DaWaII: a Tool for the Integration of Autonomous Data Marts

Luca Cabibbo, Ivan Panella, and Riccardo Torlone
Dipartimento di Informatica e Automazione
Università Roma Tre
{cabibbo, panella, torlone}@dia.uniroma3.it

Abstract

DaWaII (Data Warehouse IntegratIon) is a tool for supporting the various activities related to the integration of multidimensional databases. This problem arises in common scenarios where there is the need to combine independently developed data warehouses. The basic facility of the tool is a support for testing the validity of a matching between heterogeneous dimensions, according to a number of desirable properties. Two different strategies are then provided by the tool to perform the actual integration. The first approach refers to a scenario of loosely coupled integration, in which we just need to identify the common information between sources and perform drill-across queries over the original sources. The goal of the second approach is the derivation of a materialized view built by merging the sources, and refers to a scenario of tightly coupled integration in which queries are performed against the view. We illustrate the practical techniques implementing the two strategies and describe the functionality of the tool implementing the approach.

1. Introduction

The problem of integrating heterogeneous multidimensional databases arises in common scenarios in which information from autonomous (i.e., independently developed and operated) data marts need to be combined. Today, a common practice for building a data warehouse is to develop a collection of integrated data marts, each of which provide a dimensional view of a single business process. These data marts should be based on shared dimensions but very often, within the same company, designers develop their data marts independently and it turns out that their integration is a difficult task. The need of combining autonomous data marts arises in other common cases. For instance, when different companies get involved in a federated project or when there is the need to combine a proprietary data warehouse with external information, for instance, with multidimensional data wrapped from the Web.

We started from the observation that, differently from the general case, this problem can be tackled in a more systematic mainly because multidimensional databases are structured in a rather uniform way, along the widely accepted notions of dimension and fact.

We have then studied the problem from a conceptual point of view, by introducing a fundamental notion underlying data mart integration: dimension compatibility [1]. Intuitively, two dimensions (belonging to different data marts) are compatible if their common information is consistent. We have shown that dimension compatibility gives the ability to correlate, in a correct way, multiple data marts by means of *drill-across queries* [2], which are basically joins over common dimensions. Building on this study, in [1] we have then proposed a number of techniques that have been used to implement a practical integration tool for multidimensional databases, similar in spirit to other tools supporting heterogeneous data transformation and integration [3].

The basic issue is the integration of a pair of autonomous dimensions. We have identified a number of desirable properties that a *matching* between two dimensions (that is, a correspondence between their levels) should satisfy: the *coherence* of the hierarchies on levels, the *soundness* of the matched levels, according to the members associated with them, and the *consistency* of the functions that relate members of different levels within the two dimensions.

We propose two different approaches to the problem of integration that try to enforce matchings satisfying the above properties. The first approach refers to a scenario of loosely coupled integration, in which we need to identify the common information between sources (intuitively, the intersection), while preserving their autonomy. In this approach drill-across queries are then performed over the original sources. The goal of the second approach is rather merging the sources (intuitively, making the union) and refers to a scenario of tightly coupled integration, in which we need to build a materialized view that includes the sources. With this approach, queries are then performed against the view built from the sources. As a preliminary

tool, we have introduced a powerful technique, the *chase of dimensions*, that can be used in both approaches to test for consistency and combine the content of the dimensions to integrate.

To our knowledge, DaWaII is the first systematic project on this problem. We also believe that the proposed techniques can be applied in more general contexts in which there is the need to integrate generic heterogenous data sources that are described by means of taxonomies of concepts (e.g., ontologies).

In the rest of this paper we illustrate: the basic issues (in Section 2), the two integration techniques (in Section 3), and the functions offered by our tool (in Section 4). Further details of the overall approach can be found in [1].

2. Matching Autonomous Dimensions

2.1. The framework of reference

We refer to a very general data model for multidimensional databases based on the basic notions of *dimension* and *data mart*. A dimension represents a domain of real-world entities called *members*. Members of a dimension can be the days in a time interval or the products sold by a company. Each dimension is organized into a hierarchy $\preceq$ of *levels*, corresponding to data domains grouping dimension members at different granularity. The members of the bottom element of a dimension (w.r.t. $\preceq$) represent real world entities that are called *ground*. Within a dimension, members at different levels are related through a family of *roll-up functions*. A roll-up function relates the members of a pair of levels by mapping each member having a finer grain (e.g., a product) to a member having a coarser grain (e.g., a brand) according to $\preceq$. A data mart associates *measures* to members of dimensions and is used to represent factual data. For example, the sales of a chain of stores can be represented by a data mart that associates the number of items sold with a product, a day, and a store.

2.2. Properties of dimension matchings

The basic problem of the integration of two autonomous data marts is the definition of a *matching* between their dimensions, that is, an injective partial mapping between the corresponding levels. An example is illustrated in Figure 1, which shows a matching between two heterogeneous geographical dimensions.

We do not address the problem of how this matching is provided. Actually, our tool is able to suggest potential matchings on the basis of a number of heuristics, but this is outside the goal of our research. We refer for this problem to the huge literature on the automatic derivation of matchings.

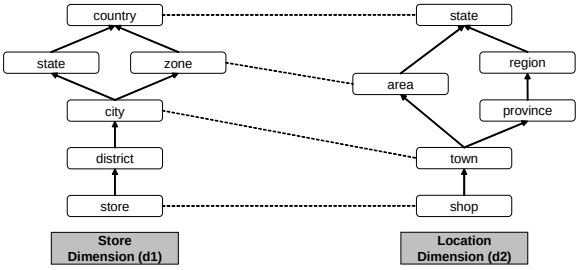

Figure 1. A matching between two dimensions

We have identified a number of desirable properties that a matching μ between two dimensions d_1 and d_2 should satisfay.

- Coherence: μ is *coherent* if, for each pair of levels l, l' of d_1 on which μ is defined, $l \preceq_1 l'$ if and only if $\mu(l) \preceq_2 \mu(l')$;
- Soundness: μ is *sound* if, for each level $l \in L_1$ on which μ is defined, $m_1(l) = m_2(\mu(l))$;
- Consistency: μ is *consistent* if, for each pair of levels $l \preceq_1 l'$ of d_1 on which μ is defined, $\rho_1^{l \to l'} = \rho_2^{\mu(l) \to \mu(l')}$.

A total matching that is coherent, sound and consistent is called a *perfect* matching.

Clearly, a perfect matching is very difficult to achieve in practice. In many cases however, autonomous dimensions actually share some information. The goal of the tool we have developed is the identification of this common information to perform drill-across operations between heterogeneous data marts.

2.3. Chase of dimensions

As a preliminary tool, we have defined a powerful technique, the *d-chase*, that is inspired by an analogous procedure used for reasoning about dependencies in the relational model and is used to test for consistency and to combine the content of the dimensions to be integrated.

Given a matching μ between two dimensions d_1 and d_2, this procedure operates over a special *matching* tableau $T_\mu[d_1, d_2]$ built over the levels of d_1 and d_2. This tableau has a tuple for each ground member m of d_1 and d_2 and includes members associated with m by roll-up functions and possibly *variables* denoting missing information. An example of a matching tableau for the matching between dimensions in Figure 1 is the following.

store	city	zone	country	district	state	prov.	region
1st	NewYork	v_1	USA	v_2	NY	v_3	v_4
2nd	LosAng.	U2	USA	Melrose	CA	v_5	v_6
1er	Paris	E1	France	Marais	v_7	v_8	v_9
1mo	Rome	E1	Italy	v_{10}	v_{11}	RM	Lazio
1st	NewYork	U1	USA	v_{12}	v_{13}	v_{14}	v_{15}
1er	Paris	E1	France	v_{16}	v_{17}	75	IledeFr

In this example, the first three tuples represent members of d_1 and the others members of d_2. The first four columns represent the matched levels and the others represent levels of the two dimensions that have not been matched. Note that a variable occurring in a tableau may represents an unknown value (for instance, in the first row, the zone in which the store *1st* is located, an information not available in the instance of d_1) or an inapplicable value (for instance, in the last row, the district in which the store *1er* is located, a level not present in the scheme of d_2).

The d-chase modifies values in a matching tableau, by applying *chase steps*. A chase step applies when there are two tuples t_1 and t_2 such that $t_1[l] = t_2[l]$ and $t_1[l'] \neq t_2[l']$ for some roll up function from l to l' and modifies the l'-values of t_1 and t_2 as follows: if one of them is a constant and the other is a variable then the variable is changed to the constant, otherwise the values are equated. If a chase step tries to identify two constants, then we say that the d-chase encounters a *contradiction*.

By applying the d-chase procedure to the matching tableau above we do not encounter contradictions and obtain the following tableau.

store	city	zone	country	district	state	prov.	region
1st	NewYork	U1	USA	v_2	NY	v_3	v_4
2nd	LosAng.	U2	USA	Melrose	CA	v_5	v_6
1er	Paris	E1	France	Marais	v_7	75	IledeFr
1mo	Rome	E1	Italy	v_{10}	v_{11}	RM	Lazio

The d-chase provides an effective way to test for consistency since we have shown that a matching μ between two dimensions d_1 and d_2 is consistent if and only if $_{DCHASE_{\rho_1 \cup \mu(\rho_2)}}(T_\mu[d_1, d_1])$ terminates without contradictions. Moreover, it turns out that if we project the result of the chase over the original dimension schemes, we obtain the original instances and, possibly, some additional information derived from the other dimension.

3. Two Approaches to Dimension Integration

We now briefly illustrate two approaches to the problem of the integration of autonomous data marts.

3.1. A loosely coupled approach

In a *loosely coupled integration* scenario, we need to identify the common information between various data sources and perform drill-across queries over the original sources. Therefore, our goal is just to select data that is shared between the sources. Thus, given a pair of dimensions d_1 and d_2 and a matching μ between them, the approach aims at deriving the operations to apply to d_1 and d_2 in order to select the portions of these dimensions that make μ perfect.

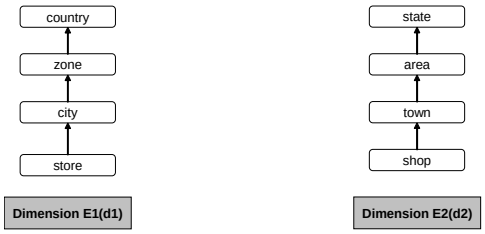

Figure 2. The dimensions generated by the LCI on the matching in Figure 1

In [1], we have proposed an algorithm that generates two expression in a *dimension algebra* that describe, in an abstract way, data manipulations over dimensions. This algorithm is based on three main steps: (i) a test for coherence that takes advantage of the transitivity of $\preceq$, (ii) a test for consistency based on the application of the d-chase, and (iii) the derivation of the selections, projections and aggregations to perform to the input dimensions in order to select common information.

As an example, the application of this algorithm to the dimension matching reported in Figure 1 generates the following expressions.

$$\sigma_{\{1st\}}(\pi_{store,city,zone,country}(d_1)),$$

$$\sigma_{\{1st\}}(\pi_{shop,town,area,state}(d_2)).$$

The schemes of the dimensions we obtain by applying these expressions to the original dimensions are reported in Figure 2.

We have proved that the execution of this algorithm always returns two expressions that correctly compute the intersection of two dimensions if and only if the dimensions are compatible [1].

3.2. A tightly coupled approach

In a *tightly coupled integration*, we want to build a materialized view that combine different data sources and perform queries over this view. The goal is the derivation of new dimensions obtained by merging the dimensions of the original data sources. In this case, given a pair of dimensions d_1 and d_2 and a matching μ between them, the integration technique aims at deriving another dimension ob-

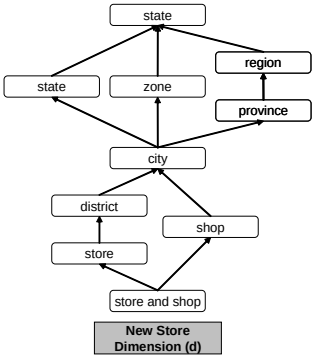

Figure 3. The dimension generated by the TCI algorithm on the matching in Figure 1

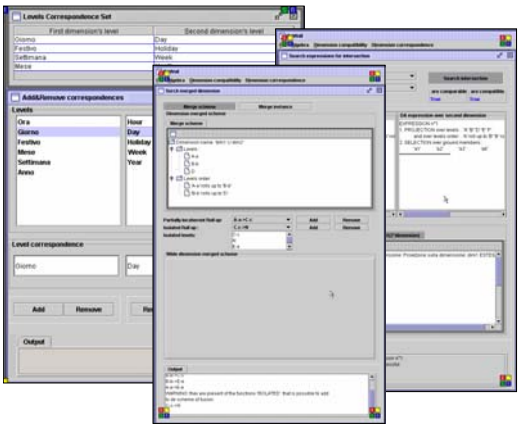

Figure 4. A prototype of the system

tained by merging the levels involved in μ and including, but taking apart, all the other levels.

In [1], we have presented an algorithm that performs this task. This algorithm is also based on three main steps: (i) a test for coherence that takes advantage of the transitivity of $\preceq$, (ii) a test for consistency based on the application of the d-chase, and (iii) the derivation of a new dimension obtained by projecting the result of the d-chase over the "union" of the schemes of the input dimensions. If the union of the schemes produces two minimal levels, the algorithm is more involved since it needs to generate an auxiliary bottom level.

As an example, consider again the matching between dimensions in Figure 1 but assume that the level store does not map to the level shop. This means that the corresponding concepts are not related. It follows that the union of the schemes of the two dimensions produces two minimal levels. In this case, the application of algorithm to this matching introduces a new bottom level below store and shop. The scheme of the dimension generated by the algorithm is reported in Figure 3.

We have shown that the execution of this algorithm always returns a dimensions d that "embeds" the original dimensions, in the sense that they can be obtained by applying a dimension expression over d [1].

4. The system

To test the effectiveness of our approach, we have designed and developed in Java an interactive tool for the integration of multidimensional databases, called DaWaII (for Data Warehouse IntegratIon), that implements the techniques described in the previous section.

Specifically, the tool is able to:

- access data marts stored in a variety of systems (DB2, Oracle, SQL Server, among others);
- extract from these systems metadata describing cubes and dimensions and translate these descriptions into a uniform internal format;
- support the user in the specification of matchings between autonomous dimensions (possibly suggesting some promising maps) by means of a graphical interface;
- test for coherence, consistency, and soundness of matchings;
- generate the intersection between two dimensions, according to the the loosely integration approach;
- merge two dimensions, according to the the tightly integration approach;
- perform drill-across queries over heterogeneous data marts whose dimensions have been matched according to either the tightly coupled approach or the loosely coupled one.

An hint of the graphical interface provided by this tool, which will be used in our demonstration, is reported in Figure 4.

References

[1] L. Cabibbo and R. Torlone. Integrating Heterogeneous Multidimensional Databases. In *17th Int. Conference on Scientific and Statistical Database Management (SSDBM'05)*, pages 205–214, 2005.

[2] R. Kimball and M. Ross. *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling*. John Wiley & Sons, Second edition, 2002.

[3] R.J. Miller, M.A. Hernández, L.M. Haas, L. Yan, C.T.H. Ho, R. Fagin, and L. Popa. The Clio Project: Managing Heterogeneity. *SIGMOD Record*, 30(1): 78–83, 2001.