Available at
www.ComputerScienceWeb.com
POWERED BY SCIENCE @ DIRECT®

Information and Software Technology 45 (2003) 587–596

INFORMATION AND SOFTWARE TECHNOLOGY

www.elsevier.com/locate/infsof

Evolving a model of transaction management with embedded concurrency control for mobile database systems

Subhash Bhalla*

Database Systems Laboratory, The University of Aizu, Aizu-Wakamatsu, Fukushima 965-8580, Japan

Received 31 October 2001; revised 21 July 2002; accepted 1 March 2003

Abstract

Transactions within a mobile database management system face many restrictions. These cannot afford unlimited delays or participate in multiple retry attempts for execution. The proposed embedded concurrency control (ECC) techniques provide support on three counts, namely—to enhance concurrency, to overcome problems due to heterogeneity, and to allocate priority to transactions that originate from mobile hosts. These proposed ECC techniques can be used to enhance the server capabilities within a mobile database management system. Adoption of the techniques can be beneficial in general, and for other special cases of transaction management in distributed real-time database management systems. The proposed model can be applied to other similar problems related to synchronization, such as the generation of a backup copy of an operational database system.

Keywords: Distributed algorithms; Mobile database systems; Non-blocking protocols; Parallel algorithms; Serializability

1. Introduction

Transaction updates by mobile clients are a desirable feature of many applications [26,16]. Most existing research efforts consider a limited case, of the read-only support for mobile clients. Few other studies consider relaxing the criteria of serializability for processing database update requests. Some more studies propose a prolonged execution sequence. In a disconnection prone system, prolonged execution of transactions is undesirable [17,26].

We consider, an environment based on transaction classification. The transactions at the server end are considered to be short and these can be easily restarted on account of few failures. The mobile client's transactions on the other hand are considered instant execution requests of highest (real-time) priority. The server is assumed to have a high capacity and receives a few cases of mobile client update requests. In many cases, the transaction processing system can execute a mobile client or mobile host (MH) update, with little or no overheads. In the study, conflicts among two mobile client transactions are separately discussed at the end for sake of simplicity.

In order to preserve serializability, the conventional systems depend on 2 phase locking (2PL) protocol [5]. Whereas the 2PL protocol enforces a two phase disciple, the criteria of serializability does not dictate the order in which a collection of conflicting transactions need to execute [5]. This option provides an opportunity to make a modified system that follows 2PL protocol at the transaction manager's (TM) level, but can be flexible at the data manager's (DM's) level. It can permit a interference free and 'non-blocked' execution for MH transactions. This change necessitates maintaining 'lock table' in the form of site level graphs. Although this is the first effort (to the best of our knowledge) to use the technique for mobile databases, many graph based techniques have been studied earlier by Eich and Garard and Reddy and Bhalla [15,24].

It is proposed to execute a mobile host update (MHU) transaction in a special priority fashion. It may need to wait for another low-priority transaction, only if, that transaction has completed and local DM is participating in the second phase of a two phase commit.

The introduction of these possibilities integrates well with the existing transaction execution models. Earlier efforts at separating read-only transactions and update transactions exist [5,13,14]. The present study is an effort that proposes an implementation strategy for isolation of

* Fax: +81-242-37-2753.
E-mail address: bhalla@u-aizu.ac.jp (S. Bhalla).

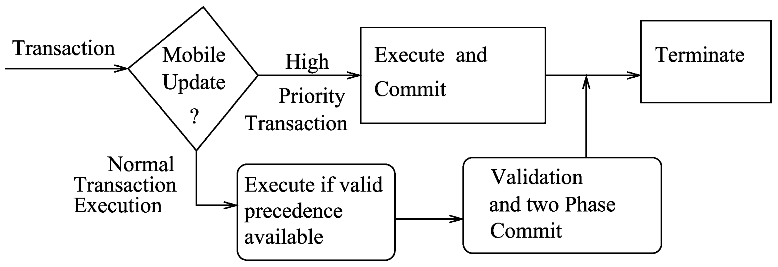
Transaction
Mobile Update ?
High Priority Transaction
Execute and Commit
Terminate
Normal Transaction Execution
Execute if valid precedence available
Validation and two Phase Commit

Fig. 1. Execution of MH update transactions in isolation through embedded 2 phase locking based concurrency control.

Serializable MHU transactions, for such an execution, that is free from interference by other transactions (Fig. 1).

The contents of this paper are organized as follows. Section 2 describes the background of the proposed approach. In Section 3, a model of the system has been presented. A stochastic process model of resource allocation of data resources has been presented, in Section 4. Based on the inferences provided by the studied model, Section 5 considers adaptation of the results for developing strategies for transaction management in mobile database systems. Finally Section 6 presents summary and conclusion.

2. Background

It is common for designers to extend the available approaches for concurrency control for use within the new system environments. However, we propose to study an analytical model and consider introduction of parallelism.

There have been some efforts at introducing parallelism within the concurrency control function. Earlier proposals attempt to eliminate interference between two classes of transactions. For example, processing Read-only transactions separately by using older versions of data, eliminate interference. Within the new classes, transactions are processed with no interference from each other's transactions. These can be considered to be executing in parallel. We propose to study the process of data allocation to executing transactions by using a stochastic process model. The model helps us in examining the parallel activity introduced by the use of classification of transactions. It also provides new insights that can lead to efficient processing of time-critical transactions. In the new environment, the time-critical transactions aim to execute with no interference from the ordinary transactions (Fig. 2). In this light, the characteristics of the 2 Phase Locking based Concurrency Control scheme have been examined, within framework of a Real-Time (time-critical) database system.

3. The system model

Based on the models of 2 phase locking and real-time computational environment with no slack time [12,18], a set of assumptions for executing transactions are organized. It is assumed that a 2 phase locking discipline is followed and the transaction execution is based on the criteria of serializability. Ideally, the MHU transactions should be able to do the following:

- a critical transaction may proceed without interference from other transactions.
- Over ride conventional delays during execution
- integrate with existing modes of transaction executions. The two phases within the 2PL protocol must execute with no blocking;
- execute and commit, i.e. if phase 1 is completed, then phase 2 needs to be completed.

In the following section, a scheme to execute transactions as per a precedence order is described.

3.1. Definitions: mobile database system

Mobile database system (MDS) consists of a set of data items (say set 'D'). The MDS is assumed to be based on a collection of (fixed) servers that are occasionally accessed by mobile hosts. Our assumptions are similar to earlier examples [3,26]. Each site supports a TM and a DM. The TM supervises the execution of the transactions. The DMs manage individual databases. Each mobile host supports a TM, that interacts with a fixed Mobile System Support (MSS) station. That performs other TM functions of interaction with other DMs. The network is assumed to

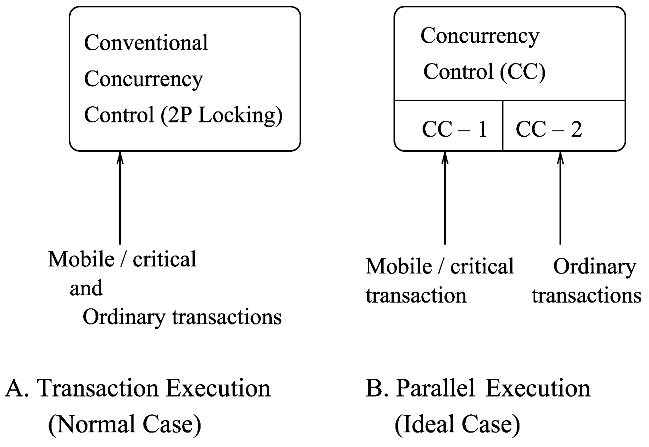
Conventional Concurrency Control (2P Locking)
Concurrency Control (CC)
CC – 1
CC – 2
Mobile / critical and Ordinary transactions
Mobile / critical transaction
Ordinary transactions
A. Transaction Execution (Normal Case)
B. Parallel Execution (Ideal Case)

Fig. 2. Transaction management for parallel execution of transactions.