Broadcast Transaction Scheduling in Mobile Computing Environments*

Simon Wu, Victor C. S. Lee†, and Kwok-wa Lam
Department of Computer Science, City University of Hong Kong
†csvlee@cityu.edu.hk

Abstract

Most of the recent studies on broadcast based data dissemination approach have an implicit assumption that the server is able to broadcast consistent data to the mobile clients. However, this assumption may not be valid unless there is a special algorithm to handle broadcasting data in a consistent and timely manner. In this work, we address a number of issues related to this problem. An efficient algorithm called the read-write set test is adopted to read the entire database, which causes little interference with the update transactions at the server. For broadcast transaction scheduling, an algorithm based on a mixed strategy is proposed to handle the case of flat broadcast disks. Using these two algorithms together with some fine tuning, the objectives of reading entire database consistently with minimum impact to the system, broadcasting fresh data, and meeting the periodicity requirement can be met.

1. Problem description

To broadcast consistent data entities to mobile clients, a broadcast transaction at the server is needed to read the entire database in a serializable way while other update transactions are being processed concurrently. However, handling this “long living” global read transaction efficiently is not trivial. In many database systems, a consistent database is obtained by suspending transactions that update the database while the entire database is being read in progress [3]. In this paper, a new algorithm called the mixed strategy is proposed to address the scheduling issues in flat broadcast disk [5] environments.

The broadcast transaction $G$ is used to read the entire database for producing a consistent, full database image for next broadcast cycle. The scheduling algorithm for $G$ should be able to reduce the interference caused by this long living transaction to other update transactions that are executing concurrently. Also, the algorithm should produce a “fresh” (i.e., up-to-date) database image in a controllable way such that the soft deadline timing constraint requirement imposed by the periodicity of broadcast cycle can be satisfied. As a result, quality of service can be committed to the mobile clients by providing fresh data on time. Otherwise, missing the deadlines may lead to delay of broadcast cycles and poor utilization of air bandwidth.

2. The mixed strategy algorithm

The proposed mixed strategy is a separate algorithm to handle the scheduling problem of broadcast transaction. In this algorithm, the concurrency control of update transactions are handled by traditional optimistic method [4] with forward validation, while the conflict resolution between $G$ and other concurrently-executing update transactions $U$ is handled by the read-write set test (RW-Set) proposed in [2]. Under the RW-Set test, an update transaction $U$ is not allowed to commit if it satisfies one of the two conditions (1) $U$ reads from the NUS (set of data entities read by $G$ and are subsequently updated by some other transactions) and then writes to the NRS (set of data entities not read by $G$ yet) such as $T_1$ in Figure 1, (2) $U$ writes to both ARS (set of data entities read by $G$) and NRS such as $T_2$ in Figure 1. An $U$ satisfying any of the above conditions implies that it cannot be serialized with $G$ and needs to be aborted. Otherwise, this algorithm can serialize $U$ either before or after $G$, thus creates a serializable history.

![Figure 1. Allowed and disallowed transactions](image-url)

Using the RW-Set test for concurrency control guarantees $G$ can be completed because it aborts any
conflicting U and has no effect to G. In order to avoid U being restarted and yet increase the overall performance of the system, some modifications to avoid the above two conditions being satisfied are introduced for the RW-Set test. These modifications were based on the idea from [1] with further enhancements.

To avoid condition (1) being satisfied, one way is to move all the data entities to be updated by U from NRS, if any, to ARS in advance. Therefore, U only writes on data entities in ARS, which is legal thus can be committed. Then, U is serialized after G. To achieve this, U will need to “read” those data entities in NRS on behalf of G before they can be moved. This strategy is known as read-forward strategy.

To avoid condition (2) being satisfied, another strategy called re-read is used. The data entities to be updated by U are moved from ARS, if any, back to NRS. This method forces the broadcast transaction to re-read the updated values from database again by moving the data entities back to NRS. Then, U is serialized before G. A positive effect of this operation is that G will not miss these updates. Figure 2 compared and summarized these two strategies.

![Figure 2: Comparison of strategies](image)

The overall idea of the mixed strategy is to schedule G at TG (Figure 3) where TG is some time earlier than TL (the latest time to schedule G such that G can be completed before TG end of a broadcast cycle) such that there is slack time TS (i.e. TS = TL − TG) for G. Any update transaction U executes concurrently with G and wants to commit is required to pass the RW-Set test. If it fails, re-read strategy is invoked. Although some slack time is consumed each time the re-read routine is invoked to handle the conflicting U, data freshness can be maintained. Later, when TS drops below a certain predefined threshold Td (i.e. TS < Td), G changes its mode of execution from re-read to read-forward strategy and becomes more aggressive. In either case, U is allowed to commit.

![Figure 3: Example of the proposed scheduling](image)

To conclude, the mixed strategy allows G to put data freshness in higher priority when there is much slack time and let G to finish on time when there is little or no slack time left by attempting to serialize all conflicting update transactions after G.

3. Conclusion and future work

There is a need to serve a consistent image of the whole database periodically in broadcast-based data dissemination environments. A broadcast transaction can be used to read the entire database. The output database image will be served as the input for next broadcast cycle. In this research, the mixed strategy is proposed as a scheduling algorithm for handling the broadcast transaction. This mixed strategy used in conjunction with the read-write set test algorithm [2] together provides a high quality broadcasting service.

In the later stage, the mixed strategy will be evaluated extensively with a series of simulation experiments. The behavior of the algorithm will be analyzed with different performance metrics and the parameters involved in the mixed strategy will be fine-tuned for better performance.

4. References