Online Appendix to The Recovery of a Schema
Mapping: Bringing Exchanged Data Back

MARCELO ARENAS and JORGE PÉREZ
Pontificia Universidad Católica de Chile

and

CRISTIAN RIVEROS
R&M Tech Ingeniería y Servicios Limitada

A. PROOFS OF SECTION 7

A.1 Proof of Lemma 7.2

To prove the lemma, we provide an algorithm that given an st-mapping $M = (S, T, \Sigma)$ such that $\Sigma$ is a set of FO-TO-CQ dependencies, and a conjunctive query $Q$ over schema $T$, computes a query $Q'$ that is a rewriting of $Q$ over the source schema $S$.

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 $\beta$, we are allowed to existentially quantify some of the positions of the free variables of $\beta$. For example, if $\beta(x_1, x_2, x_3) = P(x_1, x_2) \land R(x_2, x_3)$, then two existential replacements of $\beta(x_1, x_2, x_3)$ are $\gamma_1(x_2) = \exists u \exists v (P(u, x_2) \land R(x_2, v))$ and $\gamma_2(x_1, x_2, x_3) = \exists z (P(x_1, z) \land R(x_2, x_3))$. We note that both $\gamma_1$ and $\gamma_2$ are implied by $\beta$. In an existential replacement, we are also allowed to use the same quantifier for different positions. For example, $\gamma_3(x_2) = \exists w (P(w, x_2) \land R(x_2, w))$ is also an existential replacement of $\beta$. We note that $\gamma_3$ is implied by $\beta$ if $x_1$ and $x_3$ have the same value, that is, $\beta(x_1, x_2, x_3) \land x_1 = x_3$ implies $\gamma_3$. 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 $(\exists \bar{\gamma}(\bar{x}', \bar{z}), \theta(\bar{x}''))$, where: (1) $\exists \bar{\gamma}(\bar{x}', \bar{z})$ 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 $\theta$ ($1 \leq i, j \leq k$ 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 $\exists \bar{\gamma}(\bar{x}', \bar{z})$ and $\theta(\bar{x}'')$, respectively. Notice that $\exists \bar{\gamma}(\bar{x}', \bar{z})$ is a logical consequence

M. Arenas and C. Riveros were supported by Fondecyt grants 1070732 and 1090565; J. Pérez was supported by Conicyt Ph.D. Scholarship.

Authors’ addresses: M. Arenas, J. Pérez, Department of Computer Science, Pontificia Universidad Católica de Chile, Santiago, Chile; email: {marenas,jperez}@ing.puc.cl; C. Riveros, R&M Tech Ingeniería y Servicios Limitada, Santiago, Chile; email: criveros@rmt.cl. Permission to make digital/hard copy of all or part of this material without fee for personal or classroom use provided that the copies are not made or distributed for profit or commercial advantage, the ACM copyright/server notice, the title of the publication, and its date appear, and notice is given that copying is by permission of the ACM, Inc. To copy otherwise, to republish, to post on servers, or to redistribute to lists requires prior specific permission and/or a fee. © 20YY ACM 0362-5915/20YY/0300-0001 $5.00
of $\beta(\bar{x}) \land \theta(\bar{x}''')$. For example, the following are existential replacements of the formula $\beta(x_1, x_2, x_3) = \exists y_1 (R(x_1, x_2, y_1) \land T(y_1, x_3, x_2))$:

$$
\begin{align*}
&\{ \exists y_1 (R(x_1, x_2, y_1) \land T(y_1, x_3, x_2)), \text{true}, \} \\
&\{ \exists z_1 \exists z_2 \exists y_1 (R(z_1, x_2, y_1) \land T(y_1, x_3, z_2)), \text{true}, \} \\
&\{ \exists z_1 \exists z_2 \exists y_1 (R(z_1, z_2, y_1) \land T(y_1, x_3, z_2)), x_1 = x_2 \land x_3 = x_2 \}.
\end{align*}
$$

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 $\bar{y}$. $\bar{f} : \bar{x} \rightarrow \bar{y}$ is a substitution that replaces every variable of $\bar{x}$ by a variable of $\bar{y}$ ($\bar{f}$ is not necessarily a one-to-one function). $\bar{f}(\bar{x})$ is a tuple of variables obtained by replacing every variable $x$ in $\bar{x}$ by $\bar{f}(x)$, 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 $x_1 = y_1 \land \cdots \land x_k = y_k$.

**Algorithm QUERYREWIRING($M, Q$)**

**Input**: An st-mapping $M = (S, T, \Sigma)$ where $\Sigma$ is a set of FO-TO-CQ dependencies, and a conjunctive query $Q$ over $T$.

**Output**: An FO query $Q'$ that is a rewriting of $Q$ over the source schema $S$.

1. Assume that $Q$ is given by the formula $\exists \bar{y} \psi(\bar{x}, \bar{y})$.
2. Create a set $C_0$ of FO queries as follows. Start with $C_0 = \emptyset$ 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 $\sigma_1, \ldots, \sigma_p$ in such a way that the sets of variables of the formulas $\xi_1, \ldots, \xi_p$ are pairwise disjoint.
   b. Assume that $\xi_i$ is equal to $\varphi_i(\bar{u}_i) \rightarrow \exists \bar{v}_i \psi_i(\bar{u}_i, \bar{v}_i)$, 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 $\exists z \chi(\bar{w}, \bar{z})$ be the formula $\exists z \chi_1(\bar{w}_1, \bar{z}_1) \land \cdots \land \chi_p(\bar{w}_p, \bar{z}_p)$ 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 $(\exists z \exists \bar{s} \gamma(\bar{w}', \bar{z}, \bar{s}), \theta(\bar{w}''))$ of $\exists z \chi(\bar{w}, \bar{z})$ (up to renaming of variables in $\bar{s}$), and for every pair of variable substitutions $\bar{f} : \bar{x} \rightarrow \bar{z}$ and $\bar{g} : \bar{w}' \rightarrow \bar{z}$, check whether there exists a variable substitution $h : \bar{y} \rightarrow (\bar{z}, \bar{s})$ 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 $C_p$ the following formula:

$$
\exists \bar{u}_1 \ldots \exists \bar{u}_p \left( \bigwedge_{i=1}^p \varphi_i(\bar{u}_i) \land \theta(\bar{w}'') \land \bar{x} = f(\bar{x}) \land \bar{w}' = g(\bar{w}') \right).
$$
(3) If $C_{\varphi}$ is nonempty, then let $\alpha(\bar{x})$ be the FO formula constructed as the disjunction of all the formulas in $C_{\varphi}$. 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 $\alpha(\bar{x})$.

\[ \square \]

Notice that in the algorithm, tuple $\bar{x}$ is the set of free variables of formula (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 $\bar{s}$, we have that every variable $x$ in $\bar{x}$ is equal to some variable $u$ in $\{\bar{u}_1, \ldots, \bar{u}_p\}$ since $\bar{x} = f(\bar{x}) \land \bar{w}' = g(\bar{w}')$ is a subformula of (2). This implies that formula (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.** Assume that $\Sigma$ is given by dependency $\sigma$:

\[
\varphi(x_1, x_2) \rightarrow R(x_1, x_1, x_2),
\]

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 $\psi(x_1, x_2, x_3, y_1) = R(x_1, x_2, y_1) \land R(y_1, x_3, x_3)$. 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 $(\xi_1, \xi_2)$ from $(\sigma_1, \sigma_2)$ by renaming the variables of $\sigma_1$ and $\sigma_2$ (in such a way that the sets of variables of $\xi_1$ and $\xi_2$ are disjoint). Assume that $\xi_1$ is equal to $\varphi(u_1, u_2) \rightarrow R(u_1, u_1, u_2)$ and $\xi_2$ 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 $\xi_1$ and $\xi_2$, 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 $\chi(u_1, u_2, u_3, u_4) = R(u_1, u_1, u_2) \land R(u_3, u_3, u_4)$, 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, $(\exists s_1 (R(u_1, u_1, s_1) \land R(s_1, u_3, u_4)), u_2 = u_3)$ is one of these existential replacements: $R(g(u_1), g(u_1), s_1) \land R(s_1, g(u_3), g(u_4))$ 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 $x_1 = x_1$):

\[
\exists u_2 \exists u_3 \exists u_4 (\varphi(u_1, u_2) \land \varphi(u_3, u_4) \land u_2 = u_3 \land x_1 = x_1 \land u_1 = x_1 \land u_4 = x_3 \land u_4 = x_3).
\]

Formula $\beta(x_1, x_2, x_3)$ is added to $C_{\varphi}$. 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 (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 $C_{\varphi}$. 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 \(M = (S, T, \Sigma)\) be an st-mapping with \(\Sigma\) a set of FO-TO-CQ dependencies, \(Q\) a conjunctive query over \(T\), and \(Q'\) the output of QUERYREWIRING\((M, Q)\). It is straightforward to prove that the algorithm runs in exponential time in the size of \(M\) and \(Q\), and that the size of \(Q'\) is exponential in the size of \(M\) and \(Q\). We now prove the correctness of the rewriting algorithm. We need to show that for every instance \(I\) of \(S\), it holds that:

\[Q'(I) = \text{certain}_M(Q, I)\]

In this proof, we assume that \(Q\) is given by the formula \(\exists y \psi(x, y)\), and that \(Q'\) is given by the formula \(\alpha(x)\) (that could be false).

We first show that \(Q'(I) \subseteq \text{certain}_M(Q, I)\). The proof relies in the following claim.

**Claim A.2.** The formula \(\forall x (\alpha(x) \rightarrow \exists y \psi(x, y))\) is a logical consequence of \(\Sigma\).

**Proof.** If \(\alpha(x)\) is false, the property trivially holds. Now, assume that \(\alpha(x)\) is the disjunction of the formulas in the set \(C_\psi\) constructed after step 2 of the algorithm. We show that for every \(\beta(x) \in C_\psi\) it holds that \(\forall x (\beta(x) \rightarrow \exists y \psi(x, y))\) is a logical consequence of \(\Sigma\), which implies that \(\forall x (\alpha(x) \rightarrow \exists y \psi(x, y))\) is a logical consequence of \(\Sigma\). Assume that \(\beta(x)\) is equal to:

\[\exists \bar{u}_1 \cdots \exists \bar{u}_p \left( \bigwedge_{i=1}^p \varphi_i(\bar{u}_i) \land \theta(\bar{w}') \land \bar{x} = f(\bar{x}) \land \bar{w}' = g(\bar{w}') \right),\]

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 \(\Sigma\). In step 2(c) of the algorithm, formula \(\exists \bar{z} \chi(\bar{w}, \bar{z})\) is defined as \(\exists \bar{z}_1 \cdots \exists \bar{z}_p \chi_1(\bar{w}_1, \bar{z}_1) \land \cdots \land \chi_p(\bar{w}_p, \bar{z}_p)\), 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 \(\Phi\):

\[\forall \bar{x} (\beta(x) \rightarrow \exists \bar{w} (\exists \bar{z} \chi(\bar{w}, \bar{z}) \land \theta(\bar{w}'') \land \bar{x} = f(\bar{x}) \land \bar{w}' = g(\bar{w}'')))\]

is a logical consequence of \(\Sigma\). Given that \(\exists \bar{z} \chi(\bar{w}, \bar{z}) \land \theta(\bar{w}'')\) is an existential replacement of \(\exists \bar{z} \chi(\bar{w}, \bar{z})\), we know that \(\exists \bar{z} \chi(\bar{w}, \bar{z}) \land \theta(\bar{w}'')\) implies \(\exists \bar{z} \exists \gamma(\bar{w}', \bar{z}, \bar{s})\). Thus, we have that \(\Phi\) implies:

\[\forall \bar{x} (\beta(x) \rightarrow \exists \bar{w}' (\exists \bar{z} \exists \gamma(\bar{w}', \bar{z}, \bar{s}) \land \bar{x} = f(\bar{x}) \land \bar{w}' = g(\bar{w}')))\]

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:

\[\forall \bar{x} (\beta(x) \rightarrow \exists \bar{z} \exists \gamma(\bar{w}', \bar{z}, \bar{s}) \land \bar{x} = f(\bar{x}))\]

is a logical consequence of \(\Phi\). 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 \(\forall \bar{x} (\beta(x) \rightarrow \exists \bar{z} \exists \gamma(\bar{w}', \bar{z}, \bar{s}) \land \bar{x} = f(\bar{x}))\) is also a consequence of \(\Phi\). In this last formula, we can replace \(f(\bar{x})\) by \(\bar{x}\) and drop the conjunction \(\bar{x} = f(\bar{x})\), obtaining \(\forall \bar{x} (\beta(x) \rightarrow \exists \bar{z} \exists \gamma(\bar{w}', \bar{z}, \bar{s}) \land \bar{x} = f(\bar{x}))\). Since \(h\) is a function from \(\bar{y}\) to \((\bar{z}, \bar{s})\), we have that \(\exists \bar{z} \exists \gamma(\bar{w}', h(\bar{y}))\) logically implies formula \(\exists \bar{y} \psi(\bar{x}, \bar{y})\) (because the variables in \(\bar{y}\) are all distinct). We have shown that \(\forall \bar{x} (\beta(x) \rightarrow \exists \bar{y} \psi(\bar{x}, \bar{y}))\) is a logical consequence of \(\Phi\) and, therefore, it is a logical consequence of \(\Sigma\). This concludes the proof of the claim. □
We prove now that \( Q'(I) \subseteq \text{certain}_{A}(Q, I) \) for every instance \( I \in \text{Inst}(S) \), 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 \( J \in \text{Sol}_{A}(I) \) it holds that \( \bar{a} \in Q(J) \). Since \( \bar{a} \in Q'(I) \) we know that \( I \models \alpha(\bar{a}) \). Now let \( J \in \text{Sol}_{A}(I) \). From the Claim A.2 we know that \( \forall \bar{x}(\alpha(\bar{x}) \rightarrow \exists \bar{y} \psi(\bar{x}, \bar{y})) \) is a logical consequence of \( \Sigma \). 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 \( J \in \text{Sol}_{A}(I) \) it holds that \( \bar{a} \in Q(J) \). This was to be shown.

We now prove that \( \text{certain}_{A}(Q, I) \subseteq Q'(I) \) for every instance \( I \). We first recall the notion of \( \text{chase} \) (introduced in the proof of Theorem 7.3). Let \( I \) be an instance of \( S \). Then \( \text{chase}_{\Sigma}(I) \) is an instance of \( T \) constructed with the following procedure. For every dependency \( \sigma \in \Sigma \) of the form \( \varphi(\bar{x}) \rightarrow \exists \bar{y} \nu(\bar{x}, \bar{y}) \), with \( \bar{x} = (x_1, \ldots, x_m) \), \( \bar{y} = (y_1, \ldots, y_l) \) tuples of distinct variables, and for every \( m \)-tuple \( \bar{a} \) of elements from \( \text{dom}(I) \) such that \( I \models \varphi(\bar{a}) \), do the following. Choose an \( \ell \)-tuple \( \bar{n} \) of distinct fresh values from \( N \), and include all the conjuncts of \( \nu(\bar{a}, \bar{n}) \) in \( \text{chase}_{\Sigma}(I) \).

We say that the conjuncts of \( \psi(a_1, \ldots, a_m, \bar{n}) \) included in \( \text{chase}_{\Sigma}(I) \) are generated (or justified) by \( \sigma \).

We also make use of the notion of \( N \)-connected instances introduced in the Section 9 when proving Proposition 9.1. Recall that an instance \( I \) of \( S \) is \( N \)-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 \( t \in I \) \( R \) for \( R \in S \), and there is an edge in \( E_I \) between tuples \( t_1 \) and \( t_2 \) if there exists a value \( n \in N \) that is mentioned both in \( t_1 \) and \( t_2 \). Then \( I \) is \( N \)-connected if the graph \( G_I \) is connected.

An instance \( I_1 \) is an \( N \)-connected sub-instance of \( I \), if \( I_1 \) is a sub-instance of \( I \) and \( I_1 \) is \( N \)-connected. Finally, \( I_1 \) is an \( N \)-connected component of \( I \), if \( I_1 \) is a \( N \)-connected sub-instance of \( I \) and there is no \( N \)-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 \( C \cup N \). We say that \( \varphi(\bar{a}) \) is \( N \)-connected, if the instance that contains exactly the atoms of \( \varphi(\bar{a}) \) is \( N \)-connected. The definition of \( N \)-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 \( \text{dom}(I) \subseteq C \), then every atom in an \( N \)-connected sub-instance of \( \text{chase}_{\Sigma}(I) \) is generated by a single dependency in \( \Sigma \).

We are ready now to prove that \( \text{certain}_{A}(Q, I) \subseteq Q'(I) \) for every instance \( I \) in \( S \). Let \( I \) be an arbitrary instance of \( S \). We use the following property of \( \text{chase}_{\Sigma}(I) \).

Since \( Q \) is a conjunctive query, from [Fagin et al. 2005a] (Proposition 4.2) we know that \( \text{certain}_{A}(Q, I) = Q(\text{chase}_{\Sigma}(I)) \), where \( Q(\text{chase}_{\Sigma}(I)) \) denotes the set of tuples in \( Q \) that are \( \text{chase}_{\Sigma}(I) \) composed only by constant values. Thus, in order to prove that \( \text{certain}_{A}(Q, I) \subseteq Q'(I) \), it is enough to prove that \( Q(\text{chase}_{\Sigma}(I)) \subseteq Q'(I) \). 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 \( \bar{a} \in Q(I) \). 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 \( (y_1, \ldots, y_t) \). Since \( \text{chase}_{\Sigma}(I) \models \exists \bar{y} \psi(\bar{a}, \bar{y}) \), we know that there exists a tuple \( b = (b_1, \ldots, b_t) \) composed by constant and null values, such that \( \text{chase}_{\Sigma}(I) \models \psi(\bar{a}, \bar{b}) \). Let \( \rho_1(a_1, b_1), \ldots, \rho_p(a_p, b_p) \) be the \( N \)-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 \( \psi(x, y) \) has \( m \) atoms, then \( n = \sum k_i = m \).

Without loss of generality, we can assume that \( \rho_i(a, b) = p_1(a, b) \wedge \cdots \wedge p_p(a, b) \) (otherwise we can always reorder the atoms in \( \psi(a, b) \)). Since \( \text{chase}_\Sigma(I) \models (a, b) \), we know that for every \( i \in \{1, \ldots, p\} \), the conjuncts of \( \rho_i(a, b) \) are included in the same \( \Sigma \)-connected sub-instance of \( \text{chase}_\Sigma(I) \). Furthermore, as we have noted before, for every set of facts \( J \) that forms an \( \Sigma \)-connected sub-instance of \( \text{chase}_\Sigma(I) \), there exists a sentence in \( \Sigma \) that justifies \( J \). Then there exist \( p \) (not necessarily distinct) sentences \( (\sigma_1, \ldots, \sigma_p) \in \Sigma^p \), such that the atoms in \( \rho_i(a, b) \) are generated by \( \sigma_i \). 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 \( \xi_i \) is of the form \( \bar{x} = \exists \psi_i(\bar{u}_i, \bar{v}_i) \). Since \( \sigma_i \) generates all the atoms in \( \rho_i(a, b) \), we know that for every \( i \in \{1, \ldots, p\} \), there exists a formula \( \chi_i(\bar{u}_i, \bar{v}_i) \) and tuples \( \bar{c}_i \) and \( \bar{a}_i \) of values in \( C \) and \( N \), respectively, such that \( \chi_i(\bar{u}_i, \bar{v}_i) \) is a conjunction of \( k_i \) (not necessarily distinct) atoms from \( \psi_i(\bar{u}_i, \bar{v}_i) \) with \( \bar{u}_i \subseteq \bar{a}_i \) and \( \bar{v}_i \subseteq \bar{c}_i \), and such that \( \chi_i(\bar{c}_i, \bar{a}_i) \) 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 \( \chi_i(\bar{c}_i, \bar{a}_i) = \rho_i(\bar{a}_i, \bar{b}_i) \). Let \( \chi(\bar{w}, \bar{z}) = \chi_1(\bar{w}, \bar{z}_1) \wedge \cdots \wedge \chi_p(\bar{w}, \bar{z}_p) \), with \( \bar{w} = (\bar{w}_1, \ldots, \bar{w}_d) = (\bar{u}_1, \ldots, \bar{u}_d) \) and \( \bar{z} = (\bar{z}_1, \ldots, \bar{z}_p) = (\bar{v}_1, \ldots, \bar{v}_p) \) tuples of distinct variables. Then we have that \( \chi(\bar{c}, \bar{a}) = \psi(\bar{a}, \bar{b}) \), where \( \bar{c} = (\bar{c}_1, \ldots, \bar{c}_d) \) is a tuple of values in \( C \), and \( \bar{a} = (\bar{a}_1, \ldots, \bar{a}_p) \) is a tuple of values in \( N \). Given that the conjuncts of \( \rho_i(\bar{a}_i, \bar{b}_i) \) are facts in \( \text{chase}_\Sigma(I) \), and each \( \rho_i(\bar{a}_i, \bar{b}_i) = \chi_i(\bar{c}_i, \bar{a}_i) \) is an \( \Sigma \)-connected component of \( \psi(\bar{a}, \bar{b}) \), we have that \( \bar{a} \) is a tuple of distinct values in \( N \) (since tuples \( \bar{a}_i \) and \( \bar{b}_i \) 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 \( \bar{a} = (a_1, \ldots, a_p) \), that is, for every \( i \in \{1, \ldots, d\} \), \( c_i \) is the value assigned to variable \( w_i \), and for every \( i \in \{1, \ldots, p\} \), \( a_i \) is the value assigned to variable \( z_i \).

Focus now in the positions of \( \psi(x, y) \). For every \( i \in \{1, \ldots, r\} \), we call \( x_i \)-position to a position in \( \psi(x, y) \) where variable \( x_i \) occurs. Similarly, for every \( i \in \{1, \ldots, e\} \), a \( y_i \)-position is a position in \( \psi(x, y) \) where variable \( y_i \) occurs. Since \( \psi(\bar{a}, \bar{b}) \) and \( \chi(\bar{c}, \bar{a}) \) are syntactically equal, there is a one-to-one correspondence between the positions in \( \psi(x, y) \) and the positions in \( \chi(\bar{w}, \bar{z}) \). Then we can talk about \( x \)- or \( y \)-positions in general when referring to positions in \( \psi(x, y) \) or in \( \chi(\bar{w}, \bar{z}) \). We use this correspondence of positions and the fact that \( \psi(\bar{a}, \bar{b}) = \chi(\bar{c}, \bar{a}) \), to create an existential replacement, and functions \( f, g, h \), as in step 2(c)(iii) of the algorithm.

We know that \( \bar{a} \) is a tuple of constant values. Then from \( \psi(\bar{a}, \bar{b}) = \chi(\bar{c}, \bar{a}) \), we obtain that every element of \( \bar{a} \) 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 \( \psi(\bar{a}, \bar{b}) = \chi(\bar{c}, \bar{a}) \). 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 \( \exists \chi(\bar{w}, \bar{z}) \), where, for every such variable \( y_i \), all the \( y_i \)-positions are replaced by an existentially quantified variable \( s_i \). Let \( (\exists \chi(\bar{w'}, \bar{z}, \bar{s}), \theta(\bar{w'})) \) be such a replacement of \( \exists \chi(\bar{w}, \bar{z}) \). 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 \( \bar{a} \) is a tuple of distinct values of \( N \), \( \bar{c} \) is a tuple of values of \( C \), and \( \chi(\bar{c}, \bar{a}) = \psi(\bar{a}, \bar{b}) \)). 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
If $\bar{w}$ is 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 $V_{\bar{a}_i}$, and similarly $g(w_i) = x_j$ if $x_j$ is the representative of $V_{\bar{a}_i}$. 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 $\gamma(\bar{w}', \bar{z}, \bar{s})$ are syntactically equal (they coincide in every $x_i$- and $y_i$-position). Then we know that the formula:

$$\beta(\bar{x}) = \exists \bar{u}_1 \ldots \exists \bar{u}_p \left( \bigwedge_{i=1}^p \Phi_i(\bar{u}_i) \land \theta(\bar{w}'') \land \bar{x} = f(\bar{x}) \land \bar{w}' = g(\bar{w}') \right),$$

is added to $C_\psi$ after step 2 of the algorithm. We claim that $I \models \beta(\bar{a})$.

Next we show that $I \models \varphi_1(\bar{e}') \land \ldots \land \varphi_p(\bar{e}_p') \land \theta(\bar{e}'') \land \bar{a} = f(\bar{a}) \land \bar{e}' = g(\bar{e}')$, where $\bar{e}_i'$ is a tuple of elements in $C$ that contains $\bar{e}_i$, $\bar{e}'$ is the tuple obtained by restricting $\bar{e}$ to the variables of $\bar{w}'$, and $\bar{e}_i''$ is the tuple obtained by restricting $\bar{e}$ 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 $\theta(\bar{e}'')$ 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 $V_{\bar{a}_i}$. Thus, we have that $a_i = a_j$, which implies that $\bar{a} = f(\bar{a})$ holds. Next consider equality $\bar{e}' = g(\bar{e}')$. We know by the definition of $g$ that $g(w_i) = x_j$, if $x_j$ is the representative of $V_{\bar{a}_i}$. Thus, we have that $c_i = a_j$, which implies that $\bar{e}' = g(\bar{e}')$ holds. Finally, given that for every $i \in \{1, \ldots, p\}$, formula $\psi(\bar{a}_i, \bar{b}_i) = \chi(\bar{c}_i, \bar{n}_i)$ 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 $\bar{c}_i$ that contains the elements in $\bar{c}_i$ and such that $I \models \varphi_i(\bar{c}_i)$. We have shown that $I \models \varphi_1(\bar{c}_1') \land \ldots \land \varphi_p(\bar{c}_p') \land \theta(\bar{e}'') \land \bar{a} = f(\bar{a}) \land \bar{e}' = g(\bar{e}')$, and, hence, $I \models \beta(\bar{a})$.

We have shown that if $\text{chase}_2(I) \models \exists \bar{v} \psi(\bar{a}, \bar{y})$ for a tuple $\bar{a}$ of constants, then there exists a formula $\beta(\bar{x}) \in C_\psi$ such that $I \models \beta(\bar{a})$. Thus, since $\alpha(\bar{x})$ is the disjunction of the formulas in $C_\psi$, we have that $I \models \alpha(\bar{a})$. Recall that $\exists \bar{v} \psi(\bar{x}, \bar{y})$ defined query $Q$ and $\alpha(\bar{x})$ defines query $Q'$. Therefore, if a tuple $\bar{a}$ of constants is such that $\bar{a} \in Q(\text{chase}_2(I))$, then we have that $\bar{a} \in Q'(I)$, which implies that $Q(\text{chase}_2(I)) \subseteq Q'(I)$ and then $\text{certain}_M(Q, I) \subseteq Q'(I)$ which is the property that we wanted to obtain. This completes the proof of correctness of the algorithm.

A.2 Proof of Lemma 7.6

The following algorithm computes a rewriting of a conjunctive query given by a single atom without existential quantifiers.

**Algorithm** **QUERYREWITINGATOM**(M, Q)

**Input**: An st-mapping $\mathcal{M} = (\Sigma, \mathcal{T}, \Sigma)$ where $\Sigma$ is a set of FO-TO-CQ dependencies, and a conjunctive query $Q$ given by a single atom over $\mathcal{T}$ without existential quantifiers.

**Output**: An FO query $Q'$ that is a rewriting of $Q$ over the source schema $\Sigma$.

1. Construct a set $\Sigma'$ 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 \( \Sigma' \), 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 \( \Sigma' \) 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 \( \Sigma' \).

(4) Create a set \( C_R \) of FO queries as follows. Start with \( C_R = \emptyset \). Then for every dependency \( \varphi(\bar{z}) \rightarrow R(\bar{z}) \) in \( \Sigma' \), add formula \( \exists \bar{z}(\varphi(\bar{z}) \land \bar{z} = \bar{x}) \) to \( C_R \).

(5) If \( C_R \) is nonempty, then let \( \alpha(\bar{x}) \) be the FO formula constructed as the disjunction of all the formulas in \( C_R \). Otherwise, let \( \alpha(\bar{x}) = \text{false} \), 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 \( \Sigma \) 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 \( \Sigma' \) constructed in the step 1 of the algorithm is of size linear in the size of \( \Sigma \). The proof of correctness follows directly from the correctness of algorithm QUERYWRITING of Lemma 7.2. Just observe that if the input of the algorithm QUERYWRITING 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.

B. PROOFS OF SECTION 8

B.1 Proof of Proposition 8.1

Let \( S = \{P(\cdot), R(\cdot)\} \), \( T = \{T(\cdot)\} \) and \( \Sigma \) be the following set of st-tgds:

\[
P(x) \rightarrow \exists y T(y),
R(x) \rightarrow T(x).
\]

Assume that \( \mathcal{M}' \) is a recovery of \( \mathcal{M} \) that is specified by a set of FO-sentences over \( S \cup T \). 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 \( S \) such that \( P^I = \{a\} \) and \( R^I = \emptyset \), where \( a \) is an arbitrary element of \( C \). Since \( \mathcal{M}' \) is a recovery of \( \mathcal{M} \), there exists an instance \( J \) of \( T \) such that \( (I, J) \in \mathcal{M} \) and \( (J, I) \in \mathcal{M}' \).

We consider two cases.

—First, assume that \( J \) mentions an element \( b \in C \), that is not necessarily distinct from \( a \). Then we have that \( (I', J) \in \mathcal{M} \), where \( I' \) is an instance of \( S \) such that \( P^{I'} = \emptyset \) and \( R^{I'} = \{b\} \). Thus, given that \( (I, J) \in \mathcal{M}' \), we have that \( (I', I) \in \mathcal{M} \cap \mathcal{M}' \), which implies that \( \emptyset \not\subseteq \text{Sol}_M(I) \subseteq \text{Sol}_M(I') \) by Proposition 3.8. Let \( J' \) be an instance of \( T \) defined as \( T^{J'} = \{n\} \), where \( n \) is an arbitrary element of \( N \). We have that \( (I, J') \in \mathcal{M} \) and \( (J', I) \not\in \mathcal{M} \), which contradicts the fact that \( \text{Sol}_M(I) \subseteq \text{Sol}_M(I') \).

—Second, assume that \( J \) does not mention any element from \( C \). Assume that \( \text{dom}(J) = \{n_1, \ldots, n_k\} \), and let \( f \) be a function defined as \( f(n_i) = b_i \), where each \( b_i \) is an element
of C that is distinct from a and \(b_i \neq b_j\) for \(i \neq j\). Let \(J^*\) be the target instance that results from replacing every value \(n_i\) by \(b_i\). It is easy to see that \((I, J^*) \in \mathcal{M}\). 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 \((J^*, I)\) when we consider these instances as structures over \(S \cup T\). Thus, given that \(\mathcal{M}'\) is specified by a set of FO-sentences over \(S \cup T\), we conclude that \((J^*, I) \in \mathcal{M}'\). Therefore, there exists an instance \(J^*\) of \(\mathcal{M}\) such that \((I, J^*) \in \mathcal{M}, (J^*, I) \in \mathcal{M}'\) and \(J^*\) mentions elements of C. This leads to a contradiction, as we show in the previous case. This concludes the proof of the proposition.

### B.2 Proof of Proposition 8.2

Let \(S = \{\mathcal{S}(\cdot, \cdot)\}, T = \{\mathcal{T}(\cdot)\}\) and \(\mathcal{M} = (S, T, \Sigma)\) be an st-mapping specified by the following set \(\Sigma\) of CQ\(\exists\)-to-CQ dependencies:

\[
S(x, y) \land x \neq y \rightarrow T(x).
\]

Next we show that \(\mathcal{M}\) has no maximum recovery specified by a set of FO\(\exists\)-to-UCQ dependencies.

For the sake of contradiction, assume that \(\mathcal{M}^* = (T, S, \Sigma^*)\) is a maximum recovery of \(\mathcal{M}\), where \(\Sigma^*\) is a set of FO\(\exists\)-to-UCQ dependencies. Let \(I_1\) be a source instance such that \(S_{I_1} = \{(a, b)\}\), where \(a, b\) are arbitrary elements of \(C\) and \(a \neq b\). Given that \(\mathcal{M}^*\) is a recovery of \(\mathcal{M}\), there exists a target instance \(J_1\) such that \((I_1, J_1) \in \mathcal{M}\) and \((J_1, I_1) \in \mathcal{M}^*\).

Since \(\mathcal{M}^*\) is a maximum recovery of \(\mathcal{M}\), there exists at least one dependency \(\varphi(\bar{x}) \rightarrow \psi(\bar{x}) \in \Sigma^*\) such that \(J_1 \models \varphi(\bar{c})\), where \(\bar{c}\) is a nonempty tuple of elements from \(C\). On the contrary, assume that this is not the case. Then given that \((J_1, I_1) \models \Sigma^*\) and \(\Sigma^*\) is a set of FO\(\exists\)-to-UCQ dependencies, we conclude that either \((J_1, I_0) \models \Sigma^*\), where \(I_0\) is the empty source instance, or \((J_1, I_2) \models \Sigma^*\), where \(I_2\) is a source instance such that \(S^{I_2} = \{(c, c)\}\) \(c \in C, c \neq a, c \neq b\). The former case contradicts Proposition 3.8 since \((I_1, I_0) \in \mathcal{M} \circ \mathcal{M}^*\) and \(\text{Sol}_{\mathcal{M}(I_0)} \not\subseteq \text{Sol}_{\mathcal{M}(I_1)}\), while the latter case contradicts the same proposition since \((I_1, I_2) \in \mathcal{M} \circ \mathcal{M}^*\) and \(\text{Sol}_{\mathcal{M}(I_2)} \not\subseteq \text{Sol}_{\mathcal{M}(I_1)}\).

Given that there exists at least one dependency \(\varphi(\bar{x}) \rightarrow \psi(\bar{x}) \in \Sigma^*\) such that \(J_1 \models \varphi(\bar{c})\), where \(\bar{c}\) is a nonempty tuple of elements from \(C\), and given that \(S^{I_1} = \{(a, b)\}\), we have that \(\text{dom}(J_1) \cap C = \{a\}\). On the contrary, assume that this is not the case. First, suppose that \(a \not\in \text{dom}(J_1) \cap C\). Then given that \(J_1 \models \varphi(\bar{c})\) and \(\text{dom}(I_1) = \{a, b\}\), we conclude that every element of \(\bar{c}\) is equal to \(b\). Thus, since \((J_1, I_1) \models \Sigma^*\), \(S^{I_1} = \{(a, b)\}\) and \(\Sigma^*\) is a set of FO\(\exists\)-to-UCQ dependencies, we conclude that \((J_1, I_1) \models \Sigma^*\), where \(I_1\) is a source instance such that \(S^{I_1} = \{(b, b)\}\). Therefore, given that \((I_1, J_1) \in \mathcal{M}\), we have that \((I_1, I_3) \in \mathcal{M} \circ \mathcal{M}^*\), which implies by Proposition 3.8 that \(\text{Sol}_{\mathcal{M}(I_0)} \not\subseteq \text{Sol}_{\mathcal{M}(I_1)}\). But if \(J_0\) is the empty target instance, then \(J_0 \in \text{Sol}_{\mathcal{M}(I_1)}\) and \(J_0 \not\in \text{Sol}_{\mathcal{M}(I_1)}\), which leads to a contradiction. Second, suppose that \(b \in \text{dom}(J_1) \cap C\). Given that \(J_1 \models \varphi(\bar{c})\), \(a \in \text{dom}(J_1) \cap C\) and \(\text{dom}(I_1) = \{a, b\}\), we have that every element of \(\bar{c}\) is equal to either \(a\) or \(b\). Let \(\bar{c}'\) be a tuple generated from \(\bar{c}\) by replacing \(a\) by \(b\) and \(b\) by \(a\). Given that \(a\) and \(b\) are indistinguishable in \(J_1\), we conclude that \(J_1 \models \varphi(\bar{c}')\), which implies that \(\{(a, b)\} \not\subseteq S^{I_1}\) and, thus, contradicts the definition of \(I_1\). Third, assume that \(b \not\in S^{I_1}\).\(^\dagger\)

\(^\dagger\)Notice that if we consider \((J, I)\) and \((J^*, I)\) as structures over \(S \cup T \cup \{C(\cdot)\}\), then \(g\) is not an isomorphism from \((J, I)\) to \((J^*, I)\).
dom(J₁) ∩ C. Then there exists d ∈ dom(J₁) ∩ C such that d ≠ a and d ≠ b. Thus, given that J₁ ⊨ ϕ(⃗e), a ∈ dom(J₁) ∩ C and dom(J₁) = {a, b}, we have that every element of ⃗e is equal to a. Let ⃗e′ be a tuple generated from ⃗e by replacing a by d. Given that a and d are indistinguishable in J₁, we conclude that J₁ ⊨ ϕ(⃗e′), which implies that {(a, b)} ⊆ S₁ T and, thus, contradicts the definition of I₁.

Given that dom(J₁) ∩ C = {a}, (J₁, I₁) ⊨ Σ⁺ and Σ⁺ is a set of FO-C-to-UCQ dependencies, we conclude that (J₁, I₁) ⊨ Σ⁺ where I₄ is a source instance such that S₁ T = {(a, a)}. Thus, given that (I₁, I₁) ∈ M, we have that (I₁, I₄) ∈ M ∩ M⁺, which implies by Proposition 3.8 that Sol₄(M(I₄)) ⊆ Sol₄(M(I₁)). But if J₀ is the empty target instance, then J₀ ∈ Sol₄(M(I₄)) and J₀ ∉ Sol₄(M(I₁)), which leads to a contradiction. This concludes the proof of the proposition.

B.3 Proof of Theorem 8.3

Assume that there exists a nontrivial sentence Φ in L that is not expressible in L′, and let S be the schema of Φ. We define an st-mapping M = (S, T, Σ) as follows. Let P be a fresh relation name, T = {P(·)} and

Σ = {Φ → ∃xP(x)}.

For the sake of contradiction, assume that there exists a maximum recovery M⁺ = (T, S, Σ⁺) of M, where Σ⁺ is a nonempty set of CQ-to-L′ dependencies from \{P(·), C(·)\} to S. Furthermore, assume that Σ⁺ contains the following dependencies:

\[ \alpha_i → β_i \]
\[ γ_j(x_{j,1}, \ldots, x_{j,n_j}) → δ_j(x_{j,1}, \ldots, x_{j,n_j}) \]

1 ≤ i ≤ ℓ,
1 ≤ j ≤ m and 1 ≤ n_j

To prove the theorem, we consider two cases. In both cases, we denote by J₀ the empty instance for target schema T.

1. Assume that there exist instances I₁ of S and J₁ of T such that I₁ ⊨ Φ, (I₁, J₁) ∈ M, (J₁, I₁) ∈ M⁺ and dom(J₁) ⊆ N.

Let S be the set of indexes \{i \mid 1 ≤ i ≤ ℓ and J₁ ⊨ α_i\}. We note that S ≠ ∅.

On the contrary, assume that S = ∅, and let I₂ be an instance of S such that I₂ ∉ Φ (such an instance exists since Φ is a nontrivial sentence). Given that (J₁, I₁) ⊨ Σ⁺ and dom(J₁) ⊆ N, we have J₁ ⊨ ∃x₁,1 \cdots ∃x₁,n₁ γ₁(x₁,1, ⋮, x₁,n₁) for every i ∈ \{1, \ldots, m\}. Thus, given that S = ∅, we conclude that (J₁, I₂) ∈ M⁺ (in fact, (J₁, I₁) ∈ M⁺ for every instance I of S). Therefore, since (I₁, J₁) ∈ M, we have that (I₁, I₂) ∈ M ∩ M⁺. Thus, we have by Lemma 3.11 that Sol₄(M(I₁)) ⊆ Sol₄(M(I₁)) since M⁺ is a maximum recovery of M. But J₀ ∈ Sol₄(M(I₁)) since I₂ ∉ Φ, and J₀ ∉ Sol₄(M(I₁)) since I₁ ⊨ Φ, which leads to a contradiction.

Let Ψ be the following sentence:

\[ \bigwedge_{i ∈ S} β_i. \]

We note that this sentence is well defined since S ≠ ∅. Next we show that Φ is equivalent to Ψ. First, we assume that I is an instance of S such that I ⊨ Φ, and we prove that I ⊨ Ψ. Given that M⁺ is a recovery of M, there exists an instance J of T such that (I, J) ∈ M and (J, I) ∈ M⁺. Given that P′ ≠ ∅ and dom(J₁) ⊆ N, we know that there exists a homomorphism from J₁ to J. Thus, for every i ∈ S, we have that J ⊨ α_i
since $\alpha_i$ is a conjunctive query. We conclude that for every $i \in S$, it is the case that $I \models \beta_i$ (since $(J, I) \in \mathcal{M}^*$). Therefore, we have that $I \models \Psi$. Second, we assume that $I$ is an instance of $\mathcal{S}$ such that $I \models \Psi$, and we prove that $I \models \Phi$. On the contrary, assume that $I \not\models \Phi$. Given that $I \models \Psi$, we have that $(J_1, I) \in \mathcal{M}^*$ and, therefore, $(I_1, I) \in \mathcal{M} \circ \mathcal{M}^*$. We conclude by Lemma 3.11 that $\text{Sol}_{\mathcal{M}}(I) \subseteq \text{Sol}_{\mathcal{M}}(I_1)$ since $\mathcal{M}^*$ is a maximum recovery of $\mathcal{M}$. But $J_0 \in \text{Sol}_{\mathcal{M}}(I)$ since $I \not\models \Phi$, and $J_0 \not\in \text{Sol}_{\mathcal{M}}(I_1)$ since $I_1 \models \Phi$, which leads to a contradiction.

From the previous paragraph, we have that $\Phi$ is equivalent to $\Psi$. But this contradicts the fact that $\Phi$ is not expressible in $\mathcal{L}'$, since each $\beta_i (1 \leq i \leq \ell)$ is an $\mathcal{L}'$-sentence and $\mathcal{L}'$ is closed under conjunction.

(2) Assume that for every instance $I_1$ of $\mathcal{S}$, if $J_1$ is an instance of $\mathcal{T}$ such that $I_1 \models \Phi$, $(J_1, I_1) \in \mathcal{M}$ and $(J_1, I_1) \in \mathcal{M}^*$, then $\text{dom}(J_1) \cap C \neq \emptyset$. In this case, we consider two sub-cases.

(2.1) Assume that $m = 0$, that is, $\Sigma^* = \{\alpha_i \rightarrow \beta_i \mid 1 \leq i \leq \ell\}$, and let $\Psi$ be the following sentence:

$$\bigwedge_{i=1}^{\ell} \beta_i.$$ 

Next we show that $\Phi$ is equivalent to $\Psi$. First, we assume that $I$ is an instance of $\mathcal{S}$ such that $I \models \Phi$, and we prove that $I \models \Psi$. Given that $\mathcal{M}^*$ is a recovery of $\mathcal{M}$, there exists an instance $J$ of $\mathcal{T}$ such that $(I, J) \in \mathcal{M}$ and $(J, I) \in \mathcal{M}^*$. From the hypothesis, we have that $\text{dom}(J) \cap C \neq \emptyset$. Thus, given that each $\alpha_i (1 \leq i \leq \ell)$ is a conjunctive query over the vocabulary $\{P(i), C(i)\}$, we conclude that $J \models \alpha_i$ for every $i \in \{1, \ldots, \ell\}$. Therefore, for every $i \in \{1, \ldots, \ell\}$, it is the case that $I \models \beta_i$ (since $(J, I) \in \mathcal{M}^*$) and, hence, $I \models \Psi$. Second, we assume that $I$ is an instance of $\mathcal{S}$ such that $I \models \Psi$, and we prove that $I \models \Phi$. On the contrary, assume that $I \not\models \Phi$, and let $I_1$ be an instance of $\mathcal{S}$ such that $I_1 \models \Phi$ (such an instance exists since $\Phi$ is a nontrivial sentence). Given that $\mathcal{M}^*$ is a recovery of $\mathcal{M}$, there exists an instance $J_1$ of $\mathcal{T}$ such that $(I_1, J_1) \in \mathcal{M}$ and $(J_1, I_1) \in \mathcal{M}^*$. Thus, given that $I \models \Psi$, we have that $(J_1, I) \in \mathcal{M}^*$ and, therefore, $(J_1, I) \in \mathcal{M} \circ \mathcal{M}^*$. We conclude by Lemma 3.11 that $\text{Sol}_{\mathcal{M}}(I) \subseteq \text{Sol}_{\mathcal{M}}(I_1)$ since $\mathcal{M}^*$ is a maximum recovery of $\mathcal{M}$. But $J_0 \in \text{Sol}_{\mathcal{M}}(I)$ since $I \not\models \Phi$, and $J_0 \not\in \text{Sol}_{\mathcal{M}}(I_1)$ since $I_1 \models \Phi$, which leads to a contradiction.

From the previous paragraph, we have that $\Phi$ is equivalent to $\Psi$. But this contradicts the fact that $\Phi$ is not expressible in $\mathcal{L}'$, since each $\beta_i (1 \leq i \leq \ell)$ is an $\mathcal{L}'$-sentence and $\mathcal{L}'$ is closed under conjunction.

(2.2) Assume that $m > 0$, and let $\Psi$ be the following sentence:

$$\left(\bigwedge_{i=1}^{\ell} \beta_i \right) \land \left(\bigwedge_{j=1}^{m} \exists \delta_j(x, \ldots, x) \right),$$

where $\times$ times.

Next we show that $\Phi$ is equivalent to $\Psi$. First, we assume that $I$ is an instance of $\mathcal{S}$ such that $I \models \Phi$, and we prove that $I \models \Psi$. Given that $\mathcal{M}^*$ is a recovery of $\mathcal{M}$, there exists an instance $J$ of $\mathcal{T}$ such that $(I, J) \in \mathcal{M}$ and $(J, I) \in \mathcal{M}^*$. From the hypothesis, we have that $\text{dom}(J) \cap C \neq \emptyset$. Thus, given that each $\alpha_i (1 \leq i \leq \ell)$ is a conjunctive query over the vocabulary $\{P(i), C(i)\}$, we conclude that $J \models \alpha_i$ for every $i \in \{1, \ldots, \ell\}$. Furthermore, given that each $\gamma_j (1 \leq j \leq m)$ is a conjunctive
query over the vocabulary \{P(\cdot), C(\cdot)\}, we conclude that for every \(a \in \text{dom}(J) \cap C\) and \(j \in \{1, \ldots, m\} \):

\[
J \models \gamma_j(a, \ldots, a)_{n_j \text{ times}}
\]

Therefore, given that \((J, I) \in \mathcal{M}^*\), we have that for every \(i \in \{1, \ldots, \ell\}\), it is the case that \(I \models \beta_i\), and for every \(j \in \{1, \ldots, m\}\), it is the case that

\[
I \models \exists x \delta_j(x, \ldots, x)_{n_j \text{ times}}
\]

Thus, we have that \(I \models \Psi\). Second, we assume that \(I\) is an instance of \(S\) such that \(I \models \Phi\), and we prove that \(I \models \Phi\). On the contrary, assume that \(I \not\models \Phi\), and let \(I_1\) be an instance of \(S\) such that \(I_1 \models \Phi\) (such an instance exists since \(\Phi\) is a nontrivial sentence). Given that \(I \models \Psi\), there exists an element \(a \in \text{dom}(I)\) such that for every \(j \in \{1, \ldots, m\}\):

\[
I \models \delta_j(a, \ldots, a)_{n_j \text{ times}}
\]

Thus, if \(J_a\) is an instance of \(T\) such that \(P^{J_a} = \{a\}\), then \((J_a, I) \in \mathcal{M}^*\), and, therefore, \((I_1, I) \in \mathcal{M} \circ \mathcal{M}^*\) since \((I_1, J_a) \models \Sigma\). From Lemma 3.11, we have that \(\text{Sol}_{\mathcal{M}}(I) \subseteq \text{Sol}_{\mathcal{M}}(I_1)\) since \(\mathcal{M}^*\) is a maximum recovery of \(\mathcal{M}\). But \(J_0 \in \text{Sol}_{\mathcal{M}}(I)\) since \(I \not\models \Phi\), and \(J_0 \not\in \text{Sol}_{\mathcal{M}}(I_1)\) since \(I_1 \models \Phi\), which leads to a contradiction.

From the previous paragraph, we have that \(\Phi\) is equivalent to \(\Psi\). But this contradicts the fact that \(\Phi\) is not expressible in \(\mathcal{L}'\), since each \(\beta_i (1 \leq i \leq \ell)\) is an \(\mathcal{L}'\)-sentence and \(\mathcal{L}'\) is closed under conjunction, existential quantification and free-variable substitution. This concludes the proof of the theorem.

### B.4 Proof of Proposition 8.4

To prove the proposition, we need to introduce the notion of *Ehrenfeucht-Fraissé game*, which characterizes elementary equivalence in FO (see [Libkin 2004]). Let \(R\) be a relational schema. For every pair \(I_1, I_2\) of instances of \(R\), tuples \(a = (a_1, \ldots, a_m) \in \text{dom}(I_1)^m\) and \(b = (b_1, \ldots, b_m) \in \text{dom}(I_2)^m\) define a *partial isomorphism* from \(I_1\) to \(I_2\) if the following hold:

— For every \(i, j \leq m\), \(a_i = a_j\) iff \(b_i = b_j\).

— For every \(k\)-ary relation symbol \(R \in R\) and every sequence \([i_1, \ldots, i_k]\) of not necessarily distinct numbers from \(\{1, \ldots, m\}\), it holds that \((a_{i_1}, \ldots, a_{i_k}) \in R^{I_1}\) iff \((b_{i_1}, \ldots, b_{i_k}) \in R^{I_2}\).

— For every \(i \leq m\), \(a_i \in C\) iff \(b_i \in C\).

The Ehrenfeucht-Fraissé game is played by two players, called the *spoiler* and the *duplicator*, on two instances \(I_1, I_2\) of \(R\). In each round \(i\), the spoiler selects either a point \(a_i \in \text{dom}(I_1)\), or \(b_i \in \text{dom}(I_2)\), and the duplicator responds by selecting \(b_i \in \text{dom}(I_2)\), or \(a_i \in \text{dom}(I_1)\), respectively. The duplicator wins after \(m\) rounds if the tuples \((a_1, \ldots, a_m)\) and \((b_1, \ldots, b_m)\) define a partial isomorphism from \(I_1\) to \(I_2\), otherwise the spoiler wins.

We use notation \(I_1 \equiv_k I_2\) to indicate that the duplicator has a winning strategy in the \(k\)-round game on \(I_1\) and \(I_2\).
The quantifier rank of an FO formula is the maximum depth of quantifier nesting in it. It is well known that if \( I_1 \equiv_k I_2 \), then \( I_1 \) and \( I_2 \) agree on all FO sentences over \( R \cup \{C(\cdot)\} \) of quantifier rank \( k \) [Libkin 2004].

We have the necessary to continue with the proof of Proposition 8.4. In the proof of part (1), we use the following terminology. We say that an instance \( i \) of a schema \( R \) is the disjoint union of two instances \( I_1 \) and \( I_2 \) if \( \text{dom}(I_1) \cap \text{dom}(I_2) = \emptyset \) and \( R^I = R^{I_1} \cup R^{I_2} \) for every \( R \in R \). Furthermore, we say that \( I' \subseteq I \) is a connected component of \( I \) if: (a) for every \( a, b \in \text{dom}(I') \), there exist tuples \( t_1 \in R^I_1 \), \( t_m \in R^I_m \) and elements \( a_1, \ldots, a_{m-1} \) such that \( a \) is mentioned in \( t_1 \), \( b \) is mentioned in \( t_m \) and \( a_i \) is mentioned in \( t_i \) and \( t_{i+1} \), for every \( i \in \{1, \ldots, m-1\} \); and (b) for every \( I'' \subseteq I \) such that \( I' \subseteq I'' \) and \( I'' \) satisfies condition (a), it holds that \( I' = I'' \).

Let \( S = \{P(\cdot, \cdot)\} \), \( T = \{R(\cdot, \cdot), S(\cdot, \cdot)\} \),
\[
\Sigma = \{P(x, y) \rightarrow \exists z_1 \exists z_2 \exists z_3 (R(x, z_1) \land R(y, z_2) \land S(z_1, z_3) \land S(z_2, z_3))\},
\Gamma = \{R(x, y) \land R(x, z) \rightarrow y = z\},
S(x, y) \rightarrow x = y,
R(x, y) \land R(y, z) \land R(y', z) \rightarrow y = y',
S(x, y) \land R(y, z) \land R(y', z) \rightarrow y = y',
S(y, x) \land R(y, z) \land R(y', z) \rightarrow y = y',
R(x, y) \land R(x', y) \land R(y, z) \rightarrow x = x',
\]
and \( M = (S, T, \Sigma, \Gamma) \). For the sake of contradiction, assume that \( M^* = (T, S, \phi) \) is a maximum recovery of \( M \) that is specified by an FO sentence \( \phi \) over signature \( \{P(\cdot, \cdot), R(\cdot, \cdot), S(\cdot, \cdot), C(\cdot)\} \).

For every \( k \geq 2 \), let \( I_k \) be an instance of \( S \) such that \( P^{I_k} = \{(a_1, a_2), \ldots, (a_{k-1}, a_k)\} \) and \( C_k \) an instance of \( S \) such that \( P^{C_k} = \{(a_1, a_2), \ldots, (a_{k-1}, a_k), (a_k, a_1)\} \), where \( a_i \neq a_j \) (\( 1 \leq i < j \leq k \)).

**Claim B.1.** Let \( J \) be an instance of \( T \) such that \( R^J = \{(a_1, c), \ldots, (a_k, c)\} \) and \( S^J = \{(c, c)\} \), where \( c \neq a_i \) for every \( i \in \{1, \ldots, k\} \). Then \( J \) is a solution for both \( I_k \) and \( C_k \).

We conclude that every instance \( I_k \) is in the domain of \( M \). Thus, given that \( M^* \) is a maximum recovery of \( M \), we have by Lemma 3.11 that for every \( k \geq 2 \), there exists an instance \( J_k \) of \( T \) such that \( (I_k, J_k) \in M, (J_k, I_k) \in M^* \) and \( J_k \) is a witness for \( I_k \) under \( M \).

**Claim B.2.** For every \( k \geq 2 \), if \( J \) is a solution for \( I_k \), then \( J \) is the disjoint union of two instances \( I_1 \) and \( I_2 \), where \( J_2 \) could be the empty target instance and (a) \( \text{dom}(J_1) = \{a_1, \ldots, a_k, b_1, \ldots, b_\ell\} \), (b) \( a_i \neq c \) (\( 1 \leq i \leq k \)), (c) \( b_i \neq c \) (\( 1 \leq i \leq \ell \)), \( a_i \neq b_j \) (\( 1 \leq i \leq k \) and \( 1 \leq j \leq \ell \)), \( b_i \neq b_j \) (\( 1 \leq i < j \leq \ell \)), (c) \( R^{J_1} = \{(a_1, c), \ldots, (a_k, c), (b_1, c), \ldots, (b_\ell, c)\} \) and \( S^{J_1} = \{(c, c)\} \).

To prove the claim, assume that \( (I_k, J) \models \Sigma \). Then we have that \( J \) contains at least all the tuples shown in the following figure:
By using Claim B.2, it is straightforward to prove the following result.

ACM Transactions on Database Systems, Vol. V, No. N, Month 20YY.
CLAIM B.3. There exists a constant \( s_0 > 0 \) such that, for every \( k > s_0 \), it holds that \((I_k, J_k) \equiv_k (I'_k, J'_k)\)

Now we have all the necessary ingredients to prove the first part of the proposition. Assume that the quantifier rank of \( \varphi \) is \( k_0 \), and let \( s > \max\{k_0, s_0\} \). Next we show that \((I_s, I'_s) \in \mathcal{M} \circ \mathcal{M}'^*\). Given that \( \mathcal{M}'^* \) is specified by \( \varphi \) and \((J_s, I_s) \in \mathcal{M}^*\), we have that \((J_s, I_s) \equiv \varphi\). Thus, given that \( s > k_0 \) and \((J_s, I_s) \equiv_s (J'_s, I'_s)\) (by Claim B.3), we have that \((J'_s, I'_s) \equiv \varphi\). We conclude that \((I_s, I'_s) \in \mathcal{M} \circ \mathcal{M}'^*\), which implies that \((I_s, I'_s) \in \mathcal{M} \circ \mathcal{M}'^*\).

Given that \( \mathcal{M}^* \) is a maximum recovery of \( \mathcal{M} \) and \((I_s, I'_s) \in \mathcal{M} \circ \mathcal{M}'^*\), we know by Proposition 3.8 that \( \text{Sol}_\mathcal{M}(I'_s) \subseteq \text{Sol}_\mathcal{M}(I_s) \). Let \( J^* \) be an instance of \( \mathbf{T} \) defined as:

\[
R^{J^*} = \{(a_1, c_1), \ldots, (a_{s+1}, c_1)\} \cup \{(a_{s+2}, c_2), \ldots, (a_s, c_2)\},
\]

\[
S^{J^*} = \{(c_1, c_1), (c_2, c_2)\},
\]

where \( c_1 \neq a_i (1 \leq i \leq s), c_2 \neq a_i (1 \leq i \leq s) \) and \( c_1 \neq c_2 \). Then by definition of \( \mathcal{M} \) and Claim B.1, we have that \( J^* \in \text{Sol}_\mathcal{M}(I'_s) \). Furthermore, we have that \( J^* \not\in \text{Sol}_\mathcal{M}(I_s) \), which contradicts the fact that \( \text{Sol}_\mathcal{M}(I'_s) \subseteq \text{Sol}_\mathcal{M}(I_s) \). This concludes the proof of the first part of the proposition.

To prove part (2), let \( \mathbf{S} = \{P(\cdot, \cdot)\}, \mathbf{T} = \{R(\cdot, \cdot)\}, \Sigma = \{P(x, y) \rightarrow R(x, y)\}, \Gamma = \{R(x, y) \rightarrow R(x, z)\}; \) \( \mathcal{M} = (\mathbf{S}, \mathbf{T}, \Sigma, \Gamma) \). For the sake of contradiction, assume that \( \mathcal{M}^* = (\mathbf{T}, \mathbf{S}, \varphi) \) is a maximum recovery of \( \mathcal{M} \) that is specified by an FO sentence \( \varphi \) over signature \( \{P(\cdot, \cdot), R(\cdot, \cdot), C(\cdot)\} \). Given that \( \mathcal{M}^* \) is a maximum recovery of \( \mathcal{M} \), we know from Lemma 3.11 that for every instance \( I \) of \( \mathbf{S} \), there exists an instance \( J_I \) of \( \mathbf{T} \) such that \((I, J_I) \in \mathcal{M}, (J_I, I) \in \mathcal{M}^* \) and \( J_I \) is a witness for \( I \) under \( \mathcal{M} \). Since \((I, J_I) \in \mathcal{M}, \) we have that \( J_I \) satisfies set \( \Gamma \) of target tgd. Given an instance \( I \) of \( \mathbf{S} \), define \( \text{TrCl}(I) \) as the transitive closure of \( I \). Then we have that:

CLAIM B.4. For every source instance \( I, R^I = \text{TrCl}(I) \cup N_I \), where \( \text{TrCl}(I) \) and \( N_I \) are disjoint and for every \((a, b) \in N_I \), it holds that \( a \not\in \textbf{N} \) or \( b \not\in \textbf{N} \).

We now prove the claim. Given that \( J_I \) is a solution for \( I \) under \( \mathcal{M} \), we have that \( \text{TrCl}(I) \subseteq R^I \) and, hence, \( R^I = \text{TrCl}(I) \cup N_I \), where \( \text{TrCl}(I) \) and \( N_I \) are disjoint. For the sake of contradiction, assume that there exists a tuple \((a, b) \in N_I \) such that \( a \) and \( b \) are both elements of \( \mathbf{C} \).

Let \( I' \) be an instance of \( \mathbf{S} \) defined as \( P^I = P^I \cup \{(a, b)\} \). Given that \( J_I \) satisfies \( \Gamma \), we have that \( J_I \in \text{Sol}_\mathcal{M}(I') \) and, therefore, \( \text{Sol}_\mathcal{M}(I) \subseteq \text{Sol}_\mathcal{M}(I') \) since \( J_I \) is a witness for \( I \) under \( \mathcal{M} \). Let \( J' \) be an instance of \( \mathbf{T} \) defined as \( R^{J'} = \text{TrCl}(I') \). We have that \( J' \in \text{Sol}_\mathcal{M}(I) \), and given that \( \text{TrCl}(I) \) and \( N_I \) are disjoint, we have that \((a, b) \not\in \text{TrCl}(I) \) and, thus, \( J' \not\in \text{Sol}_\mathcal{M}(I') \). This contradicts the fact that \( \text{Sol}_\mathcal{M}(I) \subseteq \text{Sol}_\mathcal{M}(I') \), and concludes the proof of the claim.

To prove the proposition, we need to introduce some terminology. Let \( k_0 \) be the quantifier rank of \( \varphi \). An instance \((I, J)\) of \( \mathbf{S} \) or \( \mathbf{T} \) is \( \mathcal{M} \)-\textit{cyclic} if \( \text{dom}(I) = \{a_1, \ldots, a_m\}, \text{dom}(J) = \{a_1, \ldots, a_m\} \cup \{d_1, \ldots, d_n\}, \) \( P^I = \{(a_i, a_{i+1}) \mid 1 \leq i \leq m-1\} \cup \{(a_m, a_1)\} \) and \( R^I = \{(a_i, a_j) \mid 1 \leq i, j \leq m\} \cup X \), where \( X \) satisfies the following conditions: (1) if \((u, v) \in X \), then \( u \in \textbf{N} \) or \( v \in \textbf{N} \); and (2) if \((a_i, d_j) \in X \), then \((a_k, d_j) \in X \) for every
Next we show that Claim B.5, we have that if \( \phi \) is an amplification of \( (I, J) \) if \( \text{dom}(I') = \{b_1, \ldots, b_{2m}\} \cup \{d_1, \ldots, d_n\} \), \( P'' = \{(b_i, b_{i+1}) \mid 1 \leq i \leq m-1\} \cup \{(b_{m+n}, b_{m+n+1}) \mid 1 \leq i \leq m-1\} \cup \{(b_{2m}, b_{m+n+1})\}\) and \( R' = \{(b_i, b_j) \mid 1 \leq i, j \leq 2m\} \cup Y \), where \( Y \) satisfies the following conditions: (1) if \( (u, v) \in Y \), then \( u \in N \) or \( v \in N \); (2) \( X \cap \{(d_1, \ldots, d_n) \times \{d_1, \ldots, d_n\}\} = Y \cap \{\{d_1, \ldots, d_n\} \times \{d_1, \ldots, d_n\}\}; (3) if \( (b_i, b_j) \in Y \), then \( (b_k, b_k) \in Y \) for every \( k \in \{1, \ldots, 2m\} \), and if \( (d_j, b_i) \in Y \), then \( (d_j, b_k) \in Y \) for every \( k \in \{1, \ldots, 2m\} \).

**Claim B.5.** There exists a constant \( m_0 > 0 \) such that, if the domain of an \( \mathcal{M} \)-cyclic instance \((I, J)\) contains at least \( m_0 \) elements, and \((I', J')\) is an amplification of \((I, J)\), then \((I, J) \equiv_{k_0} (I', J')\).

Now we have all the necessary ingredients to prove part (2) of the proposition. Let \( m > m_0 \) and \( I_1, I_2 \) be instances of \( \mathcal{S} \) defined as:

\[
P_{I_1} = \{(b_i, b_{i+1}) \mid 1 \leq i \leq m-1\} \cup \{(b_{m+n}, b_{m+n+1}) \mid 1 \leq i \leq m-1\} \cup \{(b_{2m}, b_{m+n+1})\},
\]
\[
P_{I_2} = \{(b_i, b_{i+1}) \mid 1 \leq i \leq m-1\} \cup \{(b_{m+n}, b_{m+n+1}) \mid 1 \leq i \leq m-1\} \cup \{(b_{2m}, b_{m+n+1})\}.
\]

Next we show that \((I_2, I_1) \in \mathcal{M} \circ \mathcal{M}^*\). Let \( I \) be an instance of \( \mathcal{S} \) defined as \( P' = \{(a_i, a_{i+1}) \mid 1 \leq i \leq m-1\} \cup \{(a_n, a_1)\}. \) Given that \( \text{TrCl}(I) = \{(a_i, a_j) \mid 1 \leq i, j \leq m\} \), we have that \((I, J)\) is \( \mathcal{M} \)-cyclic by Claim B.4 and the fact that \( J \) satisfies \( \Gamma \). Thus, by Claim B.5, we have that if \((I', J')\) is an amplification of \((I, J)\), then \((I, J) \equiv_{k_0} (I', J')\).

It is straightforward to prove that there is an amplification \((I', J')\) of \((I, J)\) such that \( I' = I_1 \) and \( J' = \text{Sol}_\mathcal{M}(I_2) \). Thus, given that \( \mathcal{M}^* \) is specified by FO-sentence \( \varphi \), the quantifier rank of \( \varphi \) is \( k_0 \) and \((I, J) \equiv_{k_0} (I_1, J')\), we conclude that \((J', I_1) \in \mathcal{M}^*\). Hence, \((I_2, I_1) \in \mathcal{M} \circ \mathcal{M}^* \) since \((I_2, J') \in \mathcal{M} \) and \((J', I_1) \in \mathcal{M}^*\).

Given that \( \mathcal{M}^* \) is a maximum recovery of \( \mathcal{M} \) and \((I_2, I_1) \in \mathcal{M} \circ \mathcal{M}^* \), we know by Proposition 3.8 that \( \text{Sol}_\mathcal{M}(I_1) \subseteq \text{Sol}_\mathcal{M}(I_2) \). But let \( J \) be an instance of \( \mathcal{T} \) defined as \( R' = \text{TrCl}(I_1) \). Then we have that \( J \in \text{Sol}_\mathcal{M}(I_1) \) and \( J \not\in \text{Sol}_\mathcal{M}(I_2) \) since \((b_{m+n}, b_{m+n+1}) \in P_{I_2} \) and \((b_{m+n}, b_{m+n+1}) \not\in R'\). This contradicts the fact that \( \text{Sol}_\mathcal{M}(I_1) \subseteq \text{Sol}_\mathcal{M}(I_2) \), and concludes the proof of the proposition.

C. PROOFS OF SECTION 9

C.1 Proof of Lemma 9.2

\((\Rightarrow)\) Trivial.

\((\Leftarrow)\) 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-tgs, \( \mathcal{M} \) 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.
C.2 Proof of Lemma 9.3

(⇒) From [Fagin et al. 2005], we know that \( M \circ M' \) can be specified by a set of st-tgds. Now, as in [Fagin 2007] (Proposition 7.2) we know that chase\(_{\Sigma'}(\text{chase}\_\Sigma(I))\) is a universal solution for \( I \) under \( M \circ M' \), and then \((I, I) \in M \circ M'\) if and only if there exists a homomorphism from chase\(_{\Sigma'}(\text{chase}\_\Sigma(I))\) to \( I \) that is the identity on \( C \). The (⇒) direction of the proposition follows from the latter condition.

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

Given that \( M' \) is not a recovery of \( M \), there exists an instance \( I_1 \) of \( S \) such that \((I_1, I_1) \notin M \circ M'\). Let \( I \) be an instance of \( S \). Given that chase\(_{\Sigma'}(I)\) is a universal solution for \( I \) under \( M \) and chase\(_{\Sigma'}(\text{chase}\_\Sigma(I))\) is a universal solution for chase\(_{\Sigma'}(I)\) under \( M' \), it is straightforward to prove that if \((I, I') \in M \circ M'\), then there exists a homomorphism from chase\(_{\Sigma'}(\text{chase}\_\Sigma(I))\) to \( I' \) that is the identity on \( C \). Furthermore, if there exists a homomorphism from chase\(_{\Sigma'}(\text{chase}\_\Sigma(I))\) to an instance \( I' \), then one can conclude that \((\text{chase}\_\Sigma(I), I') \in M' \) since \((\text{chase}\_\Sigma(I), \text{chase}\_\Sigma(\text{chase}\_\Sigma(I))) \in M'\) and chase\(_{\Sigma'}(I)\) does not mention any null values as \( \Sigma \) is a set of full st-tgds. Thus, we have that if there exists a homomorphism from chase\(_{\Sigma'}(\text{chase}\_\Sigma(I))\) to an instance \( I' \), then \((I, I') \in M \circ M'\). In particular, from the previous properties, we conclude that \((I, I) \in M \circ M'\) if and only if there exists a homomorphism from chase\(_{\Sigma'}(\text{chase}\_\Sigma(I))\) to \( I \) that is the identity on \( C \). Thus, given that \((I_1, I_1) \notin M \circ M'\), there is no homomorphism from chase\(_{\Sigma'}(\text{chase}\_\Sigma(I_1))\) to \( I_1 \) that is the identity on \( C \), which implies that there exists an \( \Sigma \)-connected component \( K_1 \) of chase\(_{\Sigma'}(\text{chase}\_\Sigma(I_1))\) such that there is no homomorphism from \( K_1 \) to \( I_1 \) that is the identity on \( C \). Given that \( K_1 \) is an \( \Sigma \)-connected component and \( \Sigma \) is a set of full st-tgds, there exists a dependency \( \alpha(x) \rightarrow \exists \beta(x, y) \in \Sigma' \) and a tuple \( \bar{a} \) of elements from \( \Sigma \) such that chase\(_{\Sigma'}(I_1) \models \alpha(\bar{a}) \) and \( K_1 \) is generated from \( \exists \beta(\bar{a}, y) \) when computing chase\(_{\Sigma'}(\text{chase}\_\Sigma(I_1))\). Assume that \( \alpha(\bar{a}) \) is equal to \( T_1(\bar{a}_1) \land \cdots \land T_n(\bar{a}_n) \). 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 \( \Sigma \)-connected component of chase\(_{\Sigma'}(\text{chase}\_\Sigma(I_2))\) and there is no homomorphism from \( K_1 \) to \( I_2 \) that is the identity on \( C \). But by definition of \( I_2 \), we know that \(|I_2| \leq k_{\Sigma' \cdot k_{\Sigma'}}\), which contradicts our initial assumption.

C.3 Proof of Theorem 9.4

To prove this theorem, we reduce the problem of verifying whether a deterministic Turing Machine (DTM) accepts the empty string to the complement of the problem of verifying whether a schema mapping \( M' \) is a recovery of a schema mapping \( M \).

Let \( M = (Q, \Gamma, q_0, \delta, q_f) \) be a DTM, where \( Q \), \( \Gamma \), \( q_0 \), \( \delta \) and \( q_f \) are the finite set of states, tape alphabet, initial state, transition function and final state of \( M \). For the sake of simplicity, we assume that \( q_0 \neq q_f \), the input alphabet is \( \{0, 1\} \) and the tape alphabet \( \Gamma \) is \( \{0, 1, \bar{b}\} \), where \( \bar{b} \) is the blank symbol. Furthermore, we assume that the tape of \( M \) is infinite to the right and that in each step of \( M \) the head has to move either to the right (\( R \)) or to the left (\( L \)), that is, \( \delta \) is a total function from \( (Q \setminus \{q_f\}) \times \Gamma \) to \( Q \times \Gamma \times \{L, R\} \). Notice that we also assume that no transitions are defined for the final state \( q_f \).
Next we define data exchange settings $\mathcal{M} = (\mathcal{S}, \mathcal{T}, \Sigma)$ and $\mathcal{M}' = (\mathcal{T}, \mathcal{S}, \Sigma')$ in such a way that $\mathcal{M}$ accepts the empty string if and only if $\mathcal{M}'$ is not a recovery of $\mathcal{M}$. Relational schemas $\mathcal{S}$ and $\mathcal{T}$ are defined as follows:

$\mathcal{S} := \{D(\cdot, \cdot), T(\cdot)\}$,
$\mathcal{T} := \{D'(\cdot, \cdot), T'(\cdot), Z(\cdot), O(\cdot), E(\cdot, \cdot), L(\cdot, \cdot), P(\cdot, \cdot), U(\cdot, \cdot), S(\cdot, \cdot), H(\cdot, \cdot, \cdot), T_0(\cdot, \cdot, \cdot), T_1(\cdot, \cdot, \cdot), T_2(\cdot, \cdot, \cdot)\} \cup \{S_q(\cdot, \cdot) \mid q \in Q\}$.

Before defining sets $\Sigma$ and $\Sigma'$ of tgds, we give the intended interpretations of the predicates of $\mathcal{S}$ and $\mathcal{T}$. As it is customary when doing a logical encoding of a Turing Machine, we have a predicate $L$ to store a linear order. Since negation is not allowed in tgds, predicate $L$ is ternary and its third argument is used to indicate whether the first argument is or is not less than the second one. Predicate $Z$ in $\mathcal{T}$ is used to store elements that represent the truth value $false$ (or zero), while predicate $O$ in $\mathcal{T}$ is used to store elements that represent the truth value $true$ (or one). Thus, for example, if we want to say that $a$ is less than $b$ according to linear order $L$, then we add a tuple $(a, b, c)$ to $L$, where $c$ belongs to $O$.

As it is also customary when doing a logical encoding of a Turing Machine, we have predicates $P$ and $U$ to store the first and last elements of linear order $L$, a predicate $S$ to store the successor relation associated with $L$, a predicate $H$ to store the position of the head of DTM $M$, predicates $S_q (q \in Q)$ to store the state of $M$ and predicates $T_0, T_1, T_2$ to indicate for each cell of the tape of $M$ whether its value is either 0 or 1 or $\sqcup$, respectively. As for the case of linear order $L$, all these predicates have an extra argument that is used to indicate whether the predicate is true or false for a particular tuple. Since the equality symbol $=$ is not allowed in st-tgds, we also have an equality predicate $E$, which is ternary, as in the previous cases.

To explain the intended interpretations of predicates $T$ of $\mathcal{S}$ and $T'$ of $\mathcal{T}$, we need to show some of the dependencies in $\Sigma$ and $\Sigma'$. Set $\Sigma$ contains st-tgds:

$$D(x, y) \rightarrow D'(x, y),$$
$$T(x) \rightarrow T'(x)$$

and $\Sigma'$ contains ts-tgds:

$$D'(x, y) \rightarrow D(x, y),$$
$$T'(x) \rightarrow T(x),$$
$$Z(x) \rightarrow T(x),$$
$$O(x) \rightarrow T(x).$$

Thus, if $I$ is an instance of $\mathcal{S}$ and $J$ is a solution for $I$ under $\mathcal{M} \circ \mathcal{M}'$, then $I \subseteq J$. After giving the definitions of $\Sigma$ and $\Sigma'$, we prove that DTM $M$ accepts the empty string if and only if there exists an instance $I$ of $\mathcal{S}$ such that $(I, I) \notin \mathcal{M} \circ \mathcal{M}'$ and, thus, we conclude that $M$ accepts the empty string if and only if $\mathcal{M}'$ is not a recovery of $\mathcal{M}$.

Let $I$ be an instance of $\mathcal{S}$ and $\text{dom}(D')$ the set of all elements mentioned in $D'$. Each element in $\text{dom}(D')$ is used to denote a position in the tape of $M$ as well as a point of time in the execution of this DTM. Thus, some of the dependencies of $\Sigma$ should be used to guess a linear order $L$ on $\text{dom}(D')$. More precisely, if $\text{dom}_D(x)$ is defined as $\exists y D(x, y) \lor \exists z D(z, x)$, then we should include in $\Sigma$ an axiom of the form $\text{dom}_D(x) \land \text{dom}_D(y) \rightarrow \exists u L(x, y, u)$ stating that for every pair of elements $x$ and $y$ in $D$, there is
a truth value associated with the statement “x is less than y according to L”, but with the additional restriction that u belongs to either Z or O. We cannot impose this restriction by using an axiom of the form $\text{dom}_D(x) \land \text{dom}_D(y) \rightarrow \exists u (L(x, y, u) \land Z(u))$ because we will be explicitly saying that x is not less than y according to L, and the same happens with axiom $\text{dom}_D(x) \land \text{dom}_D(y) \rightarrow \exists u (L(x, y, u) \land O(u))$. To overcome this problem, we use the fact that $T'$ is a copy of T and all the elements of Z and O belong to $T'$ (since $Z(x) \rightarrow T(x)$ and $O(x) \rightarrow T(x)$ are in $\Sigma'$), and we replace the previous axiom by $\text{dom}_D(x) \land \text{dom}_D(y) \rightarrow \exists u (L(x, y, u) \land T'(u))$, which indeed says that u is a truth value but without explicitly stating whether this value is true of false.

In the definition of L, we would also like to say that L is defined only for the elements in $\text{dom}(D')$, thus avoiding extra elements that are not in D and can behave inadequately. The problem with this is that we cannot include in $\Sigma'$ an axiom of the form $L(x, y, z) \rightarrow \text{dom}_D(x) \land \text{dom}_D(y)$, where $\text{dom}_D(x) = \exists y D(x, y) \lor \exists z D(z, x)$, because this disjunctive sentence is not equivalent to any set of tgds, unlike $\text{dom}_D(x) \land \text{dom}_D(y) \rightarrow \exists u (L(x, y, u) \land T'(u))$ which is equivalent to a set of four tgds. To overcome this problem, we simply replace disjunction by conjunction in the definitions of $\text{dom}_D(x)$ and $\text{dom}_D(x)$, and thus we replace the previous dependency by:

\[
\begin{align*}
\mu(x) \land \mu(y) \rightarrow \exists u (L(x, y, u) \land T'(u)), \\
L(x, y, z) \rightarrow \mu(x) \land \mu(y),
\end{align*}
\]

where $\mu(x) = \exists y D(x, y) \land \exists z D(z, x)$. Hence, predicate L is only defined for the elements that appear in both columns of D.

In order to properly define L, we also need to include the axioms that define L as a linear order. In particular, we need to say that L is connected, that is, for every pair of distinct elements x and y, we have that x is less than y or y is less than x according to L. Since the equality symbol is not allowed in st-tgds, we use the equality predicate E to express this axiom. Thus, $\Sigma$ also has to include some dependencies defining E. In particular, we include in $\Sigma$ an axiom of the form $D(x, y) \rightarrow \exists u (E(x, y, u) \land Z(u))$, saying that for every tuple $(x, y)$ in D, we have that x and y are distinct elements. Thus, we also use D to store an inequality relation.

A natural question at this point is what happens if some of the implicit assumptions made above are not satisfied. For example, what happens if D or T is empty, or if D contains a tuple of the form $(x, x)$, or if Z and O share some values (which implies that some facts could be both true and false), or if $T'$ contains a value which is neither in Z nor in O, or if there is no element $a$ in D satisfying $\mu$? After defining $\Sigma$ and $\Sigma'$, we make explicit these assumptions, and we show that some of the conditions in $\Sigma$ and $\Sigma'$ ensure that, if a source instance I does not satisfy any of these assumptions, then $(I, I) \in M \circ M'$. Thus, when checking whether $M'$ is a recovery of M, we only need to take into account source instances satisfying these assumptions.

Now we are ready to define sets $\Sigma$ and $\Sigma'$ of tgds. For the sake of completeness, we also include some of the dependencies already mentioned.

**Copying axioms.** $\Sigma$ contains copying st-tgds:

\[
\begin{align*}
D(x, y) & \rightarrow D'(x, y), \\
T(x) & \rightarrow T'(x),
\end{align*}
\]
and $\Sigma'$ contains copying ts-tgds:

\[ D'(x, y) \rightarrow D(x, y), \]
\[ T'(x) \rightarrow T(x). \]

**Definition of predicates Z and O.** Let

\[ \lambda := \exists x \exists y (D(x, y) \land T(x) \land T(y)). \]

To define predicates $Z$ and $O$, we include the following st-tgd in $\Sigma$:

\[ \lambda \rightarrow \exists u O(u) \land \exists v Z(v). \]

This dependency says that if a source instance $I$ satisfies $\lambda$, then for every solution $J$ for $I$, both $Z^J$ and $O^J$ are not empty. Sentence $\lambda$ is included in this dependency to deal with source instances not satisfying some of the assumptions mentioned above (this will be formalized in Lemmas C.1, C.2 and C.3).

To complete the definition of $Z$ and $O$, we include the following ts-tgds in $\Sigma'$:

\[ Z(x) \rightarrow T(x), \]
\[ O(x) \rightarrow T(x), \]
\[ Z(x) \land O(y) \rightarrow D(x, y). \]

**Definition of equality predicate E.** Let

\[ \mu(x) := \exists y D(x, y) \land \exists z D(z, x). \]

To define the equality predicate $E$, we include the following st-tgds in $\Sigma$:

\[ \lambda \land \mu(x) \land \mu(y) \rightarrow \exists u (E(x, y, u) \land T'(u)), \]
\[ \lambda \land \mu(x) \land \mu(y) \land D(x, y) \rightarrow \exists u (E(x, y, u) \land Z(u)). \]

The first dependency says that for every pair of elements $x, y$ satisfying $\mu$, either $x$ is equal to $y$ or $x$ is not equal to $y$ according to $E$. The second st-tgd says that for every tuple $(x, y)$ in $D$, we have that $x$ is different from $y$ (according to $E$). We also need to state that $E$ is an equivalence relation, which is done by including the following dependencies into $\Sigma'$:

\[ E(x, y, u_1) \land Z(u_1) \land E(x, y, u_2) \land O(u_2) \rightarrow \exists v D(v, v), \]
\[ E(x, x, u) \land Z(u) \rightarrow \exists v D(v, v), \]
\[ E(x, y, u_1) \land O(u_1) \land E(y, x, u_2) \land Z(u_2) \rightarrow \exists v D(v, v), \]
\[ E(x, y, u_1) \land O(u_1) \land E(y, z, u_2) \land O(u_2) \land E(x, z, u_3) \land Z(u_3) \rightarrow \exists v D(v, v). \]

The first ts-tgd above says that unless there is a tuple $(a, a)$ in $D$, it could not be the case that $x$ is equal to $y$ and $x$ is not equal to $y$ according to $E$. The remaining three dependencies define $E$ as an equivalent relation, provided that $D$ does not contain a tuple $(a, a)$.

Finally, we also include in $\Sigma'$ the following ts-tgd:

\[ E(x, y, z) \rightarrow \mu(x) \land \mu(y), \]

stating that $E$ is only defined for the elements that satisfy $\mu$. 

ACM Transactions on Database Systems, Vol. V, No. N, Month 20YY.
Definition of linear order $L$. We include the following st-tgd in $\Sigma'$:

$$
\lambda \land \mu(x) \land \mu(y) \rightarrow \exists u \ (L(x, y, u) \land T'(u)),
$$

and we include the following ts-tgds in $\Sigma'$:

$$
L(x, x, u) \land O(u) \rightarrow \exists v \ D(v, v),
$$

$$
L(x, y, u_1) \land O(u_1) \land L(y, z, u_2) \land O(u_2) \land L(x, z, u_3) \land Z(u_3) \rightarrow \exists v \ D(v, v),
$$

$$
E(x, y, u_1) \land Z(u_1) \land L(x, y, u_2) \land Z(u_2) \land L(y, x, u_3) \land Z(u_3) \rightarrow \exists v \ D(v, v).
$$

The first dependency above says that $L$ is irreflexive, the second says that $L$ is transitive and the third says that $L$ is connected (unless there is a tuple $(a, a)$ in $D$). To complete the definition of $L$, we include in $\Sigma'$ the following ts-tgd:

$$
L(x, y, z) \rightarrow \mu(x) \land \mu(y),
$$

stating that $L$ is only defined for the elements that satisfy $\mu$, and we also include in $\Sigma'$ an axiom that states that $L$ is consistent with $E$ (that is, $E$ is a congruence relation for $L$):

$$
L(x, y, u_1) \land O(u_1) \land E(x, x_1, u_2) \land O(u_2) \land
E(y, y_1, u_3) \land O(u_3) \land L(x_1, y_1, u_4) \land Z(u_4) \rightarrow \exists v \ D(v, v).
$$

This dependency says that unless there is a tuple $(a, a)$ in $D$, if $x$ is equal to $x_1$ and $y$ is equal to $y_1$ according to $E$, then $x$ is less than $y$ according to $L$ if and only if $x_1$ is less than $y_1$ according to $L$.

Definition of predicate $P$. This predicate is used to store the first element of $L$. To define this predicate, we include in $\Sigma$ the following st-tgd:

$$
\lambda \land \mu(x) \rightarrow \exists u_1 \exists u_2 \ (P(u_1, u_2) \land O(u_2)).
$$

This st-tgd says that if there exists at least one element satisfying formula $\mu$, and also formula $\lambda$ is satisfied, then there is a first element for linear order $L$. Moreover, we also include the following ts-tgds in $\Sigma'$:

$$
P(x, u_1) \land O(u_1) \land L(x, y, u_2) \land O(u_2) \rightarrow \exists v \ D(v, v),
$$

$$
P(x, u_1) \land O(u_1) \land E(x, y, u_2) \land O(u_2) \land P(y, u_3) \land Z(u_3) \rightarrow \exists v \ D(v, v),
$$

$$
P(x, y) \rightarrow \mu(x).
$$

The first dependency above says that the first element of $L$ does not have a predecessor according to $L$, while the second says that unless there is a tuple $(a, a)$ in $D$, if $x$ is equal to $y$ according to $E$, then $x$ is the first element of $L$ if and only if $y$ is the first element of $L$ (that is, $E$ is a congruence relation for $P$). Moreover, the last dependency above states that $P$ is only defined for the elements that satisfy $\mu$.

Definition of predicate $U$. This predicate is used to store the last element of $L$. To define this predicate, we include the following st-tgd in $\Sigma$:

$$
\lambda \land \mu(x) \rightarrow \exists u_1 \exists u_2 \ (U(u_1, u_2) \land O(u_2)).
$$
and we include the following ts-tgds in $\Sigma'$:

$$U(x, u_1) \land O(u_1) \land L(x, y, u_2) \land O(u_2) \rightarrow \exists v \; D(v, v),$$
$$U(x, u_1) \land O(u_1) \land E(x, y, u_2) \land O(u_2) \land U(y, u_3) \land O(u_3) \land Z(u_3) \rightarrow \exists v \; D(v, v),$$
$$U(x, y) \rightarrow \mu(x).$$

**Definition of successor predicate $S$.** We include the following st-tgds in $\Sigma$:

$$\lambda \land \mu(x) \land \mu(y) \rightarrow \exists u \; (S(x, y, u) \land T'(u)),$$
$$\lambda \land \mu(x) \rightarrow \exists y \exists u \; (S(x, y, u) \land O(u)).$$

As for the case of linear order $L$, the first st-tgd is used to indicate that for every pair of elements $x$, $y$ satisfying formula $\mu$, there is a truth value associated with the statement "$y$ is a successor of $x$ according to $S$". The second st-tgd above is used to indicate that every element satisfying $\mu$ has a successor element. We note that this dependency states that even the last element of linear order $L$ has a successor. This does not create any problems, as we do not impose any restrictions on the successor of the last element (for example, we do not say that it should be greater than the last element), and we do not use this successor when coding Turing Machine $M$.

To indicate that $S$ is the successor relation associated with $L$, we include the following ts-tgds in $\Sigma'$:

$$S(x, y, u_1) \land O(u_1) \land U(x, u_2) \land Z(u_2) \land L(x, y, u_3) \land Z(u_3) \rightarrow \exists v \; D(v, v),$$
$$S(x, y, u_1) \land O(u_1) \land U(x, u_2) \land Z(u_2) \land L(x, z, u_3) \land O(u_3) \land L(z, y, u_4) \land O(u_4) \rightarrow \exists v \; D(v, v),$$
$$S(x, y, z) \rightarrow \mu(x) \land \mu(y).$$

The first dependency above says that if $y$ is a successor of $x$ according to $S$ and $x$ is not the last element of $L$, then $x$ is less than $y$ according to $L$ (unless there is a tuple $(a, a)$ in $D$). As we mentioned above, this dependency does not impose any restrictions on the successor of the last element. The second ts-tgd says that there are no elements in between $x$ and $y$ if $y$ is a successor of $x$ and $x$ is not the last element. The third dependency says that $S$ is only defined for elements satisfying $\mu$, which in particular implies that the successor of an element must satisfy $\mu$. As for the case of predicates $L$, $P$ and $U$, we also need to include an axiom saying that $S$ is consistent with the equality predicate $E$:

$$S(x, y, u_1) \land O(u_1) \land E(x, x_1, u_2) \land O(u_2) \land E(y, y_1, u_3) \land O(u_3) \land S(x_1, y_1, u_4) \land Z(u_4) \rightarrow \exists v \; D(v, v).$$

**Encoding of DTM $M$.** Now we are ready to present the dependencies that code DTM $M$.

As for the case of predicates $E$, $L$, $P$, $U$ and $S$, we include an extra argument in predicates $H$, $T_0$, $T_1$, $T_2$, and $S_q$ ($q \in Q$) to indicate whether a particular tuple is or is not in these predicates. Thus, for example, if $u$ is an element of $O$ (and thus represents value true), then $H(x, y, u)$ says that the head of $M$ is in position $y$ at time $x$, $T_0(x, y, u)$ says that the cell of the tape of $M$ in position $y$ has symbol 0 at time $x$, and likewise for symbols 1 and $B$, and $S_q(x, u)$ says that $M$ is in state $q$ at time $x$.

First, we include the following st-tgds in $\Sigma$:

ACM Transactions on Database Systems, Vol. V, No. N, Month 20YY.
We note that the first st-tgd says that for every pair of elements defined, we only include here the ts-tgd for predicate $H$ satisfying $\mu$; these dependencies state that one is defined for every $x$. Furthermore, for every pair of distinct symbols $x, y$ contains both $\Sigma$. Notice that this ts-tgd says that unless there is a tuple $\{a, a\}$: for $a \in \{0, 1, b\}$, while the last one is defined for every $q \in Q$.

Second, we include the following ts-tgds in $\Sigma'$:

\[
\begin{align*}
H(x, y, z) & \rightarrow \mu(x) \land \mu(y) \\
T_a(x, y, z) & \rightarrow \mu(x) \land \mu(y) \quad \text{for } a \in \{0, 1, b\}, \\
S_q(x, y) & \rightarrow \mu(x) \quad \text{for } q \in Q.
\end{align*}
\]

These dependencies state that $H, T_0, T_1, T_b$ and $S_q (q \in Q)$ are only defined for elements satisfying $\mu$.

Third, we include ts-tgds in $\Sigma'$ saying that predicates $H, T_0, T_1, T_b$ and $S_q (q \in Q)$ are consistent with the equality predicate $E$. Since all these dependencies are similarly defined, we only include here the ts-tgd for predicate $H$:

\[
\begin{align*}
H(x, y, u_1) \land O(u_1) \land E(x, x_1, u_2) \land O(u_2) \land \\
E(y, y_1, u_3) \land O(u_3) \land H(x_1, y_1, u_4) \land Z(u_4) & \rightarrow \exists v D(v, v).
\end{align*}
\]

Notice that this ts-tgd says that unless there is a tuple $(a, a)$ in $D$, if $x$ is equal to $x_1$, $y$, and $z$ is equal to $y_1$ according to $E$, then, according to $H$, the head of $M$ is in position $y$ at time $x$ if and only if the head of $M$ is in position $y_1$ at time $x_1$.

Fourth, to state that each cell of the tape of $M$ contains exactly one symbol at each moment, we include a ts-tgd in $\Sigma'$ saying that it could not be the case that the cell of the tape of $M$ in position $y$ at time $x$ contains neither symbol 0 nor 1 nor $b$ (unless there is a tuple $(a, a)$ in $D$):

\[
T_0(x, y, u_1) \land Z(u_1) \land T_1(x, y, u_2) \land Z(u_2) \land T_b(x, y, u_3) \land Z(u_3) \rightarrow \exists v D(v, v).
\]

Furthermore, for every pair of distinct symbols $a$ and $b$ in $\{0, 1, b\}$, we include a ts-tgd in $\Sigma'$ saying that it could not be the case that the cell of the tape of $M$ in position $y$ at time $x$ contains both $a$ and $b$:

\[
T_a(x, y, u_1) \land O(u_1) \land T_b(x, y, u_2) \land O(u_2) \rightarrow \exists v D(v, v).
\]
Fifth, we include the following ts-tgd in $\Sigma'$ stating that the head of $M$ is in at most one position at each moment:

$$H(x, y_1, u_1) \land O(u_1) \land H(x, y_2, u_2) \land O(u_2) \land E(y_1, y_2, u_3) \land Z(u_3) \rightarrow \exists v D(v, v).$$

We observe that the dependency above together with the dependencies defining transition function $\delta$ (defined below) enforce that the head of $M$ is in exactly one position at each moment. To state that $M$ is in exactly one state at each moment, we include a ts-tgd in $\Sigma'$ saying that it could not be the case that $M$ is not in any state $q \in Q$ at time $x$:

$$\left( \bigwedge_{q \in Q} (S_q(x, u^q)) \land Z(u^q) \right) \rightarrow \exists v D(v, v).$$

Furthermore, for every pair of distinct states $q_1$ and $q_2$ in $Q$, we include a ts-tgd in $\Sigma'$ saying that it could not be the case that $M$ is in both states $q_1$ and $q_2$ (unless there is a tuple $(a, a)$ in $D$):

$$S_{q_1}(x, u_1) \land O(u_1) \land S_{q_2}(x, u_2) \land O(u_2) \rightarrow \exists v D(v, v).$$

Sixth, we include the following ts-tgds in $\Sigma'$ to define the initial configuration: the state of $M$ is $q_0$, the head of $M$ is in the first position of the tape of $M$ and each position of this tape contains the blank symbol.

$$P(x, u_1) \land O(u_1) \land S_{q_0}(x, u_2) \land Z(u_2) \rightarrow \exists v D(v, v),$$

$$P(x, u_1) \land O(u_1) \land H(x, x, u_2) \land Z(u_2) \rightarrow \exists v D(v, v),$$

which says that unless there exists a tuple $(a, a)$ in $D$, if the head of $M$ is not in position $y$ at time $x$, then the symbol in this position is the same at time $u$, where $u$ is the successor of $x$.

Accepting condition for DTM $M$. To conclude the reduction, we include in $\Sigma'$ the following ts-tgd:

$$S_{q_f}(x, y) \land O(y) \rightarrow \exists v D(v, v).$$

This dependency says that if at time $x$ the state of the DTM $M$ is $q_f$ (the final state), then there exists an element $a$ such that $(a, a)$ is in $D$.

We have concluded the definitions of $\Sigma$ and $\Sigma'$. We now proceed to prove that $M$ accepts the empty string if and only if $M'$ is not a recovery of $M$. But before we need to prove some intermediate lemmas.
Lemma C.1. Let $I$ be an instance of $S$. If $\{(a, b) \in D^I \mid a \in T^I \text{ and } b \in T^I\} = \emptyset$, then $(I, I) \in \mathcal{M} \circ \mathcal{M}'$.

Proof. Assume that $\{(a, b) \in D^I \mid a \in T^I \text{ and } b \in T^I\} = \emptyset$. Then $I$ does not satisfy formula $\lambda \equiv \exists x \exists y (D(x, y) \land T(x) \land T(y))$. Thus, given that $\lambda$ is in the left-hand side of every st-tgd in $\Sigma$ except for the axioms $D(x, y) \rightarrow D'(x, y)$ and $T(x) \rightarrow T'(x)$, we conclude that the following instance $J$ of $T$ is a solution for $I$ under $\mathcal{M}$: $D'^J = D^I$, $T'^J = T^I$ and $X^J = \emptyset$, for every $X \in T \setminus \{D', T'\}$. By simply inspecting the set $\Sigma'$, it is possible to conclude that $(J, I) \models \Sigma'$ (since $X^J = \emptyset$ for every $X \in T \setminus \{D', T'\}$, we only need to show that $(J, I)$ satisfies dependencies $D'(x, y) \rightarrow D(x, y)$ and $T'(x) \rightarrow T(x)$, which is clearly the case). Thus, we have that $(I, I) \in \mathcal{M} \circ \mathcal{M}'$.

Lemma C.2. Let $I$ be an instance of $S$. If $\{a \in \text{dom}(D^I) \mid I \models \mu(a)\} = \emptyset$, then $(I, I) \in \mathcal{M} \circ \mathcal{M}'$.

Proof. Analogous to the proof of the Lemma C.2.

Lemma C.3. Let $I$ be an instance of $S$. If there is a tuple $(a, b) \in D^I$, then $(I, I) \in \mathcal{M} \circ \mathcal{M}'$.

Proof. Assume that there is a tuple $(a, b) \in D^I$. Furthermore, assume that for $(a_0, b_0) \in D^I$, we have that $a_0 \in T^I$ and $b_0 \in T^I$ (if there is no such a tuple in $D^I$, then by Lemma C.1 we conclude that $(I, I) \in \mathcal{M} \circ \mathcal{M}'$). Let $J$ be a solution for $I$ such that $D'^J = D^I$, $T'^J = T^I$, $Z^J = \{a_0\}$ and $O^J = \{b_0\}$. Given that there is a tuple $(a, b) \in D^I$, we have that $(J, I)$ satisfies every dependency in $\Sigma'$ that has formula $\exists v D(v, v)$ as its conclusion. Thus, to prove that $(J, I)$ satisfies $\Sigma'$, we only need to show that this instance satisfies ts-tgds: $D'(x, y) \rightarrow D(x, y)$, $T'(x) \rightarrow T(x)$, $Z(x) \rightarrow T(x)$, $O(x) \rightarrow T(x)$ and $Z(x) \land O(y) \rightarrow D(x, y)$, which is clearly the case. We conclude that $(I, I) \in \mathcal{M} \circ \mathcal{M}'$.

We have all the necessary ingredients to prove that $M$ accepts the empty string if and only if $\mathcal{M}'$ is not a recovery of $\mathcal{M}$.

Proof of Theorem 9.4. ($\Rightarrow$) Assume that $M$ accepts the empty string, that is, assume that from the empty tape, $M$ reaches the final state $q_f$ in $k$ steps, where $k \geq 2$ since the initial state of $M$ is different from $q_f$. Then define $I_M$ as the following instance of $S$:

$$D^{I_M} = \{(n, m) \mid n, m \in \{1, \ldots, k\} \text{ and } n \neq m\} \cup \{(f, t)\},$$

$$T^{I_M} = \{f, t\}.$$  

Next we show that $I_M$ is not a solution for $I_M$ under $\mathcal{M} \circ \mathcal{M}'$. For the sake of contradiction, assume $(I_M, I_M) \in \mathcal{M} \circ \mathcal{M}'$. Then there exists an instance $J$ of $T$ such that $(I_M, J) \models \Sigma$ and $(J, I_M) \models \Sigma'$. Since $(I_M, J)$ satisfies $D(x, y) \rightarrow D'(x, y)$ and $(J, I_M)$ satisfies $D'(x, y) \rightarrow D(x, y)$, we conclude that $D'^J = D^{I_M}$, and given that $(I_M, J)$ satisfies $T(x) \rightarrow T'(x)$ and $(J, I_M)$ satisfies $T'(x) \rightarrow T(x)$, we conclude that $T'^J = T^{I_M}$. Moreover, given that $(f, t) \in D^{I_M}$ and $T^{I_M} = \{f, t\}$, we have that $I_M$ satisfies $\lambda$. Thus, given that $(I, J)$ satisfies $\lambda \rightarrow \exists u Z(u) \land \exists v O(v)$ and $(J, I_M)$ satisfies $Z(x) \rightarrow T(x)$, $O(x) \rightarrow T(x)$ and $Z(x) \land O(y) \rightarrow D(x, y)$, we conclude that $Z'^J = \{f\}$ and $O'^J = \{t\}$.

Since $I_M \models \exists v D(v, v)$ and $(J, I_M) \models \Sigma'$, we have that $(J, I_M)$ does not satisfy the left-hand side of any dependency in $\Sigma'$ having $\exists v D(v, v)$ in its right-hand side. Furthermore, given that $k \geq 2$, we have that $\{a \in \text{dom}(D^{I_M}) \mid I_M \models \mu(a)\}$ is not empty. Thus,
by considering the dependencies in Σ and Σ′ that define predicates E, L, P, U, S, H, T_0, T_1, T_0 and S_q (q ∈ Q), we conclude that these predicates encode k steps of the run of M on the empty string. Given that M accepts the empty string, we have that (n, i) ∈ S^I_q for some n ∈ {1, . . . , k} (note that linear order L does not necessarily coincide with the usual linear order 1 < 2 < · · · < k). Thus, given that t ∈ O^I and (J, I^M) satisfies dependency S_q(x, y) ∧ O(y) → ∃v D(v, v), we have that there is a tuple (a, a) in D^I_M, which contradicts the definition of instance I_M.

(⇐) Assume that M does not accept the empty string. Thus, given that we assume that the transition function δ of M is a total function with domain (Q \ {q_f}) × {0, 1, B}, we have that for every k ≥ 1, DTM M reaches some state q ∈ (Q \ {q_f}) in k steps from the initial empty tape. We use this fact to show that M' is a recovery of M.

Let I be an instance of S. We need to show that (I, I) ∈ M ◦ M'. If {(a, b) ∈ D^I | a ∈ T^I and b ∈ T^I} = ∅ or {a ∈ dom(D^I) | I |= µ(a)} = ∅ or there is a tuple (a, a) ∈ D^I, then (I, I) ∈ M ◦ M' by Lemmas C.1, C.2 and C.3. Thus, we assume that I does not satisfy any of these conditions, and without loss of generality, we assume that dom(D^I) = {0, . . . , n} (n ≥ 1), (0, 1) ∈ D^I, 0 ∈ T^I and 1 ∈ T^I.

To prove that I is a solution for I under M ◦ M', we construct a solution J for I under M such that (J, I) |= Σ'. More precisely, we define J as follows: D^J = D^I, T^J = T^I, Z^J = {0}, O^J = {1} and

\[
E^J := \{(i, i, 1) | i ∈ \{0, . . . , n\}\} \cup \{(i, j, 0) | i, j ∈ \{0, . . . , n\} \text{ and } i ≠ j\},
\]

\[
L^J := \{(i, j, 1) | i, j ∈ \{0, . . . , n\} \text{ and } i < j\} \cup \{(i, j, 0) | i, j ∈ \{0, . . . , n\} \text{ and } i ≥ j\},
\]

\[
S^J := \{(i, i + 1, 1) | i ∈ \{0, . . . , n - 1\}\} \cup \{(n, 1, 1) \cup \{(i, j, 0) | i, j ∈ \{0, . . . , n\}, j ≠ i + 1 \text{ and } (i, j) ≠ (n, 1)\},
\]

\[
P^J := \{(0, 1)\} \cup \{(i, 0) | i ∈ \{1, . . . , n\}\},
\]

\[
U^J := \{(n, 1)\} \cup \{(i, 0) | i ∈ \{0, . . . , n - 1\}\}.
\]

Furthermore, relations H^J, T_0^J, T_1^J, T_0^J and S_q^J (q ∈ Q) are defined in such a way that they represent n + 1 steps of the run of M on the empty string. Since M does not accept the empty string, we conclude that (i, 1) /∈ S^J_q for every i ∈ {0, . . . , n} and, therefore, (J, I) trivially satisfies dependency S_q(x, y) ∧ O(y) → ∃v D(v, v) since O^J = {1}.

Furthermore, (J, I) satisfies all the other ts-tgds in Σ' having ∃v D(v, v) in its right-hand side, since H^J, T_0^J, T_1^J, T_0^J and S_q^J (q ∈ Q) code n + 1 steps of the run of M on the empty string. Thus, to prove that (J, I) |= Σ', we only need to show that this instance satisfies dependencies D^J′(x, y) → D(x, y), T^J′(x) → T(x), Z^J′(x) → T(x), O^J′(x) → T(x) and Z^J′(x) ∧ O^J′(y) → D(x, y). It is straightforward to prove that (J, I) satisfies the previous ts-tgds. Thus, we conclude that (J, I) |= Σ' and, hence, (I, I) ∈ M ◦ M'. This concludes the proof of the theorem.

C.4 Proof of Corollary 9.5

Let M = (S, T, Σ) and M' = (T, S, Σ') be the mappings used in the proof of Theorem 9.4. Recall that Σ is a set of st-tgds from S to T and Σ' is a set of ts-tgds from T to S.
where
\[ S := \{ D(\cdot, \cdot), T(\cdot) \}, \]
\[ T := \{ D'(\cdot, \cdot), T'(\cdot), Z(\cdot), O(\cdot), E(\cdot, \cdot, \cdot), L(\cdot, \cdot, \cdot), P(\cdot, \cdot), U(\cdot, \cdot), S(\cdot, \cdot, \cdot), H(\cdot, \cdot, \cdot), \]
\[ T_0(\cdot, \cdot, \cdot), T_1(\cdot, \cdot, \cdot), T_B(\cdot, \cdot, \cdot) \} \cup \{ S_q(\cdot, \cdot) \mid q \in Q \}. \]

Given that \( \Sigma \) includes dependencies \( D(x, y) \rightarrow D'(x, y) \) and \( T(x) \rightarrow T'(x) \), and \( \Sigma' \) includes dependencies \( D'(x, y) \rightarrow D(x, y) \) and \( T'(x) \rightarrow T(x) \), we have that if \((I_1, I_2) \in \mathcal{M} \circ \mathcal{M}' \), then \( I_1 \subseteq I_2 \). Thus, from Lemma 9.2, we have that \( \mathcal{M}' \) is a recovery of \( \mathcal{M} \) iff \( \mathcal{M}' \) is an inverse of \( \mathcal{M} \). Hence, from the proof of Theorem 9.4, we conclude that the problem of verifying, given schema mappings \( \mathcal{M}_1 \) and \( \mathcal{M}_2 \) specified by a set of st-tgds and a set of ts-tgds, respectively, whether \( \mathcal{M}_2 \) is an inverse of \( \mathcal{M}_1 \) is undecidable.

A careful inspection of the set of dependencies \( \Sigma \) reveals that mapping \( \mathcal{M} \) is invertible. In fact, ts-mapping specified by dependencies \( D'(x, y) \rightarrow D(x, y) \) and \( T'(x) \rightarrow T(x) \) is an inverse of \( \mathcal{M} \). Thus, from Theorem 6.3, we conclude that \( \mathcal{M}' \) is a maximum recovery of \( \mathcal{M} \) iff \( \mathcal{M}' \) is an inverse of \( \mathcal{M} \). Therefore, from the proof of Theorem 9.4, we have that the problem of verifying, given schema mappings \( \mathcal{M}_1 \) and \( \mathcal{M}_2 \) specified by a set of st-tgds and a set of ts-tgds, respectively, whether \( \mathcal{M}_2 \) is a maximum recovery of \( \mathcal{M}_1 \) is undecidable. Furthermore, from Proposition 3.24 in [Fagin et al. 2008], we conclude that \( \mathcal{M}' \) is an inverse of \( \mathcal{M} \) iff \( \mathcal{M}' \) is a quasi-inverse of \( \mathcal{M} \). Hence, from the proof of Theorem 9.4, we have that the problem of verifying, given schema mappings \( \mathcal{M}_1 \) and \( \mathcal{M}_2 \) specified by a set of st-tgds and a set of ts-tgds, respectively, whether \( \mathcal{M}_2 \) is a quasi-inverse of \( \mathcal{M}_1 \) is undecidable.