一种无线通信环境中用户移动模式的挖掘算法*

宋国杰[1], 唐世渭[1,2], 杨冬青[1], 王腾蛟[1], 叶恒强[3]

[1](北京大学 计算机科学技术系,北京 100871);
[2](北京大学 视觉与听觉信息处理国家重点试验室,北京 100871);
[3](广州新太科技股份有限公司,广东 广州 510665)
E-mail: gjsong@db.pku.edu.cn
http://db.cs.pku.du.cn

摘要: 发现无线通信环境中用户的移动模式是移动对象管理中的一个关键问题.提出一种快速挖掘该模式的算法 SAM(split and merge),用来挖掘移动对象所产生有序数据集中潜在的移动模式,从而为移动对象管理提供服务.该算法将自底向上搜索和自顶向下过滤技术相结合,采用图存储压缩数据集方法,利用非频繁项集分解子图和频繁长模式过滤数据集相结合的技术,大大减少了迭代次数,降低了 CPU 时间.最后给出了算法性能比较和算法分析.结果表明,该算法是有效的.

关 键 词: 数据挖掘;最大频繁项集;移动模式;移动对象管理;移动通信
中图法分类号: TP393 文献标识码: A

随着定位技术(如 GPS 等)、无线通信技术和电子技术的发展,使得我们可以对移动对象(如 PDA,车辆等)进行跟踪定位.移动对象管理与移动对象的跟踪、定位、记录、查询等密切相关,而这些功能的有效实现需要用户移动模式的有效支持.挖掘用户的移动模式不仅可以为移动对象管理服务,而且在交通管理、广告发送、安全、旅游等基于位置的服务中有着广泛的应用前景.

本文所要挖掘的移动模式是最大频繁移动模式集,这是因为在移动管理中往往需要最大模式[1,2].所谓最大移动模式是指在模式集中不存在任何模式为该模式的超集.求取最大模式集的方法有 Max-Miner[3],Pincer-Search[4]等.但它们有一个共同点,数据集是无序的、离散的,不适合求解连续数据集的情况.文献[1,2]提出从无线用户的移动日志中挖掘出用户的移动模式,但该算法是建立在 Apriori 思想的基础上,迭代求出所有模式后再得到最大模式,这显然是不合适的.文献[5]给出了在移动行为基础上预测用户移动的方法.文献[6,7]讨论了基于图求解的思想.

在文献[1,2]的基础上,结合求解问题的特性,我们给出了求解这一问题的方案:首先将数据集转化为移动模式图集,从而简化计数、压缩空间;然后采用自底向上和自顶向下相结合的策略,利用频繁项集向上搜索和非频繁项集向下分裂、合并、过滤移动模式图集相结合的方法,加快算法进程,提高算法效率.

本文第 1 节是定义.第 2 节给出移动模式挖掘的解决方法.第 3 节是性能比较.最后是总结.

* 收稿日期: 2001-12-08; 修改日期: 2002-04-09
基金项目: 国家重点基础研究发展规划 973 资助项目(G1999032705);北京大学-IBM 创新研究院资助项目

作者简介: 宋国杰(1975－),男,河南新乡人,博士生,主要研究领域为数据库,信息系统;唐世渭(1939－),男,浙江宁波人,教授,博士生导师,主要研究领域为数据库,信息系统;杨冬青(1945－),女,天津人,教授,博士生导师,主要研究领域为数据库,信息系统;王腾蛟(1974－),男,山东济南人,博士生,主要研究领域为数据库,信息系统;叶恒强(1964－),广东广州人,工程师,主要研究领域为移动计算.

1 定 义

为方便讨论,首先引进如下符号说明.移动对象的移动产生一个连续的运动轨迹,该运动轨迹可以由位置 l、时间 t 描述.为了使定义 1 中轨迹 P 和定义 2 中图 G 具有惟一的对应关系,我们在轨迹的边上标记一个连续的自然数列.移动对象的轨迹可形式描述如下:

定义 **1**. 设 l_i 是有限位置集中的一个元素,t_i 是到达该位置的时间,$l_i(t_i)$ 表示轨迹中的一个节点,$1 \le i \le n$,n 是序列长度,则称轨迹 $P \xrightarrow{1} l_1(t_1) \xrightarrow{2} l_2(t_2) \xrightarrow{3} \ldots \xrightarrow{n} l_n(t_n)$ 为对象移动的序列.节点集合记作 L,每条边与其对应的惟一序号形成集合 M.

由于对象的移动行为往往是一个周而复始的过程,所以上述轨迹是一个有向标记图,每个连续的运动轨迹对应一个有向标记图.我们可以利用下述定义描述该路径对应的图:

定义 **2**. 设 V 是图顶点的集合,$E \subseteq V \times V$ 是边的集合,$\mu : L \to V$ 是一函数,实现从轨迹 P 对应节点集合 L 到图顶点集 V 的一一映射,$\upsilon : M \to E$ 是一函数,实现从轨迹 P 对应集合 M 到图中边集的多对一映射,则称四元组 $G = (V, E, \mu, \upsilon)$ 是表示对象移动行为的有向标记图.

这样,我们就将定义 1 中的移动轨迹转换为定义 2 中的图.每个连续的运动轨迹就有一个惟一的有向标记图与之对应.转换例子如图 1 所示.图 1 是与该表对应的移动模式图(为作图方便,我们对时间做了特殊化,到达相同顶点具有相同的时间).

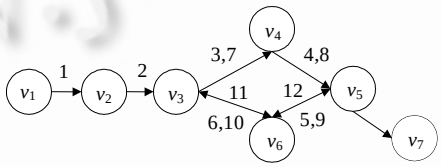

Fig.1 Moving pattern graph

图 1 移动模式图

定义 **3**. 设图 $S = (V_s, E_s, \mu_s, \upsilon_s)$ 所对应的运动轨迹,在其边对应的序号集合减去一个相同的常数 k 后,该轨迹是图 $G = (V, E, \mu, \upsilon)$ 对应轨迹的子集,则称图 S 为图 G 的子集,记作 $S \subseteq G$.

定义 **4**. 设若 $G' \subseteq G$ 且 $G \subseteq G'$,则图 $G' = G$.

2 挖掘最大移动模式

求解分两步进行:首先,将移动日志转换为移动模式图;然后,挖掘出对象最大移动模式.分述如下:

2.1 移动轨迹转换

移动日志是形如$((O_i,N_i),t)$记录组成的数据集,O_i 是起始位置,N_i 是终止位置,t 表示到达 N_i 的时间.为剔除随机移动模式,从而减少候选项集数量,在日志转换前,扫描一遍日志数据库,将所有非频繁的日志删除.

经上述预处理后,移动日志数据库是一个分段的日志序列,在理想的情况下,每个序列对应一个移动模式图.但是,如果由于特殊情况(信息采集失灵等)而无法捕捉到对象的移动信息,这时同一序列内部就会出现多个子序列,形成多个移动轨迹.轨迹转换算法描述如下:

算法 **1**. 移动轨迹转换算法.

输入:moving log database, min_sup;

输出:P_set, C_2. //P_set 移动轨迹集合

(1) Scan moving log database, filter infrequent log, get L_2

(2) set last_dest to null, set i to 1, set P to null //P 对应一条轨迹,last_dest 是上一记录的终止位置

(3) while (not end of moving log database)

(4) {set S=O_i and D=N_i

(5) if (last_dest=S) //如果连续则在 P 上添加新的节点

(6) Append (S, D) with to P; i++

(7) else{ //否则将该轨迹加入轨迹集合 P_set

(8) if (P∈P_set)

(9) P.count++

(10) else P_set=P_set∪P

(11) P=null; i=1

(12) Append (S, D) with to P}

(13) Update the occurrence count of (S, D) in C_2 //对候选 2 项集计数

(14) Last_dest=N_i}

这样,我们就将移动日志转换为一个移动轨迹集合 P_set 和已计数的候选 2 项集集合.

2.2 移动轨迹转换为移动模式图

依照定义 2 给出的方法,对移动轨迹 P_set 进行转换,得到图集 G_set.例子如下:

表 1 是过滤掉非频繁 2 项集后移动日志数据库中的一个连续移动日志序列,V 是图的顶点,m 表示本次移动在图中对应的序列号.

Table 1 Moving log sequence

表 1 移动日志序列

V	V_1	V_2	V_3	V_4	V_5	V_6	V_3	V_4	V_5	V_6	V_3	V_6	V_5
O	A	B	c	d	e	f	c	d	E	f	c	f	e
N	B	C	d	e	f	c	d	e	F	c	f	e	g(V_7)
T	7	8	9	13	17	8	9	13	17	8	17	13	15
m	1	2	3	4	5	6	7	8	9	10	11	12	13

2.3 最大模式挖掘

在挖掘长模式之前,先做这样的处理:将移动模式图集合 G_set 进行聚类,聚类准则是每个类内部图的顶点数目相同.现有的许多数据库系统提供这种聚类功能,如 IBM 的智能挖掘工具 Intelligent Miner.这样,集合 G_set 就转换为如下的形式:

$$G\_\text{set} = \{G_m \mid 1 \le m \le n\};\ G_m = \{G \mid node(G) = m\},$$

其中 n 是图中顶点最大数目,$node(G)$表示图 G 中顶点个数.

挖掘过程采用自底向上搜索和自顶向下过滤相结合技术.利用频繁项集 L_i 自底向上产生候选项集 C_{i+1},并扫描数据集 G_set 进行计数.同时利用非频繁项集 C_i 分解移动模式图集合 G_set,这样使得大量模式图分解为较小子图,然后将其合并到与之顶点数相同的集合之中.在合并后的 G_set 中,利用产生的频繁项集,自顶向下滤掉所有在 G_set 中被它包含的子图和频繁项集(可以证明这些子图对以后的挖掘过程是无用的),潜在地减少候选项集的数量.这一过程利用了分解(split)、与(and)、合并(merge)相结合的技术,因此,该算法称为 SAM 算法.

具体描述如下:

(1) 候选项集产生方法如下:如果有两个模式 $v_1v_2\ldots v_{n-1}v_n$,$v^1v^2\ldots v^{n-1}v^n$,如果 $v_2\ldots v_{n-1}v_n=v^1v^2\ldots v^{n-1}$,那么产生候选项集 $v_1v_2\ldots v_{n-1}v_nv^n$;如果 $v^2\ldots v^{n-1}v^n=v_1v_2\ldots v_{n-1}$,则产生候选项集 $v^1v^2\ldots v^{n-1}v^nv_n$.

(2) 对于任意候选项集 c∈C_k,计数的过程就是与 G_set 中所有 $node(G_m)$>$node(c)$的图集进行匹配.计数 intra_count 表示包含了 c 的图的个数;计数 inter_count 是 c 在图 G 中出现的次数,该候选项集 c 的计数为 (Inter_count$_1$+…+Inter_count$_n$),其中 n=Intra_count.

(3) 对于任意 c∈C_k 和所有 $node(G)$>$node(c)$的图 G,如果 c⊆G,则分解 G.举例说明分解过程:假设对于模式图 1,有非频繁项集 def,分解过程如下式所示,分解结果如图 2 所示.

$$V_1V_2V_3\underline{\boldsymbol{V_4}}\underline{\boldsymbol{V_5}}\underline{\boldsymbol{V_6}}V_3\underline{\boldsymbol{V_4}}\underline{\boldsymbol{V_5}}\underline{\boldsymbol{V_6}}V_3V_6V_5V_7 \Rightarrow V_1V_2V_3\underline{\boldsymbol{V_4}}\underline{\boldsymbol{V_5}}\underline{+\boldsymbol{V_5}}\underline{\boldsymbol{V_6}}V_3\underline{\boldsymbol{V_4}}\underline{\boldsymbol{V_5}}\underline{+\boldsymbol{V_5}}\underline{\boldsymbol{V_6}}V_3V_6V_5V_7.$$

(4) 假设分解后得到子图 g_1,且 $Node(g_1)$>$Node(C_k)$(C_k 为当前所求候选项集),其待合并模式图集为 $Node(G_m)$=$node(g_1)$的集合 G_m.如果有 g_2∈G_m 使得 g_2=g_1,则使 $g_2.count$ 加 1,否则将其并入集合 G_m.如果 $Node(g_1)$≤$Node(C_k)$,则将其删除.

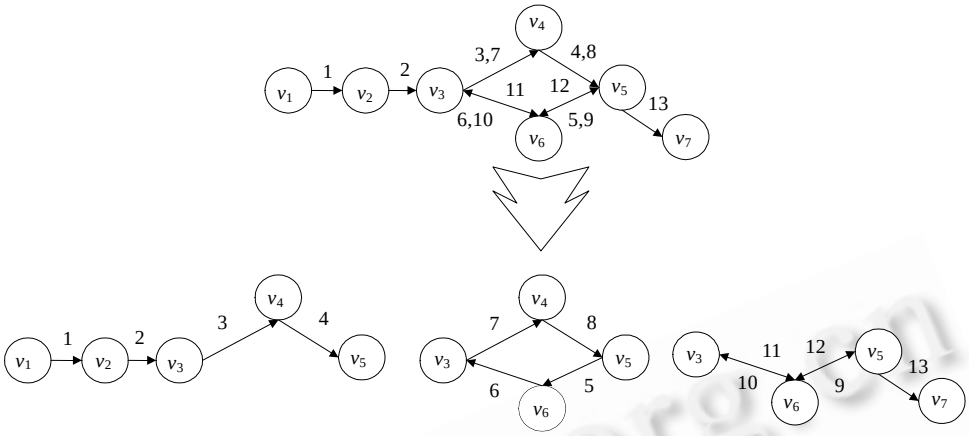

Fig.2 The result of division

图 2 分解结果

(5) 分解合并后过滤.过滤是在产生 C_{k+1} 之后进行,是为了防止过滤掉潜在的频繁项集.例如,$L_2=\{V_1V_2,V_2V_3,V_3V_5\}$,分解合并后得到频繁子图 $V_1V_2V_3$.如果首先过滤,则 $L_2=\{V_3V_5\}$,则造成潜在候选项集 $V_2V_3V_5$ 丢失.所以,首先产生 C_3,然后过滤,就会避免上述现象.过滤过程分两步:首先,如果图 G_set 中出现频繁子图 G,则删除 G_set,C_{k+1} 和 L_k 中所有满足 $g\subseteq G$ 的模式子图(项集);然后,利用当前得到的 L_k 进行对频繁项集 L_{k-1} 进行过滤.

2.4 SAM算法

2.4.1 算法描述

算法 **2**. 挖掘最长移动模式算法.

输入:G_set, min_sup;

输出:频繁移动长模式集合

```
(1)  Clustering (G_set)                    //方法见第 2.3 节
(2)  C3= Generate_candidate (L2)           //方法见第 2.3 节中的(1)
(3)  k=3
(4)  While (Ck ◊ null) {
(5)  Ck=candidate_count (G_set)            //方法见第 2.3 节中的(2)
(6)  for all c∈Ck
(7)     If (c.count<min_sup){
(8)        g=Split(G_set, c)               //方法见第 2.3 节中的(3)
(9)        Merge (G_set, g)}               //方法见第 2.3 节中的(4)
(10) Ck+1=Generate_candidate (Lk)
(11) Filter (G_set)                        //方法见第 2.3 节中的(5)
(12) k++}
(13) L={Li|i=2,…,k}∪{G|node(G)>k}
```

2.4.2 引 理

引理 **1**. 设 L 是算法求得的最大移动模式集合,则 L 是最小的.

证明:L 的最小性是指在 L 中不存在任何一个项集是集合中其他项集的子集,可由第 2.3 节中的(5)来保证.

引理 **2**. 设 L 是最大移动模式集,则其所包含的模式集皆为频繁项集.

证明:由算法知 $L=\{L_i|i=2,\ldots,k\}\cup\{G|node(G)>k\}$,$\{L_i|i=2,\ldots,k\}$ 的频繁性是显然的,下面利用反证法证明 $\{G|node(G)>k\}$ 也成立.

假设有 $g\in\{G|node(G)>k\}$是非频繁项集,表示为如下形式:$v_1v_2v_3\ldots v_i\ldots v_m,m>k$.

由移动模式图的生成算法可知,如下模式$\langle v_1,v_2\rangle\langle v_2,v_3\rangle\ldots\langle v_{m-1},v_m\rangle$皆为频繁二项集.由候选项集的定义可知,它们生成如下模式$\langle v_1,v_2,v_3\rangle\langle v_2,v_3,v_4\rangle\ldots\langle v_{m-2},v_{m-1},v_m\rangle$,可以判定其皆为频繁模式,因为如果有一个 3 项集是非频繁的,那么就会分解 g,则 g 就不会存在于$\{G|node(G)>k\}$中.

按照上述过程迭代,直至生成如下长度为 k 的模式$\langle v_1,v_2,\ldots,v_k\rangle\langle v_2,v_3,v_4,\ldots,v_{k+1}\rangle\ldots\langle v_{k-m+1},\ldots,v_m\rangle$也必为频繁的,理由同上.

矛盾:算法到 k 已经停止,如能生成上述模式集,则算法将继续进行,产生矛盾,从而假设不成立.

可知原命题成立. □

引理 **3**. 设 L 是最大频繁移动模式集,则集合 L 是完备的.

证明:L 的完备性是指移动模式子图中蕴涵的所有最长频繁模式都包含在 L 中.在 L 中,对于所有 $node(g)<k+1$ 的频繁模式,其完备性可由算法的迭代性保证.对于 L 中 $node(g)>k$ 的频繁模式,假设存在一个模式 L_m 是频繁模式,但不属于它,那么只有两种可能:(1) 迭代过程中被非频繁模式分解;(2) 被其超集模式过滤掉.对于(1)是不成立的,因为既然它是频繁模式,其所有子集也一定是频繁的,不存在被分解的可能;对于(2),如果存在其超集,则它就不是最长模式.

所以,集合 L 是完备的. □

由上述引理可知,集合 L 是我们所要求解的长模式集.

2.5 算法分析

从如下几方面对算法进行分析:(1) I/O 量.迭代算法中,候选 2 项集的数量是制约性能的主要因素,因而我们利用第 1 次 I/O 过滤掉所有的非频繁 2 项集;第 2 次 I/O 生成移动模式子图.(2) 扫描数据库规模.移动模式子图是等价的移动日志的描述,但却是以图的形式存储,压缩了数据空间.另外,在算法执行过程中,每求取一个长模式,就对数据集进行过滤,删除被其包含的子图.在候选 k 项集计数时,其匹配的图 G 皆满足 $Node(G)\geq k$,而不是扫描全部数据集,这保证了扫描数据集的最小性.(3) 迭代深度.影响迭代算法效率的因素之一就是有长度较长模式的存在,因为迭代的深度等于最长模式的长度.我们利用非频繁项集分解数据集,从而从顶部较早发现较长的模式,过滤求得频繁项集,从而加快算法结束进程,减少迭代次数.

3 性能比较

试验采用的数据集是 IBM 无线网络小组仿真程序产生的仿真数据,它可以有效地模拟移动对象在空间的移动,从而采集离散的数据点.比较对象是 LM 算法.试验的硬件环境是:OS 是 Windows 2000,CPU 为 P4,主频为 1.4G,主存为 512M.

试验目的:(1) 测试数据负载度;(2) 测试算法随支持度的变化情况;(3) 测试算法的扩展性;(4) 测试算法随节点数目的变化情况.

由于我们设计的算法是基于主存的,所以需要测试在何种条件下算法有效.我们假设支持度为 0.1%,在采样点位置集$|L|=30$ 的条件下,数据量增长,测试算法的负载度.测试结果如图 3 所示.结果表明,当算法所执行的数据量达到 200 万行时,我们的算法依然可以有效地执行.因为,对于个体的移动对象而言,它们总是活动在某些有限的区域,而且随机运动区域在过滤阶段已将之过滤掉,数据规模大大缩减.所以,在主存所产生的图的规模不会随着数据量的增长而呈线性增长,最终将趋向平缓.因此,我们的算法对于挖掘个体移动对象的移动模式是有效和可行的.

为了测试算法效率随支持度变化的情况,我们对算法 SAM 和 LM 在数据量为 200k、采样点位置集$|L|=30$时进行了测试.结果如图 4 所示,说明我们的算法 SAM 即使在较低支持度下也保持良好的性能,且执行效率较高.值得说明的是,我们所采用的比较对象 LM 也是假设其数据集在内存情况下进行比较的.其实,该算法是基于 Apriori 的思想模式,需要大量的 I/O.我们也进行了数据存放为外存情况的比较,结果表明,其效率大大降低.

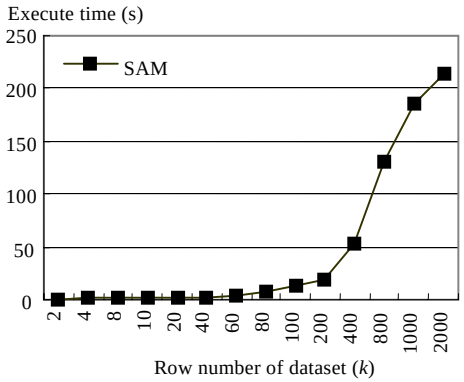

Fig.3 The test of SAM overload

图 3 SAM 负载度测试

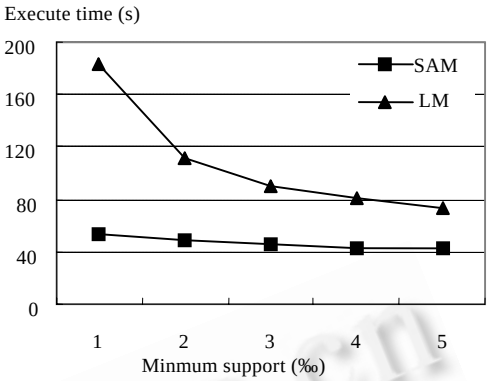

Fig.4 The test of support

图 4 支持度测试

为了测试在支持度和采样点集一定时,算法性能随数据规模变化的情况,我们测试了在支持度为 0.1%、采样点位置集|L|=30 时,数据量变化时的算法性能.结果如图 5 所示,说明随着数据量成比例的增大,算法 SAM 效率并不成比下降,具有很好的扩展性.

采样点数与算法的规模效应和效率密切相关.在数据量为 200k,支持度为 0.1%时,我们利用仿真程序产生不同的采样点集的数据.在此数据集上,我们测试算法的 LM 和 SAM 执行情况,结果如图 6 所示.结果表明,LM 受采样节点数目变化不大,但 SAM 受影响较大.这是因为,所着采样节点数目增高,主存图的规模开始相应增大,重合变少.但是,当采样节点增大到一定程度时,会有大量的路径因不满足支持度阈值而被过滤,图的规模重新变小,效率增高,形成如图 6 所示的马鞍状图形,但总体还是高效的.

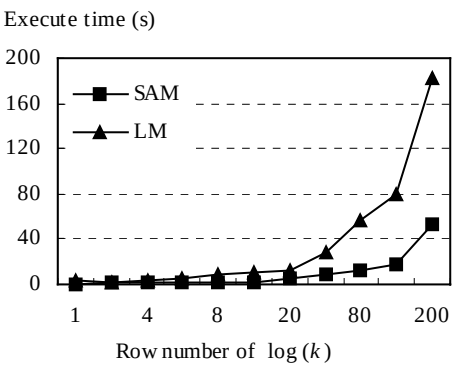

Fig.5 The scalable of algorithm

图 5 算法的扩展性

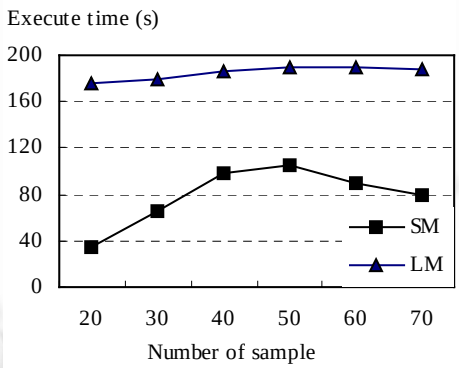

Fig.6 The influence of sampling point Number

图 6 采样点数目影响

4 小 结

本文给出了求解移动对象最大移动模式的有效解决方案.与传统方法不同之处在于:(1) 求解数据集是有序的,而不是离散的;(2) 挖掘过程不仅利用了频繁项集,而且利用了非频繁项集对数据集进行分解,从而缩小数据规模.这是以往的算法所没有的,同时也是有效的;(3) 利用图来存储数据集,从而压缩了算法扫描的数据空间;(4) 采用了有效的过滤技术,不仅过滤得到的项集,而且也对数据集进行过滤,这样就不必扫描那些明显无用的数据对象.如对计数候选 k 项集,不必扫描那些 $Node(G)<k$ 的数据集,而传统算法没有做到这一点.因此,我们提供了挖掘无限通信环境中用户移动模式的有效方法.

致谢 对 IBM 移动计算组成员和陈捷博士后、高军博士、马帅博士等人提出的宝贵建议表示感谢.

References:

[1] Peng, W-C., Chen, M-S. Mining user moving patterns for personal data allocation in a mobile computing system. In: Proceedings of the 29th International Conference on Parallel Processing. 2000.

[2] Peng, W-C., Chen, M-S. Developing data allocation schemes by incremental mining of user moving patterns in a mobile computing system. 2002. http://www2.ee.ntu.edu.tw/~mschen/msc.html.

[3] Bayardo, R. Efficiently mining long patterns from databases. In: Hass, L.M., Tiwary, A., eds. Proceedings of the ACM SIGMOD International Conference on Management of Data. New York: ACM Press, 1998. 85~93.Lin, Dao-I, Kedem, Z.M. Pincer-Search: a new algorithm for discovering the maximum frequent set. In: Schek, H.J., Saltor, F., Ramos, I., *et al*, eds. Proceedings of the 6th European Conference on Extending Database Technology. Heidelberg: Spring-Verlag, 1998. 105~119.

[5] Wu, H-K., Jin, M-H., Horng, J-T., *et al*. Personal paging area design based on mobile's moving behaviors. In: IEEE INFOCOM, 2001. 21~30.

[6] Nanopoulos, A., Manolopoulos, Y. Mining patterns from graph traversals. 2000. http://citeseer.nj.nec.com/nanopoulos01mining.html.

[7] Messmer, B.T., Bunke, H. Efficient subgraph isomorphism detection: a decomposition approach. IEEE Transactions on Knowledge and Data Engineering, 2000,12(2):307~323.

An Algorithm of Mining Personal Moving Patterns in a Wireless Communication Environment*

SONG Guo-jie[1], TANG Shi-wei[1,2], YANG Dong-qing[1], WANG Teng-jiao[1], YE Heng-qiang[3]

[1](*Department of Computer Science, Beijing University, Beijing* 100871, *China*);

[2](*National Laboratory on Machine Perception, Beijing University, Beijing* 100871, *China*);

[3](*XinTai Technology Co., Ltd, Guangzhou* 510665, *China*)

E-mail: gjsong@db.pku.edu.cn

http://db.cs.pku.du.cn

Abstract: Discovering moving pattern is a key problem of mobile management in wireless communication. In this paper, an algorithm named SAM (split and merge) is proposed to mine MFMP in sequential datasets of moving object, and then to provide services for moving object management. This algorithm combines the bottom-up search and top down filter and uses data structure——graph to store datasets, infrequent item sets to split moving pattern graph, and long moving pattern to filter data sets strategy, and then the iteration number and CPU time are reduced greatly. Lastly, the performance analysis and the comparison of the algorithms are provided. Experimental results show that the SAM algorithm outperforms other existing algorithms.

Key words: data mining; long pattern; moving pattern; mobile object management; mobile communication

* Received December 8, 2001; accepted April 9, 2002

Supported by the National Grand Fundamental Research 973 Program of China under Grant No.G1999032705; the Foundation of the Innovation Research Institute of PKU-IBM of China