ISSN 2320-6802

INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY

WINGS TO YOUR THOUGHTS.....

Executing Queries over a Network of Data Aggregators in Cloud Network

Suvarna Pawar [1], Komal Chopra[2]

[1]Associate Professor, Dept. of IT, AVCOE,
Sangamner, Maharashtra.
pawar.suvarna@gmail.com

[2]Student of M. E. IT, AVCOE,
Sangamner, Maharashtra.
komal.17chopra@gmail.com

Abstract: *To keep track of dynamic data and to gain rapid data access queries are being used constantly. In a distributed data environment user needs to get the value of aggregation function for data sets. Since the data is collected from multiple sources the need to process the query and to specify the coherency is necessary. Coherency requirement is specified by client. In this paper our system decomposes a client query into sub queries. Each sub-query is executed on selected data aggregator and they are provided with the sub query in coherency bound. The main intend of our work is to reduce the number of refresh messages. To estimate and find the number of refresh messages we build a Query Cost Model. Our estimate and our approach is used to decrease the number of refresh messages. The number of refresh messages is reduced by one third of the existing system.*

Keywords: *Queries, data dissemination, incoherency bound, natural language processing, query processing.*

1. INTRODUCTION

Web applications or distributed data use data that is timely changing means dynamic data. Applications like stock portfolio and search engines also use data that is changing rapidly. Various applications like content distribution network and sensor based monitoring makes wide use of dynamic data. Many such applications make use of dynamic data, so the systems that update the data repeatedly need to be used. For example, consider a web application. Client sends its query to the central site. To answer the query central site need to find which aggregators have the data item to answer the query. Its working is same as Google search engine. Another example would be portfolio of stock. The data values for this need to be reorganized continuously as they are changing. Thus the queries are refreshed. Sometimes client may face with some inaccuracy in the result. We are going to develop a system which takes as input query in natural language and that query is processed and converted into SQL queries at the back-end.

Data Accuracy is defined in terms of incoherency of data item. It is difference in value of data item at source and value of data item at user.

It can also be denoted as |vi (t)-ui (t)|

Where vi= value of ith data item at the source

Ui= value of ith data item at the user

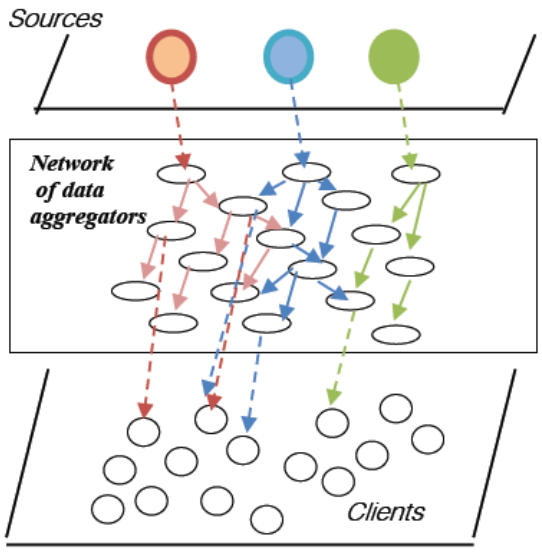

Figure 1: Data dissemination network for multiple data items

INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY
WINGS TO YOUR THOUGHTS.....

The data refresh message is sent to user whenever incoherency is increased above C i.e |vi(t)-ui(t)|>C. Data refresh need to be done from source to client. It can be done using pull or push based technique. To transfer data between source and client we use push based scheme. In push based technique source sends update message to client without any request that is on their own. In pull based technique source send update message to client only when client makes request. Incoherency bound is maintained by each data aggregator is assumed. Each data aggregator has (di,ci) pairs where di is data item which DA can disseminate at incoherency bound ci.

2. BACKGROUND

In this paper we present a method for executing multi data aggregation queries. The main plan is to reduce the number of refresh messages from data aggregator where the queries are aggregated to client. Consider a situation to better understand this concept.

Scenario: - Consider query Q=55d1+200d2+154d3

Where d1, d2, d3 are data items for stock with incoherency bound of $80.

To answer the query given in situation, user can get outcome in three probable ways:-

1. Client can get data items separately. Among data items query incoherency bound is divided.
2. To answer the query a single data aggregator can distribute all data items.
3. A single query can be divided into number of sub-queries and only one data aggregator gives their values.

Numbers of refresh messages are reliant on the division of query incoherency bound. Incoherency of a data item can be defined as difference in value of data item at source and at the node.

We will be using different methods for different purposes. The main purpose is:-

- Dividing a query into sub-queries.
- Assign an incoherency bound.
- At selected data aggregators sub-queries are executed.
- Number of refresh messages is reduced.

In this paper, we divide the query into sub-queries and it is executed at aggregator's node. The sum of the execution cost of sub-queries is nothing but number of refreshes. The cost of data dissemination depends on data that is varied and incoherency bound. We use diverse models to implement and fulfill the intend of minimizing the refresh messages. Data dissemination cost model is used to discover the number of refreshes.

3. RELATED WORK

In the existing systems dissemination tree from data source to client already exists and error filters are also installed. It was first proposed the idea of "continuous queries" as queries that are issued once and run continuously. In this paper, each data aggregator can only disseminate data at some previously specified incoherency bound. Authors use data histograms to optimally assign local thresholds at monitoring sites for monitoring at central i.e global sites. Maintaining histogram is difficult since it requires more space and time compared to sumdiff based technique. Chebyshev's inequality is also used to present that expected communication cost is inversely proportional to square of the error budget. But in our paper we assume that number of refresh message is proportional to data variance. In the existing systems queries are assigned to aggregators in CDN. But in our paper we use queries for multiple data items.

4. PROPOSED SYSTEM

In this paper we will present model for estimating the number of refresh messages and also divide the query into sub-queries. There are two factors that affect the number of messages.

1. Coherency Requirement
2. Dynamics of data.

Incoherency Bound Model

A data item is given new value only when the values increased by more than C from last value. We are using push-based method, so in this data source follows one of the schemes.

a. Source gives new value only when it is different from last value by more than C.
b. Source gives new value whenever it differs from the expected client value by more than C.

So therefore number of refresh messages is inversely proportional to square of incoherency bond.

Data dynamics model:

We use fast Fourier transform, it is also use in digital signal processing. It is useful as it helps in specifying and capturing number of changes in data value its amount n time. But it too

INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY
WINGS TO YOUR THOUGHTS.....

has a problem. We need function over FFT coefficients to find the number of refreshes. As per changes in data values the number of FFT coefficients can be increased.

Combining Model:

Thus we have seen that number of refresh messages is proportional to data R_s and inversely proportional to square of incoherency bound C^2. We should not distribute any messages when data value is constant or in coherency bound is not limited.

Query Planning For Weighted Additive Aggregation Queries:-

A query plan is required to execute in incoherency bounded continuous query. To achieve a query plan following work is need to be done.

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

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

Algorithm: Greedy Algorithm for Query Plan Selection

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

Natural language processing (NLP) is the ability of a computer program to know human speech as it is spoken. The development of NLP applications is challenging because computers traditionally require humans to speak to them in a programming language that is accurate, unmistakable and highly structured or, perhaps through a limited number of clearly-enunciated voice commands. The ultimate aim of NLP is to do away with computer programming languages altogether. Instead of particular languages such as Java or C, there would only be "human." NLP involve natural language understanding that is, enabling computers to obtain meaning from human or natural language input.

Major tasks in NLP

• Natural language generation: It converts information from computer databases into human known language. Natural Language Generation (NLG) is the natural language processing task of generating natural language from a machine representation system such as a knowledge base or a logical form.

• Natural language understanding: Convert chunks of text into more formal representations such as first-order logic structures that are easier for computer programs to operate. Natural language understanding involves the identification of the intended semantic from the multiple possible semantics which can be derived from a natural language expression which usually takes the form of organized notations of natural languages concepts.

5. CONCLUSION

This paper presents a cost-based approach which is used to reduce the number of refresh messages. We divide the queries into sub-queries and assign them incoherency bound. The methods used will help us to reduce the number of refresh messages by one third then used for existing systems. We also developed model for Natural language queries using natural language processing.

6. FUTURE WORK

Developing the same project for natural language is the future scope. The Natural language will be converted into SQL queries at the backend and then result i.e. output will be displayed. Another area of future scope will be we will consider hierarchy of data aggregators.

References

[1] "Query Planning for Continous Aggregation Queries over a Network of Data Aggregators", Rajiv Gupta and Krithi Ramamritham, *Fellow IEEE*.

[2] 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.

[3] S. Shah, K. Ramamritham, and P. Shenoy, "Maintaining

INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY
WINGS TO YOUR THOUGHTS.....

Coherency of Dynamic Data in Cooperating Repositories," Proc. 28th Int'l Conf. Very Large Data Bases (VLDB), 2002.

[4] 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.

[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. 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.

[8] 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.

[9] S. Zhu and C. Ravishankar, "Stochastic Consistency and Scalable Pull-Based Caching for Erratic Data Sources," Proc. 30th Int'l Conf. Very Large Data Bases (VLDB) 2004.

[10] A. Deligiannakis, Y. Kotidis, and N. Roussopoulos, "Processing Approximate Aggregate Queries in Wireless Sensor Networks," Information Systems, vol. 31, no. 8, pp. 770-792, 2006.