

基于并行模式挖掘和路径匹配的用户位置预测

许贤泽, 谭盛煌, 刘 静, 施 元

(武汉大学 电子信息学院, 湖北 武汉 430072)

摘 要: 为了提高移动用户位置预测的精度,提出了基于并行模式挖掘和路径匹配的移动用户位置预测方法,对传统的FP-GROWTH算法作了并行化处理,优化了节点负载分配方法,在Spark平台下挖掘用户移动频繁模式.改进了基于索引的路径相似度算法,提出基于路径最短距离的相斥度算法,提高了对轨迹数据缺失的适用性.在真实的用户轨迹数据集上实验表明,提出的基于轨迹相斥度预测方法相比马尔可夫模型和卡尔曼滤波模型拥有更高的预测精度,预测精确度平均提升7%左右.

关 键 词: 位置预测;Spark;FP-GROWTH;模式挖掘;轨迹相斥度

中图分类号: TM 715 **文献标志码:** A **文章编号:** 1005-3026(2020)06-0767-05

User Location Prediction Based on Parallel Pattern Mining and Path Matching

XU Xian-ze, TAN Sheng-huang, LIU Jing, SHI Yuan

(Electronic Information School, Wuhan University, Wuhan 430072, China. Corresponding author: TAN Sheng-huang, E-mail: 512436821@qq.com)

Abstract: In order to improve the accuracy of location prediction for mobile users, a method of location prediction for mobile users was proposed based on parallel pattern mining and path matching. Based on the traditional FP-GROWTH algorithm, the method of node load allocation was optimized, and frequent patterns of mobile users were mined on Spark platform. The index-based path similarity algorithm was improved, and the repulsion algorithm based on the shortest path distance was proposed to improve the applicability of missing trajectory data. Experiments on real user trajectory data sets show that the proposed model based on track dissimilarity prediction method has higher prediction accuracy than that of Markov model and Kalman filter model, which is improved by about 7% on average.

Key words: location prediction; Spark; FP-GROWTH; pattern mining; track dissimilarity

随着移动互联网的快速发展,智能手机已成为大众的必备工具.数据显示,2017年1~7月,我国智能手机的出货量达2.66亿部,智能手机用户规模也已达6.55亿人,这些移动设备每天产生海量的位置数据,通过这些位置数据对移动用户的轨迹预测已成为近年来的研究热点之一.准确的轨迹预测在导航^[1]、疾病预防^[2]和城市规划^[3-4]等诸多领域有着广泛应用.

当前轨迹预测方法主要分为以下三种:

1) 基于时间序列模型的位置预测方法^[5].这类方法是通过将原有时间序列构建线性或非线性

数学模型,该方法假设移动对象的运动轨迹较为平稳,移动速度、方向不会产生较大变化.在真实的城市交通中路况较为复杂,移动用户并不能产生时间序列模型所要求的平稳轨迹.

2) 基于机器学习的位置预测方法.该方法在目前位置预测中应用最为广泛,主要有Markov模型^[6]、卡尔曼滤波模型^[7]、贝叶斯模型^[8],其中以Markov模型应用最多.Markov模型主要是根据历史数据构建状态转移矩阵,当出现新用户或出现数据缺失时,位置预测准确性较低.

3) 基于频繁模式挖掘FP-GROWTH算法

挖掘运动对象运动模式的方法^[9]. 该方法从历史轨迹数据集中挖掘频繁轨迹模式, 然后与当前路径进行匹配实现位置预测. 由于传统的 FP - GROWTH 算法复杂度较高, 无法处理大量的历史轨迹数据集. 本文采用了一种并行 FP - GROWTH 算法与相斥度计算结合展开移动用户轨迹预测的方法, 将传统的 FP - GROWTH 算法分布到 Spark 并行计算框架上, 从而解决了大量数据的挖掘, 同时改进了相似度计算方法, 对缺失轨迹有更好的适用性.

1 问题描述

现有一移动用户轨迹数据集和某移动用户的当前轨迹, 预测该用户的短期轨迹. 设数据集 $X = \{X_1, X_2, \dots, X_w\}$ 其中: $X_i = (u_i, t_i, p_i)$, $i = 1, 2, \dots, w$. u_i 是用户 ID, t_i 是用户在该位置的时间, p_i 为用户所在位置的经纬度. 假设有一个移动用户的位置数据集, 该数据集是一个时间序列, 包含了用户在一定时间段内的位置数据. 首先对位置数据进行数据预处理, 然后提取出用户的轨迹集 $T_{u_i} = \{(t_i, l_i), i = 1, 2, \dots, x\}$, 时间序列 $t_{u_i} = t_1, t_2, \dots, t_x$, 位置序列 $L_{u_i} = l_1, l_2, \dots, l_x$, 其中 u_i 表示任意移动用户. 移动用户的位置预测问题可以描述如下: 设 u_i 是某一研究用户, 已知该用户的移动轨迹集为 T , 预测出该用户下一步地点 l_{x+1} .

2 基于 Spark 的 FP - GROWTH 算法

使用传统的 FP - GROWTH 算法需要将所有的数据加载到单机内存, 随着数据量的增大, FP - GROWTH 算法构建的 FP - Tree 的广度、深度都会增大^[10], 同时递归次数增加, 导致挖掘效率变低. 因此需要对原始的 FP - GROWTH 算法进行改进, 利用分布式集群实现频繁项挖掘^[11].

2.1 基本概念

设 $A = (a_1, a_2, \dots, a_n)$ 为一个集合, 集合 A 中的元素称为项, 集合 A 称为项集 A 或事务 A . 在本文中, 集合 A 表示一条轨迹, a_i 表示轨迹 A 上的采样点.

- 1) 支持度: 事务 A 在事务集合中出现的次数与事务集合的大小的比值.
- 2) k -项集: 长度为 k 的项集.
- 3) 1-项集的条件模式基: 选择其中一个 1-项集 a_i , 对 FP - Tree 中每个 a_i 节点向上回溯至根节点所得到的路径的集合, 条件模式基可看

成一个新的事务集合.

2.2 基于 Spark 的 FP - GROWTH 算法

首先对数据集进行划分, 将分割数据集分布到集群的各个节点上分别进行频繁项集挖掘, 最终将挖掘结果聚合到主节点上.

算法的实现主要包括四个部分:

- 1) 将数据集部署到分布式文件系统 (Hadoop distributed file system, HDFS) 上, 扫描 HDFS, 并行计算各 1-项集的支持度, 将 1-项集支持度计算结果聚合到主节点上, 并根据支持度排序^[11].
- 2) 修剪上述 1-项集集合, 删除支持度小于所设阈值的 1-项集, 记该 1-项集集合为 H_list .
- 3) 依据 H_list 对数据集进行划分, 由于条件模式基大小并不一致, 为了在分布式集群上提高计算速度, 需要进行负载均衡处理^[11]. 对于 H_list 中的一个 1-项集 $H_list[i]$, i 越小, $H_list[i]$ 所对应的条件模式基所含事务更少, 事务长度更短, 因此使用 FP - GROWTH 算法挖掘频繁项时速度更高. 对条件模式基的频繁项集挖掘过程是一个对树的递归操作, 因此本文采用 $H_list[i]$ 的序号与支持度的乘积估算条件模式基的负载. 以负载为权重将划分后的数据集分配到各个节点上.
- 4) 在各个节点上进行频繁模式挖掘^[11], 最后将结果聚合到主节点上.

3 相斥度计算

本文针对基于同位序轨迹点欧式距离的轨迹相似度算法的不足^[12], 提出一种基于计算当前点到历史轨迹距离加权求和的相斥度 (即两条轨迹的相离程度) 计算方法.

3.1 点到轨迹的距离

定义 1 点到轨迹的距离: 已知某条轨迹数据序列 $l = \{p_1, p_2, \dots, p_m\}$ 和轨迹点 q_i , 轨迹点 q_i 到轨迹 l 的距离定义如下:

$$D(q_i, l) = \min(\text{distance}(q_i, p_j p_{j+1})).$$

其中 $\text{distance}(q_i, p_j p_{j+1})$ 表示轨迹点 q_i 到线段 $p_j p_{j+1}$ 的距离, 且 $1 \leq j \leq m-1$.

设存在轨迹序列 $l_1 = \{p_1, p_2, \dots, p_m\}$ 和 $l_2 = \{q_1, q_2, \dots, q_n\}$, 计算轨迹 l_2 的上各点到轨迹 l_1 的最小距离. 计算 l_2 上各轨迹点到轨迹序列 l_1 最短距离的过程中, 精确求解需要计算 q_i 到 l_2 各轨迹段的距离. 考虑到运动轨迹的渐变特性, 在计算上进行一定改进.

通过数据预处理后, 若轨迹 l_1 与 l_2 相似, l_2

上起始点 q_1 到 l_1 各轨迹段距离先减小后增大,取其中极小值为 q_1 点到轨迹 l_1 最小值.若求得 l_2 上一轨迹点 q_i 到 l_1 的距离为 q_i 到轨迹段 (p_j, p_{j+1}) 的距离,计算 l_2 中 q_i 的下一个轨迹点 q_{i+1} 到轨迹 l_1 的距离,计算过程如下.

尝试计算 q_{i+1} 到 $(p_k, p_{k+1}), j \leq k < n$ 的距离,记录计算过程中的最小距离(点到前缀轨迹段和后缀轨迹段距离均大于点到当前轨迹段的距离),当最后一次运算结果大于距离最小值的 k 倍时终止运算,取其中计算过程最小值为轨迹点 q_{i+1} 到轨迹 l_1 的距离.在本文实验中, k 取值为 2 时,保证计算效率的同时拥有较高的准确率.

3.2 轨迹相斥度

在用于预测的相似路径中,越靠近当前轨迹末端的点与预测点位拥有更高的相关性,对最终的预测结果影响越大^[13],因此对这些轨迹点赋予更高的权重,对相斥度作如下定义.

定义 2 轨迹相斥度:已知某条轨迹数据序列 $l_1 = \{p_1, p_2, \dots, p_m\}$ 和轨迹数据序列 $l_2 = \{q_1, q_2, \dots, q_n\}$, l_1, l_2 轨迹相斥度定义为

$$\text{dissimilarity}(l_1, l_2) = \frac{\sum_{i=1}^n w_i D(q_i, l_2)}{\sum_{i=1}^n w_i}.$$

其中 w_i 表示轨迹点 q_i 的权重.在相斥度计算中,设定一相斥度阈值 threshold,当计算过程中相斥度大于 threshold,认为 l_1 和 l_2 相斥,终止计算.

采用与被预测路径相斥度最低的频繁轨迹与被预测路径匹配,完成用户位置预测.

4 实验结果及算法性能分析

4.1 实验数据及其预处理

本文选用了由微软亚洲研究院发布的 GeoLife GPS Trajectories 数据集,该数据集由智能手机和其他包含有 GPS 的设备记录而成,从 2007 年 4 月到 2012 年 8 月期间 182 个用户的轨迹数据,包括经度、纬度和海拔.该数据集总共包含 17 621 条轨迹数据,记录总时长超过 48 000 h,总距离超过 120 万 km.

由于采样间隔比较密集,测量误差对相邻两点之间的相对位置影响很大,因此本文根据原数据集的特点,对原始数据集进行采样,采样频率为 10.结合百度地图对数据进行基于 POI 点的停留点检测,用折中方法分割轨迹^[14],得到便于模式挖掘和相斥度计算的短序列.

4.2 实验结果与分析

4.2.1 Spark 和 MapReduce 运行效率对比

为验证基于 Spark 的 FP - GROWTH 算法在大量数据下的可用性,本实验选取了不同的数据集,分别在 5 个节点的 MapReduce 以及 Spark 运算框架上以 FP - GROWTH 算法进行频繁项挖掘,比较两种运算框架的运算效率.

FP - GROWTH 算法计算内存开销较大,对于大型数据集,单机受内存限制无法运行.实验选取不同规模的数据集,通过合理的数据集分割,可以在不影响模式挖掘效果的前提下进行分布式运算^[11].分别在 Spark 和 MapReduce 并行计算框架上进行频繁模式挖掘,实验结果如图 1 所示.

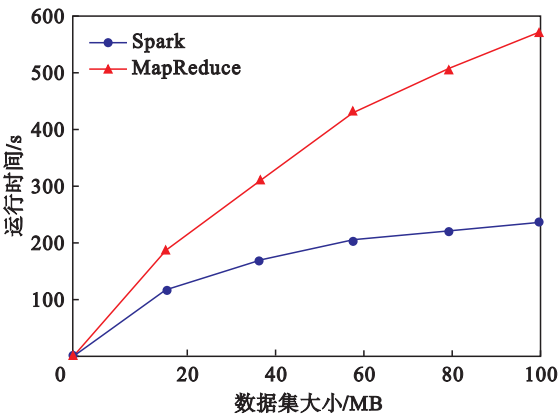


图 1 Spark 和 MapReduce 运算框架下运行时间对比
Fig. 1 Running time comparison between Spark and MapReduce

由图 1 可知,Spark 运算框架相比 MapReduce 在不同的数据集大小下,运行时间更短,MapReduce 框架比 Spark 运行时间增长更快.这是因为 MapReduce 从硬盘读写数据,计算时在内存中进行存储和运算. Spark 直接基于内存处理,避免了与硬盘的 IO 传输.因此 Spark 运算框架上的 FP - GROWTH 算法拥有更高的运算效率.

4.2.2 轨迹预测算法性能对比

为验证模型的预测准确性,对马尔可夫模型、卡尔曼滤波模型及本文提出的基于模式挖掘与路径匹配的方法进行对比.将数据集随机分成 20 份,其中一份作为测试数据集用于轨迹预测,另外 19 份用于训练模型,每个实验采用交叉验证的方式进行 20 次,取平均结果作为最终预测值.

定义 3 预测精度^[13]:在轨迹预测过程中,预测结果正确的概率.例如,对于 n 个需要预测位置,设定一个预测误差阈值,预测误差在阈值以内认为预测正确,有 t 个位置预测正确,则该预测结果的预测精度为 $P_a = t/n$.

本文采用预测精度作为轨迹预测准确性的评价标准. 分别采用马尔可夫模型、卡尔曼滤波模型以及本文的相斥度预测算法进行预测精度对比.

实验采用马尔可夫模型、卡尔曼滤波模型以及本文的相斥度预测算法进行不同长度轨迹的单步位置预测,即预测用户的下一个感兴趣的位置. 图 2 为 3 种预测方法对不同长度轨迹单步预测结果. 横轴表示当前轨迹长度,纵轴表示预测精度.

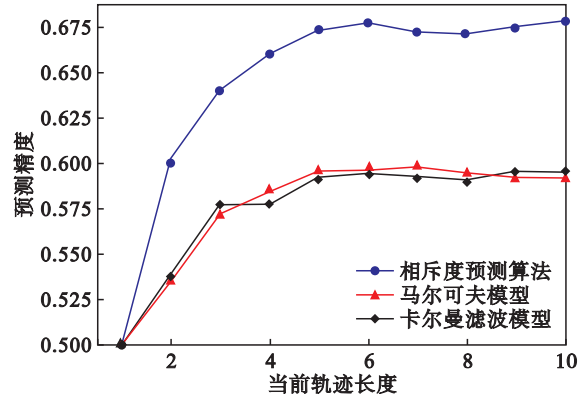


图 2 不同长度轨迹单步位置预测精度对比
Fig. 2 Accuracy comparison of single-step position prediction for different length trajectories

从图 2 可以看出,随着轨迹长度增加,三种预测方法预测精度保持上升,在轨迹长度为 5 时逐渐平稳. 保持平稳之后,基于相斥度的轨迹预测算法相比于马尔可夫模型和卡尔曼滤波模型预测精度分别提升 6. 93% 和 6. 98% .

在进行位置预测时,多步预测也是检验模型预测性能的指标之一,多步预测即连续预测多个点的位置. 本文根据不同长度轨迹的单步预测实验的结果,设置待预测轨迹长度为 5. 采用三种方法对测试集中 6 个连续位置进行预测,计算不同位置平均预测精度,如图 3 所示.

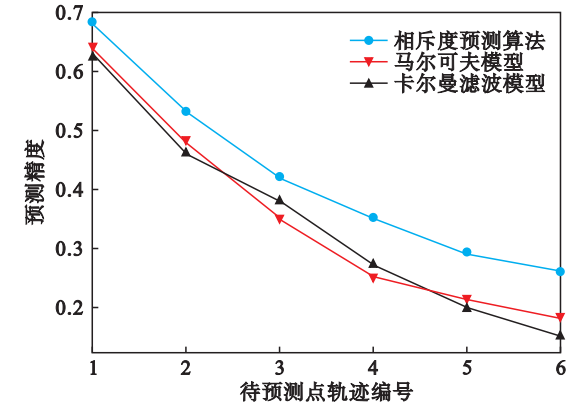


图 3 不同点位的预测精度对比
Fig. 3 Comparison of prediction accuracy at different locations

从图 3 可以看出,三种方法在步长增加时,预测精度有明显下降. 相比于马尔可夫模型和卡尔曼滤波模型,本文使用的相斥度预测算法对不同步长的预测精度在各点平均预测精度提升分别为 7. 03% 和 7. 35% .

从上面两个实验可以看出,本文采用的相斥度预测算法预测精度优于马尔可夫模型和卡尔曼滤波模型. 原因在于马尔可夫模型是概率统计模型,对于运动状态不确定性较高的对象预测精度不高. 卡尔曼滤波主要适用于线性离散系统,对轨迹点数据缺失情况适应性不佳.

5 结 论

1) 本文提出了一种基于并行模式挖掘和路径匹配的移动用户位置预测方法. 对 FP - GROWTH 算法作了并行化处理,实现了对大量用户的移动模式提取. 采用基于相斥度的轨迹预测算法进行用户位置预测,并与马尔可夫模型和卡尔曼滤波模型进行比较分析.

2) 使用分布式平台可以完成大规模的模式挖掘,且 Spark 平台相比 MapReduce 计算效率更高.

3) 相比于马尔可夫模型和卡尔曼滤波模型,相斥度预测算法在单步预测和多步预测中预测更为精准.

参考文献:

[1] Pang L,Chawla S, Liu W, et al. On detection of emerging anomalous traffic patterns using GPS data [J]. *Data & Knowledge Engineering*,2013,87(9):357 - 373.

[2] Adegboye O A,Kotze D. Disease mapping of Leishmaniasis outbreak in Afghanistan:spatial hierarchical Bayesian analysis [J]. *Asian Pacific Journal of Tropical Disease*,2012,2(4): 253 - 259.

[3] Rathore M M,Ahmad A,Paul A, et al. Urban planning and building smart cities based on the Internet of Things using big data analytics[J]. *Computer Networks*,2016,101(C):63 - 80.

[4] Batty M. Big data, smart cities and city planning [J]. *Dialogues in Human Geography*,2013,3(3):274 - 279.

[5] Zheng D, Wang S, Meng Q. Dynamic programming track-before-detect algorithm for radar target detection based on polynomial time series prediction [J]. *IET Radar, Sonar & Navigation*,2016,10(8):1327 - 1336.

[6] Qiao S, Shen D, Wang X, et al. A self-adaptive parameter selection trajectory prediction approach via hidden Markov models[J]. *IEEE Transactions on Intelligent Transportation Systems*,2015,16(1):284 - 296.