Network Data Aggregators for Continuous Aggregation Queries by Query Planning

M. PRIYANKA*¹, G. SRIDHAR*².

PG Scholar, Dept of CSE, Kakatiya Institute of Technology and Science, Warangal.
Assistant Prof. Dept of CSE, Kakatiya Institute of Technology and Science, Warangal.

ABSTRACT: The Continuous aggregation queries are used to monitor the data change with time varying for online decision making. The making of Decision can be regarded as the reflective development consequential in the selection of a course of action among different scenarios. Uninterrupted Continuous queries and aggregation are used to monitor the any changes in data within a time varying for online. This Technique involves disseminating query into sub queries and that sub queries are executed on the chosen data aggregators. Generally user requirements to obtain the value of aggregation function larger than data items for example the average of temperatures sensed by a set of sensors. In these queries the client identified a coherency requirement as measurement of the query. Such a network of data aggregators gives out a set of data items at particular coherencies. Just as a variety of fragments of a dynamic web pages are served by one or many nodes of a content allocation network our proposed method involves decomposing a client query into sub queries and carry out sub queries on sensibly taken data aggregators with their individual 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 with least number of refresh messages driven from aggregators to the client. For approximation the number of messages, we place together a query cost model which can be used to calculate approximately the total number of messages required to gratify the client specified incoherency bound.

Index Terms— Algorithms, Continuous Queries, Coherency, Data Dissemination, Distributed Query, Performance, Processing.
I. INTRODUCTION:
Applications like auctions, Personal portfolio valuations for financial decisions, weather Prediction website, espn, website makes wide use of dynamic data. Make the extensive use of active data. For such appliances data from one or more independent data sources may be aggregated to determine the data with less number of refreshes. Given the increasing number of such data Applications that make use of highly dynamic there is important concern in systems that can efficiently deliver the relevant updates. Given the growing number of such type of applications that make use of highly dynamic data. In these continuous query applications users are probably to bear some incorrectness in results. The exact data values at the equivalent data sources need not be reported as long as the query results user specified accuracy requirements. Consider a user who wants to road a portfolio of stocks in different accounts. Stock data values from possibly unlike sources are required to be aggregated to satisfy user’s requirement. These aggregation queries are long running queries as data are continuously changing and the user is interested in notifications when confident conditions hold. Thus responses to these queries are revived continuously. In these permanent query applications, Uses tolerate some inaccuracy in the results.

II. RELATED WORKS:
Value of a continuous weighted additive aggregation query at time t can be calculated as; Where Vq is the value of a client query q involving nq data items with the weight of the I th data item being Such a query encompasses SQL aggregation operators SUM and AVG besides general weighted aggregation queries such as portfolio queries, connecting aggregation of stock prices and weighted with number of distributes of stocks in the portfolio. Suppose the result for the query given by (1) needs to be continuously provided to a user at the query incoherency bound Cq. Then, the dissemination network has to ensure that whenever data values at sources change such that query incoherency bound is desecrated, the updated value should be refreshed the client. The network of aggregators can make sure that the i th data item has incoherency bound Cq then the condition ensures that the query incoherency bound Cq is satisfied X The client specified query incoherency bound needs to be translated into incoherency bounds for individual data items such that is satisfied. It should be noted that is a sufficient condition for satisfying the query incoherency bound but
not essential. This way of converting the query incoherency bound into the sub query incoherency bounds is required if data are transferred between various nodes using only push-based mechanism. We need a method for

i) Optimally dividing a client query into sub queries

ii) Transfer incoherency bounds to them

iii) The derived sub queries can be executed at chosen DAs

iv) Total query execution cost, in terms of figure of refreshes to the client is minimizing.

III. 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., $|v_i(t) - u_i(t)| > C$.

IV. Network of data aggregators:

Refreshing the Data 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 client and data. For the scalable management of push based data distribution network of data aggregators are proposed.

V. 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. We give two motivating scenarios where there are various options for executing a multi data aggregation query and must select a particular option to minimize the number of messages.

VI. 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 primary factors disturbing the number of messages that are needed to maintain the coherency constraint:

i) The coherency requirement itself
ii) Dynamics of the data

A. 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 previous step. In 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 specified server parameters. The source drives the new data value whenever it differs from the estimated value by an amount more than C.

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.

B. Data dynamics Model:
Two possible options to model data dynamics, first option the data dynamics can be quantified based on standard deviation of the data item values. Suppose both the data items are disseminated with an incoherency bound of 3. It can be seen 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 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 amount of changes, data value, 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 dataset items. we need a function over FFT coefficients which can return a scalar value. The number of FFT co-efficients can be as high as the number of changes in the data values. Among FFT coefficients 0th order coefficient identifies average value of the data items, where as higher order coefficients represent transient changes in the value of data item.

VII. Validating the hypothesis:
We did simulations with different stocks being disseminated with incoherency bound values Of $0.001, 0.01, and 0.1. This range is 0.1 to 10 Times the average standard deviation of the share price values. Number of refresh messages is plotted with data sum diff (in $) the linear relationship appears to exist for all incoherency bound values. To quantify the measure of linearity we used Pearson product moment correlation coefficient (PPMCC), a widely used measure of association, measuring the degree of linearity between two variables. 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 values signifying that data points can be considered linear with more confidence. For three values of incoherency bounds 0.001, 0.01, and 0.1; PPMCC values were 0.94, 0.96, and 0.90, respectively, i.e., 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.

VIII. QUERY PLANNING:
A query plan is an ordered set of steps used to access or modify information in a SQL relational database system. This is a specific case of the relational model concept of accessing plans. Since Oracle is declarative there are typically a large number of alternative ways to execute a given query with widely varying performance. When a query is submitted to the database management system the query optimizer evaluates some of the different, correct possible plans for executing the query and returns what it considers the best alternative to get a query plan we need to perform following tasks.
1. Determining sub queries: 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 values of sub query’s.

**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 of the difference sub query and the incoherency bound assigned to 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:** qk is the executable at ak. Each DA has the data items required to execute the sub query allocated to it for each data item dq ki required for the sub query

**Constraint2:** Query incoherency bound is satisfied: Query incoherency should be less than or equal to the query incoherency bound. 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 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 should be such that the sub query incoherency bound can be satisfied at that data aggregators. The tightest incoherency bound the data aggregator ak can satisfy for the given sub query qk for satisfying the 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.

**XI. PERFORMANCE EVALUATION:**
For performance evaluation we simulated a Network of data aggregators of 200 stock data items. Over 100 aggregator nodes such that each aggregator Can disseminate combinations of 25 to 50 data Items. Data items were assigned to different aggregators using zipf distribution skew assuming that some popular data items will be disseminated by more DAs. Data incoherency bounds, for various aggregator data items, were chosen uniformly between $0:005 and 0.02. We created 500 portfolio queries such that each query has 10 to 25 randomly, selected data items with weights varying between 2 and 10.
These queries were executed with incoherency bounds between 1.0 and 3.0. Although here we present results for stock traces similar results were obtained for sensor traces as well. In the first set of experiments, we kept data incoherency bounds at the data aggregators very low so that query can be ensured while keeping default value of as 0.

**FIG: Performance on MAX queries**

To execute an incoherency bounded uninterrupted query to assume the continuation of a network of data aggregators. Where each Data Aggregators are accomplished of distributing a set of data items at specified incoherency bounds. We developed an important measure for data dynamics in the form of sum difference 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 chosen data aggregator.

**REFERENCES:**


ABOUT THE AUTHORS:

MANCHALA PRIYANKA is currently pursuing her M.Tech Computer Science & Engineering Department in Kakatiya Institute of Technology and Science, Warangal. She received her B.Tech in Information Technology from SR Engineering College, Warangal. Her area of interests includes Software Engineering, Data mining and Warehousing.

GUJJETI SRIDHAR is currently working as a Assistant Professor
Department of Computer Science & Engineering in Kakatiya Institute of Technology and Science, Warangal. He has 8 years of experience. His research interest includes Web Mining.