INTERNATIONAL JOURNAL OF MERGING TECHNOLOGY AND ADVANCED RESEARCH IN COMPUTING

ISSN: 2320-1363

INCREASING QUERY SATISFIABILITY AND PERFORMANCE USING PROTOTYPE QUERY PLANNING TECHNIQUES

S.Mamatha*1, K.Srujana*2

M.Tech (CSE) Student, Department of CSE, PEC, Kandukur, Dist: Prakasam, AP, India

Associate professor, Department of CSE, PEC, Kandukur, Dist: Prakasam, AP, India

ABSTRACT

As per my paper Continuous Queries are used for Monitoring the time variant data & and we can able to take the Online Decision Making. Example we consider the Portfolio to known the values for the portfolio to the client. .Client sends the query request to the data aggregators then the aggregators accepts the request and can able to send the response for multiple clients at a time .Then the aggregators to client takes less time and it is a very low cost technique also .In this technique involves the disseminating the query into sub query and sub queries are executed on chosen data aggregators. We provide a technique for the optimal solution for getting the coherency with there incoherency bounds which satisfies the client query coherency requirement with least number of refresh messages sent from aggregator to client. Based on the cost based query planning the queries are executed with less number of messages.

Keywords: Algorithms, Continuous Queries, Data Dissemination, Distributed query processing, Coherency, Performance.

I. INTRODUCTION

Application like personal portfolio to evaluation for financial decisions, Weather prediction websites, espn cricket info website, Actuations for selling and buying of goods at the offering time, and stock exchange Websites for the evaluation of the updates. For such type of applications the data is dynamically accessed and updated so that users can be able to get the optimal results instantly. Increasing of such applications using the easy way of the dynamic data very easily and instantly. Stock data from possibly different sources may be required to be aggregated to satisfy the user's requirement. These queries are long running as data is continuously changing and the user is interested in the notifications when certain conditions hold. For such appliances, data from one or more independent data sources may be aggregated to determine the data with less no of refreshes. Such type of applications that make use of highly dynamic data there is an important concern in systems that can

ISSN: 2320-1363

efficiently deliver the relevant updates manually. For Instance, assume the worldwide virtual organization with the users interested in biological data, as well as the real association on the activity network. In both cases users who are not continuously interested in performing data analysis can make a part of the resources available for supporting analysis tasks needed by the others, if there own ability performing local asks is conserved. In order to make participants really independent. They should be forced no constraint on storage and computational resources to be shared on the reliability of there network connection. For answering the multi data aggregation query in circumstances, there are three alternatives for the client to get the query outcome. Firstly, the client may get the data items d1, d2,and d3 separately. The query incoherency bound can be divided among the data items in various ways ensuring that incoherency bound. Here getting the data items independently is a costly option. In this Strategy the client is interested only the aggregated value of the data item. Second, if a single Data Aggregator (DA) can disseminate all three data items required to answer the client query, the DA 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 noticeable that if we get the query result from a single DA, the number of refreshes will be (as data item updates may cancel out 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 able to convince the query coherency necessities. In such cases the query has to be executed with data from multiple aggregators.

II. DESIGN

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:

1) The coherency requirement itself and

2) Dynamics of the data

A. Incoherency Bound Model

The number of dissemination messages will be proportional to the probability of 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 modeled as a discrete time random process where each step is correlated with its earlier step. In a push-based distribution, a data source can follow the following schemes: a. Data source move forward the data value whenever it differs from the last pushed value by an amount more than C. b. Client estimates data value based on server specified parameters. The source pushes the new data value whenever it differs from the (client). 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 client and the server estimate the data value as the mean of the modeled random process, whereas in case 1 deviation from the last pushed value can be modeled as zero mean process. Using Chebyshev's inequality Thus, we hypothesize that the number of data refresh messages is inversely proportional to the square of the incoherency bound. A similar result was reported in where data dynamics were modeled as random walks. Validating the analytical model to corroborate the above analytical result we simulated data sources by reading values from the sensor and stock data traces, at periodic instances. For these experiments, every data rate at the first indicate is sent to the client. Data sources maintain last sent value for each client. The sources read new value from the trace and send the value to its clients if and only if not sending it will violate the client's incoherency bound C. For each data item the incoherency bound was varied and refresh messages, to ensure that incoherency bound, were counted. Fig. 1 shows the curves for the number of push messages, for four representative share price data items, as their corresponding incoherency bounds, and hence 1 is C2, are varied. Besides validating the analytical model, these results give one important insight into the distribution mechanism. As the incoherency bound reduces, the number of messages increases as per analytical model, but there is a saturation effect for very low values of the incoherency bound (i.e., right part of the curve). This is due to the fact that the data items have limited number of discrete changes in the value. For example, if the sensitivity of a temperature sensor is one degree then number of dissemination messages will not increase even if incoherency bound is decreased below one degree.

B. 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. We hypothesize that the cost of data dissemination for a data item can be approximated by a function of the first FFT coefficient. Specifically, the cost of data dissemination for a data item will be proportional to data sum diff defined as where si and si1 are the sampled values of a data item S at ith time instances (i.e., consecutive ticks). In practice, sum diff value for a data item can be calculated at the data source by taking running average of difference between data values for consecutive ticks. For our experiments, we calculated the sum diff values using exponential window moving average with each window having 100 samples and giving 30 percent weight to the most recent window.

III. IMPLEMENTATION STEPS

Network of data aggregators:

Data refreshes from data sources to clients can be done using push and pull based mechanisms. In push based mechanism data sources sends update messages to the clients on their own .where as in pull based mechanism data sources send messages to t he client only when the client makes a request .We assumed the push based mechanism for data transfer between data sources and clients.

Aggregate Queries and Their Execution:

The method that we present fort he executing continuous multidata aggregation queries, using a network of data aggregators, with the objective of minimizing the number of refresh from data aggregators to the client.

Scenario: Consider a client query Q=50d1+200d2+150d3, where d1, d2, d3 are different stocks in a portfolio, with a required incoherency bound of $80. Here we want to execute the query over the data aggregators to minimize the number of refresh messages. In a network of data aggregators managing data items d1-d4, various aggregators can be characterized as A1:{(d1,0.5),(d2,0.2)}, A2:{(d1,1.0),(d2,1.0),(d4,0.2)}, Aggregator a1 can serve values of d1 with an incoherency bound greater than or equal to 0.5 whereas a2 can disseminate the same data item at a looser incoherency bound of 1.0 or more. In such a network of aggregators of multiple data items all the nodes can be considered as peers since a node ai can help another node ak to maintain incoherency bound of data item d1 ,but the node ai gets values of another data item d2 from ak.

Greedy Heuristics for deriving the Sub queries:

According to the Greedy algorithm for deriving sub queries, Firstly we need to get set of maximal sub queries (Mq) corresponding to all the data aggregators in the network .The maximal sub query for a data aggregator is defined as the largest part of the query which can be disseminated by Data aggregators that is the maximal sub query has all the query data items which the Data aggregator can disseminate. The problem of choosing sub queries while minimizing query execution cost is an NP-hard problem. We give efficient algorithms to choose the set of sub queries and their corresponding incoherency bounds for a given client query. In difference, all related work in this area, propose getting individual data items from the aggregators which leads to large number of refreshes. For solving the

ISSN: 2320-1363

above problem of best feasible dividing the client query into sub queries, we first need a method to estimate the query execution cost for various alternative options. A method for calculate approximately the query execution cost is another important contribution. As we divide the client query into sub queries such that each Sub query gets executed at different aggregator nodes, the query execution cost (i.e., number of refreshes) is the sum of the execution costs of its constituent sub queries. The model of the sub query execution cost as a function of dissemination costs of the being data items involved. The data distribution cost is dependent on data dynamics and the incoherency bound associated with the data. We model the data dynamics using a data dynamics model, and the effect of the incoherency bound using an incoherency bounce model. These two models are combined to get the estimate of the data dissemination cost.

IV. PERFORMANCE EVALUATION:

We simulated a network of data aggregators of 200 stock data items over 100 aggregator nodes such that each aggregator can disseminate combination of 25 to 50 data items. Data incoherency bounds, for various aggregator data items, were chosen uniformly between $0.005 and 0.02.We created 500 portfolio queries such data 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 .

Algorithms Comparison: No sub query, equal incoherency bound (native):

Here the client query is executed with each data item being disseminated to the client independent of other data items in the query. Incoherency bound divided equally among the data items.

No sub query, optimal incoherency bound (optic):

Here the data items are disseminated independently but incoherency bound is divided among data items. so the total number of refreshes can be maximized.

Random sub query selection (random):

Here the sub queries are obtained by randomly selecting a Data aggregator in the each iteration of the greedy algorithm.

Sub query selection while minimizing sum diff (min cost):

Here we need to minimize the query cost , a sub query with minimum cost per data item can be chosen in each iteration of the algorithm.

Sub query selection while maximizing gain (mingain):

Here for each sub query ,we calculate the relative gain of executing it by finding the sum diff difference between cases when data item is Obtained separately and where all the data items are aggregated as a single query.

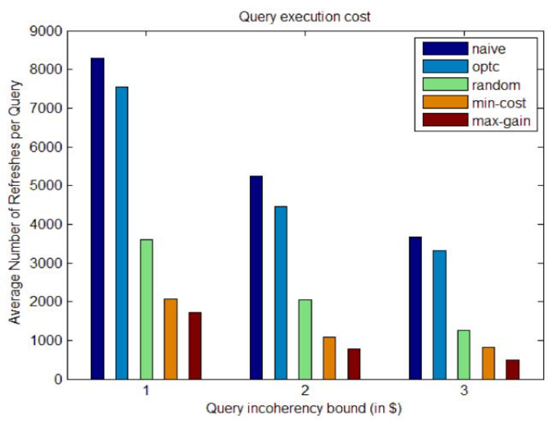


Fig2: Query Execution cost graph

In the above graph shows the average number of refreshes required for query incoherency bounds of $1-$3.The Native algorithm requires more than five times the number of messages compared to min-cost and max-gain algorithms. for incoherency bound of $3,each query, on average, requires 3,311 messages if it is executed just by optimizing incoherency bound (optic) compared to 487.when we select the query plan using the max-gain algorithm. The gains of our algorithms increase further as number of data items disseminated by Data aggregators increase .This happens as, with more data items per Data aggregators, sub query-based algorithms result in larger sub queries and we select sub queries intelligently.

Effects of Algorithmic Parameters :

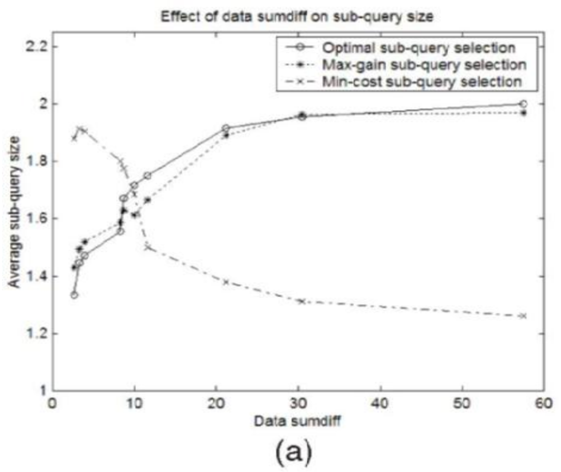
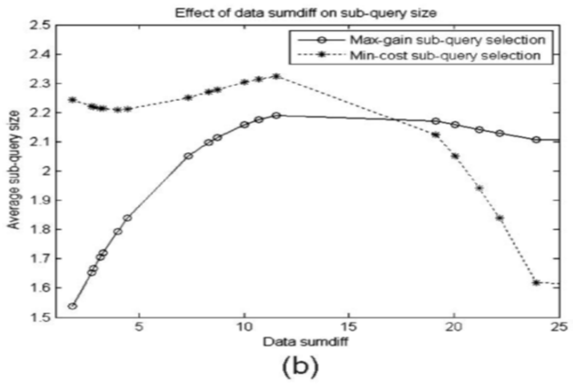


Fig3: Effect of data sumdiff on sub query size (a) Query Size=3



(b)Query Size =5

Fig4: Query Execution cost graph

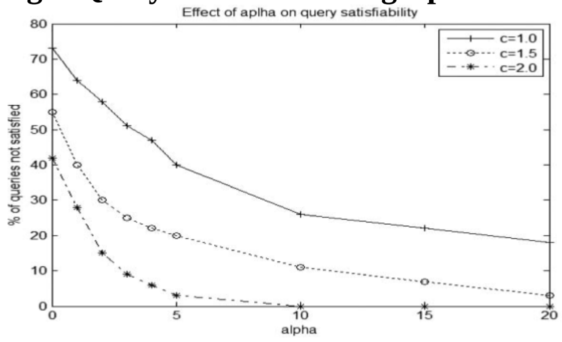
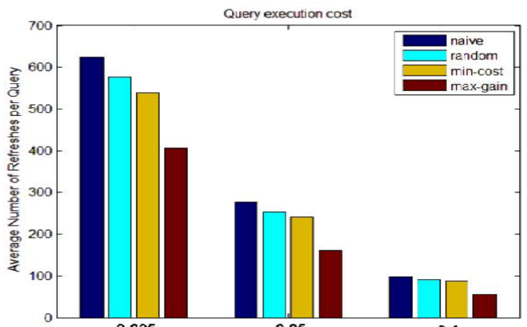


Fig5: Effect of α on query satisfiability.

ISSN: 2320-1363

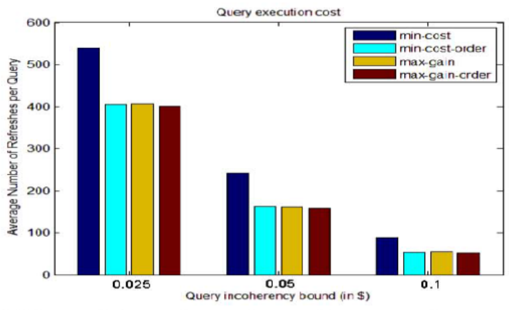
QUERY COST MODEL:

Query cost model is used to disseminating max of data items from a data aggregator. For a max query, the query result is the maximum of data item values. Thus the query dynamic is decided as per the dynamics of the data item with in the maximum value. Hence ,the query sumdiff is nothing but weighted average of data sum diffs, weighted by fraction of time when the particular data item is maximum.



(a) Comparison of algorithm

Fig6: comparisions inbetween of different algorithms



(b) Effect of *data dynamics order* on performance

Fig7: Performance related to different query plans

V.CONCLUSION

Here presents a cost-based approach to minimize the number of refreshes required to execute an incoherency bounded uninterrupted query. We assume the existence of a network of data aggregators, where each Dataaggregator is accomplished of disseminating 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 judiciously chosen data aggregator.

VI.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)*, 2013.

[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/sumdiffRaviVijay_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.
[14] S. Madden, M.J. Franklin, J. Hellerstein, and W. Hong, "TAG: A Tiny Aggregation Service for Ad-Hoc Sensor Networks," *Proc. Fifth Symp. Operating Systems Design and Implementation,* 2002.