An Object-Oriented Graph Traversal Algorithm for Data Mediation

Larence Henschen  
*Department of Electrical and Computer Engineering, Northwestern University*, henschen@eecs.nwu.edu

Tania Neild  
*Department of Electrical and Computer Engineering, Northwestern University*, neild@eecs.nwu.edu

Chris Fernandes  
*Department of Electrical and Computer Engineering, Northwestern University*, cfernand@eecs.nwu.edu

Follow this and additional works at: http://aisel.aisnet.org/amcis1996

Recommended Citation

Henschen, Larence; Neild, Tania; and Fernandes, Chris, "An Object-Oriented Graph Traversal Algorithm for Data Mediation" (1996).  
AMCIS 1996 Proceedings. 75.  
http://aisel.aisnet.org/amcis1996/75

This material is brought to you by the Americas Conference on Information Systems (AMCIS) at AIS Electronic Library (AISeL). It has been accepted for inclusion in AMCIS 1996 Proceedings by an authorized administrator of AIS Electronic Library (AISeL). For more information, please contact elibrary@aisnet.org.
Introduction

Research in heterogeneous distributed databases has been fueled by two opposing forces. The proliferation of personal workstations and small servers, plus the geographical distribution of mainframe computers, has led 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 has data that varies in both syntax and semantics. These forces lead to the need for mediation of query and information exchange between different databases [Q94]. 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 [CGH93]. 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 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 including a description of the mediator components, as well as, a five step process for the query mediation. The third section contains an example which illustrating the ability of our approach to handle both syntactic and semantic discrepancies. The final section indicates the benefits of our approach.

The Annotated Global Graph

A central object-oriented-style graph models the various concepts 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 like differences between a teacher's name, a student's name, and an advisor's name etc. Links between nodes in this graph carry annotations that are used to both 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 "student 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.

There are three type of annotations on a link: data functions, preconditions, and postconditions. In each link traversal, the data functions may be used to modify the query and/or data. This could involve transforming the format to account for a syntactic difference, adding some condition or constraint to
account for a semantic difference. Furthermore, for every link traversal along a given path, the preconditions of the link must be consistent with the postconditions of all previously traversed links in the path. This limits the paths by 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 solving a client's query:

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

Table 1.0: A Set of University Databases

Step 1. The edges from query attributes in the client database to mediator graph nodes are traced. The set of mediator graph nodes which are reached is the mediator query set. The union of the postconditions on these client links form the initial state for which all subsequent link traversals must be consistent.

Step 2. All paths between the nodes of the mediator query set a connected, and traversal continues along the server links into the other local databases.

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

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

Table 1.0 presents a set of 3 university databases, and Figure 1.0 shows a mediator graph for this distributed system. In this example, annotations on a link are preconditions if the link is being traversed in the direction of the link (with the arrowhead). Similarly, annotations on a link are postconditions if the link
is being traversed against the direction of the link. Lastly, the data functions have been omitted from the graph for simplicity.

Consider the query "What students took classes taught by their history advisor?" arising in DB3. Some answers can be derived directly in DB3. However, 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. The initial state is \{A,B,C,D,F,G,H\} in which D is the D postcondition which has been instantiated with the query's "HISTORY" constant.

**Step 2:** The solid line in Figure 1.0 indicates a path in which the preconditions are consistent with the annotations set \{A,B,C,D',F,G,H\}.

**Step 3:** Five server links are traversed 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 D': Holds(Department, 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.
Figure 1.0: Mediator Graph

**Step 4:** DB1 and DB2 tuples are returned. Since only values from the traversed server links are returned, DB1.salary is not included. These tuple sets are joined. Note that DB2 tuples, where the teacher is not in the history department, are not 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 federation and integration approaches require DB2's advisor attribute to be directly equated to the teacher name concept in DB1 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**


I, L.Henschen understand and agree to the following:

1. The paper *An Object-Oriented Graph Traversal Algorithm for Data Mediation* is not under publication consideration elsewhere.
2. The above paper will be published in the AIS proceedings.
3. At least one of the authors will attend the conference and present the paper at the designated session.
4. Authors retain the copyright ownership of papers appearing in the conference proceedings.

Sign:


Sign

No additional audio visual are needed!