ISSN-2320-7884 (ONLINE)
ISSN-2321-0257 (PRINT)

Continuous Aggregation Queries Based on Procedures of Adaptive Query Planning

Y.Leelakrishna [1], S.L.V.V.D Sarma M.Tech [2] ,J.Nagaraju M.Tech [3] , E.Ramakrishna[4]

[1]Student, EVM College of Engineering & Technology, Narsaroapet, Guntur (Dt).

[2]Assistant Professor, EVM College of Engineering & Technology, Narsaroapet, Guntur (Dt).

[3]Associate Professor, EVM College of Engineering & Technology, Narsaroapet, Guntur (Dt).

[4] MCA, EVM College of Engineering & Technology, Narsaroapet, Guntur (Dt).

Abstract: The passive web pages can transform into active environment by the continuous queries are persistent queries by providing the time varying dynamic query results useful for online decision making. To handle a large number of users with diverse interests a continuous query system must be capable of supporting server push style of Internet-based communication. A network of data aggregators has prior approaches for the scalable handling of push based data dissemination. Their implementation required Greedy Heuristics Algorithm along with pre configured incoherency bounds to manage both multiple aggregators and multiple clients for supporting server push based communications. The sub-optimal solutions are explored by existing heuristic-based approaches can only explore a limited solution space. So we propose to use an adaptive and cost-based approach. In a network of data aggregators, each dedicated and judiciously chosen aggregator serves a set of data items at specific coherencies. By our approach we can decompose a client query into sub-queries and executing sub-queries using aggregators with their individual sub-query incoherency bounds. Our cost model takes into account both the processing cost and the communication cost unlike prior approaches. Adaptive and cost-based approach has better performance in terms of both processing and communication cost than plain Greedy Heuristics approach and a practical implementation validates the proposed claim.

Keywords- greedy heuristics algorithm, aggregation, queries, process message.

1. INTRODUCTION

Continuous queries are persistent queries that allow users to receive new results when they become available. The general application such as auctions, personal portfolio valuations for financial decisions, sensors-based monitoring, route planning based on traffic information, etc., make extensive use of dynamic data. The passive web pages can transform into active environment by the continuous queries are persistent queries by providing the time varying dynamic query results useful for online decision making. They need to be able to support millions of queries due to the scale of the Internet. A continuous query system must be capable of supporting a large number of triggers expressed as complex queries against resident data storages in order to handle a large number of users with diverse interests. Many data intensive applications delivered over the Web suffer from performance and scalability issues. There is significant interest in systems that can efficiently deliver the relevant updates automatically. To know value of portfolio for a client; or the AVG of temperatures sensed by a set of sensors. In these continuous query applications, users are likely to tolerate some inaccuracy in the results. To support continuous queries for the users the system is maintained and managed by a multiple resource builders using network aggregators at a time. The exact data values at the corresponding data sources need not be reported as long as the query results

ISSN-2320-7884 (ONLINE)
ISSN-2321-0257 (PRINT)

satisfy user specified accuracy requirements. Prior Approaches use Greedy Heuristics Algorithm along with pre configured incoherency bounds to manage both multiple aggregators and multiple clients for supporting server push based communications. Adaptive and cost-based approach implementation involves

- Adaptation Attempt(to check for feasibility)
- Greedy Heuristics
- Simulated Annealing
- Process Message

In these continuous query applications, users are likely to tolerate some inaccuracy in the results. The exact data values at the corresponding data sources need not be reported as long as the query results satisfy user specified accuracy requirements.

Data Incoherency: The absolute difference in value of the data item at the data source and the value known to a client of the data. Let $v_i(t)$ denote the value of the i^{th} data item at the data source at time t. The value the data item known to the client be $u^i(t)$. But the successive incoherence at the client is given by $| v_i(t) - u_i(t) |$. As soon as a data incoherency exceeds C the data refresh message is sent to the client for the data item. i.e., $| v_i(t) - u_i(t) |> C$.

Network of Data Aggregators (DA).: Using push- or pull-based mechanisms the data refresh from data sources to clients can be done. In the pull-based mechanism data sources send messages to the client only when the client makes a request where as in the push-based mechanism data sources send update messages to clients on their own. For the data transfer between the source and the client we refer push-based mechanism. For scalable handling of push based data dissemination, network of data aggregators are proposed as that dissemination tree from sensor nodes to root already exists; and they also install error filters on partial aggregates. Data refreshes occur from data sources to the clients through one or more data aggregators.

We assume that each data aggregator maintains its configured incoherency bounds for various data items. In data dissemination a hierarchical network of data aggregators is employed such that each data aggregator serves the data item at some guaranteed incoherency bound. The data dissemination capability point of view, each data aggregator is characterized by a set of (di, ci) pairs.

Where d_i=data item
c_i=incoherency bound

The configured incoherency bound of a data item at a data aggregator can be maintained using any of following methods:

- 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.
- Data aggregator(s) with tighter incoherency bound help the DA to maintain its incoherency bound in a scalable manner.

Consider in the network the data aggregators managing the data items x_1-x_4, various aggregators can be characterized as

$A_1 := \{(x_1,0.5),(x_3,0.2)\}$
$A_2 := \{(x_1,1.0), (x_2,0.1), (x_4,0.2)\}$

Aggregator A_1 can serve values of the x_1 with an incoherency bound greater than or equal to the 0.5 where as the A_2 can disseminate the same data item at a looser incoherency bound of 1.0 or more.

QUERY AGGRIGATION AND THEIR EXECUTION

To execute in incoherency bounded continuous query plan is required. We present a technique for executing multidata aggregation queries. The theme of our scheme is to reduce the number of refresh messages from data aggregator to client. For the better understanding follow the scenario.

Scenario 1: Assume the query $Q=60X_1+190X_2+150X_3$ Where X_1, X_2, X_3 are data items for stock with incoherency bound of $75.

For the considered scenario the client can get the results as

a. Among data items Client can get data items separately on query incoherency bound is divided.
b. The query a single data aggregator can distribute to all data items to answer.
c. A single query can be divided into number of sub-queries and only one data aggregator gives their values.

ISSN-2320-7884 (ONLINE)
ISSN-2321-0257 (PRINT)

Scenario 2: For example the sensor network and an AVG query over the target set of sensors (d_1, d_2, d_3) injected at the query node. Aggregation is used in the networks for the efficient energy propagation of aggregates. Connect the *target* sensors and query nodes for constructing the aggregation tree, each node can select the path to the query node based on the preference factor. We want to select the aggregate path in the network such that we can execute minimum number of messages.

We have limited no. of options in both the case to execute the queries. In the scenario 1 we have numbers of refresh messages are dependent on the division of query incoherency bound. Data item incoherency can be defined as difference in value of data item at source and at the node. The main intent of the scenario are

a. Generating the sub-queries from the query
b. The Incoherence Bound is assigned
c. The sub-queries are executed at the selected data aggregations.
d. Reducing in the refreshing messages.

The total of the execution cost of sub-queries is nothing but number of refreshes. We use different models to implement and fulfill the aim of minimizing the refresh messages.

If the three data items require answering the client query if a single DA can disseminate, the DA construct the composite data item. Corresponding to client query the data item ($Q=60X_1+190X_2+150X_3$). The third option for the data dissemination is the divide and conquers technique. As per our scenario the query is divided in to the two ways

Plan1: Result of the sub query 60 X_1 +150 X_3 is served as the a_1 where as the $X_2$is served as the a_2.

Plan2: Result of the sub query 60 X_1 +190 X_2 is served as the a_1 where as the $X_3$is served as the a_2.

It is not guaranteed to be plan with the least no. of messages. In our example if we update X_1 and X_3, the X_1 increase and X_3 decreases and the vice versa. To achieve a query plan following work is need to be done.

Determine sub-queries: The sub- queries are achieved from client query q.

Divide incoherency bound: Divide query incoherency bound among sub-queries.

Algorithm: For Query Plan Selection we use Greedy algorithm.

result$\leftarrow\phi$

while $M_q \neq \phi$

choose a sub-query $m_i \in M_q$ *with criterion* ψ*:*

result$\leftarrow$ *result* $\cup$ *mi:* $M_q \leftarrow M_q - \{m_i\}$

for each data item $d \in m_i$

for each $m_j \in M_q$

$m_j \leftarrow m_j - \{d\}$;

if $m_j = \phi$ $M_q \leftarrow M_q - \{m_j\}$;

else calculate sumdiff for modified m_j;

return result

Greedy Algorithm for Query plan Selection

2. DATA DISSEMINATION COST MODEL

We present the model to estimate the number of refreshes required to disseminate a data item while maintaining a certain incoherency bound. Two factors affecting the number of messages that reneeded to maintain the coherency requirement:

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

Model for Incoherency Bound: A data item which needs to be disseminated at an incoherency bound C; new value of the data item will be pushed if the value deviates by more than C from the last pushed value. It will be proportional to the |v(t)-u(t)| to C at the source/aggregator and u(t)at the client, at time t. A data source in the push-based data dissemination in our scheme as follows:

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

ISSN-2320-7884 (ONLINE)
ISSN-2321-0257 (PRINT)

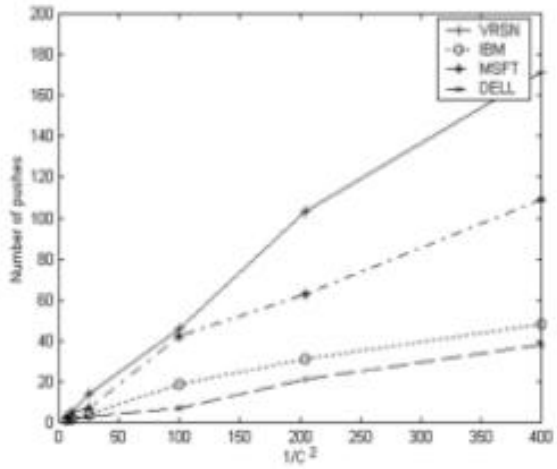

Fig 1: Number of purchases VS incoherence Bounds

In both the tray's the value at the source can be modeled as a random process with average as the value known at the client. In tray2 the client and the server estimate the data value as the mean of the modeled random process, where as in case1 deviation from the last pushed value can be modeled as zero mean process. We unbind the related work on scalable answering of aggregation queries over a network.

***Answering Incoherency Bounded Aggregation Queries*:**

Various mechanisms for efficiently answering incoherency bounded aggregation queries over continuously changing data items are proposed in the literature. For minimizing the number of pulls, both predict data values and pull instances. It lead to different messaging overheads for different DAs as opposed. We propose combinations of number of hops and remaining energy to select a particular path from various options available between any two nodes.

Construction and Maintenance of Network of Data Aggregators. The construction and maintenance of hierarchical network of data aggregators for providing scalability and fidelity in disseminating dynamic data items to a large number of clients. Finding the minimal cardinality subset of a given test suite that overs the same set of requirements as covered by the original test uite is NP complete. This can be shown by a polynomial time reuction from the *minimum set-cover* problem.

3. EXISTING SYSTEM

Push, or server push, describes a style of Internet-based communication where the request for a given transaction is initiated by the publisher or central server. Uses server push based techniques for initiating communications. Push services are often based on information preferences expressed in advance. This is called a publish/subscribe model. Whenever new content is available on one of those channels, the server would push that information out to the user. Data refreshes occur from data sources to the clients through one or more data aggregators for the scalable handling of push based data dissemination in use of the network aggregation. Uses Greedy Heuristics Algorithm along with pre configured incoherency bounds to manage both multiple aggregators and multiple clients thus delivering a better performance. While continuous query systems can transform a passive web into an active environment, they need to be able to support millions of queries due to the scale of the Internet. Continuous queries are persistent queries that allow users to receive new results when they become available. For example, users might want to issue continuous queries of the form:

"Notify me whenever the price of Dell or Micron stock drops by more than 5%and the price of Intel stock remains unchanged over next three month."

In order to handle a large number of users with diverse interests, a continuous query system must be capable of supporting a large number of triggers expressed as complex queries against resident data storages. It is better to support continuous queries for the users, at a time where the system is maintained and managed by a multiple resource builders using network aggregators.

4. PROPOSED SYSTEM

For supporting server push based communications Greedy Heuristics Algorithm along with pre configured incoherency bounds to manage both multiple aggregators and multiple clients. Query optimization strategies developed using Greedy

ISSN-2320-7884 (ONLINE)
ISSN-2321-0257 (PRINT)

Heuristics Algorithm depends on processing cost only. We can decompose a client query into sub-queries and executing sub-queries using aggregators with their individual sub-query incoherency bounds. Existing heuristic-based approaches can only explore a limited solution space and hence may lead to sub-optimal solutions. Hence the solution is obtained to use an adaptive and cost-based approach. . Data accuracy can be defined as the absolute difference in value of the data item at the data source and the value known to a client of the data, and specified in terms of incoherency of a data item. Using push- or pull-based mechanisms the data refresh from data sources to clients can be done. In the pull-based mechanism data sources send messages to the client only when the client makes a request where as in the push-based mechanism data sources send update messages to clients on their own. We assume the push-based mechanism for data transfer between data sources and clients. For scalable handling of push based data dissemination, network of data aggregators are proposed as that dissemination tree from sensor nodes to root already exists. Adaptive and cost-based approach has better performance in terms of both processing and communication cost than plain Greedy Heuristics approach.

GreedyJoinOrdering-1(R = {R},w : R → R)

Input: a set of relations to be joined and weight function

Output: a join order

```
S = 0
while (|R| > 0)
{
m = arg min Ri ∈ R w(R)
R = R \ {m}
S = S◦ < m >
}
return S
```

Greedy Heuristics approach

- The previous greedy algorithms only construct left-deep trees
- Greedy Operator Ordering (GOO) constructs bushy trees

Idea: Combine joins trees such that the intermediate result is minimal, where relations have to be joined somewhere but joins can also happen between whole join trees.

Costs: The costs for a totally ordered precedence graph G can be computed as follows:

$$C_H(G) = \sum_{i=2}^{n}[n_{1,2,\ldots,i-1}h_i(n_i)]$$
$$= \sum_{i=2}^{n}[(\prod_{j=1}^{i} s_j n_j)h_i(n_i)]$$

5. PERFORMASNCE ANALYSIS

Existing heuristic-based approaches can only explore a limited solution space and hence may lead to sub-optimal solutions. Adaptive and cost-based approach has better performance in terms of both processing and communication cost than plain Greedy Heuristics approach. We present the simulation results on query planning for increasing the network performance in real word entity. Compared to all the results in various boundary values in the query processing with fitness values.

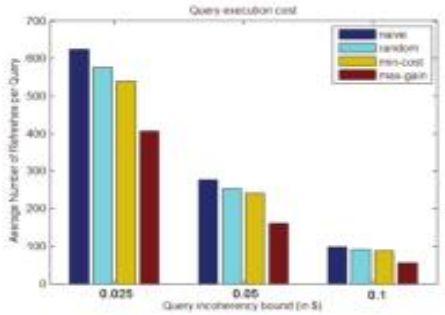

Fig 2: Comparison of greedy algorithm with heuristic results.

For checking the feasibility in network, we have to use adoption attempts in the network regarding query processing and communication process management in the network environment.

6. CONCLUSION

Continuous queries are persistent queries that allow users to receive new results when they become available. The service provider is maintained and managed by a single resource builder for supporting continuous queries for the users. In order to handle a large number of users with diverse interests, a continuous query system must be capable of supporting a large number of triggers expressed as complex queries against resident data storages. This is called a

ISSN-2320-7884 (ONLINE)
ISSN-2321-0257 (PRINT)

publish/subscribe model. Whenever new content is available on one of those channels, the server would push that information out to the user. So we propose to use an adaptive and cost-based approach. Our cost model takes into account both the processing cost and the communication cost. Adaptive and cost-based approach has better performance in terms of both processing and communication cost than plain Greedy Heuristics approach.

1. REFRENCES

[1] Rajeev Gupta, Kirthi Ramaritham, " Query Planning For Continuous Aggregation Queries Over A Network Of data Aggregators" vol 24, No. 6. June 2012.

[2] S.Shah, K. Ramamritham, and P. Shenoy, "Maintainin Coherency of Dynamic Data in Cooperating Repositories," Proc. 28th Int'l Conf.Very Large Data Bases (VLDB), 2002

[3] 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.

[4] S. Agrawal, K. Ramamritham, and S. Shah, "Construction of aTemporal Coherency Preserving Dynamic Data Dissemination Network," Proc. IEEE 25th Int'l Real-Time Systems Symp. (RTSS), 2004.

[5] 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.

[6] 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.

[7] 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.

[8] M.R. Garey and D.S. Johnson, "Computers and Intractability-A Guide to the Theory of NP-Completeness," V Klee, Ed. Freeman, New York, 1979.