Information Systems 48 (2015) 113–129

Contents lists available at ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/infosys

On the undecidability of the equivalence of second-order tuple generating dependencies

Ingo Feinerer, Reinhard Pichler, Emanuel Sallinger, Vadim Savenkov*

Vienna University of Technology, Vienna, Austria

ARTICLE INFO

Article history:
Received 20 April 2012
Received in revised form
20 February 2014
Accepted 9 September 2014
Recommended by: M. Lenzerini
Available online 18 September 2014

Keywords:
Schema mapping optimization
Database dependencies
Data integration
Data exchange

ABSTRACT

Second-order tuple generating dependencies (SO tgds) were introduced by Fagin et al. to capture the composition of simple schema mappings. Testing the equivalence of SO tgds would be important for applications like model management and mapping optimization. However, we prove the undecidability of the logical equivalence of SO tgds. Moreover, under weak additional assumptions, we also show the undecidability of a relaxed notion of equivalence between two SO tgds, namely the so-called conjunctive query equivalence.

1. Introduction

Schema mappings play an important role in several areas of database research, notably in data integration [1], data exchange [2], peer data management [3], and model management [4]. A schema mapping is given by two schemas, called the *source schema* and the *target schema*, as well as a set of *dependencies* describing the relationship between the source and target schema. The most fundamental form of schema mappings is mappings defined by a set of source-to-target tuple generating dependencies (s–t tgds): they are first-order formulas of the form $\forall \overline{x}(\varphi(\overline{x}) \rightarrow \exists \overline{y}\psi(\overline{x},\overline{y}))$, where the antecedent $\varphi(\overline{x})$ is a conjunctive query (CQ) over the source schema and the conclusion $\psi(\overline{x},\overline{y})$ is a CQ over the target schema. Intuitively, such an s–t tgd defines a constraint that the presence of certain tuples in the source database I (namely those in the image of some homomorphism h from $\varphi(\overline{x})$ to I) enforces the presence of certain tuples in the target database J (s.t. h can be extended to a homomorphism from $\psi(\overline{x},\overline{y})$ to J).

Several algebraic operators [4,5] on schema mappings have been intensively studied in recent time like computing inverses [6–9] and composing schema mappings [10,11,3,12]. Our work is rather related to the composition operator. Fagin et al. proved that, in general, s–t tgds are not powerful enough to express the composition of two mappings defined by s–t tgds [11]. To remedy this defect, the so-called second-order tuple generating dependencies (SO tgds) were introduced in [11]. SO tgds extend s–t tgds by existentially quantified function-variables and equalities of (possibly functional) terms in the antecedents of implications. Details and formal definitions are given in Section 2. It was shown in [11] that SO tgds capture exactly the closure under composition of mappings defined by s–t tgds.

Example 1 (*Fagin et al. [11]*). Consider the following three schemas. Let $\mathbf{S}_1$ consist of the unary relation symbol $\mathrm{Emp}(\cdot)$ of employees. Schema $\mathbf{S}_2$ consists of a single binary relation symbol $\mathrm{Mgr}'(\cdot,\cdot)$ that associates each employee with a manager. Schema $\mathbf{S}_3$ consists of a similar binary relation

* Corresponding author.
E-mail addresses: feinerer@dbai.tuwien.ac.at (I. Feinerer), pichler@dbai.tuwien.ac.at (R. Pichler), sallinger@dbai.tuwien.ac.at (E. Sallinger), savenkov@dbai.tuwien.ac.at (V. Savenkov).

http://dx.doi.org/10.1016/j.is.2014.09.003
0306-4379/

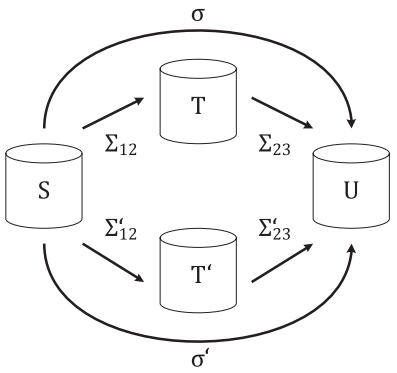

Fig. 1. Mapping compositions.

symbol $\mathsf{Mgr}(\cdot,\cdot)$ that is intended to provide a copy of Mgr' and an additional unary relation symbol $\mathsf{SelfMgr}(\cdot)$ to store employees who are their own manager.

Consider the mappings $\mathcal{M}_{12}=(\mathbf{S}_1,\mathbf{S}_2,\Sigma_{12})$ and $\mathcal{M}_{23}=(\mathbf{S}_2,\mathbf{S}_3,\Sigma_{23})$ with

$$\Sigma_{12}=\{\forall e(\mathsf{Emp}(e)\rightarrow\exists m\,\mathsf{Mgr}'(e,m))\}$$

and

$$\Sigma_{23}=\{\forall e,m(\mathsf{Mgr}'(e,m)\rightarrow\mathsf{Mgr}(e,m)),\ \forall e(\mathsf{Mgr}'(e,e)\rightarrow\mathsf{SelfMgr}(e))\}.$$

We are looking for the composition of $\mathcal{M}_{12}$ and $\mathcal{M}_{23}$. It can be verified that this composition can be expressed by the SO tgd

$$\begin{aligned}\sigma=\exists f(&\forall e\,(\mathsf{Emp}(e)\rightarrow\mathsf{Mgr}(e,f(e)))\\&\wedge\forall e\,(\mathsf{Emp}(e)\\&\wedge(e=f(e))\rightarrow\mathsf{SelfMgr}(e))).\end{aligned}$$

In this paper, we want to study the *equivalence of SO tgds*. Note that the question of equivalence naturally arises in several scenarios. Fig. 1 illustrates a *model evolution* scenario, where data structured under some schema **S** is first migrated to a database with schema η and then further transformed to meet schema **U**. Now suppose that there exists an alternative migration path from schema **S** via $\mathbf{T}'$ to schema **U**. The question if the two migration paths yield the same result comes down to checking if the dependencies σ and σ' (which represent the respective mapping compositions) are equivalent. Actually, Fig. 1 can also be thought of as illustrating a *peer data management* scenario, where some peer with data structured according to **S** provides part of its data to some other peer with schema η (resp. $\mathbf{T}'$). The latter peer in turn passes this data on to yet another peer with schema **U**. Now suppose that a user may access the data only at the peer with schema **U**. What happens if some link in this peer data network is broken, say the one corresponding to mapping Σ_{23}? Will the path of mappings from **S** via $\mathbf{T}'$ to **U** still give the user full access to the data provided by the peer with schema **S**? Testing the equivalence of σ and σ' is thus crucial for answering questions of redundancy and reliability in a peer data network.

The equivalence of mappings is also fundamental to mapping optimization. As mentioned in [13], optimizing a mapping ultimately means replacing the mapping by an "*equivalent*" one with better (computational) properties. This raises the question of how the "*equivalence*" of two mappings should be defined. Since dependencies are logical formulas, the most natural notion of equivalence is *logical equivalence*. In this paper, we show that logical equivalence of SO tgds is undecidable. In order to allow for more flexibility in optimizing mappings, Fagin et al. introduced relaxed notions of equivalence [13]. In particular, the potential of *conjunctive query* (CQ) equivalence for optimizing several kinds of mappings was studied in [13]. Intuitively, two mappings are CQ-equivalent, if conjunctive queries posed against the target database yield the same result for both mappings (for details, see Section 2). For instance, it was shown in [11] that the composition of the mappings $\mathcal{M}_{12}$ and $\mathcal{M}_{23}$ in Example 1 cannot be represented by an SO tgd without equalities in the antecedent. On the other hand, σ in Example 1 is CQ-equivalent to $\sigma'=\exists f(\forall e\,(\mathsf{Emp}(e)\rightarrow\mathsf{Mgr}(e,f(e)))$. Under the weak additional assumption that the source schema may have key dependencies, we shall prove the undecidability of CQ-equivalence of SO tgds.

State of the art: The equivalence of schema mappings has been an active line of research in recent years, initiated by Fagin et al. [13]. In particular, the authors introduced two relaxed notions of equivalence, namely CQ-equivalence and DE-equivalence (data exchange equivalence). Together with logical equivalence, we thus have a hierarchy of notions of equivalence, with logical equivalence being the most restrictive, and CQ-equivalence being the least restrictive. Recently, a relativization of CQ-equivalence, namely bounded CQ-equivalence (or CQ_n−equivalence) has been introduced, which describes equivalence of schema mappings for CQs with up to n variables [14]. For a recent survey on the topic of equivalence and optimization of schema mappings, see [15].

We now discuss the results known for the main concern of this paper, namely whether two given schema mappings are equivalent. For mappings defined by s–t tgds and target dependencies in the form of tgds and egds this problem has been well studied. In the general case, the logical equivalence of such mappings is undecidable. If the target tgds admit a finite chase, then the problem becomes decidable [13], utilizing the chase procedure [16].

For s–t tgds, logical, DE- and CQ-equivalence coincide [13]. Hence, DE- and CQ-equivalence for s–t tgds are of course decidable. In [13], the undecidability of CQ-equivalence was proved for mappings consisting of s–t tgds and a weakly acyclic set of target tgds. The undecidability was extended in [17] in two directions, namely to mappings with target egds and from CQ-equivalence to DE-equivalence. A distinction between the decidability of DE- and CQ-equivalence was shown in [17], as DE-equivalence is decidable for mappings based on s–t tgds and a weakly acyclic set of functional and inclusion dependencies, while CQ-equivalence is undecidable for s–t tgds with just a single key dependency per relation.

Altogether, the equivalence of schema mappings has been studied for the most important first-order schema mapping languages. The equivalence of SO tgds has not been studied so far.

A different, yet related problem is deciding whether a given schema mapping in one class of schema mappings is