CHAPTER 1: THE PROBLEM

1.1 Rationale

A transaction is considered as sequences of read and write operations on database together with computation steps [2]. A transaction can be thought of as a program with embedded database access queries. Let us first consider transaction according to their application areas. If data is distributed, the management of the transaction becomes more involved in coordinating the transactions and this may require special measures. The transactions that operate on distributed data are commonly known as distributed transactions. Data distribution offer opportunities for improving performance through parallel query execution. In order to reap the potential performance benefits, the cost of maintaining data consistency must be kept at an acceptable level in spite of added complexity of the environment.

The life time of transaction is divided into two stages: execution stage and committing stage [4]. During the execution stage, transactions access data through a concurrency control, while in the committing stage, a commit protocol is executed to ensure failure atomicity. For example, in Two Phase Locking protocol, if a transaction in executing stage requests data which is being locked by another transaction in conflicting modes, then the lock request will be blocked until the lock released. The lock of data cannot be released until the transaction completes the committing stage. A transaction that requests a lock can be blocked by a committing transaction for a long time due to a long delay in completing the commit procedure. The potential long delay in the committing stage will block the transaction that needs access to a data item. The concurrency control cannot access the data in their committing stage.

In the concurrency control area, this challenge has led to the development of a large number of concurrency control algorithms. The potential long delay in transaction commitment makes concurrency control wait until transaction finishes its committing stage. This is an important problem for the performance of transaction in distributed database systems. This study present a modification of concurrency control algorithms that use the commit protocol in distributed database system as an aid to concurrency control.

1.2 Theoretical Framework

Based on the characteristics of concurrency control and commit protocol, modifications in concurrency control are required to improve the performance of distributed database system which integrates concurrency control and committing protocol. For distributed databases system, locking based concurrency control algorithms are used. A transaction must have a lock before process a data.

In this study, modified concurrency control algorithm will integrate commit method by Haritsa *et al*. [3] to improve the concurrency. Haritsa *et al.* has shown using simulation, their commit protocol allow optimistically borrow data currently in the committing stage. This protocol can reduce missing deadline in real time database. The modified concurrency control allows give the locks that are still on hold by another transaction in their completion of committing stage. In modeling the concurrency control, Petri Net is used.

1.3 Problem Statement

How to improve the performance of distributed database systems by modifying locking based concurrency control using the concept of resource borrowing and lending from Haritsa's *et al.* committing protocol?

1.4 Hypothesis

The potential long delay in the committing stage will block the transaction that will access a data item. Allowing borrowing mechanism concept for accessing the resources during blocking time may improve the performance of distributed database systems.

1.5 Assumption

1. The transactions have a long delay in finishing the commitment stage.
2. The transactions can by using one operation either single read or single write access a data item.
3. The operations in transaction access only one data item at one time in distributed database systems.
4. The issues of supporting real-time communication and the impact of different network issues on system performance will not be addressed.
5. It is assumed that the network has no failure condition.
6. It is assumed that the network has enough capacity to support the transmission of message.
7. It is assumed of negligible delay in communication.

1.6 Scope of the Research

1. This study characterized transaction only on the basis of their read and update operation without considering the insertion and deletion.
2. There is no replication of distributed databases systems.

1.7 Importance of the Study

Improve the throughput performance of transaction in distributed database systems using borrowing mechanism that is integrated with locking based concurrency control.