An Object-Oriented Graph Traversal Algorithm for Data Mediation

Lawrence Henschen Tania Neild Chris Fernandes

Department of Electrical and Computer Engineering
Northwestern University
Evanston, Illinois, 60208-3118
Email: {henschen,neild,cfernand}@eecs.nwu.edu
TEL: {847-467-2299,847-491-7642} FAX: 847-491-4455

Introduction

Research in heterogenous distributed databases has been fueled by two opposing forces. The proliferation of personal workstations and small servers plus the geographical distribution of mainframe computers leads to the decentralization of information; on the other hand, management has ever increasing needs to access and combine this physically dispersed information. Moreover, the information developed independently at separate sites in a distributed database will have data that varies in both syntax and semantics, leading to the need for mediation of query and information exchange between different databases [QL94]. The variety of discrepancies that must be mediated has been well documented such as in [SPD92].

Many solutions to this problem have been proposed, most based on the notion of schema integration [CGH$^+$94]. When the data at different sites are sufficiently closely related, such a simple approach will be both satisfactory and efficient. However, as the syntactic and semantic differences increase, the simple approaches will naturally fail, and more complex methods must be developed [SLC96]. Our research is aimed at this far end of the spectrum.

Our research is based on a *mediator* which models real world knowledge in an object-oriented-style graph. The mediator is used to indicate under what circumstances and in what manner related concepts can be interchanged to answer queries across different members of a distributed system.

The next section gives an overview of our approach. The third section contains an example illustrating the ability of our approach to handle both syntactic and semantic discrepancies.

The Annotated Global Graph

A central object-oriented-style graph models the various concepts present in the member databases. Such a graph can model a rich set of relationships like ISA, PART-OF, AGGREGATE-OF, etc., which in turn can help distinguish a rich set of semantic variations such as differences between a teacher's name vs. a student's name vs. an advisor's name etc. Links between nodes in this graph carry annotations that are used both to modify queries being sent to another database and to transform data being returned to the originator of the query. Items in the schema of the member databases are attached to nodes in the mediator graph by client and server links, which are also annotated. *Client links* are traversed to send a query originating in the member to the mediator graph. Data retrieved by the members in response to a query from the mediator graph are returned along *server links*. The combination of attaching to particular nodes in the mediator graph (e.g. attaching to "teacher name" instead of "students name") plus the modifications produced by the annotations can account for a wide variation in syntax and semantics between related data in different members of the federation.

In each case of link traversal, the annotations are used to modify the query/data. This could involve transforming the format to account for a syntactic difference, adding some condition or constraint to account for a semantic difference, or even blocking traversal when the conditions already encountered are inconsistent with the relationship expressed by that link.

This representation leads naturally to a five step process for solvng a client's query:

Step 1. Trace from query attributes in the client database along client links to mediator graph nodes.

Step 2. Trace through the mediator graph and out along server links to attributes in other databases.

Step 3. The servers answer their queries by retrieving and formatting all server link attributes.

Step 4. Server tuples are returned to the mediator graph where additional processing (e.g. JOINing) may occur.

Step 5. The resulting tuples are formatted and returned to the client.

An Example

Database 1: Relational	Database 2: Relational	Database 3: Object-Oriented
d(T-Name, Class, salary)	Enrollment(S-Name, Advisor, Class)	Person(name->n) Student:Person(advisor->a, class->[]) Teacher:Person(class->c, department->d, salary->s)
T-Name teaches a class for a salary paid by department d.	A teacher advises the student, S-Name, to enroll in the class.	A person has a name. Student and Teacher are subclasses of Person. A student has an advisor, and has taken a set of classes. A teacher is paid a salary by a department to teach a class.

Table 1: A set of University Databases

Table 1 presents a set of 3 university databases, and Figure 1 shows a mediator graph for this federation. Consider the query "What students took classes taught by their history advisor?" arising in DB3. Some answers can be derived directly in DB3. But additional answers are derivable by combining data from DB1 and DB2.

Step 1: DB3 is the client. The SQL form of the query is

SELECT s.name FROM student s, teacher t

WHERE t.class in s.class and t.dept = HISTORY and s.advisor = t.name

The client links from Student.Name, Teacher.Name, Teacher.Department, Teacher.Class, Student.Class, and Student.Advisor are traversed into the mediator graph reaching three mediator concepts.

Step 2: The solid line in Figure 1 indicates a path in which the annotations are consistent.

Step 3: Five server links are followed to the History and Enrollment tables. Notice that the Teacher.Name server link annotated with Holds(Department, ISA, English) is blocked because it is inconsistent with Holds(Department,

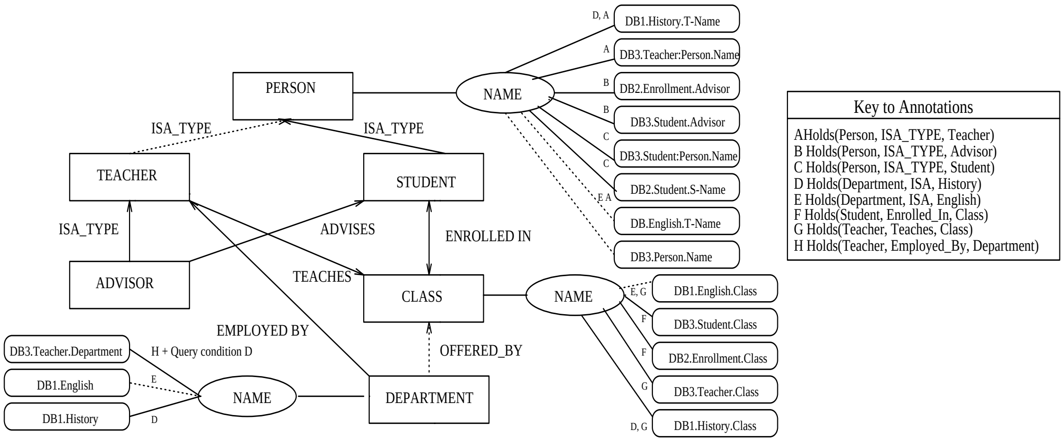

Figure 1: Mediator Graph

ISA, History). DB1 retrieves tuples which include only history teachers. DB2 also retrieves tuples, but these tuples may include advisors who are not teachers in the history department.

Step 4: DB1 and DB2 tuples are returned. Since only attributes from the traversed server links are returned, DB1.salary is projected out. These tuple sets are joined. Note that DB2 tuples where the teacher is not in the history department will not be in the result of the join.

Step 5: The new tuples (Teacher Name, Student Name, Class Name) are consistent with the client query and are reformatted and returned to DB3, the client.

Concluding Remarks

Current federated and integration approaches would require DB2's advisor attribute to be directly equated to a teacher's name for the above query to be answered completely. However, this mapping leaves the query "select all teachers who are not advisors" unable to be answered. Our approach allows the annotations to distinguish such subtle semantic differences.

References

[CGH+94] D. Clements, M. Ganesh, S.-Y. Hwang, E.-P. Lim, K. Mediratta, J. Srivastava, and H.R. Yang. Myriad: Design and implementation of federated database prototype. In *Proceedings of the International Conference on Computer Systems andEducation*, India, June 1994.

[QL94] Xiaolei Qian and T.F. Lunt. Semantic interoperation: A query mediation approach. Technical Report 02, Computer Science Laboratory, SRI International, April 1994.

[SLC96] Peter Scheuermann, Wen-Syan Li, and Chris Clifton. Dynamic integration and query processing with ranked role sets. In *Proceedings of First IFCIS International Conference on Cooperative Information Systems (CoopIS'96)*, Brussels, Belgium, June 1996.

[SPD92] Spaccapietra, Parent, and Dupont. Model independent assertions for integration of heterogenous schemas. *VLDB Journal*, pages 81–126, 1992.