

VR-tree 索引结构的轨迹 Stop 区域快速发现

李 雯, 夏士雄, 张 磊

(中国矿业大学 计算机学院, 江苏 徐州 221000)

摘 要: 移动对象轨迹 stop 区域发现是轨迹数据分析的一个重要方面. 提出基于 VR-tree 的轨迹 stop 区域快速发现方法. 首先选择一条轨迹使用 DBSCAN 算法得到 stop 区域, 把它们作为其他轨迹的 stop 区域候选, 构建 VR-tree; 然后, 对于剩余每条轨迹, 先与 VR-tree 比较, 找到与其中的 stops 相交的区域, 标记轨迹包含在这些区域中的点, 再对轨迹中其他点使用 DBSCAN 算法得到新的 stops, 并将它们作为其他轨迹的候选插入到 VR-tree 中. 每条轨迹的 stops 由与 VR-tree 比较获得的 stops 和新 stops 区域按照区域覆盖率合并得到. 实验使用真实数据集, 证明本方法与基于密度的轨迹 stop 发现方法相比, 不仅能很好发现轨迹 stops, 而且效率提高近 60%.

关键词: 轨迹数据分析; VR-tree; stop 区域

中图分类号: TP312

文献标识码: A

文章编号: 1000-7180(2011)08-0059-04

Using VR-tree Index to Fast Find Trajectory Stop Regions

LI Wen, XIA Shi-xiong, ZHANG Lei

(School of Computer Science & Technology, China University of Mining & Technology, Xuzhou 221116, China)

Abstract: Discovering stop regions is an important topic of trajectory data analysis. A novel method using VR-tree index structure is proposed for fast extracting trajectory stops. Firstly, one trajectory is selected and whose stops are found using DBSCAN algorithm, then VR-tree is built according to these stops which are considered as candidate stops of other trajectories. Then, for rest trajectory, compare with the stops storied in VR-tree to find intersecting parts, mark these points which are contained in these regions. Other points of the trajectory are used to find new stops through DBSCAN algorithm, which will be inserted into VR-tree as the candidate of other trajectories. Stops of trajectory are obtained by merging the stops which are composed of two parts, one comes from the results of comparing with VR-tree, and another is the new stops, according to the coverage with each other. Experiments using real data show that comparing with the density based methods, this method not only can find stops very well, but increase efficient nearly 60%.

Key words: trajectory data analysis; VR-tree; stops regions

1 引言

大量移动对象轨迹数据被收集并迫切需要有效分析. 发现轨迹 stop 区域是轨迹数据挖掘的一个重要方面.

聚类方法可用于发掘轨迹 stop 区域. Andrey Tietbohl 等提出 CB-SMoT^[1] 算法发现轨迹数据集中每条轨迹的 stop 区域. 郑宇等提出先从每条轨迹

提取停留点, 利用密度聚类算法对单条轨迹或多条轨迹的停留点集合进行层次聚类^[2]. Antomn Guttman 提出 R-tree 空间索引结构^[3], 可按照数据的空间位置建立动态索引, 快速检索多维空间数据. 何凯涛提出了 ETPR 树索引结构^[4], 综合考虑移动对象在速度域和空间域中的分布, 将移动对象映射到不同的速度桶, 再利用 TPR 树^[5]进行索引.

轨迹反映移动对象的路径, 由一系列采样点组

收稿日期: 2011-05-18; 修回日期: 2011-06-22

基金项目: 国家自然科学基金项目(50674086); 高等学校博士学科点专项科研基金项目(20060290508); 江苏省博士后基金(0802023C)

成,点可以表示为 $(\langle x, y \rangle, t, v)$, x, y, v 分别是对象在 t 时刻的位置坐标和速度.

轨迹 stop 区域是空间区域中高密度轨迹点的集合,可表示为 $\text{stop} = \{tid, p_{i0}, \dots, p_{i1}, \dots, p_{in}\}$, $li \in \{0, n\}$. stop 区域表征移动对象在该区域内活动比较密集.

由于轨迹数目较多,现有方法在发现轨迹 stops 的时间代价太高. 由于轨迹集中的轨迹在空间位置上有很大的相似性,多条轨迹可能经过相同的区域,把这些信息存储起来并加以利用可能减少运算时间,因此,本文提出 VR-tree(添加速度约束的 R-tree)索引结构的轨迹 Stop 区域快速发现方法.

2 基于 VR-tree 索引的轨迹 Stop 区域发现方法

本文用 stop 区域的最小外接矩形代表 stop 中轨迹点的运动区域,树形结构存储矩形边界,并在 R-tree 的基础上添加速度限定信息. VR-tree 中叶结点表示为 $E = \langle \text{MBR}, \text{ID}, \text{VBR} \rangle$, 非叶子结点表示为 $E = \langle \text{MBR}, \text{child-pointer} \rangle$. 叶结点的 MBR 为包含数据点的最小外接矩形;非叶子结点的 MBR 为包含所有子结点的最小矩形;叶子结点的 VBR 为包含数据点的速度范围;ID 为 E 的编号;child-pointer 指向该结点的所有孩子结点.

2.1 发现第一条轨迹的 Stop 区域

选取集合中其中一条采用 DBSCAN 算法挖掘该轨迹 stops. 选取的第一条轨迹点数越多,产生越多 stops,对后续影响也越大. 每个 stop_i 区域都被当作叶子结点 $E_i = \langle \text{MBR}_i, \text{ID}, \text{VBR}_i \rangle$, MBR_i 是 stop_i 内轨迹点的最小外接矩形, VBR_i 为 stop_i 中所有轨迹点的速度范围. 使用 $\text{ES} = \{E_1, \dots, E_n\}$ 构建 VR-tree.

2.2 发现其余轨迹的 Stop 区域

对其余的轨迹:首先在 VR-tree 中搜索该轨迹的点,若发现一系列连续满足条件的点,则标记这些点并从轨迹中筛选出;使用 DBSCAN 聚类未被标记的轨迹点,找到其余候选 stops,并将其作为叶子结点插入 VR-tree;将所有(VR-Tree 中的和新发现的)候选 stops 按覆盖率合并,获得最终 stops.

(1) 将轨迹与 VR-tree 进行比较

轨迹 $T = \{p_0 = (\langle x_0, y_0 \rangle, t_0, v_0), \dots, p_n = (\langle x_n, y_n \rangle, t_n, v_n)\}$ 从第一个点开始,依次在 VR-tree 中搜索:对于点集合 $s = \{p_i, \dots, p_{i+n}\}$, 如果 $\forall p_s$ 存在于 VR-tree 某结点 E 中,且 $p_s.v \in [E.$

VBR, Vmin, E. VBR, Vmax], 而且 s 中点数 $n+1 \geq \text{MinNum}$, 则称 s 为候选 stop-簇. 遍历整条轨迹,找到所有满足候选 stop-簇的 s 集合 $\text{CC} = \{s_1, \dots, s_m\}$.

s 间可能存在重叠,需要将重叠面积达到一定比例的 s 合并. 任意两项 s_i 和 s_j , 假设它们外接矩形的重叠为 MBR_{ij} , 且 s_i, s_j 及 MBR_{ij} 的面积分别为 $\text{Area}_i, \text{Area}_j$ 和 Area_{ij} , 若 $\text{Area}_{ij}/\text{Area}_i$ 或 $\text{Area}_{ij}/\text{Area}_j$ 大于 Cover_rate (可合并的最小重叠率), 则将 s_i 和 s_j 合并. 对于 CC 中所有 s_i 两两合并, 得到新集合 $\text{FinCC} = \{s'_1, \dots, s'_n\}$, 该集合中的项为候选 stop.

(2) 剩余轨迹段处理

标记并筛选轨迹 T 中已包含在 FinCC 集中的轨迹点, 筛选后的轨迹为 T' . 采用 DBSCAN 算法寻找 T' 中的候选 stops $= \{s'_{s+1}, \dots, s'_n\}$. 剩余轨迹段发现候选 stops 与已记录于 VR-tree 中 stops 可能存在少量重叠. 对该步得到的候选 stops: 计算 $s_i (i \in \{s+1, \dots, n\})$ 的外接矩形 MBR_i 及速度范围 VBR_i ; 得到集合 ES, 将 ES 当做叶子结点插入 VR-tree.

(3) 合并候选 stops

候选 stops 间可能存在重叠, 需要作合并处理, 过程和第一步中候选 stop-簇集合的合并相似.

按照上述方法挖掘剩余所有轨迹的 stops.

2.3 算法描述与分析

输入: $\text{Tra_Set} \langle T \rangle$, eps , MinPts , MinNum , Cover_rate

输出: 所有候选 stops

(1) 选取第一条轨迹, 使用 DBSCAN 算法计算 stops, 并将其作为输入构建 VR-tree;

(2) 对于其他各条轨迹, 将轨迹与 VR-tree 比较, 得到 CC 集合, 对其进行合并, 得到部分候选 stops;

(3) 在原轨迹中剔除已被发现的 stops 中的点, 使用 DBSCAN 算法找到剩余候选 stops;

(4) 将所有候选 stops 合并, 得到最终 stops.

假设 n 为轨迹 T 的点数, 从第二条开始, 与 VR-tree 比较时, 每个点至多进行树的高度次比较, 处理轨迹越多, VR-tree 越庞大, 筛选后轨迹剩余点数越少, 算法效率越高. 当各条轨迹速度相近, stops 区域较为集中时, 算法能获得较好效率, 当各轨迹 stops 区域相距较远时, 时间效率会受到影响.

3 实验及分析

为验证本文提出的基于 VR-tree 索引结构发现

stops 的方法,实现了轨迹 stop 发现原型系统 VR_Stop. 实验环境为: Windows XP, CORE 2 DUO 2.93 GHz CPU, 2 GB 内存. 本文选取真实数据集^[6]作为实验数据: 抽取 1995 年鹿的运动轨迹, 完整数据集包含 42 384 个采样点组成的 35 条轨迹.

3.1 有效性分析

由图 1、图 2 中可以看出, 针对某条轨迹, 两种方法发现 stops 的效果基本一致. 但使用 VR-tree 索引得到的 stops 之间界限不是特别清楚, 如前所述, 最终 stops 是由两部分组成, 两部分之间可能存在一些重叠, 但是重叠面积很小, 没有被合并. 图 3、图 4 所示为表 1 中轨迹号为 16 的轨迹, 从轨迹运行时间看, 使用 VR-tree 索引发现 stops 在时间上有很大优势.

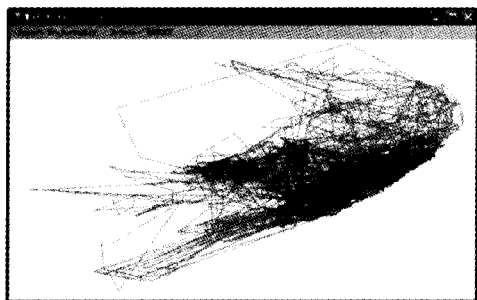


图 1 单条轨迹通过 VR-tree 索引发现 stops

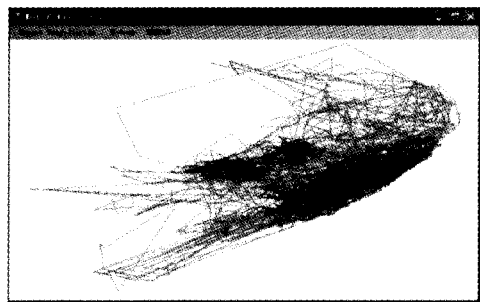


图 2 单条轨迹通过 DBSCAN 发现 stops

图 3、图 4 所示为轨迹号 16—24 的部分轨迹使用 VR-tree 索引和 DBSCAN 算法发现 stops 的效果图, 图中红圈标记为各轨迹 stop 区域. 9 条轨迹采用两种方法都获得了 17 个 stop 区域, 由于选取轨迹位于轨迹集中间位置, VR-tree 比较健壮, 通过与 VR-tree 比较得到 13 个 stop 区域, 个数较多. 轨迹集合产生的 stops 基本一致, 达到了比较满意的效果.

由表 1 可看出, 当轨迹 ID 比较小时, 使用 VR-tree 索引发现 stops 主要是筛选后轨迹通过 DBSCAN 算法得到, 当轨迹 ID 较大时, stops 主要是轨迹与 VR-tree 比较产生的. 这是因为 ID 较小时,

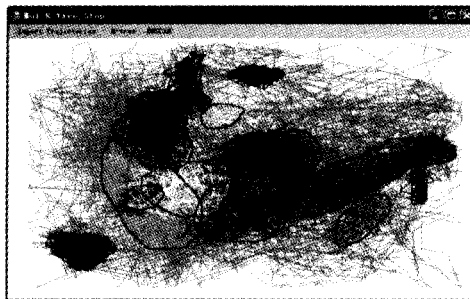


图 3 VR-tree 发现的 stops

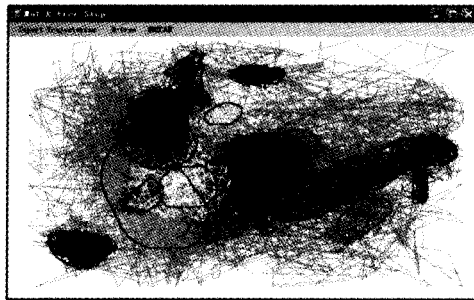


图 4 DBSCAN 发现的 stops

VR-tree 中结点个数和区域覆盖较少, 轨迹 stops 位于 VR-tree 中概率较小. 但处理较多轨迹后, VR-tree 比较健壮, stop 区域位于 VR-tree 中可能性变大.

表 1 VR-tree 索引和 DBSCAN 算法的 stops 个数与运行效率

Tra Id	count (VR-tree Stops)	count (stops of T')	count (DBSCAN Stops)	Runtime (VR-tree)	Runtime (DBSCAN)
1	0	2	2	0.44	0.44
3	0	1	1	3.6	3.11
5	1	3	3	17.1	17.9
10	1	1	2	1.6	2.2
11	0	1	1	7.2	7.3
16	2	1	3	0.27	8.2
19	1	0	1	0.84	1.78
21	1	0	1	0.13	1.53
25	1	0	1	0.02	0.2
27	1	0	1	0.1	0.3
29	1	0	1	0.02	0.95
31	1	1	1	0.14	0.29
33	1	0	1	0.16	0.41
35	1	0	1	0.06	0.26

3.2 性能分析

表 1 最后两列表明, 开始时, 轨迹运行时间比较相近, 但是当与 VR-tree 比较获得 stops 点数较多时, 使用 VR-tree 索引的运行效率提高很多.

由图 5 可看出轨迹数较少时, 两种方法所用时

间相差无几,此时 stops 主要通过 DBSCAN 算法得到;随着轨迹数增加,使用 DBSCAN 算法的时间增长很快,而使用 VR-tree 索引的运行时间增长相对较慢.由图 6 可以看出,使用 VR-tree 索引和 DBSCAN 算法发现的 stops 数量基本相等.

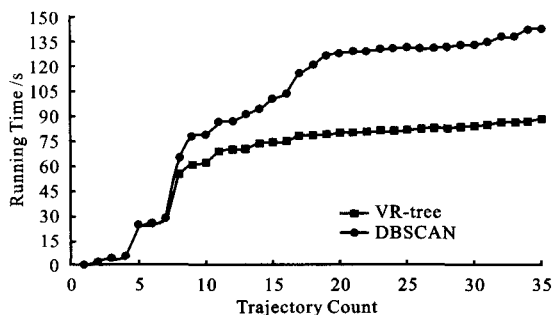


图 5 轨迹数量与运行时间的关系

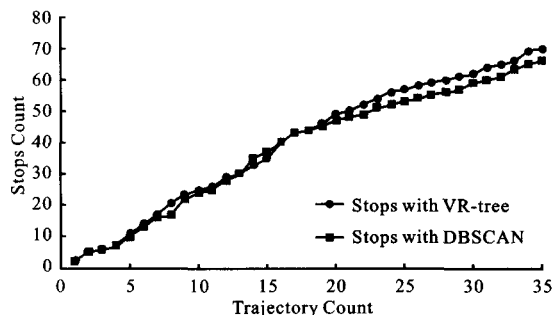


图 6 轨迹数量与生成 stops 个数关系

4 结束语

本文提出基于 VR-tree 索引结构的轨迹 stop 区域发现方法.首先选取一条轨迹使用 DBSCAN 算法发现 stops 并构建 VR-tree;然后对于其余轨迹,先在 VR-tree 中搜索,将与已处理轨迹 stop 区域相交且速度域相近的轨迹段从原轨迹中筛选;使用 DBSCAN 算法得到所有候选 stops;最后按照覆盖率大小合并候选 stops 得到最终 stops.使用麋鹿轨

迹数据进行实验,对本文提出的方法和使用 DBSCAN 算法挖掘轨迹 Stops 区域在运行效果和运行时间方面比较,可知本文方法不但挖掘 stops 效果比较准确,而且效率提高近 60%.

下一步将在此基础上,发现多轨迹的 stop 区域和轨迹间的关联关系,为用户提供个性化的服务.

参考文献:

- [1] Andrey Tietbohl, Vania Bogorny, Bart Kuijpers, et al. A clustering based approach for discovering interesting places in trajectories [C]//SAC 2008. Brazil: Ceara, 2008:863-868.
- [2] Yu Zheng, Xing Xie. Enabling smart location-based services by mining GPS traces [C]//CCCF. Canada, 2010.
- [3] Antonin Guttman. R-trees a dynamic index structure for spatial searching[C]// Proc ACM SIGMOD IntConf on Management of Data. Boston: 1984:47-57.
- [4] 何凯涛,唐宇,廖巍,等. 动态环境下移动对象索引技术研究[J]. 电子与信息学报,2008,30(10):2507-2511.
- [5] Saltenis S, Jensen C S, Leutenegger S, et al. Indexing the positions of continuously moving objects[C]// Proc. of the SIGMOD. USA, New York, 2000: 331-342.
- [6] Lee Jaegil, Han Jiawei, Whang KyuYoung. Trajectory clustering: a partition-and-group framework[C]// ACM SIGMOD 2007. China, Beijing, 2007:593-604.

作者简介:

李雯女,(1988-),硕士研究生.研究方向为知识发现与数据挖掘.

夏士雄男,(1961-),教授,博士生导师.研究方向为智能信息处理.

张磊男,(1977-),副教授.研究方向为语义与本体、数据挖掘.

(上接第 58 页)

- [2] Braganholo V, Heuser C, Vittori C. Updating relational databases though xml views[C]// Proceedings of Third International Conference on Information Integration and Web-based Applications & Services - IIWAS 2001. Austria, Linz,2001:85-94.
- [3] Sun Hongwei, Zhang Shusheng, Zhou Jingtao, et al, Constraints-preserving mapping algorithm from XML-schema to relational schema[J]. Lecture Note in Computer Science, 2002(2480):193-207.
- [4] Manolescu Ioana, Florescu Daniela, Kossmann Donald. Pushing XML queries inside relational databases[R].

Washington, Institute National of Research of Information and Automation,2001.

- [5] 赵圣猛,赵雷. 一种优化的 XML 文档模型映射方案[J]. 微电子学与计算机,2009,26(10):174-177.

- [6] 孙宏伟,张树生,周竞涛,等. 基于模型驱动的 XML 与数据库双向映像技术[J]. 计算机工程与应用,2002,38(4):25-27.

作者简介:

赵媛女,(1983-),硕士,工程师.研究方向为 Web Service、分布式计算.