

CWA-Solutions for Data Exchange Settings with Target Dependencies

André Hernich
Institut für Informatik
Humboldt-Universität
Berlin, Germany

hernich@informatik.hu-berlin.de

Nicole Schweikardt
Institut für Informatik
Humboldt-Universität
Berlin, Germany

schweika@informatik.hu-berlin.de

ABSTRACT

Data exchange deals with the following problem: given an instance over a source schema, a specification of the relationship between the source and the target, and dependencies on the target, construct an instance over a target schema that satisfies the given relationships and dependencies. Recently—for data exchange settings without target dependencies—Libkin (PODS’06) introduced a new concept of solutions based on the closed world assumption (so called CWA-solutions), and showed that, in some respects, this new notion behaves better than the standard notion of solutions considered in previous papers on data exchange.

The present paper extends Libkin’s notion of CWA-solutions to data exchange settings with target dependencies. We show that, when restricting attention to data exchange settings with weakly acyclic target dependencies, this new notion behaves similarly as before: the core is the unique “minimal” CWA-solution, and computing CWA-solutions as well as certain answers to positive queries is possible in polynomial time and can be PTIME-hard. However, there may be more than one “maximal” CWA-solution. And going beyond the class of positive queries, we obtain that there are conjunctive queries with (just) one inequality, for which evaluating the certain answers is co-NP-hard. Finally, we consider the EXISTENCE-OF-CWA-SOLUTIONS problem: while the problem is tractable for data exchange settings with weakly acyclic target dependencies, it turns out to be undecidable for general data exchange settings. As a consequence, we obtain that also the EXISTENCE-OF-UNIVERSAL-SOLUTIONS problem is undecidable in general.

Categories and Subject Descriptors

H.2.5 [Heterogeneous Databases]: Data translation;
H.2.4 [Systems]: Relational databases, Rule-based databases, Query processing; D.2.12 [Interoperability]: Data mapping

General Terms

Algorithms, Languages, Theory

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

PODS’07, June 11–14, 2007, Beijing, China.

Copyright 2007 ACM 978-1-59593-685-1/07/0006 ...\$5.00.

Keywords

Data exchange, closed world assumption, the chase, core

1. INTRODUCTION

Data exchange deals with the following problem: given a *data exchange setting* D (i.e., a source schema, a target schema, a specification of the relationship between the source and the target, and dependencies on the target) and an instance S over the source schema, construct an instance T over the target schema that satisfies the given relationships and dependencies. Such a target instance is called a *solution* for S with respect to D . Preferably, in case that solutions exist at all, one would like to find a particular solution that “reflects” the given source data as accurately as possible.

The theoretical foundations of this problem were laid in two seminal papers by Fagin, Kolaitis, Miller, and Popa [6, 7] (see also [10] for a recent survey), where the so called *universal* solutions (i.e., solutions that have homomorphisms to all the other solutions and that, in this sense, are “most general” solutions) were singled out as particularly “good” solutions — and among them, their *core*, being the “smallest” universal solution, was exhibited as one of the “most suitable” solutions.

Since then, questions concerning data exchange have received considerable attention both, for relational data [6, 7, 10, 8, 11, 1, 13, 12, 5] and for XML data [2]. The present paper focuses on relational data.

The complexity of finding solutions for data exchange settings (e.g., [6, 11]) and, in particular, the complexity of computing cores (e.g., [7, 8]) were investigated. While [6, 4] exposed a fairly general class of data exchange settings (the so-called *weakly acyclic* settings), for which solutions can be computed efficiently, it was shown in [11] that, for general settings, the EXISTENCE-OF-SOLUTIONS problem (given a data exchange setting D and a source instance S , decide whether there exists a solution for S w.r.t. D) is undecidable.

Also, the issue of how to answer queries against the target schema was addressed, usually adopting the *certain answers semantics* (i.e., a tuple is regarded to belong to the result of the query if, and only if, it belongs to the query’s result on *every* structure T that is a solution for S w.r.t. D), see e.g. [6, 1, 13]. It was noted, however, in [1] that this kind of certain answers semantics gives rise to some anomalies where the semantics behaves in a counterintuitive way, i.e., queries produce results that intuitively do not seem to be accurate.

To overcome these anomalies, Libkin [12] recently introduced—for data exchange settings without target dependencies—a new concept of solutions based on the closed world assumption (so called CWA-solutions), and showed that, in some respects, this new notion behaves better than the standard notion of solutions considered in previous papers on data exchange.

The present paper pursues Libkin’s approach of CWA-solutions and addresses data exchange settings with target dependencies. Our main contributions are as follows:

1. We develop a new notion of CWA-solutions for data exchange settings with target dependencies. This notion is based on Libkin’s concept of “justifications for introducing nulls” and, additionally, it uses a suitably “controlled” version of the well-known *chase* procedure. We show (Theorem 4.8) that these CWA-solutions are particular *universal solutions* which fit to the closed world assumption in the following sense (cf. Libkin’s [12] requirements for CWA-solutions): 1. Each null is justified by the source instance and the dependencies of the data exchange setting, and 2. nulls are not “overused”.

2. Exploring the space of all CWA-solutions, we obtain the following: CWA-solutions exist if, and only if, universal solutions exist; and the *core* of the universal solutions always is a CWA-solution (Theorem 5.1 and Corollary 5.2). Thus, if CWA-solutions exist, then the *core* is the unique “minimal” CWA-solution.

Concerning “maximal” CWA-solutions, we identify restricted kinds of data exchange settings, where a unique “maximal” CWA-solution is guaranteed to exist (Proposition 5.4). For general settings, however, “maximal” CWA-solutions may not exist and, in fact, already for very simple data exchange settings there may be exponentially many “incomparable” CWA-solutions (see Example 5.3).

3. Addressing the problem of how to compute a CWA-solution when given a data exchange setting and a source instance, we obtain that the EXISTENCE-OF-CWA-SOLUTIONS problem (given a data exchange setting D and a source instance S , decide whether there exists a CWA-solution for S w.r.t. D) is undecidable (Theorem 6.2). As a consequence, also the EXISTENCE-OF-UNIVERSAL-SOLUTIONS problem is undecidable (Corollary 6.4). When restricting attention to *weakly acyclic* settings, however, known tractability results for universal solutions carry over to CWA-solutions. In particular, for such settings the EXISTENCE-OF-CWA-SOLUTIONS problem can be solved (and CWA-solutions can be computed) with polynomial time data complexity, and for some settings, this problem is PTIME-hard (Proposition 6.6).

4. Finally, we consider the problem of query answering with respect to the four different semantics of *certain answers* and *maybe answers* introduced in [12]. We identify restricted kinds of data exchange settings with target dependencies to which Libkin’s [12] characterizations of the four semantics via the *core* and the *canonical solution* carry over (Theorem 7.1).

Considering *weakly acyclic* settings, we show that computing the certain answers of *unions of conjunctive queries* is possible with polynomial time data complexity and can be PTIME-hard (Theorem 7.6 and Proposition 7.8).

Going beyond unions of conjunctive queries, we obtain that the certain (resp., maybe) answers of (Boolean) *first-order queries* has co-NP (resp., NP) data complexity, provided that the underlying data exchange setting is *richly acyclic* (an acyclicity notion that is slightly more restrictive than the usual notion of *weak acyclicity*) (Proposition 7.4). Furthermore, there already exist conjunctive queries with just one inequality, for which evaluating the certain answers is co-NP-hard (Theorem 7.5).

The paper is structured as follows: Section 2 fixes the basic notations on data exchange that are used throughout the paper. Section 3 indicates some of the anomalies that arise from the usual certain answers semantics. The new concept of CWA-solutions for data exchange settings with target dependencies is formally introduced and illustrated with examples in Section 4. Some basic results concerning CWA-solutions are shown in Section 5. Section 6 addresses the complexity of computing CWA-solutions, whereas Section 7 deals with the query answering problem.

Due to space limitations, most of the technical details of our proofs had to be deferred to the full version of the paper.

2. PRELIMINARIES

In this section, we fix the standard notations on data exchange that are used throughout the paper.

A *schema* σ is a finite set of relation symbols, each with a fixed arity. An *instance* I over σ is represented by a finite set of *atoms*, i.e., expressions of the form $R(\bar{u})$, where R is a relation symbol in σ of some arity, say r , and \bar{u} is an r -tuple over a fixed set Dom of *values*. The *active domain* $\text{Dom}(I)$ of I is the set of values that occur in the atoms of I .

We allow instances to contain incomplete data, which is represented by (*labeled*) *nulls*. More precisely, we assume that Dom is the union of a countably infinite set Const of *constants* and a countably infinite set Null of nulls (which serve as placeholders for unknown data), disjoint from Const . Constants are denoted by lowercase letters a, b, c, \dots , and nulls by \perp , possibly with subscripts and/or superscripts. For an instance I , we let $\text{Const}(I) := \text{Dom}(I) \cap \text{Const}$ and $\text{Null}(I) := \text{Dom}(I) \cap \text{Null}$.

Let σ be a schema. A *dependency over* σ is a sentence in some logical formalism (typically first-order logic) over the vocabulary $\underline{\sigma} := \sigma \cup \{c \mid c \in \text{Const}\}$. It is satisfied in an instance I over σ if and only if it is satisfied in the $\underline{\sigma}$ -structure \underline{I} with universe $\text{Dom}(I)$, where each $R \in \sigma$ is interpreted by the relation $\{\bar{u} \mid R(\bar{u}) \in I\}$, and for each $c \in \text{Const}$, c is interpreted by c . In general, a formula¹ $\varphi(x_1, x_2, \dots, x_n)$ over σ is a formula over $\underline{\sigma}$, and for an assignment α mapping x_i to u_i for every $i \in \{1, 2, \dots, n\}$, we say that φ holds in I under α , written $I \models \varphi[u_1, u_2, \dots, u_n]$, if and only if φ holds in \underline{I} under α .

A *data exchange setting* [6, 7, 10] is a quadruple $D = (\sigma, \tau, \Sigma_{\text{st}}, \Sigma_{\text{t}})$, where σ and τ are disjoint schemas, called the *source schema* and the *target schema*, respectively, Σ_{st} is a set of dependencies between σ and τ , called *source-to-target dependencies*, and Σ_{t} is a set of dependencies over τ , called *target dependencies*. As in [6, 7, 10], we consider only data exchange settings where Σ_{st} is a finite set of *source-to-target tuple generating dependencies* (s-t-tgds, for short), and Σ_{t}

¹For a formula φ , $\varphi(x_1, x_2, \dots, x_n)$ indicates that the free variables of φ are precisely the variables x_1, x_2, \dots, x_n .

is a finite set of *target tuple generating dependencies* (target tgds, for short) and *equality generating dependencies* (egds, for short). S-t-tgds are first-order formulas of the form

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

where φ is a first-order formula over σ , and ψ is a conjunction of relational atomic formulas over τ .² Target tgds are formulas of the same form, except that φ must be a conjunction of relational atomic formulas over τ . A *tgd* is either an s-t-tgd or a target tgd. Egds are formulas of the form

$$\forall \bar{x} (\varphi(\bar{x}) \rightarrow y = z),$$

where φ is a conjunction of relational atomic formulas over τ , and y, z are variables in \bar{x} . In the following, we often omit the universal quantifiers in front of tgds and egds.

A *source instance* for D is an instance over σ which contains *only constants*. A *target instance* for D is an instance over τ , possibly with nulls. Given a source instance S for D , a *solution* for S under D is a target instance T for D such that $S \cup T$ satisfies all tgds in Σ_{st} , written $S \cup T \models \Sigma_{\text{st}}$, and T satisfies all tgds and egds in Σ_{t} , written $T \models \Sigma_{\text{t}}$.

EXAMPLE 2.1. Let $D^* := (\sigma, \tau, \Sigma_{\text{st}}^*, \Sigma_{\text{t}}^*)$ be the data exchange setting with $\sigma = \{M, N\}$, $\tau = \{E, F, G\}$ and $\Sigma^* := \Sigma_{\text{st}}^* \cup \Sigma_{\text{t}}^* = \{d_1, d_2, d_3, d_4\}$, where

$$\begin{aligned} d_1 &= M(x_1, x_2) \rightarrow E(x_1, x_2), \\ d_2 &= N(x, y) \rightarrow \exists z_1, z_2 (E(x, z_1) \wedge F(x, z_2)), \\ d_3 &= F(y, x) \rightarrow \exists z G(x, z), \text{ and} \\ d_4 &= F(x, y) \wedge F(x, z) \rightarrow y = z. \end{aligned}$$

The source instance $S^* := \{M(a, b), N(a, b), N(a, c)\}$ has among others the following solutions under D^* :

$$\begin{aligned} T_1^* &:= \{E(a, b), E(a, \perp_1), E(c, \perp_2), F(a, d), G(d, \perp_3)\}, \\ T_2^* &:= \{E(a, b), E(a, \perp_1), E(a, \perp_2), F(a, \perp_3), G(\perp_3, \perp_4)\}, \\ T_3^* &:= \{E(a, b), F(a, \perp_1), G(\perp_1, \perp_2)\}. \end{aligned}$$

Recall that a, b, c, \dots denote constants and $\perp, \perp_1, \perp_2, \dots$ nulls, and that nulls serve as placeholders for unknown data. Unknown data arises, because the dependencies in Σ^* do not fully specify the solutions for S^* under D^* . For example, d_2 only says that for $N(a, b)$ there must be a u with $F(a, u)$, but not which u . We have the choice to introduce a null as a placeholder for u , as in T_2^* and T_3^* , or to “invent” a constant from which we think that it might be u , as in T_1^* .

Fagin et al. [6] defined the notion of *universal solutions*, which are “most general” solutions. A solution T for S under D is *universal* if for every solution T' for S under D there is a homomorphism from T to T' . Here, a *homomorphism* from an instance I to an instance J is a mapping $h: \text{Dom}(I) \rightarrow \text{Dom}(J)$ such that for each $R(u_1, u_2, \dots, u_r) \in I$ we have $R(h(u_1), h(u_2), \dots, h(u_r)) \in J$, and for each $c \in \text{Const}(I)$ we have $h(c) = c$.³ It can be shown that in Example 2.1, T_2^*

²This definition of s-t-tgds is from [12]. In other papers on data exchange, φ usually is a conjunction of relational atomic formulas over σ . For technical reasons, when considering *s-t-tgds* we will always assume that every first-order quantifier in $\forall \bar{x} \forall \bar{y} \varphi(\bar{x}, \bar{y})$ is relativized to the *active domain with respect to* σ .

³Note that we use [6, 7]’s notion of homomorphisms, where nulls may be mapped to nulls or to constants. A slightly

and T_3^* are universal solutions for S^* under D^* , but T_1^* is no universal solution for S^* under D^* , because there is no homomorphism from T_1^* to T_2^* .

A *core* of an instance I is an instance $J \subseteq I$ such that there is a homomorphism from I to J , but no homomorphism from J to an instance $K \subsetneq J$. As shown in [9], every finite instance has (up to renaming of nulls) a unique core. In [7] it has also been shown that, if universal solutions for S under D exist, then there is a universal solution for S under D , denoted by $\text{CORE}_D(S)$, that is (up to renaming of nulls) equivalent to the cores of every universal solution for S under D . In Example 2.1, $\text{CORE}_{D^*}(S^*)$ is (up to renaming of nulls) equal to T_3^* .

In [6], the *certain answers semantics* has been adopted for the semantics for query answering in data exchange. Given a data exchange setting D , a source instance S for D , and a query Q over the target schema of D , the set of the *certain answers of* Q *with respect to* S , denoted by $\text{certain}^D(Q, S)$, consists of all tuples that appear in $Q(T)$ for all solutions T for S under D (i.e., all tuples that are *certainly* answers to Q). In [7], the alternative *certain universal answers semantics* has been proposed: the set of the *certain answers of* Q *on universal solutions with respect to* S , denoted by $\text{u-certain}^D(Q, S)$, consists of all tuples that appear in $Q(T)$ for all *universal* solutions T for S under D . Note that $\text{certain}^D(Q, S) \subseteq \text{u-certain}^D(Q, S)$.

3. PROBLEMS WITH THE CERTAIN (UNIVERSAL) ANSWERS SEMANTICS

It has been observed in [1] that the certain answers semantics of [6] and the certain universal answers semantics of [7] give rise to several anomalies.

Let us review one of these anomalies in an example that is basically taken from [1, 12], and that considers the following kind of *copying* data exchange settings: A data exchange setting D is *copying* if D is of the form $(\sigma, \tau, \Sigma_{\text{st}}, \emptyset)$, where $\tau = \{R' \mid R \in \sigma\}$, and $\Sigma_{\text{st}} = \{R(\bar{x}) \rightarrow R'(\bar{x}) \mid R \in \sigma\}$. I.e., under a copying data exchange setting, a source instance S is just “copied” to the target. It is therefore natural to expect from a “reasonable” query answering semantics that the answers to a query Q over the target schema can be obtained by evaluating Q on the instance S' obtained from S by renaming each $R \in \sigma$ into the corresponding element $R' \in \tau$. But this is not necessarily the case for the *certain answers semantics*. Consider, for example, the copying data exchange setting D with source schema $\{E, P\}$, and the first-order query

$$Q(x) := P'(x) \vee \exists y \exists z (P'(y) \wedge E'(y, z) \wedge \neg P'(z)).$$

Now consider the source instance S that consists of the disjoint union of two cycles of length 9, built over the nodes a_0, \dots, a_8 and the nodes b_0, \dots, b_8 , respectively, and where a_4 is the unique element that is labeled P . I.e.,

$$\begin{aligned} S &:= \{E(a_i, a_j) \mid 0 \leq i \leq 8, j = i + 1 \bmod 9\} \\ &\cup \{E(b_i, b_j) \mid 0 \leq i \leq 8, j = i + 1 \bmod 9\} \cup \{P(a_4)\}. \end{aligned}$$

Intuitively, one would expect that the “best” solution for S with respect to the copying data exchange setting D is simply the instance S' obtained from S by renaming E with different notion which, however, leads to basically the same results, was considered in [12], where homomorphisms have to map nulls to nulls.

E' and P with P' . One can easily see that the answer of query Q on this particular target instance S' consists of all nodes of S' , i.e., $Q(S') = \{a_0, \dots, a_8, b_0, \dots, b_8\}$. However, the *certain answers* semantics defined at the end of Section 2 produces only the (somewhat counterintuitive) result $\text{certain}^D(Q, S) = \{a_0, \dots, a_8\}$ (to see why, note that the target instance S'' , obtained from S' by adding the facts $P'(a_i)$ for all i , is a solution for S under D).

The *certain universal answers* semantics behaves better on such copying data exchange settings, but gives rise to a similar anomaly on data exchange settings $(\sigma, \tau \cup \{D\}, \Sigma_{\text{st}} \cup \Sigma'_{\text{st}}, \emptyset)$, where $(\sigma, \tau, \Sigma_{\text{st}}, \emptyset)$ is copying, and Σ'_{st} consists of s-t-tgds of the form $R(x_1, x_2, \dots, x_r) \rightarrow D(x_i)$ for every $R \in \sigma$ of arity r , and $i \in \{1, 2, \dots, r\}$. Details on this and also on some other anomalies can be found in [12, 1].

To overcome such anomalies, Libkin [12] introduced the concept of CWA-solutions and alternative query answering semantics. Note, however, that in [12] the concept of CWA-solutions has been defined only for data exchange settings *without* target dependencies. To point out why it is necessary to find a suitable *extension* of this notion for the case where target dependencies are present, let us return to Example 2.1: The only CWA-solutions for S^* in the sense of [12] (under the data exchange setting obtained from D^* by removing the target dependencies) are, up to renaming of nulls, the following three target instances: $\{E(a, b), F(a, \perp_1)\}$, $\{E(a, b), E(a, \perp_1), F(a, \perp_2)\}$ as well as $\{E(a, b), E(a, \perp_1), E(a, \perp_2), F(a, \perp_3)\}$. But these are no solutions for S^* under D^* !

4. CWA-SOLUTIONS IN THE PRESENCE OF TARGET DEPENDENCIES

This section formally introduces the new concept of CWA-solutions for data exchange settings with target dependencies, and illustrates it with examples.

Throughout this section, let $D = (\sigma, \tau, \Sigma_{\text{st}}, \Sigma_t)$ be a data exchange setting and S a source instance for D . Let $\rho := \sigma \cup \tau$ and $\Sigma := \Sigma_{\text{st}} \cup \Sigma_t$.

The concept of CWA-solutions is based on the *closed world assumption* (CWA, for short). The main idea is that each fact that is true in a CWA-solution is directly justified by the atoms of S and the dependencies in Σ . Informally, a good CWA-solution T for S under D should satisfy the following three requirements, adapted from [12]:

- (CWA1) Each atom of T must be justified by the atoms of S and the dependencies in Σ .
- (CWA2) No justification for producing a value through an existentially quantified variable of a tgd generates *multiple* values.
- (CWA3) Each fact that is true in T must *follow* from the atoms of S and the dependencies in Σ .

In the following, we first address requirements CWA1 and CWA2, and afterwards, CWA3.

Let us quickly recall the concept of [12] for the case where $\Sigma_t = \emptyset$. The requirements CWA1 and CWA2 are formalized by *CWA-presolutions*, which in turn are based on the concept of *justifications*. A *justification* for producing a value through an existentially quantified variable of an s-t-tgd d in Σ_{st} consists of d , which has the form $\varphi(\bar{x}, \bar{y}) \rightarrow \exists \bar{z} \psi(\bar{x}, \bar{z})$, of tuples (\bar{u}, \bar{v}) with $S \models \varphi[\bar{u}, \bar{v}]$, and a variable z in \bar{z} (to

which the value can be assigned); it is represented by the quadruple (d, \bar{u}, \bar{v}, z) . A *CWA-presolution* for S under D is then obtained as follows.

1. Choose a mapping α from the set of all justifications to Dom .
2. Start with an empty target instance. For each s-t-tgd d in Σ_{st} , where

$$d = \varphi(\bar{x}, \bar{y}) \rightarrow \exists \bar{z} \psi(\bar{x}, \bar{z}) \quad \text{and} \quad \bar{z} = (z_1, z_2, \dots, z_n),$$

and for each pair of tuples \bar{u} and \bar{v} with $S \models \varphi[\bar{u}, \bar{v}]$, let $\bar{w} := (w_1, w_2, \dots, w_n)$, where

$$w_i := \alpha(d, \bar{u}, \bar{v}, z_i)$$

and add the atoms of $\psi[\bar{u}, \bar{w}]$ (i.e., the smallest set A of atoms such that $A \models \psi[\bar{u}, \bar{w}]$) to the target instance.

So, atoms are justified since they can be derived from the source instance via the s-t-tgds, and CWA2 is satisfied by letting each justification generate at most one value.

We extend the notion of CWA-presolutions to the case $\Sigma_t \neq \emptyset$ by relaxing the definition of justifications, and using a suitably “controlled” version of the well-known chase procedure, called α -chase. A *potential justification* for introducing a value through an existentially quantified variable of a tgd in Σ is represented by a quadruple (d, \bar{u}, \bar{v}, z) , where d is a tgd in Σ of the form $\varphi(\bar{x}, \bar{y}) \rightarrow \exists \bar{z} \psi(\bar{x}, \bar{z})$, \bar{u} and \bar{v} are tuples over $\text{Dom} = \text{Null} \cup \text{Const}$ of the same length as \bar{x} and \bar{y} , respectively, and z is a variable in \bar{z} . Let \mathcal{J}_D be the set of all such potential justifications, i.e.,

$$\mathcal{J}_D := \{(d, \bar{u}, \bar{v}, z) \mid d = \varphi(\bar{x}, \bar{y}) \rightarrow \exists \bar{z} \psi(\bar{x}, \bar{z}) \text{ is a tgd in } \Sigma, \\ \bar{u} \in \text{Dom}^{|\bar{x}|}, \bar{v} \in \text{Dom}^{|\bar{y}|}, z \text{ occurs in } \bar{z}\}.$$

For a mapping $\alpha: \mathcal{J}_D \rightarrow \text{Dom}$, define the mapping $\bar{\alpha}$ as follows: if d, \bar{u}, \bar{v} are as above, and $\bar{z} = (z_1, z_2, \dots, z_n)$, then

$$\bar{\alpha}(d, \bar{u}, \bar{v}) := (\alpha(d, \bar{u}, \bar{v}, z_1), \dots, \alpha(d, \bar{u}, \bar{v}, z_n)).$$

DEFINITION 4.1. (α -chase) Let $\alpha: \mathcal{J}_D \rightarrow \text{Dom}$, and let I be an instance over ρ . An α -chase of I with Σ is a (finite or infinite) sequence I_0, I_1, I_2, \dots of instances over ρ such that $I_0 = I$, and each I_{i+1} is the result of α -applying a tgd in Σ , or applying an egd in Σ , to I_i in the following sense:

- α -application of a tgd d in Σ : Suppose that d has the form $\varphi(\bar{x}, \bar{y}) \rightarrow \exists \bar{z} \psi(\bar{x}, \bar{z})$, and that \bar{u}, \bar{v} are tuples such that

$$I_i \models \varphi[\bar{u}, \bar{v}] \quad \text{and} \quad I_i \not\models \psi[\bar{u}, \bar{\alpha}(d, \bar{u}, \bar{v})]. \quad (1)$$

Then we say that d can be α -applied to I_i with \bar{u} and \bar{v} . The *result* of this application is $I_i \cup J$, where J is the set of atoms occurring in $\psi[\bar{u}, \bar{\alpha}(d, \bar{u}, \bar{v})]$.

- application of an egd d in Σ : Suppose that d has the form $\varphi(x_1, x_2, \dots, x_n) \rightarrow x_k = x_l$, and that $\bar{u} = (u_1, u_2, \dots, u_n)$ is a tuple such that

$$I_i \models \varphi[\bar{u}] \quad \text{and} \quad u_k \neq u_l.$$

Then we say that d can be applied to I_i with \bar{u} . The application *fails* if u_k and u_l are constants. Otherwise

it *succeeds*, and the *result* is obtained from I_i by replacing all occurrences of one of the nulls in $\{u_k, u_l\}$ by the other value in $\{u_k, u_l\}$.⁴

The *result* of a finite α -chase I_0, \dots, I_m of S with Σ is I_m .

DEFINITION 4.2. (successful α -chase, failing α -chase) Let $\alpha: \mathcal{J}_D \rightarrow \text{Dom}$, and let I be an instance over ρ . Let C be an α -chase of I with Σ .

- (1) C is called *successful* if
 - (a) C is finite,
 - (b) the result of C satisfies Σ , and
 - (c) no tgd in Σ can be α -applied to the result of C .
- (2) C is called *failing* if
 - (a) C is finite, and
 - (b) there is an egd d in Σ such that d can be applied to the result J of C , and the application of d to J fails.

REMARK 4.3. The reader who is familiar with the standard chase as considered, e.g., in [6, 7, 8], might wonder why we did not use a condition like

$$I_i \models \varphi[\bar{u}, \bar{v}] \quad \text{and} \quad I_i \not\models \psi[\bar{u}, \bar{w}] \quad \text{for all } \bar{w} \in \text{Dom}^{|\bar{z}|} \quad (2)$$

instead of (1) in Definition 4.1. The reason is that α -chases are not primarily intended to *compute* solutions—where (2) would make sense—but to formalize the notion of a “reasonable” solution with respect to the closed world assumption. Condition (2) would not suffice to formalize such a notion (see Example 4.4). We should also point out that without requirement (c) in Definition 4.2 (1), it would be possible to generate an instance that does not adhere to requirement CWA2 (see Example 4.4).

EXAMPLE 4.4. Consider the data exchange setting D^* and source instance S^* from Example 2.1. For $i \in \{1, 2, 3\}$, let $\alpha_i: \mathcal{J}_{D^*} \rightarrow \text{Dom}$ as shown in the following table (* indicates that the value can be arbitrary):

$j \in \mathcal{J}_{D^*}$	$\alpha_1(j)$	$\alpha_2(j)$	$\alpha_3(j)$
(d_2, a, b, z_1)	\perp_1	b	b
(d_2, a, b, z_2)	\perp_3	c	\perp_3
(d_2, a, c, z_1)	\perp_2	b	b
(d_2, a, c, z_2)	\perp_3	d	\perp_4
(d_3, \perp_3, a, z)	\perp_4	*	\perp_1
(d_3, \perp_4, a, z)	*	*	\perp_2

Then the sequence $C = (I_0, I_1, I_2, I_3, I_4)$ with

$$\begin{aligned} I_0 &= S^* & I_1 &= I_0 \cup \{E(a, b)\} \\ I_2 &= I_1 \cup \{E(a, \perp_1), F(a, \perp_3)\} & I_3 &= I_2 \cup \{E(a, \perp_2)\} \\ I_4 &= I_3 \cup \{G(\perp_3, \perp_4)\} \end{aligned}$$

is an α_1 -chase of S^* with Σ^* : first α_1 -apply d_1 with (a, b) and the empty tuple, afterwards d_2 with a and b , followed by d_2 with a and c , and finally d_3 with \perp_3 and a . The result of C is I_4 . Since no dependency in Σ^* can be (α_1 -)applied to I_4 , C

⁴To make the result of applying an egd unambiguous, we assume that Null is linearly ordered so that if both u_k and u_l are nulls, the larger null is replaced by the smaller one.

is successful. Note that I_4 intuitively satisfies requirements CWA1 and CWA2, but could not be obtained by a successful α -chase of S^* with Σ^* , for any $\alpha: \mathcal{J}_{D^*} \rightarrow \text{Dom}$, if condition (1) in Definition 4.1 was replaced by (2) (recall Remark 4.3).

An example of a failing α_2 -chase of S^* with Σ^* is the sequence $C' = (I'_0, I'_1, I'_2, I'_3)$, where

$$\begin{aligned} I'_0 &= S^* & I'_1 &= I'_0 \cup \{E(a, b)\} \\ I'_2 &= I'_1 \cup \{F(a, c)\} & I'_3 &= I'_2 \cup \{F(a, d)\}. \end{aligned}$$

The egd d_4 can be applied to I'_3 with (a, c, d) , but this application fails, because the two constants c and d cannot be identified.

An example of a finite α_3 -chase of S^* with Σ^* that is neither successful nor failing is $C'' = (I''_0, I''_1, I''_2, I''_3, I''_4, I''_5, I''_6)$, where

$$\begin{aligned} I''_0 &= S^*, & I''_1 &= I''_0 \cup \{E(a, b)\} \\ I''_2 &= I''_1 \cup \{F(a, \perp_3)\} & I''_3 &= I''_2 \cup \{F(a, \perp_4)\} \\ I''_4 &= I''_3 \cup \{G(\perp_3, \perp_1)\} & I''_5 &= I''_4 \cup \{G(\perp_4, \perp_2)\}, \end{aligned}$$

and

$$I''_6 = S^* \cup \{E(a, b), F(a, \perp_3), G(\perp_3, \perp_1), G(\perp_3, \perp_2)\}$$

is obtained from I''_5 by applying the egd d_4 with (a, \perp_3, \perp_4) . Used to the standard chase, one is inclined to think that C'' is successful, because $I''_6 \models \Sigma^*$. But one look at I''_6 tells us that C'' should not be successful: I''_6 intuitively does not satisfy requirement CWA2 (\perp_1 and \perp_2 are both justified by (d_3, \perp_3, a, z) only). Indeed, according to requirement (c) of Definition 4.2 (1), C'' is *not* successful: d_2 can be α_3 -applied with a and c . Now, α_3 -applying d_2 with a and c yields the atom $F(a, \perp_4)$, which enables us to apply d_4 with (a, \perp_3, \perp_4) once again, which in turn enables us to α_3 -apply d_2 with a and c , and so on. It is easy to see that C'' cannot be extended to a successful α_3 -chase of S^* with Σ^* : it will have to loop forever! This is basically what is exploited in the proof of Lemma 4.5 below.

LEMMA 4.5. Let $\alpha: \mathcal{J}_D \rightarrow \text{Dom}$, and let I be an instance over ρ that contains no nulls. Then a successful α -chase of I with Σ exists if and only if there is no failing or infinite α -chase of I with Σ . Moreover, every successful α -chase of I with Σ (α -)applies only tgds and has the same result as all other α -chases of I with Σ .

DEFINITION 4.6. (CWA-presolution) A *CWA-presolution* for S under D is a target instance T for D such that for some mapping $\alpha: \mathcal{J}_D \rightarrow \text{Dom}$, $S \cup T$ is the result of a successful α -chase of S with Σ .

Clearly, a CWA-presolution for S under D is a solution for S under D and satisfies requirements CWA1 and CWA2. However, CWA3 is in general not satisfied. In Example 2.1, one possible CWA-presolution for S^* under D^* is $\{E(a, b), F(a, \perp), G(\perp, b)\}$, but the fact that a and b are connected by an F - G -path of length two does intuitively not follow from S^* and Σ^* . Such facts should be prohibited.

To make this precise, we follow [12] and represent a *fact* over the target schema τ of D by a first-order sentence φ over τ of the form $\exists \bar{x} \psi(\bar{x})$, where ψ is a conjunction of relational atomic formulas. φ is true in a target instance T for D if $T \models \varphi$. For example, the fact “ a and b are connected by an F - G -path of length two” can be represented by $\exists x (F(a, x) \wedge G(x, b))$. For a fact φ we say that

φ follows from S and Σ

if and only if φ is true in every instance I over ρ with $I|_\sigma = S$ and $I \models \Sigma$. Here, $I|_\sigma$ denotes the σ -reduct of I , i.e., the instance $\{R(\bar{u}) \in I \mid R \in \sigma\}$.

DEFINITION 4.7. (CWA-solution) A *CWA-solution* for S under D is a CWA-presolution for S under D such that every fact that is true in T follows from S and Σ . We denote the set of all CWA-solutions for S under D by $\llbracket S \rrbracket_{\text{CWA}}^D$.

A target instance T for D can be represented in the natural way by its

$$\text{“canonical fact” } \varphi_T = \exists \bar{x} \psi(\bar{x}),$$

which is defined as follows: for each $\perp \in \text{Null}(T)$, \bar{x} contains a variable x_\perp , and for each $R(\bar{u}) \in T$, ψ contains the conjunct $R(\bar{t})$, where \bar{t} is obtained from \bar{u} by replacing all occurrences of a null \perp by x_\perp . Then one can easily see that

$$\begin{aligned} \varphi_T \text{ follows from } S \text{ and } \Sigma \\ \text{if and only if} \end{aligned}$$

every fact that is true in T follows from S and Σ .

Since for instances I and J , $I \models \varphi_J$ is equivalent to the existence of a homomorphism from J to I [3], we immediately obtain the following connection between CWA-solutions and universal solutions:

THEOREM 4.8. *For every target instance T for D , the following are equivalent:*

1. T is a CWA-solution for S under D .
2. T is a universal solution for S under D , and T is a CWA-presolution for S under D .

We remark that for data exchange settings *without* target dependencies, our notions of CWA-presolution and CWA-solution coincide with the corresponding notions in [12].

EXAMPLE 4.9. Consider again Example 2.1. The target instance T_2^* is a CWA-solution for S^* under D^* : $S^* \cup T_2^*$ is the result of the successful α_1 -chase C of S^* with Σ^* from Example 4.4, and it can be shown that T_2^* is a universal solution for S^* under D^* . On the other hand, although the target instance

$$T := \{E(a, b), F(a, \perp), G(\perp, b)\}$$

is a CWA-presolution for S^* under D^* , it is no CWA-solution for S^* under D^* : there is no homomorphism from T to T_2^* , or equivalently, the fact $\exists x (F(a, x) \wedge G(x, b))$ does not follow from S^* and Σ^* . Finally, the target instance

$$T' := \{E(a, b), E(\perp_3, b), F(b, \perp_1), G(\perp_1, \perp_2)\}$$

is a universal solution for S^* under D^* , but since $S^* \cup T'$ cannot be obtained by a successful α -chase of S^* with Σ^* (the atom $E(\perp_3, b)$ is not justified), it is no CWA-solution for S^* under D^* .

5. BASIC RESULTS ON CWA-SOLUTIONS

In this section, we prove basic results concerning CWA-solutions, which are essential for Sections 6 and 7.

Given a data exchange setting D and a source instance S for D , a CWA-solution T for S under D is called *minimal* if T is contained (up to renaming of nulls) in every

CWA-solution for S under D ; T is called *maximal* if every CWA-solution for S under D is a homomorphic image of T . Minimal and maximal CWA-solutions play an important role in query answering (see Section 7).

The following theorem generalizes the analogous result for data exchange settings without target dependencies in [12]:

THEOREM 5.1. *Let D be a data exchange setting and S a source instance for D . If $\text{CORE}_D(S)$ exists, then $\text{CORE}_D(S)$ is a CWA-solution for S under D . In particular, $\text{CORE}_D(S)$ is a minimal CWA-solution for S under D .*

Note that Theorem 5.1 and Theorem 4.8 immediately lead to

COROLLARY 5.2. *For every data exchange setting D and every source instance S for D , the following are equivalent:*

1. There exists a CWA-solution for S under D .
2. There exists a universal solution for S under D .
3. $\text{CORE}_D(S)$ exists.

In contrast to Theorem 5.1, which implies that the core of the universal solutions is a minimal CWA-solution (provided that CWA-solutions exist at all), maximal CWA-solutions may not exist in general. The following example shows that even for very simple data exchange settings there may be exponentially many CWA-solutions, each of which is (up to renaming of nulls) no homomorphic image of another CWA-solution.

EXAMPLE 5.3. Let D be the data exchange setting with source schema $\{P\}$, target schema $\{E, F\}$, and dependencies

$$\begin{aligned} d_1 &= P(x) \rightarrow \exists z_1, z_2, z_3, z_4 (E(x, z_1, z_3) \wedge E(x, z_2, z_4)), \\ d_2 &= E(x, x_1, y) \wedge E(x, x_2, y) \rightarrow F(x, x_1, x_2). \end{aligned}$$

Then the source instance $S := \{P(1)\}$ has among others the following CWA-solutions under D :

$$\begin{aligned} T &:= \{E(1, \perp_1, \perp_3), E(1, \perp_2, \perp_4), F(1, \perp_1, \perp_1), \\ &\quad F(1, \perp_2, \perp_2)\}, \\ T' &:= \{E(1, \perp_1, \perp_3), E(1, \perp_2, \perp_3), F(1, \perp_1, \perp_1), \\ &\quad F(1, \perp_2, \perp_2), F(1, \perp_1, \perp_2), F(1, \perp_2, \perp_1)\}. \end{aligned}$$

Note that T is (up to renaming of nulls) no homomorphic image of another CWA-solution for S . The same applies to T' . Building on this observation, one can show that for the source instance $S_n = \{P(i) \mid 1 \leq i \leq n\}$ there are at least 2^n CWA-solutions under D with this property. Details can be found in the full version of the paper.

Nevertheless, we can show that for certain restricted data exchange settings D , a *maximal* CWA-solution for S under D , denoted by $\text{CANSOL}_D(S)$, does exist. One such restriction considers *full tgds*, i.e., tgds whose right-hand sides contain no existential quantifiers.

PROPOSITION 5.4. *Consider a data exchange setting D whose target dependencies consist of egds, or whose source-to-target dependencies and target dependencies consist of egds and full tgds. Then for every source instance S for D , every CWA-solution for S under D is a homomorphic image of $\text{CANSOL}_D(S)$.*

Details on the definition of $\text{CANSOL}_D(S)$ as well as the proof of Proposition 5.4 can be found in the full version of the paper.

6. THE COMPLEXITY OF COMPUTING CWA-SOLUTIONS

In this section, we consider the following problem: given a fixed data exchange setting D and a source instance S for D , find a CWA-solution for S under D . The decision version of this problem is called $\text{EXISTENCE-OF-CWA-SOLUTIONS}(D)$: given a source instance S for D , decide whether there is a CWA-solution for S under D .

Considering the analogous $\text{EXISTENCE-OF-SOLUTIONS}$ problem which asks for an *arbitrary* solution (rather than a CWA-solution), Kolaitis, Panttaja and Tan [11] have recently established an undecidability result. In particular, they have shown that the $\text{EXISTENCE-OF-SOLUTIONS}$ problem is undecidable for the following data exchange setting, which we call D_{emb} . The source and the target schema of D_{emb} consist of ternary relation symbols R and R' , respectively. There is one source-to-target dependency that copies R to R' , and the target dependencies are as follows:

$$\begin{aligned} d_{\text{func}} &= R'(x, y, z_1) \wedge R'(x, y, z_2) \rightarrow z_1 = z_2, \\ d_{\text{assoc}} &= R'(x, y, u) \wedge R'(y, z, v) \wedge R'(u, z, w) \rightarrow R'(x, v, w), \\ d_{\text{total}} &= \\ &R'(x_1, x_2, x_3) \wedge R'(y_1, y_2, y_3) \rightarrow \bigwedge_{1 \leq i, j \leq 3} \exists z R'(x_i, y_j, z). \end{aligned}$$

Informally, D_{emb} can be used to solve the *embedding problem for finite semigroups*, which asks, given a partial function $p: X \times Y \rightarrow Z$, whether there is a (finite) total function $f: X' \times Y' \rightarrow Z'$ such that f is associative and extends p (i.e., if $p(x, y) = z$, then $f(x, y) = z$): first encode the graph of p by $S := \{R(x, y, z) \mid p(x, y) = z\}$; then a solution for S exists if and only if the desired function f exists. Since the embedding problem for finite semigroups is known to be undecidable (see [11]), checking whether there is a solution for a given source instance S is undecidable. However, as the following example shows, this reduction does not lead to a proof of the undecidability of $\text{EXISTENCE-OF-CWA-SOLUTIONS}$:

EXAMPLE 6.1. Consider the partial function encoded by $S = \{R(0, 1, 1)\}$, and assume that there is a CWA-solution T for S under D_{emb} . Since T is finite, there is a largest integer $k \geq 0$ such that there exist pairwise distinct values $v_0, v_1, \dots, v_k \in \text{Dom}(T)$ with

$$\{R'(0, 1, v_0), R'(v_0, 1, v_1), \dots, R'(v_{k-1}, 1, v_k)\} \subseteq T.$$

Since T satisfies d_{total} , there also exists $v \in \text{Dom}(T)$ such that $R'(v_k, 1, v) \in T$, and by the choice of k we have

$$v = v_i \tag{3}$$

for some $i \in \{0, 1, \dots, k\}$.

On the other hand,

$$T' = \{R'(a, b, c) \mid a + b = c \bmod k + 2\}$$

is a solution for S under D_{emb} , and Theorem 4.8 implies that there is a homomorphism h from T to T' . In particular,

$$\begin{aligned} \{R'(0, 1, h(v_0)), R'(h(v_0), 1, h(v_1)), \dots, \\ R'(h(v_{k-1}), 1, h(v_k)), R'(h(v_k), 1, h(v))\} \subseteq T'. \end{aligned}$$

By the definition of T' we must have $h(v_j) = j + 1$ for all $j \in \{0, 1, \dots, k\}$, and $h(v) = 0$, but (3) tells us that $h(v) = h(v_i) = i + 1$ —a contradiction.

It follows that there is no CWA-solution for S under D_{emb} , although T' encodes a finite total function that is associative and extends the partial function encoded by S . Consequently, the reduction given above does not work in the context of CWA-solutions.

Nevertheless, we can show undecidability of $\text{EXISTENCE-OF-CWA-SOLUTIONS}$ with a different proof, which shows that under the CWA, data exchange settings have the power to simulate Turing machines.

THEOREM 6.2. *There exists a data exchange setting D_{halt} such that $\text{EXISTENCE-OF-CWA-SOLUTIONS}(D_{\text{halt}})$ is undecidable.*

PROOF. We construct D_{halt} such that the halting problem for Turing machines on the empty input is reducible to $\text{EXISTENCE-OF-CWA-SOLUTIONS}(D_{\text{halt}})$. More precisely, we will reduce the following variant of the halting problem, called HALT : given a deterministic one-tape Turing machine $M = (Q, \Sigma, \delta, q_0, Q_F)$ (where Q is the set of states of M , Σ is the tape alphabet, δ is the transition function, i.e., a total function from $(Q \setminus Q_F) \times \Sigma$ to $Q \times \Sigma \times \{L, R\}$, $q_0 \in Q$ is the start state, $Q_F \subseteq Q$ is the set of final states, and the tape is infinite only to the right), decide whether M halts on the empty input.

The source schema of D_{halt} contains a 5-ary relation symbol Δ to encode the graph of δ , and a unary relation symbol Q_0 for q_0 . The following list contains the relation symbols of the target schema and their intended meaning.

- Δ' : a copy of Δ
- $t \prec t'$: t' is the next step in the computation after t
- $Q(t, q, p)$: in step t , M is in state q and the head is on position p
- $I(t, p, s)$: in step t , the tape contains s at position p
- $\text{NEXTPOS}(t, p, p')$: in step t , p' is the successor position of p on the tape
- $\text{END}(t, p)$: in step t , p is the last position of the tape for which $I(t, p, \cdot)$ is defined
- $\text{COPY}_L(t, t', p)$ and $\text{COPY}_R(t, t', p)$: the tape content of all positions left/right from position p in step t must be copied to the corresponding positions for step t'

The source-to-target dependencies consist of a tgd to copy Δ to Δ' , and another tgd to initialize the “start configuration”:

$$\begin{aligned} \Delta(q, s, q', s', d) &\rightarrow \Delta'(q, s, q', s', d), \\ Q_0(q) &\rightarrow Q(0, q, 1) \wedge I(0, 1, \square) \wedge I(0, 2, \square) \\ &\quad \wedge \text{NEXTPOS}(0, 1, 2) \wedge \text{END}(0, 2). \end{aligned}$$

(Here, \square is the blank symbol that is assumed to be in Σ .) The target constraints simulate the Turing machine as follows. There are two constraints that simulate a transition—one for a transition where the tape head moves to the left, and one for a transition where the tape head moves to the right:

$$\begin{aligned} Q(t, q, p) \wedge I(t, p, s) \wedge \text{NEXTPOS}(t, p', p) \wedge \Delta'(q, s, q', s', L) \\ \rightarrow \exists t' (t \prec t' \wedge Q(t', q', p') \wedge I(t', p, s') \\ \quad \wedge \text{COPY}_L(t, t', p) \wedge \text{COPY}_R(t, t', p)), \end{aligned}$$

$$\begin{aligned} & Q(t, q, p) \wedge I(t, p, s) \wedge \text{NEXTPOS}(t, p, p') \wedge \Delta'(q, s, q', s', R) \\ & \rightarrow \exists t' (t \prec t' \wedge Q(t', q', p') \wedge I(t', p, s') \\ & \quad \wedge \text{COPY}_L(t, t', p) \wedge \text{COPY}_R(t, t', p)). \end{aligned}$$

There are two constraints that copy the “linear order” on the tape positions and the inscriptions of all unmodified tape cells:

$$\begin{aligned} & \text{COPY}_L(t, t', p) \wedge \text{NEXTPOS}(t, p', p) \wedge I(t, p', s) \\ & \rightarrow \text{COPY}_L(t, t', p') \wedge \text{NEXTPOS}(t', p', p) \wedge I(t', p', s), \\ & \text{COPY}_R(t, t', p) \wedge \text{NEXTPOS}(t, p, p') \wedge I(t, p', s) \\ & \rightarrow \text{COPY}_R(t, t', p') \wedge \text{NEXTPOS}(t', p, p') \wedge I(t', p', s). \end{aligned}$$

Finally, there is a constraint that adds a new tape cell to the end of the tape:

$$\begin{aligned} \text{END}(t, p) \wedge t \prec t' \rightarrow \exists p' (\text{NEXTPOS}(t', p, p') \\ \wedge I(t', p', \square) \wedge \text{END}(t', p')). \end{aligned}$$

The reduction from HALT to EXISTENCE-OF-CWA-SOLUTIONS(D_{halt}) is carried out as follows. On input of an instance $M = (Q, \Sigma, \delta, q_0, Q_F)$ for HALT, we create the source instance

$$S_M := \{\Delta(q, s, q', s', d) \mid \delta(q, s) = (q', s', d)\} \cup \{Q_0(q_0)\}.$$

Then, M halts on the empty input if and only if there is a CWA-solution for S_M under D_{halt} (details will be given in the full version of the paper). \square

REMARK 6.3. While the reduction of Kolaitis, Panttaja and Tan described above does not work in the context of CWA-solutions, the reduction given in Theorem 6.2 does not work in the context of (general) solutions. In fact, for every source instance S for D_{halt} , the target instance that consists of all atoms $R(\bar{u})$, where R is a relation symbol of D_{halt} ’s target schema of arity r , say, and $\bar{u} \in (\text{Const}(S) \cup \{0, 1, 2, \square\})^r$ is a solution for S under D_{halt} .

Furthermore, if we enhance D_{halt} by relation symbols that carry the final states of M , and an egd that tries to identify two constants whenever a final state is reached, then a reduction from the complement of HALT shows that EXISTENCE-OF-CWA-SOLUTIONS is undecidable even if we would allow infinite solutions: M does not halt on the empty input $\iff M$ does not reach a final state \iff an (infinite) CWA-solution exists.

Note that Theorem 6.2 and Corollary 5.2 immediately lead to

COROLLARY 6.4. *There is a data exchange setting D_{halt} such that EXISTENCE-OF-UNIVERSAL-SOLUTIONS(D_{halt}) is undecidable.*

When restricting attention to the well-known class of *weakly acyclic* data exchange settings, however, known tractability results for universal solutions carry over to CWA-solutions.

DEFINITION 6.5. (weakly acyclic [6, 4]) Let D be a data exchange setting with target schema τ and target dependencies Σ . A *position* over τ is a pair (R, i) , where R is a relation symbol of some arity, say r , and $i \in \{1, \dots, r\}$. For

a conjunction $\varphi(\bar{x})$ of relational atomic formulas over τ and a variable x , we say that x *appears at position* (R, i) in φ if φ contains a conjunct $R(t_1, \dots, t_r)$ with $t_i = x$. The *dependency graph* of Σ is a directed graph, where the vertices are the positions over τ , and for every tgd $\varphi(\bar{x}, \bar{y}) \rightarrow \exists \bar{z} \psi(\bar{x}, \bar{z})$ in Σ , every variable $x \in \bar{x}$, and every position (R, i) at which x appears in φ , there is

- an edge from (R, i) to every position at which x appears in ψ , and
- an *existential edge* from (R, i) to every position at which some variable from \bar{z} appears in ψ .

D is called *weakly acyclic* if no cycle in the dependency graph of Σ contains an existential edge.

Employing the recent result of [8] that the core of the universal solutions can be computed in polynomial time under weakly acyclic settings, Theorem 5.1, and [11, Proposition 3.1], we obtain:

PROPOSITION 6.6. *Let D be a data exchange setting that is weakly acyclic. Given a source instance S for D , we can compute a CWA-solution for S under D in time polynomial in $|S|$, if there is such a CWA-solution, and output that there is no CWA-solution for S under D otherwise. Moreover, EXISTENCE-OF-CWA-SOLUTIONS(D) can be PTIME-hard.*

Let us conclude this section with a problem that is closely related to the EXISTENCE-OF-CWA-SOLUTIONS problem: given a source instance S and a target instance T (for a fixed data exchange setting D), decide whether T is a CWA-solution for S under D . This problem is in NP, provided that D is weakly acyclic. To see this, note that by Theorem 4.8, it can be solved by (a) deciding whether T is a CWA-presolution for S under D , and (b) deciding whether T is a universal solution for S under D . It is easy to see that (a) is in NP: it suffices to “guess” an $\alpha: \mathcal{J}_D \rightarrow \text{Dom}(T)$ such that $S \cup T$ is the result of a successful α -chase of S with Σ of length at most $|T|$. Since D is weakly acyclic, (b) is in NP, too: it suffices to find a homomorphism from T to another universal solution for S , which can be computed in polynomial time [6].

7. QUERY ANSWERING

7.1 Semantics

In [12], new query answering semantics have been proposed to overcome the anomalies that arise from the certain answers semantics of [6], and the certain universal answers semantics of [7] (recall Section 3).

Before we give their definitions, let us fix some basic notation. A *valuation* of an instance I is a mapping $v: \text{Null}(I) \rightarrow \text{Const}$. A valuation v of I is extended to $\text{Const}(I)$ by letting $v(c) := c$ for each $c \in \text{Const}(I)$, to tuples $\bar{u} = (u_1, u_2, \dots, u_n)$ over $\text{Dom}(I)$ by $v(\bar{u}) := (v(u_1), v(u_2), \dots, v(u_n))$, and to I by $v(I) := \{R(v(\bar{u})) \mid R(\bar{u}) \in I\}$. Under the CWA, a *solution* T for a source instance S under some data exchange setting $D = (\sigma, \tau, \Sigma_{\text{st}}, \Sigma_t)$ represents an *unknown* solution $v(T)$ that does not contain incomplete information, where v is a suitable valuation of T . Thus, the set of *possible solutions* represented by T is

$$\text{Rep}_D(T) := \{v(T) \mid v \text{ is a valuation of } T \text{ with } v(T) \models \Sigma_t\}.$$

To answer a query Q on T we can use the set of certain or the set of maybe answers (see [14], where the latter is called possible answers) of Q on T :

$$\begin{aligned}\square_D Q(T) &:= \bigcap_{R \in \text{Rep}_D(T)} Q(R), \\ \diamond_D Q(T) &:= \bigcup_{R \in \text{Rep}_D(T)} Q(R).\end{aligned}$$

The semantics proposed in [12] are now defined via combinations of $\square_D Q(T)$ and $\diamond_D Q(T)$:

- *Certain answers semantics:*

$$\text{certain}_{\square}^D(Q, S) := \bigcap_{T \in \llbracket S \rrbracket_{\text{CWA}}^D} \square_D Q(T).$$

- *Potential certain answers semantics:*

$$\text{certain}_{\diamond}^D(Q, S) := \bigcup_{T \in \llbracket S \rrbracket_{\text{CWA}}^D} \square_D Q(T).$$

- *Persistent maybe answers semantics:*

$$\text{maybe}_{\square}^D(Q, S) := \bigcap_{T \in \llbracket S \rrbracket_{\text{CWA}}^D} \diamond_D Q(T).$$

- *Maybe answers semantics:*

$$\text{maybe}_{\diamond}^D(Q, S) := \bigcup_{T \in \llbracket S \rrbracket_{\text{CWA}}^D} \diamond_D Q(T).$$

As pointed out in [12], the anomalies indicated in Section 3 disappear with the new semantics. For instance, recall the definition of copying data exchange settings from Section 3. For such a setting D and a source instance S for D , we have $\llbracket S \rrbracket_{\text{CWA}}^D = \{T\}$, where $T := \{R'(\bar{u}) \mid R(\bar{u}) \in S\}$, and $\text{Rep}_D(T) = \{T\}$. Thus, for all queries Q over the target schema of D ,

$$\begin{aligned}\text{certain}_{\square}^D(Q, S) &= \text{certain}_{\diamond}^D(Q, S) = \text{maybe}_{\square}^D(Q, S) \\ &= \text{maybe}_{\diamond}^D(Q, S) = Q(T),\end{aligned}$$

as it intuitively should be.

The following theorem shows that answering queries under the four semantics can sometimes be reduced to evaluating the query over a minimal or a maximal CWA-solution. It is a consequence of Theorem 5.1, Proposition 5.4, and the fact that if T_1, T_2, T_3 are CWA-solutions for S under D such that T_1 is minimal and T_3 is maximal, then $\text{Rep}_D(T_1) \subseteq \text{Rep}_D(T_2)$ and $\text{Rep}_D(T_2) \subseteq \text{Rep}_D(T_3)$. It generalizes the analogous result of [12] for data exchange settings without target dependencies.

THEOREM 7.1. *Let D be a data exchange setting, S a source instance for D , and Q a query over the target schema of D . Then,*

$$\begin{aligned}\text{certain}_{\diamond}^D(Q, S) &= \square_D Q(\text{CORE}_D(S)), \\ \text{maybe}_{\square}^D(Q, S) &= \diamond_D Q(\text{CORE}_D(S)),\end{aligned}$$

and for all CWA-solutions T for S under D ,

$$\text{certain}_{\square}^D(Q, S) \subseteq \square_D Q(T), \quad \text{maybe}_{\diamond}^D(Q, S) \supseteq \diamond_D Q(T).$$

Furthermore, if the target dependencies of D consist of egds only, or if the source-to-target dependencies and the target dependencies of D consist of egds and full tgds, then

$$\begin{aligned}\text{certain}_{\square}^D(Q, S) &= \square_D Q(\text{CANSOL}_D(S)), \\ \text{maybe}_{\diamond}^D(Q, S) &= \diamond_D Q(\text{CANSOL}_D(S)).\end{aligned}$$

As an immediate consequence one obtains:

COROLLARY 7.2. *For any data exchange setting D , source instance S for D , and query Q over the target schema of D ,*

$$\begin{aligned}\text{certain}_{\square}^D(Q, S) &\subseteq \text{certain}_{\diamond}^D(Q, S) \\ &\subseteq \text{maybe}_{\square}^D(Q, S) \subseteq \text{maybe}_{\diamond}^D(Q, S).\end{aligned}$$

Note that $\text{certain}^D(Q, S)$ and $\text{u-certain}^D(Q, S)$ (as defined at the end of Section 2) are contained in $\text{certain}_{\square}^D(Q, S)$.

7.2 Data Complexity

Let **answers** be one of the semantics considered in Section 7.1. The *data complexity* of a first-order query Q over the target schema of a data exchange setting D with respect to **answers** is defined as the complexity of the language

$$\mathcal{L}_{\text{answers}}(D, Q) := \{ \langle S, \bar{u} \rangle \mid S \text{ is a source instance for } D \\ \text{and } \bar{u} \in \text{answers}^D(Q, S) \},$$

where $\langle S, \bar{u} \rangle$ is a suitable encoding of the pair (S, \bar{u}) .

The data complexity of first-order queries with respect to certain_{\square} , $\text{certain}_{\diamond}$, maybe_{\square} and maybe_{\diamond} is as in [12], i.e., in co-NP or NP, respectively, as long as we restrict attention to data exchange settings which are *richly acyclic* in the following sense:

DEFINITION 7.3. (richly acyclic) Let D be a data exchange setting with target dependencies Σ . The *extended dependency graph* of Σ is obtained from the dependency graph (see Definition 6.5) by adding the following existential edges: for every tgds $\varphi(\bar{x}, \bar{y}) \rightarrow \exists \bar{z} \psi(\bar{x}, \bar{z})$ in Σ , for every variable y in \bar{y} , and for every position (R, i) at which y appears in φ , there is an existential edge from (R, i) to every position at which some variable in \bar{z} appears in ψ . D is called *richly acyclic* if no cycle in the extended dependency graph of Σ contains an existential edge. Note that every richly acyclic data exchange setting is weakly acyclic.

PROPOSITION 7.4. *Let D be a data exchange setting that is richly acyclic, and let S be a source instance for D . Then, for every first-order query Q , the problems $\mathcal{L}_{\text{certain}_{\square}}(D, Q)$ and $\mathcal{L}_{\text{certain}_{\diamond}}(D, Q)$ are in co-NP, and $\mathcal{L}_{\text{maybe}_{\square}}(D, Q)$ and $\mathcal{L}_{\text{maybe}_{\diamond}}(D, Q)$ are in NP.*

The reason which kept us from proving Proposition 7.4 for weakly acyclic settings is that a tgds $\varphi(\bar{x}, \bar{y}) \rightarrow \exists \bar{z} \psi(\bar{x}, \bar{z})$, an assignment for the variables in \bar{x} , and a variable in \bar{z} may generate distinct values based on distinct assignments for \bar{y} (which is not the case in the standard chase), and thus, although D is weakly acyclic, there may be a source instance S and a mapping $\alpha: \mathcal{J}_D \rightarrow \text{Dom}$ such that there is no finite α -chase of S with the dependencies of D . Nevertheless, we conjecture that Proposition 7.4 is actually true for weakly acyclic settings.

The complexity bounds are “tight” in the following sense:

THEOREM 7.5. *There is a data exchange setting D with richly acyclic target dependencies and a conjunctive query Q with one inequality such that the problems $\mathcal{L}_{\text{certain}_{\square}}(D, Q)$ and $\mathcal{L}_{\text{certain}_{\diamond}}(D, Q)$ are co-NP-complete, and $\mathcal{L}_{\text{maybe}_{\square}}(D, \neg Q)$ and $\mathcal{L}_{\text{maybe}_{\diamond}}(D, \neg Q)$ are NP-complete.*

The NP-hardness (resp., co-NP-hardness) is proved by a reduction from 3-SAT (resp., its complement); details can be found in the full version of the paper.

Type of $D = (\sigma, \tau, \Sigma_{st}, \Sigma_t)$	Type of Q		
	Union of CQ	Union of CQ with ≤ 1 inequality per disjunct	FO-query
weakly acyclic	PTIME	co-NP-hard	co-NP-hard
richly acyclic	PTIME	co-NP-complete	co-NP-complete
Σ_{st} : unrestricted; Σ_t : egds	PTIME	PTIME	co-NP-complete
Σ_{st} : full tgds; Σ_t : egds and full tgds	PTIME	PTIME	PTIME

Table 1: Complexity of $\mathcal{L}_{\text{certain}_{\square}}(D, Q)$ and $\mathcal{L}_{\text{certain}_{\diamond}}(D, Q)$ for certain restrictions of D and Q

Let us mention that a slightly weaker version of Theorem 7.5, for a conjunctive query with *two* inequalities, already follows from Mařdry [13]: his proof is formulated for the certain answers semantics of [6], but it is not hard to see that it carries over to certain_{\square} and $\text{certain}_{\diamond}$. We emphasize, however, that the data exchange setting in [13] has *no* target dependencies. Also, Fagin et al. [6] presented a polynomial-time algorithm to compute $\text{certain}^D(Q, S)$ for unions of conjunctive queries with at most one inequality per disjunct (i.e., potentially infinite disjunctions of conjunctive queries, where each conjunctive query may contain an inequality), provided that the data exchange setting is weakly acyclic. Their algorithm is able to compute certain_{\square} and $\text{certain}_{\diamond}$ for such queries, if either the target dependencies of the data exchange setting consist of egds only, or the source-to-target and the target dependencies consist of egds and full tgds only. Theorem 7.5 implies that the algorithm does *not* compute certain_{\square} or $\text{certain}_{\diamond}$ for unions of conjunctive queries with at most one inequality per disjunct under more general data exchange settings, unless, of course, PTIME = co-NP.

Finally, moving from first-order queries to the class of unions of conjunctive queries (i.e., the class of potentially infinite disjunctions of conjunctive queries *without* inequalities, which in particular, comprises the class of datalog queries) we obtain that query answering is tractable for all weakly acyclic settings:

THEOREM 7.6. *Let D be a weakly acyclic data exchange setting, and let Q be a union of conjunctive queries over the target schema of D . Given a source instance S for D , we can compute $\text{certain}_{\square}^D(Q, S)$ and $\text{certain}_{\diamond}^D(Q, S)$ in time polynomial in $|S|$.*

PROOF. To evaluate such a query Q for a source instance S , we can proceed as follows: First, we use Proposition 6.6 to generate a CWA-solution T for S under D . Then, we evaluate the query Q on T and compute the set

$$Q(T)_{\perp} := \{\bar{u} \in Q(T) \mid \bar{u} \text{ contains no nulls}\}.$$

The following lemma tells us that $Q(T)_{\perp}$ coincides with $\text{certain}_{\square}^D(Q, S)$ and $\text{certain}_{\diamond}^D(Q, S)$.

LEMMA 7.7. *Let D be a data exchange setting, S a source instance for D , and Q a union of conjunctive queries over the target schema of D . For every CWA-solution T for S under D , we have*

$$\text{certain}_{\square}^D(Q, S) = \text{certain}_{\diamond}^D(Q, S) = \square_D Q(T) = Q(T)_{\perp}.$$

This lemma can be proved along the lines of the proof of the corresponding result for data exchange settings without target dependencies given in [12]. One difficulty is that in our setting, the certain answers to an individual CWA-solution T are $\square_D Q(T) = \bigcap_{R \in \text{Rep}_D(T)} Q(R)$, and not, as

in the case of data exchange settings without target dependencies, $\bigcap_{R \in \text{Rep}(T)} Q(R)$, with $\text{Rep}(T)$ consisting of all instances $v(T)$ for some valuation v of T . However, we can show that for a union Q of conjunctive queries, $\square_D Q(T)$ is the same as $\bigcap_{R \in \text{Rep}(T)} Q(R)$. Details can be found in the full version of the paper. \square

The complexity bound of Theorem 7.6 is tight in the following sense:

PROPOSITION 7.8. *There is a data exchange setting D whose target dependencies consist of full tgds only, and a conjunctive query Q such that $\mathcal{L}_{\text{answers}}(D, Q)$ is PTIME-hard, where answers is certain_{\square} , $\text{certain}_{\diamond}$, maybe_{\square} , or maybe_{\diamond} .*

Acknowledgments. We thank the anonymous referees for their helpful comments.

8. REFERENCES

- [1] M. Arenas, P. Barceló, R. Fagin, and L. Libkin. Locally consistent transformations and query answering in data exchange. In *Proc. PODS'04*, pages 229–240, 2004.
- [2] M. Arenas and L. Libkin. XML data exchange: consistency and query answering. In *Proc. PODS'05*, pages 13–24, 2005.
- [3] A. K. Chandra and P. M. Merlin. Optimal implementation of conjunctive queries in relational data bases. In *Proc. STOC'77*, pages 77–90, 1977.
- [4] A. Deutsch and V. Tannen. Reformulation of XML queries and constraints. In *Proc. ICDT'03*, pages 225–241, 2003.
- [5] R. Fagin. Inverting schema mappings. In *Proc. PODS'06*, pages 50–59, 2006.
- [6] R. Fagin, P. G. Kolaitis, R. J. Miller, and L. Popa. Data exchange: semantics and query answering. *Theoretical Computer Science*, 336(1):89–124, 2005. Full version of ICDT'03 paper.
- [7] R. Fagin, P. G. Kolaitis, and L. Popa. Data exchange: getting to the core. *ACM Transactions on Database Systems*, 30(1):174–210, 2005. Full version of PODS'03 paper.
- [8] G. Gottlob and A. Nash. Data exchange: Computing cores in polynomial time. In *Proc. PODS'06*, pages 40–49, 2006.
- [9] P. Hell and J. Nešetřil. The core of a graph. *Discrete Mathematics*, 109:117–126, 1992.
- [10] P. G. Kolaitis. Schema mappings, data exchange, and metadata management. In *Proc. PODS'05*, pages 61–75, 2005.
- [11] P. G. Kolaitis, J. Panttaja, and W. C. Tan. The complexity of data exchange. In *Proc. PODS'06*, pages 30–39, 2006.
- [12] L. Libkin. Data exchange and incomplete information. In *Proc. PODS'06*, pages 60–69, 2006.
- [13] A. Mařdry. Data exchange: On the complexity of answering queries with inequalities. *Information Processing Letters*, 94:253–257, 2005.
- [14] R. van der Meyden. Logical approaches to incomplete information: A survey. In *Logics for Databases and Information Systems*, pages 307–356. Kluwer, 1998.