Prioritized Transaction Management for Mobile Computing Systems

K.Chitra
Dept of Computer Science, GAC

Al-Dahoud Ali
Al-Zaytoonah University

Abstract

In mobile computing environment, many mobile clients are concurrently accessing the database residing at the server, in the form of reads and writes. The broadcast-based data dissemination, in mobile computing systems, poses new challenging issues on data consistency of mobile transaction processing due to frequent disconnection from the network. Although data broadcast has been shown to be an efficient method for disseminating data items in mobile computing systems, the issue on how to ensure consistency and currency of data items provided to mobile transactions, which are generated by mobile clients, has not been examined adequately. In this paper, we model smart server (SSM) and we control the concurrency of mobile clients' reads and writes to provide consistent highly dynamic data to the mobile clients which are often disconnected from the network. We dynamic transmission disks (DTD) which are broadcasts to satisfy two types of users: frequently accessed data items user, rarely accessed data item users taking into consideration of frequency of accession and also frequency of updates which optimizes size of dynamic multiversion data transmission disks in order to meet consistency and currency. And also we prioritize the reads and writes of mobile clients to maintain consistency of the data provided to the mobile clients.

1. Introduction

The characteristics of mobile computing systems such as disconnection for periods of time, frequent relocation of clients, asymmetry in communication and power limitations poses new challenges in the area of mobile communication systems [3]. The benefits outweigh the disadvantages are:

- Wireless access to the real-time database
- Allow the location of the user to change
- It facilitates processing in a real time system

Consider wireless network in which mobile clients are connected to fixed network by wireless link as shown in figure 1. An access point in each cell provides the connection between the mobile client and the fixed network. The location of a database server is fixed.

Data will be stored at the fixed database server. Many external resources are updating the data in the database. The server periodically and continuously broadcast data items from the database [1, 2]. The mobile clients receive data through broadcast channels and process their transactions locally. The set of data items to be broadcast per cycle is called bcast and the time taken to broadcast set of data items is called broadcast cycle or bcycle. The objective of this paper is to provide the consistent and current data items to the mobile clients though they are disconnected from the network for some time (Disconnection Tolerance). Given that the server knows the frequency of accession of data items accessed by the mobile clients. Hence, while scheduling data items for broadcasting, two factors are considered. They are

1. Frequency of Accession (FoA) of data items by the mobile clients
2. Frequency of Updates (FoU) of data items at the database server

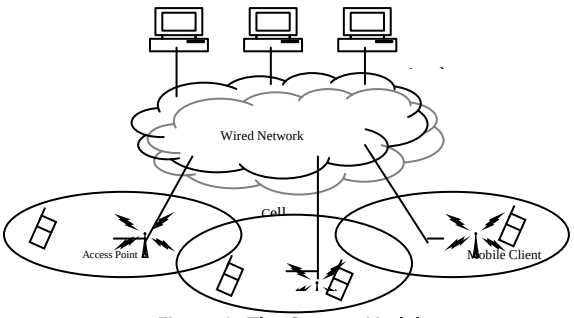

Figure 1. The System Model

2. The Prioritized Write Back (PWB) Smart Server Model (SSM)

The server maintains multiple versions of each data item. That means, when a data item is updated, the new value is stored with a version number. An m-multiversion server is one which maintains 'm' versions of each data item, i.e when the data item is updated, the oldest version is removed and the new version is stored. This server also keeps track of the update frequency of each data item. The multiversion server periodically and continuously broadcast data items from the database.

When many mobile clients are trying to access the same database, the server checks for their transaction. If they are read_transaction and write_transaction on the same data, write_transaction is executed first and then it satisfies the read_transaction. Prioritizing the write backs, the server controls concurrency and helps to provide consistent and current data to the mobile transactions [10].

And also when any data item is updated, by any external resource, the server verifies whether the current value and updated value are same. If current value and the updated value are same, the server ignores the update and new version is not created. (For example, a server is dynamically and periodically (i.e. every minute), updating cricket score. If the previous minute cricket score and current minute cricket score are same, the server ignores the update.

2.1.Algorithm at the m-multiversion Server

MT - Mobile Transaction
RMT - Read Set of Mobile Transaction
WMT – Write Set of Mobile Transaction
d ε RMT
e ε WMT
Did - Data Identity
V0 – Oldest Version
Vm-1 – Latest Version
Val[Did.v0]–Data Value of the oldest version V0
Val[Did.vm-1]–Data Value of the latest version V_{m-1}
Did.new_val - Did to be updated with new_val

```
While (MT)
{
        If ((Did = d) and (Did = e))
        //Same data needed for read and write
                Write back the data 'd'
        //prioritize write back first
        If (Val[Did.V_{m-1}] = = Did.new_val)
                Ignore the update
        Else
                Delete Val[Did.V_0]
                For(i=0;i<m-1;i++)
                {
                Val[Did.V_i] : = Val[Did.V_{i+1}]
                }
                Val[Did.V_{m-1}] : = Did.new_val;
                Commit;
        End If
        Read the data 'd' in the next broadcast cycle
        End If
}
```

3. Dynamic Data Transmission Marshaling

Most of the previous works concentrate on broadcasting frequently accessed (FoA) data items more frequently [2], [3] and [9]. Such data items are called hot data items. Broadcasting some of the hot data items more frequently will delay the broadcast of other data items and thus prolong the access time for less popular data items. As a result, the policy imposes unfair treatment for the users who are interested in less frequently accessed (FoA) data items.

Hence, proposed dynamic transmission disks for multiversion servers are very much useful in satisfying both the types of users (Less frequently accessed data items user and frequently accessed data items user). To reduce the latency of client transactions, it has been proposed that, instead of broadcasting each item once during a bcast, the frequency of broadcasting an item is determined based on the probability of it being accessed by the clients. Such a schema is called broadcast marshaling.

In a transmission data marshaling, the items of the broadcast are divided in ranges of similar access probabilities. Each of these ranges is placed on a separate marshal. In the example of Figure 2, data items of the first marshal, Marshal1, are broadcast three times as often as those in the second marshal, Marshal2. To achieve these relative frequencies, the marshals are split into smaller equal sized units called slabs; the number of slabs per marshal is inversely proportional to the relative frequencies of the marshals. In the example, the number of slabs is one (slab 1) and three (slabs 2a, 2b, and 2c) for Marshal1 and Marshal2, respectively.

Each Bcast_Schedule is generated by placing one slab from each marshal and cycling through all the slabs sequentially over all marshals. A minor cycle is a subbcycle that consists of one slab from each marshal. In the example of Figure 2, there are three minor cycles.

4. Dynamic Transmission Disks (DTD)

Now we need to arrange Bcast_Schedule so as to accommodate multiversion. A direct application of the dynamic transmission disks on multiversion broadcast is to base the distribution of each data item based on its update frequency (FoU). The dynamic transmission disks formation method does the same as shown in Figure 3. With this method, the number of versions of each data item to be placed in the dynamic transmission disk is decided by the update frequency (FoU) of that data item. Consider the update frequency is 2:1. Thus, three different versions of frequently updated hot data items and

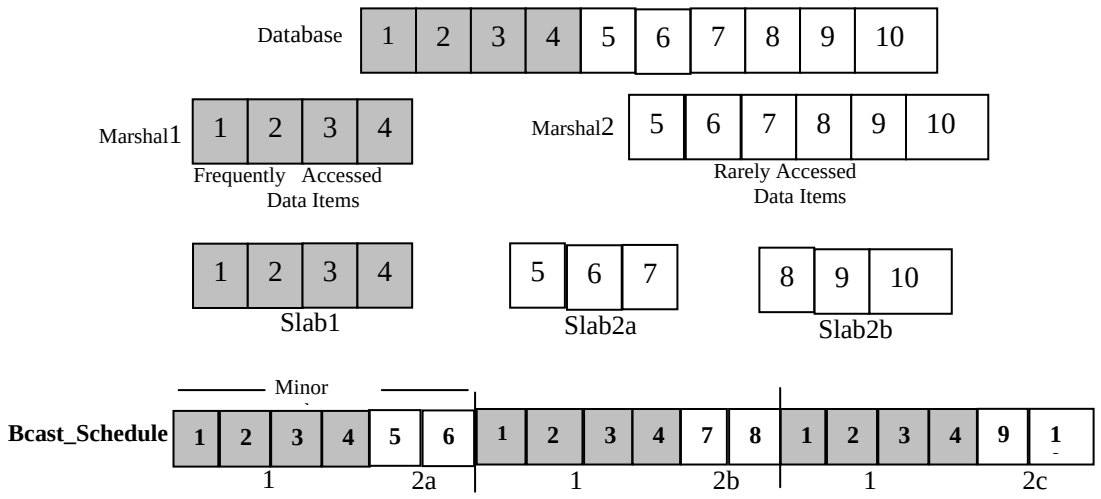

Figure 2. Transmission Data Marshaling based on FoA

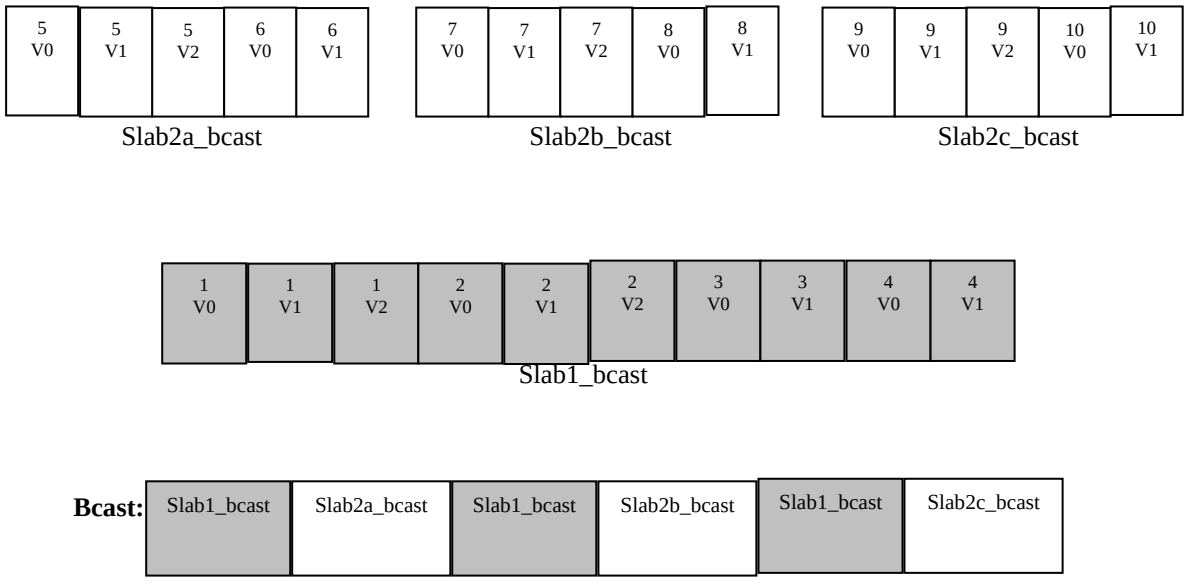

Figure 3. Dynamic Transmission Disks (DTD) based on FoU

two different versions of rarely updated hot data items (slab 1 in Figure 3) are placed on slab1_bcast, while three different versions of frequently updated cold data items and two different versions of rarely updated cold data items (slabs 2a, 2b, and 2c in Figure 3) are placed on slab2a_bcast, slab2b_bcast and slab2c_bcast. Consequently, dynamic transmission disks method works well when each transaction may access any version of an item with relative update probability.

The size of each slab is increased to accommodate old versions with relative frequency of update (FoU). The number of slabs per marshal remains fixed. The overall increase in the size of the bcast depends on how the hot data items are related to the items that are frequently updated. The increase is the largest when the hot items are the most frequently updated ones since their versions are broadcast more frequently during each bcycle. But this increase is compensated with the rarely updated hot data items and rarely updated cold data items. This approach is easily extended to multiple marshals. This approach, namely marshalling data items considering both relative frequency of accession as well as the relative frequency of update, reduces the size of multiversion broadcast disks and also it meets the major objective of this paper consistency and currency of data items received by the mobile clients. Hence the dynamic transmission disks works well even when the client is disconnected from the network as it is transmitting old versions combined with new versions and the mobile clients receive consistent and current data from the server. Hence even if the mobile client is disconnected from the network for sometime, it gets the consistent and current data items in the next broadcast cycle. With this scheme named DTD, the mobile transactions receive consistent and current data items from the server.

5. Performance Evaluation

Now, we evaluate the performance of dynamic data transmission marshaling clustered first using frequency of accession (FoA) and then using frequency of updates (FoU) named dynamic transmission disks (DTD). The proposed DTD is compared with the simple multiversioning (MV) method in which the same number of versions of all data items is clustered for transmission. We have considered the frequency of update 2:1. We increase the number of versions of frequently updated data items from 1 to 5, while the number of versions of rarely updated data item is relatively less. As shown in Figure 4, when update rate is 5%, if we cluster the number of versions of data items depending on the frequency of accession and also the frequency of update (namely DTD) the abort rate of the mobile transactions are substantially reduced. Therefore the number of successful mobile transactions getting consistent and current data items is increased. Figure 5 shows, when update rate is 10%, if we cluster the number of versions of data items depending on the frequency of accession and also the frequency of update (namely DTD) the abort rate of the mobile transactions are remarkably reduced and it reaches '0'.

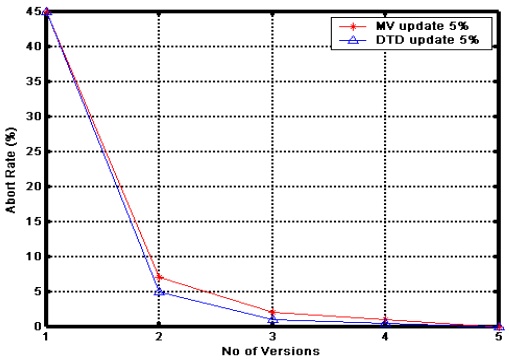

Figure 4. No of Versions Vs Abort Rate (update rate 5%)

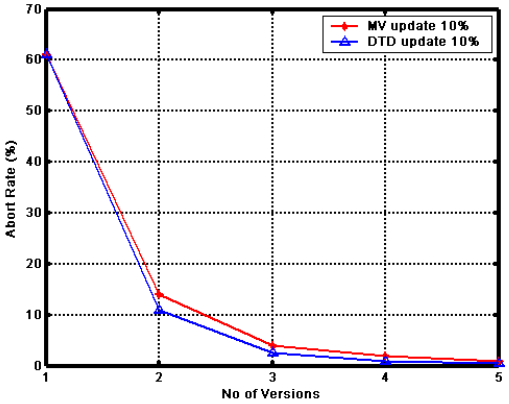

Figure 5. No of Versions Vs Abort Rate (update rate 10%)

When the mobile client is disconnected from the network for some percentage of broadcast cycle, the proposed DTD tolerates disconnection compared to simple MV method. As shown in Figure 6, the abort rate of the mobile transactions is reduced to 6% when the mobile client is disconnected from the network equivalent to the time of broadcasting 40% of the broadcast items and 14% abort rate when it is disconnected for 60% of the cycle size. Hence it is clearly shown that DTD tolerates disconnection than MV.

Another important advantage of our smart server model (SSM) with prioritized write backs (PWB) is controlling concurrency and providing consistent and current data to the mobile clients. It is used to reduce the stale access rate of the mobile transactions.

CCPWB as compared with MV, the stale access rate of CCPWB is close to zero as shown in Figure 7. It is because the concurrency controlled prioritized write backs (CCPWB) helps to maximize the

currency of the data items provided to the mobile transactions.

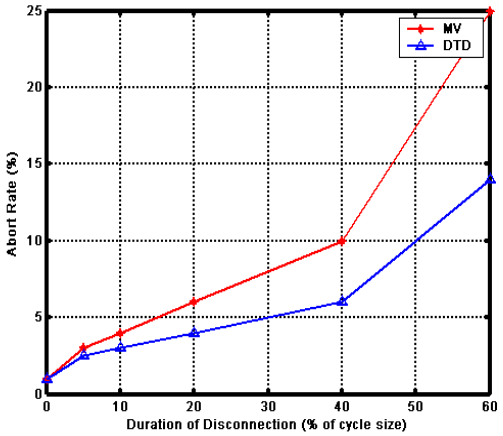

Figure 6. Duration of Disconnection Vs Abort Rate

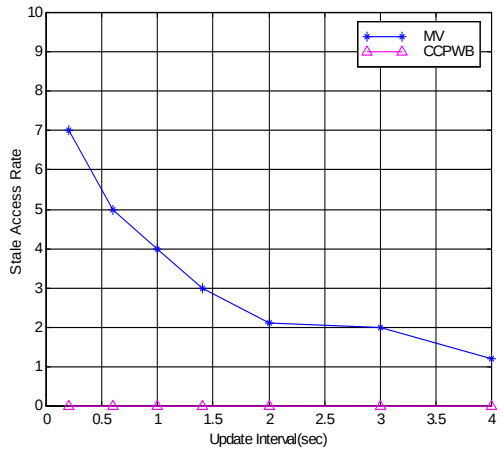

Figure 7. Stale Access Rate Vs Update Interval

6. Conclusion

Dynamic data transmission marshaling, named dynamic transmission disks (DTD), is done first using frequency of accession (FoA) and then using frequency of updates (FoU). The proposed DTD is compared with the simple multiversioning (MV) method in which the same number of versions of all data items is clustered for transmission. It has been proved that if we cluster the number of versions of data items depending on the frequency of accession and also the frequency of update, the abort rate of the mobile transactions is reduced and the mobile clients receive consistent and current data items from the server. And also when the mobile client is disconnected from the network for some time (percentage of broadcast cycle), the proposed DTD tolerates disconnection compared to simple MV method. Our Smart Server Model with prioritized write back controls concurrency and provides consistent and current data to the mobile transactions.

7. References

[1] S. Acharya, R. Alonso, M.J. Franklin, and S. Zdonik, “Broadcast Disks: Data Management for Asymmetric Communications Environments,” Proc. ACM SIGMOD Int’l Conf. Management of Data, pp. 199-210, 1995.

[2] S. Acharya, M.J. Franklin, and S. Zdonik, “Disseminating Updates on Broadcast Disks,” Proc. 22nd Int’l Conf. Very Large Data Bases, pp. 354-365, 1996.

[3] D. Barbara´, “Certification Reports: Supporting Transactions in Wireless Systems,” Proc. IEEE Int’l Conf. Distributed Computing Systems, 1997.

[4] E. Pitoura and G. Samaras, Data Management for Mobile Computing. Kluwer Academic, 1998.

[5] Mequite Software, Inc., CSIM17 Users’ Guide.

[6] T. Bowen, G. Gopal, G. Herman, T. Hickey, K. Lee, W. Mansfield, J. Raitz, and A. Weinrib, “The Datacycle Architecture,” Comm. ACM, vol. 35, no. 12, pp. 71-81, 1992.

[7] C. Mohan, H. Pirahesh, and R. Lorie, “Efficient and Flexible Methods for Transient Versioning to Avoid Locking by Read-Only Transactions,” Proc. ACM SIGMOD Int’l Conf. Management of Data, pp. 124-133, 1992.

[8] E. Pitoura and P.K. Chrysanthis, “Multiversion Broadcast,” extended version, Technical Report, TR-2002-11, Computer Science Dept, Univ. of Ioannina, Apr. 2002.

[9] E. Pitoura and P.K. Chrysanthis, “Scalable Processing of Read-Only Transactions in Broadcast Push,” Proc. 19th IEEE Int’l Conf. Distributed Computing Systems, 1999.

[10] R. Varadarajan, Manikandan Chitra, “Scheduling data reads and writes using feedback control on a weakly connected environment”, International Journal of Computer Applications in Technology, Volume 34 Issue 3, March 2009.

