Query Translation in Database sharing Peer to Peer networks

Mduduzi Nxumalo
mnxumalo@cs.uct.ac.za

Sonia Berman
sonia@cs.uct.ac.za

Department of Computer Science, University of Cape Town, 2005

Abstract

In a peer to peer database sharing network users query data from all peers using one query as if they are querying data from one database. Implementing such a facility requires solutions to the problems of schema conflicts and query translation. Query translation is the problem of rewriting a query posed in terms of one schema to the query in terms of the other schema. Schema conflicts refer to the problems which come as the results of integrating data from databases which were designed independently. This paper proposes the architecture for integrating and querying databases in the peer to peer (P2P) network.

Categories and Subject Descriptors

Peer to Peer network, Data Integration

Keywords:

Query Translation, Schema Matching

1 Introduction and Motivation

Query translation in a file sharing P2P network is easier than in database sharing [9]. This is because data is semantically rich in database sharing and this gives users the capability to make rich queries. The problems which come as the result of sharing semantically rich and heterogeneous data are structural, naming and semantic conflicts [13].

Query translation in the peer to peer system is the problem of querying data stored in different peers using one query. This problem can be related to the problem of data integration. The data integration systems integrate data from different data sources and users query this mediated data as if they were querying data from one database. The advantage of implementing data integration systems is that data can be administered and updated locally but still be shared.

The proposed solution to the problem of query translation is based on capturing enough metadata about the data stored by each peer. This metadata is captured as the peer is introduced to the system. The metadata is about attributes and relations in the schema of a peer. The metadata about relations is about restrictions posed on the definition of tuples the administrator is willing to share with other peers. Metadata about attributes is about their data types and general knowledge about attributes. All this metadata is used to define the kind of queries a peer is able to answer. Knowing the kind of queries the peer is able to answer can help to reduce avoidable connections to peers. The

proposed solution to the queries which involve aggregate and grouping functions has two steps. The first step writes queries to get data from peers and storing them in the temporary results table. The second step queries the results table to compute and format the final response (s).

The rest of this paper is organized as follows. Section 2 gives background theory. Section 3 describes the proposed architecture. Section 4 gives the summary of the results. Section 5 summaries the paper and makes conclusions. Section 6 gives refernces.

2 Background

2.1 Schema Mediation

The problem of sharing pre existing heterogeneous data is applicable in many areas in databases. These areas are information integration, peer-to-peer data management, data exchange and data warehousing [19, 20]. All these problems need schema mappings. Schema mapping [18, 19] is the problem of describing the relationship between database schemas. Schema mappings only describe the relationship between schemas without considering structural and representation conflicts [19, 13].

Data integration [6, 18] is the problem of relating data stored in different databases using a global schema. There are differences and similarities between the problem of data exchange and data integration. The difference is that in data integration data is not moved from one schema to the other but it is queried as if it was in one database. In data exchange data is migrated from the source schema to the target schema. One of the similarities is that they both need schema mapping to associate data from one schema to the other [18].

2.1.1 Schema Mapping

The problem of schema mapping is applicable when querying data from different sources. The traditional approach is to start by defining mappings between schemas of data sources. The architecture designed in [13] suggests using XML to map database schemas from different sources. In this architecture all local schemas are transformed to XML and the global schema is formed by integrating local schemas.

The architecture in [2] suggests using mapping tables. Mapping tables [2, 19] define the relationship between shared data. These tables list pairs of matching data values [19]. Their purpose is to come up with the way of sharing data from pre existing data sources in which there was no prior agreement during the design of data [19]. Mapping tables can also be used to map not only the schemas but also the relationship between the actual data stored in databases [2]. This is more appropriate if data is from the same domain. This is illustrated with sharing biological data in [2].

2.1.2 Schema Mediation architecture

The problem faced with when integrating data from different sources is that sources have different data models, query languages [23] and the heterogeneity of database schemas. The traditional approach is to have mediator and wrapper components. The mediator

defines a common data model in which all the local schemas are translated to. The advantage of the common data model is that it is much easier to write queries using one model.

The query translator is part of the mediator component. The query translator is responsible for writing queries which are ready to be passed to different data sources. A query is answerable by the mediator if it is answerable by at least one of the data sources participating in the connection [17]. The translator translates all queries using a common data manipulation language which is used together with the common data model.

The wrapper component is source specific. It is responsible for translating queries from the common data model back to the query language in which the wrapper belongs to. The mediator component can be extended to include a lot of subcomponents which optimize the performance of the architecture.

2.2 Defining the global schema

There are two main approaches of modeling the global schema; global as view (gave) and local as view (lave) [6]. The Global as view approach requires that the global schema be expressed as a view in local sources. The lave approach defines the global schema by keeping local schemas as views over the global schema; this makes it to define the global schema independently from the sources.

The disadvantage of the gav approach is that adding a new source is complicated because the new source can have an impact on the definition of some terms of the global schema as the results the global schema can be revised. The advantage of it is that query processing is easily achieved [6]. This is because gav keeps information on how to access the data stored in sources. The advantage of the las approach is that it is easy to add a new source; it only means adding more mappings to the global schema. The drawback of lav is that query processing is more difficult compared to the gav approach. An approach which combines the strengths of the two approaches is called BGLaV and is discussed in more details in [12].

The next section gives more details on traditional architecture of mediating and querying data from different sources.

2.2.1 Context Mediation

This section describes the mediator sub component which can be used to change data stored in different contexts. This architecture is discussed in more details in [1].
When organizations share data which was independently designed face the problem of ensuring semantic interoperability of the data. This is achieved by storing the information together with its context information. Context information can be seen as the metadata about its meaning. The proposed query language which suites this architecture is context SQL (C-SQL). C-SQL enables the attribute to be stored and queried together with its context called the meta attribute. The meta attribute has context information about the base attribute. An example to this is storing salaries together with their currency. In this case salary is the base attribute and currency is the meta attribute. The meta attribute defines the

context of the salary. It is possible for a base attribute to have more than one meta attributes. The salary can be in rands, dollars or yeans.

2.3 Query Translation

The aim of query translation in the peer to peer system is to use one query to query multiple databases. These databases are physically distributed and have heterogeneous schemas. The fact that schemas are heterogeneous relates the problem of query translation to the problem of data integration.

The problem of data integration is the problem of integrating data from different sources and querying them with one query. The problem of query rewriting using views is the problem of rewriting a given query using only the relations and attributes in the given set of views. Query rewriting using views is applicable to many areas of databases; data integration, query optimization and data warehousing [4, 5, 10]. In query optimization query rewriting is used to get an equivalent query with efficient query plans. In data integration it is applied in query translation [4, 10].

2.3.1 Query processing abilities

Data sources answer queries depending on the data stored in them and the kind of restrictions on the data. Query processing capability of a data source is the information which specifies the kind of queries a data source is able to answer. Yerneni et al [17] describes a possible architecture for keeping query processing capability records of data sources. This information can be stored as query templates, capability-description grammars or capability records. The mediator system keeps capability records as views.

The capability record specifies attributes and relations together with restrictions posed on them. Restrictions can even be on the kind of answers a data source returns. An example in which this is illustrated is with the amazon.com search engine. This search engine can be queried by using keywords author, title or subject but it never returns a subject as the response to the query. This means that a query to this data source can have the subject attribute in the where statement but not in the select statement.

Data sources submit their query processing capabilities in terms of templates. Queries to a data source are then submitted by filling a template which was submitted during integration. The query processing capability of the integration system can be defined based on the query processing capabilities of individual data sources in the system. The query processing capability of the integration system is displayed to the user so that the user will pose queries which are answerable by the system. The query processing capability of the integration system can also be given to other integration systems which use it as a source.

3 The Proposed Architecture

The architecture aims to capture as much metadata about the schema of the new peer as possible. This metadata is used to rewrite a query written in terms of one peer's schema to the query written in terms of the schema of the other peer.

The kind of queries a peer is able to answer is defined by the attributes and relations in the peer's schema and the nature of data stored by the peer. This architecture is based on defining query processing capability records discussed in [17].

Adding the new peer means revising the existing global schema to accommodate the relations and attributes coming with the schema of the new peer. Revising the global schema can either mean adding more attributes in one or more relations, adding new relations or adding more tuples in existing relations. New relations and attributes are added if they do not exist in the global schema already. New tuples are added if the schema is not revised; this happens if all attributes and relations of the new schema already exist in the global schema

We start by matching the new peer's schema with the global schema. We therefore need to capture the metadata about the schema of the new peer and define rules to relate it to the global schema.

The aim is to keep schemas of peers as a set of views in the global schema. This way of integrating schemas is called local as view. The advantage of using this approach is that adding a new peer only means adding more views in the global schema. The problem of query translation can then be related to the problem of query rewriting using views.

3.1 Metadata about the new peer

The set of views representing the schema of each peer is composed of relations and attributes, restrictions posed by the administrator, relationship between tables of the local schema, general knowledge about data and rules for converting data from the local context to the global context.

The required metadata about attributes includes their data types, scale or units (if applicable) and general knowledge about them.

It is important to keep metadata about data types of attributes because other queries impose restrictions on data types. An example is with the sum aggregate function of the standard SQL. This function takes only attributes of type number (float, double or integer) as a parameter. Calling this function with a clashing data type makes the query to be invalid.

General knowledge can be any known information about attributes. An example of general knowledge about the attribute is when it is known that the salary attribute of the relation "Employee" can not be less than R5000. This can be possible if a peer stores information about employees who get more than R5000.

This knowledge contributes to the query processing capability because a peer will not answer queries with conditions which do not comply with this knowledge. The query processor can then prune such conditions or decide not to send the query to the peer at all. This can reduce the number of connections which will not return any results.

Restrictions are imposed by the administrator who wants to share some of the information stored in a relation. An example scenario is sharing

information about full time employees (identified with employee_level > 3) and not about part time employees (employee level <3). These restrictions also form part of the conditions in the views which represent the local schema.

There are attributes which one needs to know about their scale, rate or units to make full meaning about them. Peers can store data in different scale and units. For example one peer can store salaries in rand/week while the other stores the in dollars/month. It is necessary to convert the data from units and scale of the peer's schema to the scale of the global schema. Converting data from the context of the local schema to the global context automatically relates it to the context of other peers.

3.2 Query Translation

A peer can pose a query which other peers will not be able to answer because they do not have enough attributes and relations. Aggregate and grouping functions require to be processed twice. First for getting responses from all peers and second to compute the final response based on the responses from peers. The rest of this section explores these further.

3.2.1 Missing Attributes

Missing attributes are attributes in the sender's query but not in any of the views which make up the schema of the receiver.

Database servers often return a null value if the required attribute in the select statement has no value. In data integration we try to query all databases as if we were querying one database. There are attributes in the sender's schema but not in the receiver's schema. The sender can pose a query whose rewriting makes missing attributes in queries sent to other peers. This means that peers will answer a query depending on whether they do have all or some of the significant attributes in the query. A significant attribute in the query changes the meaning of the query if it is missing.

We send a query to the peer even if some of the attributes in the select statement are missing. A query like this gives an approximate answer to the original query. A query is not sent to the peer if all attributes in the select statement are missing.

Missing attributes in the where statement can change the meaning of the query. We strive to get as many correct or approximate answers as possible but still get only tuples required by the query. This makes a need of trying to find out if a missing attribute or condition changes the meaning of a query. The definition of query containment helps in making this decision.

Query containment [4, 10]: a rewriting Q1 of the query Q2 is contained in Q2 if the set of tuples return by Q1 is a subset of tuples which would be returned by executing Q2 [10].

When doing a rewriting of the global query using the schema of the local schema we strive to get a contained rewriting. This makes sure that we do not send a query which can return answers which are not expected by the user. At the same time we want to get as many correct or approximate answers as possible. We therefore do not include missing attributes in the where statement

and check if the rewriting is contained in the original query. The query is sent to the peer if the rewriting is contained.

3.2.2 Aggregate and Grouping Functions

Aggregate functions take a set of values and produce a single value as the result. We explore the usage of min, max, count, sum and avg in SQL. The SQL standard requires that the parameter to the avg or sum functions be numbers. We only consider aggregate functions in the select clause which is where they commonly appear.

To compute an answer to the query with aggregate functions requires getting responses from all peers and processing them further to get the correct answer.

The summary of rules used to process the queries with aggregate functions are as follows.

- If the aggregate function is sum, min or max then send the rewritings to peers, keep the responses in the results table and query the results table to compute the sum, maximum or the minimum.
- If the function is count then send the rewritings to peers and put responses in the results table. The answer to be displayed to the user is calculated by computing the sum of the responses from peers.
- If the function is sum or avg then the data type of the parameter should be of type number. Examples of type number are float, double and integer. If a peer stores the parameter not as of type number then the translator does not send the query to it.
- The avg function requires getting the sum and the number of tuples with the aggregated attribute. This is done by sending the query to peers with the sum and count functions replacing the aggregate function. This is illustrated with an example.

Example 1: Suppose that all peers have the relation employee which has the salary attribute. If the source peer asks a query Q1 then the query to be sent to peers is Q2.

Q1: a query from the source peer is
select avg (salary) from employee;

Q2: query to be sent to each peer becomes :
select sum (salary) as sums, count (salary) as counts
from employee;

The average is then computed by putting responses from peers into a temporary results table and querying it using Q3.

Q3: Select sum (sums)/sum (counts)
From TemporaryResultsTable;

Queries with grouping functions also require special processing before displaying the answer to the user. The approach is similar to the aggregate functions. The queries sent to peers do not have the part with a grouping function. The responses from peers are put in the temporary results table. The results table is then queried with these grouping functions.

4 Results

Schema matching in a peer to peer network can be done by mapping all the schemas to the global schema and defining rules for converting data from the peer's context to the global context.

The kind of queries a peer is able to answer depends on the data types of attributes, the available attributes and relations, restrictions on shared data and general knowledge about data.

In a P2P network we can not always find the equivalent rewriting of the sender's query. This is because peers have different schemas. We therefore send queries which return some of the answers or approximate answers to other peers.

A rewriting of the original query may not be sent to the respective peers if it becomes invalid, there are clashing data types and the query has restrictions on data types, a peer does no have enough significant attributes to answer a given query or general knowledge about data proves that the peer will not be able to answer a query.

A rewriting of the peer can be sent to other peers if it is equivalent to the original query or it is a simplified query. A peer answers an equivalent query if it has all the attributes in the original query. A simplified query does not have some of the attributes in the where statement of the sender's query but will not return responses which do not comply to the conditions in the original query or it does not have some of the out put attributes.

It is therefore possible to predict if the peer will be able to answer the query before sending it to it. Predicting if the peer is able to answer the query can improve the performance of the system because a query is sent to the peer if there is no evidence that it will not be able to answer it.

Data types usually do not have an impact in determining the kind of queries a peer is able to answer. This problem was only encountered with parameters to the sum and average aggregate functions of SQL. This means that data can be stored in different formats but still be queried in the same way. The problem is that the responses will be in different formats. The user might not like to get answers in this format.

5 Conclusion

The problem of schema conflicts in the P2P network can be solved by transforming data from the local representation to the global reference. Peers answer different queries depending on their schema and the way in which data is represented. Like in all data integration systems, a peer to peer system also gets approximate answers to the user's query. Peers can answer the same query even if the data is stored in different formats. The differences in data types do not have too much influence in rewriting queries to be sent to peers. But differences in data types can return results in different formats.

Defining the kind of queries a peer is able to answer can improve the performance of the system because a query will be sent to the peer if there is no evidence that it will not be able to answer. This reduces avoidable connections to peers. Queries with aggregate and grouping functions need to be rewritten twice. The first step

gathers information required to answer the peer's query. The last step is to compute the answer to be displayed to the user using the results from peers.

6 References

1. Sciore E, Siegel M, et al. Using Semantic Values to facilitate interoperability among heterogeneous information systems. In Transactions on Database Systems, Vol.19, No.2, 1994.
2. Kemenstsietsidis A, Arenas M, Data Sharing Through Query Translation in Autonomous sources. Proceedings of the 30th VLDB Conference, 2004
3. Halevy A,Ives Z, Schema Mediation for Large-scale Semantic Data Sharing. In VLDB, pages 68-83, 2005
4. Pottinger R, Halevy, MinCon: A scalable algorithm for answering queries using views,In VLDB, pages 182-198,2001.
5. Gryz J, Query Rewriting Using Views in the Presence of Inclusion Dependencies. Information Systems , 1999.
6. Lenzerini M, Data Integration: a Theoretical Perspective. Proceedings of the twenty-first ACM SIGOM_SIGMOD_SIGACT_SIGART symposium on Principles of databases systems, 2002, ISBN:1-58113-507-6.
7. Levy A , Rajaraman A, Querying Heterogeneous information sources using source descriptions. In Proceedings of the 22nd VLDB Conference,1996.
8. R Lawrence, K Barker ,Using Unity to Semi-Automatically Integrate Relational Schema, Demonstration at ICDE 2002 – 18th International Conference on Data Engineering, 2002.
9. Tatarino I, Ives Z, et al, The Piazza Peer Data Management Project. Department of Computer Science and Engineering, University of Washington,Seattle,WA 98195.
10. Harlevy A , Answering queries using views. VLDB, pages 270-294,2001
11. Dayal U, Hwang H, View Definition and Generalization for Database Integration in a Multidatabase System. IEEE Trans. Software Engineering: 628-645,1984.
12. Xu L, Embly D, Combining the best of global-as-view and local-as-view for data integration.
13. R Lawrence, K Barker, Integrating Relational Database Schemas using a Standard Dictionary, Proceedings of the 2001 ACM symposium on Applied computing, 2001, ISBN:1-58113-287-5
14. Chung S, Mah P, Schema integration for multiple databases using the unified relational and object-oriented model.
15. Lam M, Whale J, et al. Contest Sensetive Program Analysis as Database Queries. In PODS, 2005
16. Cheung W, HSU C, The Model-Assisted Global Query System for Multiple Databases in Distributed Enterprises. October 1996

17. Yerneni R , Chen L , et al. , Computing Capabilities of Mediators. In SIGMOD, 1999.
18. Kolaitis,Schema mappings, Data exchange, and Metadata Management, 2005
19. Kang J, Naughton J , On Schema Matching with Opaque Column Names and Data Values. In ISGMOD,2003
20. Kementsietsidis A, Arenas M, et al., Mapping Data in Peer to Peer Systems: Semantic and Algorithmic Issues.
21. Dhamankar R, Lee Y, iMAP: Discovering Complex Semantic Matches between Database Schemas, In SIGMOD, pages 13-18, 2004.
22. Domenig R, Dittrich K, A query based approach for integrating heterogeneous data sources,In CIKM,2000
23. Fuxman A, Kolaiti P, et al., Peer Data Exchange. In PODS ,2005
24. Katsaros C, Niarhos L,et al. DAIMON: Data Integration for a Mobile Network, In MobiDE'05,2005.