北京大学学报(自然科学版),第 40 卷,第 3 期,2004 年 5 月
Acta Scientiarum Naturalium
Universitatis Pekinensis, Vol. 40, No. 3 (May, 2004)

Mining Maximal Moving Sequential Patterns in Mobile Environment[1)]

MA Shuai TANG Shiwei YANG Dongqing WANG Tengjiao GAO Jun
(Department of Computer Science, Peking University, Beijing, 100871)

Abstract Mining moving sequential patterns has great significance for effective and efficient location management in wireless communication systems. Mining moving sequential patterns is different from mining conventional sequential patterns, firstly it needs to consider much about the time factor in moving sequences; secondly it cares about what the next moving is for mobile user, so items must be successive in mining moving sequential patterns. A novel technique to mine moving sequential patterns is proposed. A clustering method is introduced to preprocess the original moving histories into moving sequences, whose main role is to discretize the time attribute. And then an efficient method, called PrefixTree, is presented to mine the moving sequences. Performance study shows that PrefixTree outperforms Revised PrefixSpan-2, which is revised to mine moving sequences, in mining large moving sequence databases.

Key words clustering; sequential pattern mining; moving sequential pattern mining

0 Introduction

In order to provide effective and efficient location management technologies in wireless communication systems, researchers recently tend to focus on characterizing the mobile users moving and/or calling characteristics, such as CMR (Call to Mobility Ratio), Markov model, probability distribution based on profile, mobility graph, and moving behaviors. User's moving behaviors can help to allocate personal data, pre-fetch useful information, pre-allocate wireless resources, and design personal paging areas, etc. In this paper authors focus on mining mobile user maximal moving sequential patterns.

Wen-Chih Peng et al. present a data-mining algorithm, which involves mining for user moving patterns in a mobile computing environment in[1]. Moving pattern mining is based on a roundtrip model, and their LM algorithm selects an initial location S, which is either VLR(Visitor Location Register) or HLR (Home Location Register) whose geography area contains the homes of the mobiles users. Suppose that a mobile user goes to a strange place for one month or longer, LM cannot find the proper moving pattern to characterize the mobile user. A good strategy should not give any assumption of the start point of a moving

1) 国家 863 高科技发展计划(2001AA114040),国家重点基础研究发展规划 973(G1999032705)北京大学—IBM 创新研究院基金资助项目
收稿日期: 2003-11-17; 修回日期: 2003-12-10

pattern. Still the moving patterns in [1] only consider the order of the location, but not consider enough the time factor. The mobility graph in[2] is based on the assumption that user can make at most a move in a slot τ, but it is hard to determine slot τ, which guarantees that the sojourn time (The time that a mobile user stays in a cell) can be divided exactly by τ, even τ may not exist at all.

The problem of mining sequential patterns was first introduced by R. Agrawal et al. in[3], and there have been many studies on efficient sequential pattern mining and its applications[3~8], which can be categorized into two classes: (1) Apriori-based[3~6]; (2) Projection-based[7,8]. Sequential pattern mining has broad applications, including the analysis of customer purchase behavior, web access patterns, scientific experiments, disease treatments, natural disasters, DNA sequences, and so on. In this paper, a novel and efficient technique to mine mobile user maximal moving sequential patterns is proposed.

Contributions. Firstly, a clustering method is introduced to preprocess the original moving histories into moving sequences, whose main role is to discretize the time attribute. After the data preprocessing, the original moving histories are transformed into moving sequences. Secondly, an efficient algorithm is proposed to discover the mobile users' moving sequential patterns based on prefix trees, which can effectively represent candidate frequent moving sequences. Thirdly, performance study shows that PrefixTree outperforms Revised PrefixSpan-2, which is revised to mine moving sequences, in mining large moving sequence databases.

Organization. The rest of the paper is organized as follows. Data preprocessing is given in section 1 In section 2, the algorithm of mining maximal moving sequential patterns is described. The experimental results from different points of view are shown in section 3. Discussion and future work are made in section 4.

1 Data Preprocessing

User moving history is the moving logs of a mobile user, which is an ordered pair of (c, t) where c is the cell *ID* and t is the time when the mobile user reaches cell c. Let $MH = \langle (c_1, t_1), (c_2, t_2), (c_3, t_3), \cdots (c_n, t_n) \rangle$ be a piece of moving history, and MH means the mobile user enters c_1 at t_1, leaves c_1 and enters c_2 at t_2, leaves c_2 and enters c_3 at t_3 ··and finally enters c_n at t_n. Each $(c_i, t_i) \in MH$ $(1 \leq i \leq n)$ is called an element of MH. The time difference between two successive elements in a moving history reflects a mobile users' moving speed (or sojourn time in a cell). If the difference is high, it shows the mobile user moves at a relative low speed; and if the difference is low, it shows the mobile user moves at a relative high speed.

Time is a very important factor in mining moving sequential patterns. G. Das et al. present a clustering based method to discretize a time series in[9], and Hsiao-Kuang Wu et al. also use clustering methods to preprocess the moving logs in[10]. The idea using clustering to discretize time is that if a mobile user possesses regular moving behavior, then it often moves on the same set of paths, and the arrival time to each point of the paths is similar. In the meanwhile, day is a natural periodic cycle that a mobile user usually behaves, so we consider the moving histories in a day. Thus the moving sequential patterns reflect a mobile user's moving behaviors in a day.

There are many clustering algorithms, and a clustering method called CURD was presented in[11], which preserves the ability of density based clustering method's good advantages, and is very efficient because of its nearly linear time complexity (see[11] for details). Here CURD algorithm is used to discretize time. The clustering processing is explained as below.

For each cell c in the set $CSet$, all the elements in the moving history database D are collected. The element set of c is denoted by $ES(c) = \{(c, t) \mid \exists (c, t) \in MH_i \wedge MH_i \in D\}$. Then CURD algorithm is used to cluster the element set $ES(c)$, and use Euclidean distance on time t as the similarity function between two elements in $ES(c)$. This is a clustering problem in one-dimension space. After clustering a clustering result as $\{(c, T_s, T_e) \mid T_s, T_e \in T \wedge T_s \leq T_e\}$ is obtained, which means the mobile user often enters cell c at period $[T_s, T_e]$ (T is the time domain).

The main idea of data transformation is to replace the MH elements with the corresponding clusters that they belong to. The transformed moving history is called a moving sequence, and the transformed moving history database is called the moving sequence database.

Lemma 1 *Each moving history can be transformed into one and only one moving sequence.*

The clustering method guarantees that any MH element belongs to one and only one cluster, so the above lemma is true.

2 Mining Moving Sequential Patterns

2.1 Related Definitions

Let $C = \{CT_1, CT_2, \cdots CT_n\}$ be a set of all items, each CT_i is a tuple, denoted by (c, T_s, T_e), where c is a cell id, T_s, T_e are time points, and $T_s \leq T_e$. A Moving sequence is an order list of items, and is denoted by $\langle s_1 s_2 \cdots s_m \rangle$ ($s_i \in C$). The number of items in a sequence is called the length of the sequence. A sequence with length k is called a k-sequence. A sequence $\alpha = \langle a_1 a_2 \cdots a_n \rangle$ is called a subsequence of another sequence $\beta = \langle b_1 b_2 \cdots b_m \rangle$ and β is a super sequence of α, denoted by $\alpha \subseteq \beta$, if there exists an integer $1 \leq i \leq m$ such that $a_1 = b_i$, $a_2 = b_{i+1} \cdots a_n = b_{i+n-1}$.

A moving sequence database D is a set of tuples $\langle sid, s \rangle$, where sid is the sequence id and s is a moving sequence. Every one of the sequences records the moving behavior of a mobile user with the same period, such as a day, a month, etc. A tuple $\langle sid, s \rangle$ is said to contain a sequence α if $\alpha \subseteq s$. The support of a sequence α in a moving sequence database D is the number of moving sequences in the database that contain α, i.e. $support(\alpha) = |\{\langle sid, s \rangle \mid \langle sid, s \rangle \in D \wedge \alpha \subseteq s\}|$.

Give a positive integer ζ as the support threshold, a moving sequence α is called a moving sequential pattern, if α is contained by at least ζ tuples in the database, i.e. $support(\alpha) \geq \zeta$. A moving sequential pattern α is called a maximal moving sequential pattern if there is not any other moving sequential pattern β, and $\alpha \subseteq \beta \wedge \alpha \neq \beta$.

The above definitions are similar to those of sequential pattern mining, but there are some differences, especially the definitions of subsequence and super sequence.

2.2 Problem Statement

Jian Pei et al. give a systematic study on constraint-based sequential pattern mining in[12]. In some way, it may take the problem of mining moving sequential pattern as the problem of mining sequential patterns with some constraints, which are shown as follows.

Constraint 1 An item constraint specifies that each item is a tuple, denoted by (c, T_s, T_e), $T_s, T_e \in T$, which is a time domain, so there is a natural time constraint on the item's time attribute. For example, for any two items $C_i = (c_i, T_{si}, T_{ei})$ and $C_j = (c_j, T_{sj}, T_{ej})$, and C_i must not appear after C_j in any sequence if $T_{ei} < T_{sj}$.

Constraint 2 An itemset constraint specifies that the length of an itemset in moving sequence must contain only one item, and this constraint makes a moving sequence an ordered list of items, not an ordered list of itemsets.

Constraint 3 An item order constraint specifies if two items are successive in a moving sequence α, and if $\alpha \subseteq \beta$, those two items must be successive in β. This constraint is guaranteed by the definitions of subsequence and super subsequence. This is because that the care is about what the next move is for a mobile user in mining moving sequential patterns.

Constraint 4 A network topology constraint specifies for any two successive elements $C_i = (c_i, T_{si}, T_{ei})$ and $C_j = (c_j, T_{sj}, T_{ej})$ $(j = i + 1)$ in a moving sequence, c_j must be the neighbor cell of the one of c_i.

In fact, each mobile user can only move into a neighbor cell in a wireless system. This constraint is hidden in the moving sequences and is guaranteed by the wireless topology structure.

Note The Apriori property plays an important role for efficient candidate pruning in mining sequential patterns. For example, $\langle ABC \rangle$ is a frequent length-3 sequence, and then all the length-2 subsequences $\{\langle AB \rangle, \langle AC \rangle, \langle BC \rangle\}$ must be frequent. In mining moving sequential patterns, $\langle AC \rangle$ is not a subsequence of $\langle ABC \rangle$ any more, from this meaning it still fulfill that any subsequence of a frequent moving sequence must be frequent, which is called **Pseudo-Apriori** property.

The studies on efficient sequential pattern mining and its applications can be categorized into two classes: (1) Apriori-based; (2) projection-based. The Apriori-based methods can efficiently prune candidate sequence patterns based on Aprior property, but in moving sequential pattern mining candidate sequences efficiently can't be pruned because the moving sequence only preserves Pseudo-Apriori property. In the meanwhile Apriori-based algorithms still encounter problem when a sequence database is large and/or when sequential patterns to be mined are numerous and/or long[8]. J. Han et al. propose a method, named FP-tree to mine frequent itemsets without candidate generation in[13], and FP-tree compresses the database into a set of conditional databases (a special kind of projected database), each associated with one frequent item, and mine each database separately. Projection-based methods adopt a divide and conquer idea to confine the search and the growth of subsequence fragments, and leads to efficient processing. Here based on the idea of Pseudo-Apriori property and projection, it is proposed that a novel and efficient moving sequential pattern mining algorithm based prefix trees, called PrefixTree.

2.3 PrefixTree Algorithm

Definition 1 (Prefix, Projection, and Postfix)

Given a moving sequence $\alpha = \langle a_1\ a_2\ \cdots a_n \rangle$, a moving sequence $\beta = \langle b_1\ b_2\ \cdots b_m \rangle (m \leq n)$ is called a **prefix** of α if and only if $b_i = a_i (1 \leq i \leq m)$.

Given moving sequences α and β such that β is a subsequence of α, i.e., $\beta \subseteq \alpha$. A subsequence α' of α (i.e. $\alpha' \subseteq \alpha$) is called a projection of α w.r.t. prefix β if and only if (1) α' has prefix β and (2) there is no proper super-sequence α'' of α' (i.e. $\alpha' \subseteq \alpha''$ but $\alpha' \neq \alpha''$) such that α'' is a subsequence of α and also has prefix β.

Let $\alpha' = \langle c_1\ c_2\ \cdots c_l \rangle$ be the projection of α w.r.t. prefix $\beta = \langle c_1\ c_2\ \cdots c_k \rangle$ ($k \leq l$). Sequence $\gamma = \langle c_{k+1}\ \cdots c_l \rangle$ is called the **postfix** of α w.r.t. prefix β, denoted by $\gamma = \alpha/\beta$. If β is not a subsequence of α, both projection and postfix of α w.r.t. β are empty.

From the above definitions, it can be seen that they are similar to the ones in PrefixSpan[14], and are changed according to the characteristics of moving sequences.

Definition 2 (Before Relation and After Relation)

For any two items, their time attributes have the following two relations:

For two items $C_i = (c_k, T_{si}, T_{ei})$ and $C_j = (c_m, T_{sj}, T_{ej})$, if $(T_{ei} < T_{sj})$, the item C_i is **before** item C_j, denoted by $C_i \vdash C_j$.

For two items $C_i = (c_k, T_{si}, T_{ei})$ and $C_j = (c_m, T_{sj}, T_{ej})$, if $(T_{ei} \geq T_{sj})$, the item C_i is **after** item C_j, denoted by $C_i \dashv C_j$.

From the above definitions, it can be seen that for any two items must fulfill either before or after relation, and if they fulfill before relation they cannot fulfill after relation, vice versa.

Now it is gotten an intuitionistic explanation about the before and after relations, suppose two items C_i and C_j both appear in a moving sequence. If $C_i \vdash C_j$, C_i must appear before C_j in a moving sequence, and if $C_i \dashv C_j$, C_i may appear after C_j in a moving sequence (sometimes C_i may appear before C_j).

PrefixTree Algorithm

The key idea of PrefixTree is the uses of prefix trees, so it will be show the algorithm how to construct the prefix trees. Prefix tree is a compact representation of candidate moving sequential patterns. It is easy to generate the moving sequential patterns based on the prefix trees. Every moving sequence from the root node to the leaf node is a candidate frequent moving sequences. All the moving sequential patterns can be gotten by scanning all the prefix trees once. The support of each node decreases with the depth increase, so a new frequent moving sequence is generated by tranversing the prefix trees from the root to the leaves when encountering a node whose count is less than the support threshold. The mining of mobile user maximal moving sequential patterns is divided into six steps as follows.

1) Scanning database once to find the set of frequent items, and then order them according to their values of T_s attribute.

2) Computing candidate successive items based on the after relation.

3) Generating frequent length-2 moving sequences, where the candidate frequent length-2 moving sequences are generated based on candidate successive items. After getting the frequent length-2 moving sequences, candidate successive items are reduced again based on Pseudo-Apriori property.

4) Constructing prefix trees based on candidate successive items and Pseudo-Apriori property.

5) Generating moving sequential patterns based on the prefix trees.

6) Generating maximal moving sequential patterns.

2.4 Analysis of PrefixTree Algorithm

Based on Pseudo-Apriori property, for any frequent length-n moving sequence $\alpha = \langle a_1\ a_2 \cdots a_n \rangle$, all the possible frequent length-$(n+1)$ moving sequences having prefix α have the form of $\langle a_1\ a_2 \cdots a_n b \rangle$, and if $\langle a_1\ a_2 \cdots a_n b \rangle$ is frequent, $\langle a_n b \rangle$ must be frequent based on the Pseudo-Apriori property, so b must belong to $CSI(a_n)$. From the above analysis, it is known that the prefix trees contain all the possible frequent moving sequences, and then PrefixTree algorithm can discover all the maximal moving sequential patterns.

It is needed to scan the moving sequence database once to generate the frequent items and to scan the database again to generate the frequent length-2 moving sequences and corresponding projected databases. Then the prefix trees are constructed by scanning all the projected databases, thus in all it is needed only to scan the database at most three times. Generating physical files for the projected databases is time costly work.

Now it can be considered that the cost of constructing the prefix trees. Suppose the length of the candidate moving sequential pattern is k, the maximum number of CSI of an item is m (m is limited by the number of neighbor cells), and the number of moving sequences is n. For each moving sequence, its information can be added into a prefix in $O(k\log m)$ time, so the time cost complexity for constructing all the prefix trees is $O(nk\log m)$, where k is usually a small number.

Suppose the number of nodes in a prefix tree is r, it can be gotten that all the sequential patterns can be gotten by traversing the tree once with the time complexity $O(r)$. Thus the generation of moving sequential patterns is very fast.

The generation of maximal moving sequential patterns is very simple, and the number of moving sequential patterns is much smaller than the number of moving sequences. Suppose the number of moving sequential patterns is p, and the maximal moving sequential patterns can be gotten within $O(p^2)$.

3 Experimental Results

All experiments are performed on a 1.7 GHz Pentium PC machine with 512 M main memory and 60 G hard disk, running Microsoft Windows 2000 Professional. All the methods are implemented using JBuilder 6.0.

The synthetic dataset used for the experiments comes from SUMATRA (Stanford University Mobile Activity TRAces)[14]. BALI-2: Bay Area Location Information (real-time) dataset records the mobile users' moving and calling activities in a day. The mobile user averagely moves 7.2 times in a day in 90

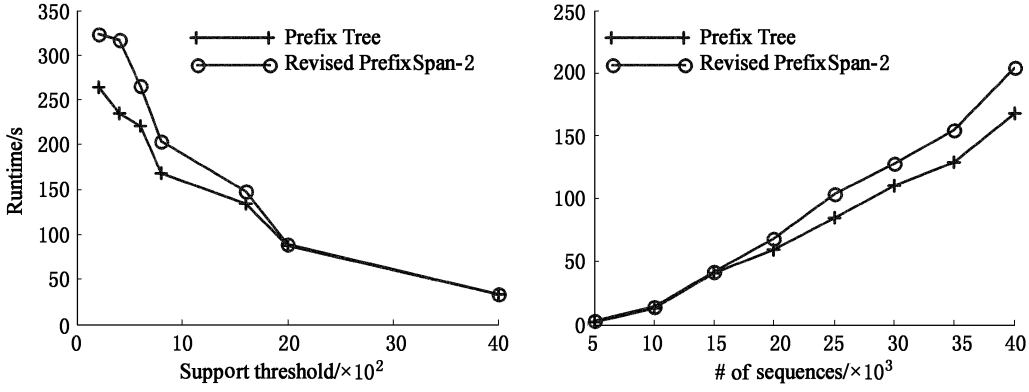
zones, so the average length of moving sequence is 8.2. about 40 000 moving sequences are extracted from BALI-2 used for the experiments.

Since the details of experimental analysis of CURD are also shown in[11], here the focus of attention is the experimental analysis of PrefixTree. The experimental results are reported on the performance of PrefixTree in comparison with PrefixSpan-2, which is recoded and revised to mine moving sequences. The reason we choose PrefixSpan-2 to compare with PrefixTree is as follows.

1) Apriori-based methods need effective candidate pruning based on Apriori property, but in mining moving sequential patterns only Pseudo-Apriori property is preserved, which decreases the algorithms' efficiency.

2) Projection-based methods adopt a divide and conquer idea to confine the search and the growth of subsequence fragments, and leads to efficient processing. PrefixSpan-2 is already a very fast algorithm in mining sequential patterns.

The experimental results of scalability with support threshold are shown in Fig. 1(a), and the number of moving sequences is about 40 000. When the support is high, there is only a limited number of moving sequential patterns, and the length of patterns is short, the two methods are close to each other according to their runtime. However as the support threshold decreases, PrefixTree is more efficient than Revised PrefixSpan-2.



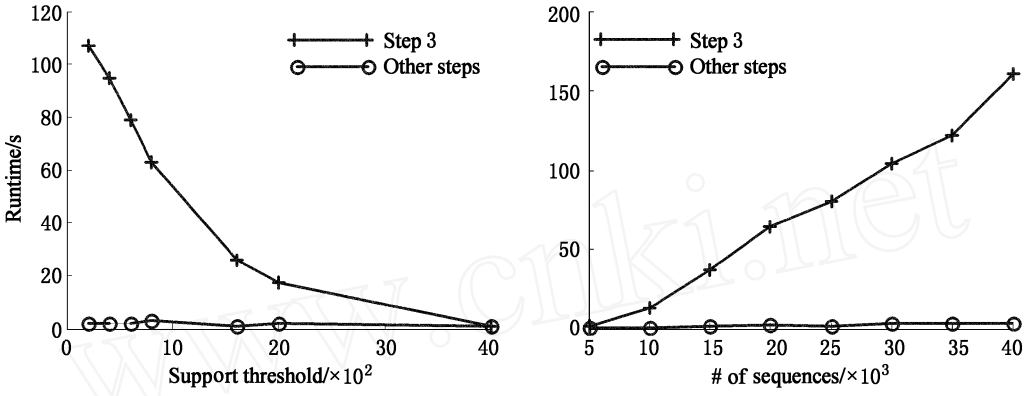
(a) varying support threshold　　(b) varying sequential number

Fig. 1　PrefixTree and Revised PrefixSpan-2

Fig. 1(b) shows the scalability of PrefixTree and Revised PrefixSpan-2 in terms of the number of moving sequences, and the support threshold is set to 800. Both methods are linearly scalable, but PrefixTree is more efficient than Revised PrefixSpan-2.

Fig. 2 shows the executing details of PrefixTree and Revised PrefixSpan-2 from the aspects of support threshold and the number of moving sequences respectively. The step 3, which needs to generate the projected databases and frequent length-2 moving sequences, occupies the main cost of PrefixTree. Compared with step 3 the other steps of PrefixTree occupy only a little time.

In summary, the performance study shows that PrefixTree is more efficient and scalable than Revised



(a) varying support threshold (b) varying sequential number

Fig. 2 Details of PrefixTree

PrefixSpan-2.

4 Discussion and Future Work

In this paper a clustering method is introduced to discretize the time attribute in moving histories, which transforms the original moving histories into moving sequences based on the clustering results. A novel and efficient method, called PrefixTree, is also proposed to mine the moving sequences. Its main idea is to generate candidate successive items and generate the prefix trees. Performance study shows that PrefixTree is more efficient and scalable than Revised PrefixSpan-2.

Since there is urgent need to mine the moving sequences more efficiently, so the next step will focus on how to incrementally mine the moving sequences.

Acknowledgement *The authors are thankful to professor Pei Jian for helping us to understand the PrefixSpan algorithm.*

References

1 Peng Wenchih, Chen Mingsyan. Mining User Moving Patterns for Personal Data Allocation in a Mobile Computing System. Online Publication, Proceedings of ICPP Conference. Toronto: IEEE Computer Society, 2000, 573～580

2 Naor Zohar, Levy Hanoch. Minimizing the Wireless Cost of Tracking Mobile Users: An Adaptive Threshold Scheme. Proceedings of INFOCOM Conference. CA: IEEE Computer Society, 1998. 720～727

3 Agrawal Rakesh, Srikant Ramakrishnan. Mining Sequential Patterns, In: Yu S. Philip, Chen L. P. Arbee, eds. Proceedings of ICDE Conference. Taibei: IEEE Computer Society, 1995. 3～14

4 Ramakrishnan Srikant, Rakesh Agrawal. Mining Sequential Patterns: Generalizations and Performance Improvements. In: Apers M. G. Peter, Bouzeghoub Mokrane, Gardarin Georges, eds. Proceedings of EDBT Conference. Avignon: Springer-Verlag, 1996. 3～17

5 Zaki J. Mohammed. SPADE: An Efficient Algorithm for Mining Frequent Sequences. Machine Learning, 2001, 41(1/2): 1～31

6 Ayres Jay, Gehrke Johannes, Yu Tomi, et al. Sequential Pattern Mining Using A Bitmap Representation. Proceedings of KDD Conference. Alberta: ACM Press, 2002. 429～435

7 Han Jiawei, Pei Jian, Mortazavi-Asl Behzad, et al. FreeSpan: Frequent Pattern-Projected Sequential Pattern Mining. Proceedings of KDD Conference. MA: ACM Press, 2000. 355～359

8 Pei J, Han J, Mortazavi-Asl B, et al. PrefixSpan: Mining Sequential Patterns Efficiently by PrefixProjected Pattern Growth. Proceedings of ICDE Conference. Heidelberg: IEEE Computer Society, 2001. 215～224

9 Das G, Lin K-I, Mannila H, et al. Rule Discovery from Time Series. In: Agrawal Rakesh, Stolorz E. Paul, Piatetsky-Shapiro Gregory, eds. Proceedings of KDD Conference. New York: AAAI Press, 1998. 16～22

10 Wu Hsiao-kuang, Jin Ming-hui, Horng Jorng-tzong. Personal Paging Area Design Based on Mobiles Moving Behaviors. Proceedings of INFOCOM Conference. Alaska: IEEE Computer Society, 2001. 21～30

11 Ma Shuai, Wang Tengjiao, Tang Shiwei, et al. A New Fast Clustering Algorithm Based on Reference and Density. In: Dong Guozhu, Tang Changjie, Wang Wei, eds. Proceedings of WAIM Conference. Heidelberg: Springer-Verlag, 2003. 214～225

12 Pei Jian, Han Jiawei, Wang Wei. Mining Sequential Patterns with Constraints in Large Databases. Proceedings of CIKM Conference. VA: ACM Press, 2002. 18～25

13 Han J, Pei J, Yin Y. Mining Frequent Patterns without Candidate Generation. In: Chen Weidong, Naughton F Jeffrey, Bernstein A Philip, eds. Proceedings of SIGMOD Conference. TX: ACM Press, 2000. 1～12

14 Stanford University Mobile Activity TRAces(SUMATRA). http://www-db.stanford.edu/sumatra/

移动环境中的最大移动序列模式挖掘

马　帅　唐世渭　杨冬青　王腾蛟　高　军

(北京大学 计算机科学技术系,北京,100871)

摘　要　在移动通信环境中,移动序列模式挖掘对于有效的提高位置管理的服务质量具有重大的意义。移动序列模式挖掘和传统的序列模式挖掘是不同的,首先,前者需要考虑更多的时间因素;其次,移动序列模式中的项之间是连续的,因为关心移动用户的下一次移动情况。本文提出了一种挖掘移动序列模式的新技术:聚类的思想引入到移动序列模式挖掘来处理移动历史的时间离散化,并且提出了一个高效的 PrefixTree 算法来挖掘移动序列。性能研究表明,PrefixTree 算法优于 PrefixSpan-2 算法。

关键词　聚类;序列模式挖掘;移动序列模式挖掘

中图分类号　TP 391