Information Processing and Management 49 (2013) 465–483

Contents lists available at SciVerse ScienceDirect

Information Processing and Management

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

XML data exchange with target constraints

Zijing Tan *, Liyong Zhang, Wei Wang, Baile Shi

School of Computer Science, Fudan University, Shanghai, China

ARTICLE INFO

Article history:
Received 5 December 2011
Received in revised form 5 September 2012
Accepted 11 September 2012
Available online 1 November 2012

Keywords:
XML
Data exchange
Constraints

ABSTRACT

Data exchange is the problem of taking data structured under a source schema and creating an instance of a target schema, by following a mapping between the two schemas. There is a rich literature on problems related to data exchange, *e.g.,* the design of a schema mapping language, the consistency of schema mappings, operations on mappings, and query answering over mappings. Data exchange is extensively studied on relational model, and is also recently discussed for XML data. This article investigates the construction of target instance for XML data exchange, which has received far less attention. We first present a rich language for the definition of schema mappings, which allow one to use various forms of document navigation and specify conditions on data values. Given a schema mapping, we then provide an algorithm to construct a canonical target instance. The schema mapping alone is not adequate for expressing target semantics, and hence, the canonical instance is in general not optimal. We recognize that target constraints play a crucial role in the generation of good solutions. In light of this, we employ a general XML constraint model to define target constraints. Structural constraints and keys are used to identify a certain entity, as rules for data merging. Moreover, we develop techniques to enforce non-key constraints on the canonical target instance, by providing a chase method to reason about data. Experimental results show that our algorithms scale well, and are effective in producing target instances of good quality.

1. Introduction

An important issue in modern information systems and e-commerce applications is to provide support for inter-operability of independent data sources. In *data exchange*, data structured under one schema (source schema) are restructured and translated into an instance of a different schema (target schema). The restructuring should follow a specification, known as a *schema mapping*, which describes the relationship between the source schema and the target schema. Data exchange is used in many tasks that require data to be transferred between existing and independently created applications, often in e-business applications. Two applications can exchange data by directly transferring data following one schema to the other schema. Alternatively, several applications can agree on a standard data schema, and then exchange data between them by converting their own data to/from this form.

The key tasks in data exchange can be roughly divided into two groups.

1. Intensional tasks, which concern the management of the schema mapping between the source and target schema. Among others, these tasks involve (a) the design of the mapping language, to assess the trade-off between the expressive power

* Corresponding author. Tel./fax: +86 21 6564 2219.
E-mail addresses: zjtan@fudan.edu.cn (Z. Tan), 09210240049@fudan.edu.cn (L. Zhang), weiwang1@fudan.edu.cn (W. Wang), bshi@fudan.edu.cn (B. Shi).

0306-4573/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.ipm.2012.09.002

and the efficiency of implementation, (b) the static analysis of the mapping, *e.g.,* the consistency of a mapping and the containment of different mappings, and (c) the operations of the mappings, *e.g.,* the inversion and composition.

2. Extensional (data-level) tasks, for the defined mapping between the source and target schema and the given source instance. These problems mostly involve (a) the construction (materialization) of target instances, and (b) query answering against the target schema, in a way that is consistent with the source data.

Data exchange has received an increasing attention from both the research community and the tool market, due to the extensive need for exchange of data with the prevalent use of the Web. After the theoretical foundation of relational data exchange is laid (Fagin, Kolaitis, Miller, & Popa, 2005; Fagin, Kolaitis, & Popa, 2005), various issues in data exchange are studied (Barcelo, 2009; Bernstein & Melnik, 2007). Practical data exchange system Clio (Miller et al., 2001; Popa, Velegrakis, Miller, Hernandez, & Fagin, 2002) is also built and partly incorporated into commercial systems.

Most existing researches have focused on the relational model, thus much less is known about XML data exchange. While commercial systems often claim to provide support for XML, this is typically implemented either through relational translations, or with simple mappings that cannot change document structure. XML is becoming one of the dominating standard for information representation and interchange on the Web; this motivates the quest for effective methods to handle the ever-growing XML data with data exchange settings.

This article studies the problem of XML data exchange, to find an instance of a target schema for an instance of a source schema, where both schemas are given by DTDs. Among other things, we identify two key factors in an effective XML data exchange: (a) a rich language for XML schema mapping, to support the document navigation and restructuring, and to bind semantically related values and (b) a powerful constraint model for target constraints, to improve the semantic expressiveness of XML, and to help achieve better target instance.

Example 1. To illustrate the features that need to be modeled in an XML data exchange, consider the source and target DTD schemas given in Fig. 1. Both DTDs describe a set of authors and their publications, and the meanings of labels are self-explanatory. In the DTD graph, (a) each node denotes an element type; (b) an edge (possibly labeled with "$*$") from a node A to a node B shows the parent–child relationship between the two element types; and (c) the list of attributes of an element type is given in bracket.

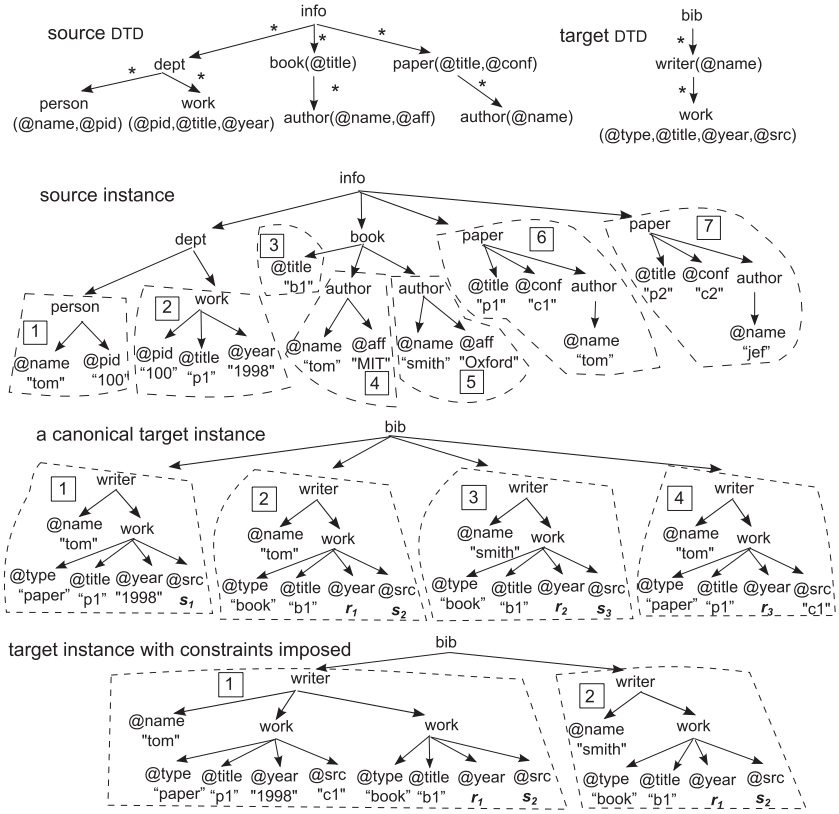

Fig. 1. Example source/target DTDs and instances.