ISTANBUL UNIVERSITY –
JOURNAL OF ELECTRICAL & ELECTRONICS ENGINEERING

YEAR : 2008 (649-655)
VOLUME : 8
NUMBER : 2

ASSESSMENT OF TRANSACTION PERFORMANCE UNDER DISTRIBUTED REAL TIME DATABASE SYSTEM

Sanasam Bimol, Kh. Manglem Singh
DOEACC Centre, Imphal (India)
bimpusana@gmail.com , manglem@gmail.com

Y. Jayanta Singh
Keane India Limited (India)
Jayanta_singh@keane.com

L.Pushparani Devi
AITECT Imphal, (India)
Laithangbam.Pushparani.Devi@gmail.com

ABSTRACT

The objective of this paper is to forecast the performance of transaction under real time distributed system. A real time database system is a transaction processing system design to handle the workload within a deadline. The objective of such scheme is to complete the processing of transaction before the deadline expires. The performance of the system depends on the factors such as database system architectures, underlying processors, disks speeds, various operating conditions and workloads. Forecasting the transaction performance depends on the basis of comparing with commit and abort of a transition in the scheme. The output of the present simulation gives the percentage of commit and abort of transaction.

Key words: *Real time system, distributed sites, transaction, deadline, distributed database.*

1. Introduction

Distributed computing is a technique that is used to solve a single problem in a heterogeneous computer network system. A major issue in building a distributed database system is the transactions atomicity. When a transaction runs across into two sites, it may happen that one site may commit and other one may fail due to an inconsistent state of transaction. Two-phase commit protocol is widely used [1,2] to solve these problems. The choice of commit protocol is an important design decision for distributed database system. A commit protocol in a distributed database transaction should uniformly commit to ensure that all the participating sites agree to the final outcome and the result may be either a commit or an abort situation. Many real time database applications are distributed in nature [2,3,4]. These include the aircraft control, stock trading, network management, factory automation etc [4,5,6]. The real time performance of Real time distributed database system (RTDBS) depends on several factors such as the database system architecture, the underlying processor, disk speed etc. Simulation model can be used to study the transaction atomicity for distributed real time database system (DRTDBS). The simulation model can be used under variety of workloads, setting and workload parameters. We mainly concentrate on the scheduling arrival rate of the workloads applied to the transaction deadline to measure the transaction performance.

The present paper is divided in four sections. Section-2 defines the concept of real time database system (RTDBS). Section-3 deals with simulation model. Section 4 - describes on formation used in proposed model and

Received Date: 08.06.2008
Accepted Date: 01.07.2008

parameter setting. The last Section-5 is about on simulation results and it is followed by conclusions in Section - 6.

2. Concept of Real Time Database System

The systems with deadlines are called as real time system (RTDBS). A transaction in a database system can have any real time constraints. A real time database system is a transaction processing system that is designed to handle workloads, where each transaction has completion deadline. The deadlines are categorized as (I) Hard deadlines: serious problem, this type of problem occurs when a task is not completed within the deadlines, (II) Firm deadlines: the task is completed after the deadlines, (III) Soft Deadlines: the task diminishes its value if the task is completed after the deadlines. Designing the real time system involved ensuring that there is enough processing power to meet the deadlines without the need of excessive hardware resources. The objective of system is to meet these deadlines, that is, to complete processing transaction before their deadlines expire. In RTDBS, the performance of the transaction commit is usually measured in terms of the numbers of transactions that complete before their deadlines. The transaction that misses the completion of processing before its deadline is just considered as killed or aborted and discarded from the system without being executed to completion [6].

Database researcher proposed variety of commit protocols such as two phase [7], Nested two-phase [7], three-phase commit [8] etc. A survey in RTDBS is in [9,10] and a detail of deadlines is discussed in [5,11,12]. Two-phase commit protocol in real-time designation has been investigated in [12,13,14,16]. The works in [5,11] concentrated in management of deadline applied to a transaction and scheduling of arrival rates of workload with experimental performance of system under variety of workloads and different methods.

3. Simulation Model

To evaluate the performance of the two-phase commit protocols, we develop a detailed simulation model of a distributed real time database system based on loose combination of the distributed database model presented in [6,15,20,21]. More details on the definitions and literature are available in [1,2, 3,5,12,14,16,17,18,19]. The proposed model consists of non-replicated manner of database distributed to all available sites, say, for example, 8 sites in our case. According to our study, we modify the model of the RTDBS from the basic model presented in [6]. The model consists of five different components as shown in Figure 1.

Source:- This component is mainly responsible for generating the transaction workload for a site. The workload model used by the source characterizes transactions in terms of the files that they access and the numbers of pages that they access and update in each file.

Transaction Manager:- The transaction manager is responsible for accepting transactions from the source and modeling their execution. Each transaction in the workloads has a master process, numbers of cohorts and possibly a number of updaters. The master resides at the site, where the transaction is submitted. Each cohort makes a sequence of read and write requests to one or more files that are stored at its sites. A transaction has one cohort at each site, where it needs to access data. To select the execution sites for a transaction's cohorts, the decision rules is: If a files is present at the originating site, use the copy there; otherwise choose uniformly from the sites that have remote copies of the file.

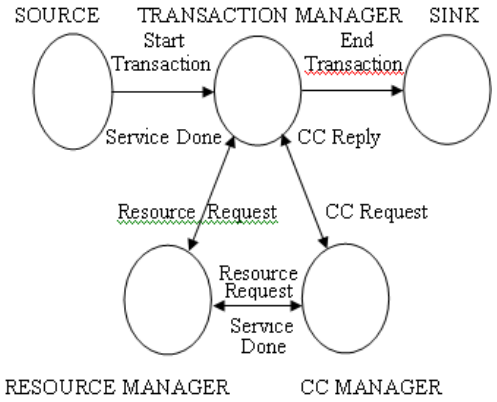

Figure 1: RTDBS Model

Resource Manager:- The resource manager manages the physical resources of sites like its CPU and its disk for reading and writing data or

message from them. It also provides the CPU and I/O service to the transaction manager and concurrency control manager. This component is not fully implemented in our work.

Sink:- The sink deals with collection of statistics for the completed transactions.

Concurrency Control Manager:- The concurrency control (CC) manager is responsible for handling concurrency control requests made by the transaction manager, including read and write access requests, requests to get permission to commit a transaction, and several types of master and cohort management requests to initialize and terminate master and cohort processes.

4. Formulation of the Proposed Model

Our work adopts the common model of distributed transaction execution. There is one process called "master" which is executed at the site, where the transaction is submitted. There are many processes called "cohorts", which are executed on behalf of the transactions at the various sites that are accessed in the transaction. In other words, each transaction has a master process that runs at its site of origination. The master process, in turns sites up a collection of cohort processes, which are involved in running the transaction. Cohorts are created by the master sending a STARTWORK massage to local transaction manager at that site. After each cohort finishes executing its portions of a query, it sends a WORKDONE massage to the master and the master initiates the execution of a process after it receives such massage from all its cohorts. When the transaction is initiated at the site of files and data items that it will access are chosen by the source, the master is, then loaded at its originating site and initiate the first phase of the protocol by sending PREPARE (to commit) massages in parallel with all its cohorts. Each cohort that is ready to commit first force-write a prepare log record to its local stable storage and then sends a vote to the master. At this stage, the cohort enters a prepared state, wherein it cannot unilaterally commit or abort the transaction, but has to wait for the final decision from the master. On the other hand, each cohort that decides to abort force writes an abort log record and sends a No vote to the master, since No vote acts like a veto, the cohort is permitted to unilaterally abort the transaction without waiting for decision from the master. After the master receives votes from all its cohorts, the second phase of the protocol is initiated. If all the votes are YES, the master moves to a committing log record and sends COMMIT message to all its cohorts. Each cohort, upon receiving the COMMIT message, moves to the committing state, forces write a commit log record, and sends an ACK (acknowledgement) message to the master. On the other hand, if the master receives even one NO vote, it moves to the aborting state, forces write an abort log record and sends ACK message to the master. Finally, the master, after receiving ACK from all the prepared cohorts, writes an end log record and then "forgets" the transaction and makes free. Then the statistics are collected in the sink. In our experiments, we consider the transactions that are executed in parallel. A single formula is used to assign deadlines to all transactions. Each transaction is assigned a deadline and its formula is given by the following equation:

$$D_T = A_T + SF \times R_T \qquad (1)$$

where D_T, A_T and R_T are its deadline, arrival time and resource time respectively of transaction T, and SF is the slack factor that provides control over the tightness and slackness of deadlines. The resource for its execution in other word is an execution time of a transaction. There are two issues related to resource time.

1. It is a function of the number of massages and force-writes, which differ from one commit protocol to another.
2. The workload generated utilizes information about transaction resource requirements in assigning deadlines.

Parameter Settings

To evaluate the performance of the commit protocols, a detailed simulation model of a distributed real time database system has been described in the previous section. The settings of workload and system parameters used in our model are shown in Table 1.

Table 1 Parameter settings

Parameters	Description	Setting values
Numsite	Number of sites in the database	8
DBsize	Number of pages in the database	2400 max.
TArival Rate	Transaction arrival rate/sites	0-8 jobs/sec.
TransType	Transaction Execution type	By default parallel
DistDegree	Degree of Distribution (Degree of transaction freedom)	3
Cohort Size	Size of Cohorts	3
WriteProbe	Page update probability	3
SlackFactor	Slack factor in Deadline	4
Page CPU	CPU page processing time	10 ms
PageDisk	Disk page access time	20 ms
Terminal Think	Time taken between completion of one transaction and submission of another	0 to 0.5 sec

Based on the transaction type, the cohorts may execute either in parallel or sequential. Each cohort makes a series of read and update access. In our model, the transaction has a single master process and multiple cohorts. The number of sites at which the transaction executed is simplified by the DistDegree parameters. At each of sites, the number of pages access by the cohorts varies uniformly between 0.5 to 1.5 times of Cohort Size. These pages are chosen randomly form among the database pages located at the sites. A page is read by the WriteProbe parameter. The CPU time to process a page is 10 milliseconds and the disk access time is 20 milliseconds. If the transaction's action deadline expires either before the completion of its local processing or before the master has written the global decision log receive, the transaction is killed and discarded.

5. Simulation Results

We conducted an extensive set of simulation experiments using the above mentioned parameters in Table 1 using simulation languages GPSS. The simulation can use different simulation languages such as C++SIM, DeNet [22,23,24] etc. Abort percentage (Abort %) and Commit percentage (Commit %) were used as measures for the performance metrics in our simulation results. Abort % is the percentage of input transactions that the system is unable to complete before their deadline and Commit % is the percentage of input transactions that the system is able to complete before their deadline. The Abort % values in the range from 0% to 20% percent are considered as "Normal Load" and above these values are called "Heavy Load". We conducted simulation under normal and heavy loads with various settings of workload parameters such as Numsites, DBsize(File size), DistDegree (File selection time), and with other corresponding parameter values.

Experiment 5.1 Comparison of Abort % and Commit % under Normal Load with 8 Files

In this experiment, we considered 8 distributed sites, with 8 files and other parameter values were kept constant. The experimental results are shown Figure 5.1. These results show that Abort %s were low and Commit %s, high. Abort %s for Sites 2, 3, 5 and 8 were above 20%.

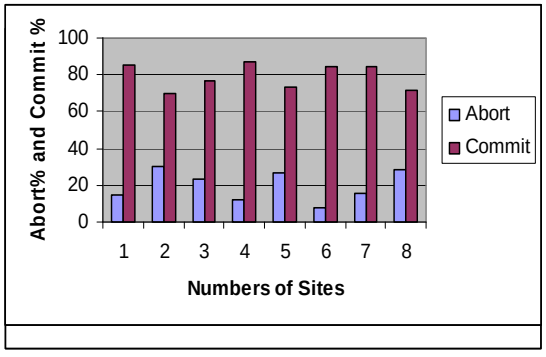

Figure 5.1 Results of individual sites under Normal Load with 8 files

Experiment 5.2 Comparison of Abort % and Commit % under Normal Load with 24 Files

In this experiment, we considered 8 distributed sites with 24 files and other parameter values were kept constant. The experimental results are shown in Figure 5.2. Sites 1, 3, 4, 5 and 8 have Abort %s above 20%. The result was for 100 transactions. Comparison of Abort %s under normal load for 100 transactions for 8 and 24 files is shown in Figure 5.3. It has been observed from the figure that Abort %s for different file sizes are almost same under normal load for 100 transactions.

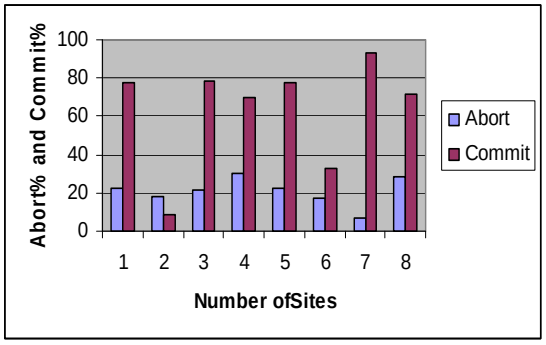

Figure 5.2 Results of individual sites under Normal Load with 24 files

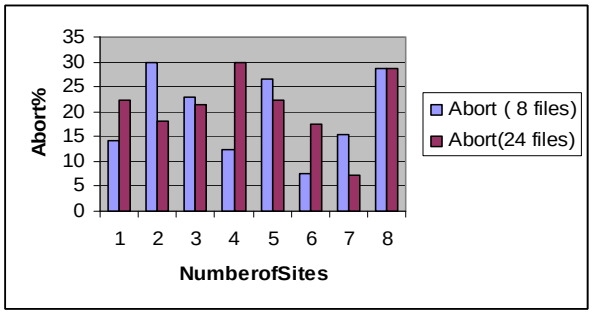

Figure 5.3 Comparison of Abort% under Normal Load for 8 files and 24 files.

Experiment 5.3 Comparison of Abort % and Commit % under Heavy Load

The goal of this experiment is to compare Abort % and Commit % for 8 sites, 24 files under heavy load of transaction for 200 transactions. Results are shown in Figure 5.4. It has been observed the figure that both Abort % and Commit % are high under heavy load.

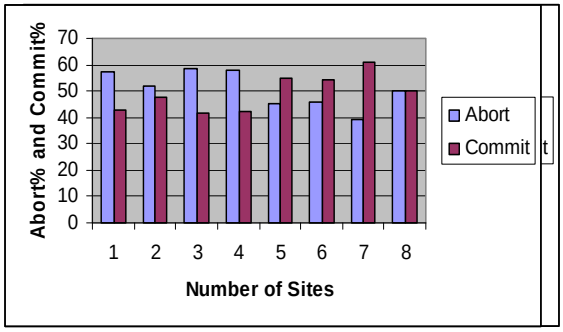

Figure 5.4 Comparison of individual sites under heavy load condition.

Experiment 5.4 Comparison of Abort % and Commit % under Normal Load by Varying Distribution Degree of the individual site

We conducted this simulation to find out Abort % and Commit % under normal load by varying distribution degree of the individual site for 8 sites with 24 files. Results are Figure 5.5. The tunable parameter in this experiment is the file selection time. These values are from 2.2 to 3, 2.4 to 3, 2.6 to 3 and from 2.8 to 3 for the sites 1, 2, 3 and 4 respectively. It has been observed that the performance improved in each site under normal load by varying file selection time. All Abort %s were less than to 20% for all sites under consideration. Comparison of the normal abort and file selection time abort are shown in Figure 5.6.

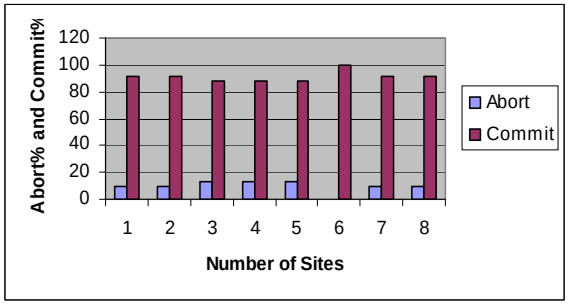

Figure 5.5 Comparative results of Abort % and Commit % under normal load by varying file selection time.

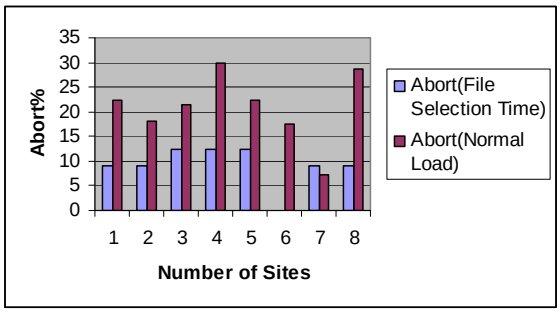

Figure: 5.6 Comparative results of Normal Abort and File Selection Time Abort

Experiment 5.5 Comparison of Abort % and Commit % with Increasing Number of Sites. This experiment was conducted to observe the improvement in the performance for higher number of sites. We considered 12 sites and 1000 transactions. Results are Figure 5.7. Abort %s for all sites were very low.

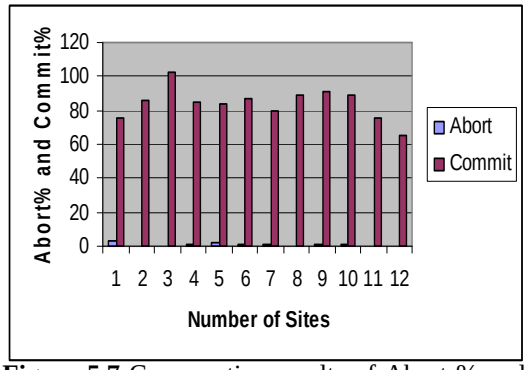

Figure 5.7 Comparative results of Abort % and Commit % with Increasing Number of Sites.

6. Conclusions

The performance of transactions was performed by comparison of transactions Abort and Commit percentages under two different workloads such as normal load and heavy load. The simulation result of 8 files under normal load showed a better performance for 100 transactions and 24 files Abort %s for different file sizes were almost same under normal load for 100 transactions. For Heavy load both Abort % and Commit % were high, it observed that could have been problem of improper distribution, like the work was assigned to a busy site. The increase in file selection time minimized the Abort percentage and gave improper performance. The performance improved in all sites under normal load. Increasing of number of sites with 1000 transaction showed that the abort percentage was very low.

Reference:

1 Silberchatz, Korth, Surdarshan (1984), "Database System Concept", *4th International Edition, McGraw Hills,. New Delhi.(India),* pp 91.

2. Bipin C. Desai (1993), "An Introduction to Database System", *Galgotia Publication Pvt. Ltd., New Delhi.(India),* 1st Edition pp520.

3. Jayant H, Ramesh G, Kriti R, and S.Seshadri (1996) " Commit Processing in Distributed Real Time Database System" *Pro of 17th IEEE Real Time System Symposium USA*, December.

4. S. Son, (1990) "Real Time Database System", *A new challenging Data Engineering*, 13 (4), December.

5. Y. Jayanta, S.C. Malhotra(2002), "Management of Atomicity problem in worst possible environment", *Library progress (International)*, 22(1), 25.

6. Jayant H, J.M.Carey and M. Livney (1992)," Data Access Scheduling in Firm Real Time Database System", *Real Time System Journal*, 4(3).

7. J. Gray (1978), " Notes on Database Operating Systems", Operating System: An advanced course, *Lecture notes in Computer Science*, 60.

8. M.Qszu, P.Valdureiz (1991), "Principles of Distributed Database System", *Prentice Hall*.

9. O. Ulusoy (1994), " Research Issues in Real Time Database system " , *Technology Report BUCEIS-94-32, Department of Computer Engineering and Information Science, Bilkent University, Turkey*.

10. O. Ulusoy,(1994) "Processing Real-time Transaction in a Replicated Database Systems", *Intel Journal of Distributed and Parallel Database* 2(4).

11. Y.Jayanta & S.C. Mehrotra (2004),"Simulation of commit processing under Distributed Real Time Database System", *IETE, National Conference, Aurangabad, India*, January.

12. Robert Abbott and Hector Garcia-Molina (1988), "Scheduling Real time

transaction: a performance Evaluation, *Programme of 14th VLDB conference Los Angeles*.

13 Y.Yoon (1994), "Transaction Scheduling and Commit Processing for Real time Distributed database System", *Ph. D thesis Korea Advance Institute of Sc. And Technology*, May.

14 H. Jayant (1991.), "Performance Analysis of Real Time Database System", *Technology Report* 92-96. University of Maryland, USA.

15. Michale J.Carey and Miron Livny (1988), "Distributed Concurrency Control Performance: A Study of Algorithms, Distribution, and Replication" *Proc. of 14th VLDB conference*, Los Angeles, California, August.

16 S.Davidson,I.Lee and V. Wolfe (1989)., "A protocol for Times Atomic Commitment", *Proc. of 9th International Conference on Distributed Computing System*".

17 Jayant, Krithi, Ramesh(2000), "The PROMPT Real Time Commit Protocol", *IEEE transaction on Parallel and distributed System*, vol 2, no 2, Feb.

18 Kenneth Reed (1998), *Data Networks* (Handbooks), IE: 61-65.

19 A.S. Tanenbaum (2001), " Computer Networks", Third Editions, *Prentice Hall of India Pvt. Ltd*.

20 Y.Jayanta & S.C. Mehrotra (2005), "A new Commit processing under Distributed Database Real time Database System", *Skyline Business Journal, Sharjah, UAE*.

21 Y.Jayanta & S.C. Mehrotra (2006), " Performance analysis of a Real Time Distributed Database System through simulation" , *15th IASTED International Conference on APPLIED SIMULATION AND MODELLING(ASM), Rhodes, Greece*, June,. http://www.actapress.com

22 Minutesmansoftware,GPSS(2001) world North Carolina ,USA,4E.[*GPSS Book*] (Student Version 4.3.5), http://www.minutmansoftware.com

23 M.C. Little and D.L. Mc Cue (1994), *C++SIM User's Guide Public release 1.5, Dept. of Computing Sc. University of New Castte Upon Tyne, U.K.*

24 Livny, M (1988), DeNet User's Guide, Version 1.0, *Comp. Sc. Dept University of Wisconsim, Madison*.

BIOGRAPHY

Sanasam Bimol obtained BSc(CS), MCM, MPM, DBM, and M.Phil(CS), form Manipur University, University of Pune, and M.K.University India) in the year 1990, 1995,1998,2000 and 2006 respectively. And presently doing MBA(IT),2008 under PT University. He was a lecturer in Department of Computer and Management in MPICMSR, Pune, University of Pune. Persently working in DOEACC Centre Impahl as a faculty. His research papers are published in International and National Journal and in conference proceedings. His area of interest are DBMS, MIS, ICT, Software Engineering and Information Security.

Kh.Manglem Singh received his B.Sc. Engg. from DEI, Agra in the year 1986, his M.E.(Control & Instrumentation) in the year 1992, M.S.(Software Systems) from BITS, Pilani in the year 1994, and his Ph.D (Digital Image Processing) from IIT, Guwahati in the year 2006. He has published thirtysix papers in National and International Journals and conference including five IEEE proceeding. He is currently working as a principal design engineer at DOEACC Centre Imphal (India).

Y. Jayanta Singh obtained his M.Sc and Ph.D in Computer Science from Dr. Babasaheb Ambedkar Marathwada University, (India) in 2000 and 2004 respectively. He is working in Keane India Ltd as Senior Software Engineer. Previously, he was a Lecturer in Department of Computer Studies in Skyline College, Sharjah, UAE.. His research papers are published in International and National Journals and in conference proceedings. His areas of interest are Networking, DBMS, ISO Process, Software Engineering and Simulation etc.

Laithangbam Pushpani Devi is working as a faculty in AITECT, Imphal(India). She obtained BSc(CS), M.Sc(CS) and M.Phil(CS) form Manipur University(India), Dr. Babasaheb Ambedkar Marathwada University, (India) and M.K.University, Madurai(India) respectively. Her research papers are published in International and National conference proceedings. Her area of interest are DBMS, Networking, ICT, and Software Engineering.