SALE ORDERING SYSTEM USING CENTRALIZED TWO PHASE LOCKING ALGORITHM

YinYinAye, Khin Phyo Thant
University of Computer Studies Yangon, Myanmar
yinyinaye87@gmail.com, khinphyothantucsy@gmail.com

Abstract

The concurrency control of a distributed system is to ensure that the consistency of the database. The concurrency control mechanism is becoming useful in the ordering system. This paper introduces a concurrency control mechanism for the sale ordering system. Centralized two phase locking is one method of the concurrency control. We use the centralized two phase locking for ordering system because the central lock manager by controlling the all lock information can reduce the additional cost and get the better performance of the system. By using the concurrency control mechanism, the data of the system may be consistent.

Keyword –concurrency control, two-phase locking, centralized two phase locking, client / server system.

1. Introduction

Concurrency control is an important functionality required in a DBMS. The purpose of the concurrency control mechanism is to ensure DB consistency during concurrent access from the different users [5] [4]. Transactions must remain isolated in order to avoid inconsistent values. Different protocols have been proposed in order to ensure concurrent transactions serialization. Concurrency control in distributed system can be classified as pessimistic and optimistic. Pessimistic algorithms synchronize the concurrent execution of the transaction early in their life cycle. Optimistic algorithms delay the synchronization of the transaction until their execution. The most known are: Two-Phase locking and Timestamp Ordering [3]. In this system, we focus on the locking base algorithm for the supermarket stock control. The centralized 2PL is controlling the one central lock manager at the server. The central lock manager is responsible for the all locking information of the clients. We used the C2PL to control the order transactions coming from the clients.

The rest of the paper is organized as follows. Sections 2 describe the related works for the concurrency control mechanism. Section 3 introduces theory back ground for the system. Section 4 presents the detail implementation of the system. Section 5 presents the concurrency control result of the system. Section 6, we present the conclusions of the system.

2. Related Works

The concurrency controls are used in many systems for their data consistent. In the mobile computing system, the concurrency control use to understand the data-objects management over a wireless devices and to study the data consistency [7]. In very large system, the concurrency control uses with nested transaction, to better and faster their system performance [6]. One of the pessimistic concurrency control mechanism locking is widely used in air line reservation system and ordering system. Strict two phase locking of the concurrency controls are used in multiple database system to correct the parallel execution. These multiple databases system purpose is to preserve the benefit of strict two phases locking while ensuring the correct execution [8]. To secure the system data, it is used the two phase locking to show the free from converting channels arising due to the data conflict between transactions [9]. In this system, concurrency

control is used to control the lock of the order item transactions.

3. Theory Background

The main idea of locking-based concurrency is to ensure that the data that is shared by conflicting operations is accessed by one operation at a time. The distributed DBMS not only manages the locks but also handles the lock management responsibilities on behalf of the transactions. The distributed DBMS takes cares of that every time the transaction issues a read or writes operation [1].

3.1. Two Phase Locking

Two-phase locking synchronized read and writes operations by explicitly detecting and preventing conflicts between concurrent operations [2]. 2PL is defined two phases. In the first phase, the transaction is set the lock on each object that will be accessed. In the second phase, the transaction confirms results by writing all modify data to persistent storage, thus releasing the acquire lock [1].

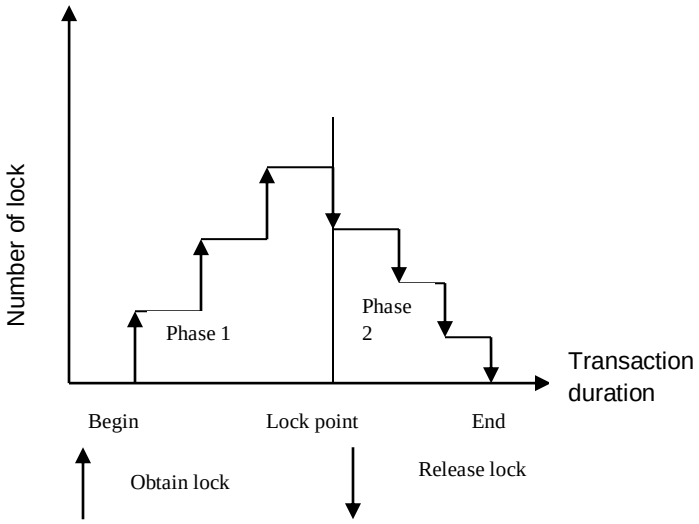


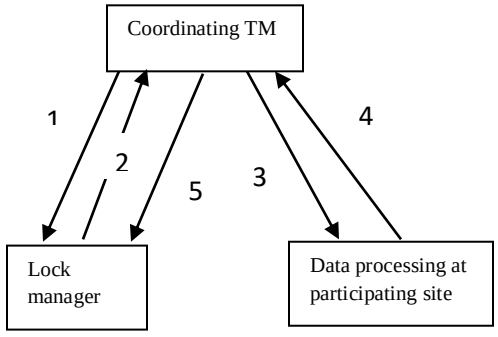
Figure1: 2PL Lock graph

Figure (1) indicates that the lock manager hold the lock on that object. The up arrow indicates obtain the lock and it perform the processes. The down arrow indicates the release the lock and after the process finished. This figure is shown the process of one transaction.

3.2. Centralized Two Phase Locking

In centralized two phase locking, one site in the network is designated as the main site where the scheduling of all transaction and lock tables for all databases are maintained. Main site is responsible to schedule and grant locks to all databases in various sites. One site acts as coordinator for all other sites. One site maintains all locking information. There is only one scheduler or lock manager. Local transaction manager involved in the global transaction requests and releases lock from the centralized local manager using normal 2PL rule. The disadvantages of centralized two phase locking are: low reliabilities and bottleneck. But the benefit of the centralized two phase is communication cost is lower (a global update operation at n sites may require a minimum of 2n+3 messages).

3.3. Communication Structure of Centralized 2PL



Messages:

1. Lock request
2. Lock granted
3. Database operation
4. End of database operation
5. Release locks

Figure2: Process of centralized 2PL

Figure (2) shows the processing between the coordinating to the transaction manager and lock manager. The transaction manager requests the lock to the lock manager and the lock manager reply the lock grant to the transaction manager. And then the

transaction manager performs the data processing at participating site. After their process finished, informs the transaction manager. Finally, the transaction manager informs the lock manager to release the lock.

3.4. Client / Server Systems

There are number different types of client/server architecture. The simple is the case where there is only one server which is accessed by multiple clients. We call is this *multiple client-single server*. From a database management perspective, this is not much different from centralized databases since the database is stored on only one machine (the server) which also hosts the software to manage it. However, there are some (important) differences from centralized systems in the way transactions are executed and caches are managed. More sophisticated client/server architecture is one where there are multiple servers in the system called the *multiple client-multiple server* approach. In this case, two alternative management strategies are possible: either each client manages it own connection to the appropriate server or each client known's of only its "home server" which they communicate with other servers as required. The former approach simplifies server code, but loads the client's machines with the additional responsibilities. This leads to what has been called "heavy client" systems. The later approach, on the other hand, concentrates the data management functionality at the servers. Thus, the transparency of data access is provided at the server interface, leading to "light clients".

From a data logical perspective, client/server DBMSs provide the same view of data as do peer-to-peer systems. They give the user the appearance of a logically single database, while at the physical level data may be distributed. Thus the primary distinction between client/server system and peer-to-peer ones are not in the level of transparency that is provided to the users and applications, but in the architectural paradigms that is used to realize this level of transparency [1].

4. Implementation of the System

In this system, the client may be required to order, when their stocks level is less than the minimum stocks level and they wanted the new items. The transactions from the client may be concurrent access to the main server. If the order items from the client are not same, the process is performed particularly, else the first transaction access to the server get the lock and another transactions are waiting in the queue. The first transaction manager request the lock to the central lock manager and the central lock manager reply the lock granted to the transaction manager. And then it will perform the processing. In this process the order data item of main server will be decrease and the order items from the client data will be increased. So, the next transaction will be able to get the correctness data. After the process is finished and yours order is success, it will release the lock and another transaction gets the lock and it will make the processing.

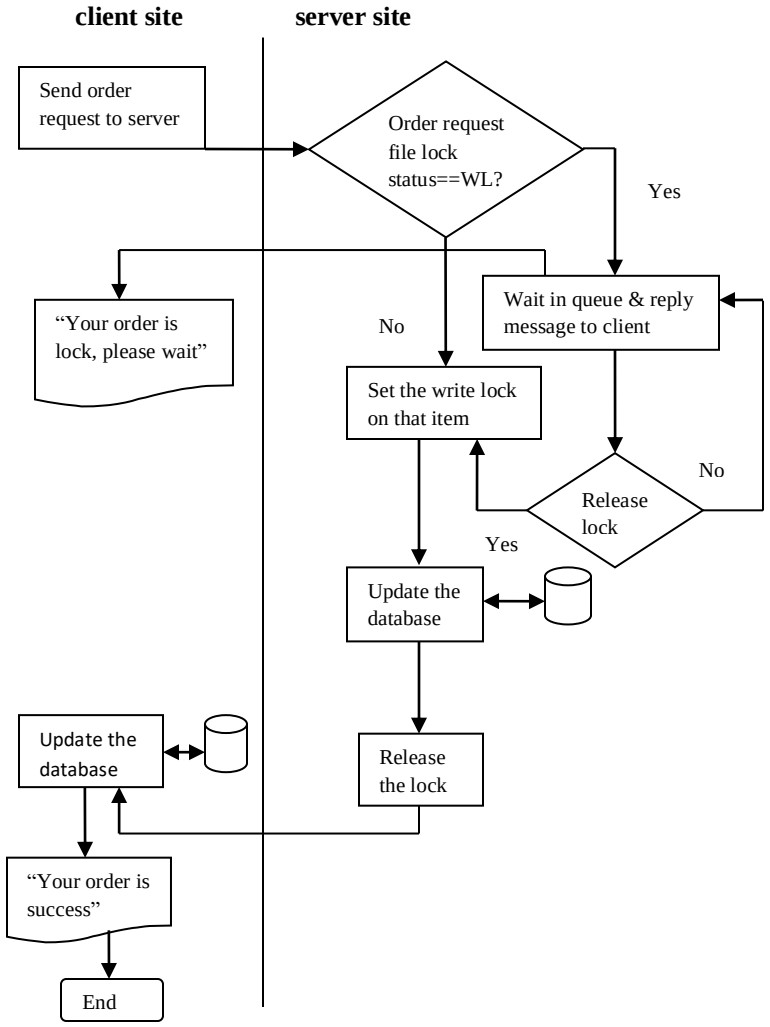


Figure3: System flow Diagram

4.1. System Architecture

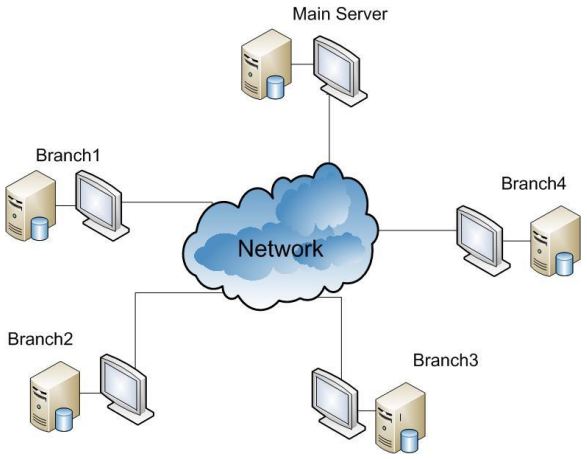


Figure 4: System architecture

This figure is shown the relationship between the sub branches and main server over a network communication. Every sub branches have the own local database. The sub branches do not communicate with one another and they are communicating with the main server.

5. Concurrency Control Result of the System

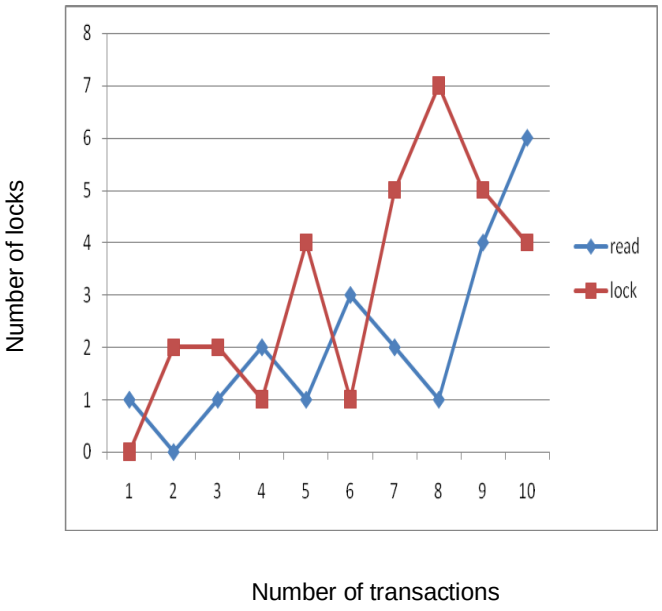


Figure 5: The action of the lock manager between the read and write phase

Figure 5 shows the set of the lock on the transactions. Now we consider for two transactions, the first transaction is read and another transaction is write. In this case, we are not considered for the read because the read locks are not conflict for the same actions. So, the number of lock is one for these two transactions. In this figure, we show the number of transaction to set the lock. Three transactions are access to the server, one transaction is read and two transactions are write we can see the number of lock in the figure.

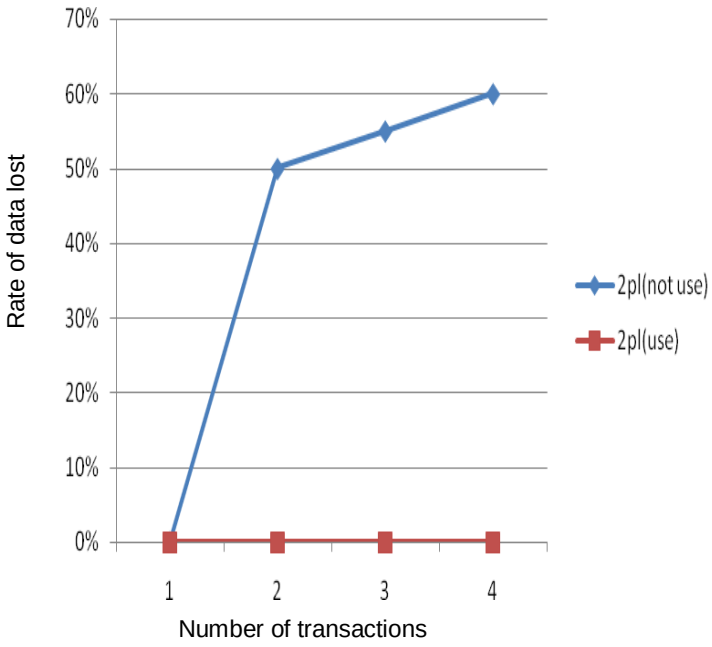


Figure 6: Using and not using the lock for the transactions in the system

Figure 6 indicates the rate of lost for the system performance it does not use the two phase locking in the system. For one transaction, it is convenient but should be considered for many transactions. If it is not use the 2PL to control the transactions in the system, the rate of the lost will be increased, gradually.

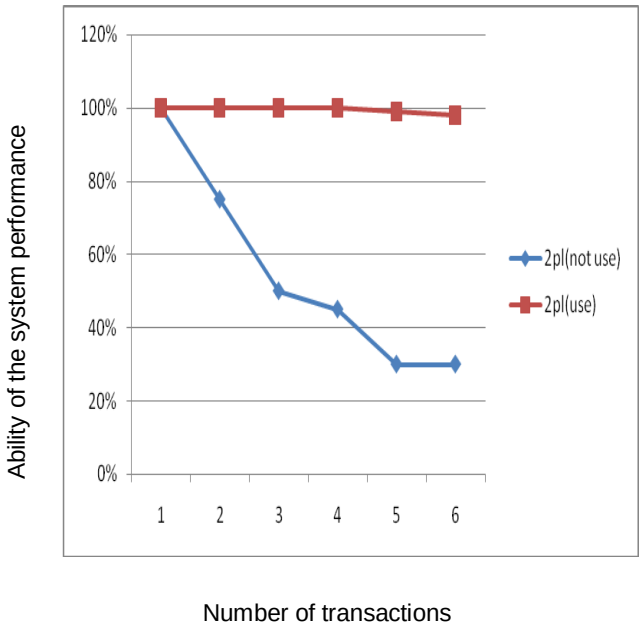


Figure 7: different ability between using and not using 2PL in the system

Figure 7 shows the differences between the using and not using 2PL in the system. If we used the 2PL in the system, we can get the better performance than not using 2PL. If we use the 2PL in the system, the lock manager will solve the concurrent transactions by using locking rules. So, we can reliable on the system performance. If we are not using 2PL in the system, we can not control many transactions and it can cause the inconsistence and lost update problems. The system performance will be decreased.

6. Conclusions

Concurrency control is a crucial mechanism for guaranteeing data consistency during transactions concurrent execution. This paper is focusing on the centralized two phases locking for the super market system. One of the central lock manager at the central site will be solved the concurrent access on the same item. By using this approach, we can get some benefits. They are: to protect the appear concurrency problems liked lost update problem and inconsistent problem, the system data consistencies are better than the other and the last one, communication cost is lower so we can reduce the additional cost for the system.

References

[1] Principle of Distributed Database Systems,Second Edition, Written by M.Tamer Ozsu Patrick Valduriez.

[2] Gray and Reuter.1994 Jim Gray.Andreus Reuter: Transaction Processing Concepts and Techniques.Morgan Kaufmann Publishers, Inc.1994

[3] Berenson et al..1995.Hl Berenson Philip A. Bernstien, Jim Gray, Melton, Elizabeth J.O'Neil. Patrick E.O'Neil. A Critique of ANSI SQL Isolation level Definitions.

[4] Andya et al..2000. Atul Adya.Barbara.Liskoy,Patrick E.O'Neil. Generalized Isolation level Definitions.ICDE 2000.

[5] Harder ans Rothermal 1993. Theo Harder,Kurt Rothermel: Concurrency control in nested transactions: The VDLDB Journal.Vol.2.No.1.

[6] Ekaterina Pavlova,Igor Nekrestyanov St. Petersbut University, Russia.Concurrency control protocol for nested transaction in real time database.

[7] Concurrency control for read only in mobile nested transactions.
LuisA.GamaMoreno.MatiasAlvarado.lgama@agitario.cic.ipn.mx. Concurrency

[8] University of versailles, PriSM laboratory, 45 avenue des Etats-Unis,78000 Versailles(France). E-mail lastname@prism.uvsq.fr.

[9] A secure two phase locking protocol. Rasikan David and Sang H.Son. Department of computer science, University of Virginia, Charlottesville, VA 22903.