

# Mining User Moving Patterns for Personal Data Allocation in a Mobile Computing System

Wen-Chih Peng and Ming-Syan Chen  
Department of Electrical Engineering  
National Taiwan University  
Taipei, Taiwan, ROC

E-mail: { mschen@cc.ee.ntu.edu.tw, wcpeng@arbor.ee.ntu.edu.tw }

## Abstract

*In this paper, we devise a new data mining algorithm which involves mining for user moving patterns in a mobile computing environment, and utilize the mining results to develop data allocation schemes so as to improve the overall performance of a mobile system. First, we devise an algorithm to capture the frequent user moving patterns from a set of log data in a mobile environment. Then, in light of mining results of user moving patterns and the properties of data objects, we develop data allocation schemes for proper allocation of personal data. Two personal data allocation schemes, which explore different levels of mining results, are devised: one utilizes the set level of moving patterns and the other utilizes the path level of moving patterns. Performance of these data allocation schemes is comparatively analyzed. It is shown by our simulation results that the user moving patterns is very important in devising effective data allocation schemes which can lead to significant performance improvement in a mobile computing system.*

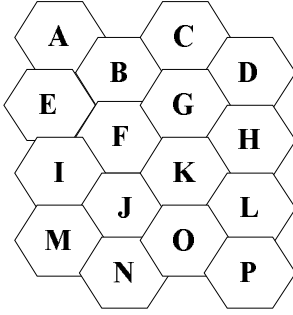
## 1 Introduction

Due to recent technology advances, an increasing number of users are accessing various information systems via wireless communication. Such information systems as stock trading, banking, wireless conferencing, are being provided by information services and application providers [8][12][18][13], and mobile users are able to access such information via wireless communication from anywhere at any time [3][7][17].

For cost-performance reasons, a mobile computing system is usually of a distributed server architecture [12], in which a service area, referring to the converge area where the server can provide services to mobile users, contains one or many cells where a cell refers to a communication area

covered by a base station. In general, mobile users tend to submit transactions to servers nearby for execution so as to minimize the communication overhead incurred [12] [16]. The properties of data objects accessed by mobile users can usually be divided into two types: the read-intensive type (or read type) and the update-intensive type (or update type). Data objects are assumed to be stored at servers to facilitate coherency control and also for memory saving at mobile units [18][19]. Since the architecture of a mobile computing system is distributed in nature, data replication is helpful because it is able to improve the execution performance of servers and facilitate the location lookup of mobile users [11][16][18][19]. The replication scheme of a data object involves how many replicas of that object to be created, and to which servers those replicas are allocated. Clearly, though avoiding many costly remote accesses, the approach of data replication increases the cost of data storage and update. Thus, it has been recognized as an important issue to strike a compromise between access efficiency and storage cost when a data allocation scheme is devised.

It is noted that various data allocation schemes have been extensively studied in the literature [18][19]. However, the data allocation schemes for traditional distributed databases are mostly designed in static manners, and the user moving patterns, which are particularly relevant to a mobile computing system where users travel between service areas frequently, were not fully explored. As mentioned above, the server is expected to take over the transactions submitted by mobile users and static data allocation schemes may suffer severe performance problems in a mobile computing system. An example scenario is given in Figure 1 where we assume that the data are replicated statically at sites A, F, K, and P under the data allocation schemes for traditional distributed databases, and the mobile user  $U_1$  is found to frequently travel in service areas of A, B and C (i.e.,  $\{A, B, C\}$  is called the moving pattern of mobile user  $U_1$ ). It can be seen that the advantage of having replicas on F, K and P



**Figure 1. The network architecture of a mobile computing system**

cannot be fully taken by mobile user  $U_1$ , and the extra cost of maintaining those replicas is not justified by the moving pattern of user  $U_1$ . In order to improve the system performance, efficient data allocation schemes based on moving patterns of mobile users are very important in a mobile computing environment. This is the very problem we shall address in this paper. Consequently, we shall explore in this paper the approach of mining user moving patterns in a mobile computing environment and utilize the mining results to develop data allocation schemes. Note that the data allocation schemes devised not only utilize the mining results but also consider the properties of data objects for improving the overall performance of a mobile system.

Specifically, for the development of data allocation schemes, we shall first devise an algorithm to capture the frequent user moving patterns from a set of log data (referred to as *moving log*) in a mobile environment. It is worth mentioning that to fully explore the temporal locality, which refers to the feature that consecutive movements of a mobile user are likely to fall into similar sites, the mining algorithm devised is enhanced in such a way that it is able to focus on the recent moving patterns within an adjustable window size (referred to as *retrospective factor*) when deriving user moving patterns. As such, by avoiding generating user moving patterns for every single movement of individual mobile users, the frequent user moving patterns can be obtained much more efficiently without compromising the quality of results obtained.

Then, in light of mining results of user moving patterns, we devise data allocation schemes that can utilize the knowledge of user moving patterns for proper allocation of personal data (referring to those data only accessible by each individual data owner). By employing the data allocation schemes devised, the occurrences of costly remote accesses can be minimized and the performance of a mobile computing system is thus improved. Two personal data allocation schemes, which explore different levels of min-

ing results and properties of data objects, are devised: one utilizes the *set level* of moving patterns and the other utilizes the *path level* of moving patterns. As can be seen later, the former is useful for the allocation of read data objects, whereas the latter is good for the allocation of update data objects. Performance of these data allocation schemes is comparatively analyzed and sensitivity analysis on several design parameters, including the retrospective factor, is conducted. It is shown by our simulation results that the knowledge obtained from the user moving patterns is very important in devising effective data allocation schemes which can lead to significant performance improvement in a mobile computing system.

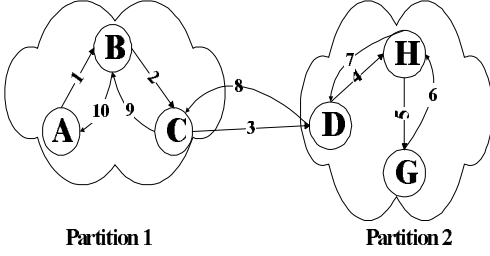
Various data mining capabilities have been explored in the literature [4]. One of the most important data mining problems is mining association rules in transaction databases [1][6]. Also, mining classification rules is an approach to develop rules that can efficiently classify data items based on certain features [15]. Mining sequential patterns is the study to explore the knowledge of ordered data [2][10]. Though dealing with various mining capabilities, these prior results were not applicable to mining user moving patterns in a mobile environment.

In addition, a significant amount of research efforts has been elaborated upon issues of data allocation in distributed systems [18][19]. We mention in passing that the authors of [19] proposed a data distribution scheme that is based on the read/write patterns of the data objects. Given some user calling patterns, the authors of [18] proposed an algorithm that employed the concept of minimum-cost maximum-flow to compute the set of sites where user profiles should be replicated. The work in [18][19] neither fully explored the data mining capability for user moving patterns in a mobile computing system nor utilized such mining results for personal data allocation. These features distinguish this paper from others.

This paper is organized as follows. Preliminaries are given in Section 2. Algorithms for mining user moving patterns in a mobile system are devised in Section 3.1 and personal data allocation schemes based on user moving patterns are developed in Section 3.2. Simulation results are presented and analyzed in Section 4. This paper concludes with Section 5.

## 2 Preliminary

The location management procedure for a mobile computing system considered in this paper is similar to the one in IS-41/GSM [9], which is a two level standard and uses a two-tier system of home location register (to be referred to as HLR) and visitor location register (to be referred to as VLR) databases. Each mobile user is associated with an HLR, whose database maintains recent mobile users'



**Figure 2. An example of moving patterns in a mobile computing system.**

records and their current locations. When a mobile user moves out the area maintained by its HLR, a copy of the mobile user's record is created in its local VLR. In addition, the record in the HLR is updated to reflect the movement of that user. This procedure is referred to as registration.

In order to capture user moving patterns, a moving log is needed. A moving log contains a pair of (old VLR, new VLR) in the database when registration occurs. In the beginning of a new path, the old VLR is null. For each mobile user, we can obtain a moving sequence  $\{(O_1, N_1), (O_2, N_2), \dots, (O_n, N_n)\}$  from the moving log.

Consider an illustrative example in Figure 2, where the number next to each link represents the sequence of movements of a mobile user. Thus, the moving log contains the moving path for that mobile user. With the assumption that backward travels are made for ease of travelling and hence need not be considered, algorithm MF in [5] will terminate the discovery of maximal moving sequences when any backward reference occurs. The set of maximal moving sequences for this moving path output by algorithm MF is  $\{ABCDHG\}$ . However, such a fragmented moving sequence is of little interest in a mobile environment where one would naturally like to know the complete travelling information, i.e.,  $\{ABCDHGHDCBA\}$  in this case, showing the very difference between these two environments. Note that by taking into consideration the backward travels, nodes may appear more than once in the same maximal moving sequence, and it is necessary to extract these sequences and take their occurrences into account when the corresponding user moving patterns are evaluated. Clearly, to deal with these problems, as well as to solve such an important issue as data allocation which is particularly relevant to a mobile environment, it is essential to develop a new algorithm for mining user moving patterns in a mobile computing system.

### 3 Data Allocation Schemes Based on Moving Patterns

In Section 3.1, we develop a solution procedure, which is composed of a sequence of algorithms in the corresponding steps, to mine user moving patterns. In light of the user moving patterns determined, we develop personal data allocation schemes in Section 3.2.

#### 3.1 Mining for Moving Patterns in a Mobile Environment

The overall procedure for mining moving patterns is outlined as follows.

##### **Procedure for mining moving patterns**

**Step 1. (Data collection phase)** Employing algorithm MM (to be described in Section 3.1.1) to determine maximal moving sequences from a set of log data and also the occurrence count of moving pairs.

**Step 2. (Mining phase)** Employing algorithm LM (to be described in Section 3.1.2) to determine large moving sequences for every  $w$  maximal moving sequences obtained in Step 1, where  $w$  is the retrospective factor which is an adjustable window size for the recent moving patterns to be considered.

**Step 3. (Pattern generation phase)** Determine user moving patterns from large moving sequences obtained in Step 2.

Note that in the data collection phase, the occurrence counts of moving pairs are updated on-line during registration procedure. For purposes of efficiency, algorithm LM is executed to obtain new moving patterns every  $w$  maximal moving sequences generated. The selection of  $w$  will be determined empirically in Section 4 later. Also, note that after large moving sequences are determined (by Step 2), moving patterns can then be obtained in a straightforward manner. A moving pattern is a large moving sequence that is not contained in any other moving patterns. For example, suppose that  $\{AB, BC, AE, CG, GH\}$  is the set of large 2-moving sequences and  $\{ABC, CGH\}$  is the set of large 3-moving sequences. Then, the resulting user moving patterns are  $\{AE, ABC, CGH\}$ . As users travel, their moving patterns can be discovered to reflect the user moving behaviors.

##### 3.1.1 Finding Maximal Moving Sequences

Algorithm MM (standing for maximal moving sequence), whose algorithmic form is given below, is devised and applied to moving sequences of each mobile user to determine the maximal moving sequences of that user and update the occurrence count of moving pairs during registration procedure.

In algorithm MM, Y is used to keep the current maximal moving sequence and F is a flag to indicate if a node is revisited. Let  $D_F$  denote the database to store all the resulting maximal moving sequences. Also, S is the home location site of a mobile user. According to the roundtrip model considered [18], the selection of S is either VLR or HLR whose geography area contains the homes of mobile users. In Step 1, moving sequences are obtained from each movement and some parameters are initialized. Then, moving sequences are scanned in Step 2. A maximal moving sequence is output and a new maximal moving sequence will be explored (by Step 5) if MM finds that  $N_i$  in the moving pair  $(O_i, N_i)$  is the same as the starting site S. Otherwise,  $N_i$  is appended into Y (by Step 3) and the occurrence count of  $(O_i, N_i)$  is updated on-line in the database (by Step 4).

*Algorithm MM* /\* Algorithm MM for finding maximal moving sequences \*/

**Step 1.** Set  $i$  to 1 and string  $Y$  to null, where  $Y$  is used to keep the current maximal moving sequence;

S is the starting point and set  $F = 0$ , where  $F$  is a flag to indicate if a node is revisited;

**Step 2. while** (not end of movings)

{Set  $A = O_i$  and  $B = N_i$ ;

**if** ( $A = S$  and  $F = 0$ )

Set  $Y = B$  and  $F = 1$ ;

**else**

**Step 3.** {Append  $B$  to string  $Y$ ;

**Step 4.** Update the occurrence count of  $(A, B)$  in database  $D_F$ ;

**Step 5. if** ( $B = S$  and  $F = 1$ )

{Output string  $Y$  to database  $D_F$ ;

Set  $Y$  to null and  $F = 0$ ;} }

### 3.1.2 Finding Large Moving Sequences

With the maximal moving sequences obtained, we next determine the large moving sequences. A sequence of  $k$  movements is called a large  $k$ -moving sequence if there are a sufficient number of maximal moving sequences containing this  $k$ -moving sequence. Such a threshold number is called support in this paper. A large moving sequence can be determined from all maximal moving sequences of each individual user based on its occurrences in those maximal moving sequences. Use *intra-sequence count* to mean the number of occurrences of a moving sequence within a maximal moving sequence, and *inter-sequence set* of a moving sequence to mean the set of maximal moving sequences which contain that moving sequence. The count of a large moving sequence is the sum of intra-sequence counts from its inter-sequence set. We develop algorithm LM (standing for large moving sequence) for the determination of large moving sequences below.

*Algorithm LM* /\* Algorithm for finding large moving sequences \*/

**Step 1.** Determining  $L_2 = \{\text{large 2-moving sequence}\}$  from moving pairs in  $C_2$ ;

**Step 2. for** ( $k = 3; L_{k-1} \neq 0, k++$ )

{  $C_k = L_{k-1} * L_{k-1}$ ;

**Step 3. for** all maximal moving sequence  $S$

{ intra-sequence = sub-sequence( $C_k, S$ );

**if** (intra-sequence > 0)

$S \in$  inter-sequence;

**Step 4. for** all candidate  $c \in$  inter-sequence

$c.\text{count} = c.\text{count} + c.\text{intra-sequence}$ ; }

**Step 5.**  $L_k = \{c \in C_k \mid c.\text{count} \geq \text{support}\}$ ;

Let  $L_k$  represent the set of all large  $k$ -moving sequences and  $C_k$  be a set of candidate  $k$ -moving sequences. As pointed out in [4], the initial candidate set generation, especially for  $L_2$ , is the key issue to improve the performance of data mining. Since occurrence counts of moving pairs, i.e.,  $C_2$ , were update on-line in the data collection phase,  $L_2$  can be determined by proper trimming on  $C_2$  efficiently (by Step 1 in algorithm LM), showing the advantage of having on-line update in algorithm MM. Also, note that  $C_k$  can be simply generated from  $L_{k-1} * L_{k-1}$  (by Step 2 in algorithm LM). For example, with the set of  $L_2$  being  $\{AB, BK\}$ , we have a  $C_3$  as  $\{ABK\}$ . As explained above, the occurrence count of each  $k$ -moving sequence is the sum of intra-sequence counts (by Step 3 in algorithm LM) in its inter-sequence set (i.e., Step 4 in algorithm LM). Note that this step is very different from that in mining the path traversal patterns [5] where there are no loops in a moving sequence (i.e., the corresponding intra-sequence count is always zero). The occurrences of each  $k$ -moving sequence in  $C_k$  are determined for the identification of  $L_k$ . After the summation of the occurrence counts in the inter-sequence set in Step 4, those  $k$ -moving sequences with counts exceeding the support are qualified as  $L_k$  (by Step 5 in LM).

## 3.2 Schemes for Personal Data Allocation

In this section, we propose efficient data allocation schemes based on user moving patterns for personal data allocation. We first describe in Section 3.2.1 the data allocation scheme in a fix pattern. Then, we devise two data allocation schemes based on two levels of user moving patterns, explicitly the set level and the path level, in Section 3.2.2 and in Section 3.2.3, respectively.

### 3.2.1 DF (Data Allocation Scheme in a Fixed Pattern)

In the scheme which allocates data in a fix pattern (referred to as scheme DF), the replication sites are determined when the database is created. Explicitly, the number of replicated sites and the sites at which the personal data can be replicated are predetermined. Though being adopted in some

User ID	Moving patterns
U <sub>1</sub>	AE, ABC
U <sub>2</sub>	BCGF
U <sub>3</sub>	BCD
U <sub>4</sub>	CGK

**Table 1. An example profile for illustrating data allocation schemes.**

traditional distributed database systems due to its ease of implementation [19], scheme DF is not suitable for mobile computing environments where mobile users move frequently. In our experimental studies in Section 4, scheme DF will be implemented and evaluated for comparison purposes.

### 3.2.2 DS (Data Allocation Scheme based on the Set of Moving Patterns)

In this subsection, the data allocation scheme based on the set of moving patterns (referred to as scheme DS) is described. Scheme DS takes advantage of moving patterns so as to reduce the number of remote accesses. Consider the example profile in Table 1 where the network architecture is the one in Figure 1. The set of replicated servers under scheme DS is the set of servers determined from moving patterns of mobile users. For example, the set of the replicated server for U<sub>1</sub> is the union of moving patterns of U<sub>1</sub> (i.e., {AE} ∪ {ABC} = {ABCE}), and that of U<sub>2</sub> is {BCGF}. In addition, as will be shown in Section 4, scheme DS is able to increase the hit ratio of local data access.

However, according to scheme DS, more replicated sites are employed as the number of sites appearing in user moving patterns increases, which, as will be seen in Section 4, though reducing the occurrences of remote access significantly, may result in too many sites replicated, thus compromising the overall performance improvement achieved. In order to eliminate the cost of maintaining replicated sites, we shall take the properties of data objects into consideration. Clearly, for a read data object, data replication is preferable in order to increase the likelihood of local reads. As a result, scheme DS favors those read-intensive data objects. On the other hand, for an update data object, scheme DS is not appropriate since scheme DS results in too many sites replicated, increasing the costs of update and communication. In view of this, we shall investigate the approach of utilizing the mining results in the path level, instead of the set level as in scheme DS. This will allow us of employing the technique of prefetches properly and ensuring the number of replicated sites not to exceed a predetermined limit (Such a limit is called prefetch size). Using the path level

knowledge, as a mobile user moves, the personal data can be prefetched to the subsequent sites predicated from user moving patterns and the advantage of mining user moving patterns can be fully exploited. This is the very motivation that the design of scheme DP is based on.

### 3.2.3 DP (Data Allocation Scheme based on the Path of Moving Patterns)

In this subsection, a data allocation scheme based on the paths of moving patterns (referred to as scheme DP) is employed to determine the candidate sites for proper prefetches. An algorithmic form of scheme DP is given below. The number of  $P_{fetch}$ , which is the predetermined prefetch size, is the number of replicated sites to obtain. If the number of path moving patterns included then, i.e.,  $P_f$ , is less than  $P_{fetch}$ , scheme DP will use the current site as the key to search moving patterns of the mobile user for finding the candidate moving patterns to use. After obtaining the candidate path moving patterns, scheme DP will determine the candidate prefetch sites (by Step 3). Following this procedure, the candidate prefetch sites are determined by Step 2 and Step 3 in scheme DP. As can be seen, the replicated sites of DP vary when the corresponding user moves.

*Scheme DP:*

**Step 1:** Search the current site in path moving patterns;

Update path moving patterns and  $P_f$ , where  $P_f$  is the number of path moving patterns included thus far;

/\* This procedure will eliminate path moving patterns which do not contain the current site \*/

**Step 2:** while ( $P_{fetch} > P_f$ ) /\*  $P_{fetch}$  is the predetermined prefetch size \*/

{ Search user moving patterns of the corresponding user by using current site as the key;

if (found)

$|P_f| = |P_f| + 1;$

Select this moving pattern as a path moving pattern; }

**Step 3:** Prefetch data on sites predicated from path moving patterns;

Table 2 shows the scenarios under different data allocation schemes with the example profile in Table 1. As can be seen from the experimental results in Section 4, scheme DP performs better than scheme DS in that it can achieve the same high local data access hit ratios as DS while incurring a much smaller cost for maintaining replicated sites than DS, showing the very advantage of employing the path level knowledge to do proper prefetches and thus suitable for update data objects.

User with an example moving path	Replicated server under DF	Local data hit of DF
U <sub>1</sub> with ABCG	AFKP	N <sub>A</sub>
U <sub>2</sub> with BCGF	AFKP	N <sub>F</sub>
U <sub>3</sub> with BCDH	AFKP	0
U <sub>4</sub> with CGH	AFKP	0

Table 2a. The scenario under DF

User with an example moving path	Replicated server under DS	Local data hit of DS
U <sub>1</sub> with ABCG	ABCE	N <sub>A</sub> +N <sub>B</sub> +N <sub>C</sub>
U <sub>2</sub> with BCGF	BCGF	N <sub>B</sub> +N <sub>C</sub> +N <sub>G</sub> +N <sub>F</sub>
U <sub>3</sub> with BCDH	BCD	N <sub>B</sub> +N <sub>C</sub> +N <sub>D</sub>
U <sub>4</sub> with CGH	CGK	N <sub>C</sub> +N <sub>G</sub>

Table 2b. The scenario under DS

User with an example moving path	Replicated servers under DP	Local data hit of DP
U <sub>1</sub> with ABCG	AE, ABC	N <sub>B</sub> +N <sub>C</sub>
U <sub>2</sub> with BCGF	BCGF	N <sub>C</sub> +N <sub>G</sub> +N <sub>F</sub>
U <sub>3</sub> with BCDH	BCD	N <sub>C</sub> +N <sub>D</sub>
U <sub>4</sub> with CGH	CGK	N <sub>G</sub>

Table 2c. The scenario under DP

**Table 2. The scenarios under different data allocation schemes for personal data allocation.**

## 4 Performance Study

The effectiveness of using the knowledge on user moving patterns for data allocation is evaluated empirically in this section. The simulation model for the mobile system considered is described Section 4.1. Experiments results of personal data allocation, including those of DF, DS and DP, are shown in Section 4.2.

### 4.1 Simulation Model for a Mobile System

To simulate the servers in a mobile computing system, we use a four by four mesh network [14], where each node represents one server and there are hence 16 servers in this model. It is assumed that there are 100000 mobile users in our simulations. A moving path is a sequence of servers accessed by a mobile user. The moving behaviors of mobile users depend on a probabilistic model. Explicitly, the probability that a mobile user moves to the server where this user came from is modeled by  $P_{back}$  and the probability that the mobile user routes to the other servers is determined by  $(1-P_{back})/(n-1)$  where  $n$  is the number of possible servers this mobile user can move to. As mentioned before, the number of consecutive moving paths employed for mining of user

Notation	Definition
retrospective factor	number of referenced moving paths for incremental mining
exploring factor	number of moving paths for a mobile user
$P_{back}$	backward probability for user movement
$P_{fetch}$	pre-fetch size for DP
$P_{fm}$	percentage of frequent moving mobile users

**Table 3. The parameters used in the simulation.**

moving patterns in LM is called the *retrospective factor* (denoted by  $w$ ). The number of moving paths used to analyze the performance of data allocation schemes is called the *exploring factor*. The prefetch size used by scheme DP is denoted by  $P_{fetch}$ . Table 3 summaries the definitions of some primary simulation parameters. The local hit ratio means the percentage that among all data accesses, data can be obtained from local servers of mobile users, and the support is the threshold used to qualify large moving sequences.

### 4.2 Experiments for Utilizing Personal Data Allocation

The effectiveness of personal data allocation schemes based on moving patterns will be evaluated in this subsection. We first examine the performance of schemes DF and DS in Section 4.2.1, and then evaluate the performance of schemes DP in Section 4.2.2.

#### 4.2.1 Experiments of DF and DS for Personal Data Allocation

To conduct the experiments to evaluate DF and DS, we set the value of the retrospective factor to be 10 and the value of the support to be 0.4. Both the local hit ratios and the corresponding server loads of DF and DS are examined with the exploring factor varied. Figure 3 shows the local hit ratios of DF and DS. It can be seen from Figure 3 that the local hit ratio of DS is significantly larger than that of DF, showing the very advantage of using the knowledge of user moving patterns in scheme DS. Note that the local hit ratio of DS tends to increase when exploring factor increases. This is due to the fact that with a fix support, more moving patterns are discovered by DS as the value of exploring factor increases.

The sensitivity of varying the values of the retrospective factor is next investigated. Specifically, we examine the impact of varying the retrospective factor to the performance of DS. Without loss of generality, we set the values of the retrospective factor  $w$  to be 10, 20 and 50, and that of the

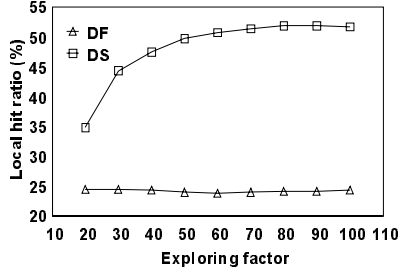


Figure 3. The local hit ratios of DF and DS with the exploring factor varied.

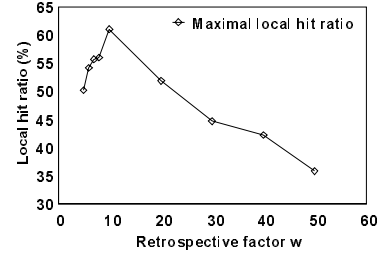


Figure 5. The maximal local hit ratio with the retrospective factor varied.

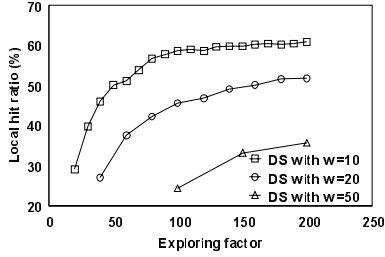


Figure 4. The performance of DS with the retrospective factor  $w$  and the exploring factor varied.

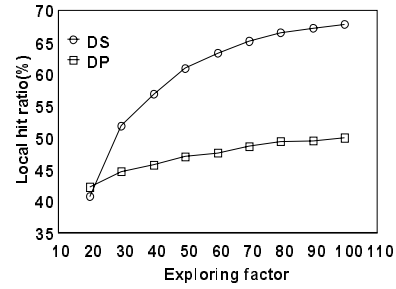


Figure 6. The local hit ratios of DS and DP with various exploring factor

support to be 0.4, and examine the performance of DS as  $w$  varies. Figure 4 shows the cases of using smaller retrospective factors will outperform those using larger retrospective factors. This can be explained by the feature of temporal locality since a larger value of  $w$  may involve some obsolete moving patterns and thus cannot well predict future moving patterns of mobile users. However, as can be seen from Figure 5, performance might degrade when the value of  $w$  is too small to be used as a window size for mining (e.g., when  $w < 10$ ), in which cases, there are insufficient moving paths available for determining moving patterns. Clearly, the selection of the value of  $w$  will be dependent upon the temporal locality in the workload and can be determined empirically.

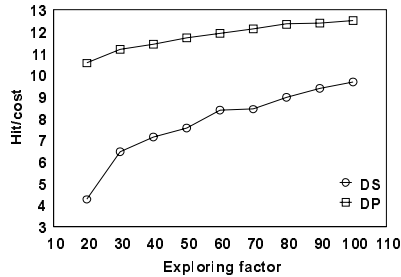
#### 4.2.2 Experiments of DP for Personal Data Allocation

As described above, using moving patterns in a set level will unavoidably cause scheme DS to incur more cost to maintain replicated servers, which is not suitable for update data objects. In view of this, scheme DP is designed to exploit the user moving patterns discovered in the path level and the data is prefetched in accordance with the knowledge discovered. As shown in Section 4.2.1, scheme DS in general

outperforms scheme DF, we shall thus focus on comparing the performance of scheme DS and that of scheme DP in this subsection.

Without loss of generality, we set the value of  $P_{fetch}$  to be 2 in scheme DP, and the value of the support to be 0.3. The local hit ratios of DS and DP with various exploring factor are shown in Figure 6. From Figure 6, it can be seen that by having the most replicated servers, the local hit ratio of DS is larger than that of DP.

Note, however, that though performing better than DP in local hit ratios, DS incurs more cost to maintain data consistency of replicated servers. A measurement, hit/cost ratio, is used to represent the local hit ratio gained by having an additional replicated server. A larger value of this measurement means that each replicated server is more efficiently used to contribute to the occurrences of local data hits. Figure 7 shows the hit/cost ratios of DS and DP. Note that DP has a larger hit/cost ratio than DS, meaning that DP employs the approach of data allocation more cost-effectively to increase the local data hit ratio, showing the very advantage of using the path level knowledge to do prefetches.



**Figure 7. The hit/cost ratios of DS and DP with various exploring factor**

## 5 Conclusions

In this paper, we devised a new data mining algorithm which involves mining for user moving patterns in a mobile computing environment and utilized the mining results to devise data allocation schemes so as to improve the overall performance of a mobile system. First, we devised algorithms (i.e., algorithm MM and LM) to capture the frequent user moving patterns from a set of log data in a mobile environment. Then, in light of mining results of user moving patterns, we devised data allocation schemes that can utilize the knowledge of user moving patterns for proper allocation of personal data. For personal data allocation, two data allocation schemes, which involve different levels of mining results, have been devised: one utilizes the set level of moving patterns (i.e., scheme DS) and the other utilizes the path level of moving patterns (i.e., scheme DP). Sensitivity analysis on various parameters, including the retrospective factor, was conducted and performance of those data allocation schemes was comparatively analyzed. It was shown by our simulation results that the knowledge obtained from the user moving patterns is very important in devising effective data allocation schemes which can lead to significant performance improvement in a mobile computing system.

### Acknowledgment

The authors are supported in part by the Ministry of Education Project No. 89-E-FA06-2-4-7 and the National Science Council, Project No. NSC 89-2219-E-002-007 and NSC 89-2213-E-002-032, Taiwan, Republic of China.

## References

- [1] R. Agrawal, T. Imielinski, and A. Swami. Mining Associations between Sets of Items in Massive Databases. In *Proceeding of ACM SIGMOD*, pages 207–216, May 1993.
- [2] R. Agrawal and R. Srikant. Mining Sequential Patterns. In *Proceeding of the 11th International Conference on Data Engineering*, pages 3–14, March 1995.
- [3] B. Bruegge and B. Bennington. Applications of Mobile Computing and Communication. *IEEE Personal Communication*, pages 64–71, February 1996.
- [4] M.-S. Chen, J. Han, and P. S. Yu. Data Mining: An Overview from Database Perspective. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):866–883, December 1996.
- [5] M.-S. Chen, J.-S. Park, and P. S. Yu. Efficient Data Mining for Path Traversal Patterns. *IEEE Transactions on Knowledge and Data Engineering*, 10(2):209–221, April 1998.
- [6] D. W. Cheung, V. T. Ng, W. Fu, and Y. Fu. Efficient Mining Association Rules in Distributed Databases. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):911–922, December 1996.
- [7] N. Davies, G. S. Blair, K. Cheverst, and A. Friday. Supporting Collaborative Application in a Heterogeneous Mobile Environment. *Computer Communication Special Issues on Mobile Computing*, 1996.
- [8] M. H. Dunham. Mobile Computing and Databases. *Tutorial of International Conference on Data Engineering*, February 1998.
- [9] EIA/TIA. Cellular Radio Telecommunication Intersystem Operations. 1991.
- [10] J. Han, G. Dong, and Y. Yin. Efficient Mining of Partial Periodic Patterns in Time Series Database. In *Proceeding of the 15th International Conference on Data Engineering*, March 1999.
- [11] J. Jannink, D. Lam, N. Shivakumar, J. Widom, and D. Cox. Efficient and Flexible Location Management Techniques for Wireless Communication Systems. *ACM Journal of Wireless Networks*, 3(5):361–374, 1997.
- [12] N. Krishnakumar and R. Jain. Escrow Techniques for Mobile Sales and Inventory Applications. *ACM Journal of Wireless Network*, 3(3):235–246, July 1997.
- [13] D. L. Lee. Data Management in a Wireless Environment. *Tutorial of International Conference on Database System for Advance Applications*, April 1999.
- [14] Y.-B. Lin. Modeling Techniques for Large-Scale PCS Networks. *IEEE Communications Magazine*, 35(2):102–107, February 1997.
- [15] R. Ng and J. Han. Efficient and Effective Clustering Methods for Spatial Data Mining. *Proceedings of the 18th International Conference on Very Large Data Bases*, pages 144–155, September 1994.
- [16] W.-C. Peng and M.-S. Chen. A Dynamic and Adaptive Cache Retrieval Scheme for Mobile Computing Systems. In *Proceeding of Third IFICIS Conference on Cooperative Information Systems (CoopIS '98)*, pages 251–258, August 1998.
- [17] M. Satyanarayanan. Mobile Information Access. *IEEE Personal Communication*, pages 26–33, February 1996.
- [18] N. Shivakumar, J. Jannink, and J. Widom. Per-User Profile Replication in Mobile Environments: Algorithms, Analysis and Simulation Results. *ACM Journal of Mobile Networks and Applications*, (2):129–140, 1997.
- [19] O. Wolfson, S. Jajodia, and Y. Huang. An Adaptive Data Replication Algorithm. *ACM Transactions on Database Systems*, 22(4):255–314, June 1997.