第48卷第1期 中国科学技术大学学报 Vol.48,No.1
2018年1月 JOURNAL OF UNIVERSITY OF SCIENCE AND TECHNOLOGY OF CHINA Jan. 2018

文章编号:0253-2778(2018)01-0057-08

一种用户频繁移动模式并行挖掘算法

朱一波,鲍培明,吉根林

(南京师范大学计算机科学与技术学院,江苏南京 210046)

摘要:针对用户的日常移动轨迹进行挖掘,可以有效地发现隐藏在用户生活中频繁出现的移动规律,即用户频繁移动模式,提出了一种基于PrefixSpan算法的用户频繁移动模式并行挖掘算法PASFORM.该算法利用了新的剪枝策略,缩小了搜索空间;引入了时间约束,挖掘出的频繁移动模式带有时间属性;使用前缀树存储频繁移动模式,缩小了存储空间;采用了并行化方法,适用于海量时空数据的挖掘.实验结果表明,该方法能够快速有效地挖掘出用户频繁移动模式.

关键词:频繁移动模式挖掘;序列模式挖掘;前缀树;并行化

中图分类号:TP391 **文献标识码**:A doi:10.3969/j.issn.0253-2778.2018.01.008

引用格式:朱一波,鲍培明,吉根林. 一种用户频繁移动模式并行挖掘算法[J]. 中国科学技术大学学报,2018,48(1):57-64.

ZHU Yibo, BAO Peiming, JI Genling. A parallel algorithm for mining user frequent moving patterns [J]. Journal of University of Science and Technology of China, 2018,48(1):57-64.

A parallel algorithm for mining user frequent moving patterns

ZHU Yibo, BAO Peiming, JI Genling

(*College of Computer Science and Technology, Nanjing Normal University, Nanjing* 210046, *China*)

Abstract: Through daily moving trajectories, one can effectively find the frequent moving rules, i.e., user frequent moving patterns. Based on PrefixSpan algorithm, a parallel algorithm named PASFORM is presented for mining user frequent moving patterns. PASFORM uses a new pruning strategy to reduce the search space and several time constraints to make mining results time-tagged. It also employs the parallel method to mine mass data and a prefix tree to save the store space. Experimental results show that PASFORM is effective and efficient.

Key words: frequent moving pattern mining; sequential pattern mining; prefix tree; parallelization

0 引言

现实生活中,移动对象的轨迹信息,尤其是用户的移动轨迹,通常表现出一定的规律.用户频繁移动模式[1]就是隐藏在大量用户的日常移动轨迹中、频繁出现的移动规律.挖掘用户的频繁移动模式可以进一步发现语义层移动模式[2]、分析居民时空行为[3]、挖掘周期模式[4]、预测用户移动模式[5],对交通管理、基于位置的服务、个性化推荐等有着极为重要的意义.

目前关于频繁移动模式挖掘的方法主要分为两大类,第一类是通过轨迹聚类来发现用户频繁移动

收稿日期:2017-05-20;**修回日期**:2017-06-23

基金项目:国家自然科学基金(41471371)资助.

作者简介:朱一波,男,1994年生,硕士生.研究方向:数据挖掘及其应用. E-mail: zyb0619@163.com

通讯作者:鲍培明,硕士/副教授. E-mail: baopeiming@163.com

模式[6-8]，第二类是通过序列模式挖掘来发现用户频繁移动模式[9-13].通过轨迹聚类的方法发现的移动模式包含的信息量丰富，但是针对性不强、不易于理解，且处理过程代价大、时间复杂度高；通过序列模式挖掘方法发现的移动模式针对性强、挖掘结果符合现实规律，但是现有的研究工作只是简单地运用序列模式挖掘的方法，没有充分考虑时间因素，缺乏通用性，并且很难适用于海量的时空数据.

本文借助序列模式挖掘的方法，提出了一种基于PrefixSpan改进算法的频繁移动模式并行挖掘算法(a parallel algirithm for mining user frequent moving patterns with time-constraints, PASFORM).

(Ⅰ) 利用新的剪枝策略，缩小了搜索空间；

(Ⅱ) 引入时间约束，挖掘出的频繁移动模式带有时间属性；

(Ⅲ) 提出并行挖掘算法，适用于海量时空数据的挖掘；

(Ⅳ) 使用前缀树存储频繁移动模式，缩小了存储空间.

1 问题定义

定义 1.1(轨迹) 轨迹(Tra)由一系列按时间先后排序的轨迹点组成，Tra=<(d_1,t_1), (d_2, t_2), …, (d_n,t_n)> ，其中，d_i表示轨迹点(预处理后用簇号或网格号标识)，t_i表示用户经过该轨迹点对应的时间，$1\leqslant i\leqslant n$.所有轨迹点的集合记作 I.

定义 1.2(元素) 在某一时间段内，出现频率大于等于阈值 δ 的轨迹点称为代表轨迹点，记为 ID，ID$\in I$.元素由时间段和该时间段内的代表轨迹点集合组成.时间段用正整数标识，记为 T；代表轨迹点的集合记作 G.G 与 T 之间用“:”分隔.G 中可能存在1个或多个不重复的ID，多个ID之间用“,”分隔并按字典序排序.如果该时间段内缺失数据，则ID用−1表示；如果没有数据缺失，但是不存在代表轨迹点，则ID用0表示.

例如某元素 e,$e.G=\{2, 5\}$,$e.T=2$,e 简写为 2, 5:2.

定义 1.3(移动序列) 移动序列由 L 个元素按时间顺序组合而成(各元素对应的时间段长度相同且不重叠)，记作 S,$S=<s_1; s_2; \cdots; s_L>$,$s_i$表示移动序列中的元素，$1\leqslant i\leqslant L$，元素之间用“;”分隔.$L$ 称为移动序列长度，S 又称为L-序列.一个用户所有的移动序列构成移动序列集，记作 SD.SD 中每一个元组用<SN,S>表示，其中 SN 用于唯一标识移动序列 S.

假设部分移动序列数据集 SD 如表1所示，其中，SN 用整数唯一标识用户的移动序列，时间段 $T\in\{1, 2, 3, 4\}$,4个时间段分别对应的时间范围为 00:00:00 ～ 05:59:59,06:00:00 ～ 11:59:59, 12:00:00 ～ 17:59:59,18:00:00 ～ 23:59:59.

表1 移动序列数据集 SD

Tab.1 Mobile dataset SD

序列标识 SN	移动序列 S
1	<1:1; 2, 5:2; 3:3; 4:4>
2	<1:1; 3:2; 4:3; 3, 5:4>
3	<1:1; 2:2; 3:3; 4:4>
4	<1:1; −1:2; 3:3;4:4>
5	<−1:1; 0:2;3:3; 4:4>

SN=1 的移动序列<1:1; 2, 5:2; 3:3; 4:4>中，在 $T=1$ 的时间段内存在 ID=1 的一个代表轨迹点；在 $T=2$ 的时间段内存在 ID 分别为 2 和 5 两个代表轨迹点.SN=5 的移动序列<−1:1; 0:2; 3:3; 4:4>中，在 $T=1$ 的时间段内 ID= −1，该时间段内缺失数据；在 $T=2$ 的时间段内 ID=0，说明该时间段内没有频繁出现的轨迹点.SD 中的移动序列又称为4-序列.

定义 1.4(子序列和超序列) 如果存在移动序列 $\alpha=<a_1; a_2; \cdots; a_n>$和 $\beta=<b_1; b_2; \cdots; b_m>$，且存在 $1\leqslant j_1<j_2<\cdots<j_n\leqslant m$，使得 $a_1.G\subseteq b_{j1}.G$, $a_2.G\subseteq b_{j2}.G$, …, $a_n.G\subseteq b_{jn}.G$ ，且 $a_1.T=b_{j1}.T$, $a_2.T=b_{j2}.T$, …, $a_n.T=b_{jn}.T$，则序列 α 称为序列 β 的子序列，序列 β 称为序列 α 的超序列，记作 $\alpha\subseteq\beta$.

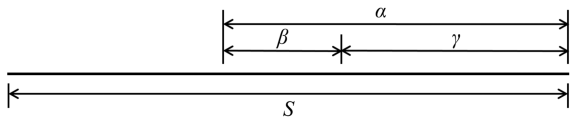
定义 1.5(前缀) 给定移动序列 $\alpha=<a_1; a_2; \cdots; a_n>$和 $\beta=<b_1; b_2; \cdots; b_m>(m\leqslant n)$，如果满足条件(1)$b_i=a_i(i\leqslant m-1)$；(2) $b_m.G\subseteq a_m.G$, $b_m.T=a_m.T$；则 β 称为 α 的前缀.

定义 1.6(投影) 给定移动序列 S、α 和 β，且 α、β 是 S 的子序列，即 $\alpha\subseteq S$,$\beta\subseteq S$，如果满足条件(Ⅰ)β 是 α 的前缀；(Ⅱ)不存在 α 的超序列 $\alpha'(\alpha\subseteq\alpha'$且 $\alpha\neq\alpha')$，使得 α'是 α 的子序列且也有前缀 β；则 α 称为 β 在 S 上的投影.

定义 1.7(后缀) 给定移动序列 S、α 和 β，α 是

β 在 S 上的投影，假设 $\alpha=<b_1; b_2; \cdots; b_{m-1}; b_m'; a_{m+1}; \cdots; a_n>$，$\beta=<b_1; b_2; \cdots; b_{m-1}; b_m>$ $(m\leqslant n)$.若移动序列 $\gamma=<a_m; a_{m+1}; \cdots; a_n>$，$a_m.G=b_m'.G-b_m.G$，$a_m.T=b_m.T$，则 γ 称为 β 在 α 上的后缀.

前缀、投影和后缀的关系如图 1 所示.对于表 1 移动序列数据集，记 SN=1 的移动序列 $S=<1:1; 2,5:2; 3:3; 4:4>$，移动序列 $\alpha=<2,5:2; 3:3; 4:4>$、$\beta=<2:2>$都是 S 的子序列，β 是 α 的前缀，α 是 β 在 S 上的投影，移动序列 $\gamma=<5:2; 3:3; 4:4>$ 是 β 在 α 上的后缀.



$\alpha\sqsubseteq S, \beta\sqsubseteq S, \beta$是$\alpha$的前缀, α是β在S上的投影, γ是β在α上的后缀

图 1　前缀、投影、后缀

Fig.1　Prefix, projection and suffix

定义 1.8(频繁移动模式)　移动序列 α 在序列数据集 SD 中的支持数记为 $\text{support}_{SD}(\alpha)$，$\text{support}_{SD}(<SN, S>\in SD) \wedge (\alpha\subseteq S)\}|$，即 SD 中 α 的超序列的个数.

给定最小支持数 ε，如果 $\text{support}_{SD}(\alpha)\geqslant\varepsilon$，即 SD 中至少有 ε 个移动序列是 α 的超序列，则 α 称为 SD 中的频繁移动模式.如果移动模式 α 的长度为 L，则 α 称为 L-模式.

频繁移动模式挖掘的问题可以描述为：给定用户移动轨迹 Tra 的集合和最小支持数 ε，找出所有的频繁移动模式.

2　相关工作

2.1　序列模式挖掘

序列模式挖掘[14]最早是由 Agrawal 等提出的，用来挖掘涉及事务间关联关系的模式.序列模式挖掘是对关联规则挖掘的进一步推广，现已广泛应用到客户购买行为预测、Web 访问模式预测、疾病诊断、自然灾害预测和 DNA 序列分析等领域[15].

经典的序列模式挖掘算法可以分为基于 Apriori 特性和基于模式增长两大类[16].基于 Apriori 特性的算法利用 Apriori 性质启发式地求解问题，易于理解、实现简单，但是需要多次扫描数据库，挖掘过程中会产生大量候选序列，不适合大数据库和长序列模式的挖掘.代表算法有 AprioriAll 算法[14]、GSP 算法[17]等.基于模式增长的算法采用分治的策略，不产生候选序列，利用投影数据库不断缩小搜索空间，特别适合稠密数据集中的挖掘问题.代表算法有 FreeSpan 算法[18]、PrefixSpan 算法[19]等，其中应用较多的是 PrefixSpan 算法.

PrefixSpan 算法运行过程中，主要耗时在递归构造投影数据库.如果序列数据集较大，项数种类较多，则算法运行速度会明显下降，因此 PrefixSpan 算法不适合挖掘长序列和海量时空数据.

2.2　频繁移动模式

借助序列模式挖掘的方法可以发现用户频繁移动模式.与序列模式挖掘相对应，频繁移动模式挖掘方法也可分为基于 Apriori 特性和基于模式增长两类，其性能与序列模式挖掘方法相似.本文研究的是基于模式增长的频繁移动模式挖掘方法.

现有的研究工作只是简单地应用序列模式挖掘的方法，虽然针对性强、挖掘结果符合现实规律，但是没有充分考虑时间因素，缺乏通用性，并且很难适用于海量的时空数据.文献[9-10]仅仅考虑了移动对象移动位置的先后顺序.文献[11-12]在数据预处理时引入了时间约束，但挖掘过程中没有考虑时间约束.文献[10,13]虽然在挖掘过程考虑了时间约束，但通用性不强，不适用于海量时空数据的挖掘.

本文提出了一种基于 PrefixSpan 改进算法的频繁移动模式挖掘并行化算法，能够解决频繁移动模式挖掘的问题，同时考虑了时间因素，适用于海量时空数据的挖掘且具有一定的通用性.

3　PASFORM 算法

3.1　频繁移动序列树

频繁移动序列树[20]是一颗前缀树，记作 BT，其根结点存储了最小支持数，除根结点外，BT 的每个结点都包含了两项属性：元素和以该元素为最后一个元素的频繁移动模式的支持数.从根结点到任意其他结点的路径代表了一个频繁移动模式，路径的长度为该频繁移动模式的长度，路径终结点的支持数为该频繁移动模式的支持数.任一结点的支持数不小于其孩子结点的支持数.

从表 1 的 SD 中挖掘出来的频繁移动模式如表 2 所示.第 1 列为 1-频繁项 β，第 2 列是 β 在投影 α 上的后缀 γ，第 3 列是从 SD 中挖掘出的以 β 开始的频繁移动模式及其支持数.

表 2 移动序列数据集 SD 的移动模式

Tab.2 Moving mode of mobile sequence dataset SD

前缀	后缀	频繁移动模式及其支持数
<1:1>	<2,5:2;3:3;4:4>, <3:2;4:3;3,5:4>, <2:2;3:3;4:4>, <3:3;4:4>	<1:1>:4,<1:1;2:2>:2,<1:1;2:2;3:3>:2, <1:1;2:2;3:3;4:4>:2,<1:1;2:2;4:4>:2, <1:1;3:3>:3,<1:1;3:3;4:4>:3, <1:1;4:4>:3
<2:2>	<5:2;3:3;4:4>, <3:3;4:4>	<2:2>:2,<2:2;3:3>:2,<2:2;3:3;4:4>:2, <2:2;4:4>:2
<3:3>	<4:4>,<4:4>, <4:4>,<4:4>	<3:3>:4,<3:3;4:4>:4
<4:4>		<4:4>:4

SD 中挖掘出来的未剪枝的频繁移动序列树如图 2 所示,其根结点存储了用户自定义的最小支持数 ε=2.从根结点经过结点 A、B、C 到结点 D 的路径,代表了移动模式 S=<1:1; 2:2; 3:3; 4:4>,S 的支持数为路径终结点 D 的支持数 2.从根结点到结点 A、B、C 的路径分别代表移动序列<1:1>、<1:1; 2:2>、<1:1; 2:2; 3:3>,它们都是 S 的前缀,也都是频繁移动模式.由于图 2 中虚线框内的序列都是 S 的子序列,且重复出现,所以在 PASFORM 算法的执行过程中会对该树进行剪枝,剪枝后虚线框内的序列不再保存在树中.

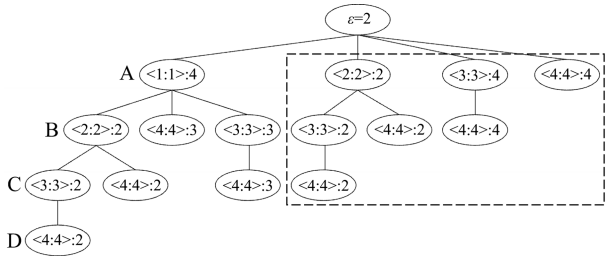


图 2 移动序列数据集 SD 的频繁移动序列树(未剪枝)

Fig.2 The frequent moving sequence tree of the mobile sequence dataset SD(unpruning)

3.2 算法思想

频繁移动模式挖掘是从一系列用户移动轨迹中找到频繁出现的、有规律的移动序列,挖掘的过程就是构建频繁移动序列树的过程.挖掘算法首先将用户移动轨迹转换为用户移动序列,然后采用序列模式挖掘方法发现用户频繁移动模式.挖掘算法的步骤包括 3 步:轨迹数据预处理,并行产生候选频繁序列,并行计算频繁序列支持数.

轨迹数据预处理完成用户移动轨迹 Tra 到移动序列数据集 SD 的转换.在并行产生候选频繁序列步骤中,构建频繁移动序列树根结点的各个子树.将 SD 按元组划分为若干个不相交集合,分发到各节点(计算机)上,并行发现各结点的候选频繁序列并更新子树.然后,合并所有子树,形成一棵缺少支持数的候选频繁移动序列树.在并行计算频繁序列支持数步骤中,计算候选频繁序列的支持数.同样地将 SD 按元组划分为若干个不相交集合,分发到各节点上,并行计算候选频繁序列在各结点的支持数.然后,汇总得到总支持数,将总支持数填入频繁移动序列树的对应位置.最后,去除不频繁序列,形成一棵完整的频繁移动序列树,步骤见算法 3.1.

算法 3.1 频繁移动模式并行挖掘算法

输入:用户移动轨迹 Tra 的集合,最小支持数 ε

输出:所有的频繁移动序列模式

第 1 步:轨迹数据预处理

利用轨迹聚类、网格划分等方法,将用户移动轨迹 Tra 的集合转换为由簇号或者网格号组成的移动序列数据集 SD.

第 2 步:并行产生频繁序列

2.1 将 SD 划分为 m 个互不相交的集合,分发到不同结点,结点 P_i 上的数据集记为 SD_i,$1 \leqslant i \leqslant m$;

2.2 各结点并行产生候选频繁序列:

2.2.1 初始化 $\beta=\varnothing$;构造一棵只有根结点的频繁移动序列树 BT_i,根结点保存 ε;建立一个空的二维表 Map_i(Map_i 的每个元组记为<β, $SD_i|\beta$>,第一个字段 β 记录前缀,第二个字段 $SD_i|\beta$ 记录 β 在 SD_i 上的所有后缀构成的数据集);

2.2.2 调用 Mod-PrefixSpan(β, SD_i, ε, BT_i, Map_i)(算法 2),得到频繁移动序列树 BT_i;

2.3 合并 BT_i 的根结点,得到频繁移动序列树 $BT=BT_1 \cup BT_2 \cup \cdots \cup BT_m$,此时 BT 缺少各候选频繁序列的支持数.

第 3 步:并行计算候选频繁序列支持数

3.1 将 SD 划分为 n 个互不相交的集合,分发到不同结

点，结点 P_j 上的数据集记为 SD_j，$1 \leqslant j \leqslant n$；

3.2 各结点扫描本地数据集 SD_j，计算 SD_j 对 BT 中各候选频繁序列的支持数；

3.3 汇总各结点的支持数，得到 BT 中各候选频繁序列的总支持数.

3.4 将总支持数填入 BT 的对应位置，去除 BT 中支持数小于 ε 的序列，形成一棵完整的频繁移动序列树，遍历 BT 输出所有的频繁移动模式.

算法 3.2　Mod-PrefixSpan(β, SD, ε, BT, Map)（改进的 PrefixSpan 算法）

输入：m-序列 $\beta = < b_1; b_2; \cdots; b_m >$，移动序列数据集 SD，最小支持数 ε，频繁移动序列树 BT，二维表 Map

输出：更新后的频繁移动序列树 BT

第 1 步：找出 β 在 SD 上的所有后缀，构成后缀数据集 SD|β.如果 β 为空，则 SD|β 就是移动序列数据集 SD.

第 2 步：当满足下列情况之一时，就放弃对 BT 的更新，算法结束.否则，将元组<β, SD|β>加入到 Map 中.

1SD|β 中的元组个数小于 ε；

2Map 中存在元组<β, SD|β>；

③ Map 中不存在元组<β, SD|β>，但存在元组<α, SD|β>，其中 α 是 β 的超序列.

第 3 步：遍历 SD|β，找到所有的 1-频繁项构成的元素 e，对每一个元素 e 执行步 4～步 6.若不存在 1-频繁项，则算法结束.

第 4 步：$\beta' = \beta$；

第 5 步：若 e.T=b_m.T，则把 e.G 添加到 β'的最后一个元素 b_m 中，同时更新 BT 中 b_m 对应的结点，即 b_m.G=b_m.G$\cup$e.G；

否则，给 β'添加一个新元素 b_{m+1}=e；若 m==0，则 BT 的根结点插入一个新结点 b_{m+1}，否则，BT 中在 b_m 结点后插入一个新结点 b_{m+1}.

第 6 步：递归调用 Mod-PrefixSpan(β', SD, ε, BT, Map).

3.3　算法分析

PASFORM 是基于 PrefixSpan 算法的改进算法，通过前缀投影挖掘序列模式，挖掘过程中不产生候选序列，采用基于投影的分治方法，投影序列收缩很快，大大缩减了检索空间.

PASFORM 算法引入了新的剪枝策略，分为更新时和更新后两种剪枝策略：算法 2 中依据步 2 的三种情况，通过 Map，表实现了 BT 树的更新时剪枝；在算法 1 的步 3.4 中，更新 BT 中候选频繁移动序列的支持数后，去除 BT 中支持数小于 ε 的序列，实现了 BT 树的更新后剪枝.

PASFORM 算法引入了时间约束，挖掘出的频繁移动序列都附带时间信息，能够反映出大量用户频繁出现的时空轨迹，而不仅仅是频繁的空间轨迹.

PASFORM 算法是数据密集型的并行算法，采用“分而治之”的策略，将原数据集划分为若干互不相交的子集，在各子集上并行处理任务.该算法包含两个并行化任务，第一个任务是并行产生候选频繁序列，第二个任务是并行计算各候选频繁序列的支持数，去除不频繁的序列.

PASFORM 算法使用频繁移动序列树存储频繁移动模式，缩小了存储空间：如果某个序列或者它的超序列在树中已经存在，则树中不再保存这个序列；如果树中的某个序列不频繁，则树中也不保存这个序列，即频繁移动序列树只保存频繁序列，不保存候选频繁序列.

频繁移动序列树存储了所有满足最小支持数的移动模式，深度遍历该频繁移动序列树可得所有的频繁移动模式.当最小支持数变大时，我们只需再次遍历该频繁移动序列树，去除不满足要求的移动模式，而不需要重新挖掘原移动序列数据集.当最小支持数变小时，需要重新挖掘原移动序列数据集.

PASFORM 适用于任何类型的轨迹数据，如 GPS 经纬度、手机基站、Wi-Fi、蓝牙，只需在挖掘前将轨迹数据作预处理，如利用轨迹聚类、网格划分等，将轨迹转换为簇号或网格号组成的移动序列.

4　实验结果和分析

4.1　Reality mining

第一个实验的实验数据为 MIT 多媒体实验室收集的 reality mining 数据集[21]，该数据集记录了 94 个志愿者从 2004 年 9 月至 2005 年 6 月（共 9 个月）的数据，数据信息包括通话记录、蓝牙信息、手机基站等.这 94 个志愿者中，68 个是一起工作的同事（90%是毕业生，10%是公司员工），剩下 26 个是商学院的新生.

实验分析了 PASFORM 算法挖掘出的用户频繁移动模式的有效性.实验选取了有效移动轨迹数据量较多的 2004 年的 9～12 月时间段，81 位数据较多的用户进行实验.我们将轨迹数据转换为手机基站号组成的移动序列，移动序列按天划分，每小时为一个时间段.实验采用的参数为 $L=24$，$\delta=0.1$，$\varepsilon=0.1$.以 81 位用户在 2004 年 11 月的轨迹数据为例，挖掘结果如表 3 所示.

表 3 用户频繁移动模式(按时间段对齐)

Tab.3 User frequent mobile mode

中上旬	25234:10;		25234:12;	25234:13;	25234:14;	25234:15;	25234:16;	25234:17;	25234:18
		25234:11;		25234:13;	25220:14;	25220:15;	25234:16;	25234:17;	25234:18
下旬	26377:10;	26377:11;	26377:12;		26377:14;		26377:16;	26377:17	26346:18
	26377:10;	26377:11;		26377:13;	26377:14;	26377:15;		26346:17	26346:18

表 3 中的用户频繁移动模式是挖掘出来的最长频繁移动模式.轨迹数据转换成的移动序列数据集共有 2 000 多条移动序列,挖掘出来的最长频繁移动模式共有 4 个.再次扫描移动序列数据集,可以得到这些频繁移动模式大致发生的时间段.比如,前两个模式主要发生在 2004 年 11 月的中上旬,后两个模式主要发生在 2004 年 11 月的下旬.中上旬的频繁移动模式在时间 10 点～18 点内主要在基站 25234 的范围内;下旬对应时间内,主要在基站 26377 的范围内.由此可见,从 11 月中上旬到下旬,这些用户的移动规律发生了变化,活动范围从基站 25234 移动到了基站 26377.

根据 MIT 实验室的身份调查报告,大部分用户是一起工作的同事,移动规律存在很强的共现性.实验证明,PASFORM 算法的挖掘结果与身份调查报告一致.

4.2 City bike

第二个实验的实验数据为纽约市的公共自行车数据集[22],数据内容是纽约市 2014 年 8 月中上旬公共自行车的租赁记录,数据信息包括租赁时间、租赁点位置、自行车 ID、用户信息等.

实验中,我们用 PASFORM 算法挖掘纽约居民的频繁移动模式.我们将居民频繁经过的地点在地图上标注出来,可视化后如图 3 所示.通过图 3 大致可以看出纽约的居民分布情况和出行规律.

我们取支持度排名前 8 的频繁移动模式,列出骑行线路及对应的骑行时间段,如表 4 所示.表 4 中地点 Central Park S & 6 Ave、Grand Army Plaza & Central Park S 和 Broadway & W 60 St.我们将频繁移动模式在地图上标注出来,如图 4 所示.分析该移动模式,我们发现一些比较有趣的现象:频繁移动模式中支持度最高的几个模式依次是:起止点都是中央公园的模式、起止点都是大将军广场的模式、起止点都是百老汇大道的模式和从大将军广场到中央公园的模式.中央公园和大将军广场是纽约著名的休闲和旅游胜地,挖掘出来的移动模式在 9 点到 21 点的每个时间段内都很频繁;百老汇大道两旁分布着众多的剧院,是纽约欣赏歌舞剧最繁华的地段,挖掘出来的频繁移动模式的时间段集中在 18 点到 21 点.由此可见,挖掘出来的居民频繁移动模式符合纽约当地情况,有一定的事实根据.

图 3 居民频繁驻留点可视化

Fig.3 Visualization of resident frequent residency

图 4 居民频繁移动模式可视化

Fig.4 Visualization of residents' frequent mobile mode

表 4　居民频繁移动模式(支持度排名前 8)

Tab.4　Resident frequent mobile mode(support is located in the top 8)

序号	起点	终点	时间段
1	Central Park S & 6 Ave	Central Park S & 6 Ave	12 点～15 点
2	Central Park S & 6 Ave	Central Park S & 6 Ave	15 点～18 点
3	Central Park S & 6 Ave	Central Park S & 6 Ave	18 点～21 点
4	Grand Army Plaza & Central Park S	Grand Army Plaza & Central Park S	12 点～15 点
5	Grand Army Plaza & Central Park S	Grand Army Plaza & Central Park S	15 点～18 点
6	Central Park S & 6 Ave	Central Park S & 6 Ave	9 点～12 点
7	Broadway & W 60 St	Broadway & W 60 St	18 点～21 点
8	Grand Army Plaza & Central Park S	Broadway & W 60 St	15 点～18 点

4.3　性能分析

本文的性能分析实验使用的是 reality mining 数据集及以该数据为种子数据的合成数据.

4.3.1　时间性能

实验对 PrefixSpan 算法与 PASFORM 算法的时间性能作了比较.实验使用的数据是合成数据,将 81 位用户在 2004 年 11 月的移动轨迹数据作为种子数据,人工添加数据形成实验分析所用的数据,实验结果如图 5 所示.

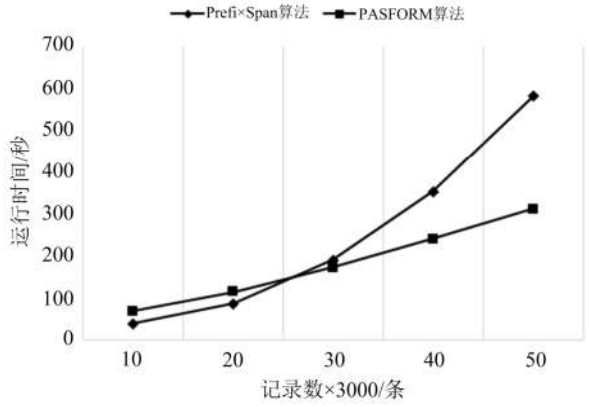


图 5　时间性能比较

Fig.5　Time performance comparison

由图 5 可知,随着数据量的增加,PrefixSpan 算法的时间性能曲线呈指数上升趋势,而 PASFORM 算法呈线性上升趋.当数据量小时,PASFORM 算法的通信开销占据了大部分时间,运行时间比 PrefixSpan 算法多;当数据量增大时,PASFORM 算法的通信开销所占据的比例逐渐缩小,数据并行处理的优势显现出来,运行时间比 PrefixSpan 算法少.

4.3.2　加速比

实验分析了不同节点个数对 PASFORM 算法的加速比的影响.实验采用的数据集仍是上述的合成数据集,节点个数依次为 2、4、6、8、10,实验结果如图 6 所示.

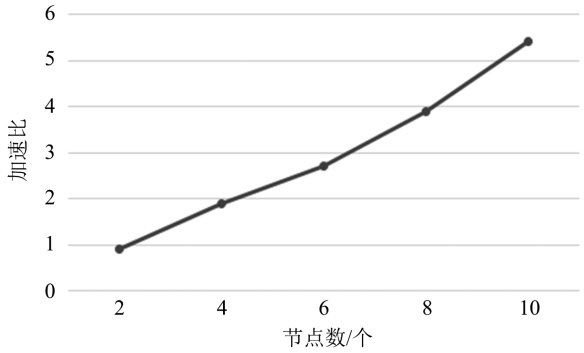


图 6　加速比比较

Fig.6　Comparison of acceleration ratio

由图 6 可知,随着节点个数增加,加速比呈线性上升趋势,但加速比明显比节点个数小,这主要是因为节点之间的通信负担在一定程度上影响了算法的效率.

4.3.3　剪枝效果

实验对 PASFORM 算法在采用剪枝策略和不采用剪枝策略的情况下,算法运行时间和挖掘出的频繁移动模式个数作了比较.实验结果如图 7 所示,图 7(a)显示的是算法运行时间的比较,图 7(b)显示的是频繁移动模式个数的比较.

图 7(a)使用的是以 81 位用户在 2004 年 11 月的移动轨迹数据作为种子数据形成的合成数据.由图可知,剪枝后 PASFORM 算法的运行时间明显减少,且随着数据量的增加,运行时间的差距逐渐增大.

图 7(b)使用的数据分别是 81 位用户在 2004 年 9～12 月的移动轨迹数据.由图可知,剪枝后 PASFORM 算法挖掘出的频繁移动模式个数明显减少,对不同月份的数据,剪枝效果大致相似.

综上所述,使用前缀树存储频繁移动模式,确实缩小了存储空间,减少了算法运行时间.

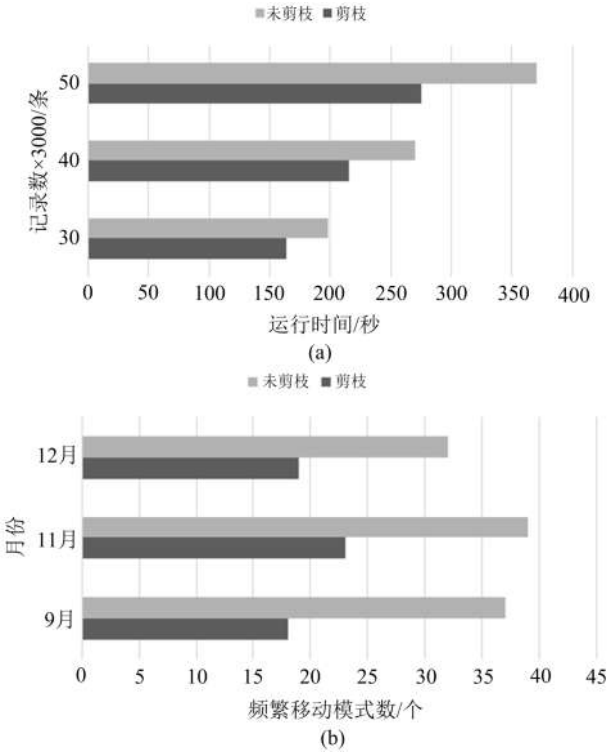


图7 剪枝效果分析

Fig.7 Analysis of pruning effect

5 结论

本文针对现有移动模式挖掘算法存在的缺点和不足,提出了一种基于PrefixSpan算法的用户频繁移动模式并行挖掘算法,该算法同时考虑时间和空间因素,能有效挖掘出用户的频繁移动模式.

参考文献(References)

[1] PENG W C, CHEN M S. Developing data allocation schemes by incremental mining of user moving patterns in a mobile computing system[J]. IEEE Transactions on Knowledge & Data Engineering, 2003, 15(1):70-85.

[2] 陈勐, 刘洋, 王月,等. 基于时序特征的移动模式挖掘[J]. 中国科学信息科学, 2016, 46(9): 1288-1297.

[3] 李雄, 马修军, 王晨星,等. 城市居民时空行为序列模式挖掘方法[J]. 地理与地理信息科学, 2009, 25(2): 10-14.

[4] LI Z H, DING B L, HAN J W, et al. Mining periodic behaviors for moving objects[C]// Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Washington: ACM Press, 2010: 1099-1108.

[5] YAVAŞ G, KATSAROS D, ULUSOY Ö, et al. A data mining approach for location prediction in mobile environments [J]. Data & Knowledge Engineering, 2005, 54(2): 121-146.

[6] LI Z H, HAN J W, JI M, et al. MoveMine: Mining moving object data for discovery of animal movement patterns[J]. ACM Transactions on Intelligent Systems & Technology, 2011, 2(4): No. 37(1-32).

[7] HUNG C C, PENG W C, LEE W C. Clustering and aggregating clues of trajectories for mining trajectory patterns and routes[J]. The VLDB Journal, 2015, 24(2):169-192.

[8] LEE J G, HAN J W, LI X L. A unifying framework of mining trajectory patterns of various temporal tightness [J]. Knowledge & Data Engineering IEEE Transactions on, 2015, 27(6):1478-1490.

[9] 王亮, 汪梅, 郭鑫颖,等. 面向移动时空轨迹数据的频繁闭合模式挖掘[J]. 西安科技大学学报, 2016, 36(4): 573-576.

[10] HUANG Q Y, LI Z L, LI J, et al. Mining frequent trajectory patterns from online footprints [C]// Proceedings of the ACM SIGSPATIAL International Workshop on Geostreaming. Burlinggame, USA: ACM Press, 2016: No. 10(1-7).

[11] LEE J W, PAEK O H, RYU K H. Temporal moving pattern mining for location-based service[J]. Journal of Systems & Software, 2004, 73(3):481-490.

[12] Qiu M, Pi D. Mining Frequent Trajectory Patterns in Road Network Based on Similar Trajectory [M]// Intelligent Data Engineering and Automated Learning: IDEAL 2016, Springer International Publishing, 2016.

[13] 刘素杰. 时间标识的移动对象频繁模式发现[D]. 徐州:中国矿业大学, 2014.

[14] AGRAWAL R, SRIKANT R. Mining sequential patterns[C]// Proceedings of the 7th International Conference on Data Engineering. Taipei, China: IEEE Press, 1995:3-14.

[15] 王虎, 丁世飞. 序列模式挖掘研究与发展[J]. 计算机科学, 2009, 36(12): 14-17.

[16] RAO V C S, SAMMULAL P. Survey on sequential pattern mining algorithms[J]. International Journal of Computer Applications, 2013, 76(12): 24-31.

[17] SRIKANT R, AGRAWAL R. Mining sequential patterns: Generalizations and performance improvements [C]// Proceedings of the 5th International Conference on Extending Database Technology. London: Springer, 1996: 3-17.

[18] HAN J W, PEI J, MORTAZAVI-ASL B, et al. FreeSpan: Frequent pattern-projected sequential pattern mining[C]// Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Boston: ACM Press, 2000: 355-359.

[19] PEI J, HAN J W, MORTAZAVI-ASL B, et al. PrefixSpan: Mining sequential patterns efficiently by prefix-projected pattern growth[C]// Proceedings of the 17th International Conference on Data Engineering. Heidelberg, Germany: IEEE Press, 2001: 215-224.

[20] LIU J, YAN S, REN J. The design of frequent sequence tree in incremental mining of sequential patterns[C]// Proceedings of the 2nd International Conference on Software Engineering and Service Science. Beijing: IEEE Press, 2011: 679-682.

[21] Reality Commoms [EB/OL]. [2017-05-06] http://realitycommons.media.mit.edu/realitymining.html.

[22] System Data [EB/OL]. [2017-05-06] https://www.citibikenyc.com/system-data.