Mapping Between Data Sources on the Web

George H. L. Fletcher
Computer Science Department
Indiana University, Bloomington, USA
gefletch@cs.indiana.edu

Catharine M. Wyss
Computer Science & School of Informatics
Indiana University, Bloomington, USA
cmw@cs.indiana.edu

Abstract

The data mapping problem is to discover effective mappings between structured representations of data. These mappings are the basic 'glue' for facilitating large-scale ad-hoc information sharing between autonomous peers in a dynamic environment. Automating their discovery is one of the fundamental unsolved challenges for information integration and sharing on the Web. We outline a general approach to automating the discovery of mappings between relational data sources which leverages new perspectives on the data mapping problem and report on a prototype implementation. Our approach utilizes heuristic search within a space delineated by basic relational transformation operators. A further novelty of our approach is that these operators include data to metadata transformations (and vice versa), allowing a generalization of previous solutions such as token-based schema matching.

1. Introduction

The vision of peer-to-peer database management systems (P2PDBMS) brings promise of ad-hoc dynamic information exchange, with support for richer semantics than the current breed of simple file-sharing peer-to-peer systems [7, 12]. The complementary vision of the Semantic Web also holds promise for intelligent complex information exchange on the Web [2, 9]. These systems cannot and should not be built from scratch, since a significant portion of data on the Web resides in non-Semantic-Web-enabled data sources [4, 9]. The participation of these data sources in P2PDBMS and Semantic Web information sharing scenarios requires new technologies which respect source autonomy while enabling ad-hoc complex information exchange.

A fundamental unsolved challenge in information integration and sharing on the Web is the *Data Mapping Problem*: automating the discovery of effective mappings between structured representations of data. These mappings are the basic 'glue' for facilitating large scale ad-hoc information exchange between autonomous peers in a dynamic environment [9]. This is a central problem that is encountered in many information management settings. Consequently, many variants of the problem have been identified and investigated: schema mapping and query discovery [19], schema matching [21], semantic mapping [6], 'matching' on information models [17], data translation [20, 23], and ontology matching [10]. The ubiquitous nature of the problem is illustrated in Figure 1, where arrows indicate mappings between data sources within or across peers. $T1$, for example, maps a local data source to the global schema of *Peer A*, and $T2$ maps data from *Peer A* to data in *Peer B*.

We are investigating the data mapping problem in the *Modular Integration of Queryable Information Sources* (MIQIS) project at Indiana University [26], a formal framework for data exchange on the Semantic Web and in P2PDBMS. Among the distinguishing features of MIQIS is a focus on the *modular* nature of information systems, encompassing XML, relational, text, 'Deep Web' [4], and other data sources. The framework fully respects the autonomy of peers to manage locally their schemata and concepts. On the Semantic Web and P2PDBMS, global consensus and monolithic architectures are unlikely and ultimately infeasible. MIQIS fully accommodates the heterogeneity and autonomy of data sources in ad-hoc dynamically evolving environments.

1.1. Contributions and Outline of Paper

This paper characterizes the general data mapping problem and presents a novel data mapping solution for relational data sources. We also report on a prototype implementation and give preliminary performance results. Our approach leverages several perspectives on the data mapping problem which are highlighted in the MIQIS project [26]:

- *Rosetta Stone Principle*: Small 'critical instances' of source and target schemas provided by the user can be effectively used to generate data maps.
- *Tuple Normal Form*: A standardized format for critical instances allows for their uniform manipulation.

Proceedings of the 2005 International Workshop on Challenges in Web Information Retrieval and Integration (WIRI'05)
0-7695-2414-1/05 $20.00 © 2005 IEEE

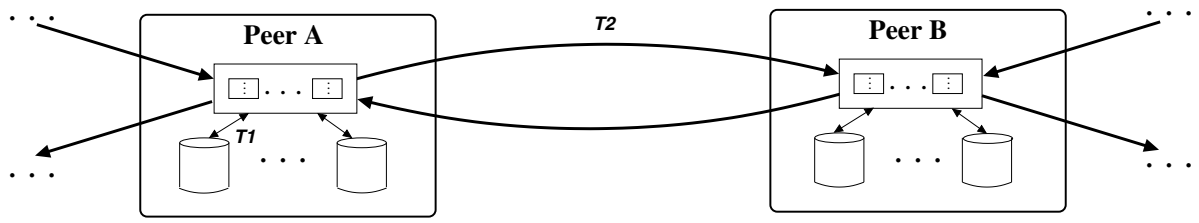

Figure 1. Data mappings as *inter*- and *intra*-peer 'glue' for P2P and Web information systems.

- *Data Mapping as a Search Problem*: We view the data mapping problem as *search* in a space of transformations on data in tuple normal form.

We briefly develop each of these points in this paper. We begin with an introduction to the data mapping problem and a discussion of our approach in Section 2. A prototype implementation and performance results are then presented in Section 3, followed by a discussion of related research in Section 4. Finally, we give concluding remarks and indications for future work in Section 5.

2. The Data Mapping Problem

How can we facilitate the discovery of an appropriate mapping between data structured under two distinct schemas with minimal user input?

Example 1 *Consider two tables containing the same student grade information and a mapping τ between them:*

S:

Name	Assignment	Percentage
Saori	Assignment1	94
Saori	Assignment2	97
Yukie	Assignment1	88
Yukie	Assignment2	89

$\downarrow\tau$

T:

Student	Assignment1	Assignment2
Saori	94	97
Yukie	88	89

τ must promote "Assignment" values in S to column names in T and match the "Name" and "Student" columns.

Can the discovery of τ be (semi) automated? We would like our mapping language to be practical and declarative. In the context of RDBMS, this means mappings must be SQL compatible queries to maximize the use of underlying RDBMS technology.

A general statement of this problem is as follows:

Definition 1 (Data Mapping Problem) *Given source data schema $\mathcal{S}$, target data schema $\mathcal{T}$, and query language $\mathcal{L}$, find a transformation $\tau \in \mathcal{L}$ (if it exists) such that for any instance s of $\mathcal{S}$ and corresponding instance t of $\mathcal{T}$, $s \overset{\tau}{\longmapsto} t$.*

This definition encompasses all variants of the data mapping problem listed in Section 1. Note that $\mathcal{S}$ and $\mathcal{T}$ are not assumed to be schemas of the same data model. It is not immediately clear how to automate a solution to the *general* problem; furthermore, it is generally believed that the full problem is 'AI-Complete' [17]. In MIQIS, we focus on subcases of the problem by following a modular approach to information exchange in P2PDBMS. In this paper we develop the MIQIS module to generate SQL compatible transformations when $\mathcal{S}$ and $\mathcal{T}$ are both *relational* schemas.

Example 2 *Suppose that three peers contain student grade information within a larger network for managing student information (Figure 2). As shown, there are many natural ways to organize even the simplest datasets such as these. To move between these representations of student data, both schema matchings and data-metadata transformations must be performed. In the previous example we saw that transforming data in database G1 into data structured under database G2 required that data values be promoted to column names and column names be matched. To move data from G3 to G1, relation names must be demoted to data values.*

2.1. Rosetta Stone Principle

A key component of our approach is the *Rosetta Stone Principle*: user-provided small 'critical' instances of source schema $\mathcal{S}$ and target schema $\mathcal{T}$ can be effectively used to guide the discovery of τ in a transformation space [26]. These canonical instances must illustrate all of the appropriate restructurings between source and target to guide the search process, and the information 'content' of the target must be contained in the source [23]. We also explicitly consider the full data mapping space for relational DBs: schema matchings (i.e., 'traditional' metadata-metadata mappings between schema elements [21]) and data-metadata mappings where data elements in one structure serve as metadata components in the other (or vice versa) [13, 18]. It is important to note that consideration of the full mapping space blurs the distinction between schema matching and schema mapping [19], since data-metadata mapping encompasses schema matching as a special case. To press

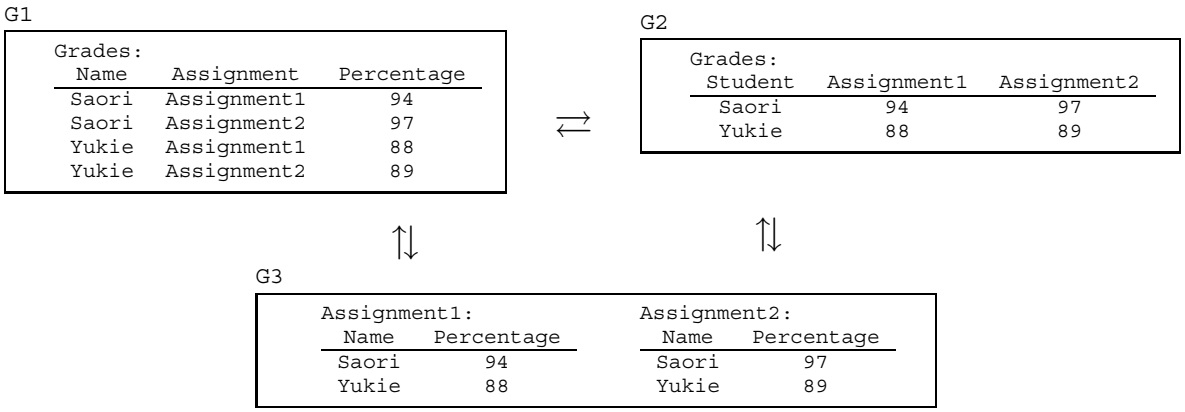

Figure 2. Mappings between student grade representations.

the point, when metadata itself is seen as data, the entirety of schema matching and schema mapping is encompassed in *data mapping*. The output of the solution presented in this paper is a data-to-data transformation that is parameterized semantically by schema information. For example, the transform generated for mapping `G1` to `G2` holds for any number of assignment values in `G1`.

2.2. Tuple Normal Form

Another key technical component of our approach is a normal form for relational data, *Tuple Normal Form* (TNF), first introduced by Litwin et al. [15]. This standardized format for representing relational data allows us to seamlessly manipulate metadata alongside data using SQL. Furthermore, multiple input relations are represented in a single TNF relation; thus TNF enables data mappings where the source and/or target information may be split over more than one relation (such as the transformations involving database `G3` in Figure 2).

For a given relation R, we compute TNF of R (denoted R^*) as follows. First, every tuple in the relation is given a unique identifier. Then, R^* is a four-column relation with attributes TID, REL, ATT, and VALUE containing the data in R in a piecemeal fashion. The TNF of an input database D (denoted D^*) is simply the union of R^* for all $R \in D$. Note that the TNF of a database can be computed in SQL using the system tables. We illustrate this with database `G3`.

Example 3 *TNF of database `G3`:*

`G3`* :

TID	REL	ATT	VALUE
t_1	Assignment1	Name	Saori
t_1	Assignment1	Percentage	94
t_2	Assignment1	Name	Yukie
t_2	Assignment1	Percentage	88
t_3	Assignment2	Name	Saori
t_3	Assignment2	Percentage	97
t_4	Assignment2	Name	Yukie
t_4	Assignment2	Percentage	89

2.3. Relational Transformation Space

We consider a fixed set of simple transformations on data in TNF. This allows us to consider data mapping discovery as an exploration of the transformation space of these operators on the source instance. Search terminates when the TNF representation of the transformed source instance becomes a superset of the TNF representation of the target instance. At this point, the transformational path is translated to a parameterized map between instances of the source schema and instances of the target schema.

In this approach, no assumptions of common domains, global schema, underlying generative ontology, or other simplifications are made. Data are treated simply as opaque objects; the search process is purely syntactically and structurally driven [3, 11]. As per the Rosetta Stone Principle, the user-provided critical source and target instances provide the initial matches which drive the search process.

All transformations between the databases in Figure 2 can be performed using compositions of the simple, compositional, invertible transformations given in Table 1. We omit a complete formal definition of the operators in this paper; these operators mimic algebraic operators developed elsewhere for federated relational systems [22, 24, 25]. These operators can be implemented in SQL on the TNF representations of relational databases.

Example 4 *Consider the basic transformations involved in restructuring the information in `G1` into the format of `G2`:*

$R_1 :=$ $\uparrow$(`G1`*, `Assignment`, `Percentage`)
Promote assignments to metadata.
$R_2 := \varkappa(R_1,$ `Assignment`$)$
Drop column "`Assignment`"
$R_3 := \varkappa(R_2,$ `Percentage`$)$
Drop column "`Percentage`"
$R_4 := \rho(R_3,$ `Name`, `Student`$)$
Rename attribute "`Name`" to "`Student`"
$R_5 := \oplus(R_4,$ `Student`$)$
Merge assignment grades for students.

The output TNF relation R_5 is exactly `G2`*.

Operation	Effect
$\downarrow(R)$	*Demote Metadata*. Cartesian product of relation R with a binary table containing the metadata of R.
$\rightarrow(R, A, B)$	*Dereference Column A on B*. $\forall t \in R$, append a new column named B with value $t[t[A]]$.
$\uparrow(R, A, B)$	*Promote Column A to Metadata*. $\forall t \in R$, append a new column named $t[A]$ with value $t[B]$.
$\wp(R, A)$	*Partition on Column A*. $\forall v \in \pi_A(R)$, create a new relation named v, where $t \in v$ iff $t \in R$ and $t[A] = v$.
$\text{ɯ}(R, A)$	*Drop* column A from relation R.
$\oplus(R, A)$	*Merge* tuples in relation R based on compatible values in column A.
$\rho(R, R', A, A')$	*Rename* relation R to R' and column A to A' .

Table 1. Basic transformations defining relational search space.

Note that the user is responsible for providing post-filters such as "Drop all students with grades less than 70", if desired. The search operators focus on bulk structural transformations rather than selections. In fact, selection conditions cannot in general be uniquely determined [12].

2.4. Data Mapping as Search

Considering the data mapping problem as a search problem is one of the main novel contributions of this paper. The eight basic transformations shown in Table 1 define a relational search space for data maps. Approaching the data mapping problem as a search problem from TNF to TNF representations allows us to leverage existing Artificial Intelligence (AI) search techniques [16]. The search process is depicted in Figure 3. As discussed above, the search space is rooted at the critical source instance in TNF and search proceeds by applying the transformations to generate new states. The branching factor of this search space must take into account the active domain in the Rosetta Stone instances s and t, which means the (unoptimized) branching factor is proportional to $|s| + |t|$. In spite of this daunting search space size, traditional AI techniques can be successfully applied to prune the space of examined states.

3. Prototype Implementation

A prototype semi-automatic search module for relational data mappings has been implemented in Scheme. In this section we discuss our prototype implementation and preliminary performance results.

3.1. Search Algorithms and Heuristics

The search routine takes as input canonical source and target instances in TNF and performs the search for a transformation from the source to the target as outlined above. A wide range of search algorithms have been developed in the AI literature. We have implemented the *A** and *Iterative Deepening A** (IDA*) [16] search procedures. We chose these procedures because of their simplicity and effectiveness. These search algorithms make use of a heuristic measure to rank search states and selectively explore the search space based on these rankings. The evaluation function f for ranking a search state x is calculated as $f(x) = g(x) + h(x)$, where $g(x)$ is the number of transforms applied to the source instance to generate x and $h(x)$ is a heuristic measure of the distance of x from the target. In brief, A* basically performs a breadth-first traversal of the search space that at each step explores the lowest f-ranked unexamined state on the search horizon, and IDA* performs a depth-bounded depth-first traversal of the same space using the f-rankings of states as a bound that is iteratively increased until the target is generated. In practice, it has been shown that IDA* has performance asymptotic to A* without the impractical space requirements [16] and hence we focus on IDA* in our preliminary experimental results.

We have tested these routines with three simple heuristic functions for a given search state x and target state t. In brief, heuristic h_1 measures the number of relation, column, and data values in the target state t which are missing in state x, heuristic h_2 measures the minimum number of promotions ($\uparrow$) and demotions ($\downarrow$) needed to transform x into the target t, and heuristic h_3 is simply $\mathbf{max}\{h_1(x), h_2(x)\}$.

3.2. Experimental Results

Our initial experiments have shown that IDA* with these heuristics performs quite well. The effectiveness of IDA* search for discovering transformations between the example student grade databases is illustrated in Figure 4 (A). As these results indicate, the effectiveness of the individual heuristics relative to each other vary widely on the type of transformation. For example, h_1 performs very well for mapping `G3` to `G1`. It performs poorly relative to the other heuristics, however, for mapping `G1` to `G2`. Overall, h_1 performed quite well, while h_2 had varying success. This indicates that more work needs to be done to find a good general purpose heuristic for all transformation scenarios.

The effectiveness of this approach for basic schema

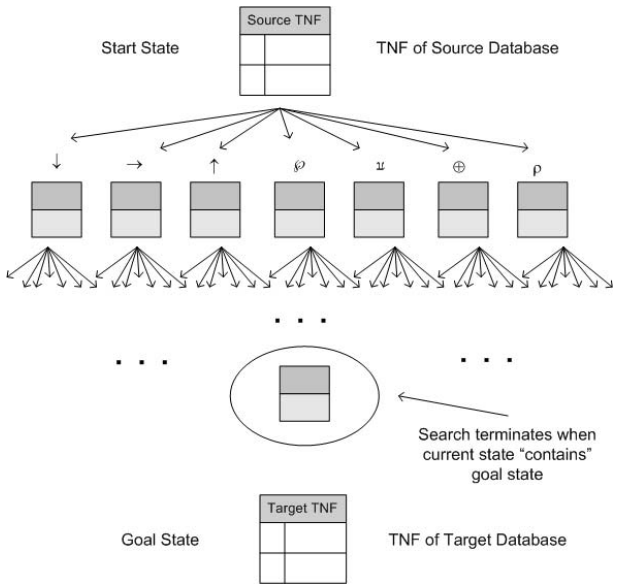

Figure 3. The search for relational data mappings.

matching (i.e., metadata-to-metadata transformations [21]) was tested on synthetic data with source and target instances with up to 60 column name matchings. This is illustrated in Figure 4 (B) for IDA* with no heuristic (i.e., basic iterative deepening depth-first search) and with heuristic h_1. Note that matching 60 schema elements with heuristic h_1 required exploring only $\approx 36,000$ states. Clearly the approach is effective with even such a simple heuristic. In particular, under this approach token-based schema matching is possible for hitherto unrealized scenarios, such as multi-relation mappings between wide source and target instances.

4. Related Work

The approaches to data mapping most closely related to ours are the works of Bilke and Naumann [3] on using duplicate values to guide schema matching, the Sphinx project [1] and Doan et al. [6] on leveraging machine learning techniques for schema matching and integration, Kang and Naughton [11] on treating data as opaque objects during schema matching, and the Clio project [19], Torlone and Atzeni [23], and Milo and Zohar [20] on semi-automating the discovery of schema mappings. To our knowledge, none of these works have considered the full space of data-metadata transformations, with only the Sphinx [1] and Clio [19] projects considering any aspects of such mappings. Our work complements and extends these works with a new perspective on the data mapping problem and a novel solution to this problem for the complete relational transformation space. To our knowledge, our approach is the first to encompass the full data mapping problem for relational sources. We emphasize, however, that the solution presented in this paper is a complementary and exploratory addition to approaches to schema matching/mapping in the literature and can be considered as a useful addition to a multi-strategy data mapping approach [5, 6].

5. Conclusions and Future Work

In this paper we have developed a novel solution to the data mapping problem for relational data sources addressing the full space of data-metadata transformations. This solution was founded on a new perspective on the data mapping problem. We reported on a prototype implementation of our solution and provided initial performance results which indicate that it is effective not only for schema matchings but also for data-metadata transformations. The general approach we have taken is applicable to the discovery of mappings between other structured representations of information such as XML and 'Deep Web' sources. One of the overarching goals of MIQIS is the development of modules for each of these transformation spaces. The results presented here are a first step in this project.

The prototype implementation developed is clearly effective even with the simplest of search procedures and heuristics. It has served mainly as a proof-of-concept, however. To enhance performance, we will incorporate basic improvements developed in the literature into our IDA* implementation [16] and develop smarter heuristics. We are also investigating other techniques such as *genetic programming* [16] for exploring the space of relational transformations.

In future work, we will also explore the application of our approach to a *domain-independent* framework for data mapping on the 'Deep Web' [4, 8] and the extension of our approach to a framework for data mapping that incorporates statistical/probabilistic name matching [3, 8, 11], where the equality check on atoms during the search process is replaced with an external similarity measure. We believe that the approach outlined in this paper can be fruitfully applied to these scenarios.

References

[1] Barbançon, François and Daniel P. Miranker, "Interactive Schema Integration with Sphinx," in *Proc. FQAS, Springer Verlag LNCS 3055*, pp. 175-190, Lyon, France, 2004.

[2] Berners-Lee, Tim, James Hendler, and Ora Lassila, "The Semantic Web," *Scientific American* 284(5):34-43, May 2001.

[3] Bilke, Alexander and Felix Naumann, "Schema Matching using Duplicates," in *Proc. IEEE ICDE*, pp. 69-80, Tokyo, Japan, 2005.

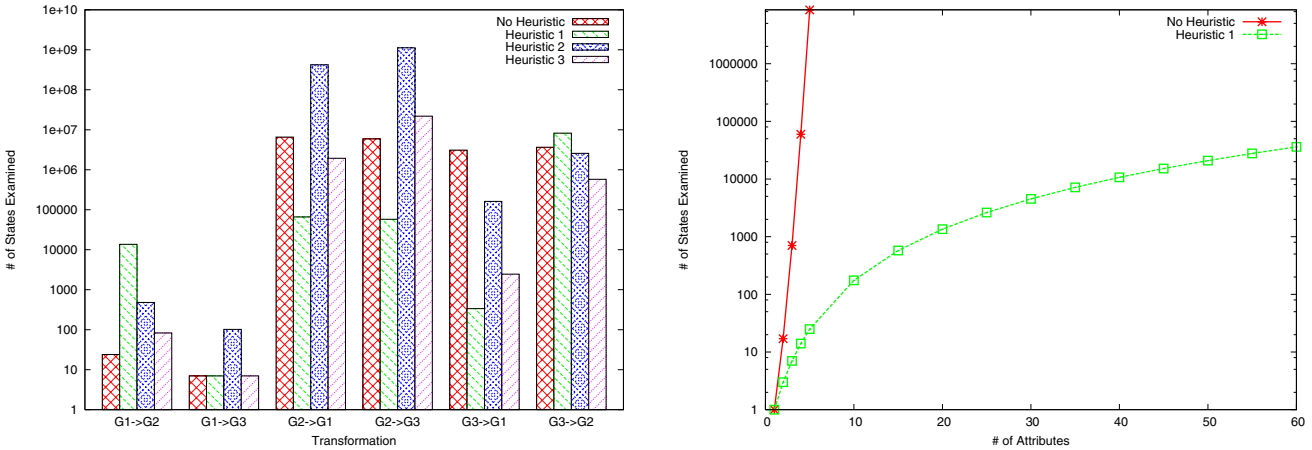

Figure 4. Number of states examined during search for (A) DB transformations (B) schema matching.

[4] Chang, K. C.-C., B. He, C. Li, and Z. Zhang, "Structured Databases on the Web: Observations and Implications," *UIUC Tech Report UIUCDCS-R-2003-2321*, February 2003.

[5] Do, Hong-Hai, and Erhard Rahm, "COMA - A System for Flexible Combination of Schema Matching Approaches," in *Proc. VLDB Conf.*, pp. 610-621, Hong Kong, China, 2002.

[6] Doan, AnHai, Pedro Domingos, and Alon Halevy, "Learning to Match the Schemas of Databases: A Multistrategy Approach," *Machine Learning* 50(3):279-301, March 2003.

[7] Halevy, A. Y., Z. G. Ives, D. Suciu, and I. Tatarinov, "Schema Mediation in Peer Data Management Systems," in *Proc. IEEE ICDE*, pp. 505-516, Bangalore, India, 2003.

[8] He, Bin and Kevin Chen-Chuan Chang, "Statistical Schema Matching Across Web Query Interfaces," in *Proc. ACM SIGMOD*, pp. 217-228, San Diego, CA, USA, 2003.

[9] Ives, Zachary G., Alon Y. Halevy, Peter Mork, and Igor Tatarinov, "Piazza: Mediation and Integration Infrastructure for Semantic Web Data," *J. of Web Sem.* 1(2):155-175, 2004.

[10] Kalfoglou, Y. and M. Schorlemmer, "Ontology Mapping: the State of the Art," *Knowledge Eng. Review* 18(1):1-31, 2003.

[11] Kang, Jaewoo and Jeffrey F. Naughton, "On Schema Matching with Opaque Column Names and Data Values," in *Proc. ACM SIGMOD*, pp. 205-216, San Diego, CA, 2003.

[12] Kementsietsidis, Anastasios, et al., "Mapping Data in Peer-to-Peer Systems: Semantics and Algorithmic Issues," in *Proc. ACM SIGMOD*, pp. 325-336, San Diego, CA, 2003.

[13] Krishnamurthy, Ravi, et al., "Language Features for Interoperability of Databases with Schematic Discrepancies," in *Proc. ACM SIGMOD*, pp. 40-49, Denver, CO, USA, 1991.

[14] Lenzerini, Maurizio, "Data Integration: A Theoretical Perspective," in *Proc. PODS*, pp. 233-246, Madison, WI, 2002.

[15] Litwin, Witold, Mohammad A. Ketabchi, and Ravi Krishnamurthy, "First Order Normal Form for Relational Databases and Multidatabases," *SIGMOD Record* 20(4):74-76, 1991.

[16] Luger, George F., *Artificial Intelligence: Structures and Strategies for Complex Problem Solving, 5th Edition*, Addison-Wesley, London, 2005.

[17] Melnik, Sergey, "Generic Model Management: Concepts and Algorithms," *Springer Verlag LNCS 2967*, 2004.

[18] Miller, Renée J., "Using Schematically Heterogeneous Structures," in *Proc. SIGMOD*, pp. 189-200, Seattle, 1998.

[19] Miller, Renée J., Laura M. Haas, and Mauricio A. Hernández, "Schema Mapping as Query Discovery," in *Proc. VLDB Conf.*, pp. 77-88, Cairo, Egypt, 2000.

[20] Milo, Tova and Sagit Zohar, "Using Schema Matching to Simplify Heterogeneous Data Translation," in *Proc. VLDB Conf.*, pp. 122-133, New York City, New York, USA, 1998.

[21] Rahm, E. and P. Bernstein, "A Survey of Approaches to Automatic Schema Matching," *VLDB J.* 10(4):334-350, 2001.

[22] Sattler, Kai-Uwe, et al., "Interactive Example-Driven Integration and Reconciliation for Accessing Database Federations," *Information Systems* 28(5):393-414, July 2003.

[23] Torlone, Riccardo and Paolo Atzeni, "A Unified Framework for Data Translation over the Web," in *Proc. IEEE WISE*, pp. 350-358, Kyoto, Japan, 2001.

[24] Wyss, Catharine M. and Dirk Van Gucht, "A Relational Algebra for Data/Metadata Integration in a Federated Database System," in *Proc. ACM CIKM*, pp. 65-72, Atlanta, GA, 2001.

[25] Wyss, Catharine M. and Edward Robertson, "Relational Languages for Metadata Integration," *ACM Transactions on Database Systems*, to appear June 2005.

[26] Wyss, Catharine M., et al., "MIQIS: Modular Integration of Queryable Information Systems," in *Proc. VLDB Workshop IIWeb*, pp. 136-140, Toronto, Canada, 2004.