I. Khalil Ibrahim, V. Dignum, W. Winiwarter, E. Weippl, Logic Based Approach to Semantic Query Transformation for Knowledge Management Applications, Proc. of the International Conference on Knowledge Management (I-Know'02), Graz, Austria, 2002.

Logic Based Approach to Semantic Query Transformation for Knowledge Management Applications

Ismail Khalil Ibrahim, Virginia Dignum
Utrecht University, The Netherlands
{ismail, virginia}@cs.uu.nl

Werner Winiwarter
University of Vienna, Austria
werner.winiwarter@univie.ac.at

Edgar Weippl
Software Competence Center Hagenberg, Austria
edgar.weippl@scch.at

Abstract: In this paper, we address the problem of integrating answering queries using views (and more generally query folding) with semantic query optimization into one optimizer that uses the semantic knowledge about information sources expressed as integrity constraints to support the query processing and optimization of knowledge management applications.

Key Words: data integration, semantic query transformation, answering queries using views, knowledge management applications.
Categories: H.1, H.2, H.3, H.4.

1. Introduction

Knowledge management systems rely more and more on highly organized and integrated data to support informed and comprehensive decision-making. Data integration is vital to knowledge management systems because it weaves together information needed to make strategic decisions, provides quick and convenient access to data, improves the quality and comprehensiveness of the data, promotes consistency and reduces cost of data collection, storage and processing, and improves existing legacy infrastructure management systems to better serve society needs.
The task of a data integration system [Wiederhold, 92] is to provide users with an automated method to access, relate, and combine data stored in multiple, autonomous, possibly heterogeneous sources [Levy et al., 95]. As a consequence, answering queries in data integration systems became a critical problem that involves selecting the relevant information sources and generating query plans to retrieve and combine the data in a uniform way.
Significant progress has been made in source selection and plan generation [Adali et al., 96], [Bressan and Goh, 98], [Sheth and Larson, 90]; the critical issue has been shifting to query processing and optimization.
The traditional cost-based query optimization techniques are computationally expensive to be used in data integration systems because of the high connection

overheads, high computation time, financial charges and temporary unavailability, as well as other problems associated with manipulating sources with different data models, semantics, and access methods [Adali et al., 96].
Query processing and optimization in data integration systems is closely related to the problem of *query folding* [Adali et al., 96], [Gryz, 98], [Qian, 96] and the problem of *semantic query optimization* [Chakravarthy et al., 90], [Minker, 87]. Query folding [Qian, 96] is the general activity of determining how a query can be answered using a set of resources, which can be relation fragments, materialized views, cached results, or component sources of a network. Semantic query optimization [Chakravarthy et al., 90] is the process of optimizing "database" queries using semantic information expressed as integrity constraints. Semantic query optimization is concerned with the transformation of queries into semantically equivalent queries with higher potential for efficient evaluation. Semantic query optimization can help in providing local optimization to subqueries and has the potential for reducing unnecessary data transmission when the query can be answered without accessing the database if enough knowledge is contained in the integrity constraints [Bressan and Ibrahim, 99].
Query folding on the other hand has obvious applications in query processing in distributed databases and query answering in federated databases in addition to its potential for centralized databases [Qian, 96].
Answering queries using views [Chakravarthy et al., 90], [Duschka and Genesereth, 97], [Levy et al., 95] – as part of the query folding problem – has applications also in global information systems, mobile computing, and maintaining physical data independence [Levy et al., 95] and has the potential for optimizing query evaluation in information integration systems because it reduces the need to re-compute views and/or data being queried and hence speeds up the querying of large amounts of data.
In this paper, we address the problem of answering queries using views (and more generally query folding) with semantic query optimization into one optimizer that uses the semantic knowledge about information sources expressed as integrity constraints to support the query processing and optimization in knowledge management systems.

2. Query Processing in Data Integration Systems

In a data integration system, query processing is performed as follows:
An information mediator [Wiederhold, 92] maintains a global model that describes a set of integrated information sources. User queries are posed in terms of a mediated schema, that is a set of virtual relations in the sense that they are not stored anywhere which are designed for a particular data integration system. To allow the data integration system to retrieve data, each information source is wrapped by a relation that translates the component information source schema into a view definition over the mediated schema. In order to answer a user query, the query should be rewritten in a way to be defined over these view schemas, which are the only available information sources.
When the information mediator receives a query, it decomposes the query into a set of subqueries expressed in the global mediation model to retrieve and combine data from individual information sources. The wrappers will translate the subqueries then into the corresponding query language before the subqueries are executed. Finally, the

information mediation system combines the retrieved data and presents the results to the user [Adali et al., 96], [Duschka and Genesereth, 97a], [Friedman et al., 99].
We believe that the power of answering queries using views can be considerably enhanced by taking into account integrity constraints that are known to hold on the base information sources [Gryz, 98].
The rewritten query, which is expressed now in terms of the component information source schema rather than the global mediated schema, is transformed into a semantically equivalent set of queries that may be processed faster and more efficiently than the original one.
Our focus in this paper will be on *query rewriting*, i.e. translating user queries, which are defined over the global mediated schema, into an equivalent query defined over the component information source schemas.
We consider the problem of finding a rewriting that uses one or more of the available information sources, the problem of finding a partial rewriting, and finding a complete rewriting, i.e. a rewriting that uses only the available information sources.
The rewriting is obtained by specifying a mapping algorithm between the head of the query and the set of the available view definitions of the information sources.
The algorithm takes as input a conjunctive user query defined over the global mediated schema and, depending on a set of conjunctive view definitions and substitution rules, generates as output an equivalent rewriting of the user query, i.e. a conjunctive query that returns the same set of results as the original one.

3. Theoretical Background

We use Datalog to express queries, views, and integrity constraints. The reader unfamiliar with the syntax and semantics of Datalog can consult [Abiteboul et al., 95], [Minker, 87], or [Ullman, 89] among the many textbooks and papers available on this topic.
An information source (front-and back office applications, legacy and mainframe systems, data warehouses and data marts, flat files, transactional databases, and e-commerce applications) consists of a set of base relations R as its *extensional database*, a set of view definitions V as its *intentional database*, and a set of integrity constraints *IC*.
The base relations are expressed as ground unit clauses of the form: $R(A_1, \ldots\ldots, A_n)$
where R is a base relation predicate and $A_1, \ldots\ldots, A_n$ are the arguments.
The view definitions are expressed as non-recursive range-restricted definite Horn clauses (i.e. with at least one atom in the body and exactly one atom in the head) of the form: $V \leftarrow R_1, \ldots\ldots, R_n$
such that:
V is a view literal whose arguments are distinct variables
R_i is a base relation literal
Each variable in the head of the view definition also appears in the body
$\forall X_k \in V_j \quad \ldots\ldots k = 1, \ldots\ldots, n$
$\qquad\qquad\qquad\qquad\qquad\qquad j = 1, \ldots\ldots, k$
$\exists R_i$ *such that* $X_k \in R_i$
Integrity constraints are expressed as non-recursive range-restricted Horn clauses.

We consider conjunctive queries or Project-Select-Join (or PSJ for short) queries. The head of a conjunctive query is a query literal and its body consists of base literals and built-in predicates

$Q \leftarrow R_1, \ldots\ldots, R_n, C_1, \ldots\ldots, C_m$

Where Q is a query literal whose arguments are distinct variables, $R_1, \ldots\ldots, R_n$ are base relation literals, $C_1, \ldots\ldots, C_m$ are built-in predicates. Variables in the head of the query are distinguished. We assume that every distinguished variable also appears in the body. As a convention we use X, Y,… for distinguished variables, W, U,… for other variables, and A, B,… for constants.

Suppose that we have a set of relational view definitions stored in the data integration system. We consider the problem of:

Given a user query Q, how to find a rewriting Q' using one or more of the available views such that Q is contained in Q', i.e. the value of Q is a subset of Q' in every database. It is well known that containment implies equivalence.

We consider the problem of finding *partial rewriting* and finding *complete rewriting*.

A partial rewriting of a query Q using a set of views V is a conjunctive query Q' whose head is a query literal, such that $Q' \subseteq Q$ and the body of Q' contains one or more view literals defined in V (it may contain base literals).

A complete rewriting of Q using V is a partial rewriting of Q using V whose body contains views and built-in predicates only (it may not contain base literals).

4. Motivational Example

As an example let us consider the task of searching bibliographic records of technical reports from digital libraries available on the WWW.

There are different online collections available on the WWW, which provide such information. Unfortunately, up to now we do not have a unified catalog of bibliographic records like the OCLC for digital libraries. Probably the best example of such a digital library containing technical reports is the Networked Computer Science Technical Reports Library (http://cs-tr.cs.cornell.edu/) (NCSTRL) but this also provides only information of the sites that have installed the same software package (Dienst) and created their bibliographic records using the same format (RFC 1807).

Let us suppose that there are two information sources providing listings of technical reports that can be downloaded from the WWW. The first one provides information about titles, authors, and subjects of the technical reports. The second one provides information about the years, titles, and the URL from which you can download the report. Suppose we want to find which technical reports authored by Gio Wiederhold can be downloaded from Stanford Web site as well as their subjects.

None of these two digital libraries in isolation can answer this query. However, by combining data from multiple sources, we can answer queries like this one, and even more complex ones. To answer our query, we would first send a query q_1 to the first site (S_1) to retrieve the titles and subjects of all the technical reports authored by Gio Wiederhold, and then send another query q_2 to the second site (S_2) to retrieve the titles of all the technical reports that can be downloaded from Stanford site. Finally, join the results of the queries q_1 and q_2 to get the answer to our query.

The reader may start thinking about the network performance issues, the high connection overheads, high computation time, financial charges, and temporary

unavailability, as well as other problems of manipulating sources with different data models, semantics, and access methods [Adali et al., 96].

5. A Logic Based Approach to Semantic Query Transformation

In traditional databases, query evaluation can be partitioned into a compilation phase followed by a modification and evaluation phase [Chakravarthy et al., 90].
[Chakravarthy et al., 90] introduced a two phased approach to using integrity constraints for optimizing queries over a single database; the semantic compilation phase where integrity constraint fragments, called residues, are computed and associated with database rules. The result is a set of semantically constrained axioms. When a query is given to the system, the semantic transformation phase uses these stored residues to generate semantically equivalent queries that may be processed faster than the original query.
In our approach to query processing and optimization in information integration systems, four phases are identified:
First phase: The substitution rules generation phase where base relations R are expressed in term of the view definitions V. The result of this phase is a set of substitution rules S that contain all the necessary information for the query rewriting phase.
Second phase: The semantic compilation phase, which is the same as the one introduced in [Chakravarthy et al., 90], where integrity constraints ICs are used to compute residues. These residues are then attached to database rules. The resulting semantically constrained axioms SCA are filtered and stored for later use.
Third phase: The query rewriting phase; when a query Q defined over the base relations is posed to the system, this phase uses the set of substitution rules generated earlier to rewrite the query in such a way as to generate an equivalent query Q' that produces the same answer as the original one but defined over the component views.
Fourth phase: The semantic query transformation phase which uses the semantically constrained axioms to transform the rewritten query Q' into one or more semantically equivalent queries.

6. Conclusion

In this paper, we present a logic-based approach to semantic query transformation [Ibrahim, 01]. By integrating semantic query optimization and answering queries using views (and more generally query folding) we can considerably enhance query processing and optimization in distributed, possibly heterogeneous knowledge management systems by trying to push the costly operations to the sources as much as possible and hence reducing processing costs and/or response time.
In our approach, each of the component information sources is seen as consisting of one or more materialized views. Materialized views, also called view instances, are the stored results of previously executed queries. Only the component information sources schemas and view definitions, i.e. the queries that generate the views, are stored in the data integration system. The semantic query compilation takes place in the component databases a priori and only the semantically constrained axioms are stored in the data integration system. In contrast to cost-based query optimization,

where the mediator does not have access to source statistics information and it may not be easy to model the source's performance, the view definitions and the semantically constrained axioms stored in the mediator contain enough knowledge to increase the potential for efficient query evaluation. For instance, in the simplest case, a query may be answered without accessing a particular data source if it is known a priori that there is no answer or all the data at that source is inconsistent with the constraints.

References

[Abiteboul et al., 95] Abiteboul, S., Hull, R., and Vianu, V., Foundations of Databases. Addison Wesley, 1995.

[Adali et al., 96] Adali, S., Candan, K., Papakonstantinou, Y., and Subrahmanian, V. S., Query Caching and Optimization in Distributed Mediator Systems. In Proc. Conf., ACM SIGMOD, pages 137-148, 1996

[Bressan and Ibrahim, 99] Bressan S., and Ibrahim, I. K., Semantic Query Transformation for the Integration of Autonomous Information Sources. In Proc. Conf., Application of Prolog (INAP'99), 1999.

[Chakravarthy et al., 90] Chakravarthy, S., Grant, J., and Minker, J., Logic Based Approach to Semantic Query Optimization. ACM Transactions on Database Systems, 1990.

[Duschka and Genesereth, 97] Duschka, O., and Genesereth, M., Answering Queries Using Recursive Views. In Proc. ACM SIGACT-SIGMOD-SIGART Symposium on Principles Of Database Systems (PODS), Tucson, Arizona., 1997.

[Friedman et al., 99] Friedman, M., Levy, A., and Millstein, T., Navigational Plans for Data Integration, In Proc. National Conference on Artificial Intelligence, 1999.

[Gryz, 98] Gryz, J., An Algorithm for Query Folding with Inclusion Dependencies. In Proc. Workshop on Intelligent Information System, 1998.

[Ibrahim, 01] Ibrahim, I. K., Semantic Query Transformation for the Intelligent Integration of Information Sources, Ph.D. thesis, Gadjah Mada University, 2001.

[Levy et al., 95] Levy, A., Mendelzon, A., Sagiv, Y., and Srivastava, D., Answering Queries Using Views. In Proc. 14th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS), pages 95-104, San Jose, CA, 1995.

[LIoyd, 84] Lloyd, J., Foundation of Logic Programming. Springer-Verlag, 1984.

[Minker, 87] Minker, J., Foundation of Deductive Databases and Logic Programming. Morgan Kaufmann, 1987.

[Qian, 96] Qian, X., Query Folding. In Proc. Data Engineering (ICDE), pages 48-55, New Orleans, LA, 1996.

[Sheth and Larson, 90] Sheth, A., and Larson, J., Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Database. ACM Computing Surveys, 22, 3, pages 183-236, 1990.

[Ullman, 89] Ullman, J., Principles of Database and Knowledge Base Systems Volumes I, II. Computer Science Press, Rockville MD, 1989.

[Wiederhold, 92] Wiederhold, W., Mediators in the Architecture of Future Information Systems. IEEE Computer, pages 38-49, March 1992.