Efficient Cache Invalidation in Mobile Environments

Po-Jen Chuang*, Ching-Yueh Hsu and Yu-Shian Chiu

Department of Electrical Engineering, Tamkang University, Tamsui, Taiwan 251, R.O.C.

Abstract

In a mobile environment, caching data items at the mobile clients is important as it reduces the data access time and bandwidth utilization. While caching is desirable, it may cause data inconsistency between the server and the mobile clients if their communication is disconnected for a period of time. To ensure information coherence between the source items and their cached items, the server can broadcast invalidation reports to the mobile clients who then use the reports to update the cached data items. Cache invalidation is indeed an effective approach to maintaining such data coherence. This paper presents a new cache invalidation strategy which is shown through experimental evaluation to maintain data consistency between the server and mobile clients in a more efficient way than existing invalidation strategies.

Key Words: Mobile Environments, Servers and Mobile Clients, Cache Invalidation, Data Consistency, Bandwidth Utilization, Access Time, Energy Consumption, Simulation and Performance Evaluation

1. Introduction

In a mobile environment, users can use mobile computers to access data without temporal or spatial limitations. To reduce data access time and bandwidth utilization for mobile communications, data items can be cached at the end of mobile clients. When the server updates its data items, the corresponding cached data at the mobile clients must be accordingly updated to maintain data correctness and consistency. The popular approach to ensuring such data coherence is cache invalidation. In cache invalidation, the server will broadcast invalidation reports (IRs) to the mobile clients who then invalidate and update their cached data items according to the reports. A number of cache invalidation strategies have been proposed in the literature (e.g. [1–7]), including the timestamp, bit-sequence, dual-report, and invalidation by absolute validity interval strategies. Some strategies, such as the timestamp algorithm [1], verify the validity of the cached data through uplink and downlink channels between the server and mobile clients. When a user needs a certain cached data item that has been invalidated, the mobile client will file a request to the server via the uplink channel and receive responses (from the server) via the downlink channel.

Cache invalidation is effective in maintaining data consistency between the server and the mobile clients. But its performance may degrade when communication between the two parties is disconnected to conserve bandwidth or power utilization. To ensure data consistency in a more efficient way (i.e., with less data access time and bandwidth utilization), this paper presents a new cache invalidation strategy. The new strategy attains data consistency at significantly reduced cost due to shrunk IR sizes, aperiodical IR broadcasting, a special mechanism using T_{lb} and no unnecessarily invalidated data items. Experimental evaluation shows that the new strategy outperforms previous strategies in such terms as the data access time, bandwidth utilization, energy consumption and so on.

*Corresponding author. E-mail: pjchuang@ee.tku.edu.tw

2. Background Study

A mobile environment contains a large number of mobile clients and a small number of database servers. The servers are connected through a wired network, while mobile clients are connected to a server through a wireless communication channel. A server will broadcast invalidation reports (IRs) to its mobile clients, periodically or aperiodically, and mobile clients will invalidate their cached data items following the content of the received IRs. An IR usually contains the data IDs or timestamps of data items but not their values. (Carrying the values will consume too much bandwidth.)

A brief survey on major cache invalidation strategies is given below to facilitate further discussion.

2.1 The Timestamp (TS) Algorithm [1]

The IR for the TS strategy includes the current timestamp (T) and a list of (o_i,t_i) -- o_i is the data ID; t_i is the recent update time for this data item. T_{lb} is the received time of the last IR for a mobile client, L is the broadcasting interval and w is the broadcasting window. The TS strategy operates as follows.

When the disconnection time between a mobile client and its server exceeds L*w, all cached data items will be invalidated. When the disconnection time is shorter than L*w: (1) If any cached item is in the IR and its cached time $t_i^c < t_i$, invalidate the item; (2) for each cached item which is not in the IR, set its t_i^c as T. If the user's query items are valid in the cache, answer by the cached items; if not, the mobile client will file requests to the server asking for the queried items. The TS strategy verifies the validity of cached data items in a rather simple way but is likely to conduct unnecessary cache invalidation: It may invalidate all cached data items including the valid ones and as a result increase subsequent cache requests, wasting the limited bandwidth resources of a wireless computing system.

2.2 The Bit-Sequence (BS) Method [2]

In the BS strategy, an IR contains a set of bit sequences, each sequence having a corresponding timestamp T_i, $(1 \leqq i \leqq n)$. Each bit represents a data item in the server. A 1 or 0 bit in a sequence indicates the item represented (by the bit) has been or has not been updated since the time specified by the timestamp. Assuming T^c is the timestamp of the last IR received by a mobile client,

(1) $T^c \geqq T_0 \rightarrow$ all cached items are valid,
(2) $T^c < T_n \rightarrow$ all cached items need to be invalidated,
(3) $T_n \leqq T^c < T_0 \rightarrow$ the sequence with timestamp T_i ($T_i \leqq T^c < T_{i-1}$) will be used to invalidate its cache.

All data items represented by the "1" bits in the sequence will be invalidated.

When a mobile client gets disconnected from the server for a long period of time, the BS strategy turns out a more desirable cache invalidation scheme as it provides more update history information (than the TS strategy). There remains a problem with the strategy: Since it is hard to predict when the IR structure will complete, more waiting time is needed.

2.3 The Dual-Report Cache Invalidation (DRCI) Strategy [3]

The DRCI strategy organizes data objects into groups and adopts two IRs: the object invalidation report (OIR) and the group invalidation report (GIR). The OIR contains the update information of data objects in the interval [T-w*L, T]. The GIR contains the update information of groups in the interval [T-W*L, T] (W > w) and the timestamp of each group. T_i, the latest timestamp of each group, is determined as follows: Ignore the update items during the [T-w*L, T] interval when determining the timestamp of a group; find the largest timestamp that is less than T-w*L. For mobile clients with shorter disconnection time, use the OIR to check the cached items directly. For those disconnected before T-w*L, use both the OIR and GIR to reduce unnecessary invalidation of all cached items (the OIR is first used to invalidate individual objects in the cache; the GIR is then used to invalidate the remaining objects whose groups have been updated).

2.4 The Invalidation by Absolute Validity Interval (IAVI) Strategy [4]

The IAVI strategy sets up an absolute validate interval (AVI) as the validity period of a cached data item. To verify the validity of a cached item, a mobile client compares the last update timestamp with the item's AVI. (The update interval is used to predict the value of the next AVI.) When the AVI expires, the cached item is invalidated. In the strategy, the server will periodically broadcast IRs to notify mobile clients about the changes of the AVIs. Invalidation can be either implicit or explicit at the side of the mobile clients.

(1) Implicit invalidation: If the sum of the update timestamp and the AVI of a cached data item is smaller than the current time, the item has passed its validity period and is assumed to be invalid. When the AVI of a cached item expires, the item goes on invalidating itself without waiting for the next IR (when the cached item is accessed).

(2) Explicit invalidation: If a cached item looks valid by the value of its AVI, the mobile client needs to consult the next received IR which contains data IDs and update time for items with changed AVIs.

The IAVI strategy is favorable because it allows most cached items to verify their validity by using AVIs and because the size of its IRs is smaller than that of the TS or BS strategy. But with explicit invalidation, a mobile client will take longer waiting time as it needs to wait for the next periodical IR.

3. The Proposed Cache Invalidation Strategy

Considering the limited resource of a mobile environment, this paper presents a new cache invalidation strategy which is able to ensure data consistency between the mobile clients and the servers with bandwidth-saving mechanism. In its operation, the new cache invalidation strategy saves the bandwidth resource by

- reducing the size of IRs and the number of involved link channels, and
- avoiding unnecessary invalidation of the cached data items.

The bandwidth-saving mechanism of the proposed strategy:

(1) If a mobile client is always in the connection mode: The server will broadcast aperiodical IRs to the mobile client to maintain data consistency between the two parties. Thus when a user makes a query, the mobile client can provide the requested and updated information at the earliest convenience. Note that if the server broadcasts periodical IRs at a certain time interval, users who make queries between two broadcasting intervals may not get the needed information -- unless the mobile client can verify the validity of the cached items.

(2) If the mobile client receives queries when it gets reconnected after T_{lb}: The mobile client will uplink the value of T_{lb} and all query items which are not in the cache memory to the server. The server then broadcasts the IR which contains only the IDs of the updated data items since T_{lb} and the latest update timestamp T. It is in this way that the IRs in our strategy can be reduced to a size smaller than that of the other strategies.

The specific way to handle a user's query:

(1) If the queried items are not in the cache memory: The mobile client will file data requests to the server.

(2) If the queried items are in the cache and their validity is certain: Use the cached data if they are valid; file the data request to the server otherwise.

(3) If the queried items are in the cache memory but their validity is uncertain (due to disconnection): The mobile client needs to verify the validity of all cached items by IRs. The mobile client will uplink T_{lb} and the queried items which are not in the cache memory to the server through the uplink channels. When the server receives T_{lb}, it will check to see how many IRs the mobile client has missed during the disconnection period. If no data items have been updated, i.e., no IRs are broadcast, during the disconnection period, the cached items in the mobile client remain valid and the server will downlink only queried items that are not in the cache memory to the client. If in the disconnection period the mobile client does miss some IRs, the server needs to broadcast, besides the queried items that are not in the cache memory, an IR containing

the IDs of the updated data items since T_{lb} and the latest update timestamp T. The mobile client then uses these received items to answer the query, invalidate the updated data items since T_{lb}, and set its T_{lb} to T.

As we can see, the new cache invalidation strategy can help mobile clients restore IRs (lost during the disconnection period) using only small uplink channel bandwidth (T_{lb}) and then use the IRs to respond to a user's query without delay (without waiting for the next periodical IR). Such a bandwidth-saving mechanism is indeed valuable for mobile environments whose bandwidth resource is restricted.

To make the most of the restricted bandwidth resource, when the server broadcasts an IR to a mobile client who uplinks T_{lb}, every other mobile client in the same area will also receive it. For example, assume A, B and C are three mobile clients in the same area. When the server broadcasts an IR to A upon request, B and C will also receive it. As a mobile client always keeps T_{lb} even in the disconnection mode, thus when B (or C) receives an IR in this way (i.e., without uplinking T_{lb}), it can compare the stored T_{lb} with T in the received IR. If T_{lb} = T, the cached items in B (or C) remain valid. If T_{lb} < T, the cached items need to be updated according to the received IR (to maintain validity) and the value of T_{lb} will be set to T.

4. Performance Evaluation

The new cache invalidation strategy is able to maintain data consistency between mobile clients and the server in a more efficient way as it employs small-sized IRs, aperiodical IR broadcasting and a special mechanism using T_{lb}. Experimental evaluation conducted to compare the performance of our strategy and other strategies (including the TS, BS, DRCI and IAVI) shows that our strategy performs constantly better due to its ability to avoid unnecessary invalidation and reduce bandwidth consumption.

4.1 The Simulation Model

The simulation model comprises the server (with 100,000 data items), mobile clients (with 5,000 items in each client's cache), uplink channels and downlink channels. One server attends to 10 mobile clients and only data items in the server will be updated. Users will put queries to the mobile clients who then file the requests to the server via the uplink channels and receive results (from the server) via the downlink channels. The mean user query arrival time is 0.5 sec. 10% of the data items in the server forms a hot update set which covers 90% of the queried requests. The mean update interval is 1860 sec; the mean disconnection time is 1000 sec. The uplink and downlink channel bandwidth is respectively 19.2k bps and 100k bps. The size of a data item is 256 bits, its timestamp 64 bits and the data ID 17 bits. (Each group of the DRCI strategy has 100 items and the group ID is 10 bits.) The broadcasting interval is 30 sec. The window for broadcasting an IR is 10 intervals.

4.2 Simulation Results

The adopted performance parameters include the size of each data item in IRs, the number of unnecessarily invalidated data items, the number of cache requests filed to the server, bandwidth utilization, cache miss ratios, access time and energy consumption.

- **The size of each data item in IRs**

Figure 1 shows that among the five cache invalidation strategies under evaluation, our strategy (indicated as NEW) produces the smallest data items in IRs -- because each item contains only the ID. (Note that the small

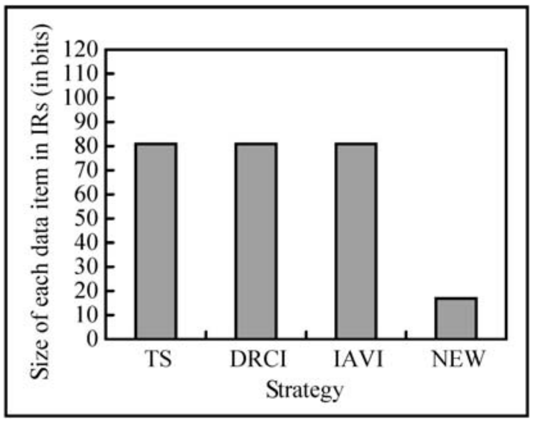

Figure 1. The size of each data item in IRs for various strategies.

size of data items in IRs can be translated into reduced downlink channel bandwidth utilization, to be discussed later.)

- **The number of unnecessarily invalidated data items**

Unnecessarily invalidated data items refer to items which are still valid but are "falsely" invalidated with other items. The number of unnecessarily invalidated data items is related to the disconnection lengths between the mobile clients and the server and may seriously waste bandwidth resources. Figure 2 shows that our strategy generates no unnecessarily invalidated data items due to its aperiodical IR broadcasting and special mechanism using T_{lb}. The result also indicates no "falsely" invalidated data items for the other strategies when the disconnection time is less than 100 sec -- because the periodical IRs broadcast by the server can verify all cached items in that time zone. But when the disconnection time rises over 1000 sec, these strategies begin to lose some IRs. To make up for the loss (of IRs), they simply invalidate all cached items, including those still valid ones. When users make new queries, the mobile clients must go to the server asking for the queried information. It is very likely that the users' requests involve data items that have been falsely invalidated, which makes the caching process redundant and resource-wasting.

Figure 2 also exhibits more unnecessarily invalidated data items at disconnection time 1000 sec than at 10000 sec. This is because most cached items will have been "necessarily" invalidated at disconnection time 10000 secretary.

- **The number of cache requests filed to the server**

A mobile client needs to send cache requests to the server in two situations: (1) when the queried data item is not stored in the cached memory, and (2) when the cached items have been invalidated. Figure 3 shows that the first situation usually happens at disconnection time < 100 sec. During such a short disconnection time (when data items stored in the cache of a mobile client still retain their validity), the mobile client will uplink a request to the server only when the queried item is not stored in its cached memory. In this situation, every cache invalidation strategy (including ours) will simply send the cache request to the server. On the other side, when a mobile client is disconnected from the server longer than 1000 sec, the other strategies will uplink cache requests to the server more frequently than our strategy. This is because they may have unnecessarily invalidated some data items in the cache during the long disconnection, while our strategy generates no unnecessary invalidation at all.

- **Bandwidth utilization**

The downlink channel bandwidth will be consumed when the server broadcasts IRs and the requested cache information to the mobile clients. In Figure 4, the BS

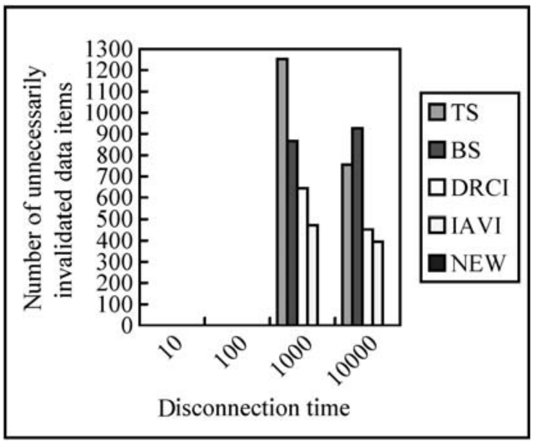

Figure 2. The number of unnecessarily invalidated data items vs. disconnection time.

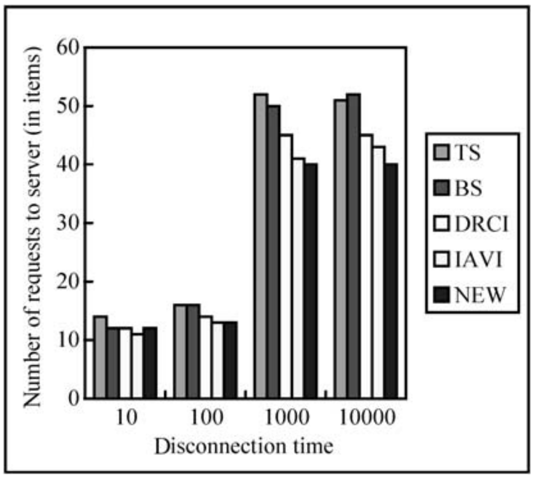

Figure 3. The number of cache requests filed to the server vs. disconnection time.

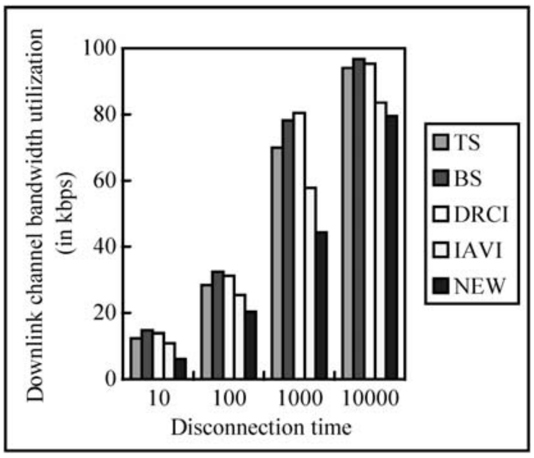

Figure 4. Downlink channel bandwidth utilization vs. disconnection time.

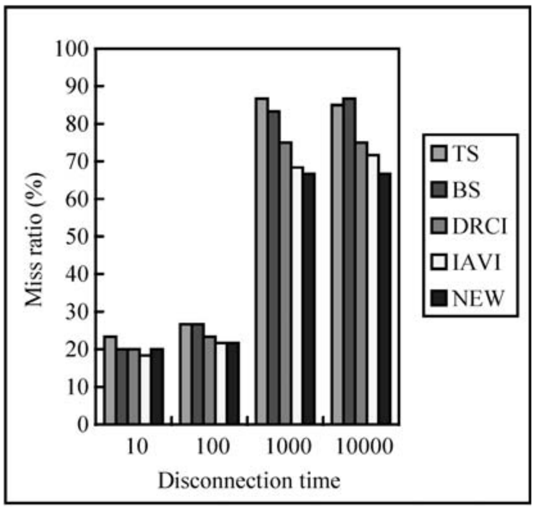

Figure 5. Cache miss ratio vs. disconnection time.

strategy is found to consume the largest downlink bandwidth due to its large-sized IRs. The next top consumers include the TS and DRCI strategies, which also require large-sized IRs. Our new strategy, by contrast, employs the least amount of downlink channels -- partly because it has the smallest IRs and partly because it generates no unnecessary invalidation after disconnection.

- **The cache miss ratio**

The cache miss ratio is the number of inaccessible or invalid data items in the cache over all queried data items. It is obvious that a cache invalidation strategy with low cache miss ratios can handle users' queries more quickly. Our strategy is shown in Figure 5 to yield the lowest cache miss ratios at longer disconnection time, such as 1000 sec and 10000 sec -- the more practical situations in a mobile environment. This is because without unnecessary invalidation of the cached items, our strategy can respond to users' queries more promptly, thus attaining higher cache hit ratios (i.e., lower miss ratios) than the other strategies.

- **The access time**

The access time is collected versus a few parameters, such as the disconnection time, the query arrival time, the update interval, the server database size, and the downlink and uplink channel bandwidth. The access time is the time elapsed since a mobile client submits a request to the server until he downloads all the requested data items [3]. A perfect server (PS) strategy, in which the server is aware of the contents of all cached items and broadcasts IRs containing only the updated information of the cached items, is added in the simulation to serve as an optimum. Simulation results collected under all parameters show that our proposed strategy yields better performance, i.e., less access time, in nearly all cases.

To give an example, the result of the access time vs. the disconnection time is presented in Figure 6. In the figure, the disconnection time varies from 10 to 10000 sec and the request arrival interval is 0.5 sec. The access time of all strategies remains unaffected when the disconnection time < 1000 sec -- because at this point the strategies can correctly verify the cached items by the broadcast IRs. When the disconnection time grows from 1000 to 10000 sec, the access time then rises for all strategies. This is because most cached data items are invalidated during the long disconnection, and as a result the mobile clients need to file almost all the requests to the server when users send in cache requests.

- **Energy consumption**

Energy consumption indicates the amount of energy consumed in invalidating cache items, uplinking requests and downloading the desired data [3]. In our simulation, energy consumption involves (1) the size of IRs, (2) the total cost for mobile clients to uplink the T_{lb} values or the requested data items, and (3) the total data items the ser-

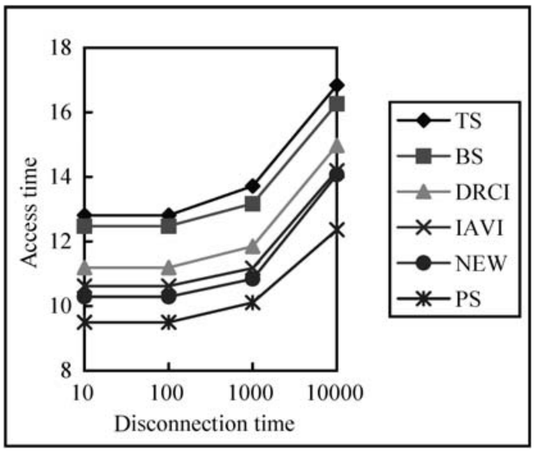

Figure 6. Access time vs. disconnection time.

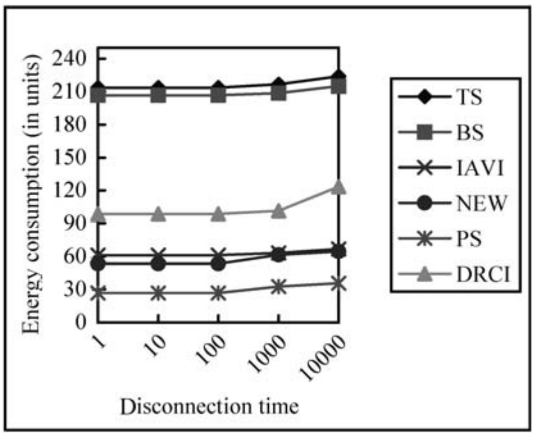

Figure 7. Energy consumption vs. disconnection time.

ver downlinks to the mobile clients. Based on [3,8], the energy consumed in receiving 1k-bits of information is 1 unit; transmitting data consumes 10 times more energy than receiving data. As Figure 7 illustrates, energy consumption rises for all strategies when the disconnection time grows. This is because more cached items are invalidated in longer disconnection, which will result in busier data transmission (including more IRs) in both the downlink and uplink channels when users send in cache requests. The result in Figure 7 indicates the lowest energy consumption for our strategy when the disconnection time < 1000 sec. When the disconnection time grows over 1000 sec, our strategy still consumes the least energy, but the amount comes quite closer to that of the IAVI strategy.

5. Conclusion

In a mobile environment, caching data items at the mobile clients is important as it can reduce the data access time and save bandwidth utilization. To achieve effective caching, when the server updates its data items, the corresponding cached items at the mobile clients must be updated accordingly to maintain data correctness and consistency. A number of cache invalidation strategies have been proposed in the literature to maintain such data consistency. In performing cache invalidation, the server will broadcast data invalidation reports (IRs) to the mobile clients who then invalidate and update their cached items based on the contents of the reports. To ensure data consistency between the server and the mobile clients in a more efficient way, this paper introduces a new cache invalidation strategy. The new strategy consists of such features as reduced IR sizes, aperiodical IR broadcasting, a special mechanism using T_{lb} and no unnecessarily invalidated items, which help maintain data consistency (between the server and the mobile clients) at reduced cost. In fact, the new cache invalidation strategy is shown through experimental evaluation to outperform previous strategies in terms of the data access time, bandwidth utilization, energy consumption and so on.

Acknowledgement

A preliminary version of this paper was presented at the 18th International Conference on Advanced Information Networking and Application, Mar. 2004.

This work was supported in part by the National Science Council, Taiwan, R.O.C., under Grant No. NSC91-2213-E-032-013.

References

[1] Hu, Q. and Lee, D. L., “Adaptive Cache Invalidation Methods in Mobile Environments,” *Proc. Int’l Conf. on High Performance Distributed Computing,* pp. 264–273 (1997).

[2] Jing, J., Elmagarmid, A., Helal, A. and Alonso, R., “Bit-Sequences: An Adaptive Cache Invalidation Met-

hod in Mobile Client/Server Environments," *Mobile Networks and Applications,* Vol. 31, pp.115–127, (1997).

[3] Tan, K.-L., Cai, J. and Ooi, B. C., "An Evaluation of Cache Invalidation Strategies in Wireless Environments," *IEEE Tran. on Parallel and Distributed Systems,* Vol. 12, pp. 789–807 (2001).

[4] Chan, E., Yuen, C.-H., Lam, K.-Y. and Leung, H. W, "An Adaptive AVI-Based Cache Invalidation Scheme for Mobile Computing Systems," *Proc. Int'l Conf. on Database and Expert Systems Applications* (2000).

[5] Wu, K.-L., Yu, P. S. and Chen, M.-S., "Energy-Efficient Caching for Wireless Mobile Computing," *Proc. Int'l Conf. on Data Engineering,* pp. 336–343 (1996).

[6] Nam, S. H., Chung, I. Y. and Hwang, C.-S., "An Efficient Cache Invalidation Scheme for Mobile Wireless Environments," *Proc. Int'l Conf. on Parallel and Distributed Systems,* pp. 289–296 (2001).

[7] Jing, J., Helal, A. S. and Elmagarmid, A., "Client-Server Computing in Mobile Environments," *ACM Computing Surveys,* Vol. 31, pp. 117–157 (1999).

[8] Forman, G. H. and Zahorjan, J., "The Challenges of Mobile Computing," *Computer,* Vol. 27, pp. 38–47 (1994).

Manuscript Received: Mar. 1, 2006
Accepted: Sep. 1, 2006