Accessing Data Integration Systems through Conceptual Schemas* (extended abstract)

Andrea Calì, Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini

Dipartimento di Informatica e Sistemistica
Università di Roma "La Sapienza"
Via Salaria 113, I-00198 Roma, Italy
lastname@dis.uniroma1.it,
http://www.dis.uniroma1.it/~*lastname*

Abstract. Data integration systems provide access to a set of heterogeneous, autonomous data sources through a so-called global, or mediated view. There is a general consensus that the best way to describe the global view is through a conceptual data model, and that there are basically two approaches for designing a data integration system. In the global-as-view approach (GAV), one defines the concepts in the global schema as views over the sources, whereas in the local-as-view approach (LAV) one characterizes the sources as views over the global schema. While it is known that query processing in LAV is a complex task, it is a common opinion that it is much easier in GAV. In this paper we show the surprising result that, when the global schema is expressed in terms of a conceptual data model, even a very simple one, query processing becomes difficult in the global-as-view approach also. We demonstrate that the problem of incomplete information arises in this case too, and we illustrate some basic techniques for effectively answering queries posed to the global schema of the data integration system.

1 Introduction

Data integration is the problem of combining the data residing at different sources, and providing the user with a unified view of these data, called global (or, mediated) schema [13, 14]. The user queries the global schema, ignoring the location and structure of the data sources, and leaving to the system the task of merging and reconciling the data at the sources.

Typically, sources adopt different ontologies, models, and systems for storing data. The use of a conceptual data model has been advocated for providing an appropriate abstraction of all the data residing at the sources [4, 19, 2]. In this paper we follow this idea, and investigate the problem of query answering in data integration systems where the global schema is expressed in terms of an extended Entity-Relationship model.

With regard to the specification of the mapping between the global schema and the sources, two basic approaches have been used to specify the mapping between the sources and the global schema [13, 15, 16]. The first approach, called *global-as-view* (also global-schema centric, or simply global-centric), requires that the global schema is expressed in terms of the data sources. The second approach,

* An extended version of this paper was published in the Proc. of the 20th Int. Conf. on Conceptual Modeling (ER 2001).

called *local-as-view* (or source-centric), requires the global schema to be specified independently from the sources. A comparison of the approaches is reported in [21]. In this paper, we concentrate on the former approach, which is generally considered sufficiently simple and effective for practical purposes.

Another important issue is the choice of the method for computing the answer to queries posed in terms of the global schema. For this purpose, the system should be able to re-express the query in terms of a suitable set of queries posed to the sources, and assemble all the results into the final answer. It is well known that processing queries in the local-as-view approach is a difficult task [20, 21, 12, 1, 11, 5, 6], similar to query answering with incomplete information [22]. On the other hand, query processing looks much easier in the global-as-view approach, where in general it is assumed that answering a query basically means unfolding its atoms according to their definitions in terms of the sources [13]. While this is a common opinion in the literature, we show that our framework poses new challenges, specially related to the need of taking the semantics of the conceptual global schema into account during query processing. We show that the idea of adopting a conceptual data model for expressing the global schema makes query processing more involved than in the simplified framework usually considered in the literature. In particular, we present the surprising result that the semantics of a data integration system is best described in terms of a set of databases, rather than a single one, and this implies that, even in the global-as-view approach, query processing is intimately connected to the notion of querying *incomplete databases.* Then we formalize the notion of correct answer in a data integration system with a conceptual global schema, and the presentation of a query processing strategy that is able to provide all correct answers to a query posed to the system.

2 The Conceptual Data Model

We present the conceptual model which is at the basis of the integration framework introduced in the next section. The model incorporates the basic features of the *Entity-Relationship* (ER) model [8], extended with subset (or is-a) constraints on both entities and relationships. Other characteristics that are not considered in this paper for the sake of simplicity (e.g., domain of attributes, identification constraints, etc.), can also be added without affecting the results in the next sections.

Conceptual schemas in our approach are depicted in the usual graphical notation of Entity-Relationship schemas, with the addition of is-a arcs. An is-a arc from an entity (resp. relationship) box to another denotes that the instances of the first entity (resp. relationship) are a subset fo those of the second.

An attribute A of an entity X can be *mandatory* or *functional.* If A is mandatory, then each instance of X must have a value for A. If A is functional, then each instance of X has at most one value for A.

The language we use to express queries over a global schema expressed in our conceptual model is that of conjunctive queries. Formally, a *conjunctive query* (CQ) Q of arity n is written in the form

$$Q(x_1, \ldots, x_n) \ \leftarrow \ conj(x_1, \ldots, x_n, y_1, \ldots, y_m)$$

where $conj(x_1, \ldots, x_n, y_1, \ldots, y_m)$ is a conjunction of atoms involving constants and variables $x_1, \ldots, x_n, y_1, \ldots, y_m$ from an alphabet of variables. The predicates in the atoms are the so-called *concepts* of the conceptual schema, i.e., its entities, relationships and attributes:

- Each entity E has an associated predicate E of arity 1. Intuitively, $E(c)$ asserts that c is an instance of entity E.
- Each attribute A for an entity E has an associated predicate A of arity 2. Intuitively, $A(c, d)$ asserts that c is an instance of entity E and d is the value of attribute A associated to c.
- Each relationship R among the entities $E_1, \ldots, E_n$ has an associated predicate R of arity n.
- Each attribute A for a relationship R among the entities $E_1, \ldots, E_n$ has an associated predicate A of arity $n + 1$. Intuitively, $A(c_1, \ldots, c_n, d)$ asserts that $(c_1, \ldots, c_n)$ is an instance of relationship R and d is the value of attribute A associated to $(c_1, \ldots, c_n)$.

3 The Formal Framework for Data Integration

We set up a formal framework for data integration. Formally a *data integration system* $\mathcal{I}$ is a triple $\langle \mathcal{G}, \mathcal{S}, \mathcal{M}_{\mathcal{G},\mathcal{S}} \rangle$, where $\mathcal{G}$ is the global schema, $\mathcal{S}$ is the source schema, and $\mathcal{M}_{\mathcal{G},\mathcal{S}}$ is the mapping between $\mathcal{G}$ and $\mathcal{S}$.

We describe the characteristics of the various components of a data integration system. The *global schema* $\mathcal{G}$ is expressed in the conceptual data model described in the previous section. The *source schema* $\mathcal{S}$ is constituted by the schemas of the source relations. Note that we assume that the sources are expressed as relational data bases. This is not a strong limitation, since, in case of sources of different type, we can assume that suitable wrappers present the data at the source in relational form. The *mapping* $\mathcal{M}_{\mathcal{G},\mathcal{S}}$ between $\mathcal{G}$ and $\mathcal{S}$ is given by associating to each concept C (either entity, relationship, or attribute) in the global schema a query $\mathcal{V}_C$, of the same arity of the predicate associated to C, over the sources. We do not pose any constraint on the language used to express the queries in the mapping: we simply assume that the language is able to express computations over relational databases.

In order to assign semantics to a data integration system $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M}_{\mathcal{G},\mathcal{S}} \rangle$, we start by considering a *source database* for $\mathcal{I}$, i.e., a database $\mathcal{D}$ for the source schema $\mathcal{S}$. Based on $\mathcal{D}$, we now specify which is the information content of the global schema $\mathcal{G}$. We call *global database* for $\mathcal{I}$ any database for $\mathcal{G}$. A global database $\mathcal{B}$ for $\mathcal{I}$ is said to be *legal* with respect to $\mathcal{D}$, or, simply, *legal for $\mathcal{I}$ with respect to $\mathcal{D}$*, if:

- $\mathcal{B}$ is legal with respect to $\mathcal{G}$,
- for each element e of $\mathcal{G}$, the set of tuples $e^{\mathcal{B}}$ that $\mathcal{B}$ assigns to e is coherent with set of tuples computed by the associated query $\mathcal{V}_e$ over $\mathcal{D}$, i.e., $\mathcal{V}_e^{\mathcal{D}} \subseteq e^{\mathcal{B}}$.

The above definition implies that sources are considered *sound*: the data they provide to the integration system satisfy the global schema, but are not necessarily complete [11].

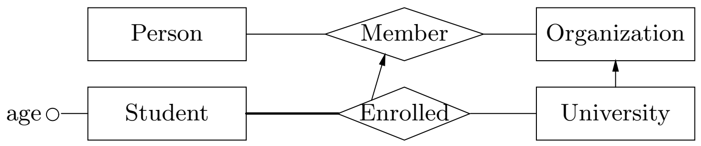

Fig. 1. Global schema of Example 1

Example 1. Figure 1 shows the global schema $\mathcal{G}_1$ of a data integration system $\mathcal{I}_1 = \langle \mathcal{G}_1, \mathcal{S}_1, M_1 \rangle$, where age is a functional attribute, $\mathsf{Student}$ has a mandatory participation in the relationship $\mathsf{Enrolled}$, $\mathsf{Enrolled}$ is-a Member, and $\mathsf{University}$ is-a $\mathsf{Organization}$. The schema models persons who can be members of one or more organizations, and students who are enrolled in universities. Suppose that $\mathcal{S}_1$ is constituted by $\mathsf{s}_1, \mathsf{s}_2, \mathsf{s}_3, \mathsf{s}_4, \mathsf{s}_5, \mathsf{s}_6, \mathsf{s}_7, \mathsf{s}_8$, and that the mapping $\mathcal{M}_1$ is as follows:

$$
\begin{array}{rlrl}
\mathsf{Person}(x) & \leftarrow \mathsf{s}_1(x) & \mathsf{Student}(x) & \leftarrow \mathsf{s}_3(x,y) \vee \mathsf{s}_4(x,z) \\
\mathsf{Organization}(x) & \leftarrow \mathsf{s}_2(x) & \mathsf{University}(x) & \leftarrow \mathsf{s}_5(x) \\
\mathsf{Member}(x,y) & \leftarrow \mathsf{s}_7(x,z), \mathsf{s}_8(z,y) & \mathsf{Enrolled}(x,y) & \leftarrow \mathsf{s}_4(x,y) \\
 & & \mathsf{age}(x,y) & \leftarrow \mathsf{s}_3(x,y) \vee \mathsf{s}_6(x,y,z)
\end{array}
$$

The *semantics* of a data integration system $\mathcal{I}$ with respect to a source database $\mathcal{D}$ for $\mathcal{I}$, denoted $sem(\mathcal{I}, \mathcal{D})$, is the set of global databases that are legal for $\mathcal{I}$ with respect to $\mathcal{D}$.

From this definition it is easy to see that, given a source database $\mathcal{D}$, different situations are possible. The first case is when no legal global database exists. This happens, in particular, when the data at the sources retrieved by the queries associated to the elements of the global schema do not satisfy the functional attribute constraints. We assume that such a problem is solved by the queries extracting data at the sources, that implement suitable data cleaning strategies that ensures the satisfaction of all functional attribute constraints. The interested reader is referred to [10] for more details of data cleaning techniques. The second case occurs when several legal global databases exist. This happens, for example, when the data at the sources retrieved by the queries associated to the global relations do not satisfy the is-a relationships of the global schema. In this case, it may happen that several ways exist to add suitable objects to the elements of $\mathcal{G}$ in order to satisfy the constraints. Each such ways yields a legal global database. The problem of incomplete information in the global-as-view approach is overlooked in traditional data integration systems, which either express the global schema as a set of plain relations, or consider the sources as exact (see, for instance, [7, 17, 3]).

Example 2. Referring again to Example 1, consider a source database $\mathcal{D}_2$, where s_1 stores p_1 and p_2, s_2 stores o_1, s_5 stores u_1, and s_4 stores (t_1, a_1), and the pairs (p_1, o_1) and (p_2, u_1) are in the join between s_7 and s_8. By the mapping $\mathcal{M}_1$, it follows that in every legal database of $\mathcal{I}_1$, $p_1, p_2 \in \mathsf{Person}$, $(p_1, o_1), (p_2, u_1) \in \mathsf{Member}$, $o_1 \in \mathsf{Organization}$, $t_1 \in \mathsf{Student}$, $u_1 \in \mathsf{University}$. Moreover, since $\mathcal{G}_1$ specifies that $\mathsf{Student}$ has a mandatory participation in the relationship $\mathsf{Enrolled}$, in every legal database for $\mathcal{I}_1$, t_1 *must* be enrolled in a certain university.

We conclude the section by defining the notion of query posed to the data integration system. Since, given a source database $\mathcal{D}$, several global databases may exist that are legal for $\mathcal{I}$ with respect to $\mathcal{D}$, we say that a tuple $(c_1, \ldots, c_n)$ is considered an answer to the query only if it is a *certain* answer, i.e., it satisfies the query in *every* database that belongs to the semantics of the data integration system.

Example 3. Referring to Example 2, consider the query Q_1 to $\mathcal{I}_1$:

$$\mathsf{Q}_1(x) \leftarrow \mathsf{Member}(x, y), \mathsf{University}(y)$$

It is easy to see that $\{p_2, t_1\}$ is the set of certain answers to Q_1 with respect to $\mathcal{I}_1$ and $\mathcal{D}_2$.

4 Answering Queries over the Global Schema

We present an algorithm for computing the set of certain answers to queries posed to a data integration system. The key feature of the algorithm is to reason about both the query and the conceptual global schema in order to infer which are the certain answers to the query. Indeed, we show that a simple unfolding strategy does not work in our setting.

Example 4. Consider again Example 3, and suppose we simply unfold the query Q_1 in the standard way, by substituting each atom with the query that $\mathcal{M}_1$ associates to the element in the atom. Then we get the query $\mathsf{q}(x) \leftarrow \mathsf{s}_7(x, z), \mathsf{s}_8(z, y), \mathsf{s}_5(y)$. If we evaluate this query over $\mathcal{D}_2$, we get $\{p_2\}$ as result, thus missing the certain answer t_1.

Next we illustrate our algorithm for computing all certain answers. The algorithm is able to add more answers to those directly extracted from the sources, by exploiting the semantic conditions expressed in the conceptual global schema. Let $\mathcal{I} = \langle \mathcal{G}, \mathcal{S}, \mathcal{M}_{\mathcal{G},\mathcal{S}} \rangle$ be an integration system, let $\mathcal{D}$ be a source database, and let Q be a query over the global schema $\mathcal{G}$. From the query Q, the algorithm obtains a new query $exp_{\mathcal{G}}(Q)$ over the elements of the global schema G in which the knowledge in $\mathcal{G}$ that is relevant for Q has been compiled in. The rest of the processing is quite obvious: $exp_{\mathcal{G}}(Q)$ is unfolded according to $\mathcal{M}$, and the resulting unfolded query $unf_{\mathcal{M}_{\mathcal{G},\mathcal{S}}}(exp_{\mathcal{G}}(Q))$ is evaluated over the source database $\mathcal{D}$.

The expansion of the user query Q requires to find a way to compile into the query the semantic relations holding among the concepts of the global schema $\mathcal{G}$. The basic idea to do so is that the relations among the elements in $\mathcal{G}$ can be captured by a suitable *rule base* $\mathcal{R}_{\mathcal{G}}$. To build $\mathcal{R}_{\mathcal{G}}$, we introduce a new predicate P', called *primed* predicate, for each predicate P associated to an element P of $\mathcal{G}$. Then, from the semantics of the ER schema we devise the following rules (expressed in Logic Programming notation [18]):

– for each entity E, attribute A and relationship R in $\mathcal{G}$, we have:

$$\begin{array}{rcl} E'(x) & \leftarrow & E(x) \\ A'(x, y) & \leftarrow & A(x, y) \\ R'(x_1, \ldots, x_n) & \leftarrow & R(x_1, \ldots, x_n) \end{array}$$

– for each is-a relation between E and E_i, or between R and R_i in an entity or relationship definition of $\mathcal{G}$, we have:

$$\begin{array}{l} E'_i(x) \;\leftarrow\; E'(x) \\ R'_i(x_1, \ldots, x_n) \;\leftarrow\; R'(x_1, \ldots, x_n) \end{array}$$

– for each attribute A for an entity E or a relationship R in an attribute definition in $\mathcal{G}$, we have:

$$\begin{array}{l} E'(x) \;\leftarrow\; A'(x, y) \\ R'(x_1, \ldots, x_n) \;\leftarrow\; A'(x_1, \ldots, x_n, y) \end{array}$$

– for each relationship R involving an entity E_i as i-th component according to the corresponding relationship definition in $\mathcal{G}$, we have:

$$E'_i(x_i) \;\leftarrow\; R'(x_1, \ldots, x_i, \ldots, x_n)$$

– for each mandatory participation of an entity E in a relationship R_j in an entity definition of $\mathcal{G}$, we have:

$$R'_j(f_1(x), \ldots, x, \ldots, f_n(x)) \;\leftarrow\; E'(x)$$

where f_i are fresh Skolem functions [18].

– for each mandatory attribute A for an entity E or a relationship R in an attribute definition of $\mathcal{G}$, we have:

$$\begin{array}{l} A'(x, f(x)) \;\leftarrow\; E'(x) \\ A'(x_1, \ldots, x_n, f(x)) \;\leftarrow\; R'(x_1, \ldots, x_n) \end{array}$$

where f is a fresh Skolem function.

Once we have defined such a rule base $\mathcal{R}_{\mathcal{G}}$, we can use it to generate the query $exp_{\mathcal{G}}(Q)$ associated to the original query Q. This is done as follows:

1. First, we rewrite Q by substituting each predicate P in the body of Q with P'. We denote by Q' the resulting query. In the following we call "primed atom" every atom whose predicate is primed.
2. Then we build a *partial resolution tree* for Q' [9]. In the partial resolution tree, a node is not expanded anymore either when no atom in the node unifies with a head of a rule, or when the node it is subsumed by (i.e., is more specific than) one of its predecessors. In the latter case, the node gets an empty node as a child; intuitively this is because such a node cannot provide any answer that is not already provided by its more general predecessor.
3. Finally we return as result the query $exp_{\mathcal{G}}(Q)$ formed as the union of all non-empty queries in the leaves of the partial resolution tree.

The following three observations are crucial for characterizing both the termination and the correctness of our algorithm:

– It is possible to show that the termination of the construction of the tree, and thus of the entire algorithm, is guaranteed.

– By exploiting results on partial evaluation of logic programs (see [9]), it can be shown that $exp_{\mathcal{G}}(Q)$ is equivalent to the original query Q with respect to the global schema $\mathcal{G}$, that is, for each database $\mathcal{B}$ that is legal for $\mathcal{G}$, the evaluation of Q yields the same result as $exp_{\mathcal{G}}(Q)$, i.e., $Q^{\mathcal{B}} = (exp_{\mathcal{G}}(Q))^{\mathcal{B}}$.
– The query $exp_{\mathcal{G}}(Q)$ returned by the algorithm is a union of conjunctive queries. Each disjunct of $exp_{\mathcal{G}}(Q)$ is a conjunctive query over the predicates of the global schema, i.e., the elements that have an associated query over the sources by virtue of the mapping.

The above observations imply that, if we evaluate $unf_{\mathcal{M}_{\mathcal{G},\mathcal{S}}}(exp_{\mathcal{G}}(Q))$ over the source database $\mathcal{D}$, we get exactly the set of certain answers of Q with respect to $\mathcal{I}$ and $\mathcal{D}$.

With regard to the characterization of the computational complexity of the algorithm, it is possible to show that , if the queries associated by $\mathcal{M}_{\mathcal{G},\mathcal{S}}$ to the elements of $\mathcal{G}$ can be evaluated in polynomial time in the size of the data at the sources, then evaluating $unf_{\mathcal{M}_{\mathcal{G},\mathcal{S}}}(exp_{\mathcal{G}}(Q))$ over $\mathcal{D}$ is also polynomial in the size of the data at the sources. It follows that our query answering algorithm is polynomial with respect to data complexity.

Example 5. Referring again to Example 3, it is possible to see that, by evaluating the unfolding of the query returned by the algorithm, the whole set of certain answers to Q_1 with respect to $\mathcal{I}_1$ and $\mathcal{D}_2$ is obtained. In particular, t_1 is obtained by processing the rule $\mathsf{Member}'(x, y) \leftarrow \mathsf{Enrolled}'(x, y)$, which takes into account that Member is a generalization of Enrolled and the rule $\mathsf{Enrolled}'(x, f(x)) \leftarrow \mathsf{Student}'(x)$, which expresses the mandatory participation of Student in Enrolled.

5 Conclusions

While it is a common opinion that query processing is an easy task in the global-as-view approach to data integration, we have shown the surprising result that, when the global schema is expressed in terms of a conceptual data model, even a very simple one, query processing becomes difficult. The difficulties basically arise because of the need of dealing with incomplete information, similarly to the case of the local-as-view approach to data integration.

After a logic-based characterization of the data integration system, we have presented a novel query processing algorithm that is able to compute all correct answers to a query posed to the global schema, by reasoning on both the query and the conceptual global schema. We have also shown that query processing, although exponential with respect to the size of the query and the global schema, remains of polynomial data complexity.

References

1. Serge Abiteboul and Oliver Duschka. Complexity of answering queries using materialized views. In *Proc. of PODS'98*, pages 254–265, 1998.

2. Sonia Bergamaschi, Silvana Castano, Maurizio Vincini, and Domenico Beneventano. Intelligent techniques for the extraction and integration of heterogeneous information. In *Proc. of the IJCAI'99 Workshop on Intelligent Information Integration*, 1999.
3. Mokrane Bouzeghoub and Maurizio Lenzerini. Introduction to the special issue on data extraction, cleaning, and reconciliation. *Information Systems*, 26(8):535–536, 2001.
4. Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, Daniele Nardi, and Riccardo Rosati. Information integration: Conceptual modeling and reasoning support. In *Proc. of CoopIS'98*, pages 280–291, 1998.
5. Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Moshe Y. Vardi. Answering regular path queries using views. In *Proc. of ICDE 2000*, pages 389–398, 2000.
6. Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Moshe Y. Vardi. View-based query processing and constraint satisfaction. In *Proc. of LICS 2000*, pages 361–371, 2000.
7. M. J. Carey, L. M. Haas, P. M. Schwarz, M. Arya, W. F. Cody, R. Fagin, M. Flickner, A. Luniewski, W. Niblack, D. Petkovic, J. Thomas, J. H. Williams, and E. L. Wimmers. Towards heterogeneous multimedia information systems: The Garlic approach. In *Proc. of the 5th Int. Workshop on Research Issues in Data Engineering – Distributed Object Management (RIDE-DOM'95)*, pages 124–131. IEEE CS Press, 1995.
8. P. P. Chen. The Entity-Relationship model: Toward a unified view of data. *ACM Trans. on Database Systems*, 1(1):9–36, March 1976.
9. Giuseppe De Giacomo. Intensional query answering by partial evaluation. *J. of Intelligent Information Systems*, 7(3):205–233, 1996.
10. Helena Galhardas, Daniela Florescu, Dennis Shasha, and Eric Simon. An extensible framework for data cleaning. Technical Report 3742, INRIA, Rocquencourt, 1999.
11. Gösta Grahne and Alberto O. Mendelzon. Tableau techniques for querying information sources through global schemas. In *Proc. of ICDT'99*, volume 1540 of *LNCS*, pages 332–347. Springer, 1999.
12. Jarek Gryz. Query folding with inclusion dependencies. In *Proc. of ICDE'98*, pages 126–133, 1998.
13. Alon Y. Halevy. Theory of answering queries using views. *SIGMOD Record*, 29(4):40–47, 2000.
14. Richard Hull. Managing semantic heterogeneity in databases: A theoretical perspective. In *Proc. of PODS'97*, 1997.
15. Alon Y. Levy. Logic-based techniques in data integration. In Jack Minker, editor, *Logic Based Artificial Intelligence*. Kluwer Academic Publisher, 2000.
16. Chen Li and Edward Chang. Query planning with limited source capabilities. In *Proc. of ICDE 2000*, pages 401–412, 2000.
17. Chen Li, Ramana Yerneni, Vasilis Vassalos, Hector Garcia-Molina, Yannis Papakonstantinou, Jeffrey D. Ullman, and Murty Valiveti. Capability based mediation in TSIMMIS. In *Proc. of ACM SIGMOD*, pages 564–566, 1998.
18. John W. Lloyd. *Foundations of Logic Programming (Second, Extended Edition)*. Springer, Berlin, Heidelberg, 1987.
19. Bertram Ludascher, Amarnath Gupta, and Maryann E. Martone. Model-based mediation with domain maps. In *Proc. of ICDE 2001*, pages 81–90, 2001.
20. Xiaolei Qian. Query folding. In *Proc. of ICDE'96*, pages 48–55, 1996.
21. Jeffrey D. Ullman. Information integration using logical views. In *Proc. of ICDT'97*, volume 1186 of *LNCS*, pages 19–40. Springer, 1997.
22. Ron van der Meyden. Logical approaches to incomplete information. In Jan Chomicki and Günter Saake, editors, *Logics for Databases and Information Systems*, pages 307–356. Kluwer Academic Publisher, 1998.