Aggregation Environment for Query Optimization in Network Monitoring

Ms. Manisha Thombare
Department of Computer Engineering
MET's Institute of Engineering
Nashik, India
University of Pune
thombaremanisha980@gmail.com

Prof. K. V. Metre
Department of Computer Engineering
MET's Institute of Engineering
Nashik, India
kvmetre@gmail.com

Abstract**—** On-line decision making often involves significant amount of time-varying data. Examples of such data include information such as network data, currency exchange and stock prices, real-time traffic and weather information and data from sensors in industrial process control applications. In these queries a client specifies a coherency requirement as part of the query. Each data aggregator serves a set of data items at specific coherencies. The client query is decomposed into sub-queries and that sub-queries are executed on judiciously chosen data aggregators with their individual sub-query incoherency bounds. Each sub-query gives the different query plan. In network monitoring data packet information is used as dynamic data items. This can be used for denial of service and intrusion detection. Also the system can be used for detecting the highest bandwidth consumption within the network. In the case of servers adhering to the HTTP protocol, clients need to frequently pull the data based on the dynamics of the data and a user's coherency requirements. In contrast, servers that possess *push* capability maintain state information pertaining to clients and push only those changes that are of interest to a user. These two canonical techniques have complementary properties with respect to the level of temporal coherency maintained, communication overheads, state space overheads, and loss of coherency due to (server) failures.

Keywords*-Data aggregator, Decision making, Incoherency bound, Network monitoring.*

*****

I. INTRODUCTION

An internet is growing on a large scale and the necessity of dynamic data is increasing day by day. The dynamic data is used for online decision making. The continuous queries are used for decision making. The continuous queries are long running queries which continuously give the information. Suppose the query for stock data is running then it will continuously update the stock information. To getting this type of updates the data dissemination techniques are used. There are two techniques are available for data dissemination such as pull based and push based. In pull based data dissemination technique user continuously requests for the data. The data is dynamic so the query coherency is set. Coherency bound is the difference of values at data source and user. Suppose s_i be the value at data source and u_i be the value at client then the $|u_i - s_i| < c$, where c is the query incoherency bound. The coherency bound can be different as per user requirement. The query incoherency bound can be in form of dollar in case of stock data or any other financial information. The incoherency bound for network monitoring will be time. In network monitoring the data sent from sever to client is monitored and then proper decisions are taken on them. Suppose in data transmission some packets contain harmful data then that packets can be filtered and then only the transmission is done. Suppose some clients are accessing unauthorized sites though they are not allowed in particular organization then this type of monitoring can also be done by using network monitoring. Network monitoring is useful to detect denial of service attack and intrusions. In the system the data aggregators are kept in network. Data source will be the information of packet transfer through systems. Data aggregators will contain particular information at particular incoherency bound. Suppose there is protocol data. The protocols are TCP (T), UDP (U), SMTP (S), FTP (F) etc. Let data aggregator 1 will contain set as (T, U, S), aggregator 2 will contain data as (T, S, F), data aggregator 3 will contain (U, S, F), data aggregator 4 will contain (T, U, F). When particular client will request for a data then the data is checked in all the data aggregators and the aggregator which satisfies the incoherency bound of a client query that data aggregator will serve the query. In system the client gives the query, the query is divided among the sub-queries. The sub-queries are executed at judiciously chosen data aggregator. The query incoherency bound is divided among sub-queries. The simulated annealing algorithm is used for optimization and execution of client query.

II. LITERATURE SURVEY

R. Gupta et.al [1] has been given greedy heuristic for optimization and execution of a client query. They have compared algorithms for optimization and proved that greedy heuristic is optimal one.
Network monitoring application for denial of service attack and traffic monitoring has been proposed with different types of queries [2].
DNS based load distribution system has been given, in this monitoring system transmits the data center load to its top level DNS resolver to direct traffic so overload will be avoided. In Akamai technology server delivers static and dynamic contents over HTTP and HTTPS [3].
S. Shah et.al [4] have been given client assignment techniques such as min cost flows and stable marriages. Experiment performed by continuously polling http://www.finance.yahoo.com. Experiment was performed with 1000 traces of a day.
Greedy heuristic and simulated annealing algorithm is used for optimization. They have given that simulated annealing algorithm derives best solution and gives best dissemination scheme over all the other algorithms [5].

The different types of examples of aggregation queries have been given. The dynamic content distribution of data aggregators is also given by the author [6].
Technique for building and maintaining the aggregation trees is given. The important resources such as number of messages between the nodes of the network and total energy consumption are minimized.
Aggregation query have given the following example:
SELECT AggrFun(s.value) FROM Sensors s WHERE inclusionConditions(s) = true SAMPLE PERIOD e FOR t.
To denote epoch duration (e) and the lifetime of a query (t) the syntax of TinyOS is borrowed. The predicate *inclusionConditions* has been added in order to specify which sensor nodes will participate in the query evaluation per epoch. At each query epoch, all the sensor nodes that include their collected data in the query result are termed in our framework as *epoch participating nodes*. For queries that wish to collect data from all the sensor nodes at each epoch, the above predicate always evaluates to *true*.

III. METHODOLOGY

The client query is divided among sub-queries. The sub-queries are then given to the data aggregator(s) which contains maximum part of that query. The client query can be satisfied by one or more data aggregators. Data aggregators will contain particular information at particular incoherency bound. Suppose there is network data containing protocol information. The protocols are TCP (T), UDP (U), SMTP (S), FTP (F) etc. Let data aggregator 1 will contain set as (T, U, S), aggregator 2 will contain data as (T, S, F), data aggregator 3 will contain (U, S, F), data aggregator 4 will contain (T, U, F). Suppose client is requesting for number of TCP packets are greater than 250. The central node will contain simulated annealing algorithm and the client query. The central node will take decision where to execute client query by considering client coherency bound which is along with query. The data aggregator(s) which satisfies the client incoherency requirement executes the client query. If the client incoherency bound is satisfied by the data aggregator1 then that particular data item(s) will be disseminated by that particular data aggregator. The architecture of the system is as given below:

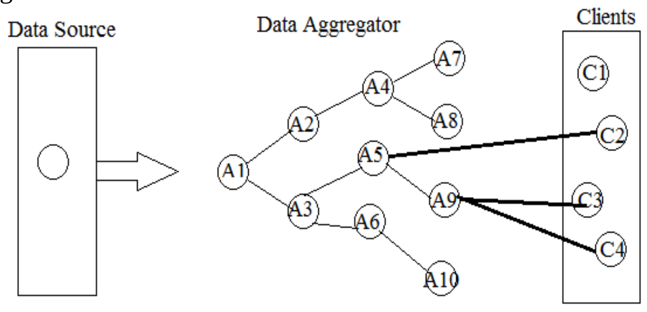

Fig.1. Data Dissemination architecture for various clients from network of data aggregators

As shown in above architectural diagram, the data source is the data packet capture. Data aggregator contains the database of the packet information. Client will give the query, which will be execute and optimized by using simulated annealing algorithm.

Simulated annealing algorithm will be used for the optimization and execution of a client query. The algorithm is as follows:
Simulated annealing is a natural process a material is heated and slowly cooled under controlled conditions to increase the size of the crystals in the material and reduce their defects. This has the effect of improving the strength and durability of the material. The heat increases the energy of the atoms allowing them to move freely, and the slow cooling schedule allows a new low-energy configuration to be discovered and exploited.

```
Input: ProblemSize, $iterations_{max}$, $temp_{max}$

Output: $S_{best}$
Begin
$S_{current} \leftarrow$ CreateInitialSolution(ProblemSize);
 $S_{best} \leftarrow S_{current}$;
 for i = 1 to $iterations_{max}$ do
 $S_i \leftarrow$ CreateNeighborSolution($S_{current}$);
 $temp_{curr} \leftarrow$ CalculateTemperature(i, $temp_{max}$);
 if Cost($S_i$) $\leq$ Cost($S_{current}$) then
 $S_{current} \leftarrow S_i$;
 if Cost($S_i$) $\leq$ Cost($S_{best}$) then
 $S_{best} \leftarrow S_i$;
 end
else if Exp(( Cost($S_{current}$ )−Cost($S_i$ )/$temp_{curr}$ ) > Rand() then
 $S_{current} \leftarrow S_i$;
 ends
 end
 return $S_{best}$;
End
```

The systems connected in LAN are the data nodes/aggregators i.e. problem size. When the algorithm is applied on the data aggregators then it gives the set of optimal solutions.
By using this simulated annealing algorithm the query will be optimized. The algorithm gives the various set of data aggregators. Among various data aggregator sets the optimal one is selected. Example of this can be given as below:
Suppose there are two data aggregators, DA1 and DA2 and the data items are $d_1 – d_4$.
DA1: (d_1, 0.2) (d_3, 0.1)
DA2: (d_1,1.0) (d_2,0.2)(d_4,0.3)
Data aggregator DA1 can serve the data items equal to greater than 0.2. Suppose a single data aggregator is disseminating the results then the data items are composited to client query (C_q : 50 d_1+150 d_2+150 d_3) and disseminated results to client so as the query incoherency bound is not violated. Different data aggregators disseminate different sub-sets of data items, no data aggregator may have all data items required to execute the client query which is as in example.
In some cases one data aggregator cannot disseminate the result for client query then that query is executed by using the multiple data aggregators.
Suppose, consider a client query C_q: 50 d_1+150 d_2+150 d_3 with incoherency bound of 90.
We want to execute this query on data aggregator DA1 and DA2 to minimize the number of refreshes. Suppose query is divided in number of sub queries then query plan will be given as follows:

Plan1: DA1{50 d_1+150 d_3}; DA2{ d_2}
Plan2:DA1{ d_3}; DA2{50 d_1+150 d_2}

From these two query plans one plan is selected which is optimal one.
Pull based data dissemination and push based data dissemination these are two techniques for the dissemination of the results. In pull based client have to demand for data updates time to time. Push based technique disseminates data itself. Whenever new data comes, the incoherency bound of that data is checked and the results are disseminated to the client.
The system flow diagram is as given below:

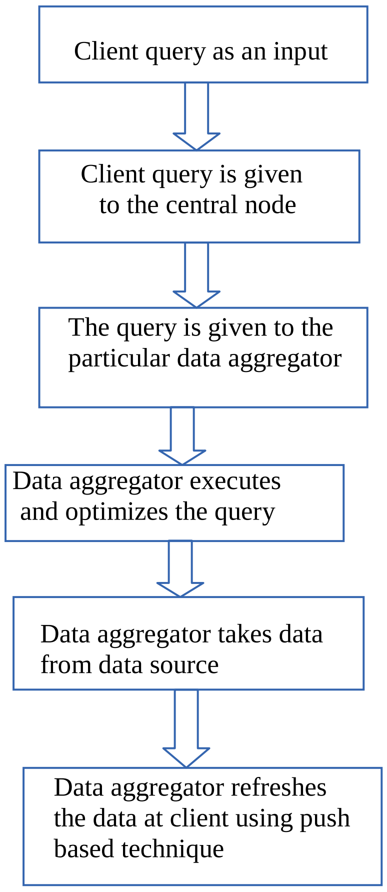
Client query as an input
Client query is given to the central node
The query is given to the particular data aggregator
Data aggregator executes and optimizes the query
Data aggregator takes data from data source
Data aggregator refreshes the data at client using push based technique

Fig.2. System Flow

The data dissemination models can be explained as below:
1. Pull model: In pull model user requests for particular data. The data aggregator requests the data to data source and then the results are disseminated to the client. The architecture of pull model is as given below:

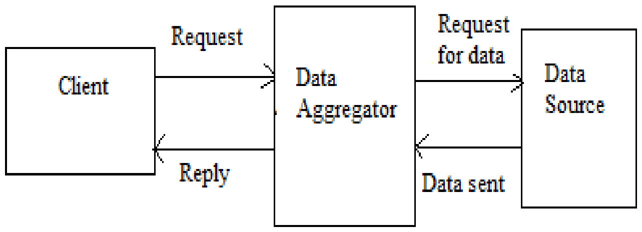

Fig.3. Pull based data dissemination

2. Push model:
In push based data dissemination model as soon as the data is available to the data aggregator it disseminates the results. While disseminating the results the data aggregaor checks the incoherency bound of the client query.

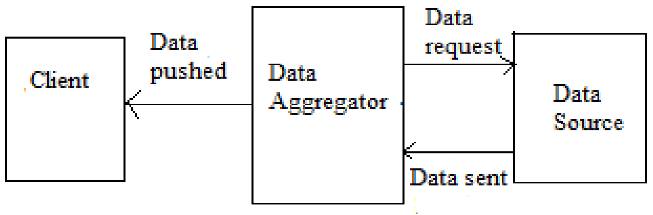

Fig.4. Push based data dissemination

IV. EXPERIMENTATION

Multiple systems connected in LAN are used as data aggregators each having database and xampp server. Xampp server itself contains the database in mysql and apache server. The packet capturer is a data source. The database involves all the information related to data packets such as source address, destination address, source port, destination port, size and time. The data packets are identified by the identifier such as source address and destination address. Client gives the query to ther central node which is in sql format given through xampp. Central node decides where to execute the client query. Central node contains the client query and simulated annealing algorithm. Simulated annealing algorithm selects the aggregators randomly and chooses the optimal set of data aggregators. The client query executed on the optimal set of data aggregators. The output will return to the central node. Central disseminates the results to the client.
Suppose we have set of nodes as 192.168.0.1, 192.168.0.2, 192.168.0.3, 192.168.0.4, 192.168.0.5, 192.168.0.6.
The simulated annealing algorithm will select the random nodes/data aggregators and will check that the solution is optimal or not. The iterations are made till the optimal solution and the optimal query plan is considered as a solution set. The database contains the dynamic information of the data packets. If the solution is optimal one then that set of data aggregators is selected and the query is executed on that set. Central node takes client query processes that query and executes that query on data aggregators and sends results to the client with the help of data source.

References

[1] R. Gupta, K. Ramamritham, "Query Planning for Continuous Aggregation Queries over a Network of Data Aggregators", *IEEE 2012 Transaction on Knowledge and Data Mining*, Volume 24, Issue:6

[2] C. Olston, J. Jiang and J. Widom,"Adaptive filters for continuous Queries over Distributed Data Streams", *SIGMOD* 2003

[3] J. Dilley, B. Maggs, J. Parikh, H. Prokop, R. Sitaraman and B. Weihl, "Globally Distributed Content Delivery", *IEEE Internet Computing* Sept 2002.

[4] S. Shah, K. Ramamritham, C. Ravishankar, "Client Assignment in Content Dissemination Networks for Dynamic Data", *VLDB* 2005

[5] Y. Zhou, B. Chin Ooi and Kian-Lean Tan,"Disseminating Streaming Data in a Dynamic Environment: An Adaptive and Cost Based Approach", The VLDB Journal, Issue 17, pg. 1465-1483,2008

[6] Krihi Ramamritham, "Maintaining Coherent Views over Dynamic Distributed Data", Computer Science and Engineering Department.

[7] Antonios Deligiannakis, Yannis Kotidis,Vassilis Stoumpos and Alex Delis, "Building Efficient Aggregation Trees for Sensor Networks' Event-Monitoring Queries".

Ms. Manisha Babanrao Thombare is post graduate student of computer engineering at MET Bhujbal Knowledge City, Nasik under University of Pune.

K.V. Metre, is working at MET's IOE, Nashik, Maharashtra, India as an Assistant Professor. She completed BE from VNIT, Nagpur and post graduation in Computer Engineering from Dr. Babasaheb Ambedkar Technological University, Lonere. She is persuing Ph.D. from RTM Nagpur University. She has presented papers at National and International conferences.Her areas of interest include Database, Operating System.