Automating the Schema Matching Process for Heterogeneous Data Warehouses

Marko Banek[1], Boris Vrdoljak[1], A. Min Tjoa[2], and Zoran Skočir[1]

[1] Faculty of Electrical Engineering and Computing, University of Zagreb, Unska 3, HR-10000 Zagreb, Croatia
{marko.banek,boris.vrdoljak,zoran.skocir}@fer.hr
[2] Institute of Software Technology and Interactive Systems, Vienna University of Technology, Favoritenstr. 9-11/188, A-1040 Wien, Austria
amin@ifs.tuwien.ac.at

Abstract. A federated data warehouse is a logical integration of data warehouses applicable when physical integration is impossible due to privacy policy or legal restrictions. In order to enable the translation of queries in a federated approach, schemas of the federated and the local warehouses must be matched. In this paper we present a procedure that enables the matching process for schema structures specific to the multidimensional model of data warehouses: facts, measures, dimensions, aggregation levels and dimensional attributes. Similarities between warehouse-specific structures are computed by using linguistic and structural comparison, where calculated values are used to create necessary mappings. We present restriction rules and recommendations for aggregation level matching, which builds the most complex part of the process. A software implementation of the entire process is provided in order to perform its verification, as well as to determine the proper selection metric for mapping different multidimensional structures.

1 Introduction

Increasing competitiveness in business and permanent demands for greater efficiency (either in business or government and non-profit organizations) enforce independent organizations to integrate their data warehouses. Data warehouse integration enables a broader knowledge base for decision-support systems, knowledge discovery and data mining than each of the independent warehouses could offer separately. Large corporations integrate their separately developed regional warehouses, newly merged companies integrate their warehouses to enable the business to be run centrally, while independent organizations join their warehouses leading to a significant benefit to all participants and/or their customers.

Copying data from the source heterogeneous warehouses to a new central location would be the easiest way to perform the integration. However, the privacy policy of the organizations, as well as the legal protection of sensitive data in the majority of cases prohibits the data to be copied outside the organizations where they are stored. Hence, a federated data warehouse is created, which implies that the integration must be performed from a logical point of view, using a common conceptual model, while

I.Y. Song, J. Eder, and T.M. Nguyen (Eds.): DaWaK 2007, LNCS 4654, pp. 45–54, 2007.

the heterogeneous source warehouses exist physically. The user of such a federated solution observes the whole federation as a single unit i.e. she must not notice that several heterogeneous parts of the warehouse actually exist. The logical existence of the federated warehouse does not have any impact on the users of local component warehouses [13]. In [3, 12], we developed the conceptual model of a federated data warehouse that unifies data warehouses of different health insurance organizations.

Queries on the federated data warehouse must be translated into sub-queries that correspond to the schemas of local warehouses. Since the multidimensional data model prevails in data warehouse design, it is necessary for the query translation to discover mappings between particular structures of the federated multidimensional conceptual model (facts, measures, dimensions, aggregation levels and dimensional attributes) and their matching counterparts in the multidimensional conceptual models of the local warehouses.

In this paper, the process of creating the mappings between the matching warehouse-specific components will be automated as much as possible in order to shorten the data warehouse integration process as a whole. The contribution of our work is to describe a match discovery procedure for data warehouse-specific structures, with particular emphasis to aggregation levels. We enhance the match discovery techniques for database schemas, enabling their application to data warehouse schemas. Software that automates the integration process is provided and the entire procedure evaluated on an example.

The paper is structured as follows. Section 2 gives an overview of the related work. Basic strategies for matching multidimensional structures are presented in Section 3. Similarity functions for multidimensional structures are explained in Section 4, while the mapping strategies are shown in Section 5. Section 6 evaluates the performance of the algorithm that matches data warehouse schemas automatically. Conclusions are drawn in Section 7.

2 Related Work

There exist two basic approaches to automated matching of database schemas, semi-structured data schemas and ontologies: schema-based and instance-based [10].

Schema-based matching considers only schema information, not instance data. The most prominent approaches are ARTEMIS-MOMIS [1], Cupid [7] and “similarity-flooding” [8, 9]. ARTEMIS/MOMIS and Cupid are based on linguistic matching. Using a word thesaurus, structure names are compared and their similarity is expressed as a probability function. Complex structures are decomposed into substructures where linguistic comparison can be performed and thereafter a formula is used to compute the similarity of complex structures. The “similarity-flooding” algorithm translates database tables or semi-structured sources into graphs whose structures are then compared. Names of the graph vertices are simply compared as strings and no semantic knowledge is required.

Instance-based approaches apply various learning and mining techniques to compare instance data, together with schema metadata (thus being able to outperform schema-based techniques). iMAP [5] is a comprehensive approach for semi-automatic

discovery of semantic matches between database schemas, which uses neural networks and text searching methods.

An automated check of match compatibility between data warehouse dimensions is performed in [4], but match candidates must first be proposed manually by the integration designer. To the best of our knowledge, none of the existing frameworks for automated schema matching can be successfully applied to match data warehouses, due to the specific features of the multidimensional conceptual model of data warehouses.

3 Matching Strategy for Solving Heterogeneities Among Multidimensional Structures

Our algorithm aims at automating the matching of data warehouse schemas in cases when the access to data warehouse content is prohibited (e.g. medical data warehouses). Hence, we develop a matching algorithm that is based exclusively on analyzing data warehouse schemas.

3.1 Classification of Schema Heterogeneities

Heterogeneities in data warehouse schemas arise either when (1) different structures (relational, multidimensional etc.) are used to represent the same information, or (2) different specifications (i.e. different interpretations) of the same structure exist. A survey of heterogeneities that arise in "multidatabase systems", i.e. the integration of different relational databases is given in [6]. Attribute conflicts are due to different names and domains and can be divided into one-to-one (see Fig. 1: *city* in table *insurant* corresponding to *municipality* in table *patient* is an example of naming conflict while two *country* attributes with different data types is an example of domain conflict), and -to-many (*street_name* and *street_number* in table *insurant* corresponding to a single attribute *address*). Table conflicts occur when two tables in different databases describe the same information, but different names, data types or constraints are used to represent the information (there is a table conflict between *insurant* and *patient*).

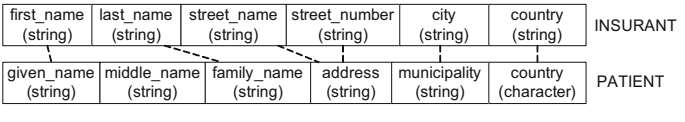

Fig. 1. To compatible tables with different attribute and table conflicts

Heterogeneities that are specific to data warehouses and the multidimensional conceptual model are analyzed in [2]. *Diverse aggregation hierarchies* can manifest either as *the lowest level conflict* or *inner level conflict.* In the first case, two semantically corresponding dimensions (e.g. time dimensions in DWH_1 and DWH_2 in Fig. 2) have different lowest (i.e. basic) grain level (*hour* and *day*) which means that the granularity of their facts is also different. In the second case there is a common aggregation level (not necessarily the lowest), but the aggregation at coarser grain levels takes different ways (levels *month* and *week* in DWH_1 and DWH_2).

The *dimensionality* conflict corresponds to a different number of dimensions associated to the same fact (the *hour* and *time* dimensions in DWH_1 corresponding to a single *time* dimension in DWH_2). Finally, *schema-instance conflicts* appear when some context of the fact in one data warehouse becomes the content (value) of dimensions in the other (*insurance* and *patient* are part of measure names in DWH_1, while being values of *cost_type* dimension in DWH_2).

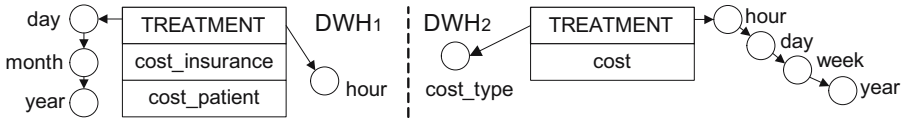

Fig. 2. Data warehouses with heterogeneities specific to the multidimensional model

3.2 The Matching Algorithm

The matching process determines that two multidimensional structures S_1 and S_2, belonging to data warehouses DWH_1 and DWH_2, respectively, are mutually equivalent. The two structures, S_1 and S_2, may be either be facts, measures, dimensions, aggregation levels or dimensional attributes, but we state that both of them must belong to the same type of multidimensional structures. Mapping cardinalities can be one-to-one, one-to-many or many-to-many. -to-many mappings are more to be expected for attributes and measures than for aggregation levels, dimensions or facts.

The algorithm that automatically matches heterogeneous data warehouse schemas consists of two basic phases: (1) comparison of match target structures and (2) creation of mappings. During the first phase the equivalence of multidimensional structures is determined as the value of a probability function called similarity function. Similarity between multidimensional structures is calculated (using a heuristic algorithm given in Section 4) by comparing their names as well as their data types (for attributes and measures) or substructures (for aggregation levels, dimensions and facts). While the same basic idea is used by ARTEMIS/MOMIS and Cupid, these frameworks cannot solve the heterogeneities that specifically occur in the multidimensional model of data warehouses, especially the hierarchical organization of aggregation levels in dimensions. In the second phase, heuristic rules are applied to determine which structures should be mapped as equivalent, among a much larger number of possible matches. The result of the automated matching process must as much as possible commensurate with the solution that a data warehouse designer would produce manually.

Our matching algorithm recognizes four levels of matching: (1) facts, (2) dimensions and measures, constructing the facts, (3), aggregation levels, constructing the dimensions, and (4) dimensional attributes, constructing the aggregation levels.

Similarity calculation, as the first part of the matching algorithm, starts with atomic structures, dimensional attributes and measures, as their similarity is computed from the similarity of their names and data types. Similarity between aggregation levels is calculated next, taking into account their names and the already calculated similarity between attributes of which they consist. The process continues with dimension similarity, while similarity between facts is determined at the end.

On the other hand, we opine that mapping (the second phase of the automated matching process) must start with the most complex multidimensional structures: the facts. If we determine that two facts are compatible (i.e. that they match), we map their measures and dimensions in the next phase. The mapping candidates are only dimensions and measures of the two facts and no other measures and dimensions in the warehouse schema. Next, we proceed to aggregation levels and then, finally, to attributes.

3.3 Mapping Aggregation Levels

Mapping facts, measures, dimensions and dimensional attributes is similar to matching database structures. For instance, given two corresponding aggregation levels L_1 and L_2, an attribute A_{1i}, which belongs to L_1 can be mapped to any attribute A_{2j} belonging to L_2. Each attribute is mapped to its most similar counterpart according to the value of the similarity function.

Hierarchical structure imposes several inherent limits to aggregation level matching, as the existing partial order must be preserved. The limitations and additional heuristic recommendations are shown in the rest of this section.

Prohibition of mappings that violate the partial order in hierarchies. Let D_1 and D_2 be two mapped dimensions, each of them (for reasons of simplicity) consist of a single hierarchy. Let L_{1i} be an aggregation level in D_1 and L_{2j} an aggregation level in D_2. Let L_{1i} and L_{2j} be equivalent, matching levels and let their mapping be already registered, as shown in the left part of Fig. 3. No level in D_1 representing finer granularity than L_{1i} can be mapped to a level in D_2 representing coarser granularity than L_{2j}. Similarly, no level in D_1 representing coarser granularity than L_{1i} can be mapped to a level in D_2 representing finer granularity than L_{2j}. The mapping between L_{1i} and L_{2j} is represented by a solid line. Invalid mappings are showed as dashed lines. The coherence of the partial orders in dimensions has also been stated as a necessary condition for dimension compatibility in [4].

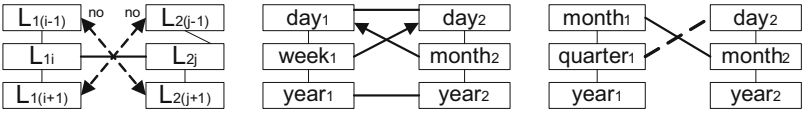

Fig. 3. Preserving the partial order in hierarchies

Mapping unmatched levels to the counterparts of their parent levels. This strategy is used to solve inner level conflicts. Let us suppose that the equivalent levels day_1 and day_2 have already been mapped (see the middle part of Fig. 3). The low value of the similarity function between $week_1$ and $month_2$ proves their incompatibility. Week records in D_1 and month records in D_2 are both aggregations obtained by summing the day records. Thus, $week_1$ can be mapped to the counterpart of its parent (i.e. the nearest finer) level, day_1, which is day_2. Similarly, $month_2$ can be mapped to day_1. When a query on D_1 concerning $week_1$ is translated in order to correspond to D_2, records of the level day_2 need to be summed.

Prohibition of mapping unmatched levels to the counterparts of their descendants. The rule is not inherent to the multidimensional model like both previous rules (it would be possible to create valid mappings that are not in accordance with it), but is experience-based. Mapping L_{1i} to $L_{2(j+1)}$ or $L_{2(j+2)}$, as well as L_{2j} to $L_{1(i+1)}$ or $L_{1(i+2)}$ (see the left part of Fig. 4) should be forbidden. An explanation can be obtained by watching the matching process in the other direction. The goal of the mapping process is to map every level to the coarsest possible counterpart that semantically corresponds to the target (in order to preserve the richness of the hierarchy). Supposing that $L_{1(i+2)}$ is equally similar to L_{2j} and $L_{2(j+2)}$ we prefer mapping it to $L_{2(j+2)}$. Back to the initial problem, mapping L_{2j} to $L_{1(i+2)}$, can be viewed in the opposite direction, as mapping $L_{1(i+2)}$ to L_{2j} (the right part of Fig. 4). $L_{1(i+2)}$ is already mapped to $L_{2(j+2)}$, which is coarser than L_{2j}, so mapping it also to L_{2j} would be redundant and useless.

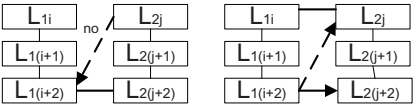

Fig. 4. Mapping unmatched levels to the counterparts of their descendants

4 Similarity Functions for Multidimensional Structures

As stated in Section 3, the process of schema matching starts with calculating similarities between various multidimensional structures and then applying a match selection algorithm based on those similarities.

The basic idea for similarity calculation is the fact that complex structures' similarity is computed from the similarities between their less complex substructures, by means of some mathematical expression. Therefore, the similarity between any two complex multidimensional structures of the same kind can recursively be translated into a set of calculations at the atomic level: the level of dimensional attributes and measures.

We introduce a new formula for calculating similarity between two attributes or two measures. Their similarity is obtained from the similarity of their names (*nsim*) and data type compatibility coefficient *tcoeff* raised to the power determined by a non-negative real number *texp* (the latter enables us to calibrate the formula):

$$sim_{att}(A_1, A_2) = nsim(A_1, A_2) \cdot tcoeff(A_1, A_2)^{texp}, \qquad texp \in [0, \infty). \tag{1}$$

Similarity of names is actually the semantic similarity of words that stem from the name. Semantic similarity functions present the degree of relatedness of two input words (word senses), which is calculated by using WordNet [14], a large thesaurus of English language, hand-crafted by psycholinguists. WordNet organizes terms according to human, native speaker's perception, providing a list of synonyms, antonyms and homonyms, as well as the subordination-superordination hierarchy and part-whole relations.

We use the semantic similarity calculation method presented by Yang and Powers [15] to calculate name similarity. Words (word senses) in WordNet are interpreted as graph vertices, connected by edges representing subordination-superordination and

part-whole relations (each edge is given a weight according to the relation type). All possible paths between two target vertices are constructed and weights multiplied across paths. The highest weight product becomes the linguistic similarity between two target words.

We assume the notion of data type compatibility such that attributes sharing their data type are more related than those that do not. We reduce the data types to four basic ones: *numeric*, *string*, *datetime* and *boolean*. Rarely appearing data types (date, boolean) are more indicative than the frequent string and numeric types, either when their compatibility indicates a higher similarity or when their incompatibility suggest that attributes do not correspond. We empirically determine the following values of *tcoeff* for two compatible basic types: 1.0 for two *datetime* or *boolean* attributes, and 0.9 for two *numeric* or two *string* attributes. Different combinations of incompatible data types imply the following values of *tcoeff*: 0.8 for *numeric-string*, 0.7 for *datetime-boolean* and 0.75 for other combinations.

Similarity between complex multidimensional structures S_1 and S_2 is expressed as a weighted sum (i.e. linear combination) of their name similarity (*nsim*) and structural similarity (*ssim*), as stated in by Madhavan et al. [7]:

$$sim(S_1,S_2) = w_{name} \cdot nsim(S_1,S_2) + (1 - w_{name}) \cdot ssim(S_1,S_2), \qquad w_{name} \in [0, 1]. \tag{2}$$

In [7] structure similarity is calculated recursively, with some initial values being adjusted in several steps. We use a different approach, adapting the formula for calculating semantic similarity among entity classes of different ontologies [11]. *Neighborhood similarity* between two entity classes not only takes into account their names, but also other classes surrounding them within a certain radius. We therefore make an analogy between aggregation levels (and dimensions and facts, respectively) and entity classes. Attributes (and aggregation levels and dimensions/measures, respectively) surrounding them correspond to neighborhood classes within radius one (each attribute is directly related to the target aggregation level).

5 Mapping Multidimensional Structures

Creation of mappings is the second main phase of the algorithm for automated warehouse schema matching. This process is determined by two basic factors: (1) constraints and (2) selection metrics. While there are no particular constraints to mapping facts, dimensions, measures and dimensional attributes, the aggregation level mapping is restricted by the rules given in Section 3.3, in order to preserve the partial order in hierarchies. A selection metric defines how the calculated similarity values are used to determine which mappings are “better” or “best” among all possible mappings.

The selection metric issue has long been studied in graph theory as the *problem of stable marriage in bipartite graphs*. Bipartite graphs consist of two disjoint parts such that no edge connects any two vertices in the same part (i.e. vertices in the same part are of the same sex, while an edge symbolizes a possible marriage and can thus exists only between vertices in different parts of the graph). Each edge is given a weighting coefficient that corresponds to the similarity between multidimensional structures represented by the vertices. The task of a selection metric (which is also called a filter), is to select

an appropriate subset of edges as mappings and eliminate all others. In the original version of the stable marriage problem, only 1:1 mappings (i.e. the monogamous ones) are allowed.

In [9] six different filters were tested. We implemented three of them: *best sum*, *threshold* and *outer*. The best sum filter maps combinations with the highest possible total sum of edge similarities. For the graph given in the left part of Fig. 5, it would produce combinations (a_1,b_2) and (a_2,b_1) as their sum, 1.20, is greater than the sum of the other possible combination (1.04). The threshold and outer filter allow each of the partners to make the choice of their own. First, *relative similarities* are computed as fractions of the absolute similarities of the best match candidates for any given element:

$$sim_R(a,b_j) = sim(a,b_j) / \max_j sim(a,b_j) . \tag{3}$$

Vertices with the highest relative similarity choose first (if there are several such vertices, one of the two vertices that are adjacent to the edge with the highest absolute similarity will take the lead; in particular case a_1 or b_1). The threshold filter creates a mapping if relative similarities at both sides of the selected edge are greater than a threshold. Thus, only mapping (a_1,b_1) is created (the right part of Fig. 5). The outer filter checks the threshold only at the side that "proposed" a match, mapping (a_2,b_2) as the second pair.

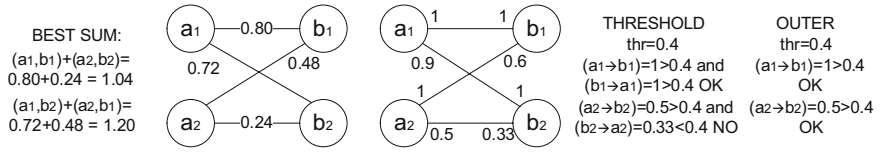

Fig 5. Mapping the same structures differently using the *best sum*, *threshold* and *outer* filter

Polygamous (i.e. 1:N and M:N) matches are allowed for multidimensional structures. The standard best sum, threshold and outer filter can be applied to map facts, dimensions, measures and attributes. Restriction rules to mapping aggregation levels create additional dynamic constraints. If we falsely map level $quarter_1$ to day_2 (see the right part of Fig. 3), we will not be able to produce the correct mapping between $month_1$ and $month_2$.

6 Evaluation of Algorithm Performance

The quality of the automated matching process is measured by its *accuracy*, as defined in [8, 9]. The measure estimates how much effort it costs the user to modify the automatically proposed match result P={(x_1, y_1),…,(x_n, y_n)} into the intended result I={(a_1, b_1), … ,(a_m, b_m)}, i.e. how many additions and deletions of map pairs have to be made. Let $c = |P \cap I|$ be the number of correct suggestions. The difference (n–c) is the number of false positives to be removed from P, and (m–c) the number of missing

matches that need to be added. Assuming (for the reason of simplicity) that deletions and additions of match pairs require the same amount of effort, accuracy, i.e. the labor savings obtained by using our matching technique is defined as:

$$1 - [(n - c)+(m - c)] / m. \quad (4)$$

In a perfect match, n = m = c, resulting in accuracy 1. Negative accuracy (for c/n < 0.5 i.e. when more than half of the created matches are wrong) suggests that it would take the user more effort to correct the automatically proposed map than to perform the matching manually from scratch.

We evaluate the proposed algorithm by matching the schema of the federated data warehouse for health insurance organizations developed in [3, 12] to the local warehouse of one of the organizations. Both warehouses consist of mutually compatible three facts representing patient encounters, performed therapies and medicine prescriptions. In both cases facts share dimensions (many, but not all which are compatible).

The values of match accuracy for each of the three filters are given in Table 1. The outer and threshold filter perform best with relative similarity thresholds between 0.8 and 0.7. For the most critical part of the matching process, aggregation levels, the outer and best sum filter outperform threshold. Threshold outperforms other filters when mapping other multidimensional structures.

Table 1. Comparison of three mapping selection metrics (filters)

	OUTER				THRESHOLD				BEST SUM			
	n	m	c	acc	n	m	c	acc	n	m	c	acc
facts	3	3	3	**1.00**	3	3	3	**1.00**	3	3	3	**1.00**
measures	6	6	6	**1.00**	6	6	6	**1.00**	6	6	6	**1.00**
dimensions	8	7	7	0.86	7	7	7	**1.00**	7	7	7	**1.00**
agg. levels	24	24	22	**0.83**	22	24	20	0.75	24	24	22	**0.83**
dim. attrib.	122	57	51	-0.35	64	57	47	**0.52**	64	57	44	0.42

7 Conclusion

This paper presents an approach to automating the schema matching process for heterogeneous data warehouses in order to shorten the warehouse integration process. The approach is based on analyzing warehouse schemas and can be used when the access to *data content* is restricted. We defined a match discovery procedure capable of solving heterogeneities among data warehouse-specific structures: facts, measures, dimensions, aggregation levels and dimensional attributes. A particular relevance was given to aggregation level matching, as the partial order in dimension hierarchies must be preserved. A software implementation of the approach has been developed and different selection filters tested on a case study example. The best sum and outer filter performed better than threshold when mapping aggregation levels, while the latter was the best solution for other multidimensional structures.

References

1. Bergamaschi, S., Castano, S., Vincini, M.: Semantic Integration of Semistructured and Structured Data Sources. SIGMOD Record 28, 54–59 (1999)
2. Berger, S., Schrefl, M.: Analysing Multi-dimensional Data accross Autonomous Data Warehouses. In: Tjoa, A.M., Trujillo, J. (eds.) DaWaK 2006. LNCS, vol. 4081, pp. 120–133. Springer, Heidelberg (2006)
3. Banek, M., Tjoa, A.M., Stolba, N.: Integrating Different Grain Levels in a Medical Data Warehouse Federation. In: Tjoa, A.M., Trujillo, J. (eds.) DaWaK 2006. LNCS, vol. 4081, pp. 185–194. Springer, Heidelberg (2006)
4. Cabibbo, L., Torlone, R.: Integrating Heterogeneous Multidimensional Databases. In: Proc. Int. Conf. Scientific and Stat. Database Management '05, pp. 205–214, IEEE Comp. Soc. (2005)
5. Dhamankar, R., Lee, Y., Doan, A.-H., Halevy, A.Y., Domingos, P.: iMAP: Discovering Complex Mappings between Database Schemas. In: Proc. SIGMOD Conf. 2004, pp. 383–394. ACM Press, New York (2004)
6. Kim, W., Seo, J.: Classifying Semantic and Data Heterogeneity in Multidatabase Systems. IEEE Computer 24/12, 12–18 (1991)
7. Madhavan, J., Bernstein, P.A., Rahm, E.: Generic Schema Matching with Cupid. In: Proc. Int. Conf. on Very Large Data Bases '01, pp. 49–58. Morgan Kaufmann, San Francisco (2001)
8. Melnik, S., Garcia-Molina, H., Rahm, E.: Similarity Flooding: A Versatile Graph Matching Algorithm and Its Application to Schema Matching. In: Proc. Int. Conf. on Data Engineering 2002, pp. 117–128. IEEE Computer Society, Los Alamitos (2002)
9. Melnik, S., Garcia-Molina, H., Rahm, E.: Similarity Flooding: A Versatile Graph Matching Algorithm. Technical Report (2001), http://dbpubs.stanford.edu/pub/2001-25
10. Rahm, E., Bernstein, P.A.: A survey of approaches to automatic schema matching. VLDB J. 10, 334–350 (2001)
11. Rodríguez, M.A., Egenhofer, M.J.: Determining Semantic Similarity among Entity Classes from Different Ontologies. IEEE Trans. Knowl. Data Eng. 15, 442–456 (2003)
12. Stolba, N., Banek, M., Tjoa, A.M.: The Security Issue of Federated Data Warehouses in the Area of Evidence-Based Medicine. In: Proc. Conf. Availability, Reliability and Security '06, pp. 329–339. IEEE Computer Society, Los Alamitos (2006)
13. Sheth, A.P., Larson, J.A.: Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases. ACM Computing Surveys 22, 183–236 (1990)
14. Princeton University Cognitive Science Laboratory: WordNet, a lexical database for English Language (Last access: March 25, 2007) http://wordnet.princeton.edu
15. Yang, D., Powers, D.M.W.: Measuring Semantic Similarity in the Taxonomy of WordNet. In: CRPIT 38, pp. 315–322, Australian Computer Society (2005)