Data Exchange: Computing Cores in Polynomial Time

Georg Gottlob*
Oxford University
georg.gottlob@comlab.ox.ac.uk

Alan Nash†
University of California, San Diego
anash@cs.ucsd.edu

ABSTRACT

Data exchange deals with inserting data from one database into another database having a different schema. We study and solve a central computational problem of data exchange, namely, computing the core of a universal solution to a data exchange problem. Fagin, Kolaitis, and Popa [9], have shown that among the universal solutions of a solvable data exchange problem, there exists a most compact one (up to isomorphism), "the core" (of any universal solution), and have convincingly argued that this core should be the solution of choice. They stated as an important open problem whether the core of a universal solution can be computed in polynomial time in the general setting where the source-to-target constraints are arbitrary tuple generating dependencies (TGDs) and the target constraints consist of equation generating dependencies (EGDs) and weakly-acyclic TGDs. In this paper we solve this problem by developing new efficient methods for computing the core of a universal solution. This positive result shows that the core approach of Fagin, Kolaitis, and Popa is feasible and applicable in a very general setting and thus provides a further momentum to the use of cores in data exchange.

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; F.2.2 [**Nonnumerical Algorithms and Problems**]: Computations on discrete structures;

General Terms: Algorithms, Theory, Databases

Keywords: Chase, core, complexity, conjunctive queries, constraints, data exchange, data integration, dependencies, query evaluation, tractability, universal solutions

1. INTRODUCTION

Data exchange research. Data exchange research is an important area of database theory that aims at understanding and developing foundations, methods, and algorithms for transferring data between differently structured databases. While data integration usually deals with query translation and with query processing among multiple databases, data exchange aims at *materializing* a target database stemming from some source databases. Clearly, once transferred to the target database, the data can be queried according to the target schema. While data exchange has been recognized as an important problem for several decades, systematic research on foundational and algorithmic issues of this problem has started only a few years ago with the fundamental work of Fagin, Kolaitis, Miller, and Popa [8].

The basic data exchange problem. The basic and most fundamental data exchange problem for relational databases was formally defined by Fagin et al. [8]. Given a source database schema σ, a target database schema τ, a source database instance S, and a set of constraints Σ, find a target database instance T such that S and T satisfy all constraints in Σ, denoted by $(S, T) \models \Sigma$. As in [8, 9], we restrict our attention to the following types of constraints:

Tuple generating dependencies (TGDs) which are first order sentences of the form $\forall \bar{u}\,(\phi(\bar{u}) \rightarrow \exists \bar{v}\,\psi(\bar{u}, \bar{v})$, where ϕ and ψ are conjunctions of atoms and $\bar{u}$ and $\bar{v}$ are lists of variables and where the variables in $\bar{u}$ all appear in ϕ. If $\bar{v}$ is empty, then we speak about a *full TGD*.

Equality generating dependencies (EGDs) which are first order sentences of the form $\forall \bar{u}\,(\phi(\bar{u}) \rightarrow v = w)$, where ϕ is a conjunction of atoms, $\bar{u}$ is a list of variables, and v and w are single variables from the list $\bar{u}$.

We will usually omit the universal quantifiers when writing a TGD or EGD.

Source-to-target constraints are those where the premise $\phi(\bar{u})$ contains atoms whose predicate symbols are relation names of the source signature σ only and where the conclusion $\psi(\bar{u}, \bar{v})$ is made of atoms whose predicate symbols are relation names of the target signature τ only. All atoms occurring in target constraints refer to the target signature only.

We consider the case where the set $\Sigma = \Sigma_{st} \cup \Sigma_t$ of constraints consists of source-to-target constraints Σ_{st} encoding conditions on the mapping between source and target data, and the target constraints Σ_t which express data dependencies on the target database.

This setting allows us to formulate a very large class of constraints. For example, let a source database contains a relation

$$student(\text{STUDNAME}, \text{BIRTHDATE}, \text{SSN}, \text{ZIPCODE})$$

and let the target database contains two relations

$$person(\text{NAME}, \text{BORN}, \text{SSN}, \text{ZIPCODE}, \text{PHONE})$$

*Research supported by REWERSE - a research "Network of Excellence" EU project reference number 506779.
†Research partly supported by the Wolfgang Pauli Institute and a Microsoft Research Fellowship.

PODS'06, June 26–28, 2006, Chicago, Illinois, USA.

$$zc(\text{ZIPCODE}, \text{STATE}).$$

Then a typical source-to-target constraint would be

$$st1: \ student(u_1, u_2, u_3, u_4) \ \rightarrow \ \exists v \, person(u_1, u_2, u_3, u_4, v),$$

while the target constraints could be

$$t1: \ person(u_1, u_2, u_3, u_4, u_5) \ \rightarrow \ \exists v \, zc(u_5, v)$$
$$t2: \ zc(u, v) \wedge zc(u, w) \ \rightarrow \ v = w.$$

Here st_1 and t_1 are inclusion dependencies, and t_2 is a functional dependency. Note that $t1$ and $t2$ together express a foreign key constraint. It is easy to see that, in addition to functional, inclusion dependencies, and foreign key constraints, also multivalued dependencies and even join dependencies can be expressed by formulas in our setting. Thus, this setting is very general and encompasses all major dependencies used in database design and for database maintenance.

The problem is to check whether a target instance T exists, and if so, to compute one. We allow *labeled null values* in form of variables to appear in target instances. Intuitively, they can be used as placeholders for unknown values. For example, a tuple

$$\langle Doe, 19880203, 1234567, 94305 \rangle$$

of the *student* source relation could be translated into a tuple

$$\langle Doe, 19880203, 1234567, 94305, x_1 \rangle$$

of the *person* relation, where x_1 is a variable representing a null value. For further, more detailed examples, see [8, 10].

Homomorphisms, Universal solutions and Cores. A *universal solution* of a data exchange problem is a target instance T which is more general than other solutions, i.e., such that for each other solution T' there exists a homomorphism $T \rightarrow T'$. Fagin, Kolaitis, Miller, and Popa [8] have shown that universal solutions of data exchange problems can be obtained via the well known *chase* procedure [3, 2, 15]. One first chases the set S with the source-to-target TGDs Σ_{st} and obtains in polynomial time an initial target instance $T = S^{\Sigma_{st}}$. Then one chases T with the target constraints Σ_t to obtain a universal solution T^{Σ_t} whenever this chase terminates. In order to guarantee termination, Fagin et al [8, 9] restrict themselves to target TGDs which are *weakly acyclic*. Weak acyclicity is a syntactic condition on TGDs ensuring that there are no cyclic dependencies among argument positions involving existential constraints (see Section 4 for a precise definition). Weak acyclicity [5, 8] has been so far the most general known sufficient condition for termination of the chase. Fagin and his colleagues [8] have shown that universal solutions are very useful for query answering. In particular, any universal solution can be used to obtain the *certain answers tuples* to a conjunctive query over the target schema, i.e., those answer tuples that are contained in *all* solutions of the data exchange problem.

There can be several universal solutions to a data exchange problem and these solutions can noticeably differ in size. However, as observed in [9], there is – up to isomorphism – one particular universal solution, called *the core* (of any universal solution), which is the most compact one. More specifically, the core of a universal solution U is (up to isomorphism) the smallest subset V of U such that V is homomorphically equivalent to U. Fagin, Kolaitis, and Popa [9] argue that the core of a data exchange problem should be the solution of choice.

Main research question tackled. Computing the core of an arbitrary instance is, however, NP-hard, as this is equivalent to computing the core of a graph [13, 9], computing the smallest equivalent subquery contained in a conjunctive query [4], or computing the condensation of a clause [11]. Therefore, Fagin et al. [9] wondered, whether the core of a universal solution of a data exchange problem whose target TGDs are weakly acyclic can be efficiently computed.

> **Problem [9]:** Given a data exchange problem whose source-to-target constraints are TGDs and whose target constraints consist of weakly-acyclic TGDs and arbitrary EGDs, can the core of a universal solution be computed in polynomial time?

This is precisely the problem we tackle in this paper. While there has been some progress and partial answers, the problem remained open for about three years. It was also mentioned as an important open Problem in Kolaitis' invited PODS'05 talk and paper *Schema Mappings, Data Exchange, and Metadata Management* [14]. The main result of the present paper is the following positive answer to this question.

> **Theorem 10.** The core of a universal solution of a data exchange problem whose source-to-target constraints are TGDs and whose target constraints consist of weakly-acyclic TGDs and arbitrary EGDs can be computed in polynomial time.

The proposed solution to the problem is technically rather involved. It led us to several insights and results that may be of more general interest, even outside the context of data exchange.

Before giving an informal outline of our new methods in Section 3, we first report in Section 2 on pertinent previous work that solved some relevant parts of the problem. Our exposition in these two sections is necessarily informal and approximate. These sections are followed by more technical material. Section 4 contains technical preliminaries and precise definitions. In Section 5, we derive some properties of retractions. In Section 6 we solve the core computation problem for weakly-acyclic TGDs as target constraints, and in Section 7 we show how to also handle EGDs.

2. PREVIOUS RESULTS

Fagin, Kolaitis, and Popa [9] proved that the core of a universal solution can be computed in polynomial time in two restricted settings:

- When the set Σ_t of target constraints is empty.
- When the set Σ_t of target constraints contains EGDs only.

They provided two different methods to obtain these results, of which one is directly relevant to our present work, namely, the *blocks method*. This method is based on the observation that the Gaifman graph of the variables of the result T of applying the source-to-target TGDs to a ground source instance S consists of connected components whose size is bounded by a constant b. Such instances T have a nice property: checking whether there is a homomorphism from any $T \in K$ where K is any set of instances with such bound b into any arbitrary other instance T' is feasible in polynomial time. In fact, this test essentially boils down to check whether each of these blocks has a homomorphism to T'. The core of T can then be obtained by checking whether T admits an endomorphism[1] $h : T \longrightarrow T$ such that $|h(T)| < |T|$ and if so, replacing T by $h(T)$. This process is repeated until T cannot be further shrunk via endomorphisms. The result the core.

Fagin et al. then extended this method to the case where Σ_t consists of EGDs. The difficulty here is that EGDs, when applied, can

[1]i.e., a homomorphism from T into itself.

merge blocks by equating variables from different blocks. Thus, after chasing EGDs over T, the result T^{Σ_t} has, in general, lost the bounded block-size property. However Fagin et al., by an insightful *Rigidity Lemma* [9] show that after equating a sequence of variables while enforcing EGDs, the final remaining variable is *rigid*, i.e., can be mapped only to itself in every endomorphism. The resulting instance T^{Σ_t} has thus the bounded block-size property if we treat such variables as constants.

In [10] Gottlob has shown that computing cores is tractable if the target dependencies Σ_t consist of *full* TGDs, i.e., TGDs without existentially quantified variables, and arbitrary EGDs. Note that a set of full TGDs is weakly acyclic.

For full TGDs the situation is rather complex. While $T = S^{\Sigma_{st}}$ has bounded block size, T^{Σ_t} has in general neither bounded block size nor rigid variables. In fact, while T^{Σ_t} contains no additional variables, different blocks of T can be merged through the creation of new atoms. A full TGD of the form $r(x, y) \wedge r(z, t) \rightarrow r(y, t)$ may obviously merge blocks. This situation is depicted in figure 1, where the original blocks of T are depicted as ovals, and where some new atoms created via full TGDs are depicted as black boxed. These atoms may connect previously separate blocks and as a result, very large blocks may arise. However, it was shown in [9] that

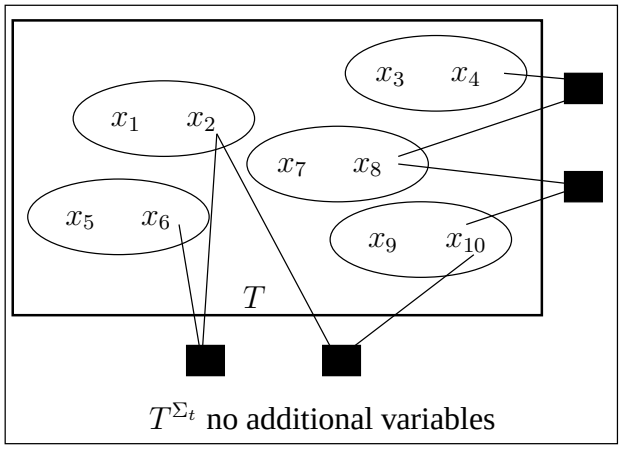

Figure 1: Structure of the target instance T^{Σ_t} in case Σ_t consists of full TGDs only

for checking whether a universal solution $B \subseteq T^{\Sigma_t}$ is is not yet the core, it suffices to look for a non-injective mapping h from the domain of T to the domain of B such that $h(T) \subset B$ is a homomorphism $T \longrightarrow B$. [2] If such a mapping exists, it can be found in polynomial time by exploiting the bounded block-size of T. We further showed that this mapping $h : T \longrightarrow B$ is actually a non-injective endomorphism $T^{\Sigma_t} \longrightarrow B$ (recall that T and T^{Σ_t} have the same domains). Moreover, h can be transformed in polynomial time into a non-injective *retraction* of of T^{Σ_t}. A retraction is an idempotent endomorphism. We needed to consider retractions and not arbitrary endomorphisms, because for a retraction r it holds that $r(T^{\Sigma_t}) \models \Sigma_t$, while this is not always true for endomorphisms in general (as will be made clear through Example 1 in Section 5). Starting from $B = T^{\Sigma_t}$, by successively replacing B with B' as described, we eventually reach the core.

This tractability result was then extended in [10] to the setting where Σ_t contains EGDs in addition to full TGDs. This was achieved by a simulation of EGDs through full TGDs. Note that with full TGDs and EGDs we can express functional, join, and multivalued dependencies, but not inclusion dependencies or foreign key constraints.

Another relevant class of data exchange problems arises when the set Σ_t of target constraints is restricted to contain weakly acyclic *simple TGDs* and arbitrary EGDs. A simple TGD is one whose left side consists of a single atom with no repeated variables. In [10] it was shown that for this class of problems, the core can be computed in polynomial time, too. The proof is based on the observation that chasing with simple TGDs does not change hypertree width [12]. This class is practically relevant, because it covers as target constraints the important class of functional dependencies and acyclic inclusion dependencies, and thus also foreign key constraints (as long as they do not destroy weak acyclicity). However it does not include multivalued or join dependencies. Fagin [7] has shown tractability for a slightly larger class by different means.

3. OUTLINE OF MAIN NEW IDEAS

In this section, we outline the main ideas underlying our solution of the core computation problem for the general case, when the target constraints Σ_t may consist of weakly-acyclic TGDs and arbitrary EGDs. Such constrains encompass all major types of data dependencies. We will first deal with weakly-acyclic TGDs as target constraints and then show how EGDs can be added.

For weakly-acyclic TGDs the situation is yet more complicated than for full TGDs. Again, let T denote a target instance obtained by chasing the source instance S with the source-to-target constraints Σ_{st} (for an arbitrary chase order). As before, T has bounded block-size. However, while in the case of full TGDs, T already contained *all* variables of T^{Σ_t}, now there can be further variables outside T and these variables can appear in large (unbounded) blocks of T^{Σ_t}, see Figure 2. We cannot proceed, as in

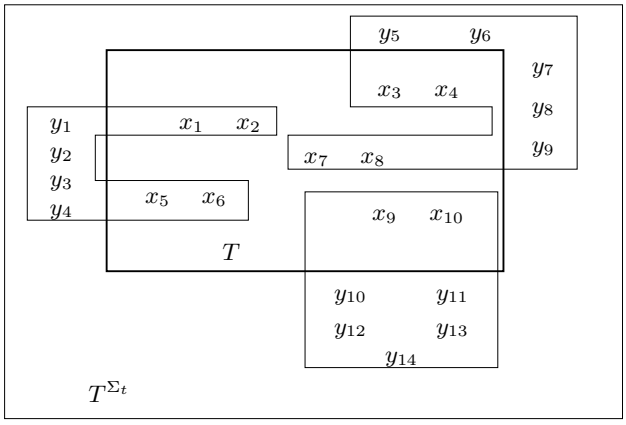

Figure 2: Structure of the target instance T^{Σ_t} in case Σ_t consists of weakly acyclic TGDs

the case of full TGDs, by trying to find a non-injective homomorphism h from T to some intermediate instance $B \subset T^{\Sigma_t}$ and hope that h extends automatically to a non-injective endomorphism of $T^{\Sigma_t} \rightarrow B$. In fact, there may not exist any non-injective homomorphism $T \rightarrow B$, while there may well exist a non-injective endomorphism h of $T^{\Sigma_t} \rightarrow B$ such that $h(T^{\Sigma_t}) \subset B$, where $h(x) = h(y)$ for two distinct values x and y that are *not both in* T. But variables outside T may appear in large blocks and it is thus not at all obvious how homomorphisms involving them can be found in polynomial time.

[2] In[9], such mappings were called "useful" endomorphisms.

Our solution of this problem is based on the following ideas.

Idea 1. For technical reasons, we compute *non-injective retractions* $T^{\Sigma_t} \to B$ rather than arbitrary (i.e., not necessarily idempotent) non-injective endomorphisms. The reason is that if h is a retraction, then $h(T^{\Sigma_t}) \models \Sigma_t$ (Theorem 4). Note that this is *not true* for arbitrary endomorphisms (see Example 1). In order to find a non-injective retraction, we can always find a non-injective endomorphism of T^{Σ_t} first and then transform it in polynomial time into a suitable retraction (Theorem 5).

If x is a variable of T^{Σ_t} that was introduced at some chase step via an existential TGD ξ from Σ_t, then the *parents* of x are all values occurring in the atoms that made ξ fire and the *siblings* are all other new variables introduced at the same chase step. The ancestors of a variable are defined in the usual way via the transitive closure of the parent relation.

Idea 2. We observe that each variable of T^{Σ} has a bounded set of ancestors and a bounded set of siblings of ancestors. Our idea is to exploit this fact.

The next idea deals with *how* to exploit the bounded set of ancestors.

Idea 3. Assume that we have already constructed a retraction $h : T^{\Sigma_t} \to B \subseteq T^{\Sigma_t}$ and we want to see whether B can be further shrunk. This means that we need to see whether two distinct values x, y of B can be further "lumped together" by a homomorphism h' such that $h'(x) = h'(y)$. To this aim, we define, for all pairs of distinct values x, y of B, the sub-instance $T_{xy} \subset T^{\Sigma_t}$ which contains all atoms over the set of values of T, x y, and their siblings, and all ancestors of x and y and the siblings of these ancestors. Given that T has bounded block-size and that the number of ancestors and siblings of each variable is bounded by a constant, these sets T_{xy} all have bounded block-size. We can then check for each T_{xy} in polynomial time whether there is a homomorphism $T_{xy} \to B$ such that x and y are mapped to the same element z, see Figure 3.

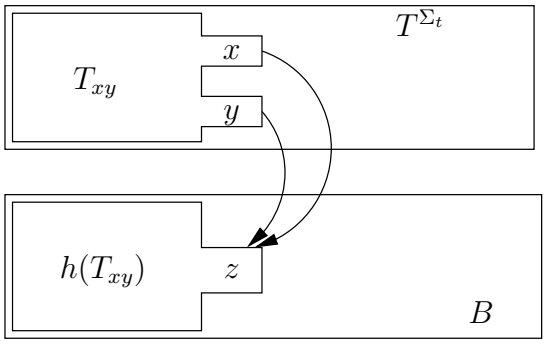

Figure 3: Improvement of a homomorphism

If this is possible for some T_{xy}, then this homomorphism h can be extended in polynomial time to an non-injective endomorphism $h : T^{\Sigma_t} \to B$ such that $h(T^{\Sigma_t}) \subset B$. Then, as explained before, from such a h we can compute in polynomial time a retraction h' of T^{Σ_t} such that $h'(T^{\Sigma_t}) \subset h(B) \subset B$ and thus B can be replaced by a smaller instance $h(B)$ which also satisfies Σ_t. Otherwise B is already the core.

In order to establish that each homomorphism $T_{xy} \to B \subseteq T^{\Sigma_t}$ mapping x and y to the same element can be extended in polynomial time to to an endomorphism $T^{\Sigma_t} \to B$ such that $h(B) \subset B$, we show a slightly stronger result:

Idea 4. We show that whenever a subset A of T^{Σ_t} contains T and is closed under ancestors and siblings, and whenever B satisfies Σ_t, then any homomorphism $h : A \to B$ can be extended in polynomial time to an homomorphism from $T^{\Sigma_t} \to B$. (Theorem 7)

These are the four main new ideas we used in order to show that the core can be computed in polynomial time in case Σ_t consists of TGDs only. To cover, in addition EGDs, we need two further ideas

Idea 5. We simulate EGDs by full TGDs by introducing an additional binary relation E which stands for 'equal' and by adding some consistency rules which say that values which are marked as equal in the E relation are indistinguishable by the other relations of the target database.

This simulation introduces a big new problem. Adding the new full TGDs may create new cycles and will in general yield a set of TGDs which is not weakly acyclic. Therefore, there is a risk that the chase will not terminate. Here comes our final idea that solves this problem, too.

Idea 6. We observe that for a particular chase order which can be statically determined, TGDs with existentially quantified variables will never fire on premises containing variables whose ancestor-tree exceeds a certain depth. Thus the chase terminates in polynomial time, and the crucial bounded-ancestors property is still guaranteed.

This concludes the rather superficial presentation of our main ideas. In the rest of this paper we will make these ideas more concrete and provide the glue for putting them together.

4. TECHNICAL PRELIMINARIES

Basics. A schema σ is a list of relation symbols and their arities. An instance A *over* σ has one relation for every relation symbol in σ, of the same arity. We write $\mathrm{arity}(R)$ for the arity of a relation R.

We define the domain $\mathrm{dom}(A)$ of an instance A as the set of *values* which appear in A. In this paper, we consider only finite instances with two types of values: constants and variables. The latter are also known as *labeled nulls*. We write $\mathrm{var}(A)$ for the variables in A and $\mathrm{const}(A)$ for the constants in A; therefore $\mathrm{dom}(A) = \mathrm{var}(A) \cup \mathrm{const}(A)$. If A contains no variables, we say it is *ground*.

We write $\bar{a}$ for a *tuple* $(a_1, \ldots, a_r)$ where the arity r is usually clear from context. We write $a \in \bar{a}$ whenever $a = a_i$ for some i satisfying $1 \leq i \leq r$. In some cases, we overload the notation $\bar{a}$ to denote just a list of variables $a_1, \ldots, a_r$; which use is intended should be clear from context. In particular, sometimes we write $\bar{a} \in X$ to denote $a_1, \ldots, a_r \in X$ which is equivalent to $(a_1, \ldots, a_r) \in X^r$. This is a useful convention since it is cumbersome to write instead $\bar{a} \in X^r$ when the arity r of $\bar{a}$ has not been explicitly introduced. Similarly, we write $\bar{a} \notin X$ to denote $a_1, \ldots, a_r \notin X$, which is different from $(a_1, \ldots, a_r) \notin X^r$.

The *Gaifman graph* $\mathcal{G}(T)$ of instance T is the undirected graph G with vertex set $V^G := \mathrm{dom}(T)$ where there is an edge between $x, y \in \mathrm{dom}(T)$ iff x and y appear together in some tuple of some relation of T. The *Gaifman graph of variables* $\mathcal{G}_V(T)$ of instance T is $\mathcal{G}(T)$ restricted to the variables in T. A *block* of T is a connected component of $\mathcal{G}_V(T)$. We write $\mathrm{blocks}(T)$ for the set of all blocks of T. If $v \in \mathrm{var}(T)$ then $\mathrm{block}(v, T)$ denotes the block of T containing v. Let $V \subseteq \mathrm{var}(T)$, then $\mathrm{blocks}(V, T) = \bigcup_{v \in V} \{\mathrm{block}(v, T)\}$, i.e., $\mathrm{blocks}(V, T)$ is the set of all blocks of T that contain at least one variable from V. The *block size* of an instance T, denoted by $\mathrm{blocksize}(T)$ is the maxi-

mum number of variables appearing in a block of T. We say that a set of instances K has *bounded block size* if there is a bound b such that the block size of every $T \in K$ is $\leq b$.

We adopt the RAM model for our complexity bounds.

Homomorphisms. A function $f : \mathrm{dom}(A) \to \mathrm{dom}(B)$ is a *homomorphism* if whenever $R(\bar{a})$ holds in A, $R(h(\bar{a}))$ holds in B and if $h(c) = c$ for every constant in A. We write $A \to B$ in case there is a homomorphism between A and B. If $A \to B$ and $B \to A$, we say that A and B are homomorphically equivalent, which we write $A \leftrightarrow B$. If K is a set of instances, we say that T is *universal* for K if for all $A \in K$, $T \to A$.

An injective homomorphism whose inverse is also a homomorphism.is an *isomorphism*. A homomorphism $h : A \to A$ is an *endomorphism* of A. Since we only consider finite instances, notice that an endomorphism is surjective iff it is injective. An endomorphism r on A is a *retraction* if r is the identity on its range. That is, if r is idempotent ($r \circ r = r$). If also $r(A) = B \subseteq A$, we say that B is a *retract* of A and we write $A \hookrightarrow B$. If $A \hookrightarrow B$, then $A \leftrightarrow B$; that is, A and B are homomorphically equivalent since r is a homomorphism $A \to B$ and the identity is a homomorphism $B \to A$. A retraction is *proper* if it is not surjective (which in the case of finite instances is the same as not injective). An instance is a *core* if it has no proper retractions. A core C of an instance A is a retract of A which is a core. That is, C is a minimal retract of A. Cores of an instance A are unique up to isomorphism and therefore we can talk about *the core* of A, which we denote $\mathrm{core}(A)$. It follows that A and B are homomorphically equivalent iff their cores are isomorphic. In symbols: $A \leftrightarrow B$ iff $\mathrm{core}(A) \approx \mathrm{core}(B)$.

Constraints. In this paper, we only consider constraints ξ of the form

$$\phi(\bar{u}) \to \exists \bar{v}\, \psi(\bar{u}, \bar{v})$$

where ϕ and ψ are conjunctions of atoms, which may include equations. We adopt the standard convention that all variables which are not otherwise quantified are universally quantified and we require that all variables $\bar{u}$ appear in ϕ. Such constraints are known as *embedded (implicational) dependencies* [6]. We call ϕ the *premise* and ψ the *conclusion*. If $\bar{v}$ is empty, then ξ is a *full dependency*. If ψ consists only of equations, then ξ is an *equality-generating dependency (EGD)*. If ψ consists only of relational atoms, then ξ is a *tuple-generating dependency (TGD)*. Every set Σ of embedded dependencies is equivalent to a set of TGDs and EGDs. We write $A \models \Sigma$ if the instance A satisfies all the constraints in Σ. In this paper we only consider *finite* sets of constraints and to simplify the presentation we do not say this explicitly in the statement of every result.

We define the *width* of ξ to be the arity of $\bar{u}$ and the *height* of ξ to be the arity of $\bar{v}$. For a set Σ of constraints, the width of Σ is the maximum width of a constraint in Σ and the height of a Σ is the maximum height of a constraint in Σ.

Chase: Basics. The chase is a well-known algorithm which proceeds step by step as follows. Assume Σ is a set of TGDs and EGDs. We set $A_0^\Sigma = A$. To obtain A_{s+1}^Σ from A_s^Σ we proceed as follows. If there is some constraint $\xi \in \Sigma$ of the form

$$\phi(\bar{u}) \to \exists \bar{v}\, \psi(\bar{u}, \bar{v})$$

such that $A_s^\Sigma \not\models \xi$, we say that ξ *applies* to A_s^Σ. In this case, there must be some $\bar{a}$ such that $A_s^\Sigma \models \phi(\bar{a})$, but no $\bar{b}$ such that $A_s^\Sigma \models \psi(\bar{a}, \bar{b})$. For every such $\bar{a}$, we say that ξ *applies* to A^Σ on $\bar{a}$. There may be several constraints in Σ that apply to A_s^Σ and for each of them, several tuples they apply on. Of these a constraint ξ and a tuple $\bar{a}$ are chosen by some total order on the pairs $(\xi, \bar{a})$. Then A_{s+1}^Σ is obtained from A_s^Σ as follows. If ξ is a TGD, then we add to $\mathrm{dom}(A_s^\Sigma)$ new variables $\bar{b}$ and to the relations in A_s^Σ the tuples that make up the conclusion $\psi(\bar{a}, \bar{b})$ to get $A_{s+1}^\Sigma \models \psi(\bar{a}, \bar{b})$. If ξ is an EGDs, we may assume that ψ consists of a single equality $u_i = u_j$. If a_i and a_j are two distinct constants, the chase *fails*. Otherwise, $A_{s+1}^\Sigma := h(A_s^\Sigma)$ where h satisfies $h(a_i) = h(a_j) = a_i$ and is the identity on all other values. Either way, $A_{s+1}^\Sigma \models \xi(\bar{a})$ and $A_s^\Sigma \to A_{s+1}^\Sigma$.

Chase: Parents, ancestors and siblings. Assume, the variables $\bar{b}$ are new variables introduced in a chase step that corresponds to a firing of a rule whose precondition is $\phi(\bar{a})$ and which makes $\psi(\bar{a}, \bar{b})$ true, as above. If $b \in \bar{b}$, we call every value in $\bar{a}$ a *parent* of b and any other variable in $\bar{b}$ a *sibling* of b. If Σ has width w and height e, then every $x \in \mathrm{dom}(A^\Sigma)$ has at most $e - 1$ siblings and at most w parents. We take the *ancestor* relation to be the transitive closure of the parent relation. We define the *depth* $\mathrm{depth}(x)$ of a value x to be one more than the depth of its deepest parent and the depth of the values in $\mathrm{dom}(A)$ to be zero. Notice that constants have no ancestors, no siblings, and depth zero. Also notice that a variable may have few ancestors yet may not be introduced by any short chase sequence, for example, because it may not be introduced until many full TGDs fire.

Chase: Order, termination, and universality. If for some A_s^Σ no constraint applies, we say that the chase *terminates* and we set $A^\Sigma := A_s^\Sigma$. If there is no such step, A^Σ is undefined. A^Σ is also undefined when the chase fails. In general, A^Σ depends on the order of the chase, but to keep the notation simple we will not explicitly indicate this order. If A^Σ is defined, then $A^\Sigma \models \Sigma$, $A \to A^\Sigma$, and A^Σ is universal for $\{B : A \to B, B \models \Sigma\}$ [15, 2, 6, 3, 1]. We write $A^{\Sigma,\Sigma'}$ for $(A^\Sigma)^{\Sigma'}$.

Since the chase does not always terminate, it is natural to ask for sufficient conditions for its termination. The following wide, sufficient condition on Σ for the termination of the chase on any instance, weak acyclicity was introduced in [5] and [8].

DEFINITION 1. [8, 5] A *position* is a pair (R, i) (which we write R^i) where R is a relation symbol of arity r and i satisfies $1 \leq i \leq r$. We say that x occurs in R^i in ϕ if there is an atom of the form $R(\ldots, x, \ldots)$ in ϕ where x appears in the ith position. The *dependency graph* of a set Σ of TGDs is a directed graph where the vertices are the positions of the relation symbols in Σ and, for every TGD ξ of the form

$$\phi(\bar{u}) \to \exists \bar{v}\, \psi(\bar{u}, \bar{v})$$

there is

1. an edge between R^i and S^j whenever some $u \in \bar{u}$ occurs in R^i in ϕ and in S^j in ψ and
2. an edge between R^i and S^j whenever some $u \in \bar{u}$ appears in R^i in ϕ and some $v \in \bar{v}$ occurs in S^j in ψ. Furthermore, these latter edges are labeled with $\exists$ and we call them *existential edges*.

Σ is *weakly acyclic* if its dependency graph has no cycles with an existential edge. We say that a set Σ of TGDs and EGDs is weakly acyclic if the set of TGDs in Σ is weakly acyclic.

DEFINITION 2. If Σ is a weakly-acyclic set of TGDs, then the *depth* $\mathrm{depth}(R^i)$ of a position R^i of a relation symbol R in Σ is the maximal number of existential edges in a path in the dependency graph for Σ ending at that position. The *depth* of Σ is the maximal depth of a position of a relation symbol in Σ. Notice that chasing a ground instance A with weakly-acyclic TGDs Σ of depth d, results in an instance A^Σ where the depth of every value is at most d.

THEOREM 1 ([8, 5]). *For every weakly-acyclic set Σ of TGDs and EGDs, there are b and c such that, for any A, regardless of the order of the chase and except for the case where the chase fails due to EGDs,*

1. *A^Σ is defined, and*
2. *A^Σ can be computed in $O(|A|^b)$ steps and in time $O(|A|^c)$.*

Data Exchange We consider the setting where we have two schemas σ and τ which do not share any relation symbols. Given an instance S over σ and instance T over τ, the instance (S,T) over $\sigma \cup \tau$ is the instance which has all the relations in S and all those in T. Given a set of constraints Σ over $\sigma \cup \tau$, we say that T is a *solution* for S under Σ if $(S,T) \models \Sigma$. When Σ is clear from context, we simply say that T is a solution for S. We say that U is a *universal solution* for S if it is a solution for S and if it is universal for the set of all solutions for S. As in [8], we assume that source instances of a data exchange problem are ground.

A constraint ξ over (σ,τ) is *source-to-target* if the premise of ξ is over σ and the conclusion of ξ is over τ. Notice that any set of source-to-target TGDs is weakly acyclic. We will consider the special case of settings where $\Sigma = \Sigma_{st} \cup \Sigma_t$ with Σ_{st} a set of source-to-target TGDs and Σ_t a set of TGDs and EGDs. With these restrictions, $(\sigma, \tau, \Sigma_{st}, \Sigma_t)$ is known in the literature [8] as a *data exchange setting*.

THEOREM 2 ([8]). *If $\Sigma := \Sigma_{st} \cup \Sigma_t$ where*

- *Σ_{st} is a set of source-to-target embedded dependencies and*
- *Σ_t is a weakly-acyclic set of TGDs,*

and $(S,\emptyset)^\Sigma$ is defined and is equal to (S,U) for some U, then U is a universal solution for S under Σ.

The following result was given in somewhat different form in [9].

THEOREM 3 ([9]).

1. *If Σ is a set of source-to-target constraints of height e, S is ground, and $(S,T) = (S,\emptyset)^\Sigma$, then $\mathrm{blocksize}(T) \leq e$.*
2. *If $\mathrm{blocksize}(A) \leq c$, then we can check whether $A \to B$ holds in time $O(|B|^c)$.*

5. RETRACTIONS AND CORES

A homomorphism $r : A \to B \subseteq A$ is a *retraction* if r is the identity on $\mathrm{dom}(B)$. That is, r is an idempotent endomorphism. In this case we say that B is a *retract* of A and we write $A \hookrightarrow B$. Clearly, if $A \hookrightarrow B$, then $A \leftrightarrow B$; that is, A and B are homomorphically equivalent. A retraction is *proper* if it is not surjective (which in the case of finite instances is the same not injective). An instance is a *core* if it has no proper retractions. A core C of an instance A is a retract of A which is a core. That is, C is a minimal retract of A. Cores of an instance A are unique up to isomorphism and therefore we can talk about *the core* of A. It follows that A and B are homomorphically equivalent iff their cores are isomorphic.

THEOREM 4. *If Σ is a set of embedded dependencies, then Σ is closed under retractions. That is: if $A \models \Sigma$ and $A \hookrightarrow B$, then $B \models \Sigma$.*

PROOF. Assume ξ is an embedded dependency of the form

$$\phi(\bar{u}) \to \exists \bar{v}\, \psi(\bar{u}, \bar{v})$$

with $\bar{v}$ possibly empty, $A \models \xi$, and $h : A \hookrightarrow B$. Then if $B \models \phi(\bar{b})$ for $\bar{b} \in \mathrm{dom}(B)$, then also $A \models \phi(\bar{b})$. Therefore, since $A \models \xi$, there are $\bar{a} \in \mathrm{dom}(A)$ such that $A \models \psi(\bar{b}, \bar{a})$. This implies $B \models \psi(h(\bar{b}), h(\bar{a}))$ and, since h is a retraction and $\bar{b} \in \mathrm{dom}(B)$, $h(\bar{b}) = \bar{b}$, so $B \models \psi(\bar{b}, \bar{c})$ for $\bar{c} = h(\bar{a})$. That is, $B \models \xi$. □

On the other hand even full dependencies are not closed under endomorphisms, as the following example, adapted from [10], shows.

EXAMPLE 1. Assume A is an instance with a single binary relation R containing the tuples $\{(x,z),(x,a),(z,y),(a,z),(a,a)\}$ where x, y, z are variables and a is a constant, Σ consists of the single constraint

$$R(u,w), R(w,w), R(w,v) \to R(u,v),$$

and $h(x) = x, h(y) = z, h(z) = a, h(a) = a$. Then $A \models \Sigma$ and h is an endomorphism of A, but $h(A)$, which consists of R with tuples $\{(x,a),(a,z),(a,a)\}$ does not satisfy Σ since R does not contain (x,z). □

In general, one can obtain the core of an instance A by successively applying non-surjective endomorphisms. However, if an instance A satisfies some constraints Σ, then even though its core C satisfies Σ by Theorem 4, the image $h(A)$ of A under an endomorphism h may not satisfy Σ as Example 1 shows. In Theorem 9 we will compute the core of an instance U which satisfies some constraints Σ by computing a chain $U = U_0 \supset U_1 \supset U_2 \supset \ldots \supset U_n$, but we will need each U_i to satisfy Σ. Therefore, we will ensure that for each i, $U_i \hookrightarrow U_{i+1}$ and in order to do this, we use the following result, which was essentially proved in [10].

THEOREM 5 ([10]). *Given an endomorphism $h : A \to A$ such that $h(x) = h(y)$ for some $x, y \in \mathrm{dom}(A)$, there is a proper retraction r on A such that $r(x) = r(y)$. Moreover, such retraction can be found in time $O(|\mathrm{dom}(A)|^2)$.*

6. WEAKLY-ACYCLIC TGDS

In this section we prove the following result.

THEOREM 6. *For every $\Sigma := \Sigma_{st} \cup \Sigma_t$ where*

- *Σ_{st} is a set of source-to-target embedded dependencies and*
- *Σ_t is a weakly-acyclic set of TGDs*

and every ground instance S, a core of a universal solution U for S under Σ can be computed in time $O(|\mathrm{dom}(S)|^b)$ for some b which depends only on Σ.

In order to prove Theorem 6, we need several intermediate results. In the proof we chase $(S,\emptyset)$ with Σ_{st} to obtain (S,T), then chase T with Σ_t to obtain U. Then (idea 1) we compute the core of U by successively applying proper retractions (Theorem 9) instead of non-surjective endomorphisms. In order to find such proper retractions efficiently, we (idea 3) identify a set of fragments T_{xy} of U: one such fragment T_{xy} only slightly larger (idea 2) than T satisfying $T \subseteq T_{xy} \subseteq U$ for every pair of distinct values $x, y \in \mathrm{dom}(U)$ (Lemma 1). Finding a homomorphism from T_{xy} to any instance B is easy, since it is easy for T. Furthermore (idea 4), such a homomorphism $T_{xy} \to B$ can be extended to a homomorphism $U \to B$ when $B \models \Sigma$ (Theorem 7). This is enough to show that we can check efficiently whether any $U' \subseteq U$ satisfying $U' \models \Sigma$ has a proper retraction (Theorem 8) since U' has a proper retraction iff there are distinct values $x, y \in \mathrm{dom}(U')$ such that there is a homomorphism $T_{xy} \to U'$. Notice that we need $U' \models \Sigma$ and therefore we compute the core by successively applying proper retractions (cf. Theorem 4) instead of non-surjective

endomorphisms (cf. Example 1). At the end of this section, we present an algorithm that performs all these steps.

The following lemma corresponds to idea 2 in Section 3.

LEMMA 1. *For every weakly-acyclic set Σ of TGDs of depth d, width w, and height e, instance T, and $x, y \in \mathrm{dom}(T^\Sigma)$, there is T_{xy} satisfying*

1. $x, y \in \mathrm{dom}(T_{xy})$,
2. $|\mathrm{dom}(T_{xy})| \leq |\mathrm{dom}(T)| + 2edw^d$,
3. $T \subseteq T_{xy} \subseteq T^\Sigma$,
4. $\mathrm{dom}(T_{xy})$ *is closed under parents and siblings, and*
5. T_{xy} *can be computed in time $O(|\mathrm{dom}(T)|^c)$ for some c which depends only on Σ.*

PROOF. Assume Σ, x, y, and T satisfy the hypotheses. Then every value in T^Σ has depth at most d. If x has nonzero depth, then it was introduced into relations in T^Σ by some step of the chase, which must have fired on some set of tuples of T^Σ.

For any x, set A_x to be x and all its ancestors and E_x to be A_x and all siblings of elements in A_x. That is, E_x is the smallest set containing x which is closed under parents and siblings. An easy induction on depth shows that $|A_x| \leq dw^d$ and $|E_x| \leq edw^d$. We can compute A_x and E_x in time $O(|\mathrm{dom}(T)|^d)$ where d depends only on Σ. Similarly, compute A_y and E_y. Set $T_{xy} := T \cup (T^\Sigma|(E_x \cup E_y))$. Clearly, T_{xy} can be computed in time $O(|\mathrm{dom}(T)|^c)$ for some c that depends only on Σ and it satisfies requirements 1 through 4. □

The following theorem corresponds to idea 4 in Section 3.

THEOREM 7. *If Σ is a set of weakly-acyclic TGDs, and B, T, and W are instances satisfying*

1. $B \models \Sigma$,
2. $T \subseteq W \subseteq T^\Sigma$, *and*
3. $\mathrm{dom}(W)$ *is closed under ancestors and siblings,*

then any homomorphism $h : W \to B$ can be extended in time $O(|\mathrm{dom}(T)|^b)$ to a homomorphism $h' : T^\Sigma \to B$ where b depends only on Σ.

PROOF. Assume 1, 2, and 3 hold and there is a homomorphism $h : W \to B$. Then h can be extended to the desired h' in time $O(|\mathrm{dom}(T)|^b)$ where b depends only on Σ as follows. Assume that the chase of T with Σ terminates in t steps. That is, $T^\Sigma = T_t^\Sigma$. We will compute a sequence of homomorphisms $h_0 := h \subseteq h_1 \subseteq \ldots \subseteq h_t$ such that $h_s : T_s \to B$ where $T_s := T_s^\Sigma \cup W$. Since $T_0^\Sigma = T \subseteq W$, the homomorphism $h_0 = h$ is a homomorphism $T_0 = W \to B$. Since $W \subseteq T^\Sigma$ we have $T^\Sigma = T_t^\Sigma = T_t$ and therefore $h' := h_t$ is the desired extension.

To obtain the homomorphism h_{s+1} from the homomorphism h_s we proceed as follows. Assume that T_{s+1}^Σ is obtained from T_s^Σ by firing a constraint ξ of the form

$$\phi(\bar{u}) \to \exists \bar{v}\, \psi(\bar{u}, \bar{v})$$

on $\bar{a} \in \mathrm{dom}(T_s^\Sigma)$ where in the case of full TGDs $\bar{v}$ is empty. That is $T_s^\Sigma \models \phi(\bar{a})$ and $T_{s+1}^\Sigma \models \psi(\bar{a}, \bar{b})$ for some $\bar{b} \in \mathrm{dom}(T_{s+1}^\Sigma) - \mathrm{dom}(T_s^\Sigma)$. Since h_s is a homomorphism $T_s \to B$ and ϕ is monotonic, $B \models \phi(h_s(\bar{a}))$. The only tuples introduced into relations in T_{s+1}^Σ are those in $\psi(\bar{a}, \bar{b})$. Therefore it is sufficient to define $h_{s+1} \supseteq h_s$ so that $B \models \psi(h_{s+1}(\bar{a}), h_{s+1}(\bar{b}))$.

If ξ is a full TGD, we set $h_{s+1} := h_s$. Since $B \models \Sigma$, we have $B \models \psi(h_s(\bar{a}))$. Otherwise, since T_s is closed under siblings, there are only two cases to consider:

1. $\bar{b} \in \mathrm{dom}(T_s)$ and
2. $\bar{b} \notin \mathrm{dom}(T_s)$.

In case (1) we set $h_{s+1} := h_s$. Since $\bar{b} \in \mathrm{dom}(T_s) - \mathrm{dom}(T_s^\Sigma)$ and $T_s = T_s^\Sigma \cup W$, we must have $\bar{b} \in \mathrm{dom}(W)$. Since W is closed under parents, $\bar{a} \in \mathrm{dom}(W)$ and therefore $W \models \psi(\bar{a}, \bar{b})$. Since $h_0 : T_0 = W \to B$ is a homomorphism, $B \models \psi(h_0(\bar{a}), h_0(\bar{b}))$ and therefore, since $h_{s+1} \supseteq h_0$, $B \models \psi(h_{s+1}(\bar{a}), h_{s+1}(\bar{b}))$.

In case (2) we set $h_{s+1}(x) := h_s(x)$ for any $x \in \mathrm{dom}(T_s)$ and $h_{s+1}(\bar{b}) := \bar{c}$ for some $\bar{c}$ such that $B \models \psi(h_s(\bar{a}), \bar{c})$. Such $\bar{c}$ exists because $B \models \Sigma$. Then $B \models \psi(h_{s+1}(\bar{a}), h_{s+1}(\bar{b}))$.

Since Σ is weakly-acyclic, by Theorem 1 t is $O(|\mathrm{dom}(T)|^p)$ for some p which depends only on Σ and this implies that the extension $h' = h_t$ can be obtained in time $O(|\mathrm{dom}(T)|^b)$ where b depends only on Σ. □

THEOREM 8. *For any weakly-acyclic set Σ of TGDs and instance T, we can check whether any retract U' of $U = T^\Sigma$ has a proper retraction (i.e., whether U' is not a core) and find it in time $O(|\mathrm{dom}(T^\Sigma)|^b)$ where b depends only on Σ and $\mathrm{blocksize}(T)$.*

PROOF. For every $x, y \in \mathrm{dom}(U')$, compute T_{xy} with the properties given in Lemma 1 and test whether there is a homomorphism $h : T_{xy} \to U'$ such that $h(x) = h(y)$. Then U' has a proper retraction iff there are such x, y, h by Claims 1 and 2 below.

Such T_{xy} exist and can be computed in time $O(|\mathrm{dom}(T)|^c)$ for some c which depends only on Σ by Lemma 1. Therefore, since there are at most $|\mathrm{dom}(U')|^2$ pairs (x, y), the result follows from Claims 3 and 4 below.

Claim 1: *r is a proper retraction on U' iff there are $x, y \in \mathrm{dom}(U')$ such that $r(x) = r(y)$.* This is obvious.

The following claim corresponds to idea 3.

Claim 2: *If $x, y \in \mathrm{dom}(U')$, then there is a homomorphism $h : T_{xy} \to U'$ such that $h(x) = h(y)$ iff there is a retraction r on U' such that $r(x) = r(y)$.*

Proof. Since U' is a retract of U and $U \models \Sigma$, we have $U' \models \Sigma$ by Theorem 4. Therefore, if there is a homomorphism $h : T_{xy} \to U'$ such that $h(x) = h(y)$, then h can be extended to a homomorphism $h' : U \to U'$ by Theorem 7 (remember we have $U = T^\Sigma$ and we can set $W := T_{xy}$ and $B := U'$ to satisfy the hypotheses of the theorem). Then $h'' := h'|U'$ is an endomorphism of U' with $h''(x) = h''(y)$. By Theorem 5, there is a retraction r of U' such that $r(x) = r(y)$.

Conversely, if there is a retraction r on U' such that $r(x) = r(y)$, then since U' is a retract of U, we know that there is a retraction $r' : U \hookrightarrow U'$. If there is also a retraction r on U' such that $r(x) = r(y)$, then $r'' = r \circ r'$ satisfies $r''(x) = r''(y)$. Therefore $h := r''|T_{xy}$ is a homomorphism $T_{xy} \to U'$ satisfying $h(x) = h(y)$.

Claim 3: *Given $x, y \in \mathrm{dom}(U')$ and a homomorphism $h : T_{xy} \to U'$ such that $h(x) = h(y)$, a retraction r on U' such that $r(x) = r(y)$ can be found in time $O(|\mathrm{dom}(U)|^c)$ for some c which depends only on Σ.*

Proof. This follows directly from the proof of Claim 2, since Lemma 1, Theorem 7, and Theorem 5 guarantee that h', h'', and r can all be found in time $O(|\mathrm{dom}(U)|^{c'})$ for some c' which depends only on Σ.

Claim 4: *Checking whether, for any $x, y \in \mathrm{dom}(U')$, there is a homomorphism $h : T_{xy} \to U'$ such that $h(x) = h(y)$ (and if so, finding it) can be done in time $O(|\mathrm{dom}(U)|^{c''})$ for some c'' which depends only on Σ and $blocksize(T)$.*

Proof. Set $s := \mathrm{blocksize}(T)$. Since $|\mathrm{dom}(T_{xy})| \leq |\mathrm{dom}(T)| + 2edw^d$ by Lemma 1, the set

$$\{T_{xy} : x, y \in \mathrm{dom}(U'), T \in K\}$$

has block size bound $s + 2edw^d$ and therefore Theorem 3 implies the claim except for the additional requirement that $h(x) = h(y)$. Handling this additional requirement is straightforward. □

THEOREM 9. *For every s and weakly-acyclic set Σ of TGDs, there is b such that for any T with $\mathrm{blocksize}(T) \leq s$, the core of T^Σ can be computed in time $O(|\mathrm{dom}(T)|^b)$.*

PROOF. Set $U := T^\Sigma$. By Theorem 1, U can be computed in time $O(|\mathrm{dom}(T)|)^d$ for some d that depends only on s and Σ. To compute the core C of U efficiently we set $U_0 := U$ and we compute a sequence $U_0, U_1, \ldots, U_n$ such that

1. $U_0 \supset U_1 \supset \ldots \supset U_n$,
2. $U_0 \hookrightarrow U_1 \hookrightarrow \ldots \hookrightarrow U_n$,
3. $U_0, U_1, \ldots, U_n \models \Sigma$, and
4. U_n is a core.

Then U_n is the core of U. Given U_m satisfying 1 and 2 above, we compute U_{m+1} as follows. We check whether U_m has a proper retraction r and find it. By Theorem 8, this can be done in time $O(|\mathrm{dom}(U)|^c)$ for some c which depends only on s and Σ. If so, we set $U_{m+1} := r(U_m)$. Then U_{m+1} satisfies 1 and 2 above. Since Σ is closed under retractions by Theorem 4, U_{m+1} also satisfies 3 above. If U_m has no proper retraction, set $n := m$. By 1, we must have $n \leq |\mathrm{dom}(U)|$ and therefore we can compute U_n, the core of U, in time $O(|\mathrm{dom}(U)|^{c+1})$ from which the result follows. □

PROOF. (**Theorem 6**) Assume Σ and S satisfy the hypotheses. First compute $(S, \emptyset)^{\Sigma_{st}}$, which is equal to (S, T) for some T. Next compute $U := T^{\Sigma_t}$. By Theorem 1, $(S, \emptyset)^{\Sigma_{st}}$ and T^{Σ_t} are well defined and can be computed in time $O(|\mathrm{dom}(S)|^c)$ where c depends only on Σ. We have

$$(S, \emptyset)^\Sigma = (S, \emptyset)^{\Sigma_{st}, \Sigma_t} = (S, T)^{\Sigma_t} = (S, T^{\Sigma_t}) = (S, U)$$

and therefore, by Theorem 2, U is a universal solution for S under Σ. By Theorem 3 for any fixed Σ, the set of all T obtained as above has bounded block size. Therefore, by Theorem 9 there exists b such that for any S the core of U can be computed in time $O(|\mathrm{dom}(T)|^b)$. The result follows from Theorem 1. □

The procedure FINDCORE computes the core of a universal solution for a ground instance S under a set Σ as above in time $O(|\mathrm{dom}(S)|^b)$ for some b which depends only on Σ. The correctness and efficiency of FINDCORE follow from Theorem 6 above.

Procedure FINDCORE
Input: Source ground instance S of a data exchange problem
Output: Core of a universal solution for S

1. Chase $(S, \emptyset)$ with Σ_{st} to obtain $(S, T) = (S, \emptyset)^{\Sigma_{st}}$.
2. Chase T with Σ_t to obtain $U := T^{\Sigma_t}$.
3. For every x, y where $x \in \mathrm{var}(U)$, $y \in \mathrm{dom}(U)$, and $x \neq y$
4. Compute T_{xy} as in Lemma 1.
5. Look for a homomorphism $h : T_{xy} \to U$ such that $h(x) = h(y)$.
6. If there is such h, then
7. Extend h to an endomorphism h' on U.
8. From h', compute a proper retraction r on U.
9. Set $U := r(U)$.
10. Repeat from step 3.
11. Return U.

7. ADDING EGDS

In this section we prove the following result.

THEOREM 10. *For every $\Sigma := \Sigma_{st} \cup \Sigma_t$ where*

- *Σ_{st} is a set of source-to-target embedded dependencies and*
- *Σ_t is a weakly-acyclic set of TGDs and EGDs.*

and every ground source instance S, a core of a universal solution U for S under Σ can be computed in time $O(|\mathrm{dom}(S)|^b)$ where b depends only on Σ.

In order to prove Theorem 10, we introduce a set $\bar{\Sigma}$ of TGDs which 'simulates' a set Σ of TGDs and EGDs (idea 5), in particular in the sense that the core of a solution for S under Σ is the same as the core of a solution for S under $\bar{\Sigma}$ (Lemma 2). If Σ is a weakly-acyclic set of TGDs and EGDs, $\bar{\Sigma}$ is not necessarily weakly-acyclic and Example 2 shows that the chase with $\bar{\Sigma}$ does not terminate on all orders. Nevertheless, we show that for a certain class of chase orders which we call *nice* (idea 6) the chase is guaranteed to terminate whenever Σ is weakly-acyclic (Theorem 11). This is enough to complete the proof of Theorem 10.

The procedure to compute the core of a universal solution for S under a set Σ as in Theorem 6 is the same as FINDCORE (see Section 6) except that it includes an initial step to compute $\bar{\Sigma}$. After that, it proceeds with $\bar{\Sigma}$ instead of Σ.

Given a set of TGDs and EGDs Σ over some signature τ, we define $\bar{\Sigma}$ over the signature $\tau \cup \{E\}$ where E is a new binary relation symbol by the following replacements on Σ.

1. Replace all equations with E-atoms. That is, replace $x = y$ with $E(x, y)$.
2. Add the *equality* constraints:
   (a) $E(x, y) \to E(y, x)$
   (b) $E(x, y), E(y, z) \to E(x, z)$
   (c) $R(x_1, \ldots x_k) \to E(x_i, x_i)$ for every $R \in \tau$ and $i \in \{1, \ldots, k\}$ where k is the arity of R
3. Add the *consistency* constraints
   (d) $R(x_1, \ldots x_k), E(x_i, y) \to R(x_1, \ldots, y, \ldots x_k)$ for every $R \in \tau$ and $i \in \{1, \ldots, k\}$ where k is the arity of R and where y appears in the same position in R as x_i.

In step 1, EGDs are replaced by full TGDs which simulate them. For example, "$R(x, y), R(x, z) \to y = z$" becomes "$R(x, y), R(x, z) \to E(y, z)$."

The auxiliary relation E is not essential; $\bar{\Sigma}$ can be replaced with the constraints obtained from $\bar{\Sigma}$ by removing the equality constraint (2b) and then using resolution through E on the remaining constraints. However, the presentation is easier with an explicit relation E.

Σ and $\bar{\Sigma}$ are over different signatures, but to simplify the presentation we will pretend that a model A of Σ also contains the relation E given by the identity relation on its domain. As a result of this convention, $A \models \Sigma$ implies $A \models \bar{\Sigma}$. Conversely, if E^A is the identity on $\mathrm{dom}(A)$ and $A \models \bar{\Sigma}$, then $A \models \Sigma$.

LEMMA 2. *Under the hypotheses of Theorem 10, $\bar{\Sigma}$ consists only of source-to-target embedded dependencies and a set (not necessarily weakly-acyclic) of target TGDs such that C is the core of a universal solution for Σ iff C is the core of a universal solution for $\bar{\Sigma}$.*

PROOF. If U is a universal solution for S under Σ, then U is also a solution for S under $\bar{\Sigma}$. Furthermore, U is universal for the solutions for S under $\bar{\Sigma}$ as follows. Assume that T is a solution for S under $\bar{\Sigma}$. Pick an element $\hat{x}$ from every equivalence class $[x]$ under E and define e so that $e(y) = \hat{x}$ whenever $y \in [x]$. It is easy to check that e is a retraction. Then $T' = e(T)$ is a solution for S under $\bar{\Sigma}$ and $E^{T'}$ is the identity on $\mathrm{dom}(T')$ and therefore T' is

also a solution for S under Σ. Therefore, there is a homomorphism $h : U \to T' \subseteq T$, so h is also a homomorphism $U \to T$. This shows that U is also a universal solution for S under $\bar{\Sigma}$.

Conversely, assume U is a universal solution for S under $\bar{\Sigma}$. Define e as above for U instead of T. Then $U' = e(U)$ is a solution for S under Σ. Furthermore, U' is universal for the solutions for S under Σ as follows. Assume that T is a solution for S under Σ. Then T is also a solution for S under $\bar{\Sigma}$ and therefore there is a homomorphism $h : U \to T$. It follows that $h' := h|U'$ is a homomorphism $U' \to T$.

Then if C is the core of U, it must also be the core of U' since U' is a retract of U and therefore homomorphically equivalent to U. □

If Σ is a weakly-acyclic set of TGDs and EGDs, $\bar{\Sigma}$ is not necessarily weakly-acyclic. It is natural to wonder whether $A^{\bar{\Sigma}}$ is defined for any A; that is, whether the chase with $\bar{\Sigma}$ terminates *for all orders* as it does for Σ (see Theorem 1). The following example shows that this is not so.

EXAMPLE 2. Assume that Σ consists of the constraints:

$R(x,y)$	$\to$	$\exists z\, T(x,y,z)$
$S(x,z)$	$\to$	$\exists y\, T(x,y,z)$
$T(x,y,z), T(x,u,v)$	$\to$	$u = y$
$T(x,y,z), T(x,u,v)$	$\to$	$v = z$

Then $\bar{\Sigma}$ consists of the constraints:

ξ_1	$R(x,y)$	$\to$	$\exists z\, T(x,y,z)$
ξ_2	$S(x,z)$	$\to$	$\exists y\, T(x,y,z)$
ξ_3	$T(x,y,z), T(x,u,v)$	$\to$	$E(u,y)$
ξ_4	$T(x,y,z), T(x,u,v)$	$\to$	$E(v,z)$
ξ_5	$E(x,y)$	$\to$	$E(y,x)$
ξ_6	$E(x,y), E(y,z)$	$\to$	$E(x,z)$

together with 7 equality constraints $\alpha_1, \ldots, \alpha_7$ of the kind (c) and 7 consistency constraints $\beta_1, \ldots, \beta_7$ corresponding to positions R^1, R^2, S^1, S^2, T^1, T^2, and T^3.

It is easy to verify that Σ is a weakly-acyclic set of TGDs and EGDs. Yet, $\bar{\Sigma}$ is not weakly acyclic. For example, there is a cycle of length 4 through the positions R^1, T^3, and E^2 where the edges are given by constraints ξ_1, ξ_4, and β_1 and the first edge is existential.

If the instance A contains only the tuples $R(1,2)$ and $S(1,3)$, then the chase of A with $\bar{\Sigma}$ does not terminate for the chase order that applies ξ_1, ξ_2, ξ_3, ξ_4, β_2, and β_4 repeatedly in the pattern shown in Figure 4. Each line in the table indicates a constraint that fired on some tuple and the new tuple that was introduced as a result. We consider variables in alphabetic order. For example the third line below indicates that ξ_3 fired under the assignment $(u,v,x,y,z) := (2,a,1,b,3)$ introducing the tuple $(2,b)$ into the relation E. This chase continues forever. □

Fix some weakly-acyclic set Σ of TGDs and EGDs. Consider the dependency graph associated with the TGDs in Σ. If R is a relation, we say that a tuple $\bar{a}$ is *good* for R if the depth of every value in it is smaller than or equal to the depth of the corresponding position in R. That is, $\mathrm{depth}(a_i) \leq \mathrm{depth}(R^i)$. If $\phi(\bar{x})$ is a conjunction of atoms with variables $\bar{x}$, we say that a tuple $\bar{a}$ of the same arity as $\bar{x}$ is good for ϕ if the depth of every value a_i in it is smaller than or equal to the the depth of every position in ϕ where x_i appears. When R or ϕ are clear from context, we simply say that $\bar{a}$ is good. When we consider $\bar{\Sigma}$, we still use the dependency graph associated with the TGDs in Σ and ignore the E relation. Notice that if a TGD fires on a tuple that is good for its premise, then all tuples introduced by its conclusion are good.

New tuple	constraint	fired on
$T(1,2,a)$	ξ_1	$(1,2)$
$T(1,b,3)$	ξ_2	$(1,3)$
$E(2,b)$	ξ_3	$(2,a,1,b,3)$
$E(3,a)$	ξ_4	$(b,3,1,2,a)$
$R(1,b)$	β_2	$(1,2,b)$
$S(1,a)$	β_4	$(1,3,a)$
$T(1,b,c)$	ξ_1	$(1,b)$
$T(1,d,a)$	ξ_2	$(1,a)$
$E(2,d)$	ξ_3	$(2,c,1,d,3)$
$E(3,c)$	ξ_4	$(d,3,1,2,c)$
$R(1,d)$	β_2	$(1,2,d)$
$S(1,c)$	β_4	$(1,3,c)$
$\ldots$		

Figure 4: Non-terminating chase

DEFINITION 3. We say that a chase order is *nice* if whenever several constraints apply, a constraint of the kind which appears earliest in the following list is fired:

1. an equality constraint,
2. a consistency constraint,
3. a constraint firing on a tuple which is good for its premise,
4. a constraint firing on a tuple which is bad for its premise.

EXAMPLE 3. If we chase A from Example 2 with the constraints Σ from that example in a nice order, we get the terminating chase shown in Figure 5. After the steps shown, all constraints in $\bar{\Sigma}$ are satisfied. □

New tuple	constraint	fired on
$E(1,1)$	α_1	$(1,2)$
$E(2,2)$	α_2	$(1,2)$
$E(3,3)$	α_4	$(1,3)$
$T(1,2,a)$	ξ_1	$(1,2)$
$E(a,a)$	α_7	$(1,2,a)$
$T(1,b,3)$	ξ_2	$(1,3)$
$E(b,b)$	α_6	$(1,b,3)$
$E(2,b)$	ξ_3	$(2,a,1,b,3)$
$E(b,2)$	ξ_5	$(2,b)$
$E(3,a)$	ξ_4	$(b,3,1,2,a)$
$E(a,3)$	ξ_5	$(3,a)$
$R(1,b)$	β_2	$(1,2,b)$
$S(1,a)$	β_4	$(1,3,a)$
$T(1,2,3)$	β_7	$(1,2,a,3)$
$T(1,b,a)$	β_7	$(1,b,3,a)$

Figure 5: Terminating Chase

It turns out that the only constraints which apply to a bad tuple are consistency constraints (Lemma 4), so we never fire constraints of the kind 4 in the definition of nice order. This implies that bad tuples are only introduced by consistency constraints and Theorem 11 below, similar to Theorem 1, follows.

If A is a model of $\bar{\Sigma}$, we write $x \equiv y$ if $E(x,y)$ holds in A. We extend the equivalence relation E on elements of the universe to tuples as follows: if $\bar{a}$ and $\bar{b}$ are two r-tuples, then $\bar{a} \equiv \bar{b}$ iff $a_i \equiv b_i$ for $1 \leq i \leq r$.

LEMMA 3. *If ϕ is a conjunction of relational atoms, A satisfies all equality and consistency constraints, $A \models \bar{a} \equiv \bar{b}$ and $A \models \phi(\bar{a})$, then also $A \models \phi(\bar{b})$.*

PROOF. Assume the hypotheses and furthermore that the variables of ϕ are $\bar{x}$. Then for every atom of the form $R(x_{i_1}, \ldots, x_{i_k})$ in ϕ we have $A \models R(a_{i_1}, \ldots, a_{i_k})$. To show that $A \models \phi(\bar{b})$ it is enough to show that also $A \models R(b_{i_1}, \ldots, b_{i_k})$. Since A satisfies the equality and consistency constraints, $a_{i_1} \equiv b_{i_1}, \ldots, a_{i_k} \equiv b_{i_k}$, and $A \models R(a_{i_1}, \ldots, a_{i_k})$, we also have $A \models R(b_{i_1}, a_{i_2}, \ldots, a_{i_k})$, $A \models R(b_{i_1}, b_{i_2}, a_{i_3}, \ldots, a_{i_k})$, ... $A \models R(b_{i_1}, \ldots, b_{i_k})$ by the consistency constraints, as desired. □

We write $[a]$ for the equivalence class $\{b : b \equiv a\}$ of a. We write $\pi_i R$ for the ith projection $\{c_i : R(\bar{c})\}$ of relation R.

LEMMA 4. *If Σ is a weakly-acyclic set of TGDs and EGDs, then at every step $A_s^{\bar{\Sigma}}$ of the chase of A with $\bar{\Sigma}$ using a nice order such that $A_s^{\bar{\Sigma}}$ satisfies the equality and consistency constraints, the following holds:*

1. *For every relation R, if $a \in \pi_i R$, then $[a] \subseteq \pi_i R$.*
2. *For every relation R, if $a \in \pi_i R$, then there exists $b \equiv a$ such that $\mathrm{depth}(b) \leq \mathrm{depth}(R^i)$.*
3. *If $\phi(\bar{a})$ holds where ϕ is a conjunction of relational atoms, then $\phi(\bar{b})$ holds for a tuple $\bar{b}$ good for ϕ such that $\bar{a} \equiv \bar{b}$.*
4. *If ξ fires on $\bar{a}$, then $\bar{a}$ is good for the premise of ξ.*

PROOF. (1) follows directly from the fact that the consistency constraints are satisfied.

We show 2, 3, and 4 by induction on s. Clearly they hold for $s = 0$ since then all values are constants and have depth 0. Now assume that 2, 3, and 4 hold for $A_r^{\bar{\Sigma}}$ for all $r < s$.

(2) If all values in $A_s^{\bar{\Sigma}}$ are constants, then 2 holds trivially. Otherwise there must be a largest value $r < s$ such that a constraint ξ which is not an equality or consistency constraint fired in $A_r^{\bar{\Sigma}}$. Assume ξ fired on tuple $\bar{a}$. Since 4 holds and $A_r^{\bar{\Sigma}}$ satisfies all equality and consistency constraints because the chase order is nice, $\bar{a}$ is good. Therefore, every new tuple introduced by the conclusion of ξ into a relation R is good for that relation R. This implies that 2 holds for $A_{r+1}^{\bar{\Sigma}}$ The equality and consistency constraints which fire after ξ only add values equivalent to those already in R, so 2 also holds for $A_t^{\bar{\Sigma}}$ for every every t such that $r < t \leq s$.

(3) For every i, pick b_i to be a value of minimal depth in $[a_i]$. Since 2 holds, $\bar{b}$ is good for ϕ. Assume the variables (all free) of ϕ are $\bar{x}$. Then for every atom of the form $R(x_{i_1}, \ldots, x_{i_k})$ in ϕ we have $A_s^{\bar{\Sigma}} \models R(a_{i_1}, \ldots, a_{i_k})$. Since 1 holds, we also have $A_s^{\bar{\Sigma}} \models R(b_{i_1}, \ldots, b_{i_k})$. It follows that $A_s^{\bar{\Sigma}} \models \phi(\bar{b})$.

(4) Assume ξ is of the form $\phi(\bar{x}) \rightarrow \exists \bar{y} \psi(\bar{x}, \bar{y})$. We must have $A_s^{\bar{\Sigma}} \models \phi(\bar{a})$. If $\bar{a}$ is not good, we get a contradiction as follows. By 3 we must have $A_s^{\bar{\Sigma}} \models \phi(\bar{b})$ for a tuple $\bar{b}$ good for ϕ such that $\bar{a} \equiv \bar{b}$. Since we are chasing with a nice order, we must have $A_s^{\bar{\Sigma}} \models \xi(\bar{b})$ and this implies that there must be $\bar{c}$ such that $A_s^{\bar{\Sigma}} \models \psi(\bar{b}, \bar{c})$. By Lemma 3, we have $A_s^{\bar{\Sigma}} \models \psi(\bar{a}, \bar{c})$. That is, $A_s^{\bar{\Sigma}} \models \xi(\bar{a})$ so ξ can not fire on $\bar{a}$. □

LEMMA 5. *If Σ is a weakly-acyclic set of TGDs and EGDs of depth d, then for any s all values in $A_s^{\bar{\Sigma}}$ obtained by using a nice chase order have depth $\leq d$.*

PROOF. This follows from Lemma 4 part 4, since constraints which fire on tuples which are good for their premise introduce tuples which are good for the relations they are introduced into. □

THEOREM 11. *For every weakly-acyclic set Σ of TGDs and EGDs, any instance A, and any nice chase order,*

1. *$A^{\bar{\Sigma}}$ is defined, and*
2. *$A^{\bar{\Sigma}}$ can be computed in $O(|\mathrm{dom}(A)|^b)$ steps and in time $O(|\mathrm{dom}(A)|^c)$ where b and c depend only on Σ.*

PROOF. Immediate from Lemma 5. □

THEOREM 12. *For every weakly-acyclic set Σ of TGDs and EGDs, any instance T, and any retract U' of $U = T^{\Sigma}$, we can check whether U' has a proper retraction (i.e., whether U' is not a core) and find it in time $O(|\mathrm{dom}(T^{\Sigma})|^b)$ where b depends only on Σ and the block size of T.*

PROOF. Similar to that of Theorem 8, except using Theorem 11 instead of Theorem 1. □

THEOREM 13. *For every weakly-acyclic set Σ of TGDs and EGDs and any instance T, the core of T^{Σ} for a nice chase order can be computed in time $O(|\mathrm{dom}(T)|^b)$ where b depends only on Σ and the block size of T.*

PROOF. To compute such a core, we replace Σ with $\bar{\Sigma}$ and proceed as in the proof of Theorem 9, except we use Theorem 12 instead of Theorem 8. By Lemma 2, the core so computed using $\bar{\Sigma}$ is the desired core. □

PROOF. (**Theorem 10**) Similar to that of Theorem 6, using Theorem 13 instead of Theorem 9. □

8. REFERENCES

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

[2] Aho, Beeri, and Ullman. The theory of joins in relational databases. *ACM TODS*, 4(3):297–314, 1979.

[3] Beeri and Vardi. A proof procedure for data dependencies. *JACM*, 1984.

[4] A. K. Chandra and P. M. Merlin. Optimal implementation of conjunctive queries in relational data bases. In *ACM STOC*, pages 77–90, 1977.

[5] A. Deutsch and V. Tannen. Reformulation of XML Queries and Constraints. In *ICDT*, 2003.

[6] R. Fagin. Horn clauses and database dependencies. *J. ACM*, 29(4):952–985, 1982.

[7] R. Fagin. Extending the core greedy algorithm to allow target tgds with singleton left-hand sides. 2005. Unpubl. Manuscript, 2005.

[8] R. Fagin, P. G. Kolaitis, R. J. Miller, and L. Popa. Data Exchange: Semantics and Query Answering. In *ICDT*, 2003. full version in: Theor. Comput. Sci. 336(1): 89-124 (2005).

[9] R. Fagin, P. G. Kolaitis, and L. Popa. Data Exchange: Getting to the Core. In *ACM PODS*, pages 90–101, 2003. "Full version in ACM TODS, 30(1):147-210(2005)".

[10] G. Gottlob. Computing cores for data exchange: New algoritms and practical solutions. In *PODS*, 2005. Extended version of the present paper. Currently available at: www.dbai.tuwien.ac.at/staff/gottlob/extcore.pdf.

[11] G. Gottlob and C. G. Fermüller. Removing redundancy from a clause. *Artif. Intell.*, 61(2):263–289, 1993.

[12] G. Gottlob, N. Leone, and F. Scarcello. Hypertree decompositions and tractable queries. *JCSS*, 64(3):579–627, 2002.

[13] P. Hell and J. Nešetřil. The core of a graph. *Discr. Math.*, 109(1-3):117–126, 1992.

[14] P. Kolaitis. Schema mappings, data exchange and metadata management. In *PODS*, 2005.

[15] Maier, Mendelzon, and Sagiv. Testing implication of data dependencies. *ACM TODS*, 1979.