Interscheme Properties' Role in Data Warehouses

Pasquale De Meo
Università "Mediterranea" di Reggio Calabria, Italy

Giorgio Terracina
Università della Calabria, Italy

Domenico Ursino
Università "Mediterranea" di Reggio Calabria, Italy

INTRODUCTION

In this article, we illustrate a general approach for the semi-automatic construction and management of data warehouses. Our approach is particularly suited when the number or the size of involved sources is large and/or when it changes quite frequently over time. Our approach is based mainly on the semi-automatic derivation of interschema properties (i.e., terminological and structural relationships holding among concepts belonging to different input schemas). It consists of the following steps: (1) enrichment of schema descriptions obtained by the semi-automatic extraction of interschema properties; (2) exploitation of derived interschema properties for obtaining in a data repository an integrated and abstracted view of available data; and (3) design of a three-level data warehouse having as its core the derived data repository.

BACKGROUND

In the last years, an enormous increase of data available in electronic form has been witnessed, as well as a corresponding proliferation of query languages, data models, and data management systems. Traditional approaches to data management do not seem to guarantee, in these cases, the needed level of access transparency to stored data while preserving the autonomy of local data sources. This situation contributed to push the development of new architectures for data source interoperability, allowing users to query preexisting autonomous data sources in a way that guarantees model language and location transparency.

In all the architectures for data source interoperability, components handling the reconciliation of involved information sources play a relevant role. In the construction of these components, *schema integration* (Chua, Chiang, & Lim, 2003; dos Santos Mello, Castano & Heuser, 2002; McBrien & Poulovassilis, 2003) plays a key role.

However, when involved systems are numerous and/or large, schema integration alone typically ends up producing a too complex global schema that may, in fact, fail to supply a satisfactory and convenient description of available data. In these cases, schema integration steps must be completed by executing *schema abstraction* steps (Palopoli, Pontieri, Terracina & Ursino, 2000). Carrying out a schema abstraction activity amounts to clustering objects belonging to a schema into homogeneous subsets and producing an abstracted schema obtained by substituting each subset with one single object representing it.

In order for schema integration and abstraction to be correctly carried out, the designer has to understand clearly the semantics of involved information sources. One of the most common ways for deriving and representing schema semantics consists in detecting the so-called *interschema properties* (Castano, De Antonellis & De Capitani di Vimercati, 2001; Doan, Madhavan, Dhamankar, Domingos & Levy, 2003; Gal, Anaby-Tavor, Trombetta & Montesi, 2004; Madhavan, Bernstein & Rahm, 2001; Melnik, Garcia-Molina & Rahm, 2002; Palopoli, Saccà, Terracina & Ursino, 2003; Palopoli, Terracina & Ursino, 2001; Rahm & Bernstein, 2001). These are terminological and structural properties relating to concepts belonging to different schemas.

In the literature, several manual methods for deriving interschema properties have been proposed (see Batini, Lenzerini, and Navathe (1986) for a survey about this argument). These methods can produce very precise and satisfactory results. However, since they require a great amount of work, to the human expert, they are difficult to be applied when involved sources are numerous and large.

To handle large amounts of data, various semi-automatic methods also have been proposed. These are much less resource consuming than manual ones; moreover,

interschema properties obtained by semi-automatic techniques can be updated and maintained more simply. In the past, semi-automatic methods were based on considering only structural similarities among objects belonging to different schemas. Presently, all interschema property derivation techniques also take into account the context in which schema concepts have been defined (Rahm & Bernstein, 2001).

The dramatic increase of available data sources led also to a large variety of both structured and semi-structured data formats; in order to uniformly manage them, it is necessary to exploit a unified paradigm. In this context, one of the most promising solutions is XML. Due to its semi-structured nature, XML can be exploited as a unifying formalism for handling the interoperability of information sources characterized by heterogeneous data representation formats.

MAIN THRUST

Overview of the Approach

In this article we define a new framework for uniformly and semi-automatically constructing a data warehouse from numerous and large information sources characterized by heterogeneous data representation formats. In more detail, the proposed framework consists of the following steps:

- Translation of involved information sources into XML ones.
- Application to the XML sources derived in the previous step of almost automatic techniques for detecting interschema properties, specifically conceived to operate on XML environments.
- Exploitation of derived interschema properties for constructing an integrated and uniform representation of involved information sources.
- Exploitation of this representation as the core of the reconciled data level of a data warehouse[1].

In the following subsections, we will illustrate the last three steps of this framework. The translation step is not discussed here, because it is performed by applying the translation rules from the involved source formats to XML already proposed in the literature.

Extraction of Interschema Properties

A possible taxonomy classifies interschema properties into terminological properties, subschema similarities, and structural properties.

Terminological properties are synonymies, homonymies, hyponymies, overlappings, and type conflicts. A synonymy between two concepts A and B indicates that they have the same meaning. A homonymy between two concepts A and B indicates that they have the same name but different meanings. Concept A is said to be a hyponym of concept B (which, in turn, is a hypernym of A), if A has a more specific meaning than B. An overlapping exists between concepts A and B, if they are neither synonyms nor hyponyms of the other but share a significant set of properties; more formally, there exists an overlapping between A and B, if there exist non-empty sets of properties $\{p_{A1}, p_{A2}, ..., p_{An}\}$ of A and $\{p_{B1}, p_{B2}, ..., p_{Bn}\}$ of B such that, for $1 \leq i \leq n$, p_{Ai} is a synonym of p_{Bi}. A type conflict indicates that the same concept is represented by different constructs (e.g., an element and an attribute in an XML source) in different schemas.

A subschema similarity represents a similitude between fragments of different schemas.

Structural properties are inclusions and assertions between knowledge patterns. An inclusion between two concepts A and B indicates that the instances of A are a subset of the instances of B. An assertion between knowledge patterns indicates either a subsumption or an equivalence between knowledge patterns. Roughly speaking, knowledge patterns can be seen as views on involved information sources.

Our interschema property extraction approach is characterized by the following features: (1) it is XML-based; (2) it is almost automatic; and (3) it is semantic.

Given two concepts belonging to different information sources, one of the most common ways for determining their semantics consists of examining their neighborhoods, since the concepts and the relationships in which they are involved contribute to define their meaning. In addition, our approach exploits two further indicators for defining in a more precise fashion the semantics of involved data sources. These indicators are the types and the cardinalities of the elements, taking the attributes belonging to the XML schemas into consideration.

It is clear from this reasoning that concept neighborhood plays a crucial role in the interschema property computation. In XML schemas, concepts are expressed by elements or attributes. Since, for the interschema property extraction, it is not important to distinguish concepts represented by elements from concepts represented by attributes, we introduce the term *x-component* for denoting an element or an attribute in an XML schema.

In order to compute the neighborhood of an x-component, it is necessary to define a semantic distance[2] between two x-components of the same schema; it takes