http://dx.doi.org/10.18090/samriddhi.v7i2.8631

Transaction Processing in Mobile Database Systems

Ashish Jain*[1]

ABSTRACT

In a mobile computing environment, a potentially large number of mobile and fixed users may simultaneously access shared data; therefore, there is a need to provide a means to allow concurrent management of transactions. Specific characteristics of mobile environments make traditional transaction management techniques no longer appropriate. This is due the fact that the ACID properties of transactions are not simply followed, in particular the consistency property. Thus, transaction management models adopting weaker form of consistency are needed and these models can now tolerate a limited amount of consistency. In this paper we have proposed (execution framework based on common ground shared by most of mobile transaction models found in the literature and investigate it under different execution strategies. More over, the effects of the fixed host transaction are identified and included in the evaluation The integration between wired and wireless environments confirms that the execution strategy is critical for the performance of a system. Neither MHS nor FHS are optimal in all situations and the performance penalties and wasted wireless resources can be substantial. A combined strategy CHS at least matches the best performance of the FHS and MHS and shows better performance than both in many cases.

Index Terms*—mobile data base, transaction processing, execution strategies, execution framework, integrated environment.*

1. INTRODUCTION

Wide area and wireless computing suggest that there will be more competition for shared data since it provides users with ability to access information and services through wireless connections that can be retained even while the user is moving. Further, mobile users will have to share their data with others. Those users may have access to the shared data by reliable wired communication (i.e. fixed host transaction or unreliable wireless communication (i.e. mobile host transaction). The task of ensuring consistency of shared data becomes more difficult in mobile computing because of limitations of wireless communication and restrictions imposed due to mobility and portability [1]. The access to the future information systems through mobile computers will be performed with the help of mobile transactions. Research in mobile database systems (MDBS) has received a lot of interests in the past decade [2], [3]. Transactions in a mobile database system are usually associated with relaxation of ACID properties because of its long-lived nature and the limitation inherited from the wireless environment. Any abortion of a mobile transaction may result in a high cost associated with scarce wireless resources, while fixed transactions might only cause a little degraded level of system performance. In the past two decades, researchers have proposed various transaction models [4] either to tackle the isolation

1.* Ashish Jain, Assistant Professor-I, Electronics & Communication Engineering Department, School of Management Sciences, Lucknow, India, e-mail: ashishjain500@yahoo.com

and atomicity properties of mobile transaction [5] or guarantee the consistency of mobile transaction Most of the previous studies in mobile database system often assume that the system may consist of only one single type of transaction(i.e. mobile transaction). Without paying any attention to the effect of interaction between both types of transactions being shared the same databases simultaneously. Different assumptions have been made on the system and transaction models, e.g., the execution strategy and structures of the transaction, such that different scheduling techniques can be engineered to satisfy the different requirements of mobile transactions. However, little work has been done in the development of an integrated approach that can handle a MDBS satisfactorily where a mixed population of fixed and mobile transactions exists simultaneously. However, for many MDBS applications, such a mixture of workloads is very common. In this paper we proposed an execution frame work for different execution strategies. In section 2 we show the difference between transaction in mobile environment and traditional transaction, section 3 describe the mobile database architecture, section 4 compare between three execution strategies, section 5 present the execution frame work used to compare the strategies, section 6 conclude our work.

2. MOBILE DATABASE ARCHITECTURE

As discussed before, there can be different models for mobile computing systems. Among them, we choose the model most likely to be a computing environment for mobile database systems. In the system, some computers are fixed and some are mobile. Among the mobile computers, some play the role of mobile clients and some the role of mobile hosts.This architecture is widely accepted in existing research for mobile database applications [1], where a global database is distributed among the fixed network nodes. An example of such architecture is shown Fig. 1. In this system architecture, all the network nodes in the wired part are fixed units. Mobile units retain communication through the wireless links. Some fixed units, called base stations or mobile support stations, have special functionality with a wireless interface to communicate with mobile units. Each mobile station provides networking services for all the mobile units within a given geographic area called cell. In other words, each cell has a base station and mobile units within that cell access data on remote nodes through the local base station.

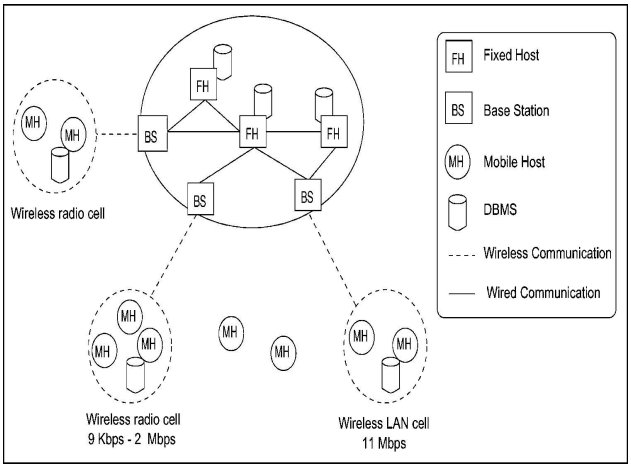

Fig.1: Architecture of Mobile Database Systems

A global database (GDB) is defined as a finite set of data items. A GDB is partitioned among fixed hosts as well as mobile hosts (GDB = MDB FDB). Where the data items on fixed hosts make up the fixed database (FDB) of the GDB (i.e. FDB = FDB_i) and the data items on mobile hosts make up the mobile database (MDB) of the GDB (i.e. MDB = MDB_i). Moreover, the data items on a single mobile host are called a mobile part of the MDB. Accordingly, a data item in the FDB is referred to as a fixed data item and a data item in the MDB is referred to as a mobile data item. Furthermore, each data item has one primary copy

S-JPSET : Vol. 7, Issue 2, ISSN : 2229-7111 (Print) & ISSN : 2454-5767 (Online)

and thus one owner. Only its owner can update the primary copy of a data item. A data item in the MDB is owned by a mobile host and a data item in the FDB is owned by a fixed host. We further assumed that each mobile host has local storage and computing capability, and is able to estimate and exchange its status regarding the mobility information. The fixed data items are replicated by caching on a mobile host, whereas the mobile data items have a full replication on a fixed host. Moreover, a mobile transaction does not access the data on other mobile units (MDBi MDBj = for all i, j such that i j). That is, it can access only local mobile data items and fixed data items.

3. EXECUTION STRATEGIES

3.1 Fixed Host Execution Strategy (FHS)

In this strategy the execution of mobile transactions with a copy of required data on the mobile unit is transferred to the fixed host. The fixed host coordinates the execution of the transaction on behalf of the mobile host and returns the final results back to the mobile unit. As such, a mobile transaction can be processed just like a traditional distributed transaction in a client server model. The only difference is that the propagation of updates and the return of results to a mobile host may be delayed as a consequence of wireless communication. This execution strategy is relatively simple. Its main advantages are:(1) maintaining strict data consistency because an entire transaction is managed and executed on fixed host, traditional transaction schemes can be applied, (2) reducing battery consumption because the operations are shifted to the fixed network, computation tasks on a mobile host are avoided and (3) the level of concurrency at the fixed host is increased due to avoidance of long delays result from poor communication. However, a shortcoming of fixed host execution strategy is that no autonomous operations are allowed during the disconnection of a mobile unit because required data on the fixed host is not available. This policy is suitable when the entire database is allocated on the fixed machines. In other words, a mobile host only plays the role of a client, or a remote device. This strategy has been taken an underlying assumption, of the processing model, by many researchers.

3.2 *Mobile Host Execution Strategy (MHS)*

To allow continuous computation when a disconnection occurs, the data stored at the fixed hosts can be duplicated on a mobile unit rather than moving data to a fixed host. Instead of making a full replication on a local disk, using cached copies is a common technique to support autonomous operations, increase the availability [6], and minimize network access. Since data involved in a mobile transaction is always locally available, this policy allows transaction processing independent of fixed data services. Hence, the autonomous operations can be carried out at the mobile unit. In addition, this policy uses less battery power compared to the FHS strategy, regarding data transmission, because the network connection between a mobile unit and a fixed host is asymmetric. A fixed host typically has a stronger transmitter and unlimited power. Therefore, transporting data from a fixed host to a mobile unit is likely to be more efficient than transporting data in the opposite direction. There are two method of transporting data. Firstly, in planned mode. In this mode required data is transferred from fixed hosts to a mobile unit before a disconnection occurs. This mode requires that a disconnection protocol knows which data will be used in the near future. Secondly, non planned mode. In this mode, data is downloaded based on current transaction requirements and network conditions. For instance, if a transaction uses data

 S-JPSET : Vol. 7, Issue 2, ISSN : 2229-7111 (Print) & ISSN : 2454-5767 (Online)

items X and Y, and at this time point the bandwidth is high, then the data will be transferred to the mobile unit. Considering two types of disconnection, the first mode is more suitable for predictable disconnection, while the second mode is suitable for unpredictable disconnection.

3.3 Combined Host Execution Strategy (CHS)

For mobile transactions having a number of subtransactions which pass through different resources availability during their execution life time, neither MHS nor FHS on its own can give better performance in terms of system throughput and battery consumption in al1 situations. Since the conditions of the mobile environment are dynamically reflected in the transaction processing, an adaptive approach to control the execution of a transaction is desirable. Once that takes fixed host and mobile host execution strategies as two basic options for each subtransaction, a decision is made for a given transaction request based on available mobility information to select a specific execution strategy for each subtransaction of the same mobile transaction in an optimal way corresponding to the current mobile computing environment. With the combined strategy, a mobile subtransaction can be scheduled with a data request or a transaction request. In this approach we can gain two main advantages: (1) a mobile transaction can be tentatively committed based on locally cached data and other transactions can be submitted during disconnection so that autonomous operations can be supported. (2) A mobile transaction can be either executed on a mobile host or a fixed host based on a run-time decision.

4. EXECUTION FRAME WORK

There have been many different models proposed for Mobile Transactions (MT) [1]. In these models mobile transaction supposed to be decomposed in to set of subtransaction which make it possible for the atomicity to be relaxed. The common base between these models their extension of advanced transaction models. Since our work here is concentrate on evaluating execution strategies under different network connectivity conditions rather than evaluating a specific mobile transaction model. We make a general assumption that a mobile transaction MT is defined as a set of mobile subtransactions. The update made by a subtransaction at the execution place should be reflected on the opposite part of the network. (I.e. if the execution takes place at the mobile host, the update made should be reflected on the fixed host and vice versa). So each mobile subtransaction SMT is decomposed into two subtransactions: namely, a basic subtransaction Tb and a complementary subtransaction Tc. Corresponding to this, there are two commit points: local commit and global commit, respectively. A local commit records the status of all basic subtransaction processing when their processing is done, as it may not be possible for their complementary sub-transactions to be issued due to network disconnection at this time. A global commit implies that all complementary subtransaction has been executed and modified data items have been propagated to their master copies. Global commit can happen only when the network is connected. Successful commitment of a mobile transaction occurs if and only if their basic and complementary sub transactions are successfully committed. So the processing of each mobile subtransaction consists of two phases, a basic subtransactions executes in the first phase of processing a mobile subtransaction. Major transaction operations are performed in this phase. A complementary subtransaction derived from a basic subtransaction and occurs in the second phase of mobile subtransaction processing and its execution takes place when the data items on both

S-JPSET : Vol. 7, Issue 2, ISSN : 2229-7111 (Print) & ISSN : 2454-5767 (Online)

the fixed part and mobile part are connected. A complementary subtransaction largely performs updates on the data items that are modified by its basic subtransaction and the place where a basic subtransaction executes is always opposite to the one where its complementary subtransaction does. In general each subtransaction of mobile transactions is scheduled as a data request or a transaction request. The basic subtransaction of a data request transaction is processed on a mobile transaction host with cached data from the fixed part. The corresponding complementary subtransaction is processed on a fixed host. Conversely, the basic subtransaction of a transaction request is processed on a fixed host with the replication of the data items from the mobile host. The matching complementary subtransaction is processed on the mobile host. In this way, the effect of data modification can be propagated to its master copy. When a mobile subtransaction is issued from a mobile host, its basic subtransaction is processed with cached data if network connectivity between the mobile transaction host and its local base station is unsatisfactory (i.e., poor bandwidth or disconnected). If the required data is not available, the transaction is aborted, otherwise after completed; a complementary subtransaction is issued and put into a queue. When the mobile transaction host restores its connection, the queued complementary sub transactions are first submitted through a base station to a fixed host. If a complementary subtransaction fails, the mobile subtransaction is aborted: otherwise it should be waiting for global commitment. If network connectivity is high a mobile transaction is started. The location where its basic subtransaction is processed depends on the execution strategy (see Fig. 2.)

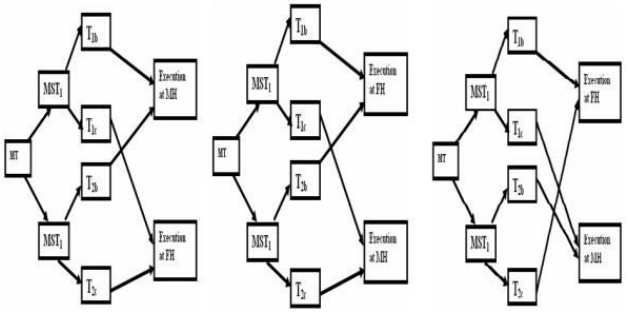

Fig.2: Execution frame work for the three strategies

If the decision is made to use a data request, the fixed data items required by the transaction are downloaded from the fixed host (the size of downloaded data is determined by the amount of valid data in the cache of the mobile transaction host). After this, the process of mobile transaction processing is similar to the one above when network connectivity is poor. On the other hand, if the decision is made to use a transaction request. The mobile data items required by the transaction are uploaded from a mobile part owned by the mobile transaction host, to a fixed host, where the basic subtransaction is processed. Similarly, at this time, if network connectivity between the mobile transaction host and its local base station is below some threshold, the complementary subtransaction will be queued and executed later when network connectivity is improved. If network connectivity remains strong, the complementary subtransaction that performs updates of mobile data items on the mobile transaction host can be issued as soon as the basic subtransaction is completed. The execution of mobile transaction can be described as pseudo code in algorithm.

5. MOBILE TRANSACTION PROCESSING

Begin *For each mobile subtransaction MST Mobile host analyzes the subtransaction MST to build* T_{basic} *and* $T_{complemnetary}$ *based on its read and write requests and cash status (we assume that MH has some server capability).*

 S-JPSET : Vol. 7, Issue 2, ISSN : 2229-7111 (Print) & ISSN : 2454-5767 (Online)

***If** the subtransaction T is scheduled as D-req transaction then*

***Begin** For all data item reads and writes by T_{basic}, if D_i not in the MDB_i cash, MH sends a request to MTM for all the required read values and request for lock. After MTM acquires the necessary locks, it returns values to MH. Execute the basic transaction on the MH and send $T_{complementary}$ to be executed on the fixed host.*

End

Else** if the subtransaction T is scheduled as T-req transaction **then

***Begin** For all data item reads and writes by T_{basic} MDB_i MH sends a request to MTM for all operation with data items D_i MDB_i required by these operations.After MTM acquires the necessary read locks, Execute the basic transaction on the FH and send $T_{Complementary}$ to the mobile host.*

End

***If all** mobile subtransactions successfully finish their execution*

***then** Mobile transaction is committed*

End

5. CONCLUSION

Mobility brings in new dimension to the existing solutions to the problems in distributed databases. We have reviewed some of the problems and existing solutions in that direction. We have highlighted the merits and demerits of existing solutions. We have found from the previous papers that, no comparative performance evaluation of models is presented. We observe that there is a need to investigate the properties of mobility, which can impact most the transaction processing. Also, there is a need to evaluate various transactions processing where a mixed fixed and mobile transactions are coexist in the system. In this paper we have introduce a frame work to investigate three transaction execution strategies based on the general assumption that the information of environment current state are available.

REFERENCES

[1] P. K. Chrysanthis, "Transaction processing in a mobile computing environment," in. *Proc. of IEEE workshop on Advances in Parallel and Distributed Systems*, October, 1993, pp. 77-82.

[2] Park and S. J. Hyun, "A dynamic mobile transaction management strategy for data-intensive applications based on the behaviors of mobile hosts," in *Proc. (367) Information Systems and Databases - 2002*.

[3] S. K. Madria and B. Bhargava, "System defined prewrites to increase concurrency in databases," in. P*roc. of First East-European Symposium on Advances in Databases and Information Systems* (in co-operation with ACM-SIGMOD), St.-Petersburg, September, 1997.

[4] K. Heo, H. Kang, Un-Chul Moon, and J. R Lee "Performance evaluation of vehicle-mounted mobile relay in next generation cellular networks", KSII Transactions on Internet and Information Systems / May, 2011

[5] M. H. Eich and A. Helal, "A mobile transaction model that captures both data and movement behavior," ACM/Baltzer Journal on Special Topics on Mobile Networks and Applications (1997).

[6] Q. Lu and M. Satyanaraynan, "Improving data consistency in mobile computing using isolation only transactions," in. *Proc. of the Fifth Workshop on Hot Topics in Operating Systems*, Orcas Island, Washington, May, 1995.

