Decision making on operational data: a remote approach to distributed data monitoring

Decision making on operational data: a remote approach to distributed data monitoring

V. Benson
Kingston University, UK

Abstract

Information gathering and assimilation is normally performed by data mining tools and Online analytic processing (OLAP) operating on historic data stored in a data warehouse. Data mining and OLAP queries are very complex, access a significant fraction of a database and require significant time and resources to be executed. Therefore, it has been impossible to draw the data analysis benefits in operational data environments. When it comes to analysis of operational (dynamic) data, running complex queries on frequently changing data is next to impossible. The complexity of active data integration increases dramatically in distributed applications which are very common in automated or e-commerce applications.

We suggest a remote data analysis approach to find hidden patterns and relationships in distributed operational data, which does not adversely affect routine transaction processing. Distributed data integration on frequently updated data has been performed by analysing SQL commands coming to the distributed databases and aggregating data centrally to produce a real-time view of fast changing data. This approach has been successfully evaluated on data sources for over 30 data sources for hotel properties. This paper presents the performance results of the method, and its comparative study of the state-of-the art data integration techniques. The remote approach to data integration and analysis has been built into a scalable data monitoring system. It demonstrates the ease of application and performance results of operational data integration.
Keywords: database applications, distributed data, online analysis processing, data mining, SQL.

1 Introduction

In today's highly competitive environment success of a business enterprise depends on the effective use of its data assets. Fast pace of the business world demands immediate response and precise decision making on vast amount of data. Complex data is handled by decision support systems, and lately a trend to link on-line analysis processing (OLAP) and data mining has emerged. The conventional view to integration and analysis of data is to use Online Analytic Processing (OLAP) [1]. The databases involved are usually large and fast response is not expected as the queries access large parts of the database and are very complex. OLAP databases are usually stored in OLAP servers often called data warehouses [3] which represent analysis –oriented structures that store sets of non-volatile integrated data. The major capability of a data warehouse lies in OLAP analysis tools, which explore and summarise data into multidimensional views, i.e. data cubes. Hierarchical structure of a data cube represents data in such a way that each dimension consists of a set of categorical descriptors. Typical OLAP aggregation consists in the process of consolidating data into a cumulative value and is well supported by summarisation by elementary operators, such as SUM, AVERAGE, MAX, MIN and COUNT [4].

In a generic example of OLAP, such as observing the sales levels in different cities over a certain time period, a decision query is a computation over measures. The query should use descriptors as criteria to identify the targets and perform computation on the quantitive measures [4]. OLAP is a powerful tool for structuring data for analysis and structuring data cubes. However, if it is necessary to integrate current frequently updated rather than historical information, data warehousing is no longer an option [5].

Consider a similar example of data observation but taking place in an online hotel reservation system. The hotel property information and room availability data exist on dozens of DBMS's. Each hotel chain runs their own reservation system, which complicates integrated analysis of fast changing data about room bookings. Applying warehousing to this problem creates an out-of date reservation availability data incoherent with real-time information [5]. Warehousing is not capable of supporting real-time or operational decisions concerning analysis of distribution of reservations in a certain location in order to provide re-routing options or avoid overbooking.

The only way to guarantee active data aggregation in this fast changing distributed environment is to run decision queries in real time. This, however, imposes a resource problem. Data integration in advance is not applicable here, whereas integration on demand, especially when monitoring dynamic data activity, imposes a tremendous burden on the DBMS's involved.

This paper proposes a data federation approach for multi-database systems with frequently updated information, which are essential in data monitoring and automated applications. Distributed operational data analysis focuses on avoiding the additional impact of federated transactions on performance of local databases. Global view of data in a multi-database system is constructed by direct access of data and thereon is maintained by centralised analysis of SQL

traffic from local nodes. In the beginning of the paper the classification of SQL commands relevant to data federation is discussed. Further, the hotel availability monitoring case study is demonstrated as an example of multi-database aggregation of active data. Performance of the approach is evaluated further in the paper, the application domain and future research objectives of the method are discussed in the conclusion.

2 Related work

Distributed data analysis and application of data mining algorithms coupled with OLAP techniques have attracted significant interest in the recent research. In [4] a number of approaches to OLAP and data-mining for decision support in two-dimensional datasets and multidimensional data are discussed. There has been a trend to develop extensions to SQL language supporting OLAP and data mining. The authors in [4] describe an extension to the decision support query language for data mining tasks, whereas OLAP operators are designated to perform prediction, association and sequencing at different abstraction levels of a data view. The paper presents the conclusion that Online Analytical Mining as the process of extracting knowledge combined with online analysis will be a natural addition to two- and multi-dimensional data sets analysis. Chaudri and Dayal [1] proposes to integrate data mining capabilities using SQL language. The author describes a data-mining system based on extended SQL for relational data. The model developed operates as a client-server middleware that performs a decision tree classifier over MS SQL Server 7.0. The SQL language is added a model that enables discovery of association rules.

Stonebracker [5] raises the necessity of data integration and analysis in e-commerce applications. He describes a number of data federation systems capable of constructing a composite view of disparate data sources. Such systems aggregate data on demand and allow the user to run SQL commands for retrieval and asynchronous updates of the composite view. The functionality of the systems described lack the aspect of advanced analytical capabilities that OLAP tools can offer in data summarisation. Analysis of fast changing data, such as stock quotes, price lists, flight and hotel reservations, is not handled well in current OLAP systems. However, it is sometimes vital to integrate active data in the decision making process. It was pointed out in [6] that current OLAP systems also lack the necessary flexibility by relying on physical integration of data, which can be a complex and time-consuming task.

3 Dynamic data aggregation

In order to obtain the real-time dynamics of a changing parameter, for example, a stock quote varying from two hours or two months ago, each time this parameter changes at the data source, the application needs to recalculate its new value. Frequent polling of the databases involved to obtain the newly changed value congests the network traffic of routine database transactions and takes a

significant toll on the system load. Processing the parameter updates away from the database system can relieve some of the system resources.

By treating database transactions as data events we can automate the process of updating the monitored parameter and perform dynamic data integration using a centralised remote application. Data events that carry out changes in the local database content are essentially update transactions, expressed in SQL that alter certain values stored in a database. Values of parameters that are requested for monitoring can be directly retrieved from the database records or summarised by integration of the results from several queries. The approach to observation of dynamic data integrated from distributed sources can be described as a process of the following steps:

- A distributed transaction is executed on the data systems involved. The value/s obtained becomes the starting point of the monitoring process.
- Without performing querying of the data systems, SQL instructions at each system is monitored by a thin client application at each node.
- The client application on the data system determines if each SQL instruction will have changed the monitored parameter.
- When the SQL update is relevant to the monitored parameter, the change is analysed by the central application and the new value of the dynamic parameter is updated.
- The dynamics of the monitored parameter/s is visualised to the user as a chart or a series of time-stamped values.

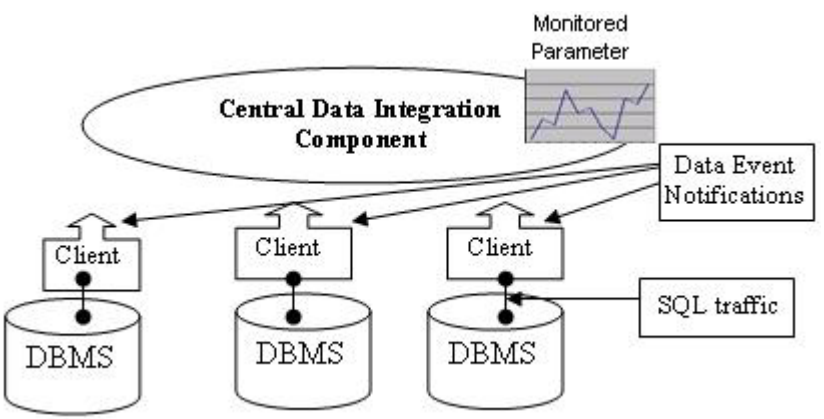

Figure 1: Dynamic data integration system.

4 System architecture

The distributed architecture of the dynamic data integration system offers a significant reduction of the network traffic accounted for frequent database polling. It replaces the transaction processing workload on each data source by a lightweight event messaging mechanism. The event notification is performed by a thin client application, which copies relevant SQL instructions and notifies the central data integration component, fig 1.

Dynamic data processing does not require accessing the data system and is performed remotely to the databases involved. This architecture provides for greater flexibility and scalability. It allows performance of data monitoring in a distributed environment from multiple independent data sources, alleviating interoperability constraints on the participating databases.

5 Data aggregation measures

Evaluation of the monitored parameter occurs when event alerts notify about the changes in database content. Among the multitude of transactions routinely executed on a data set only a certain number actually modify data. A number of simple commands were considered to determine their impact on the database content. Database commands in SQL form can be classified in the following categories with respect to what sort of changes they carry:

Changing Database records: INSERT, DELETE, UPDATE change only values of the attributes.
Modifying Database structure: CREATE, DROP, ALTER change the overall structure of a database and content.
Non Modifying: SELECT by itself does not change either a database schema or its content.

Event alerts, which are communicated to the dynamic data integration system, may be relevant or irrelevant to the monitored parameter. In order to determine if a local database transaction changes the value of the monitored parameter, the string copy of a command shall be parsed and syntactically analysed by the central application. Depending on the type of the command, its syntactic processing technique is different, but the principle of matching the target attributes of a command against the monitoring query constraints and target remains the same.

The algorithm for re-evaluation of the monitored parameter with respect to a relevant database transaction is application and database specific. Depending on the database structure and relation properties, updates can affect the numeric value of the monitored parameter in different ways.

6 Evaluation

Performance evaluation of the Dynamic Data Integration approach presented in this paper has been carried out by simulating data activity in an online

reservation network. Data sources involved property information for several hotels and their reservation records. Properties could manage their on-line data about hotels, which may include addition or removal of amenities, photos, and other information. A new property reservation was performed by insertion of a new record into the relevant booking table. Based on the new booking, information for the current occupancy of that hotel property was updated for the dates of the stay. Monitoring of occupancies of various properties on based on certain dates and booking profiles was carried out. Figure 2 illustrates the implementation of the decision query construction for observation of the occupancy measure based on a number of factors such as dates, location, features and summarised using OLAP principles and elementary operations(Sum, Average, Max, Min, Count) [4].

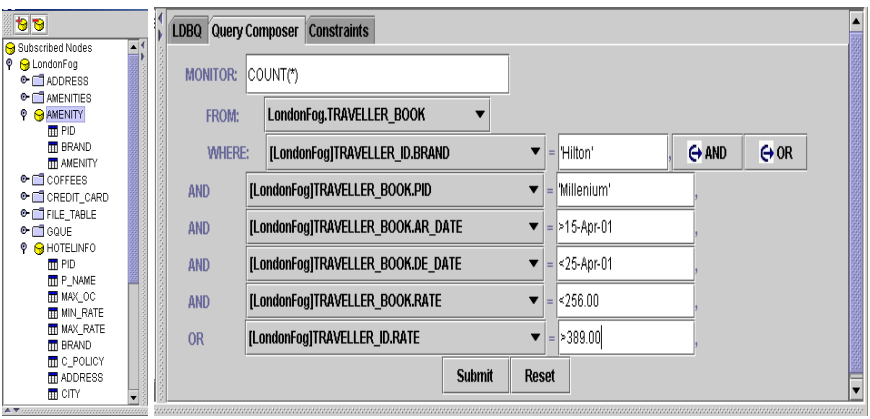

Figure 2: Implementation of the dynamic data observation mechanism. Decision Query construction and data sources view.

During the experiments, activity on data sources was implemented by reading prepared SQL commands from a data file and sending them for execution. The number of commands, types of transactions, and transaction's complexity was varied in several experimental scenarios.

The performance comparison was carried out by executing an SQL expression on the data source, i.e. through the direct database access versus logical calculation of the integration query. The maximum and minimum execution times are presented for the evaluation of an integration versus SQL query. Queries with an increasing number of constraints have been compared. The results of the execution time comparison are shown in fig. 3.

As shown in the experimental runs, database polling takes at least twice as long as the evaluation of the monitored parameter remotely. Event-based monitoring parameter observation offers a closer (i.e. faster) way of determining the latest updated value and performs evaluation process without increasing database SQL transaction load.

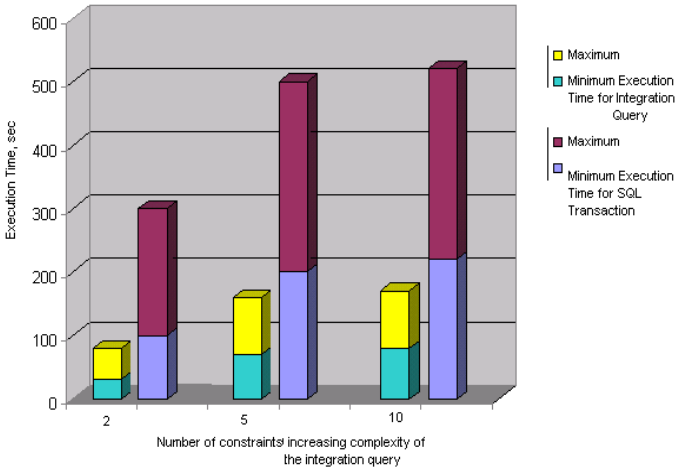

Figure 3: Performance comparison chart.

7 Conclusions and future work

Motivated by the need for federation of dynamically changing data in distributed data systems and the subsequent need for efficient query processing we have presented a remote approach to continuous integration of dynamic data from multiple sources. A centralised approach to integration of the fast changing data discussed in this article made it possible to avoid the adverse affects of extra transaction processing load due to frequent database access by complex federation queries. It was suggested here to eliminate database polling, but to use the syntax of incoming SQL instructions submitted to local data systems to draw conclusions on how data is being updated. Calculated once the parameters under observation of integrated queries are updated based on the analysis of SQL commands. Analysis of dynamic data is performed centrally, this offers greater flexibility and scalability of the system compared to physical federation [6]. Successful implementation of the method was carried in practice through operational monitoring of the distributed data for occupancy at multiple hotel properties. The experimental data shows that the time needed for evaluation of the integrated query remotely is more effective then direct access of federated database records. Future work in this area will be directed towards optimisation of the matching mechanism of the integration query and expanding the application domain of the remote approach to dynamic data integration and investigation of its compatibility with current OLAP systems. Failure recovery enhancement will also be considered.

References

[1] Chaudri, S. & Dayal, U. An overview of data warehousing and OLAP technology. *ACM SIGMOD Record* 26(1):65-74, 1997.

[2] Thomsen E., *OLAP Solutions: Building Multidimensional Information Systems*, Wiley, 1997.
[3] Lewis P., Bernstein A., Kifer M., Databases and Transaction Processing: An Application-Oriented Approach, Addison Wesley, New York, 2002.
[4] Messaoud R.B., Bussaid O., Rabaseda S., *Proc. of DOLAP'04*, Washington, DC, USA 2004.
[5] Stonebracker M., Too Much Middleware. *ACM SIGMOD Record* Vol. 31, No.1, March 2002.
[6] Pederson D, Riis K., Pedersen T., Query optimization for OLAP-XML Federations, Proc. of the 5th ACM international workshop on Data Warehousing and OLAP, McLean, Virginia, USA , Pages: 57 – 64, 2002.