A Compressed Technique to Optimize the Processing of Real Time Data in Centralized and Client-Server Data Base Systems

Amit Sinha
Associate Professor, Department of IT
ABES Engineering College, Ghaziabad
Ghaziabad, India

Rajendra Kr. Isaac
Professor, Faculty of Engg. and Technology
Sam Higginbottom Institute of Agriculture,
Technology & Sciences, Allahabad, India

ABSTRACT
The processing of real time transaction is an important issue in present scenario. It depends on various constraints of architecture such as data size, bandwidth speed, network latency and machine configuration etc. The present paper proposes a mathematical model to improve in the response time in centralized and client-server paradigm. The work also shows the improvement through simulation done in MatLabR2014.

Keywords
Real-time systems, Compressed Technique, Client-Server Data

1. INTRODUCTION
Real-time systems can be defined as those computing systems that are designed to operate in a timely manner. That is, performing certain actions within specific timing constraints; e.g., producing results while meeting predefined deadlines. The notion of correctness of a real-time system is contingent upon the logical correctness of the produced results as well as the timing at which such results are produced. The real time transactions require a database. Many real time database applications are distributed in nature. The real time performance of Real time distributed database system (RTDBS) depends on several factors like database system architecture, disk speed and the underlying processor etc. The proposed model can be used to study the transaction atomicity for real time database system with variety of workloads parameters in a centralized and client-server environment. The work mainly concentrates on the scheduling arrival rate of the workloads applied to the transaction deadline to measure the transaction performance.

The processing of Real time transaction (RTT) can be observed through its dependants. Few of the dependants rely on network features while others depend on architectural configuration.

The present work possesses following hypothesis:

(i) The processing of RTDB depends on various constraints of machine.

(ii) The processing time of RTDB may be reduced by compressing the data sent.

2. LITERATURE REVIEW
Nandanwar et al (2013) said that time constraint was the main factor in real time operating system and it affected the deadline of the process. To achieve deadline, proper scheduling algorithm is required to schedule the task. In this paper an Adaptive scheduling algorithm was developed which was the combination of Earliest Deadline First (EDF) and Ant Colony Optimization (ACO). The result of execution time is compared with the EDF and ACO scheduling algorithm. The goal of an Adaptive scheduling algorithm is to show the switching between the scheduling algorithms and to decrease the system failure and increase the system performance.

Kanitkar et al (1998) worked on various issues affecting client server real time data base system (RTDBS). Here, they observed that there is scope for replication in data-shipping client-server DBMSs offering opportunities for improved transaction response times. To support this replication, they described a two-stage protocol for transaction processing (2STP). They extended the conventional client-server data-shipping mechanism by allowing clients to update and query cached objects that have replicas in multiple sites. The effectiveness of the two-stage transaction processing mechanism is supported by means of queuing analysis and detailed simulation experiments comparing 2STP with a global lock-based data-shipping protocol. This improvement in transaction processing efficiency is achieved at the cost of longer downtimes for crash recovery.

Park et al (1999) worked on client server databases partitioning the clients into logical clusters. In conventional two-tier client-server databases, clients access and modify shared data resident in a common server. As the number of clients increases, the centralized database server can become a performance bottleneck. In order to overcome this scalability problem, a three-tier client-server configuration has been proposed that features the partitioning of clients into logical clusters. Here, the objective is to maximize the data sharing among the clients in each cluster. They proposed a genetic algorithm to create such client clusters and evaluate two different techniques for generating the initial solution populations. They compare the performance of the two-tier and three-tier configurations with respect to the transaction turnaround times and object response times. Their experimental results indicate that the clustered architecture can offer improved performance over its two-tier counterpart.

Kanitkar et al (2005) said that in the traditional client-server databases, a transaction and its requisite data have to be collocated at a single site for the operation to proceed. This has usually been achieved by moving either the data or the transaction. However, the availability of high-bandwidth network options has led users of today's systems to expect real-time guarantees about the completion time of their tasks. In order to offer such guarantees in a client-server database system, a transaction should be processed by any means that allows it to meet its deadline. At the end, they explored the option of moving both transactions and data to the most promising sites for successful completion. The suitability of a client for processing a transaction is measured with respect to

the availability of the transaction's required data in its local cache.

3. RTT DEPENDANTS

The present work proposes a compressed model to optimize the real time transaction (RTT). The processing time depends on several constraints such as speed of bandwidth, amount of data to be transferred, RAM size, configuration / architecture of clients & server, level of problems i.e. priority considerations, latency, cycle per instruction, number of instructions in a given program and cycle time etc. These constraints are the materials for the developed model. The existing mathematical model to calculate response time in client-server architecture is

$$T(CS) = p1((C_{req} + C_{res})T1 + T(p)) + p2\,(1-p1)((C_{req} + C_{NF})T1 + (C_{req} + C_{res})T2 + T(p))\ldots.+ (pn\,(1-p1)(1-p2)\ldots(1-pn))\,(C_{req} + C_{NF})\,(T1 + T2 + \ldots + T(n-1)) + (C_{req} + C_{res})Tn \quad \textit{eq.(1)}$$

Where

T(CS) is the response time, P(i) is the probability of fulfilling the request at server i,

C_{req} is the size of the data requested for processing, C_{res} is the size of the data responded after processing, C_{NF} is the message to respective client for not finding the service at that server, Ti is the time to carry the data with the server I, T(p) is the time to process the data.

4. OBJECTIVE

The present work has following objectives:

(i) To evaluate the use of client server database for real time processing.

(ii) To study the various constraints affecting the processing time.

(iii) To develop an algorithm to optimize transaction time.

5. METHODOLOGY

The client data is sent to the server and processing time is calculated. The response time of real time transaction depends on several constraints. The proposed methodology requires a 'client-server' application running on different machines with different architecture. The present work proposes a compressed mathematical model as shown

Response Time

$$T(CS)_{NEW} = p1((\sigma C_{req} + \sigma C_{res})T1 + T(p)) + p2\,(1-p1)((\sigma C_{req} + C_{NF})T1 + (\sigma C_{req} + \sigma C_{res})T2 + (pn\,(1-p1)(1-p2)\ldots(1-pn))\,(\sigma C_{req} + C_{NF})\,(T1 + T2 + \ldots + T(n-1)) + (\sigma C_{req} + \sigma C_{res})Tn \quad \textit{eq. (2)}$$

Where $T(CS)_{NEW}$ is the new response time and σ is the compression ratio.

Here the compression methods are assumed to be the best and latest.

6. EXPERIMENTAL SET-UP AND IMPLEMENTATION

The work requires an application to send the data and calculate the response time. The evaluation is done through simulation and implemented in MatLabR2014. The proposed mathematical model is implemented in two different cases with a comparison with the existing model. The response time in each case is measured. These cases are:

(i) Data with different sizes sent on same machine.

(ii) A definite size of the data sent on different bandwidth speeds.

Case: 1

The model is tested with the following data structures:

Machine configuration: Client side: I4 with 8GB RAM, Server side: I4 with 16GB RAM

Compression ratio is 40% (σ = 0.40) and assuming the data is processed at first client.

The sample result is shown in Table-I:

Table I: Comparison of response time w.r.t. Data Sze

Data Size (KB)	T(CS) (ms)	Data Size with σ = 0.40 (KB)	$T(CS)_{NEW}$ (ms)	Approx. % Improvement
100	20	60	17	17.6%
150	21	95	18	16.7%
165	20	100	19	5.3%
188	21	119	19	10.5%
223	23	151	19	21.1%
252	23	188	20	15%
268	25	192	21	19%

Similar improvements were observed when data were found at second and subsequent servers.

Case:2

The model is tested with fixed data size 100 KB and different bandwidth speed. The following observations are made (a sample snap-shot):

Table II: Comparison of response time w.r.t. Bandwidth

Bandwidth speed (MB)	T(CS) (ms)	$T(CS)_{NEW}$ (ms) with compressed data size (60 KB)	Approx. % Improvement
10	12	10	20%
15	10	9	11.1%
25	10	9	11.1%
40	8	7	14.3%
55	7	6	16.7%
60	7	5	40%
70	5	4	25%

Similar improvements were observed when data were found at second and subsequent servers.

The above findings show that a significant improvement is found when a compressed data is sent to servers.

7. CONCLUSION

The processing of real time transaction depends on various constraints. The data size which may be the query or information is an important variable for each constraint. The existing method uses the values of these constraints. A method is proposed in this paper that compressed the data first before sending to server and to the client after processing. A lesser response time is observed through this methodology. The proposed mathematical is tested in two different cases viz. varying data sizes to same machine and fixed data size with different machines. The improvement in response time increases in both centralized and client-server architecture with more than one servers.

8. REFERENCES

[1] Miyazawa, Masanori, Hayashi, Michiaki (2014), In-network real-time performance monitoring with distributed event processing, Network Operations and Management Symposium (NOMS), IEEE, Vol. 25, pg. 1 – 5.

[2] Parka H. Joon , Patrick H. Kimb, Michael Marsicoa, , Naim Rasheeda (2014), Data Mining Strategies for Real-Time Control in New York City, The 5th International Conference on Ambient Systems, Networks and Technologies, ScienceDirect, Vol. 39, pg. 225-237.

[3] Ozgur Ulusoy (2014), Transaction processing in distributed active real-time database systems, Journal of Systems and Software, ScienceDirect, Vol. 42, pg. 247–262.

[4] Haque, W. ; Toms, A. ; Germuth, A, (2013), Dynamic Load Balancing in Real-Time Distributed Transaction Processing, Computational Science and Engineering, IEEE, Vol. 25, pg. 268 – 274.

[5] Muthukumar P., Suresh P., Shalini Punithavathani S., Nafeesa Begum J., (2012), A realistic approach for the deployment of national knowledge repositories by leveraging ETL tools, Recent Trends In Information Technology, IEEE, Vol. 35, pg. 542 – 547.

[6] Doshi P, Raisinghani V. (2011), Review of dynamic query optimization strategies in distributed database, Electronics Computer Technology (ICECT), IEEE, Vol. 6, pg. 145 – 149.

[7] Tekin C., Zhang S., van der Schaar, (2014), Distributed Online Learning in Social Recommender Systems, IEEE, Vol. 99, Pg. 1-10.

[8] Huang Jiewen, Venkatraman Kartik, Abadi Daniel J., (2014), Query optimization of distributed pattern matching, Data Engineering, IEEE,, Vol. 30, pg. 64 – 75.

[9] Tandon, A., Motani, M. Varshney, L.R., (2014), On code design for simultaneous energy and information transfer, Information Theory and Applications Workshop, IEEE, Vol. 23, pg. 1 – 6.