Cache Coherency in Location-Dependent Information Services for Mobile Environment

Jianliang Xu, Xueyan Tang, Dik Lun Lee, and Qinglong Hu *

University of Science and Technology, Clear Water Bay, Hong Kong
{xujl, tangxy, dlee, qinglong}@cs.ust.hk

Abstract. Caching frequently accessed data at the client is an attractive technique for improving access time. In a mobile computing environment, client location becomes a piece of changing information. As such, location-dependent data may become obsolete due to data updates or client movements. Most of the previous work investigated cache invalidation issues related to data updates only, whereas few considered data inconsistency caused by client movements. This paper separates location-dependent data invalidation from traditional cache invalidation. For location-dependent invalidation, several approaches are proposed and their performance is studied by a set of simulation experiments. The results show that the proposed methods substantially outperform the NSI scheme which drops the cache contents entirely when hand-off.

1 Introduction

Mobility opens up new classes of applications in a mobile computing environment. Location-dependent information service is one of these applications which are gaining increasing attention [5,1,9]. Through location-dependent information service, mobile clients can access location sensitive information, such as traffic report, hotel information and weather broadcasting, etc. The Advanced Traveler Information Systems($ATIS$) project has explored this in depth [11]. With recent development in micro-cell/pico-cell systems, precise location-dependent information is available, making applications which require precise location data (e.g., navigation maps) a reality.

Caching is a widely used technique in mobile computing environments to improve performance [3,2,7]. However, frequent client disconnections and movements across cells make cache invalidation a challenging issue [2]. A lot of work has been done on cache invalidation [3,8,6,4,12]. Most of the previous work studied the cache consistency problem incurred by data updating (hereafter called *temporal-dependent update*). In a mobile computing environment, besides the temporal-dependent updates, cache inconsistency can also be caused by location changing (hereafter called *location-dependent update*). Location-dependent

* The author is now with Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada.

H.V. Leong et al. (Eds.), MDA'99, LNCS 1748, pp. 182–193, 1999.

updates refer to the case where the cached data become obsolete when the mobile client moves. Therefore, cache invalidation should be performed for both temporal-dependent and location-dependent updates. Temporal-dependent and location-dependent invalidations affect each other. For example, if the temporal-dependent update rate is much higher than the location-dependent update rate, the temporal-dependent updates will dominate cache invalidation, and vice versa.

This paper investigates the integration of temporal-dependent and location-dependent invalidation. For location-dependent invalidation, we propose Bit Vector with Compression (BVC), Grouped Bit Vector with Compression ($GBVC$), and Implicit Scope Information (ISI) methods. For temporal-dependent cache invalidation, we adopt the AAW_AT (Adaptive Invalidation Report with Adjusting Window) method [6]. Experiments are conducted under various system parameter settings. When both query throughput and uplink cost are considered, all the three location-dependent invalidation strategies outperform the No Scope Information (NSI) method which drops the cache contents entirely when hand-off. In particular, ISI and $GBVC$ are close to the optimal strategy and better than BVC. The $GBVC$ scheme is the best for most system workloads.

The rest of this paper is organized as follows. Section 2 describes some assumptions in this study. Section 3 proposes three location-dependent invalidation schemes. The simulation model and experimental results are presented in Section 4 and 5, respectively. Finally, Section 6 concludes the paper.

2 Assumptions

The geographical coverage area for the information service is partitioned into *Service Areas*, and each service area is attached with a data server. The service area may cover one or multiple cells. Each service area is associated with an ID (SID) for identification purposes. The SID is broadcast periodically to all the mobile clients (MCs) in that service area.

The database associated with each service area is a collection of data items. The *valid scope* of an item value is defined as the set of service areas where the item value is valid. All valid scopes of an item form its *scope distribution*. A scope distribution may be shared by several data items. In a large-scale information system, the number of scope distributions can be very large. Every data server keeps a complete copy of the database, i.e., the same data items are replicated on all the data servers but probably with different values in different data servers. Data servers or MSSs are assumed to know the scope of each data item. When a mobile client moves into a new service area, validity checking of cached data needs to be carried out. When a data item is updated, there is a certain delay, δ, involved to maintain consistency for replicas. For simplicity, we assume that $\delta = 0$. Thus, we do not need to consider the consistency problem caused by update delays.

3 Cache Invalidation Methods

In general, there are two types of cache invalidation strategies, namely *client-initiated method* and *server-initiated method*. In the client-initiated method, the client monitors the states of the cached items and initiates the validity checking procedure. In the server-initiated method, the server monitors the states of the cached items and informs the client to purge the obsolete data. Since temporal-dependent invalidation is usually carried out at the server side, it is difficult for the client to know whether cached data are valid or not. As such, the server-initiated method is usually adopted. In contrast, location-dependent invalidation is caused by the movement of the client. Therefore, it is hard for the server to know the state of the client cached data and the client-initiated method is a simpler approach.

3.1 Temporal-Dependent Invalidation

For temporal-dependent invalidation, periodic broadcasting of invalidation reports at the server side is an efficient strategy [3,8]. Every mobile client, if active, listens to the reports and invalidates its cache accordingly. In the various invalidation report approaches, adaptive cache invalidation algorithms work well under various system workloads [6]. In adaptive cache invalidation algorithms, the next invalidation report (IR) is dynamically decided based on the system workload.

3.2 Location-Dependent Invalidation

As discussed above, temporal-dependent cache invalidation is initiated by the data server. On the other hand, location-dependent cache invalidation is client-initiated. In order to validate the cached data, the mobile client can send the IDs of the cached items uplink to the data server. The data server decides whether these items are still valid according to the mobile client's current location and sends the checking result back to the client. This method is simple. However, it increases network traffic substantially due to the expensive send-up-checking. To achieve better performance, it is better to attach scope information to the data. In that case, invalidation efficiency depends heavily on the organization of the scope information. In the following paragraphs, we propose several efficient methods to build scope information.

Bit Vector with Compression (BVC) Recall that every service area is associated with an SID to distinguish it from the others. BVC uses a bit vector to record scope information. The length of the bit vector is equal to the number of service areas in the system and every cached data item is associated with a bit vector. A "1" in the nth bit vector indicates that the data item value is valid in the nth service area while "0" means it is invalid in the nth service area.

For example, if there are 12 service areas in the system, then a bit vector with 12 bits is constructed for each cached data item. If the value of the bit vector for a data item is 000000111000, it means this data item value is valid in the 7th, 8th, and 9th service areas only.

Obviously, the overhead still would be significant when the system is large. Noticing the locality of validity scope, it is possible to perform compression on the bit vector.

Grouped Bit Vector with Compression (*GBVC*) To remedy the significant overhead in BVC, we can build a vector corresponding to only part of the service areas. The reason is two-fold. Firstly, the mobile client seldom moves to a service area that is far away from its home service area. Consequently, partial data scope information is sufficient. Secondly, even if the mobile client moves to a distant service area, it takes quite some time for the client to do so. During this period, the data item may have already been updated on the data server, and therefore the complete scope information about distant service areas is useless. Hence, it is more attractive to store the data item scope information of only adjacent service areas. This is where the idea of Grouped Bit Vector with Compression scheme ($GBVC$) comes from.

The whole geographical area is divided into disjoint districts and all the data service areas within a district form a group. Here, the service area ID (SID) consists of two parts: *group ID* and *intra-group ID*. Scope information attached to a data item includes the group ID and a bit vector (BV) which corresponds to all the service areas within the group. Thus, an attached bit vector has the following form: (group ID, BV).

With the same example used in the previous section, the whole geographical area is further assumed to be divided into two groups, such that service areas 1-6 form group 0 and the rest form group 1. With the $GBVC$ method, one bit is used to construct the group ID and a six-bit vector is used to record the service areas in each group. For the data item mentioned earlier, in group 0, the attached bit vector is $(0, 000000)$; in group 1, the attached bit vector is $(1, 111000)$. As can be seen, the overhead for scope information is reduced in the $GBVC$ method.

When the mobile client checks the data item's validity, it compares the group ID of the current service area with the one associated with the cached data item. If they are not the same, the data item will be invalidated. Otherwise, the client checks the *intra-group ID*th bit in the bit vector to determine whether the cached item should be invalidated.

Implicit Scope Information (*ISI*) This strategy divides the database into multiple logical sections. Data items with the same valid scope distribution are placed in the same section. Hence, data in the same section share the same location validation information. Different logical sections imply different scope information.

Each possible valid scope of the data item value is specified by a 2-tuple (SDN, SN), where SDN is the scope distribution number and SN denotes the

SID	1	2	3	4	5	6	7	8	9	10	11	12
(SDN)Scope Distribution #1	1	2	3	4	5	6	7	8	9	10	11	12
(SDN)Scope Distribution #2	1			2			3			4		
(SDN)Scope Distribution #3	1		2			3		4			5	

Fig. 1. An Example with Different Distributions

scope number within this distribution. The 2-tuple is attached to the data item during delivery. The cached item i has the format $\{D_i, SDN_i, SN_i\}$, where D_i is the item value. For example, suppose there are three types of scope distributions (see Figure 1) and data item 4 has distribution 3. If item 4 is cached from service area 6 (i.e., $SID = 6$), then $SDN_4 = 3$ and $SN_4 = 3$. That implies that the cached item 4's value is valid in the service areas 6 and 7 only.

The location-dependent report consists of the valid SNs for each scope distribution and the SID of that service area. For example, in service area 8, the MSS broadcasts {8, 3, 4, 8} to the clients, where the first three numbers are the SN values for the three scope distributions and the last number is the SID of the current service area. In this way, scope information delivered along with the data item can be greatly reduced.

After retrieving this location-dependent report, for item i, the client compares the cached item's SN_i with the SDN_ith SN in the location-dependent report received. If they are the same, the cached item value is valid. Otherwise, the data item value is invalid. For example, in service area 8, the client checks for the cached data item 4 whose $SDN_4 = 3$ and $SN_4 = 3$. In the broadcast report, the SDN_4th (i.e. 3rd) SN equals to 4. Therefore, the client knows that data item 4's value is invalid.

4 Simulation Model

This section describes the simulation model used to evaluate the performance of the cache invalidation methods proposed in the previous sections. The performance metrics used are system throughput (i.e., number of queries answered per interval per cell) and average uplink cost for answering a query.

4.1 System Model

The system consists of $CellNumber$ (i.e. $CellNumber_x \times CellNumber_y$) cells with hexagonal shapes and $UserNumber$ users. In the simulation model, every cell is assumed to be a service area. Therefore, client moving across service areas is the same as hand-off. Initially, the users are uniformly distributed in all $CellNumber$ cells, i.e. each cell has $UserNumber/CellNumber$ users. During the experiment, every user moves between cells independently according to the same movement pattern.

An MSS serves as a data server for each cell. The database of a data server contains $DatabaseSize$ items and is uniformly divided into $Scope\_N$ classes. The data items in one class have the same valid scope distribution. In other words, there are $Scope\_N$ valid scope distributions in a database. The mean size of a data item's valid scope is $ScopeSize_i (i = 1, 2, \ldots, Scope\_N)$.

The network is modeled as a *preempt resume* server with invalidation report and location-dependent report having the highest broadcast priority and the rest of the messages having the same priority. All other messages are served on an $FCFS$ basis. The bandwidth is always fully utilized so that the throughput represents the true efficiency of the algorithm. Table 1 shows the default system parameter settings used in the experiment.

Table 1. The default system parameter settings.

Parameter	**Setting**	**Parameter**	**Setting**
Simulation Time	200000 seconds	CellNumber	4096(64x64) cells
User Density	20 mobile clients/cell	Database Size	1000 data items
Data Item Size	1024 bytes	ScopeClass	10
Broadcast Period	20.0 seconds	Control Message Size	64 bytes
Downlink Bandwidth	1000 bps	Mean ScopeSize for	1, 4, 4, 16, 16, 64,
Uplink Bandwidth	1000 bps	Each ScopeClass	64, 256, 1024, 4096

4.2 Client Execution Model

Each client is simulated by a process and runs a continuous loop that generates streams of queries and hand-offs independently. The queries are read-only and are location-dependent in the sense that the query results reflect the current location (cell) of the mobile user. Each query is separated from the completion of the previous query by either an exponentially distributed $Think\ Time$ or an exponentially distributed $Disconnection\ Time$. In the model, each client has a probability p to enter the disconnection mode at the beginning of each broadcast interval. If the desired data items have a valid copy in the client, then an immediate reply is returned. Otherwise, the client sends the data IDs uplink to the server. The server returns the data by broadcasting them on the channel. Once the requested data items arrive, the client retrieves them off the channel and stores them in the cache. The cache page replacement policy is the LRU scheme.

The time that a mobile client spends in a cell follows an exponential distribution with mean time of $Residence\ Time$. After the client stays in a cell for a while, it moves to another neighboring cell. The destination cell is chosen based on uniform probability (i.e. 1/6 probability for each of the six neighbors). All mobile clients follow the same move pattern independently.

The client query access pattern follows hot-spot and cold-spot with the setting of 80/20, that is, 80% queries access first 100 hot data items and 20% queries

access the rest of the database. In both cases, queries are uniformly distributed in the different scope classes. In reality, the user may access a hint of data items whose values are "correct" for another location (cell). For simplicity, it is assumed that no such queries exist and all accesses are directed to the local server. Table 2 shows the default parameter settings for mobile clients.

Table 2. Default parameter settings for mobile clients.

Parameter	Setting	Parameter	Setting
Client Cache Size	10% of database size	Mean Think Time	50.00 seconds
Mean Residence Time	600.00 seconds	Mean Disconnection Time	400.00 seconds
Mean Data Items Ref. by a Query	10 data items	Prob. of Client Disc. per Interval	0.1

4.3 Server Execution Model

The server is simulated by a process which generates a stream of updates with an exponentially distributed *Update Inter-arrival Time*. Unlike queries, all updates are assumed to be randomly distributed across the whole database. For data items with several "correct" values in different cells, the updates arrive at those data servers at the same time. Thus, the updates happen simultaneously for these data items. Furthermore, the server broadcasts invalidation reports and location-dependent reports, which consists of SID, SNs etc., periodically to the clients. The default parameters used to describe the server model are given in Table 3.

Table 3. Default parameter settings for data servers.

Parameter	Setting
Mean Update Inter-arrive Time	500.00 seconds
Mean Data Items Updated Each Time	5 data items

5 Experiment Results

The simulation is carried out using *CSIM* [10]. We collect data from the first 64th (i.e. 8×8) cells after the system becomes stable (after 100000 seconds in the simulation). Various experimental results are presented in this section. Besides the strategies proposed in the previous sections, two additional strategies, namely No Scope Information (NSI) and *Optimal*, are also included for comparison. NSI represents the strategy in which no extra mechanism is used for location-dependent cache invalidation, i.e., the client simply drops the entire cache after

hand-off. *Optimal* denotes an ideal strategy in which the client has complete valid scope information. We evaluate the $GBVC$ strategy for various group sizes and find that 64 is a reasonable choice under our experiment settings. Therefore, we only represent the data collected from the $GBVC$ strategy with group size 64 in this section. For the BVC strategy, the system performance is evaluated under compression ratio 1:1 and 1:2.5, denoted by BVC and $BVC0.4$ respectively.

All of the above strategies are evaluated under the NC scheme for validity checking time, where after hand-off the MC does not make cache invalidation until the query is issued. Both temporal-dependent and location-dependent cache invalidation are realized in the simulation. For temporal-dependent cache invalidation, AAW_AT, which shows the best performance in the absence of location-dependent invalidation [6], is adopted.

5.1 Performance for Changing Mean Residence Time

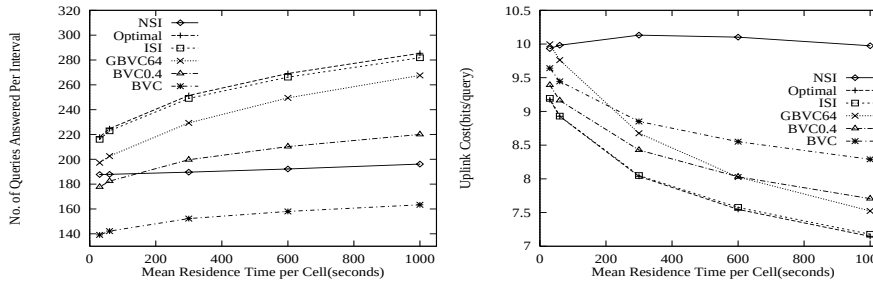

Fig. 2. Performance for Changing Mean Residence Time

System performance is first investigated by varying mean client residence time in a cell from 30 seconds to 1000 seconds. The shorter the duration time in a cell, the more frequent the client hand-off. As shown in Figure 2, ISI and $GBVC64$ improve the throughput and uplink cost substantially over NSI. In particular, ISI performs almost the same as *Optimal*. Although BVC makes use of the additional valid scope information and achieves less uplink cost (due to a higher cache hit ratio[1]) than NSI, its system throughput is worse than NSI. The reason is that BVC introduces too much overhead on the downlink channels and cache storage. On the other hand, due to less overhead, $BVC0.4$ shows better performance than NSI when hand-off occurs less frequently.

All the strategies except NSI are sensitive to client hand-off. When hand-off occurs frequently, the client can easily move out of the valid scope of its cached items. Thus, the effectiveness of location-dependent cache invalidation decreases. When hand-off becomes less frequent (i.e., duration is greater than 600s), all the curves become flat. The reason is that, in this case, location-dependent in-

[1] We do not plot the cache hit ratio due to the space limitation. It is observed that the uplink cost is proportional to the cache miss ratio.

validation seldom occurs in the mobile client and, therefore temporal-dependent invalidation dominates cache invalidation.

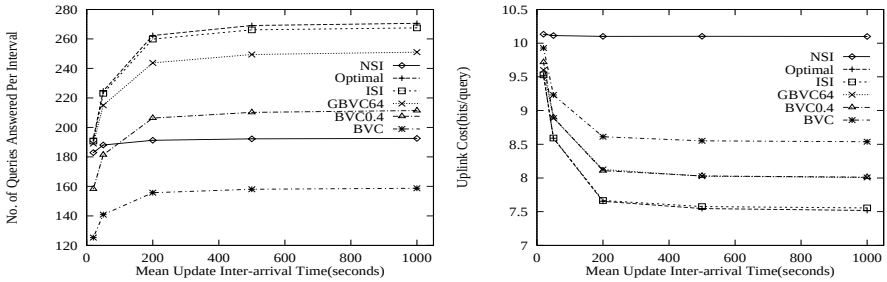

Fig. 3. Performance for Changing Mean Data Update Rate

5.2 Performance for Changing Mean Data Update Rate

Figure 3 presents the results when data update inter-arrival time is varied. Similar to the previous experiments, ISI has the best performance in terms of both throughput and uplink cost and $GBVC64$ is second followed by $BVC0.4$. When updates become intensive (left part of the figures), temporal-dependent invalidation occurs more frequently. Thus, even if the client preserves valid items in the new cell after hand-off, these item values will be purged soon due to frequent temporal-dependent invalidation. Location-dependent invalidation in this case is not a dominant factor in cache invalidation. Hence $GBVC$ and ISI have a similar performance as NSI when the data update rate is high. On the other hand, as the data update rate decreases, $GBVC$ and ISI improve the performance greatly and all the curves become flat. The reason is that the data item values kept after hand-off become more useful and the effect of location-dependent cache invalidation is much more significant than temporal-dependent invalidation.

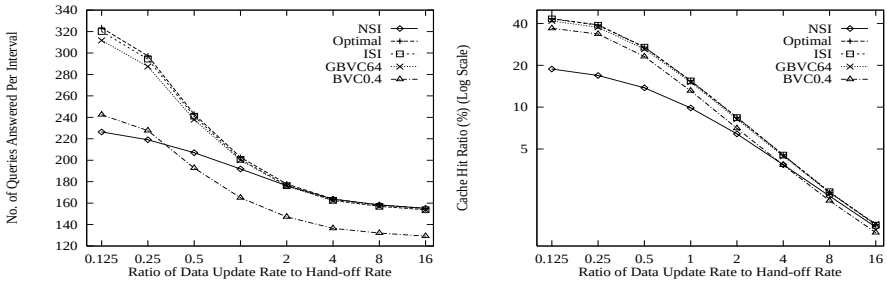

Fig. 4. Performance for Changing Ratio of Data Update Rate over Hand-off Rate

5.3 Impact of Data Update Rate and Hand-off Rate

To view the effect of relative intensity between temporal-dependent update and location-dependent update, we vary the ratio of mean data update rate to hand-off rate from 0.125 to 16 as shown in Figure 4. When the data update rate is lower than the hand-off rate, i.e., the ratio is less than one, except for the $BVC0.4$ method, all location-dependent cache invalidation schemes outperform the NSI approach. Due to the high overhead, the performance of the $BVC0.4$ method is worse than the NSI approach when the ratio is greater than 0.3. When the ratio is greater than one, i.e., data update rate becomes high, the location-dependent cache invalidation schemes lose their advantages over the NSI approach. This suggests that location-dependent cache invalidation is desired only when the mean data update rate is low compared with the hand-off rate.

5.4 Impact of System Capacity

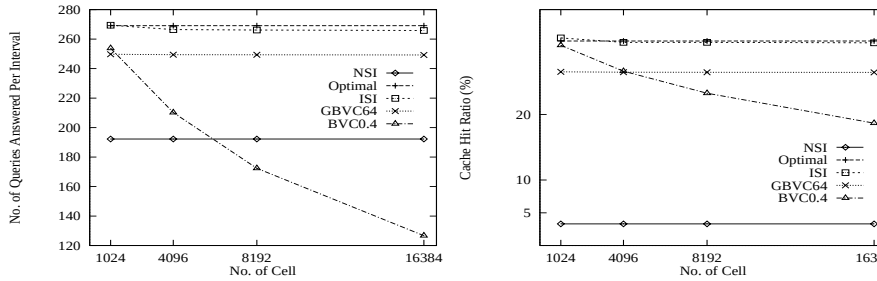

Fig. 5. Performance for Changing Cell Number

In this subsection, the scalability of the location-dependent strategies is evaluated by varying the cell number from $1024(32 \times 32)$ to $16384(128 \times 128)$ (see Figure 5). The performance of $BVC0.4$ deteriorates rapidly as the number of cells increases, whereas the other schemes are hardly affected. It is interesting to notice that although the cache hit ratio of $BVC0.4$ is higher than that of NSI when there are 16384 cells in the system, it still performs worse than NSI in terms of throughput. This is due to the high overhead of $BVC0.4$.

5.5 Impact of Changing Scope Distribution Number

For a large-scale information system, the scope distribution number might be very large. The adaptability of different location-dependent strategies to the scope distribution number is measured in this set of experiments (refer to Figure 6).

The performance of the ISI method depends heavily on the number of scope distributions. When the scope distribution number increases, the system

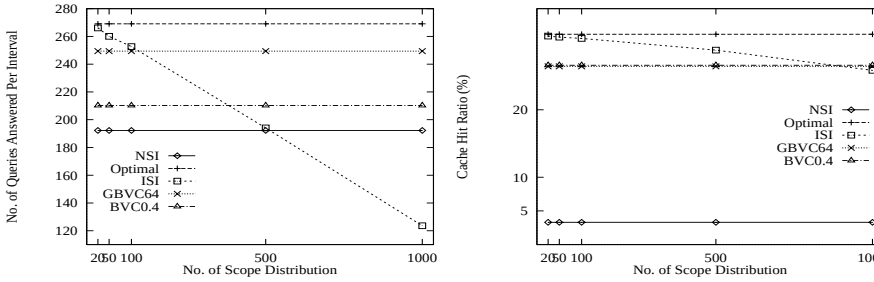

Fig. 6. Performance for Changing Scope Distribution Number

throughput decreases dramatically. On the contrary, all the other schemes remain the same as they do not rely on this system parameter.

In summary, the proposed location-dependent schemes are able to enhance system performance significantly. While the BVC method cannot scale up to system capacity and the ISI scheme does not perform well in a system with a large number of scope distributions, the $GBVC$ method shows the best stability and scalability.

6 Conclusion

In this paper, we have studied cache coherency issues in the context of location-dependent information services and proposed several cache invalidation schemes. Cache invalidation issue caused by both temporal- and location-dependent updates has been addressed. For location-dependent updates, several cache invalidation schemes have been proposed. They differ from each other in scope information organization.

We have conducted a set of experiments to evaluate these schemes in the scenario where temporal-dependent and location-dependent updates coexist. It is found that the proposed strategies are able to achieve a much better performance than the baseline scheme which drops the cache contents entirely when hand-off. While BVC cannot scale up to system capacity and the ISI scheme does not perform well in a system with a large number of scope distributions, the $GBVC$ method shows the best stability and scalability. Moreover, we observe that when the data update rate is much higher than the client hand-off rate, location-dependent cache invalidation may not be necessary.

For future work, various invalidation schemes can be evaluated under a one-dimension cellular model and a non-uniform user movement pattern etc. Moreover, a location-dependent cache invalidation scheme which considers temporal-dependent update frequencies is currently under investigation.

Acknowledgment

The authors would like to thank Dr. Tin-Fook NGAI for his valuable comments.

References

1. A. Acharya, B. R. Badrinath, T. Imielinski, and J. C. Navas. A www-based location-dependent information service for mobile clients. In *The 4th International WWW Conference*, January 1994.
2. B. R. Badrinath and T. Imielinski. Replication and mobility. In *Proc. of the 2nd Workshop on the Management of Replicated Data*, pages 9–12, 1992.
3. D. Barbara and T. Imielinksi. Sleepers and workaholics: Caching strategies for mobile environments. In *Proceedings of SIGMOD'94*, pages 1–12, May 1994.
4. B. Y. Chan, A. Si, and H. V. Leong. Cache management for mobile databases: Design and evaluation. In *Proceedings of 14th ICDE*, pages 54–63, 1998.
5. M. H. Dunham and V. Kumar. Location dependent data and its management in mobile databases. In *Proceedings of 9th International Workshop on Database and Expert System Applications*, pages 414–419, 1998.
6. Q. L. Hu and D. L. Lee. Cache algorithms based on adaptive invalidation reports for mobile environments. *Cluster Computing*, 1(1):39–48, Feb. 1998.
7. Q. L. Hu, D. L. Lee, and W.-C. Lee. Data delivery techniques in asymmetric communication environments. In *Proceedings of MobiCom'99*, August 1999.
8. J. Jing, O. Bukhres, A. K. Elmargarmid, and R. Alonso. Bit-sequences: A new cache invalidation method in mobile environments. *MONET*, 2(II), 1997.
9. V. Persone, V. Grassi, and A. Morlupi. Modeling and evaluation of prefetching policies for context-aware information services. In *Proceedings of MobiCom'98*, pages 223–231, October 1998.
10. H. Schwetman. *CSIM user's guide (version 18)*. MCC Corporation, http://www.mesquite.com, 1992.
11. S Shekhar, A. Fetterer, and D.-R. Liu. Genesis: An approach to data dissemination in advanced traveler information systems. *IEEE Data Engineering Bulletin*, 19(3):40–47, 1996.
12. K.-L. Wu, P. S. Yu, and M.-S. Chen. Energy-efficient caching for wireless mobile computing. In *Proceedings of 12th ICDE*, pages 336–345, Feb. 26-March 1 1996.