Using Description Logics to Model and Reason About Views

Alon Y. Levy[1] and **Marie-Christine Rousset** [2]

Several advanced applications of database systems require the modeling, maintenance, and usage of large collections of views. Prime examples include mediator systems that provide access to multiple information sources, data mining and archeology, mobile databases, data warehouses, and decision support systems. Furthermore, some database vendors are considering the maintenance of materialized views also as a means for query optimization. As a result, problems concerning materialized views have recently received a lot of attention in the database community.

A view in a relational database is essentially an answer to a query. If the answers to the query are physically maintained, the view is said to be materialized. Naturally, when there are many views (either materialized or not) there are complex relationships between the contents of various views. For example, one view may be guaranteed to be a superset of another, or two views may be guaranteed to be mutually disjoint. In order to perform tasks involving large collections of views, a system needs the capability to reason about the relationships between views, and about the relationship between a view and a query.

Example: Information Integration

To illustrate some of these problems, consider the application of providing uniform access to multiple information sources such as the many structured databases available today on the World Wide Web (WWW). Currently, to find a piece of information the user must first find the sources that may be relevant to his query, decide which ones to access, and then interact with each one individually, using their specific schema and query interface. Furthermore, there are no tools to aid the user in combining information from multiple structured sources. The goal of a mediator system is to free the user from these burdens. In particular, in such a system the user expresses *what* he or she wants and the system answers the query using the available sources. The system automatically finds the relevant sources, interacts with each one separately, and combines information from multiple sources to answer user queries. In order to be able to perform these tasks, the system requires a representation of the *contents* of the information sources.

The Information Manifold system [3] provides uniform access to structured information sources on the WWW. In the system the user poses queries using a set of relations and classes called the *world-view*. The contents of the sources are described as views over the world-view relations. As shown in [3], such an architecture has the advantage that it is possible to express the fine-grained distinctions about the contents of different sources; it is possible to easily add or delete information sources from the collection of available sources, and in doing so we do not have to change the world-view schema frequently.

However, in order to answer queries, the system must be able to reason about the contents of the views and their relationship to the given query. In particular, this entails detecting which sources have information that may overlap with a given query, which sources are disjoint from the query, and which sources are redundant given the content of other sources. More generally, the system must be able to find a way to answer the query given the set of sources, described as views. This has been called the problem of *rewriting a query using views*.

Description Logics

Description logics are declarative languages that have been designed especially for the purpose of representing and reasoning about interrelated sets of objects. A description logic is a subset of first order logic with equality that contains only unary relations, representing sets of objects in the domain (referred to as *concepts*) and binary relations (called *roles*). A concept describes a class of elements in the domain, and is defined by the conditions that must be satisfied by elements in the class. Several algorithms have been developed for performing various kinds of reasoning in description logics. Most importantly, algorithms have been developed to test *subsumption* of concepts, i.e., to determine that one concept is always a superset of another. Using subsumption algorithms one can also check whether two concepts are necessarily disjoint and whether two concepts can overlap. In particular, it is possible to detect when a concept is unsatisfiable, which is useful when dealing with large numbers of concepts.

Therefore it seems likely that description logics are a very natural formalism for modeling views, because the reasoning services provide some of the important tools needed to perform tasks that involve a large number of views. The idea is to model views as concepts in a description logic. While this approach has great promise, there are several problems that need to be addressed in order to close the gap between the current capabilities of description logics and the needs of applications using views in relational and object oriented systems. We describe below several research works that we have been pursuing in order to close this gap, and point out several open research questions. It should be noted that our discussion is limited to the usage of description logics as a modeling language of views. In order to use description logics in a large scale application, other issues need to be addressed as well, such as persistence, storage and efficient indexing.

It should be emphasized that for length considerations, this position paper includes only references to our own work. Related work is discussed in our papers.

[1] AT&T Laboratories, levy@research.att.com
[2] L.R.I. U.R.A C.N.R.S, University of Paris-Sud, mcr@lri.lri.fr

ECAI 96. *12th European Conference on Artificial Intelligence*
Edited by W. Wahlster
Published in 1996 by John Wiley & Sons, Ltd.

Closing the Gap

Relations with Higher Arity and More Complex Queries

A significant limitation of description logics is that they consider only unary and binary predicates. Furthermore, previous research has only considered answering atomic queries from description logic knowledge bases. That is, there are algorithms to answer queries of the form $C(s)$ or $R(s,t)$, where C is a description, R is a role and s,t are either objects or variables. In order to deal with views over relational databases, it is necessary to have the ability to model relations with arbitrary arity, and to answer a more rich class of queries, such as conjunctive queries, unions of conjunctive queries and recursive queries.

As a first step in providing this capability we have developed the CARIN family of languages [4]. The CARIN languages extend the expressive power of datalog with that of a description logic. A CARIN knowledge base includes a terminology $\mathcal{T}$ in some description logic $\mathcal{L}$, and a set of *extended* Horn rules. An extended Horn rule allows the antecedent to contain atoms whose predicate is a concept or a role defined in $\mathcal{T}$. The key issue that arises in CARIN is the design of sound and complete inference procedures. We have shown several important results concerning inference in CARIN. We have developed a sound and complete algorithm for reasoning in non-recursive CARIN knowledge bases whose description logic is $\mathcal{ALCNR}$ (which is a relatively expressive description logic) [4]. In particular, our result provides the first algorithm for answering arbitrary conjunctive queries from $\mathcal{ALCNR}$ knowledge bases. Our algorithm can also be extended to obtain a sound and complete test for subsumption of conjunctive queries over $\mathcal{ALCNR}$. In analogy with query containment algorithms in databases, such an algorithm is a key building block in many optimization algorithms. We have also considered the inference problem in CARIN knowledge bases that contain recursive rules [5]. We have shown that some of the basic constructors of description logics (namely $\forall R.C$ and $\leq n\, R$) each in isolation cause the inference problem to become undecidable when combined with recursive Horn rules. However, we have identified the maximal subset of $\mathcal{ALCNR}$ that can be combined with recursive Horn rules while maintaining decidability. CARIN (with the description logic CLASSIC) is used as the representation language of the Information Manifold system [3].

Answering Queries Using Views

As explained above, a central problem that arises in applications using materialized views is the problem of rewriting queries using views. Informally, the problem is the following. Let the database relations be $R_1, \ldots, R_m$, and $V_1, \ldots, V_n$ be views defined over the database relations. Given a query Q, can we find a query Q', that uses *only* the views, and is equivalent to Q over all database instances? Variants of the problem differ depending on the expressive power we allow in expressing $V_1, \ldots, V_n$ and Q.

As seen in our example, this problem is central to a system providing integrated access to a collection of data sources, because the system must find a way of answering a query using the sources, and these are described as views. This problem also arises in the context of the view maintenance problem and in applications where using the views may lead to performance improvements as opposed to accessing the database (e.g., mobile databases). Several authors have proposed solutions to this problem in the context of relational databases. In order to apply description logics to modeling of views, we must extend these solutions to views described by concepts in a description logic or more generally, CARIN.

In [1] it is shown that the problem of query containment stands at the core of the rewriting problem. The algorithm for *existential entailment* described in [4] generalizes containment checking to conjunctive queries over description logics, and therefore provides the key for solving the rewriting problem for CARIN. In order to solve the rewriting problem we need to specify the space of candidate rewritings that need to be checked. We are currently exploring how this space depends on different description logics.

Aggregation Functions

Grouping and aggregation are an important feature in database query languages such as SQL, and are particularly important in decision support applications that involve complex queries. As an example of using grouping and aggregation consider a relation

$$EMP(name, department, salary)$$

giving the department number and salary of each employee. For managerial purposes, one may want to obtain a table of department numbers, each with the maximal salary of an employee in the department. To solve this query, the query processor will group the tuples of EMP by the attribute $department$, and then for each group will compute the aggregate function, which in this case is the maximum salary. Naturally, large collections of views will have views that involve aggregation, and in order to use description logics for modeling views, an important problem is to extend description logics with the appropriate constructors and their associated subsumption algorithms. For example, we should be able to define the class C_1 to be the class of departments for which the maximum salary is less than \$100,000. Then, we should be able to infer that C_1 is a subsumed by the class C_2 which is defined to be the class for which the average salary is less than \$100,000. A collection of inference rules that enable deducing containment relations between relational views involving aggregation was described in [2] (though in general, the problem is undecidable).

REFERENCES

[1] Alon Y. Levy, Alberto O. Mendelzon, Yehoshua Sagiv, and Divesh Srivastava, 'Answering queries using views', in *Proceedings of the 14th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, San Jose, CA*, (1995).

[2] Alon Y. Levy and Inderpal Singh Mumick, 'Reasoning with aggregation constraints', in *Proceedings of the Conference on Extending Database Technology, EDBT-96*, (March 1996).

[3] Alon Y. Levy, Anand Rajaraman, and Joann J. Ordille, 'Query answering algorithms for information agents', in *Proceedings of the AAAI Thirteenth National Conference on Artificial Intelligence*, (1996).

[4] Alon Y. Levy and Marie-Christine Rousset, 'CARIN: a representation language integrating rules and description logics', in *Proceedings of the European Conference on Artificial Intelligence, Budapest, Hungary*, (1996).

[5] Alon Y. Levy and Marie-Christine Rousset, 'The limits on combining recursive horn rules and description logics', in *Proceedings of the AAAI Thirteenth National Conference on Artificial Intelligence*, (1996).