Volume 4, Issue 3, March 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Information Retrieval Perspective Approach for Continuous Queries Using Data Aggregators

Pramod M. Bagal
Research Scholar M. Tech., Dept. of Information Tech.
SSSIST, RGPV
Sehore (M.P.), India

Mr. Gajendra Singh Chandel
HOD, Dept. of Information Tech.
SSSIST, RGPV
Sehore (M.P.), India

Abstract— Data aggregators aim to help users to access continuous queries and retrieve relevant information or items from large data, by automatically finding and suggesting products or services of likely interest based on observed evidence of the users preferences. The purpose of this study is to improve execution of continuous queries to produce accurate result with minimum time period with the help of data aggregators. Apart from that, data aggregators to use monitoring the changes to provide useful results helpful to large number of users with high security, accuracy, efficiency, and scalability. Hence, we are investigates algorithmic approach to answer continuous queries and accuracy associated with their uncertainty. This assessment requires a clear information retrieval method for queries evaluation methodology against the quantifying information loss through data aggregation. The conclusion of this study is that usage of data aggregators to simplify query to improve result and analyzing significant amount of information.

Keywords— Data aggregator, Continuous queries, Information Retrieval, Data dissemination, Incoherency bound.

I. INTRODUCTION

Now days most of applications those are commercially in use having utilized the concepts of data aggregators in order answer the continuous queries. The Web is becoming a universal medium for information publication and use. Applications 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 [1, 6, 10]. For such applications, data from one or more independent data sources may be aggregated continuous queries to determine if some action is warranted to improve performance.

Over the last 10-15 years, computer technologies have been introduced to help people deal with these vast amounts of information and they have been widely used in research as well as above such applications [2]. As an example, data values from possibly different sources are required to be aggregated to satisfy user's requirement. Based upon policies the typical aggregation may look like the following:

- Keep raw data for 24 hours
- Data greater than 24 hours but less than 7 days aggregate to once per hour
- Data greater than 7 days aggregate to 24 hours

The actual policies for each user may be more or less complex than what is depicted above but it does show a typical answering strategy.

We propose the approach to answering and improve information retrieval to answering continuous queries.

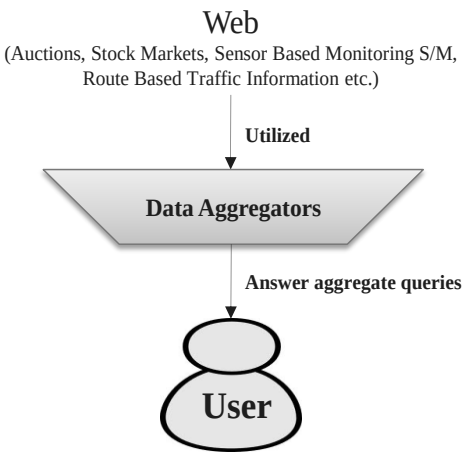

Fig. 1. Function of data aggregators

II. NEED AND SIGNIFICANCE OF THE WORK

This section demonstrates the need and significance of our approach with various functionalities. Also, we demonstrate what the purpose of aggregation is and why data refresh is required to satisfy user requirement.

A. Data aggregation

Data aggregation is any process in which information is gathered and expressed in a summary form, for purposes such as statistical analysis. A common aggregation purpose is to get more information about particular groups based on specific variables such as age, profession, or income.

The information about such groups can be used for web personalization to choose content and advertising likely to appeal to an individual belonging to one or more groups for which data has been collected. For example, a site that sells music CDs might advertise certain CDs based on the age of the user and the data aggregate for their age group.

B. Data Incoherency

Data accuracy can be defined in terms of incoherency of data item. It is absolute difference in value of data item at source and value of data item at client [1]. It can also be denoted as $|vi(t)-ui(t)|$. Where vi=value of i^{th} data item at the source, Ui=value of i^{th} data item at the client

The data refresh message is sent to client whenever incoherency is increased exceeds C i.e. $|vi(t)-ui(t)|>C$.

C. Data Aggregator (DA) and Network of Data Aggregators

Data aggregators can be called as organizers involved in compiling information from detailed database on individuals and selling information to others.

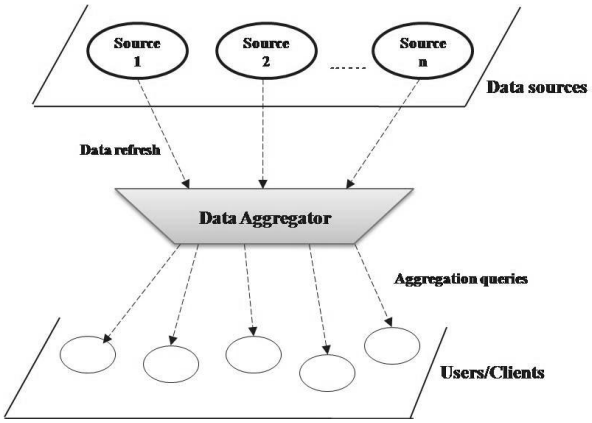

Fig. 2. Query execution at single data aggregator

Fig. 3.

DA is one kind of secondary or alternate servers it serves as data sources (data items) in the computation of results from detailed database. For performing online analysis dynamic data is prominent thing. DA collects the information from designated databases and giving the data to the user.

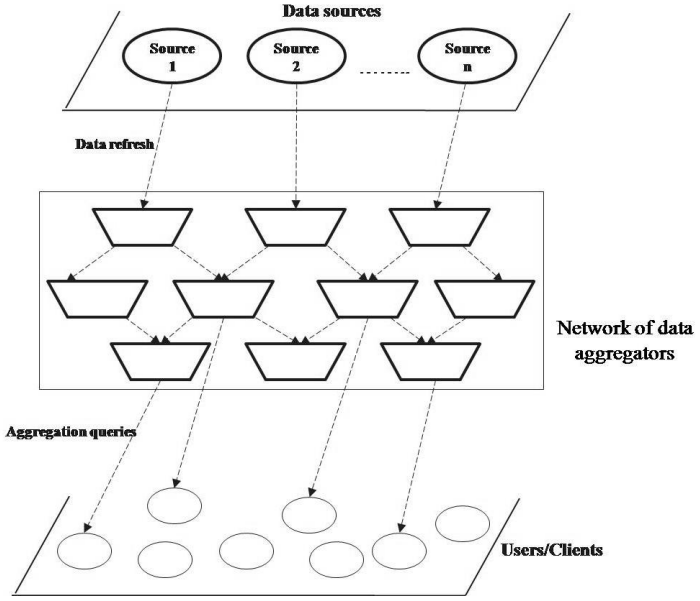

Fig. 4. Query execution at network of data aggregators

Fig. 5.

Data refresh/transfer data need to be done from source to client. The data refreshes can be done using two mechanisms [1, 6].

- Pull Based Technique: Source send update message to client only when client makes request.
- Push Based Technique: Source sends update message to client without any request that is on their own.

Fig. 2 and 3 shows data refreshes occur from data sources to clients through one or more data aggregators. The required data updates for query evaluation at a DA can be obtained either by sources pushing them continuously or the DA pulling them whenever required.

D. Classification of Queries on the Basis of Usage

Client's query may be simple query or complex/ multi-query. As an example, consider a user who wants to track the temperature of various locations on different months. Temperature data values from possibly aggregate different sources are required to satisfy user requirements. Sending multiple queries to a data source at a time may produce inaccurate result to client and reduces the performance, by not reporting immediately.

- Simple Queries: Queries that can access data from single data source based on one matching condition are Simple Queries.
- Complex Queries: are compositions of simple queries which access data from multiple data sources and have more matching condition.

Consider the scenario,

- Scenario-1: Retrieve the temperature details of Chennai.
- Scenario-2: Retrieve the temperature details of all states of India, where temperature is more than 35 degree Celsius, in the month of January.

Scenario-1 deals with temperature about single location, having single matching conditions whereas Scenario-2 deals with multiple matching conditions like finding the states, temperature more than 35 degree Celsius, only on January month. This work deals with the complex queries as like scenario-2. It is also applicable for simple queries.

- Continuous Queries: Continuous queries are persistent queries that allow users to receive new results when they become available.
- Aggregate Queries: That possibly aggregate different source is required to satisfy user requirement.

For example, users might want to issue continuous queries of the form:

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

Fig. 4 Refer the comparison between execution time of the queries [8]. An aggregate function maps a finite bag of values into some domain [7].

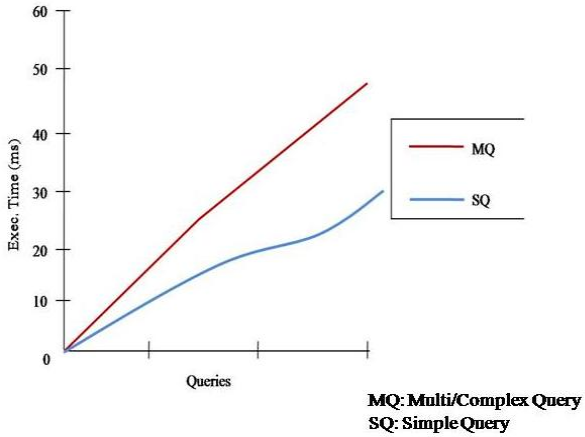

Fig. 6. Queries and its execution time evaluation

III. LITERATURE REVIEW

In [1], the authors ensure that each data item for a client query is disseminated by one and only one data aggregator (refer Fig. 2). Also author prove that the problem of choosing sub queries while minimizing query execution cost is an NP-hard problem.

In [2], author the issue of aggregate diversity in recommender systems has been largely untouched. In [3], none of these work focus on the issue of storing and reutilizing parse trees. In [4], they can not address the problem of so-called concept drift and shift.

In [6], consider different pull- and push-based mechanisms to execute various types of continuous aggregation queries. In any client-server model, it's possible to refresh a data value from a server either by a client pulling the value from the server, or the server pushing it to the client. In the case of pull, the user might need to pull periodically from the server to know whether the data value has changed and to get the updated results. In the case of push, the user expresses interest in the data value to the server, and the server sends the value whenever a relevant update occurs.

In [10], involves studying two core problems: (1) how to realize meaningful and dynamic operation migration under varying system load, and (2) how to evaluate the migrated operations in the database efficiently. In this paper, prototype architecture is proposed and solution concepts for the two problems are presented.

One important question [1, 2, 3, 6] for satisfying client requests through a network of nodes is how to select the best node (s) to satisfy the request.

In [14], the authors conclude some important open problems, such as parametric query optimization, Parallelism, Routing/adaptation policies.

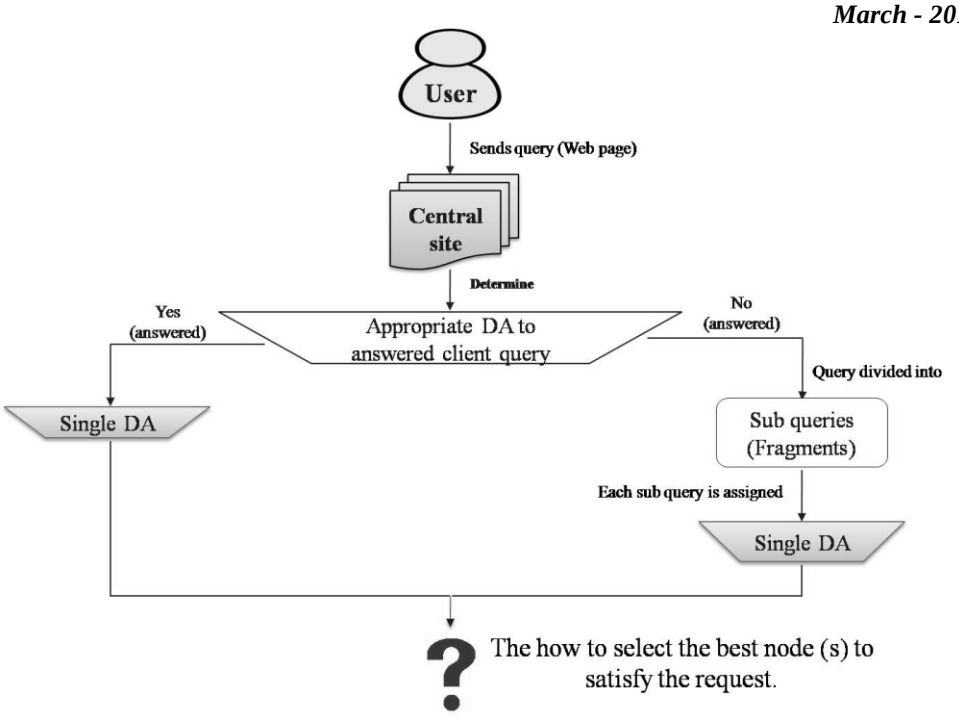

Fig. 7. Existing approach

There are two primary factors affecting the number of messages that are needed to maintain the coherency requirement: 1) the coherency requirement itself and 2) dynamics of the data.

In data dissemination network of DA's, data refreshes occur from data sources to the client through one or more DA's. In this scenario assume that each DA maintains its configured incoherency bounds for various data items. Here the strategy present to estimate the information retrieval approach required to disseminate a data item while maintaining a certain incoherency bound shown in fig.6.

Thus, From a data dissemination capability point of view, each DA is characterized by a set of (d_i,c_i) pairs, where d_i=data item c_i=incoherency bound.

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

- Data source refreshes the data value of the DA whenever DA's incoherency bound is about to get violated. This method is scalability problems.
- Data aggregators with tighter incoherency bound help the DA to maintain its incoherency bound in scalable manner as explained in [1].

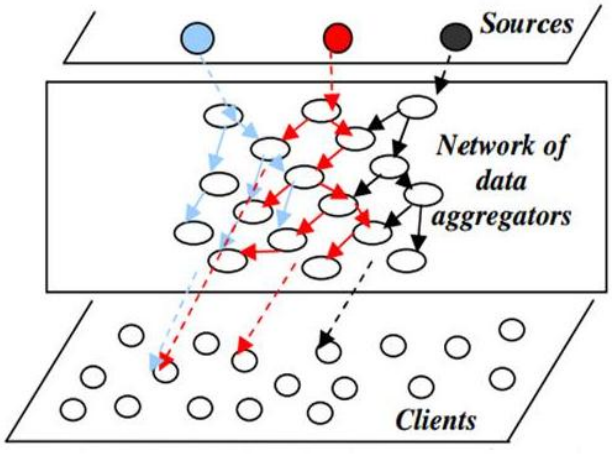

Fig. 8. Data dissemination network for multiple data items

E. Disadvantage

- Each client's data specifications are satisfied by single DA. In such case, data aggregators may need to disseminate extensive data items which will lead to huge number of refresh messages, so increase in delay.
- The exact data value at the corresponding data source need not be reported as long as the query result satisfies user specified accuracy requirements.

Apart from this, need to find the information retrieval approach that satisfies client coherency requirement with the least number of refreshes requires. Hence, developing efficient strategies for multiple invocations of our methodology, to considering hierarchy of data aggregators, is an area for research. Also, there is need to changing a information retrieval perspective approach when continuous queries are executed.

Data aggregators are dynamically assigned to sub query based on attributes so query execution plan is applicable for dataset having limited attributes. This will lead to poor performance for dataset having more dimensions. So, we present efficient technique for handling more dimensions is an area for future work.

IV. PROPOSED METHODOLOGY

Use continuous querying clients receive continuous updates to queries run on the servers. Continuous queries are only run by a client on its servers. Use continuous querying in your clients to receive continuous updates to queries run on the servers. That can use a Continuous query in any of client regions.

Continuous queries are used to monitor changes to time varying data and to provide results useful for online decision making. In proposed experimental approach present a low-cost, scalable technique to answer continuous aggregation queries using dynamic data items.

According to that work distinguishes itself by employing sub query plan to minimize number of refreshes. A method for estimating the query execution cost is another important contribution of this work.

A. Advantages

- Manage both multiple aggregators and multiple clients to delivering a better performance in terms to saves the time and the user spending low cost.
- Adaptive query execution cost in terms of number of refreshes is minimized between data sources and clients.

B. Executing Queries Using Sub Queries

To answering continuous query, the method is required which includes the set of sub queries, their individual incoherency bounds and data aggregators which can execute these sub-queries. In experimental result, need to find the optimal query execution plan which satisfies client coherency requirement with the least number of refreshes. Following is a mechanism for information retrieval perspective;

- Task 1: Split the aggregation query into sub queries.
- Task 2: Allocate the query incoherency bound among them.

While satisfying the following conditions:

- Condition-1: Query incoherency bound is satisfied.
- Condition-2: The chosen DA should be able to provide all the data items appearing in the sub query assigned to it.
- Condition-3: Data incoherency bounds at the chosen DA should be such that the sub-query incoherency bound can be satisfied at the chosen DA.
- Objective: Number of refreshes should be minimized.

V. CONCLUSION AND FUTURE SCOPE

In this literature presents approach that allows users to receive new results when they become available. Whenever new content is available on one of those channels, the active data aggregator would push that information to the user. Also, experimental result to minimize the number of refreshes required to execute an incoherency bound for continuous query where each data aggregator is capable. Our work can be extended in a number of directions. First this strategy reduces time taken for executing sub-queries. It is showed that dividing multi-queries into independent sub-queries and executing it parallel, improves performance and reduces execution time. Second, developing strategy for multiple invocations and information retrieve effectively as data dynamics changes. Third, data aggregators are dynamically assigned to sub query based on attributes so query execution plan is applicable for dataset having limited attributes. This will lead to poor performance and information loss.

Also, last interesting improvement in security strategy currently proves difficult to answer. This work gives rise to several interesting directions for future research.

REFERENCES

[1] Rajeev Gupta and Kirthi Ramamrithm, "Query Planning for Continuous Aggregation Queries Over a Network of data Aggregators", *IEEE Trans. on Knowledge and Data Engineering*, vol. 24, no. 6, June 2012, pp. 1065-1079.

[2] Gediminas Adomavicius and YoungOk Kwon, "Improving Aggregate Recommendation Diversity Using Ranking-Based Techniques", *IEEE Trans. on Knowledge and Data Engineering*, vol. 24, no. 5, May 2012, pp. 896-911.

[3] Luis Tari,Student, Phan Huy Tu, Jo¨rg Hakenberg, Yi Chen,Member, Tran Cao Son, Graciela Gonzalez, and Chitta Baral, "Incremental Information Extraction Using Relational Databases", *IEEE Trans. on Knowledge and Data Engineering*, vol. 24, no. 1, January 2012, pp. 86-99.

[4] Jose Antonio Iglesias, Plamen Angelov, Agapito Ledezma, and Araceli Sanchis, "Creating Evolving User Behavior Profiles Automatically", *IEEE Trans. on Knowledge and Data Engineering*, vol. 24, no. 5, May 2012, pp. 854-867.

[5] Lawrence Carin, George Cybenko, and Jeff Hughes, "Cybersecurity Strategies: The QuERIES Methodology", *Published by the IEEE Computer Society*, August 2008, pp. 20-26.

[6] Rajeev Gupta and Krithi Ramamritham, "Scalable Execution of Continuous Aggregation Queries over Web Data", *Published by the IEEE Computer Society*, January/February 2012, pp. 43-50.

[7] Serge Abiteboul, T.-H. Hubert Chan, Evgeny Kharlamov, Werner Nutt, and Pierre Senellart, "Capturing Continuous Data and Answering Aggregate Queries in Probabilistic XML", ACM Transactions on Database Systems, pp. A:4-A:45.

[8] Yan-Nei Law and Carlo Zaniolo, "Improving the accuracy of continuous aggregates and mining queries on data streams under load shedding", *Int. J. Business Intelligence and Data Mining*, 2008, pp. 99-117.

[9] Tyson Condie, Neil Conway, Peter Alvaro, Joseph M. Hellerstein, John Gerth, Justin Talbot, Khaled Elmeleegy, and Russell Sears, "Online Aggregation and Continuous Query support in Map Reduce", *SIGMOD'10*, Indianapolis, Indiana, USA, June 2010.

[10] Yuanzhen Ji, "Database Support for Processing Complex Aggregate Queries over Data Streams", *EDBT/ICDT '13*, Genoa, Italy, March 2013.

[11] Like Gao and X. Sean Wang, "Improving the Performance of Continuous Queries on Fast Data Streams: Time Series Case", Dept. of Information and Software Engineering, George Mason University, Fairfax, Virginia.

[12] Lory Al Moakar, Panos K.Chrysanthis, Christine Chung, Shenoda Guirguis, Alexandros Labrinidis, Panayiotis Neophytou, and Kirk Pruhs, "Auction-based Admission Control for Continuous Queries in a Multi-Tenant DSMS", *International Journal of Next-Generation Computing*, Vol. 3, No. 3, November 2012, pp.247-273.

[13] E. Pourabbas and, A. Shoshani, "Improving Estimation Accuracy Of Aggregate Queries On Data Cubes", *Data and Knowledge Engineering*, 2009, pp. 1-23.

[14] Amol Deshpand, Zachary Ives, and Vijayshankar Raman, "Adaptive Query Processing: Why, How, When, What Next?", *VLDB '07*, September 23-28, 2007, Vienna, Austria.

