Recovery of Mobile Transactions *

M. M. Gore and R. K. Ghosh
Department of Computer Science and Engineering,
Indian Institute of Technology, Kanpur, India
gore, rkg@cse.iitk.ac.in

Abstract

This paper addresses the recovery and rollback problem in mobile transactions (MT). We propose mobile transaction recovery protocols to model recovery in mobile computing environment. In the proposed model the mobile host (MH) and the mobile support station (MSS) communicate with each other through messages which are logged at MSS. The image of mobile transaction is created through these message logs on MSS. In the recovery of mobile transactions these image transactions and message logs are extensively used. The recovery algorithms presented in this article works in the mobile computing environment, with different kind of failures and transaction rollback.

1. Introduction

The mobile computing environment consists of stationary and mobile hosts MH. The wireless enabled stationary hosts which help MH retain network connectivity are known as mobile support stations MSS. In order to have freedom in movement, the MHs are light in weight and consequently resource poor [5]. Because of random nature of network connection and greater probability of failure and theft, the modeling of mobile computing applications requires careful attention and more so in the mobile database applications which have to preserve serializability [6], [9]. The complexity of database recovery mechanism is more involved in mobile computing environment. In this paper we propose a solution for the above problem by message logging mechanism. The basic goal of this article is to provide a fault tolerant binding between mobile host applications to static hosts which are linked through computer networks. This goal is accomplished with the help of mobile transaction recovery protocols and message logging. These protocols and message logs are used in data structures and recovery algorithms presented in the paper.

* Partially supported by project number AICTE/CSE/96/264
\(^1\) Also from M.N.R. Engineering College Allahabad

Fault tolerance and recovery are essential for DBMS applications, many researchers have focussed attention on the design and implementation of recovery algorithms. Haerder and Reuter [4], provides a useful taxonomy of recovery techniques and their respective characteristics. These concepts are amplified in further details by Bernstein, Hadzilacos, and Goodman [1] and by Gray and Reuter [3]. Mohan et al. [7] proposed ARIES (Algorithm for Recovery and Isolation exploiting Semantics), which supports partial rollbacks of transactions, fine granularity locking and recovery using write-ahead logging. ARIES makes use of repeating history paradigm. Panagos and Biliris [8] presented a scheme for client server architecture which is based on write ahead redo logging. Rajeev Rastogi et al. [10] presented recovery schemes for distributed main-memory databases, specifically for client server and shared disk architecture. Gore and Ghosh [2] presented a recovery scheme for distributed, collaborative transactions. Their algorithm uses message logging concepts in a generalized ARIES framework.

The outline of this paper is as follows. Section 2 presents a brief description about various definitions, proposed protocols, and system model besides data structures used in the recovery algorithms. Section 3 deals with recovery algorithms and related topics. Discussion on various aspects of recovery procedures is in section 4.

2. Protocols and Data Structures

This section presents mobile transaction recovery protocols. This is followed by system model. Finally major data structures used in recovery algorithm are presented.

2.1. The Mobile Transaction Recovery Protocols

Keeping in mind the inherent characteristics of mobile platforms following protocols are proposed to ensure seamless execution of mobile transactions MT and their recovery in the case of failures.
- **Timeout Protocol:** This protocol is executed by MSS. A timeout parameter is set by mutual agreement of the MH and MSS before commencing the transaction. After expiry of timeout the MSS is free to initiate the roll back activity for the transactions of the said MH. The protocol is designed to recover from inconsistencies due to unusually long inactivity of the MH.

Since MH can interact with several MSS in the course of execution of MT, the time constraint parameter need to be set with each of these MSS.

- **Disconnect Protocol:** This protocol is executed by MH. It is used to redefine the time constraints. It may be required that MH has to disconnect itself from MSS. The disconnection may be warranted due to fading power of battery at MH or fading strength of wireless signals. As in case timeout protocol, the rollback for the transactions of the MH is initiated if disconnection extends beyond the stipulated time limit.

- **Hand-off Protocol:** This protocol is executed by MH. MH asks its current MSS to keep its corresponding image transaction in prepared state. Then MH informs current MSS the address of its subsequent new MSS. After the connection is established with the new MSS, MH conveys it the address of its previous MSS. This way the complete information of the MH and, therefore, of the MT is available to both the MSS.

- **Migration Protocol:** Unlike the Hand-off, in this protocol new settings of MH are communicated to previous MSS by the current MSS, as wireless link is no longer possible between MH and the previous MSS. This migration information should reach to the previous MSS before it executes the timeout or disconnect protocol.

These four protocols effectively model the mobile computing scenario for transaction processing.

The interactions among the MH and MSS are triggered through messages. Besides message data, messages also carry message identifier, issuing transaction id, mobile host id, image transaction id, and message type. In the Figure 1 this scenario is shown. The mobile transaction MT-1 is being executed at MH. The actions of this transaction are communicated through messages to MSS-I. These form image transaction for MT-1 as IT$_1 = 1^{(l)}$. The messages also carry the control information for different protocols detailed above. After some time MH moves on to MSS-J either through hand-off or through migration. The image transaction at MSS-I is put on at prepared to commit and new image transaction IT$_1 = 1^{(l)}$ is formed at MSS-J.

The system model for mobile transaction recovery is as follows.

![Figure 1. The Mobile Transaction Scenario](image)

- There are N MSS. Each MSS has a Transaction Manager, a Lock Manager, a Storage Manager, a Recovery Manager, and a Communication Manager. All MSS posses stable storage. The database is distributed among these MSS. The logging is performed at all MSS.

- MT have their image transactions at one or more MSS. Each MSS maintains its own portion of the log.

- Recovery managers co-ordinate among themselves for multiple site recovery, though it is possible that only a single site has failed.

### 2.2. Data Structures

We modified the ARIES data structures to make them adaptable for for the mobile transactions. Alongside we also added message table with existing data structures like log, transaction table, dirty page table and the pages.

#### The Log

The log knows everything as it maintains complete history. It is a sequence of log records. Each log record contains some housekeeping information but major part of it contains redo/undo information. In the Figure 2 details of individual log record are given. The Mesg record establishes a relationship between the MT and its image transaction is established. There may be more than one MT from a MH and all those can be identified uniquely.

LSN helps in distinguishing one log record from other. In the recovery and roll back two LSNs are very important, they are

- SaveLSN: This is used as low water mark for partial roll back of a transaction. A SaveLSN = 0 means the corresponding transaction is to be rolled back completely.
LogRec: Record
LSN: Log sequence number;
Type: {Update, Comp, Msg, BgnChkpt, EndChkpt, OsFileReturn, End;};
ResponsibleTrID: Transaction identifier;
/* compare with ARIES */
PrevLSN: Log sequence number;
UndoNextLSN: Log sequence number;
PageID: Page Identifier;
Data: Case (Type) of
  Update or Comp: Record
  DataItemName: Name of the data item;
  BeforeImage: Value;
  AfterImage: Value;
  endrecord;
  
  Msg: Record
  MessID: Message Identifier;
  MHID: Mobile Host Identifier;
  sentTrID: Mobile Transaction identifier;
  recdTrID: Image Transaction identifier;
  Type: { Data, Control};
  endrecord;
End: 
  Status: {Commit, Rollback};
EndChkpt: Record
TrTable: Transaction Table;
DirtyPages: Dirty Pages Table;
endrecord;
endcase;
endrecord;

Figure 2. The log record

- MasterRec: This is the LSN of begin checkpoint log record of the latest completed checkpoint before the crash. From this LSN the history is repeated with the help of log records and transaction tables and dirty page tables are reconstructed.

The Transaction Table

The entries of a transaction table are TrID the transaction identifier, LastLSN, UndoNextLSN and ImageLSN the log sequence numbers and Status telling whether transaction has committed, aborted, prepared or is active. Among these ImageLSN is starting LSN of a mobile image transaction. Its log record houses information about MH, and corresponding mobile transaction identifier. for other transactions its value is zero. The transaction table resides in the volatile memory and its contents are copied to stable storage during a checkpoint and get reflected in the end checkpoint log record.

The Page and The Dirty Page Table

The page contains PageLSN field besides the PageID. The PageLSN reflects the latest update made to this page as it is LSN of Log of the said update action.

The dirty page table has two entries one is PageID, the page identifier while other is RecLSN. The RecLSN is used in the redo pass after a crash has taken place. In the dirty pages table the minimum of RecLSN provides the pointer to the log from where the updates have not being reflected to non volatile storage.

The Message Table

The entries in a message table are shown in Figure 3. The record entries in the message tables contain all the information which has passed from MT to its image transaction at a site.

MessageTableEntry: Record
  MessID: Message Identifier;
  MHID: Mobile Host Identifier;
  sentTrID: Mobile Transaction identifier;
  recdTrID: Image Transaction identifier;
  PrevLSN: Log Sequence Number;
  Type: {Data, Control};
  Data: List of update records;
  Control: Protocol Information;
  endrecord;

Figure 3. An entry in the message table

The communication manager at a site puts the message from the mobile host in the stable storage and then notifies the transaction manager which then logs the received message actions against the corresponding image transaction. Thus at each site there is one messages table. The argument for putting the message entries on the stable storage is that they are invariant unlike transaction table or dirty page table entries which change as transactions make progress. In the check-pointing process we do take transaction table and dirty page table entries to the stable storage and in the event of crash we build the transaction table and dirty page table through checkpoint record. For message table none of these actions are required. Further communication manager will have a track of messages it received in the history.

3. Algorithms

This section describes how messages are used in the normal processing and the role played by message tables in the event of partial and total rollback, and the different phases of recovery after a crash has taken place.

3.1. Normal Processing

In the normal processing the locking, latching and concurrency control is similar to ARIES. The image transaction is spawned by the MSS for the given MT of the MH. The MSS also registers the time constraint to be used for possible execution of timeout protocol. In the subsequent mes-
sages actions of transaction are communicated by MH to MSS. These actions are repeated by the image transaction of the mobile transaction. When hand-off or migration protocol is executed the image transaction at the initial MSS is kept in the prepared state while at subsequent MSS a new image transaction is spawned corresponding to the said MT. The recovery managers at respective MSS make necessary amendments through the wireline network to be used for commit or abort of the said MT.

3.1.1. Rollback

The rollback activity for a mobile transaction works in two different modes. At the first instance all the actions of the MT which are executed at MH but are not communicated to MSS can be rolled back without any complications. In the second instance, for rolling back the communicated actions, the MH has to explicitly notify the message identifier till which actions need to be rolled back. The recovery manager at MSS checks its message log to determine the LSN till which it has to roll back. Message identifier also indicates whether rollback should stop at the MSS under consideration or must also span the preceding MSS. The ImageLSN forms the limiting point for the roll back at a MSS. The rollback instruction is passed on to the preceding MSS if required. Thus a partially concurrent rollback can take place at different MSS. The rollback algorithm for mobile transaction environment at one MSS is described in Figure 4.

```
Rollback (RollbackLSN, TrID )
UndoNxt := TrTable[TrID].UndoNxtLSN;
WHILE (RollbackLSN < UndoNxt ) Do
    LogRec := LogRead ( UndoNxt )
    SELECT ( LogRec.Type )
    WHEN ( Update ) Do
        {undo operation; UndoNxt := LogRec.PrevLSN};
    WHEN ( Comp ) Do
        {undo operation; UndoNxt := LogRec.PrevLSN};
    WHEN ( Msg ) Do
        {UndoNxt := LogRec.PrevLSN};
    Otherwise UndoNxt := LogRec.PrevLSN;
    End SELECT /* Log Rec Type */
    End WHILE /* rollback at MSS is complete */
End Rollback
```

Figure 4. Rollback Actions

Thus it is seen that proper and flawless rollback of a given mobile transaction may require co-operation of recovery managers at more than one site. If the rollback ends at the current MSS further actions are continued again from the point rollback finished. For rollback ending at preceding MSS the image transaction with remaining unrolled-back actions is set prepared to commit while further actions of MT are commenced at the current MSS.

3.2. Restart Procedure

The various types of failure which occur in distributed databases are transaction failures, communication failures, media failures and a set of site failures which include individual site, a group of sites or all the sites. A transaction failure occurs due to aborting a transaction, and recovery is done by partial or total rollback of transaction as discussed in the previous subsection. A communication failure occurs due to the failure of communication links such that two sites can not communicate with each other. In our model we have assumed that wireline and wireless networks are FIFO and reliable, and any such failures are handled by computer networks. A media failure occurs due to the damage of non-volatile storage, which homes database and the log. One common remedy of such failures is to have media mirrored on two different devices and at different locations to reduce probability of simultaneous failures. In this paper we assume that media survives a crash. In the event of site failure the contents of volatile storage are lost.

The first action after a crash is to bring system back to a consistent state. ARIES does this in three passes namely analysis, redo and undo. Since the messages are stored in stable storage before delivering, the message table survives the crash. In the event of crash, recovery procedure initializes both dirty page table and active transaction table and puts back entries in them by repeating the history from Begin.Chkpt of the last completed checkpoint.

The restart procedure for MH crash, works in a slightly different manner. If crash took place and MH is up before execution of timeout protocol by MSS, it can resume its normal operation by getting the current state of the MT from the MSS. The analysis, redo, and undo passes which are described below pertain to crash of a MSS.

3.3. Analysis Pass

In proposed scheme analysis pass remains identical with ARIES except for the fact that the messages to be handled for mobile transaction environment. The messages constitute possible analysis points for mobile transactions. All mobile transaction images active at the time of crash can be recreated using message table. The adapted RESTART-ANALYSIS algorithm is shown in the Figure 5.

3.4. Redo Pass

The Redo pass takes database to the state just before the system crashed. In the case of image transactions the actions are already available in the message logs which help redoing those actions.
Figure 5. The Analysis Pass

3.5. Undo Pass

The Undo pass rolls back the active transactions at the time of crash in reverse chronological order. The image transactions of MT are spared from undoing as these are primarily being executed at MH.

4. Conclusion

In this paper we have presented a recovery mechanism for transactions in a mobile computing environment. We have modeled mobile transactions through explicit recovery protocols and messages. In the life of a transaction, a message event is the point of activity transfer and the position of control transfer in normal processing as well as in rollback and recovery. We have illustrated functionality of the system in normal circumstances. The algorithms for crash recovery and partial and total rollback in a mobile transaction environment have been presented and discussed in detail. The benefits of having image transaction for a mobile transaction are also illustrated in crash recovery of a MH. By taking care of crash recovery of both MH and MSS the mobile computing environment can become more useful for applications such as electronic commerce.