2016 International Conference on Computer Communication and Informatics (*ICCCI* -2016), Jan. 07 – 09, 2016, Coimbatore, INDIA

Concurrency Control in Distributed Database System

Qasim Abbas, Hammad Shafiq, Imran Ahmad, *|Mrs. Sridevi Tharanidharan
Department of Computer Science,
COMSATS Institute of Information and Technology, Sahiwal, Pakistan
*Amaze college of Animation and Technology – Pondicherry, India
| King Khalid University – Kingdom of Saudi Arabia.
qasim.abbas28@gmail.com, hammadshafiq1@gmail.com, imran2275@gmail.com, sreeshansujith@gmail.com

Abstract—**Concurrency manipulates the control of concurrent transaction execution. Distributed database management system enforce concurrency manipulate to make sure serializability and isolation of transaction. Lots of research has been done on this area and a number of algorithms have been purposed. In this article, we are comparing few algorithms for preserving the ACID property (atomicity, consistency, isolation, and durability) of transactions in DDBMS.**

Keywords— Distributed Database, Transactions, Locking, Serializability, ACID

I. INTRODUCTION

A database is a set of data which deals with organizational activities. DDB is a scheme that allows decentralization for the management of data through same or common language. Concurrency control deals with the issues in coordinating concurrent accesses to a database in a multi-user fashion [1]. Locking is a method used to control concurrent access to data.A lock is a variable having data items associated with it[2]. It is a status of the item and it tells about all operations that are possibly applicable on it. For each item in database one lock is available. Locks are used as a means of synchronizing the access by concurrent transactions to the database item. Some algorithms that offer concurrency manipulate are phase locking, Time stamping, Multi-model timestamp and many others[3][4][5].

II. DISTRIBUTED DATABASE

Distributed database (DDB) is a set of shared data distributed over a computer network. Data can be access through it with high speed.The system uses the medium to communicate with each other, example of mediums are telephone lines and high-speed networks. DDBconsist of sites that are loosely-coupled which does not share parts physically.

978-1-4673-6680-9/16/$31.00 ©2016 IEEE

A. TYPES OF DISTRIBUTED DATABASE

Homogeneous distributed database

- Data is distributed in it but all data center uses the same DBMS software.
- On user's request, all become agree to unite in processing.
- It provides single system look to the user.

Heterogeneous distributed database

- Data is distributed in it but all data center uses the different DBMS software.
- Different DDBMS control the different nodes under it.

III. CONCURRENCY CONTROL

Concurrency control inDB manages the actual concurrent execution associated with operations. It is done in a way that consistency of data will be maintained. An algorithm in concurrency control will be said to be honest if the very same degree of support will be supplied to every class of transaction and the service definition varies with application. The problem in concurrency control is to prevent database updates carried out by one particular user from interfering having database retrievals and also updates carried out by another user[6].It may cause several issues with the consistency with the data in case when the transactions are updating data concurrently.

To understand it more deeply let us consider an example of the reservation system in which two persons A and B try to book the same seat at a time. Fig. 3 shows that seat no.18 are available for booking. Both A and B try to reserve that seat. The problem will arise in booking. It is due to lack of concurrency control in the database of that reservation system. So, it is necessary to take measures of it.

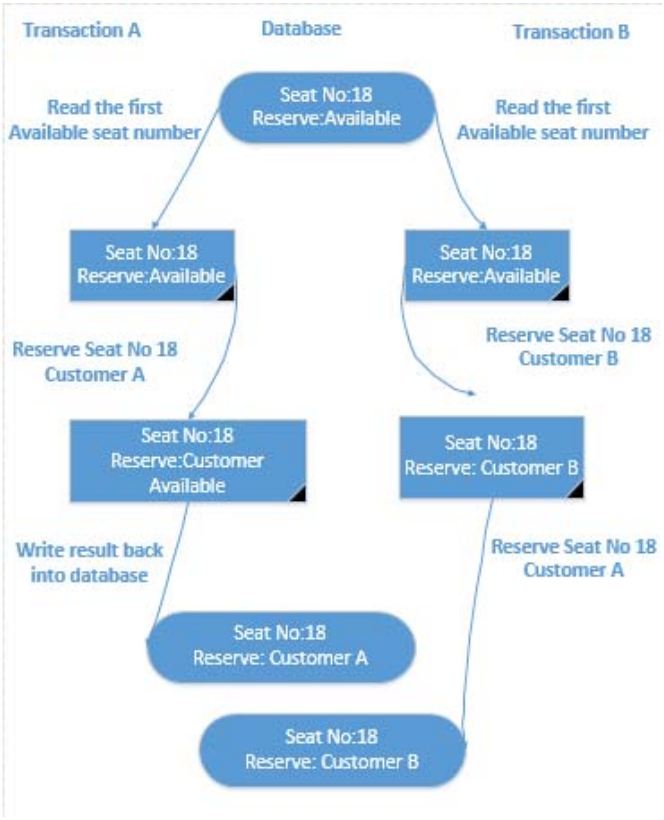

Figure 1 Example of Reservation System

IV. LOCKING

The lock is defined as a variable of a data item.It shows the status of the item regarding possible operations that can be applied on it. Each data item has associated lock with it. [7] Locks are widely used as a technique of synchronizing the access through concurrent transactions towards database item. After reading or writing data if locking or unlock data items is not used, we may get inconsistent data.Deadlock may occur in case if we do not unlock a data item before requesting a lock on another data item.

A. Distributed Concurrency Control Algorithms:

Let us summarize few concurrency control algorithms in this section. For this, we explain some important terms related to these algorithms.A transaction processing system in distributed environment maintains the ACID (atomicity, consistency, isolation and durability) property.To maintain ACID properties, it can be done by using two features:
• when a failure takes place, recoverable process log his or her actions and thus can restore earlier states.
•Commit protocol allows coordination between multiple processes to commit or abort of a transaction.

Transaction: A transaction in DBMSis a sequence of reads and writes.The four properties of a transaction are ACID properties. ACID stands for (Atomicity, Consistency, Isolation and Durability).

- Atomicity: Consider a transaction as a single unit of operation. It deals with either all tasks related to the transaction must be completed or none of the tasks is carried out.
- Consistency: Correctness of a transaction is called consistency. A transaction is correct if it obeys all integrity constraints and does not violate integrity constraints from one database state to another.
- Isolation: According to this property, a transaction which is executing should hide its data from the other concurrent transactions so that other transaction cannot read or modify.
- Durability: This property deals with the permanent storage of data once it is committed. Data should not be erased from the database on COMMIT even if system crashes or aborts.

There are two modes of locking data items:

1. Exclusive mode: Data can be both read and write in this mode.

2. Shared mode: Data item can only be read in this mode. No write operation exists in this.

B. Two-Phase Locking (2PL)

In distributed systems, 2PL is an algorithm to gain the property of distributed resources without creating the possibility for deadlock. This algorithm consists of two phases as shown in fig. 1:

Growing Phase: In this phase, a transaction can only acquire locks. It does not release any lock in growing phase. The moment it acquires the first lock it wants, the transaction enters the growing phase. Transaction keeps acquiring all the locks it would need. It cannot discharge any lock in this phase even though it offers finished dealing with a locked data item.A point reaches, where all the locks which transaction may need have been acquired. This point is Lock Point.

Shrinking Phase: When all the locks transaction has been acquired, it enters the shrinking phase. Now transaction has to release all acquired locks and cannot acquire more locks in this phase.After crossing the Lock Point, the transaction enters

the shrinking phase as soon as it releases the first lock.Transaction releases all locks in this phase[8].

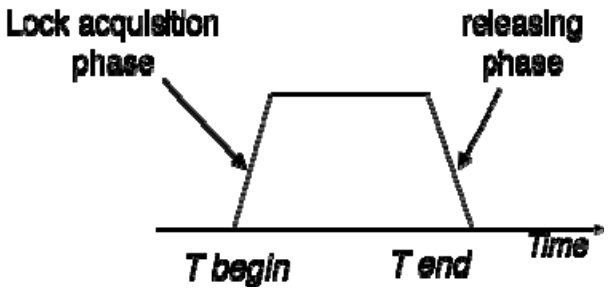

Figure 2 Two-Phase Locking

To maintain the serializability this protocol uses the law of reading any, write all. Data items are locked by setting Read locks on any copy of the item. It locks local copy whenever transaction wants to read them. When an update is needed write lock is required on all copies.Transaction converts read locks into write lock to update an item.

1. Strict Two-Phase Locking:

It also consists of two phases. The first phase is about lock acquisition, same as in 2PL[3]. But the second phase is different. Locks do not release till it reaches commit point.After committing, Strict-2PL releases all the locks at once. Fig. 2 shows the diagram of strict-2PL.

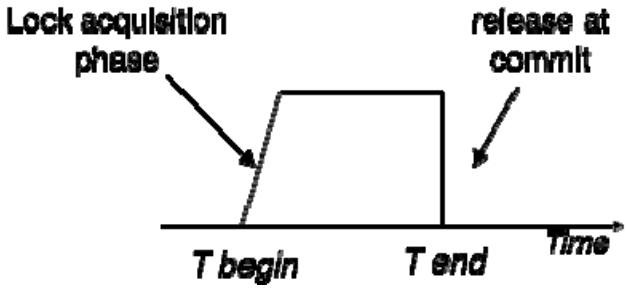

Figure 3 Strict Two-Phase locking

C. Wound-Wait and Wait-Die: In this technique, a transaction is assigned with an initial startup time. Both of these algorithms produce serializable logs and prevent deadlocks[8].

Wound-Wait is an algorithm, in which if Ti which is a requesting transaction is of high priority or older than Tj transaction, Ti wounds Tj or Ti waits otherwise. It is different from two-Phase locking algorithm in dealing with deadlocks.

Wait-die algorithm is an algorithm which is opposite to Wound-wait in such a way that if a transaction Ti is of high priority or older than transactionTj, then Ti wait for Tj or otherwise Ti transaction aborts and starts afresh [9].

D. Basic Timestamp Ordering (BTO):

Timestamp ordering (TO) is a process in which transaction is identified with its unique timestamp created by DBMS. [3]It is an order in which transactions will execute in the system so; we can say that it is the start time of a transaction.

BTO ensures the validity of timestamp order. When a transaction does not follow this order then it restarts with a new timestamp.In this timestamp algorithm, read request is allowed if the timestamp of the transaction which requests, is greater as compare to the write timestamp of the object otherwise it rejects read request. For write request if the timestamp of requesting transaction is either greater than read timestamp of the object or it is greater than Write timestamp of the object. Otherwise, it rejects write request[8].

Comparison between the algorithms:

- 2PL performs well in the centralized environment as compared to distributed environment whereas timestamp ordering performs well in both environments.
- In the case of Strict-2PL, it works as same as 2PL but its advantage is no other transaction even reads anything you write until you commit. For example, a transaction will only read committed data. It also has a disadvantage is transactions may end up in waiting. For example, insert may lock the whole table due to phantom problems.
- The disadvantage of wound wait is the construction of wait-for graph is, even more, extreme when the database is distributed and the wait-for graph must be constructed from a collection of lock tables at different sites. It also rollbacks the transaction even if there is no deadlock.

- Waiting time of Wait-Die, in which old transaction wait for young transactions to complete, may slow down.
- According to Wait-Die, younger transactions on request may die or restart. But it may conflicts with the same old transaction if it restarts with the same timestamp.
- In Wait-Die old transactions never restart. [4] In the case of Wound-Wait old transaction may restart several times.
- In BTO, the problem with Time Ordering (TO) scheduler is that it is memory expensive to maintain timestamps.
- Another disadvantage of BTO is that whenever a transaction is aborted and started again with new timestamp results in cyclic restart in which they abort without ever complete.

V. CONCLUSION & FUTURE WORK

In this article, we have discussed concurrency control in distributed database systems. We have also described and compared some of the algorithms on concurrency control like 2PL, Strict 2PL, Wound-Wait, Wait-Die and Basic Timestamp Ordering.ACID properties are very important for the transactions in the database.

In future, we will study and compare 2PLocking algorithms and will try to improve 2PLocking techniques in distributed databases.

REFERENCES

[1]Sonal Kanungo, Morena Rustom. D," Analysis and Comparison of Concurrency Control Techniques", International Journal of Advanced Research in Computer and Communication Engineering Vol. 4, Issue 3, March 2015

[2] "The Effects of Concurrency Control on the Performance of a Distributed Data Management System," Proc. 4th Berkeley Workshop on Dist. Data Mgmt. and Comp. Networks, August. 2000.

[3] Bernstein. P.. and Goodman. N.. "Concurrency Control in Distributed Database Systems". ACM Computing Surveys 13(2). June 1981.

[4] Rosenkrantz. D.. Stearns. R.. and Lewis. P . . "System Level Concurrency Control for Distributed Database Systems". ACM Transactions on Database Systems 3(2), June 1978.

[5] Swati Gupta, Kuntal Saroha, Bhawna, Fundamental Research of Distributed Database, IJCSMS International Journal of Computer Science and Management Studies, Vol. 11, Issue 02, Aug 2011

[6] "Distributed Concurrency Control Performance: A Study of Algorithm, Distribution, Replication", Comp. Sciencece. Deptt. Madison, 2002.

[7] "Locking and Deadlock Detection in Distributed Databases," Proc. 3rd Berkeley Workshop on Dist. Data Mgmt. and Camp. Networks, August, 2000.

[8] Arun Kumar Yadav& Ajay Agarwal, An Approach for Concurrency Control in of Distributed Database System, Vol. 1, No. 1, pp. 137-141, January-June (2010)

[9] "Transactions and Consistency in Distributed Database Systems," ACM Trans. on Database Sys. 7.3. Sept. 2004.