Concurrency Control Mechanism for Room Selling Using Short-Term Locks

Aye Pwint Phyu, May Kyi Nyein
Computer University (Taung-Ngu)
babypooh121@gmail.com , maykyi2009@gmail.com

Abstract

A database is a structured collection of data items that can be accessed concurrently by several transactions. The concurrency control in distributed databases is an important problem. This paper is intended to build concurrency control system by using OSN (ordering by serialization number) method that uses time interval technique with short-term locks and that OSN scheduler is located on the server site. By using the OSN method, this proposed system can control the concurrent execution of transactions as well as the consistency of database can be maintained. And also it can avoid the deadlock by using short-term locks with time interval. In this paper, room selling for condominium is used as a case study.

1. Introduction

Nowadays, most of the real-time application areas are designed under the concept of Distributed System. In a distributed system, individual computers are communicated by interconnection network over some hardware concept and they share some particular resources attempting to achieve parallel computation. The sharing of resources is a main motivation for constructing distributed systems. Resources may be managed by servers and accessed by clients or they may be encapsulated as objects and accessed by other client objects. Networks of computers are everywhere. The Internet is one, as are the many networks of which it is composed. [5]

Distributed system is one in which hardware or software components located at networked computers communicated and coordinate their actions only by passing messages. This simple definition covers the entire range of systems in which networked computers can usefully be deployed. In the distributed system, when the number of users increases the ability to work well as failure handing, concurrency of components and transparency. [5]

In a network of computers, concurrent program execution is the norm. Clients can do their works on their computer, sharing when necessary. The capacity of the system to handle shared resources can be increased by adding more resources to the network. The coordination of concurrently executing programs is that share resources is also an important.

This paper intends to build the concurrency control system by using time interval technique with short-term locks for room selling of condominium. In this paper, two types of locks are used such as read lock or s lock and write lock or x lock. Before accessing the remote database, the user must acquire the corresponding lock. This proposed system can control the level of concurrent execution of transactions as well as the consistency of database can be maintained. Java RMI can be used to achieve the communication among concurrent processes.

The related works about the concurrency control are discussed in section 2. The concurrency control method used in this system such as locking and order by serialization number (OSN) is also described in section 3. The section 4 is the proposed design of this system and finally the conclusion of this paper is described in section 5.

2. Related Works

There are several researchers who have been proposed to use database management system and distributed concurrency control. A database system is to store information and to allow users to retrieve and update that information on demand. A work [3] explains distributed concurrency control to prevent deadlock when multiple users are accessing concurrently. A. Rgawam, M.Currency and M. Livny [1] proposed two-phase locking protocol before operating on any object a transaction must acquire a lock on that object. After releasing a lock, a transaction must never go on to acquire any more locks. Shared lock is a read operation on data and exclusive lock is an update operation on data. P. Bernstein [2] said that algorithm for transaction manager and lock manager, transaction manager communicate between data processor and lock manager and lock manager decide granted lock or not.

3. Concurrency Control in Database

Concurrency refers to many transactions accessing the same database at the same time. The database system can facilitate the processes read and write requests on behalf of transactions. [6]

A serial execution of transactions provides database's consistency because each individual transaction controls database's consistency. If concurrent transactions depend on each other, these concurrent transactions can be conflicted. So, the conflict transactions can be solved by the concurrency control method. [6]

In this system, concurrency control is performed by OSN (Ordering by serialization numbers) method. The OSN method provides the serializability of concurrent transactions.

3.1 Locking

Whenever a transaction accesses a data item, it locks it. A transaction which wants to lock a data item which is already locked by another transaction must wait until the other transaction has released the lock (unlock). The operation conflict rules are as follow: Two kinds of locks, exclusive locks (X locks) and shared locks (S locks), X and S locks are sometimes called write locks and read locks, respectively [4]. These locks are shown in table 1.

Table 1. Lock Table

Lock request \ Lock being held	Shared	Exclusive
Shared	Yes	No
Exclusive	No	No

If a transaction 'T' has already performed a 'read' operation on a particular object, then a concurrent transaction 'U' must not 'write' that object until 'T' commits or aborts. If a transaction 'T' has already performed a 'write' operation on a particular object, then a concurrent transaction 'U' must not 'read' or 'write' that object until 'T' commits or aborts. [4]

Even locking method can lead to deadlocks. If two processes each try to acquire the same pair of locks but in the opposite order, a deadlock may result. The deadlock condition is shown in figure 1. When the transaction T1 and T2 wait the lock to release from each other, the deadlock may occur. At the deadlock condition, transaction T1 wants to get B but that is currently locked by the transaction T2 and transaction T2 wants A that is locked by the transaction T1. And therefore transaction T1 waits B to release by transaction T2 and transaction T2 also waits A to release from transaction T1. In the deadlock situation, both transactions will wait each other and can not proceed anyway.

T1	T2	
Lock-X(A)		
read(A)		
write(A)	Lock-X(B)	
	read(B)	
	write(B)	
Lock-X(B)	Lock-X(A)	
read(B)	read(A)	**deadlock**
write(B)	write(A)	

Figure 1. Deadlock Condition

3.2 Ordering by Serialization Numbers (OSN Method)

The OSN method (Ordering by Serialization Number) provides for serializability by combining time interval technique with short term locks. The method works in the certifier mode. [6]

The write operations issued by the transactions are performed on the private copies of the data. Only at the end of the transaction if a validation test is passed by the transaction, the transaction is certified (in other words validated) and its operations are applied to the database. If the validation test is not passed, the transaction is waited on the OSN Scheduler. [6] The OSN method is shown in figure 2.

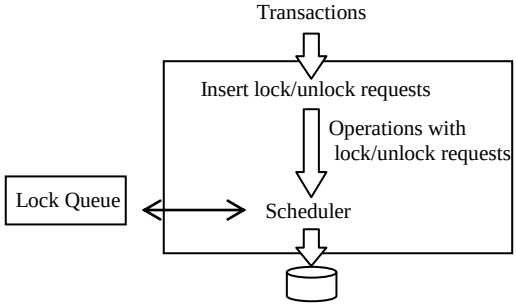

Figure 2. OSN Method

In the validation phase of the OSN method, the method tries to find a serialization number for the transaction that will provide serializability. The OSN method is the kind of optimistic scheduler. A scheduler working in the certifier mode immediately schedules each operation it receives. [6]

4. Proposed System

The goal of this paper is to demonstrate prevention of concurrent database update problems with short term locking protocol using room selling of condominium as a case study. Therefore, concurrent update problems cannot occur in databases of this room selling system and this

proposed system can display information about condo and room to their concurrent users efficiently.

In this system, there are two types of user such as administrators and the users or room buyers. The administrator can insert new condo and related information into the system and can also update this information. After updating the room and condo information by the administrator, the updated information can be seen by the room buyers or users as soon as the update is completed. The overview design of the proposed system is shown in figure 3.

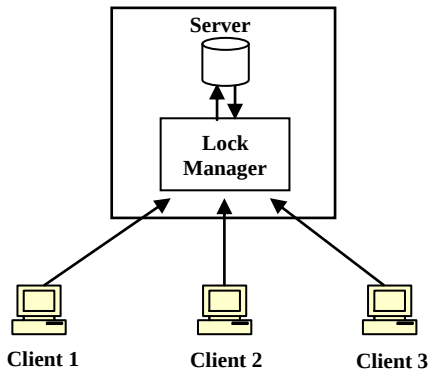

Figure 3. Overview of the Proposed System

When the lock request arrives at the lock manager or OSN scheduler, the scheduler takes a validation test. If the lock request is 'read', the scheduler tests whether other transaction gets 'write' lock on this data item. If the other transaction gets 'write' lock on this data item, the requested transaction is sent to queue. As soon as the 'write' lock is released, the transaction on queue will get lock on this data item. If the lock request is 'write', the scheduler will check both 'read' and 'write' lock on this data item. The algorithm for this lock and scheduler is shown in figure 4:

```
class Client{
Scheduler s=new Scheduler();
                          //Scheduler object (Remote)
Server server=new Server();

Client(){
   displayInfo(RoomDetail);
   choose();
}

void choose(){
   request=lockRequest(); //Read-Write
   Boolean lockReq=s.sendRequest(request);
   If(lockReq){
      server.Process(lockReq);
      s.commit();
   }else Display "Lock Message"
}
}
```

```
class Scheduler(){
   Boolean readLock, writeLock;

   boolean sendRequest(request){
      if (request == 'Read'){
         if (writeLock == 0){
            readLock = 1;
            return true;
         }else{
            waitOnQueue();
            return false;
         }
      }
      else if (request == 'Write'){
         if (writeLock == 0 And readLock = 0){
            writeLock = 1;
            readLock = 1;
            return true;
         } else{
            waitOnQueue();
            return false;
         }
      }
      else return false;
   }

   void commit(){
      if (request == 'Read') readLock=0;
      else if (request == 'Write'){
         writeLock = 0;
         readLock = 0;
      }
   }
}
```

Figure 4. Algorithm of the Proposed System

To view the room information, the user must wait to get read lock ('s' lock) and to buy the room, the write lock ('x' lock) is required. To set the time intervals for the lock, the administrator must input the time limit. In this system, this time interval is dynamic and the administrator can update as needed. The system flow diagram for this proposed system is shown in figure 5.

Locking in concurrency control system can lead to deadlock condition. Therefore this proposed system intends to use lock with time intervals and OSN method to avoid the deadlock condition. When the time is expired, the current lock must be released.

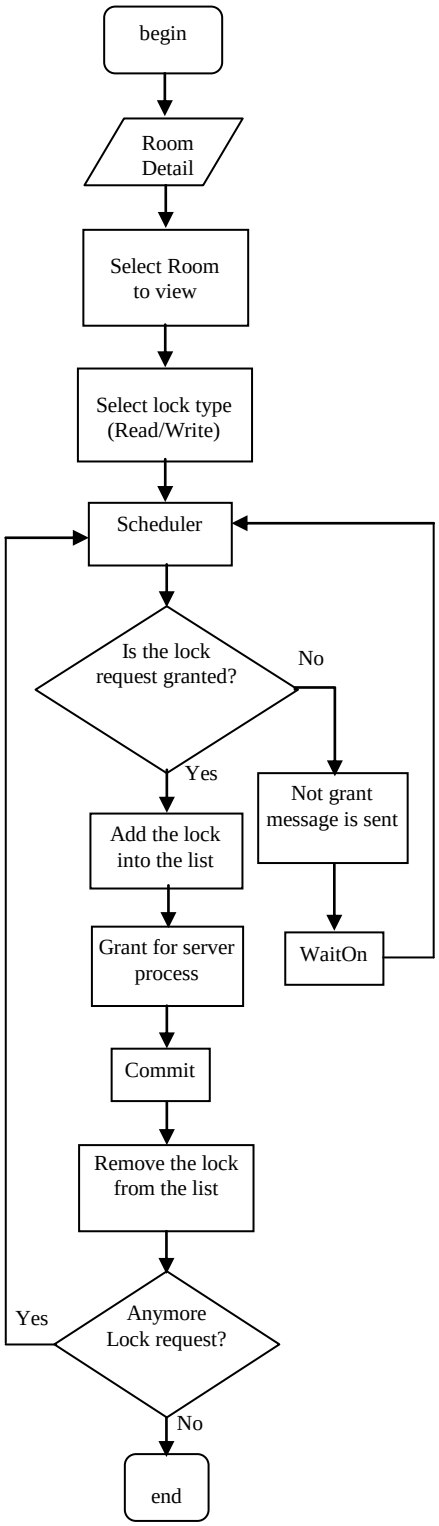

Figure 5. System Flow Diagram

When the client makes request to server, the server will return detail condo information to the requested client. If the client wants to view or buy a room, the client must acquire the respective lock from the scheduler. If the scheduler grants the client's requested lock, the client can take process for this data item in the server.

After the transaction completes successfully (commit or rollback) or the time for the lock is expired, this transaction must release the lock it acquires for other transaction waiting on the queue.

5. Conclusion

This OSN scheduler can avoid starvation because the transaction works within the time interval. When the transaction gets the time interval by using short-term lock, the working transaction does not need to another Read or Write lock. And therefore deadlock is avoided and the OSN scheduler can solve the concurrency control problems in this proposed system.

6. References

[1] A. Rgawam, M. Currency, M, Livny. "Concurrency Control Performance Modeling: Alternatives and Implications". http://www.cs.berkeley.edu/brewer/cs262/Conc.Control.pdf.

[2] Bernstein, P, SHIPMAN, D.W. AND ROTHNIE, J.B. Query processing in SDD- 1, ACM Trans. Database syst.5,1,(March 1980), 18-51

[3] Connolly, C .Begg. "Distributed Concurrency Control", Addison – Wesley, 2002.

[4] C. J. Date, "An Introduction to Database Systems", Seventh Edition, 2000.

[5] C. Georage, D. Jean and K. Tim, "Distributed Systems Concepts and Design" (Third Edition), Person Education Ltd, Edinburgh Gate, Harlow, Essex Cm20 2JE, England.

[6] H. Ugur and D. Asuman, "Concurrency Control in Distributed Databases through Time Intervals and Short-Term Locks", AUGUST 1989.