International Journal of Modern Engineering Research (IJMER)
www.ijmer.com Vol. 3, Issue. 5, Sep - Oct. 2013 pp-3196-3199 ISSN: 2249-6645

A Novel Approach To Answer Continuous Aggregation Queries Using Data Aggregations

K. Suresh[1], Sk. Karimulla [2], B. Rasagna[3]
1, M.TECH Scholar, QCET, Nellore
2, Assistant Professor, QCET, Nellore
3, Assistant Professor, MLRITM, Hyderabad

ABSTRACT: *Decision making can be regarded as the reflective development consequential in the selection of a course of action among several alternative scenarios. Continuous and uninterrupted aggregation queries are used to monitor the changes in data with time varying for online decision making. Usually a user requirements to obtain the value of some aggregation function larger than distributed data items, for example, to be familiar with value of portfolio for a client; or the average of temperatures sensed by a set of sensors. In these queries a client identifies a coherency requirement as measurement of the query. In this we suggested a low-cost, scalable technique to answer continuous aggregation queries using a network of aggregators of dynamic data items. Such a network of data aggregators, each and every data aggregator gives out a set of data items at specific coherencies. Just as a variety of fragments of a dynamic web pages are served by one or more nodes of a content allocation network, our proposed method involves decomposing a client query into sub-queries and carry out sub-queries on sensibly chosen data aggregators with their individual sub query incoherency boundaries. We present that a technique for getting the optimal set of sub queries with their incoherency boundaries which gratifies client query's coherency obligation with least number of refresh messages driven from aggregators to the client. For approximation the number of refresh messages, we put together a query cost model which can be used to calculate approximately the number of messages required to gratify the client specified incoherency bound.*

Index Terms— *Algorithms, Continuous Queries, Data Dissemination, Coherency, Performance, Distributed Query Processing*

I. INTRODUCTION

Applications such as sensors-based monitoring, auctions, personal portfolio valuations for financial decisions, route planning depends on traffic information, etc., make wide use of dynamic data. For such type of applications, data from one or more independent data sources may be aggregated to determine if some action is warranted. Given the growing number of such type of applications that make use of highly dynamic data, there is imperative interest in systems that can efficiently deliver the relevant updates repeatedly. In these continuous query applications, users are probably to bear some incorrectness in the results. That is, the exact data values at the equivalent data sources need not be reported as long as the query results gratify user specified accuracy requirements.

Data incoherency: Data accuracy can be specified in terms of incoherency of a data item, defined as the complete dissimilarity in value of the data item at the data source and the value known to a client of the data. Let $v_i(t)$ indicate the value of the i^{th} data item at the data source at time t; and let the value the data item known to the client be $u_i(t)$. Then, the data incoherency at the client is given by $|v_i(t)- u_i(t)|$. For a data item which needs to be refreshed at an incoherency bound C a data refresh message is sent to the client as soon as data incoherency exceeds C, i.e., $|v_i(t)- u_i(t)| > C$.

Network of data aggregators (DA): Data refresh from data sources to consumers can be done using push or pull based mechanisms. In a push based system data sources send bring up to date messages to clients on their own while in a pull based system data sources send messages to the customer only when the client makes a request. In this assume the push based system for data transfer between data sources and clients. For scalable management of push based data distribution, network of data aggregators are proposed.

In this assume that each data aggregator maintains its configured incoherency bounds for different data items. From a data distribution capability point of vision, each and every data aggregator is characterized by a set of (d_i, c_i) pairs, where d_i is a data item which the DA can disseminate at an incoherency bound c_i. The configured incoherency bound of a data item at a data aggregator can be maintained using any of following methods:
1) The data source refreshes the data value of the DA whenever DA's incoherency bound is about to get violated. This method has scalability problems. 2) Data aggregator with tighter incoherency bound help the DA to maintain its incoherency bound in a scalable manner

1.1. Aggregate Queries and Their Execution

While executing continuous multi data aggregation queries, using a network of data aggregators, with the objective of minimizing the number of restores from data aggregators to the client. First, we give two motivating scenarios where there are various options for executing a multi data aggregation query and one must select a particular option to minimize the number of messages.

1.2. Problem statement & contributions

Our simulation studies show that for continuous aggregation queries:

- Our method of dividing query into sub queries and executing them at individual DAs requires less than one third of the number of refreshes required in the existing schemes.
- For reducing the number of refreshes more dynamic data items should be part of a sub query involving larger number of data items.

Our method of executing queries over a network of data aggregators is practical since it can be implemented using a mechanism similar to URL-rewriting in content distribution networks (CDNs). Just like in a content distribution networks, the client sends its query to the central site. For getting appropriate aggregators (edge nodes) to answer the client query (webpage), the central site has to first determine which data aggregators have the data items required for the client query. If the client query cannot be respond by a single data aggregator, the query is divided into sub queries (fragments) and each sub query is assigned to a single data aggregator. In case of a content distribution networks webpage's division into fragments is a page design issue, whereas, for permanent aggregation queries, the issue has to be handled on per query basis by considering data dissemination capabilities of data aggregators.

This strategy ignores the fact that the client is interested only in the aggregated value of the data items and various aggregators can disseminate more than one data item. Second, if a single Data Aggregator (DA) can disseminate all three data items required to answer the client query, the Data Aggregator can construct a compound data item corresponding to the client query and disseminate the result to the client so that the query incoherency bound is not violated. It is obvious that if we obtain the query result from a single Data Aggregator, the number of refreshes will be (as data item updates may cancel not in each other, thereby maintaining the query results within the incoherency bound). Further, even if an aggregator can refresh all the data items, it may not be capable to satisfy the query coherency requirements. In such cases the query has to be implemented with data from multiple aggregators.

II. DATA DISSEMINATION COST MODEL

To estimate the number of refreshes required to disseminate a data item while maintaining a certain incoherency bound. There are two most important factors affecting the number of messages that are needed to maintain the coherency requirement:

1) The coherency requirement itself
2) Dynamics of the data.

1.3. Incoherency Bound Model

Consider a data item which needs to be distributed at an incoherency bound C that is new value of the data item will be pushed if the value deviates by more than C from the last pushed value Thus, the number of dissemination messages will be proportional to the probability of | v(t)- u(t) | greater than C for data value v(t) at the source/aggregator and u(t) at the client, at time t. A data item can be replicated as a discrete time random process where each step is correlated with its previous step. In a push-based distribution, a data source can follow one of the following schemes:

1. Data source pushes the data value whenever it differs from the last pushed charge by an amount more than C.
2. Client estimates data value based on server specified parameters. The source drives the new data value whenever it differs from the (client) estimated value by an amount more than C.

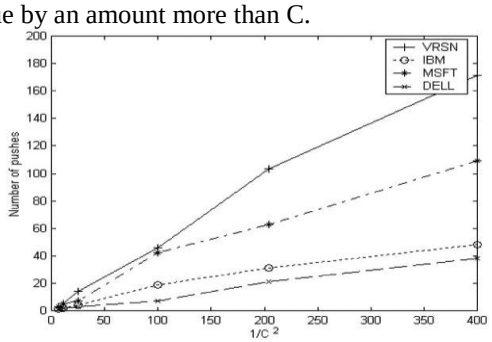


Fig.1: Number of pushes versus incoherency bounds

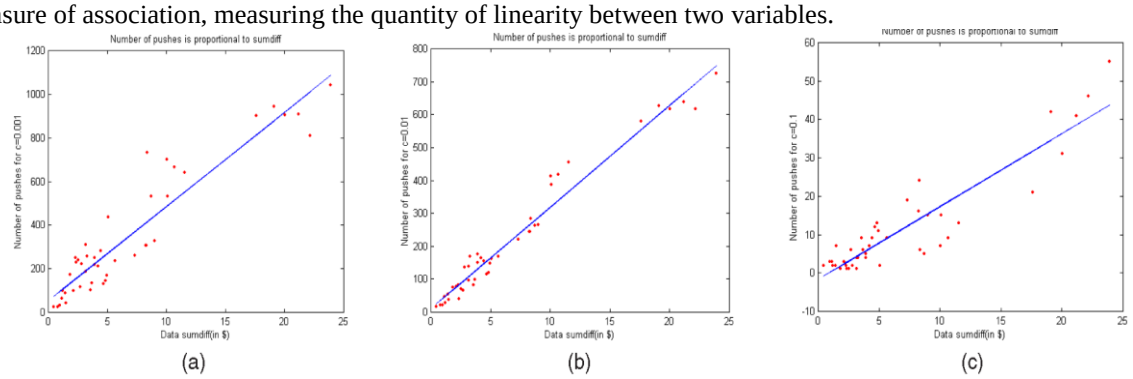
In both these cases, value at the source can be modeled as a random process with average as the value known at the client. In case 2, the server and the client estimate the data value as the mean of the modeled random process, whereas in case 1 difference from the last pushed value can be modeled as zero mean process.

2.2. Data dynamics Model

Two possible options to model data dynamics, as a first option, the data dynamics can be quantified based on standard deviation of the data item values. Suppose both data items are disseminated with an incoherency bound of 3. It can be seen that the number of messages required for maintaining the incoherency bound will be 7 and 1 for data items d1 and d2, respectively, whereas both data items have the same standard deviation. Thus, we need a measure which captures data changes along with its temporal properties. This motivates us to examine the second measure. As a second option we considered Fast Fourier Trans- form (FFT) which is used in the digital signal processing domain to characterize a digital

signal. FFT captures number of changes in data value, amount of changes, and their timings. Thus, FFT can be used to model data dynamics but it has a problem. To estimate the number of refreshes required to disseminate a data item we need a function over FFT coefficients which can return a scalar value. The number of FFT coefficients can be as high as the number of changes in the data value. Among FFT coefficients, 0th order coefficient identifies average value of the data item, whereas higher order coefficients represent transient changes in the value of data item.

Validating the hypothesis: We did simulations with different stocks being disseminated with incoherency bound values of $0:001, 0.01, and 0.1. This variety is 0.1 to 10 times the regular standard deviation of the share price values. Number of invigorate messages is plotted with data sum diff (in $) the linear relationship appears to exist for all incoherency bound values. To enumerate the measure of linearity we used Pearson product moment correlation coefficient (PPMCC), a widely used measure of association, measuring the quantity of linearity between two variables.



(a) (b) (c)

Fig.2: Number of pushes versus data sumdiff (a) C ¼ 0:001, (b) C ¼ 0:01, and (c) C ¼ 0:1.

It is calculated by summing up the products of the deviations of the data item values from their mean. PPMCC varies between 1 and 1 with higher (absolute) values signifying that data points can be considered linear with more confidence. For three principles of incoherency bounds 0.001, 0.01, and 0.1; PPMCC values be 0.94, 0.96, and 0.90, respectively and average deviation from linearity was in the range of 5 percent for low values of C and 10 percent for high values of C. Thus, we can conclude that, for lower values of the incoherency bounds, linear relation- ship between data sum diff and the number of refresh messages can be assumed with more confidence.

2.3 Combining Data Dissemination Models

Number of refresh messages is proportional to data sumdiff R_s and inversely proportional to square of the incoherency bound (C^2). Further, we can see that we need not disseminate any message when either data value is not changing ($R_s = 0$) or incoherency bound is unlimited ($1/C^2 = 0$). Thus, for a given data item S, disseminated with an incoherency bound C, the data distribution cost is proportional to R_s/C^2. In the next section, we use this data dissemination cost model for developing cost model for additive aggregation queries.

III. QUERY PLANNING

A query plan is an ordered set of steps used to access or modify information in a SQL relational database management system. This is a detailed case of the relational model concept of access plans. Since Oracle is declarative, there are typically a large number of alternative ways to execute a specified query, with extensively varying performance. When a query is suggested to the database, the query optimizer appraises some of the unlike, accurate possible plans for executing the query and returns what it considers the best alternative. Thus to get a query plan we need to perform following tasks.
1. Determining sub queries: For the client query get sub queries for each data aggregator.
2. Dividing incoherency bound: Divide the query incoherency bound among sub queries to get the value of sub query.

Optimization objective:

Number of refresh messages is minimized. For a sub query the estimated number of refresh messages is given by the ratio of sum diff of the sub query and the incoherency bound assigned to it and the proportionality factor k. Thus the total no of refresh messages is estimated as the summation of the ratio of the sub query of a given query and in coherency bound associated to it.

Constraint1: q_k is executable at ak: Each DA has the data items required to execute the sub query allocated to it, i.e., for each data item d_q k_i required for the sub query

Constraint2: Query incoherency bound is satisfied: Query incoherency should be less than or equal to the query incoherency bound. Meant for preservative aggregation queries, value of the consumer query is the sum of sub query values. As different sub queries are distributed by different data aggregators, we need to ensure that sum of sub query incoherencies is less than or equal to the query incoherency bound.

Constraint3: Sub query incoherency bound is satisfied: Data incoherency bounds at ak should be such that the sub query incoherency bound can be satisfied at that data aggregator. The tightest incoherency bound, which the data aggregator ak can satisfy for the given sub query q_k. For satisfying this constraint we ensure the following is the outline of our approach for solving this constraint Optimization problem we prove that determining sub queries at the same time as minimizing Zq, as specified by is NPhard.

IV. PERFORMANCE EVOLUTION

4.1 Comparison of Algorithms

We consider various other query plan options. Each query can be executed by distributing individual data items or by getting sub query values from network of data aggregators. Set of sub queries can be selected using sumdiff-based approaches or any other random selection. Sub query (or data) incoherency bound can either be pre decided or optimally allocated. Various arrangements of these proportions are covered in the following algorithms:

1. No sub query, equal incoherency bound (naive): In this the client query is performed with each data item being disseminated to the client independent of other data items in the query. Incoherency bound is separated evenly among the data items. This algorithm proceeded as a baseline algorithm.
2. No sub query, optimal incoherency bound (optc): In this algorithm as well data items are distributed independently but incoherency bound is separated among data items so that total number of invigorate can be minimized.
3. Random sub query selection (random): In this case, sub queries are obtained by randomly selecting a network of data aggregators in the each iteration of the greedy algorithm. This algorithm is designed to see how the random selection works in comparison to the sumdiff-based algorithms.
4. Subquery selection while minimizing sumdiff (mincost)
5. Subquery selection while maximizing gain (maxgain)

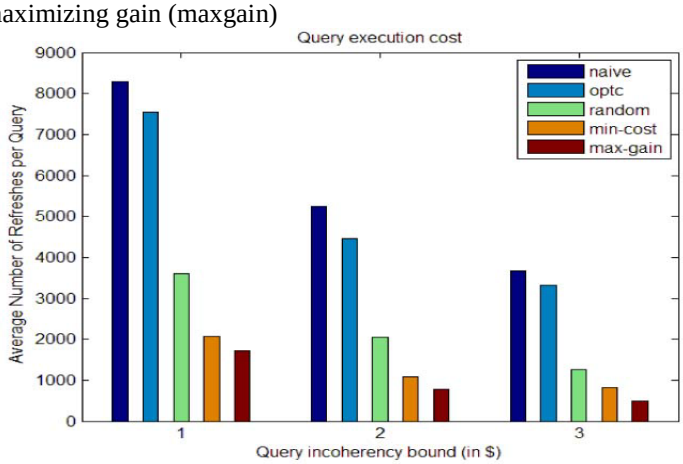


Fig. 5: Performance evaluation of algorithms.

IV. CONCLUSION

Here current a cost-based approach to minimize the number of refreshes required to execute an incoherency bounded uninterrupted query. We assume the continuation of a network of data aggregators, where each Data Aggregator is accomplished of distributing a set of data items at their pre specified incoherency bounds. We developed an important measure for data dynamics in the form of sum diff which is a more appropriate measure compared to the widely used standard deviation based measures. For optimal query implementation we divide the query into sub queries and evaluate each sub query at a thoughtfully chosen data aggregator.

REFERENCES

[1] A. Davis, J. Parikh, and W. Weihl, "Edge Computing: Extending Enterprise Applications to the Edge of the Internet," Proc. 13th Int'l World Wide Web Conf. Alternate Track Papers & Posters (WWW), 2004.
[2] D. VanderMeer, A. Datta, K. Dutta, H. Thomas, and K. Ramamritham, "Proxy-Based Acceleration of Dynamically Generated Content on the World Wide Web," ACM Trans. Database Systems, vol. 29, pp. 403-443, June 2004.
[3] J. Dilley, B. Maggs, J. Parikh, H. Prokop, R. Sitaraman, and B. Weihl, "Globally Distributed Content Delivery," IEEE Internet Computing, vol. 6, no. 5, pp. 50-58, Sept. 2002.
[4] S. Rangarajan, S. Mukerjee, and P. Rodriguez, "User Specific Request Redirection in a Content Delivery Network," Proc. Eighth Int'l Workshop Web Content Caching and Distribution (IWCW), 2003.
[5] S. Shah, K. Ramamritham, and P. Shenoy, "Maintaining Coherency of Dynamic Data in Cooperating Repositories," Proc. 28th Int'l Conf. Very Large Data Bases (VLDB), 2002.
[6] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein, Introduction to Algorithms. MIT Press and McGraw-Hill 2001.
[7] Y. Zhou, B. Chin Ooi, and K.-L. Tan, "Disseminating Streaming Data in a Dynamic Environment: An Adaptive and Cost Based Approach," The Int'l J. Very Large Data Bases, vol. 17, pp. 1465-1483, 2008.
[8] "Query Cost Model Validation for Sensor Data,"www.cse.iitb.ac. in/~grajeev/sumdiff/ RaviVijay_BTP06.pdf, 2011.
[9] R. Gupta, A. Puri, and K. Ramamritham, "Executing Incoherency Bounded Continuous Queries at Web Data Aggregators," Proc. 14th Int'l Conf. World Wide Web (WWW), 2005.
[10] A. Populis, Probability, Random Variable and Stochastic Process. Mc. Graw-Hill, 1991.
[11] C. Olston, J. Jiang, and J. Widom, "Adaptive Filter for Continuous Queries over Distributed Data Streams," Proc. ACM SIGMOD Int'l Conf. Management of Data, 2003.
[12] S. Shah, K. Ramamritham, and C. Ravishankar, "Client Assignment in Content Dissemination Networks for Dynamic Data," Proc. 31st Int'l Conf. Very Large Data Bases (VLDB), 2005.
[13] NEFSC Scientific Computer System, http://sole.wh.whoi.edu/ ~jmanning//cruise/serve1.cgi, 2011.