Transaction Management Policy in Distributed Real Time System

Bandaru Vishnu Roopini, Akkireddy Lakshmi Harika, Kurra Manasa Devi, Jaladi Raja Sree

Abstract— Managing the transactions in real time distributed computing system is not easy, as it has heterogeneously networked computers to solve a single problem. If a transaction runs across some different sites, it may commit at some sites and may failure at another site, leading to an inconsistent transaction. The complexity is increase in real time applications by placing deadlines on the response time of the database system and transactions processing. Such a system needs to process transactions before these deadlines expired. A series of simulation study have been performed to analyze the performance under different transaction management under conditions such as different workloads, distribution methods, execution mode-distribution and parallel etc. The scheduling of data accesses are done in order to meet their deadlines and to minimize the number of transactions that missed deadlines. A new concept is introduced to manage the transactions in database size for originating site and remote site rather than database size computing parameters. With this approach, the system gives a significant improvement in performance.

Index Terms— Digital Circuits, Real time system, transaction management, missed deadlines, database size.

I. INTRODUCTION

Real-Time Database Systems are becoming increasingly important in a wide range of applications, such as telecommunications, mobile communication systems, nuclear reactor control, traffic control systems, computer integrated manufacturing, robotics and military systems. A Real-Time Database System (RTDBS) is a transaction processing system that is designed to handle workloads where transactions have deadlines. The objective of the system is to meet these deadlines. As the world gets smarter and more informatics, demands on IT will grow. Many converging technologies are coming up like emerging IT delivery model-cloud computing. Demands of the real time distributed database are also increasing.

Many transaction complexities are there in handling concurrency control and database recovery in distributed database systems. Two-phase commit protocol most widely used to solve these problems.

To ensure such transaction atomicity, commit protocols are implemented in distributed database system. A uniform commitment is guarantee by a commit protocol in a distributed transaction execution to ensure that all the participating sites agree on a final outcome.

Manuscript received May, 2013.

B.Vishnu Roopini, Electronics and Computer Engineering, Guntur Dt, Andhra Pradesh, India.

A.L.Harika, Electronics and Computer Engineering, Guntur Dt, Andhra Pradesh, India.

K.Manasa Devi, Computer Science Engineering, Guntur Dt, Andhra Pradesh,India.

J.Rajasri, Computer Science Engineering, Guntur Dt, Andhra Pradesh, India.

Result may be either a commit or an abort condition. Many real time database applications in areas of communication system and military systems are distributed in nature. In a real time database system the transaction processing system that is designed to handle workloads where transactions have complete deadlines. To ensure transaction atomicity, commit protocol are implemented in distributed database system. Experimental performances of transaction scheduling under variety of workloads and different system configuration are to be evaluated through proposed model. Many database researchers have proposed varieties of commit protocols like two phase commit and Nested two phase commit, Presumed commit and Presume abort, Broadcast Two phase commit , Three phase commit etc. These require exchanges of multiple messages, in multiple phases, between the participating sites where the distributed transaction
executed. Several log records are generated to make permanent changed to the data disk, demanding some more transaction execution time. Proper scheduling of transactions and management of its execution time are important factors in designing such systems.

Transactions processing in any database systems can have real time constraints. The scheduling transactions with deadlines on a single processor memory resident database system have been developed and to be evaluated the scheduling through proposed model. A real time database system is a Transaction processing system that designed to handle workloads where transactions have complete deadlines. In case of faults, it is not possible to provide such guarantee. Real actions such as firing a weapon or dispensing cash may not be compostable at all. Proper scheduling of transactions and management of its execution time are the important factors in designing such systems. In such a database, the performance of the commit protocol is usually measured in terms of number of transactions that complete before their deadlines. The transaction that miss their deadlines before the completion of processing are just killed or aborted and discarded from the system without being executed to completion.

II. TRANSACTION DETAILS

This study is in continuation of work in the same domain. The study follows the real time processing model and transaction processing addressing timeliness. This model has six components:

(i) a source
(ii) a transaction manager
(iii) a concurrency control manager
(iv) a resource manager
(v) a recovery manager
(vi) a sink to collects statistics on the completed transactions.

A network manager models the

behavior of the communications network. The definitions of the components of the model are given below.

A. The source

This component is responsible for generating the workloads for a site. The workloads are characterized in terms of files that they access and number of pages that they access and also update of a file.

B. The transaction manager

The transaction manager is responsible for accepting transaction from the source and modeling their execution. This deals with the execution behavior of the transaction. Each transaction in the workload has a general structure consist of a master process and a number of cohorts. The master resides at the sites where the transaction was submitted. Each cohort makes a sequence of read and writes requests to files that are stored at its sites. A transaction has one cohort at each site where it needs to access data. To choose the execution sites for a transaction's cohorts, the decision rule is: if a file is present at the originating site, use the copy there; otherwise, choose uniformly from among the sites that have remote copies of the files. The transaction manager also models the details of the commit and abort protocols.

C. The concurrency control manager

It deals with the implementation of the concurrency control algorithms. In this study, this module is not fully implemented. The effect of this is dependent on algorithm that chooses during designing the system.

D. The resource manager

The resource manager models the physical resources like CPU, Disk, and files etc for writing to or accessing data or messages from them.

E. The sink

The sink deals for collection of statistics on the completed transactions.

F. The Network Manager

The network manager encapsulates the model of the communications network. It is assuming a local area network system, where the actual time on the wire for messages is negligible.

III. TRANSACTION MODEL AND THEIR PARAMETER

The proposed model is discussed below. A common model of a distributed transaction is that there is one process, called as Master, which is executed at the site where the transaction is submitted, and a set of processes, called Cohorts, which executes on behalf of the transaction at these various sites that are accessed by the transaction. In other words, each transaction has a master process that runs at its site of origination. The master process in turn sets up a collection of cohort's processes to perform the actual processing involved in running the transaction. When cohort finishes executing its portion of a query, it sends an execution complete message to the master. When the master received such a message from each cohort, it starts its execution process. When a transaction is initiated, the set of files and data items that, it will access are chosen by the source. The master is then loaded at its originating site and initiates the first phase of the protocol by sending PREPARE (to commit) messages in parallel to all the cohorts. Each cohort that is ready to commit, first force-writes a prepared log record to its local stable storage and then sends a YES vote to the master. At this stage, the cohort has entered a prepared state wherein it cannot unilaterally commit or abort the transaction but has to wait for final decision from the master. On other hand, each cohort that decides to abort force-writes an abort log record and sends a NO vote to the master. Since a NO vote acts like a veto, cohort is permitted unilaterally abort the transaction without waiting for a response from the master.

After the master receives the votes from all the cohorts, it initiates the second phase of the protocol. If all the votes are YES, it moves to a committing state by force-writing a commit log record and sending COMMIT messages to all the cohorts. Each cohort after receiving a COMMIT message moves to the committing state, force-writes a commit log record, and sends an acknowledgement (ACK) message to the master. If the master receives even one NO vote, it moves to the aborting state by force writing an abort log record and sends ABORT messages to those cohorts that are in the prepared state. These cohorts, after receiving the ABORT message, move to aborting state, force-write an abort log record and send an ACK message to the master. Finally, the master, after receiving acknowledgement from all the prepared cohorts, writes an end log record and then forgets and made free the transaction. The statistics are collected in the Sink. The database is modeled as a collection of DBsize pages that are uniformly distributed across all the NumSites sites. At each site, transactions arrive under Poisson stream with rate Arrival Rate and each.

Transaction has an associated firm deadline. The deadline is assigned using the formula

$$DT=AT+SF*RT$$

Here DT, AT, SF and RT are the deadline, arrival rate, Slack factor and resource time respectively, of transaction T. The Resource time is the total service time at the resources that the transaction requires for its execution. The Slack factor is a constant that provides control over the tightness or slackness of the transaction deadlines.

In this model, each of the transaction in the supplied workload has the structure of the single master and multiple cohorts. The number of sites at which each transaction executes is specifying by the File selection time (DistDegree) parameter. At each of the execution sites, the number of pages accessed by the transaction's cohort varies uniformly between 0.5 and 1.5 times Cohort Size. These pages are chosen randomly from among the database pages located at that site. A page that is read is updated with probability of Write rob. Summary of the simulation parameter is given in table I.

IV. PARAMETER SETTINGS

The values of the parameter set in the simulation are given in table II. The CPU time to process a page is 10 milliseconds while disk access times are 20 milliseconds.

TABLE I. PROPOSED MODEL PARAMETERS

Parameters	Set Values	Parameters	Set Values
NumSites	8	File Selection Time	3
Dbsizevary	max.200 for generating site and 2200 for remote site	Page CPU	10ms
Arrival Rate	6 to 8 job/sec	Page Disk	20ms
Slack factor	4	Terminal Think	0 to 0.5 sec
Write rob	0.5		

TABLE II. ASSUMED VALUES OF PROPOSED MODEL PARAMETERS

V. ANTICPATION OF RESULTS

The experiment has to be perform using different simulation language like C++Sims, DeNet etc in reports. For this study, GPSS World can be use as a simulator. Literatures are also collected from several recent studies. The study for performance evaluation starts by first developing a base model. Further experiments were constructed around the base model experiments by varying a few parameters and process of execution at a time.

The performance metric of the experiments is Miss Percent that is the percentage of input transaction that the system is unable to complete before their deadline. A study can be analyze the performance of the system under different workload with varying the arrival rate of the transaction, dynamic slack factors, execution mode etc. A study can be analyzed the performance using this new concept of varying database size for generating site and remote site. The anticipated experimental results are discussed below.

A. Comparison of Centralized and Distributed systems

This anticipated experiment compares the performance of the system under centralized and distributed. The distributed systems have higher percentage of miss Transactions than centralized system. This higher miss percentage is due to distance between cohorts. This leads to design of a new perfect distributed commit processing protocol to have a real-time committing performance.

B. Impact of distribution methods

This anticipated experiment is to be conducted to know the impact of difference between distribution methods to the performance of the system. As an example, we take Exponential distribution and Poisson distribution. The assignment and committing of transactions to cohorts are passed under scheduler using Exponential distribution and Poisson distribution and the statistics of the simulation outputs are to be noted. The Exponential give more uniform assignment and committing of transactions than Poisson. Poisson throws higher numbers of transactions giving more collisions of transactions and large number miss percentage of transactions than Exponential. So on many experiments of such similar types can be conducted by using more different distribution rules.

Parameters	Description
NumSites or Select file	Number of sites in the Database
Dbsize_generating_site	Number of pages in the database at same location.
Dbsize_remote_site	Number of pages in the database at remote location.
Arrival Rate	Transaction arrival rate/site
Slack factor	Slack factor in Deadline formula
File Selection Time	Degree of Freedom (Dist Degree)
Write Probe	Page update probability
Page CPU	CPU page processing time
Page Disk	Disk page access time
Terminal Think	Time between completion of 1 transaction & submission of another
Num write	Number of Write Transactions
Number Read T	Number of Read Transactions

C. Impact execution mode: Distribution and Parallel

This anticipated experiment compares the output of the system putting the cohorts in parallel with that of distribution execution. From this we can conclude following points. Parallel execution of the cohorts reduces the transaction response time. The time required for the commit processing is partially reduced. This is because the queuing time is shorted in parallel and so there are much fewer chances of a cohort aborting during waiting phase.

D. Impact of slack to Throughput

In this set of experiments, the impact of slack factor to observed on the throughput of the system. The throughput initially decreases with increase in slack factor due to constraint of distributed real time database. Still there are lots more to study required about other parameters to improve the throughput of the overall system.

E. Transaction Management

The transactions can be managed in many different ways. In most of the earlier works done database size computing. A new concept is introduced to manage the transactions in database size for originating site and remote site rather than database size computing parameters, where the values of the parameters are changes or adjust automatically depending on the requirements during the execution the experiment.

VI. CONCLUSIONS

A series of simulation study have been performed to analyze the performance under different transaction management situation such as different workloads, distribution methods, execution mode-Distribution and Parallel, impact of dynamic slack factors to throughput.

The scheduling of data accesses are done in order to meet their deadlines and to minimize the number of transactions that missed deadlines.

ACKNOWLEDGMENT

We were very much thankful to our guide Sridhar sir for supporting us and our college K.L.University for providing us such a facility to research on various topics and we are grateful to IJSCE publications for recognizing our paper and our parents and friends for supporting us in each and every aspect.

REFERENCES

[1] Silberschatz, Korth, Sudarshan,2002, Database system concept,4th (I.E), McGraow-Hill Pub. 698-709,903
[2] Gray. J,1978,"Notes on Database Operating Systems", Operating Systems: An Advanced Course, Lecture notes in Computer Science
[3] Mohan, C, Lindsay B and Obermark 1986, Transaction Management in the R* Distributed Database Management Systems, ACM TODS, 11(4).
[4] Lampson B and Lomet D,1993, "A new Presumes Commit Optimization for Two phase Commit", Pro.of 19th VLDB Conference.
[5] Oszu M, Valduriez P,1991, Principles of Distributed Database Systems, Prentice-Hall.
[6] Kohler W, 1981,A survey of Techniques for Synchronization and Recovery in Decentralized Computer System, ACM Computing Surveys, 13(2)
[7] Nostrum D, Nolin M,2006, Pessimistic Concurrency Control and Versioning to Support Database Pointers in Real-Time Databases, Proc. 16th Euro micro Conf. on Real-Time Systems
[8] Ramamritham, Son S. H, and Dippy L,2004, Real-Time Databases and Data Services, Real-Time Systems J., vol. 28, 179-216.
[9] Robert A and Garcia-Molina H,1992, Scheduling Real-Time Transactions, ACM Trans. on Database Systems, 17(3).
[10] Levy E., Korth H and Silberschatz,1991,An optimistic commit protocol for distributed transaction management, Pro.of ACM SIGMOD Conf.
[11] Jayant. H, Carey M, Livney,1992, "Data Access Scheduling in Firm Real time Database Systems", Real Time systems Journal, 4(3)
[12] Udai Shanker, "Some Performance Issues In Distributed Real Time Database System", PhD Thesis, December,2005
[13] Jayant Singh and S.C Mehrotra, 2006,"Performance analysis of a Real Time Distributed Database System through simulation" 15th IASTED International Conf. on APPLIED SIMULATION & MODELLING, Greece
[14] Jayant Singh and S.C Mehrotra,2009 "A study on transaction scheduling in a real-time distributed system", EUROSIS"s Annual Industrial Simulation Conference, UK.
[15] Jayant H. 1991, "Transaction Scheduling in Firm Real-Time Database Systems", Ph.D. Thesis, Computer Science Dept. Univ. of Wisconsin, Madison.

Bandaru Vishnu Roopini, was born in 1992 at Guntur ,Andhra Pradesh.She is now pursuing B.Tech final year from K.L.University. She is interested operating systems, real time systems and database management systems.

Akkireddy Lakshmi Harika, was born in 1992 at Rajahmundry,Andhra Pradesh.Sheis now pursuing B.Tech final year from K.L.University.She is interested in realtime systems,database management system,operating systems .

Kurra.ManasaDevi, was born in 1992 in chirala, Andhra Pradesh. She is currently pursuing his B.Tech final year from K.L.University. She is interested operating systems, real time systems and database management systems

Jaladi Rajasree, was born in 1992 in Vijayawada, Andhra Pradesh. He is pursuing B.Tech final year from K.L.University. She is interested in realtime systems,database management system,operating systems.