International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2019 IJSRCSEIT | Volume 5 | Issue 1 | ISSN : 2456-3307
DOI : https://doi.org/10.32628/CSEIT19517

Comparison and an Improved Validation Optimistic Approach for Concurrency Control

Monika Patel[1], Prof. Dr. Dhiren B. Patel[2]

[1]MCA Department, S. K. Patel Institute Computer Studies, Gandhinagar, Gujarat, India

[2]Department of Computer Science, Gujarat Vidhyapith, Ahmedabad, Gujarat, India

ABSTRACT

Concurrency Control entails the synchronization of accesses to the distributed database, such that the integrity of the database is maintained. Devising an efficient concurrency control technique is challenging. There is a need for improvised concurrency control technique to coordinate the simultaneous execution of transactions in a multi-processing database system. Traditionally, concurrency control techniques are locking, timestamp ordering and optimistic. These techniques have been evaluated by analytical modeling. In this undertaken work, the analytical modeling has been investigated and evaluated. The concurrency control in order to improve optimistic validation method with the traditional techniques, a new technique has been proposed.

Keywords : DBMS, DDBMS, Concurrency, Lock Maintenance, Restart Overhead, Abort Ratio, Deadlock Possibility

I. INTRODUCTION

Concurrency Control is a feature that is used to coordinate the simultaneous execution of transactions in a multi-processing database system while preserving data integrity. It is the activity to manage the actions of transactions that operate in parallel access, shared data and potentially interfere with one another. The concurrency control problem is exacerbated in a distributed DBMS (DDBMS) because users may access data stored in many different computers in a distributed system, and a concurrency control mechanism at one computer cannot instantaneously know about interactions at other computers.

A distributed database is a collection of multiple, logically interrelated databases distributed over a computer network. The distributed database management system is defined as the software system that permits the management of the DDBS to make the distribution transparent to the users.

In Concurrency Distributed Transactions, each server manages a set of objects and is responsible for ensuring that they remain consistent when accessed by concurrent transactions. Therefore, each server is responsible for applying concurrency control to its own objects. The members of distributed transactions' servers are jointly responsible for ensuring that they are performed in a serially equivalent manner.

II. RELETED WORK

Traditionally, concurrency control techniques have been divided into three categories – locking, timestamp ordering and optimistic. Concurrency control techniques have been evaluated by analytical modeling. Kung and Robinson have the first to develop the Optimistic Concurrency Control.

a. Problem Analysis and Analytic Modeling

A distributed database means multiple servers or multiple site from where data can be read or written. When same data accessed and updated by many users, in the case where the multiple copy of same data can exist. In such situation there is a possibility that data read or written wrong which creates inconsistency in the database.

Problems occur when multiple users access the same data item. It may happen - data item X can be read by Transaction T1 and written by Transaction T2 at the same time, creates conflicts in the database. When same data accessed from many terminals there may be a chance of concurrency. Concurrency may result in an inconsistent data in the database. To overcome this problem concurrency can be controlled by concurrency control – means coordination between two or more transaction to access the data. Commonly known methods of concurrency control are 2-phase locking, optimistic, and ordering.

b. 2-phase Locking

In a distributed transaction, the locks on an object are held locally. The local lock manager can decide whether to grant a lock or make the requesting transaction wait. However, it cannot release any locks until it knows that the transaction has been committed or aborted at all the servers involved in the transaction. When locking is used for concurrency control, the objects remain locked and are unavailable for other transactions during the atomic commit protocol, although an aborted transaction releases its locks after phase 1 of the protocol.

A lock manger in different servers set their locks independently of one another. It is possible that different servers may impose different ordering on transactions. Consider the following interleaving of transactions T1 and T2 at servers X and Y [8].

c. Timestamp ordering concurrency control

In a single site transaction, the lock manger issues a unique timestamp to each transaction when it starts. Serial equivalence is enforced by committing the versions of the timestamps of transactions that accessed them. In distributed transactions, we require that each manager issue global and unique timestamps. A global a unique timestamp is issued to the transaction by the first manager accessed by the transaction. The transaction timestamp is passed to the manager at each site whose data perform an operation in the transaction.

When timestamp ordering is used for concurrency control, conflicts are resolved as each operation performed. If the decision of a conflict requires the transaction to be aborted, the manger will be informed and it will abort the transaction from all the sites.

d. Optimistic Method

The optimistic concurrency control method allows all transaction to execute and validate before it will be committed. Transaction numbers are assigned at the start of the transaction which is executed in the serial manner. A distributed transaction is validated by the several independent sites, each site validate for the data used from that site.

There are four phases in the transaction execution in the optimistic concurrency control method:

1. Read: Since reading a value of an object cannot cause a loss of integrity, reads are completely

unrestricted. A transaction reads the values of a set of objects and assigns them to a set of local variables.
2. Compute: The transaction calculates a set of values for data objects called the write set. These values are assigned to a set of matching local variables. Thus, all writes after calculation take place on a transaction's copy of the objects of the database.
3. Validate: The transaction's local read set and write set are validated against a set of committed transactions.
4. Commit and Write: If the transaction succeeds in validation, it is considered committed in the system and is assigned a timestamp, otherwise the transaction is rolled back or restarted at either the compute phase or the read phase. If a transaction succeeds in the validation phase, its write set is made global and the values of the write set become values of entities in the database at each node.

III. PROPOSED WORK

Model for Concurrent Execution in Distributed Databases

The below Fig 1. Shows the basic model for concurrent execution in distributed databases. Distributed databases mean there are multiple users who are accessing data from various sites and it may happen that two or more than two users accessing the same data at the same time, at this situation problem comes.

The model shows the basic multiuser environment with concurrent transactions and it's after results. Assume n numbers of transactions are simultaneously accessing the same database and executed the numbers of operations in the one transaction. That means several updates, read, delete operations are going to be performed on the single database at time.
As soon as transaction starts its execution database program identify the table and column in which data is going to be read from or written into. Transaction value updated only when it does not affect the consistency of the database. If updating of transaction not making database inconsistent, then transaction is updated to the database and it is recorded in the successfully completed transaction table, and if updating of transaction makes database inconsistent then updated values removed from the temporary memory and it is recorded in the roll backed transaction table. Finally, all the transactions are recorded in the transaction log table.

In this work, an Improved Optimistic Validation Method has been proposed, to improve the optimistic method. In this method, all the transactions are executed without restriction and when they need to update the database, they have to pass the validation test but in the improved method read transaction does not go for validation check, so it decreases the possibilities of restart or abort the transaction.

The major problem with optimistic method was when one transaction go for validation, due to some reason it gets aborted because the resources used by this transaction is also used by any other transaction so there is the possibility of conflict. Once the transaction is aborted it gets the same timestamp and starts execution, but it can also happen that the same resources used by the other transaction and so that transaction again gets aborted.

When transaction gets aborted several times and does not get finished or does not get the chance to execute, which means the transaction is in the condition of starvation. Starvation is the situation where transaction doesn't get the fair chance to execute.

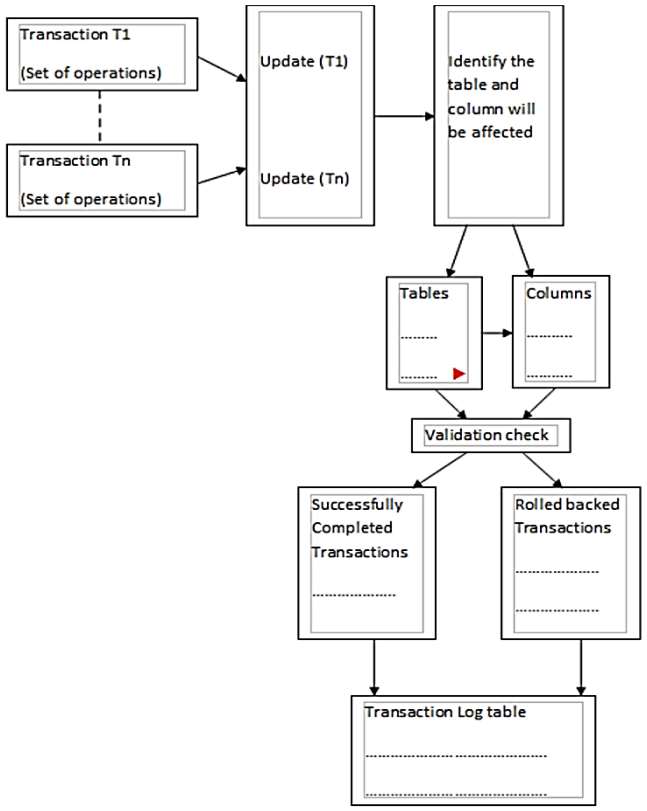

Fig-1 : Model for Concurrent Execution in distributed database

Comparison of ORDER and Optimistic Method with common criteria

The below table shows the comparison between the two methods ORDER and Optimistic, in which we can analyse that the ORDER method uses locks and Optimistic method does not, uses locks. But it checks the validation at time of committing the transaction. Table shows the basic comparison using the basic assumptions.

Table -1 : Comparison of ORDER and Optimistic Method

Criteria	ORDER [3]	Optimistic[5]
Any Lock Maintenance Overhead?	Yes	No
Any Deadlock Possibility?	Yes (Possible in some case)	No
Abort Ratio?	Yes (In case of Deadlock)	Yes
Any Restart Overhead?	No	Yes
Any Waiting for Resources?	Yes	No
Validation for Read Transaction?	No	Yes

Improved Optimistic Validation Method

In Improved Optimistic Validation Method, it was observed that, in most applications, the possibilities of two clients' transactions accessing the same data item are low. Transactions are allowed to proceed as though there were no possibilities of conflict with other transactions until the client completes its task and issues a *terminatetransaction* request. When a conflict arises, some transaction is generally aborted and will need to be restarted by the client.

This method was developed as an extension of the optimistic method proposed by Kung and Robinson [3]. It was assumed that, the conflicts are very rare in the database system, and the number of read transactions are very high than the write transaction.

The Improved Optimistic Validation Method adds new concept to the previous optimistic method. In this method, there are three phases namely Operational Phase, Validation Phase, Update Phase. In this method, read transaction does not need to perform validation phase, which minimizes the execution time of the transaction. This improved method has another feature of starvation, which checks for the aborted transaction, and give it a right chance to execute.

Validation Algorithm

Tn: Transaction
StartT: The highest sequence number at the start of Tn
ValidateT: The highest sequence number at the beginning of its validation phase
declare valid=true
declare op=read/write
if(op==read)then valid=true
else

for t=StartT+1 to ValidateT do
if(writeset(t)&readset(Tn)=={})then
valid=true
else
valid=false
if valid=true then go to update phase

Above algorithm will provide efficient results in situation where the possibility for read transaction is more.

IV. RESULTS

The Improved Optimistic Validation Method gives optimum performance when the system has minimum conflict ratio and the transactions are read in most cases. The proposed method does not disturb read transaction, increases the execution of transaction. The additional feature has been added of the starvation. Due to the abort ratio, sometime one of the transactions is not able to perform and gets aborted many times, in that case proposed method helps the transaction to execute with different conditions.

Comparison of ORDER, Optimistic and Improved Optimistic Validation Method

In this section of chapter, we will compare the methods which are previously developed and new one. The comparison criteria have been selected as per the need of the database environment. We will compare three methods listed below in the Table 2.

Method 1: ORDER
Method 2: Optimistic Method
Method 3: Improved Optimistic Validation Method

Table -2 : Comparison of ORDER, Optimistic and Improved Optimistic Validation Method

Criteria	Method 1 [5]	Method 2 [3]	Method 3
Any Lock Maintenance Overhead?	Yes	No	No
Any Deadlock Possibility?	Yes (Possible in some case)	No	No
Abort Ratio?	Yes (In case of Deadlock)	Yes	Yes
Any Restart Overhead?	No	Yes	Yes (in few case)
Any Waiting for Resources?	Yes	No	No
Validation for Read Transaction?	No	Yes	No
Starvation for execution?	-	Yes	No

The above table shows the comparison between three methods with some common criteria. Method 3 is improved method, developed during this research work. If we look into the table, method 3 has been improved in two parameters – Validation for read transactions and starvation for execution.

V. CONCLUSION

The Improved Optimistic Validation Method solves above discussed problems in the field of concurrency control. The proposed method provides non-locking algorithm for the transaction execution. It reduces the overhead of lock acquisition and lock releases. The improved method uses no locks, meets no deadlock leads to minimum abort ratio.

The assumption was made that, proposed method suits to the environment having minimum conflict. And the most of the transaction has read operation to perform. The main aim to develop this method is to improve the efficiency and transaction execution speed.

The Improved Optimistic Validation Method provides the direct access of the data without any locking. In the method each transaction gets the timestamp which is the real arrival time of the transaction. There are three phases of the method:

Operational phase, Validation Phase, and Update phase.

Limitations and Future Enhancements

The Improved Optimistic Validation Method is efficient to work with the system having minimum conflicts, at the same time the proposed method may not sufficient in the case where possibility of conflict is very high and number of write transactions is high. A method can be developed, which can control over write transactions using locking concept. A new method can be derived to enhance proposed method the way, read transactions should not be disturbed.

VI. REFERENCES

[1]. B. Bhargava, Concurrency Control in Database Systems, IEEE Transaction on Knowledge and Data Engineering, JANUARY/FEBRUARY 1999.

[2]. A. Thomasian, Concurrency Control: Methods, Performance, and Analysis, ACM Computing Surveys, March 1998.

[3]. H. T. Kung, J. T. Robinson, On Optimistic Methods for Concurrency Control, ACM Transactions on Database Systems, Vol 6, No. 2, June 1981.

[4]. P. L. Lehman, AND S. B. Yao, Efficient locking for concurrent operations on B-trees. Submitted for publication.

[5]. R. Shrinivasan, Network-Aided Concurrency Control in Distributed Databases, January 2002.

[6]. S. Ceri, G. Pelagatti, Distributed Databases Principals and Systems, McGraw-Hill International Edition, 1985.

[7]. A. Thomasian, Distributed Optimistic Concurrency Control Methods for High Performance Transaction Processing, IEEE Transaction on Knowledge and Data Engineering, VOL. 10, NO. 1, January/February 1998.

[8]. G. Coulouris, J. Dollimore and T. Kindberg T. Distributed Systems Concepts and Design, published by Pearson Education Limited 2005.

[9]. Morris R., Wong W., Performance analysis of locking and optimistic concurrency control algorithms, May 1985.

[10]. M. Mohamed, M. Badawy, A. El-Sayed., Survey on Concurrency Control Techniques, Communications on Applied Electronics (CAE) – ISSN : 2394-4714 Foundation of Computer Science FCS, New York, USA,2016

Cite this article as :

Monika Patel, Dr. Dhiren B Patel, "Comparison and an Improved Validation Optimistic Approach for Concurrency Control", International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), ISSN : 2456-3307, Volume 5 Issue 1, pp. 16-21, January-February 2019.
Available at doi :
https://doi.org/10.32628/CSEIT19517
Journal URL : http://ijsrcseit.com/CSEIT19517