathrm{chase}_{\Sigma_2}(L)$, it is easy to prove that $D^I_R = D^L_R$ for every $R \in \mathbf{S}_2$. Thus, we have that $\widehat{P}^I = \widehat{P}^L$ for every $P \in \mathbf{S}_1$ and that $D^I_R = D^L_R$ for every $R \in \mathbf{S}_2$ from which we conclude that $I = L$. This prove that $\mathcal{M}_1$ and $\mathcal{M}_2$ also satisfy (M2), completing the proof of correctness of MERGE. $\square$

8. CONCLUSIONS AND FUTURE WORK

The importance of schema mappings is widely recognized nowadays (Hernández et al., 2002; Bernstein, 2003; Melnik, 2004; Fagin, Kolaitis, Popa, & Tan, 2005; Melnik et al., 2005; Haas, Hernández, Ho, Popa, & Roth, 2005; Fagin, 2007; Fuxman et al., 2006; Hernández et al., 2007; Arenas, Pérez, Reutter, & Riveros, 2009a; Arenas, Pérez, et al., 2010; Fagin et al., 2011). In particular, the data-management community recognizes the need to develop techniques to manipulate these schema mapping specifications, in order to provide an integral solution to fundamental data-interoperability tasks such as data exchange, data integration, and peer data management. This dissertation presented several contributions to the formal study of schema mapping operators, with special attention on their use in data exchange systems. We proposed formal semantics for the inverse, extract and merge operators, we presented algorithms to compute them, and study expressiveness issues. More importantly, we studied the issue of finding mapping specification languages that are closed under some operations. We also introduced the novel notions of information and redundancy in schema mappings that we proved are essential to build a general framework to study some fundamental properties of existing mapping operators as well as to formalize new operators.

Several questions remain unanswered, and there are several open problems that can be used to build an interesting program for future research. In particular, although almost all the definitions for schema mapping operators presented in this dissertation apply to general mappings (where mappings are simply sets of pair of instances), the results on mapping languages, algorithms, and also some tools and characterizations, are proved for the relational case. Thus, as a first direction for future research it would be certainly relevant to study the problem of operating schema mappings in a data model beyond the relational model. One candidate to begin with is the XML data model. Although XML-schema mapping languages have been proposed and studied (Arenas & Libkin, 2008; Amano, Libkin, & Murlak, 2009; Terwilliger, Bernstein, & Melnik, 2009; Arenas, Barceló, Libkin, & Murlak,

2010), little attention has been paid to the formal study of XML-schema mapping operators. For the case of the composition operator, a first insight has been given by Amano et al. (2009), showing that the previous results obtained for the relational model are not directly applicable over XML. Inversion of XML-schema mappings, as well as the application of the other operators considered in this dissertation in the XML context, remains an unexplored field.

In the relational model, there are still interesting problems to explore. Notice that almost all the technical results (algorithms, closure properties, characterizations) presented in this dissertation, consider mapping languages given by $\mathcal{L}_1$-TO-$\mathcal{L}_2$ dependencies, that is, implication formulas of the form $\varphi(\bar{x}) \rightarrow \psi(\bar{x})$ with $\varphi(\bar{x})$ a query in $\mathcal{L}_1$ and and $\psi(\bar{x})$ a query in $\mathcal{L}_2$. Thus, it would be interesting to study the application of the concepts presented in this dissertation to mappings specified in other formalisms. A natural possibility is to consider mappings specified by *bidirectional-implication* formulas of the form $\varphi(\bar{x}) \leftrightarrow \psi(\bar{x})$, that in general provide more information about the relationship between source and target instances than mappings specified by implication formulas. One particular case where bidirectional-implication formulas would be very useful, is the case of the merge operator where the mapping is not actually used to exchange data, but to establish the correspondence of existing pieces of data. Little research has been carried out to study operators for mappings specified in this kind of formalisms. Nash et al. (2005) considered the composition operator for mappings specified by languages beyond implication formulas. Melnik, Adya, and Bernstein (2008) also considered a special form of bidirectional-implication formulas to define mappings when studying the notion of *data-roundtripping*, which is closely related to the notion of inverting mappings. In any case, most of the questions considered in this dissertation remain open for the composition, inversion, and the other operators when schema mappings are specified by bidirectional-implication formulas.

Besides studying operators like inversion and composition for database models beyond the relational model, or for mapping languages beyond $\mathcal{L}_1$-TO-$\mathcal{L}_2$ dependencies, we believe that future efforts have to be focused in providing a unifying framework for these operators. A natural question, for instance, is whether there exists a schema mapping language that is

closed under both composition and inverse. In this respect we have made some progress by proposing the language of plain SO-tgds, which is closed under CQ-composition and admits CQ-maximum recoveries. It follows easily from the results of this dissertation that the language of plain SO-tgds is not closed under the notion of CQ-maximum recovery. Thus, an interesting open question is whether some extension of the language of plain SO-tgds enjoy such a closure property.

Another problem to explore is to define a notion of *completeness* for schema mapping operators. Given an abstract complex task on schema mappings, like schema evolution or the problem of building a global schema in a data integration system, under which conditions a set of operators is considered to be *complete* for that task? We would also need to answer questions like, can the result of some operator over schema mappings be obtained as a combination of other operators? As pointed out by Melnik et al. (2005), these are fundamental open problems on this area. Answering these questions would allow us to characterize good sets of schema mapping operators that are complete and minimal to achieve certain complex tasks.

As many information-system problems involve not only the design and integration of complex application artifacts, but also their subsequent manipulation, the definition, formalization and implementation of operators over schema mappings will continue to play a fundamental role in the research on data-interoperability. We believe that the results presented in this dissertation together with the solutions to the proposed open problems, will play a significant role to advance our understanding of this fundamental area of data management.

REFERENCES

Abiteboul, S., & Duschka, O. (1998). Complexity of answering queries using materialized views. In *PODS* (p. 254-263).

Amano, S., Libkin, L., & Murlak, F. (2009). XML schema mappings. In *Pods* (p. 33-42).

Arenas, M. (2006). Personal communication.

Arenas, M., Barceló, P., Fagin, R., & Libkin, L. (2004). Locally consistent transformations and query answering in data exchange. In *PODS* (p. 229-240).

Arenas, M., Barceló, P., Libkin, L., & Murlak, F. (2010). *Relational and XML data exchange*. Morgan & Claypool Publishers.

Arenas, M., Barceló, P., & Reutter, J. L. (2009). Query languages for data exchange: beyond unions of conjunctive queries. In *ICDT* (p. 73-83).

Arenas, M., Fagin, R., & Nash, A. (2010). Composition with target constraints. In *ICDT* (p. 129-142).

Arenas, M., & Libkin, L. (2008). XML data exchange: Consistency and query answering. *J. ACM*, *55*(2).

Arenas, M., Pérez, J., Reutter, J. L., & Riveros, C. (2009a). Composition and inversion of schema mappings. *SIGMOD Record*, *38*(3), 17–28.

Arenas, M., Pérez, J., Reutter, J. L., & Riveros, C. (2009b). Inverting schema mappings: Bridging the gap between theory and practice. *PVLDB*, *2*(1), 1018–1029.

Arenas, M., Pérez, J., Reutter, J. L., & Riveros, C. (2010). Foundations of schema mapping management. In *PODS* (pp. 227–238).

Arenas, M., Pérez, J., & Riveros, C. (2008). The recovery of a schema mapping: bringing exchanged data back. In *PODS* (p. 13-22).

Arenas, M., Pérez, J., & Riveros, C. (2009). The recovery of a schema mapping: bringing exchanged data back. *TODS*, *34*(4).

Arocena, P. C., Fuxman, A., & Miller, R. J. (2010). Composing local-as-view mappings: closure and applications. In *ICDT* (p. 209-218).

Bernstein, P. (2003). Applying model management to classical meta data problems. In *CIDR*.

Bernstein, P., Halevy, A. Y., & Pottinger, R. (2000). A vision of management of complex models. *SIGMOD Record*, *29*(4), 55-63.

Bernstein, P., & Melnik, S. (2007). Model management 2.0: manipulating richer mappings. In *SIGMOD* (p. 1-12).

Buneman, P., Davidson, S. B., & Kosky, A. (1992). Theoretical aspects of schema merging. In *EDBT* (p. 152-167).

ten Cate, B., & Kolaitis, P. G. (2009). Structural characterizations of schema-mapping languages. In *ICDT* (p. 63-72).

ten Cate, B., & Kolaitis, P. G. (2010). Structural characterizations of schema-mapping languages. *Commun. ACM*, *53*(1), 101-110.

Dawar, A. (1998). A restricted second order logic for finite structures. *Inf. Comput.*, *143*(2), 154-174.

Du, D.-Z., & Ko, K.-I. (2000). *Theory of computational complexity*. Wiley-Interscience.

Duschka, O. M., & Genesereth, M. R. (1997). Answering recursive queries using views. In *PODS* (p. 109-116).

Fagin, R. (1982). Horn clauses and database dependencies. *JACM*, *29*(4), 952-985.

Fagin, R. (2007). Inverting schema mappings. *TODS*, *32*(4).

Fagin, R., Kolaitis, P. G., Miller, R. J., & Popa, L. (2005). Data exchange: semantics and query answering. *TCS*, *336*(1), 89-124.

Fagin, R., Kolaitis, P. G., Nash, A., & Popa, L. (2008). Towards a theory of schema-mapping optimization. In *PODS* (p. 33-42).

Fagin, R., Kolaitis, P. G., & Popa, L. (2005). Data exchange: getting to the core. *TODS*, *30*(1), 174-210.

Fagin, R., Kolaitis, P. G., Popa, L., & Tan, W.-C. (2005). Composing schema mappings: Second-order dependencies to the rescue. *TODS*, *30*(4), 994-1055.

Fagin, R., Kolaitis, P. G., Popa, L., & Tan, W. C. (2008). Quasi-inverses of schema mappings. *TODS*, *33*(2).

Fagin, R., Kolaitis, P. G., Popa, L., & Tan, W.-C. (2009). Reverse data exchange: coping with nulls. In *PODS* (pp. 23–32).

Fagin, R., Kolaitis, P. G., Popa, L., & Tan, W.-C. (2011). Schema mapping evolution through composition and inversion. In Z. Bellahsene, A. Bonifati, & E. Rahm (Eds.), *Schema matching and mapping* (p. 191-222). Springer.

Fuxman, A., Hernández, M. A., Ho, C. T. H., Miller, R. J., Papotti, P., & Popa, L. (2006). Nested mappings: Schema mapping reloaded. In *VLDB* (p. 67-78).

Haas, L. M., Hernández, M. A., Ho, H., Popa, L., & Roth, M. (2005). Clio grows up: from research prototype to industrial tool. In *Sigmod conference* (p. 805-810).

Halevy, A. (2000). Theory of answering queries using views. *SIGMOD Record*, *29*(1), 40-47.

Halevy, A. (2001). Answering queries using views: A survey. *VLDB J.*, *10*(4), 270-294.

Hell, P., & Nešetřil, J. (2004). *Graphs and Homomorphisms*. Oxford University Press.

Hernández, M. A., Ho, H., Popa, L., Fuxman, A., Miller, R. J., Fukuda, T., et al. (2007). Creating nested mappings with clio. In *Icde* (p. 1487-1488).

Hernández, M. A., Popa, L., Velegrakis, Y., Miller, R. J., Naumann, F., & Ho, C.-T. (2002). Mapping XML and relational schemas with Clio. In *Icde* (p. 498-499).

Imielinski, T., & Lipski, W. (1984). Incomplete information in relational databases. *Journal of the ACM*, *31*(4), 761-791.

Kolaitis, P. G. (2005). Schema mappings, data exchange, and metadata management. In *PODS* (p. 61-75).

Lenzerini, M. (2002). Data integration: a theoretical perspective. In *PODS* (p. 233-246).

Levy, A., Mendelzon, A., Sagiv, Y., & Srivastava, D. (1995). Answering queries using views. In *PODS* (p. 95-104).

Levy, A., Rajaraman, A., & Ordille, J. J. (1996). Querying heterogeneous information sources using source descriptions. In *VLDB* (p. 251-262).

Li, X., Quix, C., Kensche, D., & Geisler, S. (2010). Automatic schema merging using mapping constraints among incomplete sources. In *CIKM* (p. 299-308).

Libkin, L. (2004). *Elements of Finite Model Theory* (1st ed.). Springer-Verlag.

Madhavan, J., & Halevy, A. Y. (2003). Composing mappings among data sources. In *VLDB* (p. 572-583).

Maier, D., Mendelzon, A., & Sagiv, Y. (1979). Testing implications of data dependencies. *TODS*, *4*(4), 455-469.

Melnik, S. (2004). *Generic model management: concepts and algorithms* (Vol. 2967). Springer.

Melnik, S., Adya, A., & Bernstein, P. A. (2008). Compiling mappings to bridge applications and databases. *ACM Trans. Database Syst.*, *33*(4).

Melnik, S., Bernstein, P. A., Halevy, A. Y., & Rahm, E. (2005). Supporting executable mappings in model management. In *SIGMOD* (p. 167-178).

Nash, A., Bernstein, P. A., & Melnik, S. (2005). Composition of mappings given by embedded dependencies. In *PODS* (p. 172-183).

Papadimitriou, C. M. (1994). *Computational complexity*. Reading, Massachusetts: Addison-Wesley.

Pottinger, R., & Bernstein, P. (2003). Merging models based on given correspondences. In *VLDB* (p. 826-873).

Pottinger, R., & Bernstein, P. (2008). Schema merging and mapping creation for relational sources. In *EDBT* (p. 73-84).

Pottinger, R., & Halevy, A. Y. (2001). Minicon: A scalable algorithm for answering queries using views. *VLDB J.*, *10*(2-3), 182-198.

Riveros, C. (2008). *Recovering Information in Data Exchange*. Master's thesis, Department of Computer Science, Pontificia Universidad Católica de Chile, Santiago, Chile.

Segoufin, L., & Vianu, V. (2005). Views and queries: determinacy and rewriting. In *PODS* (p. 49-60).

Terwilliger, J. F., Bernstein, P. A., & Melnik, S. (2009). Full-fidelity flexible object-oriented XML access. *PVLDB*, *2*(1), 1030-1041.

APPENDIX A. QUERY REWRITING TOOLS

A.1. Source Rewriting in Schema Mappings

In this section we show how source rewritings of conjunctive queries can be computed. The main goal of this section is to provide formal proofs for Lemmas 3.3.1 and 3.3.9.

It should be noticed that the problem of computing rewritings of queries has been extensively studied in the database area (Levy et al., 1995; Abiteboul & Duschka, 1998) and, in particular, in the data integration context (Halevy, 2000, 2001; Lenzerini, 2002). In particular, the class of CQ-TO-CQ dependencies corresponds to the class of GLAV mappings in the data integration context (Lenzerini, 2002), and, as such, the techniques developed to solved the query rewriting problem for GLAV mappings can be reused in our context. It is important to notice that most of the query rewriting techniques have been developed for two sub-classes of GLAV mappings, namely GAV mappings, which essentially corresponds to the class of mappings specified by full CQ-TO-CQ dependencies (Lenzerini, 2002), and LAV mappings, which are mappings specified by CQ-TO-CQ dependencies of the form $R(x_1, \ldots, x_k) \rightarrow \psi(x_1, \ldots, x_k)$, where R is a source predicate (Lenzerini, 2002). However, it is possible to reuse a large part of the work in this area as a GLAV mapping can be represented as the composition of a GAV and a LAV mapping.

Example A.1.1. Assume that $\mathcal{M}$ is specified by dependency:

$$R(x) \wedge S(x) \quad \rightarrow \quad \exists y\, T(x, y).$$

Then $\mathcal{M}$ is equivalent to the composition of a GAV mapping specified by dependency $R(x) \wedge S(x) \rightarrow U(x)$ and a LAV mapping specified by dependency $U(x) \rightarrow \exists y\, T(x, y)$, where U is an auxiliary relation. □

More formally, let $\mathcal{M}$ be a mapping specified by a set of CQ-TO-CQ dependencies and Q a conjunctive query over the target of $\mathcal{M}$. Then one can obtain a rewriting of Q over the source as follows. First, one constructs, as in the above example, a GAV mapping $\mathcal{M}_1$ and a LAV mapping $\mathcal{M}_2$ such that $\mathcal{M} = \mathcal{M}_1 \circ \mathcal{M}_2$. Second, one obtains a rewriting Q' of

Q over the source of $\mathcal{M}_2$ by adopting one of the algorithms proposed in the literature for query rewriting for LAV mappings (Levy, Rajaraman, & Ordille, 1996; Duschka & Genesereth, 1997; Pottinger & Halevy, 2001). Finally, one obtains a rewriting Q'' of Q' over the source of $\mathcal{M}_1$, which is the desired rewriting of Q, by simply unfolding Q' according to the dependencies of mapping $\mathcal{M}_1$ (Lenzerini, 2002).

It should be noticed that the time complexity of the rewriting procedure described above is exponential in the size of the mapping and the query, and that this procedure can also be used for the case of mappings specified by FO-TO-CQ dependencies. If $\mathcal{M}$ is specified by a set of FO-TO-CQ dependencies, then by using the same idea as in Example A.1.1, it is possible to show that $\mathcal{M}$ is equivalent to the composition of a mapping $\mathcal{M}_1$ specified by a set of full FO-TO-CQ dependencies and a LAV mapping $\mathcal{M}_2$. Thus, given that the query unfolding process can be carried out over a set of full FO-TO-CQ dependencies in the same way as for GAV mappings, the process described above can be used to compute in exponential time the rewriting of a target conjunctive query over the source of $\mathcal{M}$.

For the sake of completeness, in this paper we propose a novel exponential-time algorithm that given a mapping $\mathcal{M}$ specified by a set of FO-TO-CQ st-dependencies and a conjunctive query Q over the target schema, produces a rewriting of Q over the source of $\mathcal{M}$. This algorithm does not follow the approach described above, as it directly uses the dependencies specifying $\mathcal{M}$ to construct a query rewriting (it does not decompose $\mathcal{M}$ into the composition of two mappings). In particular, the time complexity of the algorithm is exponential, so it could be used as an alternative query rewriting algorithm. Another reason to include a complete algorithm is that, in several results in this dissertation we need to reason about the language used as the output of the query rewriting algorithm depending on the language used to specify the input mappings. For example, when the input mapping $\mathcal{M}$ is specified by FO-TO-CQ dependencies, our algorithm generates an FO query. Moreover, the output of our algorithm in this case is a query constructed as a combinations (by using disjunctions, conjunctions, existential quantification, and equalities) of the premises of the dependencies that defines $\mathcal{M}$. On the other hand, if the input mapping is specified

by st-tgds, then our algorithm generates a query in $\text{UCQ}^{=}$ (see Lemma 3.3.3), and if the input is specified by $\text{CQ}^{\neq}$-TO-CQ dependencies, then our algorithm produces a query in $\text{UCQ}^{=,\neq}$ (see Lemma 6.2.5).

A.1.1. Proof of Lemma 3.3.1

To prove the lemma, we provide an algorithm that given an st-mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ such that Σ is a set of FO-TO-CQ dependencies, and a conjunctive query Q over schema $\mathbf{T}$, computes a query Q' that is a rewriting of Q over the source schema $\mathbf{S}$.

We first introduce the terminology used in the algorithm. The basic notion used in the algorithm is that of *existential replacement*. In an existential replacement of a formula β, we are allowed to existentially quantify some of the positions of the free variables of β. For example, if $\beta(x_1, x_2, x_3) = P(x_1, x_2) \wedge R(x_2, x_3)$, then two existential replacements of $\beta(x_1, x_2, x_3)$ are $\gamma_1(x_2) = \exists u \exists v\, (P(u, x_2) \wedge R(x_2, v))$ and $\gamma_2(x_1, x_2, x_3) = \exists z\, (P(x_1, z) \wedge R(x_2, x_3))$. We note that both γ_1 and γ_2 are implied by β. In an existential replacement, we are also allowed to use the same quantifier for different positions. For example, $\gamma_3(x_2) = \exists w\, (P(w, x_2) \wedge R(x_2, w))$ is also an existential replacement of β. We note that γ_3 is implied by β if x_1 and x_3 have the same value, that is, $\beta(x_1, x_2, x_3) \wedge x_1 = x_3$ implies γ_3. In an existential replacement, these equalities are also included. Formally, given a formula $\beta(\bar{x})$ where $\bar{x} = (x_1, \ldots, x_k)$ is a tuple of distinct variables, an *existential replacement* of $\beta(\bar{x})$ is a pair of formulas $(\exists \bar{z}\, \gamma(\bar{x}', \bar{z}), \theta(\bar{x}''))$, where: (1) $\exists \bar{z}\, \gamma(\bar{x}', \bar{z})$ is obtained from $\beta(\bar{x})$ by existentially quantifying some of the positions of the free variables of $\beta(\bar{x})$, and $\bar{z}$ is the tuple of fresh variables used in these quantifications, (2) $\theta(\bar{x}'')$ is a conjunction of equalities such that $x_i = x_j$ is in θ ($1 \leq i, j \leq k$ and $i \neq j$) if we replace a position with variable x_i and a position with variable x_j by the same variable z from $\bar{z}$, and (3) $\bar{x}'$ and $\bar{x}''$ are the tuples of free variables of $\exists \bar{z}\, \gamma(\bar{x}', \bar{z})$ and $\theta(\bar{x}'')$, respectively. Notice that $\exists \bar{z}\, \gamma(\bar{x}', \bar{z})$ is a logical consequence of $\beta(\bar{x}) \wedge \theta(\bar{x}'')$. For example, the following are existential replacements of the formula $\beta(x_1, x_2, x_3) = \exists y_1\, (R(x_1, x_2, y_1) \wedge T(y_1, x_3, x_2))$:

$$\big(\exists y_1\, (R(x_1, x_2, y_1) \wedge T(y_1, x_3, x_2)),\ \underline{\text{true}}\big),$$
$$\big(\exists z_1 \exists z_2 \exists y_1\, (R(z_1, x_2, y_1) \wedge T(y_1, x_3, z_2)),\ \underline{\text{true}}\big),$$

$$\big(\exists z_1 \exists z_2 \exists y_1 \, (R(z_1, z_1, y_1) \wedge T(y_1, z_2, z_2)),\ x_1 = x_2 \wedge x_3 = x_2\big).$$

In the first existential replacement above, we have replaced no position, thus obtaining the initial formula $\beta(x_1, x_2, x_3)$ and sentence true (this is a valid existential replacement). In the second existential replacement, although we have replaced some positions of free variables by existentially quantified variables z_1 and z_2, we include sentence true since no positions with distinct variables are replaced by the same variable from (z_1, z_2).

In the algorithm, we use the following terminology for tuples of variables: $\bar{x} \subseteq \bar{y}$ indicates that every variable in $\bar{x}$ is also mentioned in $\bar{y}$, $(\bar{x}, \bar{y})$ is a tuple of variables obtained by placing the variables of $\bar{x}$ followed by the variables of $\bar{y}$, $f : \bar{x} \to \bar{y}$ is a substitution that replaces every variable of $\bar{x}$ by a variable of $\bar{y}$ (f is not necessarily a one-to-one function), $f(\bar{x})$ is a tuple of variables obtained by replacing every variable x in $\bar{x}$ by $f(x)$, and if $\bar{x} = (x_1, \ldots, x_k)$ and $\bar{y} = (y_1, \ldots, y_k)$, we use formula $\bar{x} = \bar{y}$ as a shorthand for $x_1 = y_1 \wedge \cdots \wedge x_k = y_k$.

Algorithm QUERYREWRITING($\mathcal{M}, Q$)

Input: An st-mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ where Σ is a set of FO-TO-CQ dependencies, and a conjunctive query Q over $\mathbf{T}$.

Output: An FO query Q' that is a rewriting of Q over the source schema $\mathbf{S}$.

(1) Assume that Q is given by the formula $\exists \bar{y} \psi(\bar{x}, \bar{y})$.

(2) Create a set $\mathcal{C}_\psi$ of FO queries as follows. Start with $\mathcal{C}_\psi = \emptyset$ and let m be the number of atoms in $\psi(\bar{x}, \bar{y})$. Then for every $p \in \{1, \ldots, m\}$ and tuple $((\sigma_1, k_1), \ldots, (\sigma_p, k_p)) \in (\Sigma \times \{1, \ldots, m\})^p$ such that $k_1 + \cdots + k_p = m$, do the following.

   (a) Let $(\xi_1, \ldots, \xi_p)$ be a tuple obtained from $(\sigma_1, \ldots, \sigma_p)$ by renaming the variables of the formulas σ_1, ..., σ_p in such a way that the sets of variables of the formulas ξ_1, ..., ξ_p are pairwise disjoint.

   (b) Assume that ξ_i is equal to $\varphi_i(\bar{u}_i) \to \exists \bar{v}_i \, \psi_i(\bar{u}_i, \bar{v}_i)$, where $\bar{u}_i$ and $\bar{v}_i$ are tuples of distinct variables.

(c) For every tuple $(\chi_1(\bar{w}_1, \bar{z}_1), \ldots, \chi_p(\bar{w}_p, \bar{z}_p))$, where $\chi_i(\bar{w}_i, \bar{z}_i)$ is a conjunction of k_i (not necessarily distinct) atoms from $\psi_i(\bar{u}_i, \bar{v}_i)$, $\bar{w}_i \subseteq \bar{u}_i$, $\bar{z}_i \subseteq \bar{v}_i$, and such that $\bar{w}_i$ and $\bar{z}_i$ are tuples of distinct variables, do the following.

(i) Let $\exists \bar{z}\, \chi(\bar{w}, \bar{z})$ be the formula $\exists \bar{z}_1 \cdots \exists \bar{z}_p(\chi_1(\bar{w}_1, \bar{z}_1) \wedge \cdots \wedge \chi_p(\bar{w}_p, \bar{z}_p))$ with $\bar{w} = (\bar{w}_1, \ldots, \bar{w}_p)$ and $\bar{z} = (\bar{z}_1, \ldots, \bar{z}_p)$.

(ii) Then for every existential replacement $(\exists \bar{s} \exists \bar{z}\, \gamma(\bar{w}', \bar{z}, \bar{s}), \theta(\bar{w}''))$ of $\exists \bar{z}\, \chi(\bar{w}, \bar{z})$ (up to renaming of variables in $\bar{s}$), and for every pair of variable substitutions $f : \bar{x} \to \bar{x}$ and $g : \bar{w}' \to \bar{x}$, check whether there exists a variable substitution $h : \bar{y} \to (\bar{z}, \bar{s})$ such that $\psi(f(\bar{x}), h(\bar{y}))$ and $\gamma(g(\bar{w}'), \bar{z}, \bar{s})$ are syntactically equal (up to reordering of atoms). If this is the case, then add to $\mathcal{C}_\psi$ the following formula:

$$\exists \bar{u}_1 \cdots \exists \bar{u}_p \left(\bigwedge_{i=1}^{p} \varphi_i(\bar{u}_i) \wedge \theta(\bar{w}'') \wedge \bar{x} = f(\bar{x}) \wedge \bar{w}' = g(\bar{w}') \right). \tag{A.2}$$

(3) If $\mathcal{C}_\psi$ is nonempty, then let $\alpha(\bar{x})$ be the FO formula constructed as the disjunction of all the formulas in $\mathcal{C}_\psi$. Otherwise, let $\alpha(\bar{x})$ be false, that is, an arbitrary unsatisfiable formula (with $\bar{x}$ as its tuple of free variables).

(4) Return the query Q' given by $\alpha(\bar{x})$.

□

Notice that in the algorithm, tuple $\bar{x}$ is the set of free variables of formula (A.2) since both $\bar{w}'$ and $\bar{w}''$ are subsets of $(\bar{u}_1, \ldots, \bar{u}_p)$. Also notice that since $\psi(f(\bar{x}), h(\bar{y}))$ and $\gamma(g(\bar{w}'), \bar{z}, \bar{s})$ are identical (up to reordering of atoms), f is a function from $\bar{x}$ to $\bar{x}$, g is a function from $\bar{w}'$ to $\bar{x}$, and h is a function from $\bar{y}$ to $(\bar{z}, \bar{s})$, we have that every variable x in $\bar{x}$ is equal to some variable u in $(\bar{u}_1, \ldots, \bar{u}_p)$ since $\bar{x} = f(\bar{x}) \wedge \bar{w}' = g(\bar{w}')$ is a subformula of (A.2). This implies that formula (A.2) is domain independent since each formula $\varphi_i(\bar{u}_i)$ is assumed to be domain independent. Thus, we also have that $\alpha(\bar{x})$ and Q' are domain independent.

Example A.1.2. Assume that Σ is given by dependency σ:

$$\varphi(x_1, x_2) \quad \rightarrow \quad R(x_1, x_1, x_2), \tag{A.3}$$

where $\varphi(x_1, x_2)$ is an FO formula over the source schema, and that $Q(x_1, x_2, x_3)$ is the conjunctive query $\exists y_1\, \psi(x_1, x_2, x_3, y_1)$, where $\psi(x_1, x_2, x_3, y_1) = R(x_1, x_2, y_1) \wedge R(y_1, x_3, x_3)$. Given that $\psi(x_1, x_2, x_3, y_1)$ has two atoms, the algorithm considers the tuples $(\sigma_1, 2)$ from $(\Sigma \times \{1, 2\})^1$ and $((\sigma_1, 1), (\sigma_2, 1))$ from $(\Sigma \times \{1, 2\})^2$, where $\sigma_1 = \sigma_2 = \sigma$, to construct a source rewriting of query $Q(x_1, x_2, x_3)$. We show here how tuple $((\sigma_1, 1), (\sigma_2, 1))$ is processed.

First, the algorithm generates a tuple (ξ_1, ξ_2) from (σ_1, σ_2) by renaming the variables of σ_1 and σ_2 (in such a way that the sets of variables of ξ_1 and ξ_2 are disjoint). Assume that ξ_1 is equal to $\varphi(u_1, u_2) \rightarrow R(u_1, u_1, u_2)$ and ξ_2 equal to $\varphi(u_3, u_4) \rightarrow R(u_3, u_3, u_4)$. The algorithm continues by considering all the tuples $(\chi_1(u_1, u_2), \chi_2(u_3, u_4))$ such that $\chi_1(u_1, u_2)$ and $\chi_2(u_3, u_4)$ are nonempty conjunctions of atoms from the consequents of ξ_1 and ξ_2, respectively. In this case, the algorithm only needs to consider tuple $(R(u_1, u_1, u_2),\ R(u_3, u_3, u_4))$. The algorithm uses this tuple to construct formula $\chi(u_1, u_2, u_3, u_4) = R(u_1, u_1, u_2) \wedge R(u_3, u_3, u_4)$, and then looks for all the existential replacements of $\chi(u_1, u_2, u_3, u_4)$ that can be made identical to $\exists y_1\, \psi(x_1, x_2, x_3, y_1)$ by substituting some variables. For instance, $(\exists s_1\, (R(u_1, u_1, s_1) \wedge R(s_1, u_3, u_4)),\ u_2 = u_3)$ is one of these existential replacements: $R(g(u_1), g(u_1), s_1) \wedge R(s_1, g(u_3), g(u_4))$ is syntactically equal to $\psi(f(x_1), f(x_2), f(x_3), h(y_1))$, where $f(x_1) = f(x_2) = x_1$, $f(x_3) = x_3$, $g(u_1) = x_1$, $g(u_3) = g(u_4) = x_3$ and $h(y_1) = s_1$. The algorithm uses functions f, g and condition $u_2 = u_3$ from the existential replacement to generate the following formula $\beta(x_1, x_2, x_3)$ (omitting trivial equalities like $x_1 = x_1$):

$$\exists u_1 \exists u_2 \exists u_3 \exists u_4\, (\varphi(u_1, u_2) \wedge \varphi(u_3, u_4) \wedge u_2 = u_3 \wedge x_2 = x_1 \wedge u_1 = x_1 \wedge u_3 = x_3 \wedge u_4 = x_3).$$

Formula $\beta(x_1, x_2, x_3)$ is added to $\mathcal{C}_\psi$. It is important to notice that $\beta(x_1, x_2, x_3)$ represents a way to deduce $\exists y_1\, \psi(x_1, x_2, x_3, y_1)$ from $\varphi(x_1, x_2)$, that is, $\beta(x_1, x_2, x_3) \rightarrow \exists y_1\, \psi(x_1, x_2, x_3, y_1)$ is a logical consequence of formula (A.3).

In the last step of the algorithm, an FO formula $\alpha(x_1, x_2, x_3)$ is generated by taking the disjunction of all the formulas in $\mathcal{C}_\psi$. In particular, formula $\beta(x_1, x_2, x_3)$ above is one of these disjuncts. The algorithm returns $\alpha(x_1, x_2, x_3)$, which is a rewriting over the source of conjunctive query $Q(x_1, x_2, x_3)$. □

Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ be an st-mapping with Σ a set of FO-TO-CQ dependencies, Q a conjunctive query over $\mathbf{T}$, and Q' the output of QUERYREWRITING($\mathcal{M}, Q$). It is straightforward to prove that the algorithm runs in exponential time in the size of $\mathcal{M}$ and Q, and that the size of Q' is exponential in the size of $\mathcal{M}$ and Q. We now prove the correctness of the rewriting algorithm. We need to show that for every instance I of $\mathbf{S}$, it holds that:

$$Q'(I) = \text{certain}_{\mathcal{M}}(Q, I).$$

In this proof, we assume that Q is given by the formula $\exists \bar{y} \psi(\bar{x}, \bar{y})$, and that Q' is given by the formula $\alpha(\bar{x})$ (that could be false).

We first show that $Q'(I) \subseteq \text{certain}_{\mathcal{M}}(Q, I)$. The proof relies in the following claim.

CLAIM A.1.3. *The formula* $\forall \bar{x}(\alpha(\bar{x}) \rightarrow \exists \bar{y} \psi(\bar{x}, \bar{y}))$ *is a logical consequence of* Σ.

PROOF. If $\alpha(\bar{x})$ is false, the property trivially holds. Now, assume that $\alpha(\bar{x})$ is the disjunction of the formulas in the set $\mathcal{C}_\psi$ constructed after step 2 of the algorithm. We show that for every $\beta(\bar{x}) \in \mathcal{C}_\psi$ it holds that $\forall \bar{x}(\beta(\bar{x}) \rightarrow \exists \bar{y} \psi(\bar{x}, \bar{y}))$ is a logical consequence of Σ, which implies that $\forall \bar{x}(\alpha(\bar{x}) \rightarrow \exists \bar{y} \psi(\bar{x}, \bar{y}))$ is a logical consequence of Σ. Assume that $\beta(\bar{x})$ is equal to:

$$\exists \bar{u}_1 \cdots \exists \bar{u}_p \left(\bigwedge_{i=1}^{p} \varphi_i(\bar{u}_i) \wedge \theta(\bar{w}'') \wedge \bar{x} = f(\bar{x}) \wedge \bar{w}' = g(\bar{w}') \right),$$

where for every $i \in \{1, \ldots, p\}$, it holds that $\varphi_i(\bar{u}_i) \rightarrow \exists \bar{v}_i\, \psi_i(\bar{u}_i, \bar{v}_i)$ is a dependency in Σ. In step 2(c)i of the algorithm, formula $\exists \bar{z} \chi(\bar{w}, \bar{z})$ is defined as $\exists \bar{z}_1 \cdots \exists \bar{z}_p (\chi_1(\bar{w}_1, \bar{z}_1) \wedge$

$\cdots \wedge \chi_p(\bar{w}_p, \bar{z}_p))$, where $\chi_i(\bar{w}_i, \bar{z}_i)$ is a conjunction of atoms from $\psi_i(\bar{u}_i, \bar{v}_i)$, with $\bar{w}_i \subseteq \bar{u}_i$ and $\bar{z}_i \subseteq \bar{v}_i$. Thus, we have that sentence Φ:

$$\forall \bar{x} \left(\beta(\bar{x}) \rightarrow \exists \bar{w} \left(\exists \bar{z}\, \chi(\bar{w}, \bar{z}) \wedge \theta(\bar{w}'') \wedge \bar{x} = f(\bar{x}) \wedge \bar{w}' = g(\bar{w}')\right)\right)$$

is a logical consequence of Σ. Given that $(\exists \bar{s} \exists \bar{z}\, \gamma(\bar{w}', \bar{z}, \bar{s}), \theta(\bar{w}''))$ is an existential replacement of $\exists \bar{z}\, \chi(\bar{w}, \bar{z})$, we know that $\exists \bar{z}\, \chi(\bar{w}, \bar{z}) \wedge \theta(\bar{w}'')$ implies $\exists \bar{s} \exists \bar{z}\, \gamma(\bar{w}', \bar{z}, \bar{s})$. Thus, we have that Φ implies:

$$\forall \bar{x}\big(\beta(\bar{x}) \rightarrow \exists \bar{w}'\big(\exists \bar{s} \exists \bar{z} \gamma(\bar{w}', \bar{z}, \bar{s}) \wedge \bar{x} = f(\bar{x}) \wedge \bar{w}' = g(\bar{w}')\big)\big).$$

Now, we can safely replace $\bar{w}'$ by $g(\bar{w}')$, and drop the conjunction $\bar{w}' = g(\bar{w}')$ and the existential quantification over $\bar{w}'$. Then we obtain that sentence:

$$\forall \bar{x}\big(\beta(\bar{x}) \rightarrow \exists \bar{s} \exists \bar{z}\, \gamma(g(\bar{w}'), \bar{z}, \bar{s}) \wedge \bar{x} = f(\bar{x})\big)$$

is a logical consequence of Φ. Thus, given that $\gamma(g(\bar{w}'), \bar{z}, \bar{s})$ is syntactically equal to $\psi(f(\bar{x}), h(\bar{y}))$, we know that $\forall \bar{x}(\beta(\bar{x}) \rightarrow \exists \bar{s} \exists \bar{z}\, \psi(f(\bar{x}), h(\bar{y})) \wedge \bar{x} = f(\bar{x}))$ is also a consequence of Φ. In this last formula, we can replace $f(\bar{x})$ by $\bar{x}$ and drop the conjunction $\bar{x} = f(\bar{x})$, obtaining $\forall \bar{x}(\beta(\bar{x}) \rightarrow \exists \bar{s} \exists \bar{z}\, \psi(\bar{x}, h(\bar{y})))$. Since h is a function from $\bar{y}$ to $(\bar{z}, \bar{s})$, we have that $\exists \bar{z} \exists \bar{s}\, \psi(\bar{x}, h(\bar{y}))$ logically implies formula $\exists \bar{y}\, \psi(\bar{x}, \bar{y})$ (because the variables in $\bar{y}$ are all distinct). We have shown that $\forall \bar{x}(\beta(\bar{x}) \rightarrow \exists \bar{y}\, \psi(\bar{x}, \bar{y}))$ is a logical consequence of Φ and, therefore, it is a logical consequence of Σ. This concludes the proof of the claim. $\square$

We prove now that $Q'(I) \subseteq \mathrm{certain}_{\mathcal{M}}(Q, I)$ for every instance $I \in \mathrm{Inst}(\mathbf{S})$, by using the above claim. Let I be an arbitrary instance, and assume that $\bar{a}$ is a tuple of constant values such that $\bar{a} \in Q'(I)$. We need to show that for every $J \in \mathrm{Sol}_{\mathcal{M}}(I)$ it holds that $\bar{a} \in Q(J)$. Since $\bar{a} \in Q'(I)$ we know that $I \models \alpha(\bar{a})$. Now let $J \in \mathrm{Sol}_{\mathcal{M}}(I)$. From the Claim A.1.3 we know that $\forall \bar{x}(\alpha(\bar{x}) \rightarrow \exists \bar{y} \psi(\bar{x}, \bar{y}))$ is a logical consequence of Σ. Then since $(I, J) \models \Sigma$ and $I \models \alpha(\bar{a})$, it holds that $J \models \exists \bar{y} \psi(\bar{a}, \bar{y})$, which implies that $\bar{a} \in Q(J)$. Thus we have that for every $J \in \mathrm{Sol}_{\mathcal{M}}(I)$ it holds that $\bar{a} \in Q(J)$. This was to be shown.

We now prove that $\mathrm{certain}_{\mathcal{M}}(Q, I) \subseteq Q'(I)$ for every instance I. First recall that given an instance I of $\mathbf{S}$, the instance $\mathrm{chase}_{\Sigma}(I)$ of $\mathbf{T}$ is constructed with the following procedure (see Section 2.4). For every dependency $\sigma \in \Sigma$ of the form $\varphi(\bar{x}) \rightarrow \exists \bar{y}\, \nu(\bar{x}, \bar{y})$, with $\bar{x} = (x_1, \ldots, x_m)$, $\bar{y} = (y_1, \ldots, y_\ell)$ tuples of distinct variables, and for every m-tuple $\bar{a}$ of elements from $\mathrm{dom}(I)$ such that $I \models \varphi(\bar{a})$, do the following. Choose an ℓ-tuple $\bar{n}$ of distinct fresh values from $\mathbf{N}$, and include all the conjuncts of $\nu(\bar{a}, \bar{n})$ in $\mathrm{chase}_{\Sigma}(I)$. We say that the conjuncts of $\psi(a_1, \ldots, a_m, n_1, \ldots, n_\ell)$ included in $\mathrm{chase}_{\Sigma}(I)$ are *generated* (or *justified*) by σ.

We also make use of the notion of $\mathbf{N}$-*connected* instances introduced in the Section 3.4 when proving Lemma 3.4.2 and Theorem 3.4.3. Recall that an instance I of $\mathbf{S}$ is $\mathbf{N}$-*connected* if the following holds. Let $G_I = (V_I, E_I)$ be a graph such that V_I is composed by all the tuples $t \in R^I$ for $R \in \mathbf{S}$, and there is an edge in E_I between tuples t_1 and t_2 if there exists a value $n \in \mathbf{N}$ that is mentioned both in t_1 and t_2. Then I is $\mathbf{N}$-connected if the graph G_I is connected. An instance I_1 is an $\mathbf{N}$-*connected sub-instance* of I, if I_1 is a sub-instance of I and I_1 is $\mathbf{N}$-connected. Finally, I_1 is an $\mathbf{N}$-*connected component* of I, if I_1 is an $\mathbf{N}$-connected sub-instance of I and there is no $\mathbf{N}$-connected sub-instance I_2 of I such that I_1 is a proper sub-instance of I_2. We extend these definitions for formulas that are conjunctions of atoms. Let $\varphi(\bar{x})$ be a conjunction of atoms, and $\bar{a}$ a tuple of values in $\mathbf{C} \cup \mathbf{N}$. We say that $\varphi(\bar{a})$ is $\mathbf{N}$-connected, if the instance that contains exactly the atoms of $\varphi(\bar{a})$ is $\mathbf{N}$-connected. The definition of $\mathbf{N}$-connected components of a conjunction of atoms $\psi(\bar{a})$, is defined as for the case of instances. Notice that if I is such that $\mathrm{dom}(I) \subseteq \mathbf{C}$, then every atom in an $\mathbf{N}$-connected sub-instance of $\mathrm{chase}_{\Sigma}(I)$ is generated by a single dependency in Σ.

We are ready now to prove that $\mathrm{certain}_{\mathcal{M}}(Q, I) \subseteq Q'(I)$ for every instance I in $\mathbf{S}$. Let I be an arbitrary instance of $\mathbf{S}$. We use the following property of $\mathrm{chase}_{\Sigma}(I)$ (see Section 2.4). Since Q is a conjunctive query, we know that $\mathrm{certain}_{\mathcal{M}}(Q, I) = Q(\mathrm{chase}_{\Sigma}(I))_{\downarrow}$, where $Q(\mathrm{chase}_{\Sigma}(I))_{\downarrow}$ denotes the set of tuples in $Q(\mathrm{chase}_{\Sigma}(I))$ composed only by constant values. Thus, in order to prove that $\mathrm{certain}_{\mathcal{M}}(Q, I) \subseteq Q'(I)$, it is enough to prove that $Q(\mathrm{chase}_{\Sigma}(I))_{\downarrow} \subseteq Q'(I)$. Next we show this last property.

Recall that Q is defined by formula $\exists\bar{y}\psi(\bar{x},\bar{y})$ and Q' by $\alpha(\bar{x})$. Assume that $\bar{x}$ is the tuple of distinct variables $(x_1,\ldots,x_r)$ and let $\bar{a}=(a_1,\ldots,a_r)$ be a tuple of constant values such that $\bar{a}\in Q(\mathrm{chase}_\Sigma(I))_\downarrow$. Then we know that $\mathrm{chase}_\Sigma(I)\models\exists\bar{y}\psi(\bar{a},\bar{y})$. We need to show that $\bar{a}\in Q'(I)$, that is, we need to show that $I\models\alpha(\bar{a})$. In order to prove this last fact, we show that after step 2 of the algorithm, there exists a formula $\beta(\bar{x})\in\mathcal{C}_\psi$ such that $I\models\beta(\bar{a})$.

Assume that in formula $\psi(\bar{x},\bar{y})$, $\bar{y}$ is the tuple of distinct variables $(y_1,\ldots,y_\ell)$. Since $\mathrm{chase}_\Sigma(I)\models\exists\bar{y}\psi(\bar{a},\bar{y})$, we know that there exists a tuple $\bar{b}=(b_1,\ldots,b_\ell)$ composed by constant and null values,such that $\mathrm{chase}_\Sigma(I)\models\psi(\bar{a},\bar{b})$. Let $\rho_1(\bar{a}_1,\bar{b}_1),\ldots,\rho_p(\bar{a}_p,\bar{b}_p)$ be the $\mathbf{N}$-connected components of $\psi(\bar{a},\bar{b})$, and assume that $\rho_i(\bar{a}_i,\bar{b}_i)$ is a conjunction of k_i (not necessarily distinct) atoms. Notice that if $\psi(\bar{x},\bar{y})$ has m atoms, then $k_1+\cdots+k_p=m$. Without loss of generality, we can assume that $\psi(\bar{a},\bar{b})=\rho_1(\bar{a}_1,\bar{b}_1)\wedge\ldots\wedge\rho_p(\bar{a}_p,\bar{b}_p)$ (otherwise we can always reorder the atoms in $\psi(\bar{a},\bar{b})$). Since $\mathrm{chase}_\Sigma(I)\models\psi(\bar{a},\bar{b})$, we know that for every $i\in\{1,\ldots,p\}$, the conjuncts of $\rho_i(\bar{a}_i,\bar{b}_i)$ are included in the same $\mathbf{N}$-connected sub-instance of $\mathrm{chase}_\Sigma(I)$. Furthermore, as we have noted before, for every set of facts J that forms an $\mathbf{N}$-connected sub-instance of $\mathrm{chase}_\Sigma(I)$, there exists a sentence in Σ that justifies J. Then there exist p (not necessarily distinct) sentences $(\sigma_1,\ldots,\sigma_p)\in\Sigma^p$, such that the atoms in $\rho_i(\bar{a}_i,\bar{b}_i)$ are generated by σ_i. Let $(\xi_1,\ldots,\xi_p)$ be a tuple of dependencies obtained by renaming the variables of $(\sigma_1,\ldots,\sigma_p)$ in such a way that the set of variables of the formulas $\xi_1,\ldots,\xi_p$ are pairwise disjoint. Assume that every ξ_i is of the form $\varphi_i(\bar{u}_i)\rightarrow\exists\bar{v}_i\psi_i(\bar{u}_i,\bar{v}_i)$. Since σ_i generates all the atoms in $\rho_i(\bar{a}_i,\bar{b}_i)$, we know that for every $i\in\{1,\ldots,p\}$, there exists a formula $\chi_i(\bar{w}_i,\bar{z}_i)$, and tuples $\bar{c}_i$ and $\bar{n}_i$ of values in $\mathbf{C}$ and $\mathbf{N}$, respectively, such that $\chi_i(\bar{w}_i,\bar{z}_i)$ is a conjunction of k_i (not necessarily distinct) atoms from $\psi_i(\bar{u}_i,\bar{v}_i)$ with $\bar{w}_i\subseteq\bar{u}_i$ and $\bar{z}_i\subseteq\bar{v}_i$, and such that $\chi_i(\bar{c}_i,\bar{n}_i)$ is syntactically equal to $\rho_i(\bar{a}_i,\bar{b}_i)$, up to reordering of atoms. Without loss of generality we can assume that $\chi_i(\bar{c}_i,\bar{n}_i)=\rho_i(\bar{a}_i,\bar{b}_i)$. Let $\chi(\bar{w},\bar{z})=\chi_1(\bar{w}_1,\bar{z}_1)\wedge\cdots\wedge\chi_p(\bar{w}_p,\bar{z}_p)$, with $\bar{w}=(\bar{w}_1,\ldots,\bar{w}_p)=(w_1,\ldots,w_d)$ and $\bar{z}=(\bar{z}_1,\ldots,\bar{z}_p)=(z_1,\ldots,z_e)$ tuples of distinct variables. Then we have that $\chi(\bar{c},\bar{n})=\psi(\bar{a},\bar{b})$, where $\bar{c}=(\bar{c}_1,\ldots,\bar{c}_p)$ is a tuple of values in $\mathbf{C}$, and $\bar{n}=(\bar{n}_1,\ldots,\bar{n}_p)$ is a tuple of values in $\mathbf{N}$. Given that the conjuncts of $\rho_i(\bar{a}_i,\bar{b}_i)$

are facts in $\mathrm{chase}_\Sigma(I)$, and each $\rho_i(\bar{a}_i, \bar{b}_i) = \chi_i(\bar{c}_i, \bar{n}_i)$ is an **N**-connected component of $\psi(\bar{a}, \bar{b})$, we have that $\bar{n}$ is a tuple of distinct values in **N** (since tuples $\bar{n}_i$ and $\bar{n}_j$ do not share any values, for every $i \neq j$). Through the rest of the proof, we assume that $\bar{c} = (c_1, \ldots, c_d)$ and $\bar{n} = (n_1, \ldots, n_e)$, that is, for every $i \in \{1, \ldots, d\}$, c_i is the value assigned to variable w_i, and for every $i \in \{1, \ldots, e\}$, n_i is the value assigned to z_i.

Focus now in the *positions* of $\psi(\bar{x}, \bar{y})$. For every $i \in \{1, \ldots, r\}$, we call *x_i-position* to a position in $\psi(\bar{x}, \bar{y})$ where variable x_i occurs. Similarly, for every $i \in \{1, \ldots, \ell\}$, a *$y_i$-position* is a position in $\psi(\bar{x}, \bar{y})$ where variable y_i occurs. Since $\psi(\bar{a}, \bar{b})$ and $\chi(\bar{c}, \bar{n})$ are syntactically equal, there is a one-to-one correspondence between the positions in $\psi(\bar{x}, \bar{y})$ and the positions in $\chi(\bar{w}, \bar{z})$. Then we can talk about x_i- or y_i-positions in general when referring to positions in $\psi(\bar{x}, \bar{y})$ or in $\chi(\bar{w}, \bar{z})$. We use this correspondence of positions and the fact that $\psi(\bar{a}, \bar{b}) = \chi(\bar{c}, \bar{n})$, to create an existential replacement, and functions f, g, and h, as in step 2(c)ii of the algorithm.

We know that $\bar{a}$ is a tuple of constant values. Then from $\psi(\bar{a}, \bar{b}) = \chi(\bar{c}, \bar{n})$, we obtain that every element of $\bar{n}$ is equal to an element of $\bar{b}$. Furthermore, this last fact implies that every variable of $\bar{z}$ occurs in a y_i-position of $\chi(\bar{w}, \bar{z})$, otherwise, it could not be the case that $\psi(\bar{a}, \bar{b}) = \chi(\bar{c}, \bar{n})$. Consider now the variables y_i such that a variable of $\bar{w}$ occurs in a y_i-position of $\chi(\bar{w}, \bar{z})$. Construct an existential replacement of $\exists \bar{z} \chi(\bar{w}, \bar{z})$ where, for every such variable y_i, all the y_i-positions are replaced by an existentially quantified variable s_i. Let $(\exists \bar{s} \exists \bar{z} \gamma(\bar{w}', \bar{z}, \bar{s}), \theta(\bar{w}''))$ be such a replacement of $\exists \bar{z} \chi(\bar{w}, \bar{z})$. Notice that in the formula $\gamma(\bar{w}', \bar{z}, \bar{s})$, every variable of $\bar{w}'$ occurs in an x_i-position. We define now function h as follows. Let $h : \bar{y} \rightarrow (\bar{z}, \bar{s})$ be a function such that, $h(y_i) = z_j$ if z_j occurs in a y_i-position, and $h(y_i) = s_i$ otherwise. Notice that h is well defined since if variable z_j occurs in a y_i-position, then z_j occurs in every y_i-position (given that $\bar{n}$ is a tuple of distinct values of **N**, $\bar{c}$ is a tuple of values of **C**, and $\chi(\bar{c}, \bar{n}) = \psi(\bar{a}, \bar{b})$). We define now functions $f : \bar{x} \rightarrow \bar{x}$ and $g : \bar{w}' \rightarrow \bar{x}$. For that purpose, we construct first a partition of the set of variables of $(\bar{x}, \bar{w}')$, and then, we let f and g assign to every variable a representative of its equivalent class. Consider then, for every value a in $\bar{a}$, the set V_a of all the variables x_i of $\bar{x}$ such that x_i is assigned value a (that is, $a_i = a$), plus all the variables w_j of $\bar{w}'$

such that w_j is assigned value a (that is, $c_j = a$). Note that, since $\chi(\bar{c}, \bar{n}) = \psi(\bar{a}, \bar{b})$ and every variable of $\bar{w}'$ occurs in an x_i-position, sets V_a do form a partition of $(\bar{x}, \bar{w}')$. Choose as a representative of every equivalent class, the variable x_i with minimum index in the equivalent class. Then let f and g be such that, $f(x_i) = x_j$ if x_j is the representative of V_{a_i}, and similarly $g(w_i) = x_j$ if x_j is the representative of V_{c_i}. By the definition of the existential replacement, and the definitions of functions f, g, and h, and since $\psi(\bar{a}, \bar{b}) = \chi(\bar{c}, \bar{n})$, we have that $\psi(f(\bar{x}), h(\bar{y}))$ and $\gamma(g(\bar{w}'), \bar{z}, \bar{s})$ are syntactically equal (they coincide in every x_i- and y_i-position). Then we know that the formula:

$$\beta(\bar{x}) \;=\; \exists \bar{u}_1 \cdots \exists \bar{u}_p \left(\bigwedge_{i=1}^{p} \varphi_i(\bar{u}_i) \wedge \theta(\bar{w}'') \wedge \bar{x} = f(\bar{x}) \wedge \bar{w}' = g(\bar{w}') \right),$$

is added to $\mathcal{C}_\psi$ after step 2 of the algorithm. We claim that $I \models \beta(\bar{a})$.

Next we show that $I \models \varphi_1(\bar{c}_1^\star) \wedge \cdots \wedge \varphi_p(\bar{c}_p^\star) \wedge \theta(\bar{c}'') \wedge \bar{a} = f(\bar{a}) \wedge \bar{c}' = g(\bar{c}')$, where $\bar{c}_i^\star$ is a tuple of elements in $\mathbf{C}$ that contains $\bar{c}_i$, $\bar{c}'$ is the tuple obtained by restricting $\bar{c}$ to the variables of $\bar{w}'$, and $\bar{c}''$ is the tuple obtained by restricting $\bar{c}$ to the variables of $\bar{w}''$. Notice that an equality $w_j = w_k$ appears in the formula $\theta(\bar{w}'')$ if $j \neq k$ and both w_j and w_k occur in a y_i-position. Then since $\psi(\bar{a}, \bar{b}) = \chi(\bar{c}, \bar{n})$, we know that b_i (the value assigned to y_i) is equal to both c_j and c_k and, thus, $c_j = c_k$ holds. We conclude that $\theta(\bar{c}'')$ holds. Consider now equality $\bar{a} = f(\bar{a})$. We know by the definition of f that $f(x_i) = x_j$, if x_j is the representative of V_{a_i}. Thus, we have that $a_i = a_j$, which implies that $\bar{a} = f(\bar{a})$ holds. Next consider equality $\bar{c}' = g(\bar{c}')$. We know by the definition of g that $g(w_i) = x_j$, if x_j is the representative of V_{c_i}. Thus, we have that $c_i = a_j$, which implies that $\bar{c}' = g(\bar{c}')$ holds. Finally, given that for every $i \in \{1, \ldots, p\}$, formula $\psi_i(\bar{a}_i, \bar{b}_i) = \chi_i(\bar{c}_i, \bar{n}_i)$ is justified by dependency $\varphi_i(\bar{u}_i) \rightarrow \exists \bar{v}_i \psi_i(\bar{v}_i, \bar{w}_i)$, there exists a tuple $\bar{c}_i^\star$ that contains the elements in $\bar{c}_i$ and such that $I \models \varphi_i(\bar{c}_i^\star)$. We have shown that $I \models \varphi_1(\bar{c}_1^\star) \wedge \cdots \wedge \varphi_p(\bar{c}_p^\star) \wedge \theta(\bar{c}'') \wedge \bar{a} = f(\bar{a}) \wedge \bar{c}' = g(\bar{c}')$, and, hence, $I \models \beta(\bar{a})$.

We have shown that if $\mathrm{chase}_\Sigma(I) \models \exists \bar{y} \psi(\bar{a}, \bar{y})$ for a tuple $\bar{a}$ of constants, then there exists a formula $\beta(\bar{x}) \in \mathcal{C}_\psi$ such that $I \models \beta(\bar{a})$. Thus, since $\alpha(\bar{x})$ is the disjunctions of the formulas in $\mathcal{C}_\psi$, we have that $I \models \alpha(\bar{a})$. Recall that $\exists \bar{y} \psi(\bar{x}, \bar{y})$ defined query

Q and $\alpha(\bar{x})$ defines query Q'. Therefore, if a tuple $\bar{a}$ of constants is such that $\bar{a} \in Q(\mathrm{chase}_\Sigma(I))$ then we have that $\bar{a} \in Q'(I)$, which implies that $Q(\mathrm{chase}_\Sigma(I))_\downarrow \subseteq Q'(I)$ and then $\mathrm{certain}_\mathcal{M}(Q, I) \subseteq Q'(I)$ which is the property that we wanted to obtain. This completes the proof of correctness of the algorithm.

A.1.2. Proof of Lemma 3.3.9

The following algorithm computes a rewriting of a conjunctive query given by a single atom without existential quantifiers.

Algorithm QUERYREWRITINGATOM($\mathcal{M}, Q$)

Input: An st-mapping $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ where Σ is a set of FO-TO-CQ dependencies, and a conjunctive query Q given by a single atom over $\mathbf{T}$ without existential quantifiers.

Output: An FO query Q' that is a rewriting of Q over the source schema $\mathbf{S}$.

(1) Construct a set Σ' of dependencies as follows. Start with $\Sigma' = \emptyset$. For every dependency $\sigma \in \Sigma$ of the form $\varphi(\bar{u}) \rightarrow \exists \bar{v} \psi(\bar{u}, \bar{v})$ do the following.
   (a) For every atom $P(\bar{u}')$ that is a conjunct in $\psi(\bar{u}, \bar{v})$ such that $\bar{u}' \subseteq \bar{u}$, add dependency $\varphi'(\bar{u}') \rightarrow P(\bar{u}')$ to Σ', where $\varphi'(\bar{u}') = \exists \bar{u}'' \varphi(\bar{u})$ with $\bar{u}''$ the tuple of variables in $\bar{u}$ that are not mentioned in $\bar{u}'$.

(2) Rename the variables of the dependencies in Σ' in such a way that the obtained dependencies have pairwise disjoint sets of variables.

(3) Assume that Q is given by the atom $R(\bar{x})$, where $\bar{x}$ is a tuple of not necessarily distinct variables that are not mentioned in the dependencies of Σ'.

(4) Create a set $\mathcal{C}_R$ of FO queries as follows. Start with $\mathcal{C}_R = \emptyset$. Then for every dependency $\varphi(\bar{z}) \rightarrow R(\bar{z})$ in Σ', add formula $\exists \bar{z}(\varphi(\bar{z}) \wedge \bar{z} = \bar{x})$ to $\mathcal{C}_R$.

(5) If $\mathcal{C}_R$ is nonempty, then let $\alpha(\bar{x})$ be the FO formula constructed as the disjunction of all the formulas in $\mathcal{C}_R$. Otherwise, let $\alpha(\bar{x})$ be false, that is, an arbitrary unsatisfiable formula (with $\bar{x}$ as its tuple of free variables).

(6) Return the query Q' given by $\alpha(\bar{x})$. □

It is straightforward to see that the algorithm runs in time $O(\|\Sigma\|^2)$ in the general case, and in time $O(\|\Sigma\|)$ if Σ is a set of full FO-TO-CQ dependencies, each dependency with a single atom in its conclusion. Just notice that in the latter case, the set Σ' constructed in the step 1 of the algorithm is of size linear in the size of Σ. The proof of correctness follows directly from the correctness of algorithm QUERYREWRITING of Lemma 3.3.1. Just observe that if the input of the algorithm QUERYREWRITING is a query Q given by the single atom $R(\bar{x})$ with no existentially quantified variables, then in the step 2 of the algorithm the parameter m is equal to 1. Also notice that an atom with existentially quantified variables cannot be transformed into $R(\bar{x})$ by applying existential replacements and variable substitutions.

A.2. Strong Determination and Target Rewritability in Schema Mappings

The goal of this section is to prove Lemmas 6.2.3 and 6.2.4. These proofs are included in Sections A.2.1 and Section A.2.2, respectively. But for presenting these proofs, we first need to recall the notion of *strong determination* and apply it to the classical setting of query rewriting using views (Levy et al., 1995; Segoufin & Vianu, 2005).

We introduce first the needed terminology regarding *views* and *rewriting of queries using views*. A set of *views* $\mathcal{V}$ over a schema $\mathbf{S}$ is a non-empty set of queries over $\mathbf{S}$. We assume that every n-ary query Q in a set of views has an associated *name* R_Q that is an n-ary relational symbol. If $\mathcal{V}$ is the set of views $\{Q_1, Q_2, \ldots, Q_n\}$ we say that $\{R_{Q_1}, R_{Q_2}, \ldots, R_{Q_n}\}$ is the *schema* of $\mathcal{V}$.

Given a set of views $\mathcal{V}$ over schema $\mathbf{S}$, an instance I of $\mathbf{S}$ and a query Q over schema $\mathcal{V}$, the evaluation of Q over I can be computed following two strategies. The first one is to *unfold* in Q the definition of each view and then evaluate the resulting query over I. We denote by Q^{unf} the query obtained after unfolding each view definition in Q. Thus, the first strategy is to compute $Q^{\text{unf}}(I)$. The second strategy is to *materialize* the evaluation of each view to construct a new instance over schema $\mathcal{V}$ and then simply evaluate Q directly over this materialization. We denote by $\text{chase}_{\mathcal{V}}(I)$ the new materialized instance

over schema $\mathcal{V}$ obtained from I. We use this notation since the new instance can be obtained by using the standard chase procedure: for every query Q in $\mathcal{V}$ we compute $Q(I)$ and include all the obtained tuples in $(R_Q)^{\text{chase}_\mathcal{V}(I)}$. Thus, the second strategy amounts to compute $Q(\text{chase}_\mathcal{V}(I))$. Notice that for every query Q over $\mathcal{V}$ and every instance I of $\mathbf{S}$ it holds that $Q^{\text{unf}}(I) = Q(\text{chase}_\mathcal{V}(I))$.

We recall now the notion of *strong determination*[1]. Given two instances I_1 and I_2 over $\mathbf{S}$ and a set $\mathcal{V} = \{Q_1, \ldots, Q_n\}$ of views (queries) over $\mathbf{S}$, we use $\mathcal{V}(I_1) \subseteq \mathcal{V}(I_2)$ to denote that $Q_i(I_1) \subseteq Q_i(I_2)$ for every $i \in \{1, \ldots, n\}$. Notice that $\mathcal{V}(I_1) \subseteq \mathcal{V}(I_2)$ if and only if $\text{chase}_\mathcal{V}(I_1) \subseteq \text{chase}_\mathcal{V}(I_2)$. We say that query Q over $\mathbf{S}$ is *rewritable using* $\mathcal{V}$ if there exists a query $Q_\mathcal{V}$ over schema $\mathcal{V}$ such that the evaluation of $Q_\mathcal{V}$ over I is the same as the evaluation of Q over I for every instance I of $\mathbf{S}$. That is, for every instance I of $\mathbf{S}$ it holds that $Q_\mathcal{V}(\text{chase}_\mathcal{V}(I)) = Q(I)$, or equivalently that $(Q_\mathcal{V})^{\text{unf}}(I) = Q(I)$. Finally, we say that $\mathcal{V}$ *strongly determines* Q, and write $\mathcal{V} \Rightarrowtriangle Q$, when for every pair of instances I_1 and I_2 of $\mathbf{S}$ if $\mathcal{V}(I_1) \subseteq \mathcal{V}(I_2)$ then $Q(I_1) \subseteq Q(I_2)$.

Segoufin and Vianu (2005) introduced the notion of *determinacy* of queries given a set of views which is tightly related to our notion of strong determination. A set of views $\mathcal{V}$ over $\mathbf{S}$ *determines* a query Q if for every pair of $\mathbf{S}$ instances I_1, I_2, it holds that $\mathcal{V}(I_1) = \mathcal{V}(I_2)$ implies $Q(I_1) = Q(I_2)$. It is clear form the definition of strong determination that if $\mathcal{V}$ strongly determines Q, then $\mathcal{V}$ determines Q according to the definition introduced by Segoufin and Vianu (2005). Moreover, Segoufin and Vianu (2005) studied the relationship between the notion of determinacy and rewriting of queries using views. In particular, they show that there may exist sets of monotone views $\mathcal{V}$ that determines a monotone query Q that cannot be rewritten as a monotone query over $\mathcal{V}$. In contrast we show in this section that for the case of strong determination this characterization is possible. That is, a set of views $\mathcal{V}$ strongly determines a query Q if and only if Q can be rewritten as a monotone query over $\mathcal{V}$ (see Lemma A.2.3).

[1]We use a slightly different notation compared with the definition of strong determination in Section 6.2.1 since in this section we call *views* to the set of queries over the schema $\mathbf{S}$ that participates in the definition of strong determintation.

Before stating our main result regarding strong determination and rewriting we need to introduce a technical lemma about a normal form for general queries.

LEMMA A.2.1. *Every n-ary consistent query Q with $n \geq 1$ over a source schema $\mathbf{S}$ can be defined as a nonempty (and possibly infinitary) disjunction of* FO *formulas of the form*

$$\gamma(\bar{x}) \;=\; \exists \bar{y}\Big(\alpha(\bar{x}', \bar{y}) \wedge \delta_{(\bar{x}', \bar{y})} \wedge \chi(\bar{x}', \bar{y}) \wedge \theta(\bar{x}', \bar{x}'') \wedge \forall u\, \omega_{(\bar{x}', \bar{y})}(u)\Big) \tag{A.4}$$

where:

- *$\bar{x}$ is an n-ary tuple of variables, $\bar{x}'$ and $\bar{x}''$ are tuples of variables in $\bar{x}$ with no variables in common, and $\bar{x} = (\bar{x}', \bar{x}'')$,*
- *$\alpha(\bar{x}', \bar{y})$ is a nonempty conjunction of relational atoms over $\mathbf{S}$ that mentions exactly variables $(\bar{x}', \bar{y})$,*
- *$\delta_{(\bar{x}', \bar{y})}$ is a conjunction of inequalities of the form $u \neq v$ for every pair of distinct variables u and v in $(\bar{x}', \bar{y})$,*
- *$\chi(\bar{x}', \bar{y})$ is a (possibly empty) conjunction of negations of relational atoms of the form $\neg R(\bar{u})$ for every m-ary relational symbol $R \in \mathbf{S}$ and m-ary tuple $\bar{u}$ of variables in $(\bar{x}', \bar{y})$ such that $R(\bar{u})$ is not a conjunct in $\alpha(\bar{x}', \bar{y})$.*
- *$\theta(\bar{x}', \bar{x}'')$ is a conjunction of equalities of the form $u = v$ with u a variable in $\bar{x}'$ and v a variable in $\bar{x}''$ and such that all the variables in $\bar{x}''$ are mentioned in at least one equality,*
- *$\omega_{(\bar{x}', \bar{y})}(u)$ is a disjunction of equalities of the form $u = v$ for every variable v in $(\bar{x}', \bar{y})$.*

PROOF. Let Q be a n-ary query over $\mathbf{S}$. Consider the set of all the pairs $(I, (a_1, \ldots, a_n))$ where $I \in \mathrm{Inst}(\mathbf{S})$, $(a_1, \ldots, a_n) \in \mathrm{adom}(I)^n$ and $(a_1, \ldots, a_n) \in Q(I)$. It is clear that Q can be defined as a (possibly infinitary) disjunction of formulas $\gamma_{(I,(a_1,\ldots,a_n))}(x_1, \ldots, x_n)$ that define the instance I up to isomorphism replacing elements in $(a_1, \ldots, a_n)$ by free variables $(x_1, \ldots, x_n)$ and the remaining elements by existentially quantified variables (plus the

necessary equalities $x_i = x_j$ whenever $a_i = a_j$). Every formula $\gamma_{(I,(a_1,\ldots,a_n))}(x_1,\ldots,x_n)$ can be written in the form (A.4). □

The next lemma deals with the case when Q is a Boolean query. We need to consider separately the case in which Q is satisfied by the empty instance. The empty instance $I_\emptyset$ over $\mathbf{S}$ is an instance such that for every relational symbol R in $\mathbf{S}$ it holds $R^{I_\emptyset} = \emptyset$. We denote by $\varphi_\emptyset^{\mathbf{S}}$ a formula over $\mathbf{S}$ such that $I \models \varphi_\emptyset^{\mathbf{S}}$ if and only if $I = I_\emptyset$. Formula $\varphi_\emptyset^{\mathbf{S}}$ can be defined as follows. Let m be the maximum of the arities of the relational symbols in $\mathbf{S}$ and consider the tuple of variables $\bar{x} = (x_1,\ldots,x_m)$. Then $\varphi_\emptyset^{\mathbf{S}} = \forall\bar{x}\alpha(\bar{x})$ where $\alpha(\bar{x})$ is a conjunction of negation of relational atoms of the form $\neg R(x_1,\ldots,x_k)$ for every k-ary relational symbol R in $\mathbf{S}$.

LEMMA A.2.2. *Let Q be a Boolean query that is neither a tautology nor inconsistent. Then Q is a disjunction of Boolean* FO *formulas where every disjunct is either of the form* (A.4)*, with $\bar{x}'$, $\bar{x}''$ and $\bar{x}$ empty tuples, or is the formula $\varphi_\emptyset^{\mathbf{S}}$.*

PROOF. We consider a construction similar to the one in the proof of Lemma A.2.1. Consider first the set of all the nonempty instances I such that $Q(I) = \underline{\text{true}}$ and then construct the disjunction of the formulas φ_I that define every I up to isomorphism. Moreover, if $Q(I_\emptyset) = \underline{\text{true}}$ then also include formula $\varphi_\emptyset^{\mathbf{S}}$ as a disjunct. □

We can now state the relationship between strong determination and rewriting of queries using views. In the lemma we do not make any assumption about the language used to specify views or queries over a schema $\mathbf{S}$.

LEMMA A.2.3. *Let $\mathcal{V}$ be a set of views over schema $\mathbf{S}$, and Q a query over schema $\mathbf{S}$. Then $\mathcal{V} \Rightarrow Q$ if and only if Q can be rewritten as a monotone query over $\mathcal{V}$.*

PROOF. In the proof we use the following notation. Given a conjunction of relational atoms $\alpha(\bar{x})$ over a schema $\mathbf{S}$, we denote by $I_{\alpha(\bar{x})}$ the instance of $\mathbf{S}$ constructed as follows: for every relational symbol $R \in \mathbf{S}$ and relational atom $R(\bar{u})$ occurring in $\alpha(\bar{x})$, we include tuple $\bar{u}$ in $R^{I_{\alpha(\bar{x})}}$.

Assume first that Q can be rewritten as a monotone query $Q_{\mathcal{V}}$ over $\mathcal{V}$. We show now that $\mathcal{V} \rightrightarrows Q$. Let I_1 and I_2 be two instances over $\mathbf{S}$ and assume that $\mathcal{V}(I_1) \subseteq \mathcal{V}(I_2)$. Then by the monotonicity of $Q_{\mathcal{V}}$ we obtain that $(Q_{\mathcal{V}})^{\mathrm{unf}}(I_1) \subseteq (Q_{\mathcal{V}})^{\mathrm{unf}}(I_2)$ and then $Q(I_1) \subseteq Q(I_2)$. We have shown that if $\mathcal{V}(I_1) \subseteq \mathcal{V}(I_2)$ then $Q(I_1) \subseteq Q(I_2)$ for every pair of instances I_1 and I_2 which implies that $\mathcal{V} \rightrightarrows Q$.

Assume now that $\mathcal{V} \rightrightarrows Q$. We need to show that Q can be rewritten as a monotone query $Q_{\mathcal{V}}$ over schema $\mathcal{V}$. First notice that if Q is an inconsistent query over $\mathbf{S}$, then it is equivalent to any inconsistent query over $\mathcal{V}$. Similarly, if Q is a tautology then any tautology over $\mathcal{V}$ is equivalent to Q. Thus, we only have to consider the case when Q is a query that is neither inconsistent nor a tautology. Then assume that Q is n-ary (with $n \geq 0$) and let $\bar{x}$ be an n-tuple of distinct variables. By Lemmas A.2.1 and A.2.2 we know that Q is defined by an FO formula $\varphi(\bar{x})$ that is a nonempty (and possibly infinitary) disjunction of formulas of the form (A.4) plus possibly a formula $\varphi_{\emptyset}^{\mathbf{S}}$ if Q is a Boolean query.

Before describing how to construct a rewriting of Q we need some technical claims. First, assume that Q is not Boolean, thus tuple $\bar{x}$ in $\varphi(\bar{x})$ is not empty and every disjunct in $\varphi(\bar{x})$ is of the form (A.4). Let $\gamma(\bar{x}) = \exists \bar{y}\big(\alpha(\bar{x}', \bar{y}) \wedge \delta_{(\bar{x}', \bar{y})} \wedge \chi(\bar{x}', \bar{y}) \wedge \theta(\bar{x}', \bar{x}'') \wedge \forall u \; \omega_{(\bar{x}', \bar{y})}(u)\big)$ be one of the disjunct in $\varphi(\bar{x})$ in the form (A.4). Consider the instance $I_{\alpha(\bar{x}', \bar{y})}$, we claim that every element in $\bar{x}'$ appears in $\mathrm{chase}_{\mathcal{V}}(I_{\alpha(\bar{x}', \bar{y})})$. To derive a contradiction, assume not. Then some element x of $\bar{x}'$ does not appear in $\mathrm{chase}_{\mathcal{V}}(I_{\alpha(\bar{x}', \bar{y})})$. Let $\bar{x}^{\star}$ be the tuple obtained from $\bar{x}'$ replacing the element x by a fresh element $x^{\star}$. Notice that a function that maps x to $x^{\star}$ and is the identity otherwise, is an isomorphism between $I_{\alpha(\bar{x}', \bar{y})}$ and $I_{\alpha(\bar{x}^{\star}, \bar{y})}$. Thus, since x does not appear in $\mathrm{chase}_{\mathcal{V}}(I_{\alpha(\bar{x}', \bar{y})})$ we obtain that $\mathrm{chase}_{\mathcal{V}}(I_{\alpha(\bar{x}', \bar{y})}) = \mathrm{chase}_{\mathcal{V}}(I_{\alpha(\bar{x}^{\star}, \bar{y})})$ and then $\mathcal{V}(I_{\alpha(\bar{x}', \bar{y})}) = \mathcal{V}(I_{\alpha(\bar{x}^{\star}, \bar{y})})$. Since $\mathcal{V} \rightrightarrows Q$ we obtain that $Q(I_{\alpha(\bar{x}', \bar{y})}) = Q(I_{\alpha(\bar{x}^{\star}, \bar{y})})$. Consider now the tuple $\bar{u} = (\bar{x}', \bar{v})$ where $\bar{v}$ is a tuple of elements from $\bar{x}'$ that satisfy the equalities in $\theta(\bar{x}', \bar{v})$. By the form of construction of $I_{\alpha(\bar{x}', \bar{y})}$ and $\bar{u}$, we know that $I_{\alpha(\bar{x}', \bar{y})} \models \gamma(\bar{u})$ and since $\gamma(\bar{x})$ is a disjunct in the formula defining Q, we have that $\bar{u} \in Q(I_{\alpha(\bar{x}', \bar{y})})$. Notice that $\bar{u}$ contains the element x, thus, since x does not appear in $I_{\alpha(\bar{x}^{\star}, \bar{y})}$ we have that $\bar{u} \notin Q(I_{\alpha(\bar{x}^{\star}, \bar{y})})$. This is a contradiction with the fact that $Q(I_{\alpha(\bar{x}', \bar{y})}) = Q(I_{\alpha(\bar{x}^{\star}, \bar{y})})$. We have shown that $\mathrm{chase}_{\mathcal{V}}(I_{\alpha(\bar{x}', \bar{y})})$ contains all

the elements in $\bar{x}'$. Thus, we can assume that $\mathrm{chase}_{\mathcal{V}}(I_{\alpha(\bar{x}',\bar{y})})$ is an instance of the form $I_{\beta(\bar{x}',\bar{y}')}$ where $\beta(\bar{x}',\bar{y}')$ is a nonempty conjunction of relational atoms of schema $\mathcal{V}$ and $\bar{y}'$ is a (possible empty) tuple of variables from $\bar{y}$.

Assume now that Q is Boolean. Then we know that the every disjunct in φ is either of the form (A.4) (with $\bar{x}$ the empty tuple) or is the formula $\varphi^{\mathbf{S}}_{\emptyset}$. Let γ be a disjunct in φ and assume first that $\gamma = \exists\bar{y}\big(\alpha(\bar{y}) \wedge \delta_{\bar{y}} \wedge \chi(\bar{y}) \wedge \forall u\ \omega_{\bar{y}}(u)\big)$ is in the form (A.4). We claim that $\mathrm{chase}_{\mathcal{V}}(I_{\alpha(\bar{y})})$ is not empty. To derive a contradiction, assume that $\mathrm{chase}_{\mathcal{V}}(I_{\alpha(\bar{y})})$ is the empty instance. Then we have that the evaluation of every view in $\mathcal{V}$ over $I_{\alpha(\bar{y})}$ is $\underline{\text{false}}$ which implies that $\mathcal{V}(I_{\alpha(\bar{y})}) \subseteq \mathcal{V}(I)$ for every instance I of $\mathbf{S}$. Thus, since $\mathcal{V} \rightarrowtriangle Q$ we have that $Q(I_{\alpha(\bar{y})}) \subseteq Q(I)$ for every I. Notice that $Q(I_{\alpha(\bar{y})}) = \underline{\text{true}}$, thus, $Q(I) = \underline{\text{true}}$ for every instance of I of $\mathbf{S}$. This is a contradiction since we are assuming that Q is not a tautology. We have shown that $\mathrm{chase}_{\mathcal{V}}(I_{\alpha(\bar{y})})$ is not empty. Thus, we can assume that $\mathrm{chase}_{\mathcal{V}}(I_{\alpha(\bar{y})})$ is an instance of the form $I_{\beta(\bar{y}')}$ where $\beta(\bar{y}')$ is a nonempty conjunction of relational atoms of schema $\mathcal{V}$ and $\bar{y}'$ is a (possible empty) tuple of variables from $\bar{y}$. Assume now that $\gamma = \varphi^{\mathbf{S}}_{\emptyset}$. Then we know that $Q(I_{\emptyset}) = \underline{\text{true}}$. We claim that $\mathrm{chase}_{\mathcal{V}}(I_{\emptyset})$ is not the empty instance. Similarly to the previous case, if we suppose that $\mathrm{chase}_{\mathcal{V}}(I_{\emptyset})$ is empty, we have that $\mathcal{V}(I_{\emptyset}) \subseteq \mathcal{V}(I)$ for every instance I of $\mathbf{S}$. Then from $\mathcal{V} \rightarrowtriangle Q$ we have that $Q(I_{\emptyset}) \subseteq Q(I)$ for every I and since $Q(I_{\emptyset}) = \underline{\text{true}}$ we obtain that $Q(I) = \underline{\text{true}}$ for every I. This is a contradiction since we are assuming that Q is not a tautology. We have shown that if $\varphi^{\mathbf{S}}_{\emptyset}$ is one of the disjunct in φ, then $\mathrm{chase}_{\mathcal{V}}(I_{\emptyset})$ is not empty. This implies that at least one view in $\mathcal{V}$ evaluated over the empty instance generates a nonempty result. We are assuming that evaluation of a query over an instance can only give as result elements that are present in the active domain of the instance. Thus, if a view in $\mathcal{V}$ generates a nonempty result over the empty instance, then that view is a Boolean query, and thus, the evaluation is a 0-ary predicate. Therefore, we can assume that $\mathrm{chase}_{\mathcal{V}}(I_{\emptyset})$ is an instance of the form I_{β} where β is a nonempty conjunction of 0-ary relational atoms of schema $\mathcal{V}$.

We are ready now to describe how to construct a formula $\psi(\bar{x})$ over schema $\mathcal{V}$ defining a monotone query $Q_{\mathcal{V}}$ that is a rewriting of Q. For every disjunct $\gamma(\bar{x})$ in $\varphi(\bar{x})$ of the form (A.4), if $\mathrm{chase}_{\mathcal{V}}(I_{\alpha(\bar{x}',\bar{y})}) = I_{\beta(\bar{x}',\bar{y}')}$ then we include the formula $\exists\bar{y}'\ \beta(\bar{x}',\bar{y}') \wedge \delta_{(\bar{x}',\bar{y}')} \wedge$

$\theta(\bar{x}', \bar{x}'')$ as a disjunct in $\psi(\bar{x})$. Moreover, whenever φ is a Boolean formula that contains $\varphi_{\emptyset}^{\mathbf{S}}$ as a disjunct, if $I_\beta = \mathrm{chase}_{\mathcal{V}}(I_\emptyset)$ then we include formula β as a disjunct in ψ. Let $Q_{\mathcal{V}}$ be the query over $\mathcal{V}$ defined by formula $\psi(\bar{x})$. First notice that $Q_{\mathcal{V}}$ is a monotone query. We show next that $Q_{\mathcal{V}}$ is a rewriting of Q. Thus, we need to show that for every instance I of $\mathbf{S}$, it holds that $(Q_{\mathcal{V}})^{\mathrm{unf}}(I) = Q(I)$, or equivalently that $Q_{\mathcal{V}}(\mathrm{chase}_{\mathcal{V}}(I)) = Q(I)$.

We show first that $Q(I) \subseteq Q_{\mathcal{V}}(\mathrm{chase}_{\mathcal{V}}(I))$. Let $\bar{a}$ be a (possibly empty) tuple in $Q(I)$. Then we know that there exists a disjunct $\gamma(\bar{x})$ in $\varphi(\bar{x})$ such that $I \models \gamma(\bar{a})$. Assume first that $\gamma(\bar{x}) = \exists \bar{y}\big(\alpha(\bar{x}', \bar{y}) \wedge \delta_{(\bar{x}', \bar{y})} \wedge \chi(\bar{x}', \bar{y}) \wedge \theta(\bar{x}', \bar{x}'') \wedge \forall u\, \omega_{(\bar{x}', \bar{y})}(u)\big)$ is in the form (A.4). Let $\bar{a}'$ be the tuple of elements in $\bar{a}$ that correspond to the positions of variables $\bar{x}'$ in tuple $\bar{x}$. From $I \models \gamma(\bar{a})$ and the specific form of formula $\gamma(\bar{x})$ we obtain that there exists an isomorphism $h : (\bar{x}', \bar{y}) \to \mathrm{adom}(I)$ such that $h(\bar{x}') = \bar{a}'$ and $h(I_{\alpha(\bar{x}', \bar{y})}) = I_{\alpha(h(\bar{x}', \bar{y}))} = I$. Now assume that $\mathrm{chase}_{\mathcal{V}}(I_{\alpha(\bar{x}', \bar{y})})$ is an instance of the form $I_{\beta(\bar{x}', \bar{y}')}$. Then we know that $\psi(\bar{x})$ has a disjunct of the form $\exists \bar{y}'\ \beta(\bar{x}', \bar{y}') \wedge \delta_{(\bar{x}', \bar{y}')} \wedge \theta(\bar{x}', \bar{x}'')$. On the other hand, since $I_{\alpha(\bar{x}', \bar{y})}$ and I are isomorphic instances via isomorphism h we have that $\mathrm{chase}_{\mathcal{V}}(I)$ is an instance of the form $I_{\beta(h(\bar{x}', \bar{y}'))}$. Thus, we have that $I_{\beta(h(\bar{x}', \bar{y}'))} = I_{\beta(\bar{a}', h(\bar{y}'))} \models \exists \bar{y}'\ \beta(\bar{a}', \bar{y}') \wedge \delta_{(\bar{a}, \bar{y}')}$. Moreover, if $\bar{a}''$ is the the tuple of elements in $\bar{a}$ that correspond to the positions of variables $\bar{x}''$ in tuple $\bar{x}$, from $I \models \gamma(\bar{a})$ we know that $\theta(\bar{a}', \bar{a}'')$ holds. Thus, we have that $\mathrm{chase}_{\mathcal{V}}(I) = I_{\beta(h(\bar{x}', \bar{y}'))} \models \psi(\bar{a})$ which implies that $\bar{a} \in Q_{\mathcal{V}}(\mathrm{chase}_{\mathcal{V}}(I))$. This was to be shown.

Assume now that $\gamma = \varphi_{\emptyset}^{\mathbf{S}}$. Notice that in this case we have that tuple $\bar{a}$ is the empty tuple. Now, since $I \models \gamma$ we have that $I = I_\emptyset$. Moreover, if $I_\beta = \mathrm{chase}_{\mathcal{V}}(I_\emptyset)$, then we have that ψ has a disjunct of the form β and thus $I_\beta \models \psi$ which implies that $\mathrm{chase}_{\mathcal{V}}(I_\emptyset) = \mathrm{chase}_{\mathcal{V}}(I) \models \psi$. We have shown that $\mathrm{chase}_{\mathcal{V}}(I) \models \psi$, thus the empty tuple is in $Q_{\mathcal{V}}(\mathrm{chase}_{\mathcal{V}}(I))$. This completes the proof that $Q(I) \subseteq Q_{\mathcal{V}}(\mathrm{chase}_{\mathcal{V}}(I))$.

We prove now that $Q_{\mathcal{V}}(\mathrm{chase}_{\mathcal{V}}(I)) \subseteq Q(I)$. Assume that $\bar{a}$ is a (possibly empty) tuple in $Q_{\mathcal{V}}(\mathrm{chase}_{\mathcal{V}}(I))$. Then there exists a disjunct $\xi(\bar{x})$ in $\psi(\bar{x})$ such that $\mathrm{chase}_{\mathcal{V}}(I) \models \xi(\bar{a})$. Since $\xi(\bar{x})$ is a disjunct in $\psi(\bar{x})$ we know that there exists a disjunct $\gamma(\bar{x})$ in $\varphi(\bar{x})$ such that:

a) $\gamma(\bar{x}) = \exists\bar{y}\big(\alpha(\bar{x}',\bar{y}) \wedge \delta_{(\bar{x}',\bar{y})} \wedge \chi(\bar{x}',\bar{y}) \wedge \theta(\bar{x}',\bar{x}'') \wedge \forall u\ \omega_{(\bar{x}',\bar{y})}(u)\big)$ is in the form (A.4), it holds that $I_{\beta(\bar{x}',\bar{y}')} = \mathrm{chase}_{\mathcal{V}}(I_{\alpha(\bar{x}',\bar{y})})$, and $\xi(\bar{x})$ is of the form $\exists\bar{y}'\ \beta(\bar{x}',\bar{y}') \wedge \delta_{(\bar{x}',\bar{y}')} \wedge \theta(\bar{x}',\bar{x}'')$, or

b) $\gamma = \varphi^{\mathbf{S}}_{\emptyset}$, it holds that $I_\beta = \mathrm{chase}_{\mathcal{V}}(I_\emptyset)$ and $\xi = \beta$.

Consider first case a). Given that $\mathrm{chase}_{\mathcal{V}}(I) \models \xi(\bar{a})$ we know that $\mathrm{chase}_{\mathcal{V}}(I) \models \exists\bar{y}'\ \beta(\bar{a}',\bar{y}') \wedge \delta_{(\bar{a},\bar{y}')}$ and that $\theta(\bar{a}',\bar{a}'')$ holds where $\bar{a}'$ and $\bar{a}''$ are the tuples of elements from $\bar{a}$ corresponding to the positions of variables $\bar{x}'$ and $\bar{x}''$ in tuple $\bar{x}$. Then we know that there exists a 1-1 mapping $h : (\bar{x}',\bar{y}') \to \mathrm{adom}(\mathrm{chase}_{\mathcal{V}}(I))$ such that $h(\bar{x}') = \bar{a}'$ and $I_{\beta(h(\bar{x}',\bar{y}'))} \subseteq \mathrm{chase}_{\mathcal{V}}(I)$. Consider now a 1-1 mapping g with domain $(\bar{x}',\bar{y})$ that is equal to h for every element in $(\bar{x}',\bar{y}')$ and is the identity otherwise. Notice that $I_{\alpha(\bar{x}',\bar{y})}$ and $I_{\alpha(g(\bar{x}',\bar{y}))}$ are isomorphic instances. Thus, we have that $\mathrm{chase}_{\mathcal{V}}(I_{\alpha(g(\bar{x}',\bar{y}))})$ is an instance of the form $I_{\beta(g(\bar{x}',\bar{y}'))}$ which is a subinstance of $\mathrm{chase}_{\mathcal{V}}(I)$. We have shown that $\mathrm{chase}_{\mathcal{V}}(I_{\alpha(g(\bar{x}',\bar{y}))}) \subseteq \mathrm{chase}_{\mathcal{V}}(I)$ which implies that $\mathcal{V}(I_{\alpha(g(\bar{x}',\bar{y}))}) \subseteq \mathcal{V}(I)$. Since $\mathcal{V} \rightrightarrows Q$ we obtain that $Q(I_{\alpha(g(\bar{x}',\bar{y}))}) \subseteq Q(I)$. Now, notice that $I_{\alpha(g(\bar{x}',\bar{y}))} = I_{\alpha(\bar{a}',g(\bar{y}))} \models \gamma(\bar{a})$, thus, $I_{\alpha(g(\bar{x}',\bar{y}))} \models \varphi(\bar{a})$ which implies that $\bar{a} \in Q(I_{\alpha(g(\bar{x}',\bar{y}))})$. Finally, since $Q(I_{\alpha(g(\bar{x}',\bar{y}))}) \subseteq Q(I)$ we have that $\bar{a} \in Q(I)$ which was to be shown.

Consider now case b). In this case we have that $\bar{a}$ is the empty tuple. Now, since $\mathrm{chase}_{\mathcal{V}}(I) \models \beta$ then we have that $I_\beta = \mathrm{chase}_{\mathcal{V}}(I_\emptyset) \subseteq \mathrm{chase}_{\mathcal{V}}(I)$. Therefore we have that $\mathcal{V}(I_\emptyset) \subseteq \mathcal{V}(I)$, and thus, since $\mathcal{V} \rightrightarrows Q$ we have that $Q(I_\emptyset) \subseteq Q(I)$. Notice that $I \models \gamma$ which implies that $Q(I_\emptyset)$ contains the empty tuple and then $Q(I)$ contains the empty tuple which was to be shown. This completes the proof that $Q_{\mathcal{V}}(\mathrm{chase}_{\mathcal{V}}(I)) \subseteq Q(I)$. □

A.2.1. Proof of Lemma 6.2.3

Recall that given an st-mapping $\mathcal{M} = (\mathbf{S},\mathbf{T},\Sigma)$ where Σ is a set of FO-TO-CQ st-dependencies, the set of queries (views) $\mathcal{C}_{\mathcal{M}}$ is constructed as follows (see Lemma 4.2.3): for every dependency of the form $\varphi(\bar{x}) \to \psi(\bar{x})$ in Σ the set $\mathcal{C}_{\mathcal{M}}$ contains a query that is a rewriting of $\psi(\bar{x})$ over the source schema $\mathbf{S}$. We know that such a rewriting always exists and can be expressed as an FO query (see Lemma 3.3.1). Furthermore, if Σ is a set of st-tgds, then the rewriting of $\psi(\bar{x})$ over the source can be expressed as a query in $\mathrm{UCQ}^{=}$

(see Lemma 3.3.3). Moreover, by following the proof of Lemma 3.3.1 we can conclude that every target conjunctive query Q can be rewritten as a query Q' that is a monotone query over the set of views $\mathcal{C}_\mathcal{M}$ (that is, a monotone combination of the queries in $\mathcal{C}_\mathcal{M}$). We use this last fact in the proof of Lemma 6.2.3.

PROOF OF LEMMA 6.2.3. **Only if.** Assume first that Q is target rewritable in $\mathcal{M}$. We need to show that $\mathcal{C}_\mathcal{M} \rightarrowtriangle Q$. By Lemma A.2.3 it is enough to show that Q can be rewritten as a monotone query over the schema $\mathcal{C}_\mathcal{M}$. In what follows we show this last property.

Since Q is target rewritable in $\mathcal{M}$, then there exists a query Q' over $\mathbf{T}$ such that $Q(I) = \mathrm{certain}_\mathcal{M}(Q', I)$ for every source instance I. The strategy of this part of the proof is the following. We construct first a query Q'' over $\mathbf{T}$ defined by a (possibly infinitary) disjunction of queries in $\mathbf{CQ}^{=,\neq,\mathbf{C}}$, such that $\mathrm{certain}_\mathcal{M}(Q', I) = \mathrm{certain}_\mathcal{M}(Q'', I)$ for every source instance I. We then construct a source query $Q^\star$ that is a monotone combination of the queries in $\mathcal{C}_\mathcal{M}$ and such that $\mathrm{certain}_\mathcal{M}(Q'', I) = Q^\star(I)$ for every source instance I. Finally, since $Q(I) = Q^\star(I)$ for every I, we obtain that Q can be rewritten as a monotone query over the set of views $\mathcal{C}_\mathcal{M}$, and thus, by Lemma A.2.3 we obtain that $\mathcal{C}_\mathcal{M} \rightarrowtriangle Q$.

To construct the query Q'' we first define a set $\mathrm{tip}(Q')$ as follows:

$$\begin{aligned}\mathrm{tip}(Q') = \{(J, \bar{a}) \mid \bar{a} \text{ is a tuple of elements from } \mathrm{dom}(J) \cap \mathbf{C} \text{ and} \\ \text{for every instance } J' \text{ of } \mathbf{T}, \text{ if there exists a homomorphism} \\ h : \mathrm{dom}(J) \rightarrow \mathrm{dom}(J'), \text{ then } \bar{a} \in Q'(J')\}.\end{aligned}$$

Now for every $(J, \bar{a}) \in \mathrm{tip}(Q')$ we define a formula $\chi_{(J,\bar{a})}(\bar{x})$ over $\mathbf{T}$ as follows. Assume that $\bar{a} = (a_1, \ldots, a_k)$, where $k \geq 0$, and $\mathrm{dom}(J) = \{c_1, \ldots, c_\ell, n_1, \ldots, n_m\}$, where $\{c_1, \ldots, c_\ell\} \subseteq \mathbf{C}$, $\ell \geq 0$, $\{n_1, \ldots, n_m\} \subseteq \mathbf{N}$ and $m \geq 0$. Moreover, let $\{y_1, \ldots, y_\ell, z_1, \ldots, z_m\}$ be a set of variables and ρ a substitution defined as $\rho(c_i) = y_i$ and $\rho(n_j) = z_j$, for every $i \in [1, \ell]$ and $j \in [1, m]$. Let $\bar{x} = (x_1, \ldots, x_k)$, $\bar{y} = (y_1, \ldots, y_\ell)$ and $\bar{z} =$

$(z_1, \ldots, z_m)$. Then the formula $\chi_{(J,\bar{a})}(\bar{x})$ is given by:

$$\exists \bar{y} \exists \bar{z} \left(\theta(\bar{x}, \rho(\bar{a})) \wedge \delta(\bar{y}) \wedge \mathbf{C}(\bar{y}) \wedge \left(\bigwedge_{\substack{R \in \mathbf{T} \\ \bar{t} \in R^J}} R(\rho(\bar{t})) \right) \right). \tag{A.5}$$

where $\theta(\bar{x}, \rho(\bar{a}))$ is the formula $x_1 = \rho(a_1) \wedge \cdots \wedge x_k = \rho(a_k)$, $\delta(\bar{y})$ is the conjunction of inequalities $y_i \neq y_j$ for every $i, j \in [1, \ell]$ such that $i \neq j$, and $\mathbf{C}(\bar{y})$ is the formula $\mathbf{C}(y_1) \wedge \cdots \wedge \mathbf{C}(y_\ell)$. Now consider the query Q'' over $\mathbf{T}$ given by the formula:

$$\chi(\bar{x}) \; = \bigvee_{(J,\bar{a}) \in \mathrm{tip}(Q')} \chi_{(J,\bar{a})}(\bar{x}).$$

Notice that $\chi(\bar{x})$ is a disjunction of formulas in $\mathrm{CQ}^{=,\neq,\mathbf{C}}$. We show next that $\mathrm{certain}_{\mathcal{M}}(Q', I) = \mathrm{certain}_{\mathcal{M}}(Q'', I)$.

We show first that $\mathrm{certain}_{\mathcal{M}}(Q', I) \subseteq \mathrm{certain}_{\mathcal{M}}(Q'', I)$. Assume that $\bar{a} \in \mathrm{certain}_{\mathcal{M}}(Q', I)$, and let $J^\star$ be the canonical universal solution for I under $\mathcal{M}$. Given that $J^\star$ is a universal solution for I under $\mathcal{M}$ and $\mathcal{M}$ is specified by a set of FO-TO-CQ st-dependencies, we know that if there exists a homomorphism $h : \mathrm{dom}(J^\star) \to \mathrm{dom}(J)$ then $J \in \mathrm{Sol}_{\mathcal{M}}(I)$. Then given that $\bar{a} \in \mathrm{certain}_{\mathcal{M}}(Q', I)$ and $\bar{a}$ is a tuple of constants, we have that if there exists a homomorphism $h : \mathrm{dom}(J^\star) \to \mathrm{dom}(J)$ then $\bar{a} \in Q'(J)$. This implies that $(J^\star, \bar{a}) \in \mathrm{tip}(Q')$, and thus the formula $\chi_{(J^\star,\bar{a})}(\bar{x})$ is a disjunct in $\chi(\bar{x})$. Now, by definition of $\chi_{(J^\star,\bar{a})}(\bar{x})$, we know that $J^\star \models \chi_{(J^\star,\bar{a})}(\bar{a})$. Hence, given that $\chi_{(J^\star,\bar{a})}(\bar{x})$ is closed under homomorphisms and $J^\star$ is a universal solution for I under $\mathcal{M}$, we conclude that for every $J \in \mathrm{Sol}_{\mathcal{M}}(I)$ it holds that $J \models \chi_{(J^\star,\bar{a})}(\bar{a})$. We have shown that $\chi_{(J^\star,\bar{a})}(\bar{x})$ is a disjunct in $\chi(\bar{x})$ and that for every $J \in \mathrm{Sol}_{\mathcal{M}}(I)$ it holds that $J \models \chi_{(J^\star,\bar{a})}(\bar{a})$. This implies that for every $J \in \mathrm{Sol}_{\mathcal{M}}(I)$ it holds that $J \models \chi(\bar{a})$ and thus $\bar{a} \in \mathrm{certain}_{\mathcal{M}}(Q'', I)$ which was to be shown.

We show now that $\mathrm{certain}_{\mathcal{M}}(Q'', I) \subseteq \mathrm{certain}_{\mathcal{M}}(Q', I)$. Assume that $\bar{a} \in \mathrm{certain}_{\mathcal{M}}(Q'', I)$, and again let $J^\star$ be the canonical universal solution for I under $\mathcal{M}$. We know that $J^\star \models \chi(\bar{a})$ (since $J^\star \in \mathrm{Sol}_{\mathcal{M}}(I)$), and then, there exists a disjunct $\chi_{(J,\bar{b})}(\bar{x})$ of $\chi(\bar{x})$ such that $(J, \bar{b}) \in \mathrm{tip}(Q')$ and $J^\star \models \chi_{(J,\bar{b})}(\bar{a})$. Thus, by definition of $\chi_{(J,\bar{b})}(\bar{x})$, we have that there

exists an instance J' of $\mathbf{T}$ such that (1) $(J, \bar{b}) \cong (J', \bar{a})$, (2) $(J', \bar{a}) \in \mathrm{tip}(Q')$, and (3) there exists a homomorphism $h : \mathrm{dom}(J') \to \mathrm{dom}(J^\star)$ (which is the identity on the constants). Next we use this property to prove that $\bar{a} \in \mathrm{certain}_{\mathcal{M}}(Q', I)$.

Let $J'' \in \mathrm{Sol}_{\mathcal{M}}(I)$. Given that $J^\star$ is a universal for I under $\mathcal{M}$ solution, we have that there exists a homomorphism $g : \mathrm{dom}(J^\star) \to \mathrm{dom}(J'')$. Then $h \circ g : \mathrm{dom}(J') \to \mathrm{dom}(J'')$ is a homomorphism that is the identity on the constants. Therefore, since $(J', \bar{a}) \in \mathrm{tip}(Q')$ and there is a homomorphism from J' to J'' we obtain that $\bar{a} \in Q'(J'')$. We have shown that for every $J'' \in \mathrm{Sol}_{\mathcal{M}}(I)$, it holds that $\bar{a} \in Q'(J'')$. Thus, we have that $\bar{a} \in \mathrm{certain}_{\mathcal{M}}(Q', I)$, which was to be shown.

Up to this point we have shown that $\mathrm{certain}_{\mathcal{M}}(Q', I) = \mathrm{certain}_{\mathcal{M}}(Q'', I)$ for every source instance I, and since Q' is a target rewriting of Q we have that $Q(I) = \mathrm{certain}_{\mathcal{M}}(Q'', I)$. We show next that Q'' can be rewritten back over the source as a query $Q^\star$ such that $Q^\star(I) = \mathrm{certain}_{\mathcal{M}}(Q'', I)$ for every source instance I and such that $Q^\star$ is a *monotone combination* of the queries in $\mathcal{C}_{\mathcal{M}}$.

Let $(J, \bar{a}) \in \mathrm{tip}(Q')$ and $\chi_{(J,\bar{a})}(\bar{x})$ the formula defined in (A.5). Define $\gamma_{(J,\bar{a})}(\bar{x}, \bar{y})$ as the formula obtained from $\chi_{(J,\bar{a})}(\bar{x})$ by removing the existential quantification over $\bar{y}$ and the formulas $\delta(\bar{y})$ and $\mathbf{C}(\bar{y})$, that is:

$$\gamma_{(J,\bar{a})}(\bar{x}, \bar{y}) \;=\; \exists \bar{z} \left(\theta(\bar{x}, \rho(\bar{a})) \wedge \left(\bigwedge_{\substack{R \in \mathbf{T} \\ \bar{t} \in R^J}} R(\rho(\bar{t})) \right) \right). \tag{A.6}$$

Given that $\mathcal{M}$ is a mapping specified by a set of FO-TO-CQ dependencies and $\gamma_{(J,\bar{a})}(\bar{x}, \bar{y})$ is a conjunctive query (with equalities) over $\mathbf{T}$, we have that there exists a formula $\gamma^\star_{(J,\bar{a})}(\bar{x}, \bar{y})$ that is a monotone query over $\mathcal{C}_{\mathcal{M}}$ that defines a rewriting of $\gamma_{(J,\bar{a})}(\bar{x}, \bar{y})$ over the source (see Lemma 3.3.1). Since $\gamma^\star_{(J,\bar{a})}(\bar{x}, \bar{y})$ is a source rewriting of $\gamma_{(J,\bar{a})}(\bar{x}, \bar{y})$, we have that for every source instance I and tuples $\bar{b}$ and $\bar{c}$ from $\mathrm{dom}(I)$, it holds that:

$$I \models \gamma^\star_{(J,\bar{a})}(\bar{b}, \bar{c}) \quad \text{iff} \quad K \models \gamma_{(J,\bar{a})}(\bar{b}, \bar{c}) \text{ for every } K \in \mathrm{Sol}_{\mathcal{M}}(I). \tag{A.7}$$

Consider now the formula

$$\chi^{\star}_{(J,\bar{a})}(\bar{x}) \;=\; \exists \bar{y}\Big(\gamma^{\star}_{(J,\bar{a})}(\bar{x},\bar{y}) \wedge \delta(\bar{y})\Big)$$

and the query $Q^{\star}$ defined by the formula

$$\chi^{\star}(\bar{x}) \;=\; \bigvee_{(J,\bar{a})\in \mathrm{tip}(Q')} \chi^{\star}_{(J,\bar{a})}(\bar{x}).$$

Notice that since $\gamma^{\star}_{(J,\bar{a})}(\bar{x},\bar{y})$ is a monotone query over $\mathcal{C}_{\mathcal{M}}$ we have that $\chi^{\star}_{(J,\bar{a})}(\bar{x})$ is a monotone query over $\mathcal{C}_{\mathcal{M}}$, and thus, $\chi^{\star}(\bar{x})$ is also a monotone query over $\mathcal{C}_{\mathcal{M}}$. We show next that $Q^{\star}(I) = \mathrm{certain}_{\mathcal{M}}(Q'', I)$ for every source instance I.

We prove first that $\mathrm{certain}_{\mathcal{M}}(Q'', I) \subseteq Q^{\star}(I)$. Recall that Q'' is defined by the formula

$$\chi(\bar{x}) \;=\; \bigvee_{(J,\bar{a})\in \mathrm{tip}(Q')} \chi_{(J,\bar{a})}(\bar{x}).$$

Now, let $\bar{b} \in \mathrm{certain}_{\mathcal{M}}(Q'', I)$. We need to show that $\bar{b} \in Q^{\star}(I)$. Let $J^{\star}$ be the canonical universal solution for I under $\mathcal{M}$. Since $J^{\star} \in \mathrm{Sol}_{\mathcal{M}}(I)$ we have that $J^{\star} \models \chi(\bar{b})$ and then there exists a disjunct $\chi_{(J,\bar{a})}(\bar{x})$ in $\chi(\bar{x})$ such that $J^{\star} \models \chi_{(J,\bar{a})}(\bar{b})$. Notice that $\chi_{(J,\bar{a})}(\bar{x}) = \exists \bar{y}\,(\gamma_{(J,\bar{a})}(\bar{x},\bar{y}) \wedge \delta(\bar{y}) \wedge \mathbf{C}(\bar{y}))$. Thus, from $J^{\star} \models \chi_{(J,\bar{a})}(\bar{b})$, we conclude that there exists a tuple $\bar{c}$ of pairwise distinct elements from $\mathrm{dom}(J^{\star}) \cap \mathbf{C}$ such that $J^{\star} \models \gamma_{(J,\bar{a})}(\bar{b},\bar{c})$. Therefore, given that $\mathcal{M}$ is a mapping specified by a set of FO-TO-CQ st-dependencies, $\gamma_{(J,\bar{a})}(\bar{x},\bar{y})$ is a conjunctive query $J^{\star}$ is the canonical universal solution for I under $\mathcal{M}$, and $\bar{b}$ and $\bar{c}$ are tuples of constants values, we conclude that for every $K \in \mathrm{Sol}_{\mathcal{M}}(I)$ it holds that $K \models \gamma_{(J,\bar{a})}(\bar{b},\bar{c})$. Thus, from (A.7) we conclude that $I \models \gamma^{\star}_{(J,\bar{a})}(\bar{b},\bar{c})$. But this implies that $I \models \chi^{\star}_{(J,\bar{a})}(\bar{b})$ since $\bar{c}$ is a tuple of pairwise distinct elements and, thus, since $\chi^{\star}_{(J,\bar{a})}(\bar{x})$ is a disjunct in $\chi^{\star}(\bar{x})$ we have that $I \models \chi^{\star}(\bar{b})$. This last fact implies that $\bar{b} \in Q^{\star}(I)$ which was to be shown.

We prove now that $Q^{\star}(I) \subseteq \mathrm{certain}_{\mathcal{M}}(Q'', I)$. Assume that $\bar{b} \in Q^{\star}(I)$. We need to prove that $\bar{b} \in \mathrm{certain}_{\mathcal{M}}(Q'', I)$. Now, since $\bar{b} \in Q^{\star}(I)$ then there exists a disjunct $\chi^{\star}_{(J,\bar{a})}(\bar{x})$ in $\chi^{\star}(\bar{x})$ such that $I \models \chi^{\star}_{(J,\bar{a})}(\bar{b})$. By the definition of $\chi^{\star}_{(J,\bar{a})}(\bar{x})$ we know that there exists a tuple $\bar{c}$ of pairwise distinct elements from $\mathrm{dom}(I)$ such that $I \models \gamma^{\star}_{(J,\bar{a})}(\bar{b},\bar{c})$. Thus,

from (A.7) we conclude that for every $K \in \mathrm{Sol}_{\mathcal{M}}(I)$ it holds that $K \models \gamma_{(J,\bar{a})}(\bar{b}, \bar{c})$. But we know that $\bar{c}$ is a tuple of pairwise distinct elements from $\mathbf{C}$ and, thus, $K \models \chi_{(J,\bar{a})}(\bar{b})$ for every $K \in \mathrm{Sol}_{\mathcal{M}}(I)$. Finally, since $\chi_{(J,\bar{a})}(\bar{x})$ is a disjunct in $\chi(\bar{x})$ we conclude $K \models \chi(\bar{b})$ for every $K \in \mathrm{Sol}_{\mathcal{M}}(I)$, and thus, $\bar{b} \in \mathrm{certain}_{\mathcal{M}}(Q'', I)$. This was to be shown.

Finally, we have shown that if Q is target rewritable in $\mathcal{M}$, then there exists a query $Q^\star$ that is a monotone query over $\mathcal{C}_{\mathcal{M}}$ such that $Q(I) = Q^\star(I)$ for every instance I of $\mathbf{S}$. By Lemma A.2.3 we obtain that $\mathcal{C}_{\mathcal{M}} \Rightarrow Q$, completing the proof of the "only if" part of the lemma.

If. Assume now that $\mathcal{C}_{\mathcal{M}} \Rightarrow Q$. We need to show that Q is target rewritable in $\mathcal{M}$. That is, we need to show that there exists a query Q' over $\mathbf{T}$ such that $Q(I) = \mathrm{certain}_{\mathcal{M}}(Q', I)$ for every source instance I. Since $\mathcal{C}_{\mathcal{M}} \Rightarrow Q$ from Lemma A.2.3 we know that Q can we written as a monotone query over $\mathcal{C}_{\mathcal{M}}$. Assume that Q is an n-ary query and let $\bar{x} = (x_1, \ldots, x_n)$ be a tuple of variables. Thus, we know that Q is defined by formula $\varphi(\bar{x})$ that is a (possible infinitary) disjunction of formulas of the form

$$\gamma(\bar{x}) = \exists \bar{y} \big(\beta(\bar{x}', \bar{y}) \wedge \delta_{(\bar{x}', \bar{y})} \wedge \theta(\bar{x}', \bar{x}'') \big) \tag{A.8}$$

where:

- $\bar{x}'$ and $\bar{x}''$ are tuples of variables in $\bar{x}$ with no variables in common, $\bar{x} = (\bar{x}', \bar{x}'')$, and $\theta(\bar{x}', \bar{x}'')$ is a conjunction of equalities of the form $u = v$ with u a variable in $\bar{x}'$ and v a variable in $\bar{x}''$ and such that all the variables in $\bar{x}''$ are mentioned in at least one equality,
- $\delta_{(\bar{x}', \bar{y})}$ is a conjunction of inequalities $u \neq v$ for every pair of distinct variables u and v in $(\bar{x}', \bar{y})$, and
- $\beta(\bar{x}', \bar{y})$ is a nonempty conjunction of queries in $\mathcal{C}_{\mathcal{M}}$.

We first show that the conjunction $\beta(\bar{x}', \bar{y})$ in formula (A.8) can be rewritten into a target query. We construct a formula $\beta^\star(\bar{x}', \bar{y})$ over $\mathbf{T}$ as follows. First notice that $\beta(\bar{x}', \bar{y})$ is a conjunction of formulas $\alpha(\bar{u}) \in \mathcal{C}_{\mathcal{M}}$ with $\bar{u}$ a tuple of variables in $(\bar{x}', \bar{y})$. Moreover we

know that for every conjunct $\alpha(\bar{u})$ in $\beta(\bar{x}', \bar{y})$ there exists a dependency $\varphi(\bar{u}) \rightarrow \psi(\bar{u}) \in \Sigma$ such that $\alpha(\bar{u})$ is a rewriting of $\psi(\bar{u})$ over the source schema. Thus, to construct $\beta^\star(\bar{x}', \bar{y})$ we replace every conjunct $\alpha(\bar{u})$ of $\beta(\bar{x}', \bar{y})$ by $\psi(\bar{u})$. We show next that $\beta^\star(\bar{x}', \bar{y})$ is a target rewriting of $\beta(\bar{x}', \bar{y})$.

Let I be a source instance and assume that $f : (\bar{x}', \bar{y}) \rightarrow \mathbf{C}$ is an assignment of constants to the variables in $(\bar{x}', \bar{y})$ such that $I \models \beta(f(\bar{x}', \bar{y}))$. We show first that for every $J \in \mathrm{Sol}_{\mathcal{M}}(I)$ it holds that $J \models \beta^\star(f(\bar{x}', \bar{y}))$. Let $J \in \mathrm{Sol}_{\mathcal{M}}(I)$ and let $\psi(\bar{u})$ be the conjunctions in $\beta^\star(\bar{x}', \bar{y})$ obtained from formula $\alpha(\bar{u})$ in $\beta(\bar{x}', \bar{y})$. Since $I \models \beta(f(\bar{x}', \bar{y}))$ we know that $I \models \alpha(f(\bar{u}))$, and given that $\alpha(\bar{u})$ is a source rewriting of $\psi(\bar{u})$ we obtain that $J \models \psi(f(\bar{u}))$. This last fact holds for every conjunct in $\beta(f(\bar{x}', \bar{y}))$ and $\beta^\star(f(\bar{x}', \bar{y}))$ implying that $J \models \beta^\star(f(\bar{x}', \bar{y}))$ which was to be shown. The opposite direction is similar. Assume that for every $J \in \mathrm{Sol}_{\mathcal{M}}(I)$ it holds that $J \models \beta^\star(g(\bar{x}', \bar{y}))$ where $g : (\bar{x}', \bar{y}) \rightarrow \mathbf{C} \cup \mathbf{N}$ is an assignment of variables to constant and null values (notice that in this case we cannot directly assume that g assigns only constant values). Consider the conjunctions $\alpha(\bar{u})$ in $\beta(\bar{x}', \bar{y})$ and its associated conjunctions $\psi(\bar{u})$ in $\beta^\star(\bar{x}', \bar{y})$. Since for every $J \in \mathrm{Sol}_{\mathcal{M}}(I)$ we have that $J \models \beta^\star(g(\bar{x}', \bar{y}))$ then $J \models \psi(g(\bar{u}))$ for every $J \in \mathrm{Sol}_{\mathcal{M}}(I)$. Thus, given that $\alpha(\bar{u})$ is a source rewriting of $\psi(\bar{u})$ we have that $I \models \alpha(g(\bar{u}))$ obtaining that $I \models \beta(g(\bar{x}', \bar{y}))$. This completes the proof that $\beta^\star(\bar{x}', \bar{y})$ is a target rewriting of $\beta(\bar{x}', \bar{y})$.

To continue with the proof of the lemma, recall that the query Q is defined by a formula $\varphi(\bar{x})$ that is a (possibly infinitary) disjunction of formulas $\gamma(\bar{x})$ of the form (A.8). We show now that the query $\varphi^\star(\bar{x})$ obtained from $\varphi(\bar{x})$ replacing every disjunct $\gamma(\bar{x})$ by

$$\gamma^\star(\bar{x}) = \exists \bar{y} \left(\beta^\star(\bar{x}', \bar{y}) \wedge \mathbf{C}(\bar{x}', \bar{y}) \wedge \delta_{(\bar{x}', \bar{y})} \wedge \theta(\bar{x}', \bar{x}'') \right) \tag{A.9}$$

is a target rewriting of $\varphi(\bar{x})$. Let I be a source instance and assume $I \models \varphi(\bar{a})$ for some tuple $\bar{a}$ of constant values. Then $I \models \gamma(\bar{a})$ for some $\gamma(\bar{x})$ in the form (A.8) that is a disjunct of $\varphi(\bar{x})$. Let $\bar{a}'$ and $\bar{a}''$ be the tuples of elements from $\bar{a}$ corresponding to the positions of variables $\bar{x}'$ and $\bar{x}''$ in tuple $\bar{x}$. Then, we know that there exists a tuple $\bar{b}$ of constant values such that $I \models \beta(\bar{a}', \bar{b}) \wedge \delta_{(\bar{a}', \bar{b})} \wedge \theta(\bar{a}', \bar{a}'')$. Consider now an instance $J \in \mathrm{Sol}_{\mathcal{M}}(I)$.

Since $\beta^\star(\bar{x}', \bar{y})$ is a target rewriting of $\beta(\bar{x}', \bar{y})$ we obtain that $J \models \beta(\bar{a}', \bar{b})$ and since $\mathbf{C}(\bar{a}', \bar{b}) \wedge \delta_{(\bar{a}', \bar{b})} \wedge \theta(\bar{a}', \bar{a}'')$ holds, we have that $J \models \exists \bar{y} \big(\beta(\bar{a}', \bar{y}) \wedge \mathbf{C}(\bar{a}', \bar{b}) \wedge \delta_{(\bar{a}', \bar{y})} \wedge \theta(\bar{a}', \bar{a}'')\big)$, and thus $J \models \gamma^\star(\bar{a})$. We have shown that if $I \models \varphi(\bar{a})$ then for every $J \in \mathrm{Sol}_{\mathcal{M}}(I)$ it holds that $J \models \gamma^\star(\bar{a})$ for a disjunct $\gamma^\star(\bar{x})$ of $\varphi(\bar{x})$, and thus, for every $J \in \mathrm{Sol}_{\mathcal{M}}(I)$ it holds that $J \models \varphi^\star(\bar{a})$. The opposite direction is similar. Assume that $\bar{a}$ is a tuple of values such that for every $J \in \mathrm{Sol}_{\mathcal{M}}(I)$ we have that $J \models \varphi^\star(\bar{a})$. Let $J^\star$ be the canonical universal solution for I under $\mathcal{M}$. Then we know that $J^\star \models \varphi^\star(\bar{a})$ and then, there exists a disjunct $\gamma^\star(\bar{x})$ (of the form (A.8)) in $\varphi^\star(\bar{x})$ such that $J^\star \models \gamma^\star(\bar{a})$. Thus, there exists a tuple $\bar{b}$ such that $J^\star \models \beta^\star(\bar{a}', \bar{b}) \wedge \mathbf{C}(\bar{a}', \bar{b}) \wedge \delta_{(\bar{a}', \bar{b})} \wedge \theta(\bar{a}', \bar{a}'')$. Notice that $\beta^\star(\bar{x}', \bar{y})$ is a conjunctive query, and since $(\bar{a}', \bar{b})$ are constant values we know that there exists a homomorphism h (that is the identity on $\mathbf{C}$) such that every conjunct in $h(\beta^\star(\bar{a}', \bar{b}))$ is a fact in $J^\star$. Moreover, by the properties of universal solutions we know that for every $J \in \mathrm{Sol}_{\mathcal{M}}(I)$ there exists a homomorphism g from $J^\star$ to J. Therefore, the homomorphism $h \circ g$ is such that every conjunct in $(h \circ g)(\beta^\star(\bar{a}', \bar{b}))$ is a fact in J, and thus, $J \models \beta^\star(\bar{a}', \bar{b})$. We have shown that for every $J \in \mathrm{Sol}_{\mathcal{M}}(I)$ it holds that $J \models \beta^\star(\bar{a}', \bar{b})$, and thus, since $\beta^\star(\bar{x}', \bar{y})$ is a target rewriting of $\beta(\bar{x}', \bar{y})$ we have that $I \models \beta(\bar{a}', \bar{b})$. Now we have that $I \models \beta(\bar{a}', \bar{b})$ and we also know that $\mathbf{C}(\bar{a}', \bar{b}) \wedge \delta_{(\bar{a}', \bar{b})} \wedge \theta(\bar{a}', \bar{a}'')$ holds, then $I \models \exists \bar{y} \big(\beta(\bar{a}', \bar{y}) \wedge \mathbf{C}(\bar{a}', \bar{y}) \wedge \delta_{(\bar{a}', \bar{y})} \wedge \theta(\bar{a}', \bar{a}'')\big)$, and thus, $I \models \gamma(\bar{a})$ which implies that $I \models \varphi(\bar{a})$ which was to be shown. □

A.2.2. Proof of Lemma 6.2.4

Recall that for sets of queries (views) $\mathcal{C}_1$ and $\mathcal{C}_2$ we say that $\mathcal{C}_1$ strongly determines $\mathcal{C}_2$, and write $\mathcal{C}_1 \Rrightarrow \mathcal{C}_2$, when for every query $Q \in \mathcal{C}_2$ it holds that $\mathcal{C}_1 \Rrightarrow Q$. We prove now Lemma 6.2.4.

PROOF OF LEMMA 6.2.4. Let $\mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma)$ be a mapping with Σ a set of FO-TO-CQ st-dependencies. From Lemma 4.2.3 we know that the following property of the set $\mathcal{C}_{\mathcal{M}}$ holds: for every pair of instances I_1, I_2, if $\mathcal{C}_{\mathcal{M}}(I_1) \subseteq \mathcal{C}_{\mathcal{M}}(I_2)$ then $\mathrm{Sol}_{\mathcal{M}}(I_2) \subseteq \mathrm{Sol}_{\mathcal{M}}(I_1)$.

Consider now the mappings $\mathcal{M}_1$ and $\mathcal{M}_2$ in the statement of the lemma. We show next that $\mathcal{M}_1 \preceq_{\mathrm{S}} \mathcal{M}_2$ if and only if $\mathcal{C}_{\mathcal{M}_2} \Rrightarrow \mathcal{C}_{\mathcal{M}_1}$. Notice that $\mathcal{C}_{\mathcal{M}_2} \Rrightarrow \mathcal{C}_{\mathcal{M}_1}$ if and only if for

every pair of source instances I and K, if $\mathcal{C}_{\mathcal{M}_2}(I) \subseteq \mathcal{C}_{\mathcal{M}_2}(K)$ then $\mathcal{C}_{\mathcal{M}_1}(I) \subseteq \mathcal{C}_{\mathcal{M}_1}(K)$. Thus, we know that $\mathcal{C}_{\mathcal{M}_2} \Rightarrow \mathcal{C}_{\mathcal{M}_2}$ if and only if for every pair of source instances I and K, if $\mathrm{Sol}_{\mathcal{M}_2}(K) \subseteq \mathrm{Sol}_{\mathcal{M}_2}(I)$ then $\mathrm{Sol}_{\mathcal{M}_1}(K) \subseteq \mathrm{Sol}_{\mathcal{M}_1}(I)$. This last property is exactly the characterization of $\preceq_{\mathrm{s}}$ in Proposition 6.1.8. Thus, we have shown that $\mathcal{M}_1 \preceq_{\mathrm{s}} \mathcal{M}_2$ if and only if $\mathcal{C}_{\mathcal{M}_2} \Rightarrow \mathcal{C}_{\mathcal{M}_1}$. □

A.2.3. Proof of Lemma 6.2.6

The proof of this lemma is heavily based on the proof of Lemmas A.2.3 and 6.2.3.

PROOF OF LEMMA 6.2.6. We describe how the procedure TARGETREWRITING works. Let $\alpha(\bar{x})$ be a formula in $\mathrm{UCQ}^{=,\neq}$ that is target rewritable under $\mathcal{M}$. By combining Lemmas A.2.3 and 6.2.3 we know that $\alpha(\bar{x})$ can be written as a monotone query over $\mathcal{C}_{\mathcal{M}}$. Thus the procedure TARGETREWRITING first rewrites $\alpha(\bar{x})$ to obtain a logically equivalent formula that is a monotone query over $\mathcal{C}_{\mathcal{M}}$. To achieve this, we first write $\alpha(\bar{x})$ as a finite disjunction of formulas of the form

$$\exists \bar{y}\big(\gamma(\bar{x}', \bar{y}) \wedge \delta_{(\bar{x}', \bar{y})} \wedge \theta(\bar{x}', \bar{x}'')\big).$$

where $\bar{x} = (\bar{x}', \bar{x}'')$ where $\bar{x}'$ and $\bar{x}''$ have no variables in common, $\theta(\bar{x}', \bar{x}'')$ is a conjunction of equalities that respect the inequalities in $\alpha(\bar{x})$ and such that for every u in $\bar{x}''$ we have that $\theta(\bar{x}', \bar{x}'')$ contains an equality $u = v$ with v in $\bar{x}'$, $\gamma(\bar{x}', \bar{y})$ is a subset of the conjuncts of $\alpha(\bar{x})$ with variables in $\bar{x}'$ replaced according to the equalities in $\theta(\bar{x}', \bar{x}'')$, and $\delta_{(\bar{x}', \bar{y})}$ is a conjunction of inequalities for every pair of distinct variables in $(\bar{x}', \bar{y})$. Then we use the process described in the proof of Lemma A.2.3 to obtain the desired rewriting over $\mathcal{C}_{\mathcal{M}}$. Notice that in this case we can effectively obtain such a rewriting since the number of disjunct defining $\alpha(\bar{x})$ is finite. The process in the proof of Lemma A.2.3 essentially chase with $\mathcal{C}_{\mathcal{M}}$ every instance $I_{\gamma(\bar{x}', \bar{y})}$ for every disjunct $\gamma(\bar{x}', \bar{y})$ defining $\alpha(\bar{x})$. If after chasing $I_{\gamma(\bar{x}', \bar{y})}$ the instance obtained is $I_{\gamma^\star(\bar{x}', \bar{y}')}$, then we include $\exists \bar{y}'\big(\gamma^\star(\bar{x}', \bar{y}') \wedge \delta_{(\bar{x}', \bar{y}')} \wedge \theta(\bar{x}', \bar{x}'')\big)$ as a disjunct in the rewriting.

Notice that after the above described process we have the formula $\alpha(\bar{x})$ written as a finite disjunction of formulas of the form (A.8) as described in the "if" part of the proof of Lemma 6.2.3. Thus, we can effectively compute a target rewriting of $\alpha(\bar{x})$ by using the process described in the proof of that lemma. Moreover, by inspecting the process in the proof of Lemma 6.2.3 it is not difficult to see that the formula $\beta(\bar{x})$ generated as a target rewriting of $\alpha(\bar{x})$ is in $\mathrm{UCQ}^{=,\neq,\mathbf{C}}$ such that if an inequality $x \neq y$ occurs in $\beta(\bar{x})$ then the atoms $\mathbf{C}(x)$ and $\mathbf{C}(y)$ also occurs in $\beta(\bar{x})$. □