基于滑动窗口的分布式轨迹流聚类

毛嘉莉[1),2)] 陈 鹤[1)] 宋秋革[1)] 金澈清[1)] 周傲英[1)]

[1)](华东师范大学数据科学与工程学院 上海 200062)

[2)](西华师范大学计算机学院 四川 南充 637009)

摘 要 随着移动定位技术的蓬勃发展和移动定位设备的广泛应用,衍生了海量移动对象的位置信息.该类位置信息包含地理坐标、速度、方向以及时间戳等信息,被实时采集且持续增加,形成了大规模高速的分布式轨迹数据流.及时、有效地对分布式的轨迹流数据进行在线聚类分析,可以实时获取移动对象的共同移动趋势.由于轨迹数据流固有的海量、高速、偏态分布、时变进化且存在概念漂移的特性以及在线聚类的严格时空开销需求,基于静态轨迹数据的聚类方法不能直接应用于分布式轨迹流的在线聚类分析.分布式的轨迹流聚类研究面临巨大挑战,研究工作仍处于初期探索阶段.为解决上述问题,面对地理上分散采集的轨迹流数据,亟需设计高效的并行聚类分析任务及确保传输开销最小化的通信机制来满足低处理延迟的实时聚类需求.该文首先设计了分布式聚类概要数据结构以实时获取相似轨迹簇的时空特征,继而维护持续进化的分布式轨迹数据流.在此基础上,以减少通信开销提高分布式轨迹流聚类效率为目标,提出了一个在线处理分布式轨迹数据流的增量聚类算法(OCluDTS).OCluDTS方法使用基于滑动窗口模型的两层分布式框架,通过多个远程节点并行聚类局部轨迹流以及协调者节点合并局部聚类结果的方式,确保分布式轨迹流聚类获得与集中式方法相同的精度.此外,为了进一步降低 OCluDTS 算法的总执行开销,提出了仅限于聚类更新的远程节点传输聚类结果给协调者节点以及基于协调者节点相似性计算的剪枝策略等优化措施.最后,理论分析以及基于真实轨迹数据集的实验结果验证了 OCluDTS 算法处理大规模分布式轨迹流数据时的有效性和高效性.

关键词 分布式轨迹流;聚类;滑动窗口;分布式时序轨迹聚类特征;概念漂移

中图法分类号 TP311 **DOI 号** 10.11897/SP.J.1016.2018.02120

Distributed Trajectory Streams Clustering over Sliding Window Model

MAO Jia-Li[1),2)] CHEN He[1)] SONG Qiu-Ge[1)] JIN Che-Qing[1)] ZHOU Ao-Ying[1)]

[1)](*School of Data Science and Engineering, East China Normal University, Shanghai* 200062)

[2)](*Computer School, China West Normal University, Nanchong, Sichuan* 637009)

Abstract With the widespread application of modern mobile devices and the vigorous development of location acquisition technologies, numerous moving objects have been relaying their locations continuously. Such the location information involves the geographic coordinates, speed, direction, as well as timestamps, etc. They are collected in real time and the amount of data continues to increase, and hence a tremendous amount of distributed trajectory data streams are generated. This necessitates to conduct the analysis task like clustering upon the distributed trajectory data streams timely and effectively to gain insights about the common moving trends of the moving objects. Due to the massive volume, high velocity, skewness distribution, time-varying evolution property of the streaming trajectories, the existing trajectory clustering techniques on the static

收稿日期:2017-08-06;在线出版日期:2018-03-05.本课题得到国家自然科学基金(61702423,61370101,61532021,U1501252,U1401256,61402180)、四川省教育厅自然科学重点项目(17ZA0381)、西华师范大学国家培育项目(16C005)和西华师范大学英才基金(17YC158)资助.**毛嘉莉**,女,1979 年生,博士,副教授,中国计算机学会(CCF)会员,主要研究方向为基于位置的服务和大数据分析.E-mail:jlmao1231@stu.ecnu.edu.cn.**陈 鹤**,男,1992 年生,硕士研究生,主要研究方向为轨迹计算.**宋秋革**,女,1992 年生,硕士研究生,主要研究方向为数据挖掘和基于位置的服务.**金澈清**,男,1977 年生,博士,教授,博士生导师,中国计算机学会(CCF)会员,主要研究领域为时空数据处理和基于位置的服务等.**周傲英**,男,1965 年生,博士,教授,博士生导师,中国计算机学会(CCF)会员,主要研究领域为大数据处理和分析等.

trajectory data set cannot be directly applied to the distributed trajectory data streams. In addition, the key issue also comes from the strict space- and time-complexities of processing the continuously arrived trajectory data, combined with the concept drift that emerged in the streaming cases. Therefore, the study of clustering upon the distributed trajectory streams have faced the huge challenges, which makes this topic is still in its early stage of exploration. To address the above mentioned issues, it is imperative to develop the efficient parallel clustering analysis tasks, and exploit the effective communication mechanisms that minimize the transmission overhead to meet the real-time clustering requirement with low processing latency in processing the trajectory streams that collected in geographically dispersed manner. In this paper, we firstly present a novel distributed synopsis structure to extract the clustering characteristic of the trajectory cluster and further keep the track of the evolving distributed trajectory data streams. On the basis of that, with the aim of reducing the communication overhead and improving the clustering performance upon the distributed trajectory data streams, we develop an incremental algorithm for online clustering upon the distributed trajectory streams (called OCluDTS). OCluDTS algorithm leverages the sliding window model and consists of two phases. At the first phase (called as the local clustering phase), the most recent arrived sets of trajectories at each time instant in current window for all the remote site are conducted clustering in parallel to obtain the local clustering results. At the second phase (called as the local clustering results merging phase), the local clustering results of all the remote sites are transferred to the coordinator to participate in the re-clustering process to derive the final clustering results. This two-phase clustering mechanism could guarantee achieving the same high clustering precision as the centralized solution. Moreover, the pruning mechanism of similarity calculation, and the optimization strategy that testing first and transferring later enable OCluDTS algorithm to further boost the efficiency. Finally, we conduct extensive experiments on the real data set to evaluate the effectiveness and efficiency of OCluDTS through comparing it with the centralized algorithm. Theoretical analysis and experimental results show that OCluDTS can achieve the superior performance in clustering on the massive amounts of streaming trajectories.

Keywords　distributed trajectory streams; clustering; sliding window model; distributed temporal trajectory cluster feature; concept drift

1　引　言

随着传感器网络技术、通信技术和定位技术的迅猛发展，各类移动智能终端（如智能手机、平板电脑等）得以广泛应用，衍生了海量移动对象的位置信息. 该类位置信息包含了地理坐标、速度、方向以及时间戳等信息，被实时采集且持续增加，形成了时变进化的轨迹流. 快速分析轨迹流数据可以及时有效地发现移动对象的演化规律，服务于广泛的位置服务应用，如基于位置的社交网络[1]、路径规划[2-3]、智能交通管理[4]、道路设施优化[5]等.

作为一种无监督的移动模式发现方法，聚类分析将海量轨迹数据集划分成若干相似的簇，以提取不同移动对象的代表路径或共同移动趋势[6-13]. 例如，在飓风登录预报应用中，发现飓风的共同移动趋势可以提升预报的精度；在动物迁移规律分析应用中，提取动物的共同移动行为可以揭示动物迁移的缘由. 与静态轨迹聚类相比，面向轨迹流的聚类更具挑战. 传统的轨迹聚类方法采用先存储再处理的策略，需要多遍扫描数据，并不适用于轨迹流. 在轨迹数据流的演变进化过程中，移动对象的轨迹在不同时刻可能分属不同簇. 此外，当一部分移动对象产生新的轨迹数据时，另一部分对象可能已经停止移动. 因此，亟需设计驻留内存的概要数据结构（Synopsis data structure）实时维护这些持续到达的轨迹数据集的摘要信息，并结合滑动窗口模型技术在处理新到达数据的同时消除过时数据的影响，以有限的系统资源处理无限增长的轨迹流数据.

在一些位置服务应用中，轨迹数据流由地理上分

布的不同区域的中心服务器持续采集得到.例如,遍布于城市道路的智能卡口查控系统对经过道路卡口的车辆进行 24 h 不间断记录和监测,并将采集到的包括车牌号、车辆经过的时间、速度、方向以及卡口位置等数据实时上传给所在区域的交管中心服务器.各区域中心服务器持续收到的关于车辆的时序位置信息(所经卡口的位置序列)及移动特征(时速、方向等)形成了分布式轨迹流数据.如果将这些分布式轨迹流数据传输到城市交通管理指挥中心服务器集中存储再进行聚类分析,涉及较高的轨迹数据传输、存储和计算开销,无法满足低处理延迟的实时分析需求.因此,亟需设计高性能的分布式聚类方法对驻留在各区域中心服务器(节点)上的轨迹流数据进行在线聚类分析,完成不同路段各行驶方向实时车速/流量的监测统计、可疑(超速)车辆识别等功能,从而帮助交通管理部门做出决策并采取相应的交通控制措施.

现有大多数轨迹流聚类研究[11-13]主要以集中式架构来处理轨迹数据流,面对分散采集的海量、高速且不断演化的分布式轨迹数据流,系统资源的有限性和聚类分析的实时响应需求使得分布式轨迹流聚类面临更大挑战:(1)为了实时处理在各个节点上不断演化的局部轨迹数据流,需要使用滑动窗口模型等技术将轨迹流转变成固定时间间隔内的有限轨迹数据集进行处理;(2)轨迹数据是呈偏态分布的,各节点局部轨迹聚类结果不能较好反映对象的全局移动规律,因此,需要设计先局部聚类再合并产生全局聚类结果的处理方法;(3)节点间传输数据产生的通信开销成为了分布式轨迹流聚类算法效能的瓶颈.为保证节点间通信开销最小化,同时获得与集中式聚类相同精度的聚类结果,需要设计适于分布式聚类的概要数据结构,实时提取各类轨迹的数据特征.采取节点间以概要数据结构形式传输局部聚类结果的方式,相比直接传输轨迹数据给中心节点再集中聚类,大大节省了通信开销.此外,分布式流算法对节点的个数非常敏感,线性增长的节点数将导致更大的通信开销,需要设计有效的节点间通信策略以提高算法的可伸缩性.

当前,存在多种分布式流数据处理架构,如 S4、Storm、Spark Streaming 等,它们在数据处理能力方面具有高可扩展性、容错性和实时性,但是,尚无工作支持轨迹流的聚类分析.本文基于 Spark Streaming 计算框架研究了分布式轨迹流的在线聚类问题.在前期集中式轨迹流聚类研究[11]的基础上,本文提出了一种分布式轨迹流的在线聚类方法.在以 M 个远程节点和 1 个协调者节点组成的分布式网络中,采用远程节点并行聚类局部轨迹数据流以及协调者节点合并产生全局聚类结果的两层聚类思想,将各远程节点实时到达的分布式轨迹流数据划分为若干个相似簇,确保获得与集中式轨迹流聚类相同精度的聚类结果.本文使用开源架构解决分布式轨迹流聚类的策略可以扩展至其他分布式轨迹流的分析研究(如分布式轨迹流的相似性查询、分布式轨迹流的异常检测问题等).本文的研究内容及贡献如下:

(1)面向由多个远程节点和单个协调者节点组成的分布式网络环境,首次提出了分布式轨迹数据流的在线聚类问题.

(2)设计了一种分布式时序轨迹聚类特征指数直方图(DF),并结合滑动窗口模型技术,在增量维护时间窗口内各个轨迹簇聚类特征的同时,删除过时轨迹数据对聚类精度的影响.

(3)提出了一个两阶段的分布式轨迹流聚类方法 OCluDTS.各个远程节点对最新到达的局部轨迹流数据并行聚类,并传输局部聚类结果给协调者节点;协调者节点汇齐局部聚类结果进行合并聚类,产生全局聚类结果.此外,进一步提出了减少远程节点传输开销及协调者节点合并聚类开销的优化策略,在保证聚类质量的同时显著减少执行时间.

(4)基于真实的出租车行驶轨迹数据集进行了大量对比实验.理论分析和实验结果验证了 OCluDTS 算法的有效性和高效性.

本文第 2 节介绍与分布式轨迹流聚类问题相关的研究工作;第 3 节给出定义及问题描述;第 4 节提出分布式轨迹流聚类的算法框架,详细介绍 OCluDTS 算法的技术细节及优化策略;第 5 节使用实际轨迹数据集对 OCluDTS 算法进行实验分析;最后,总结全文并展望后续工作.

2 相关工作

近年来,轨迹数据的聚类分析[6-13]与分布式数据流的聚类[14-17]均已被广泛研究,然而,分布式轨迹流的聚类分析由于有限的系统资源(内存和计算能力)、海量数据的高速到达率、轨迹数据的偏态分布性以及聚类的实时处理需求等特点,目前仍处于初期探索阶段.

2.1 分布式数据流聚类

为了处理带有噪声或不完整数据的分布式数据流,Zhou 等人采用远程节点与协调者节点的两层框

架，提出了一个基于期望值最大化的分布式数据流聚类方法 CluDistream[14]. 他们通过最大化数据聚类可能性的方式提取数据流的分布情况，首次提出了先测试再聚类(test-and-cluster)的策略以减少平均处理开销. Cormode 等人基于局部聚类结果合并全局聚类的思想提出了系列 k-中心聚类策略(包括最远点聚类、并行猜测聚类等)的分布式数据流聚类算法[15]，通过实验对 localFP、localPG、globalFP 及 globalPG 等四种分布式方法的通信开销和聚类精度进行了评测. Zhang 等人基于三种拓扑设置提出了一系列近似 k 中值分布式数据流聚类算法[16]，减少了各节点间的最大传输开销. Yin 等人针对分布式的时间序列数据流提出了一个单遍增量聚类技术 DSIC[17]. 基于分层的传感器网络，DSIC 先用哈尔小波转换压缩时间序列数据，再以动态时间规整(DTW)距离度量对近似时间序列数据进行增量分层划分成类.

然而，轨迹数据流不同于传统的数据流，持续采集的位置信息不仅具有时序相关性，而且表现为呈偏态分布的数据流，因此，上述分布式数据流的聚类方法不能直接用于处理分布式轨迹流.

2.2　集中式轨迹流聚类

在轨迹流聚类的相关研究中，Jensen 等人设计了一个可增量维护的聚类特征，并提出了移动对象的持续聚类框架[18]. Li 等人提出了移动微簇的概念，可增量维护聚簇的特征[19]. 但是，上述方法只关注增量聚类移动对象而非轨迹数据. 文献[6]中，Li 等人提出了一个轨迹的增量聚类框架 TCMM，包括轨迹线段的微聚类维护和宏聚类产生两个阶段. 但是，该方法既不针对在线聚类问题，也没有考虑轨迹的时间特性，不适于处理大规模高速到达的轨迹流数据.

Costa 等人结合 LIFTING 和傅里叶变换方法提出了一个处理轨迹流聚类的框架[12]，通过使用不可分傅里叶变换方法对轨迹流数据进行预先精简表示以减少处理的数据量. Yu 等人提出了 CTraStream 方法用于聚类轨迹流数据，包括在线的轨迹线段流聚类和基于 TC 树索引结构的近邻簇持续更新过程[13]. 但是，以上两种轨迹流聚类方法都没有考虑删除过时轨迹对聚类精度的影响. 随着轨迹数据的不断增加，受过时数据的影响，类的大小会逐渐增大，相应地，类中心会逐渐偏移，导致“概念漂移”产生，降低了聚类结果的有效性. 鉴于时变轨迹流带来的概念漂移问题，本文前期工作基于滑动窗口模型提出了一个两阶段(包括在线增量微聚类与离线宏聚类)的集中式轨迹流聚类方法 TSCluWin[11]. TSCluWin 方法在并入新到达的轨迹数据的同时消除了过时的轨迹数据，能有效获取当前时间窗口内聚类分布的变化，避免了概念漂移对聚类有效性的影响. 然而，该方法只是针对轨迹数据集中式采集存储的应用，不能有效处理地理上分散采集获得的轨迹流数据.

2.3　轨迹数据的分布式处理

面对静态轨迹数据的分布式批处理，Deng 等人结合 GPU 技术提出了一个基于密度的并行轨迹聚类算法 Tra-POPTICS[20]，但是该方法使用外存作为中间结果数据的存储介质，需要巨大的 I/O 开销，不适合处理持续到达的分布式轨迹流数据. 为解决分布式轨迹数据流上的 Top-k 相似性查询问题，Zhang 等人提出了 DT-KST 算法[21]. 该算法利用 Haar 小波的多粒度特性，将数据由粗粒度到细粒度逐层发送到远程节点供其逐步剪枝，在保证查询实时性的同时，能有效降低节点间的通信开销. Galic 等人基于分布式流引擎设计了一个分布式处理时空数据流的框架，通过定义的时空数据类型与相应的操作集合，支持时空数据流上的实时查询[22]. 然而，上述两种方法都不适用于解决分布式轨迹数据流的在线聚类问题.

综上所述，由于轨迹数据固有的时序相关性与偏态分布性，已有的分布式数据流聚类方法与集中式轨迹流聚类方法不能直接用于分布式轨迹流聚类. 此外，轨迹数据的分布式处理方法不能用于解决分布式轨迹流的在线聚类问题. 鉴于此，本文基于滑动窗口模型提出了一种通信开销较小的分布式轨迹流在线聚类方法，持续评测时变进化轨迹数据的分布情况与移动趋势.

3　问题定义

本节将给出基本符号定义和概念，如表 1 所示.

表 1　符号定义列表

符号	定义
S_i	远程节点的局部轨迹流
W	滑动窗口大小
$p_i^{(j)}$	一个移动对象 $o^{(j)}$ 在时刻 t_i 的位置
L_i	连接两个连续位置点的轨迹线段
DF	分布式时序轨迹聚类特征指数直方图
TF	分布式时序轨迹聚类特征
ε	误差阈值(定义 5)
cen	DF 或 TF 的中心点位置
θ	DF 或 TF 的角度
k	协调者节点上 DF 数的最大值
α	空间距离度量权重
M	分布式网络中远程节点数

本文考虑一个由单个协调者节点与 M 个远程节点组成的分布式网络. 令 S 为 M 个远程节点收到的局部轨迹流集合,即 $S=\bigcup_{i=1}^{M}S_i$. 其中,单个远程节点 i 到达的局部轨迹流 S_i 定义如下.

定义 1. 局部轨迹流(local trajectory stream). 一个远程节点上到达的局部轨迹流是由多个移动对象不断增长的位置记录序列组成,记为 $S_i=\{p_1^{(1)},p_1^{(2)},p_1^{(3)},\cdots,p_2^{(1)},p_2^{(2)},p_2^{(3)},\cdots\}$. 其中,$p_j^{(l)}$ 是移动对象 $o^{(l)}$ 在时刻 t_j 的位置,即 $p_j^{(l)}=(x_j^{(l)},y_j^{(l)})$.

为了实时评测轨迹流数据的分布情况,需要对最近时段内不断到达的轨迹数据集进行在线聚类. 滑动窗口模型可将无限的轨迹数据流切分成若干有限的子轨迹流,保证聚类分析仅针对当前时间窗口内最新到达的轨迹数据集进行处理. 因此,对于各远程节点持续到达的轨迹流数据,使用基于时间的滑动窗口模型将其转变成固定时间间隔内的有限轨迹数据集进行处理,即实时提取并增量维护最近到达数据的聚类特征,又继而降低概念漂移对轨迹聚类精度的影响.

定义 2. 基于滑动窗口(Sliding time-based Window)的局部轨迹流. 给定一个局部轨迹流 S,滑动窗口大小 W,窗口开始时刻 t_s,基于滑动窗口的局部轨迹流是一个有限的轨迹流元素集,即 $S_W=\{p_s^{(1)},\cdots,p_s^{(l)},\cdots p_{s+W-1}^{(1)},\cdots,p_{s+W-1}^{(l)},\cdots\}$.

每当窗口向前滑动一个时刻,当前时间窗口的开始时刻与结束时刻均增 1. 滑动窗口模型使得在任意时刻只考虑对最近到达的轨迹进行聚类,即仅当前时间窗口 (t_s,t_{s+W-1}) 内的轨迹是有效的,而 t_s 之前的轨迹被视为过期数据. 在当前时间窗口内,同一对象任意两个相邻的位置点 $(p_i,p_{i+1})(s\leq i<s+W-1)$ 可以连接成一个轨迹线段 L_i,即 $L_i=(p_i,p_{i+1})$,因此,移动对象的轨迹可看作线段的集合,表示为 $\{L_1,L_2,\cdots\}$.

对当前时间窗口内的轨迹线段进行聚类需评测轨迹线段间的相似性. 在实际应用中,两个移动对象产生的轨迹线段如果空间距离较近但时间间隔较远,应视为不相似. 因此,在轨迹相似性计算时,本文将轨迹线段的相似性评测定义为空间距离与时间间距的加权和度量.

定义 3. 差异性度量(Distance Measurement). 给定一个空间邻近权重 $\alpha(0\leq\alpha\leq1)$,任意两个时刻 (t_i,t_j) 到达的轨迹线段 L_i 和 L_j,轨迹线段的空间距离度量 $SD(\cdot,\cdot)$,两条轨迹线段的长度分别表示为 $\|L_i\|$ 和 $\|L_j\|$,以 $\mathrm{sig}(\cdot)$(这里,$\mathrm{sig}(\cdot)=\frac{1}{1+e^{-(\cdot)}}$)对空间距离与时间距离进行归一化处理. 则两条轨迹线段的差异性度量如下:

$$Dist(L_i,L_j)=\alpha\cdot\mathrm{sig}\left(\frac{SD(L_i,L_j)}{\|L_i\|+\|L_j\|}\right)+(1-\alpha)\cdot\mathrm{sig}(|t_i-t_j|).$$

通常赋予空间近似性度量更大权重,设定 $\alpha>\frac{1}{2}$. 一般而言,一个较小的 $Dist$ 值表示两条轨迹线段的相似性高,即时间、空间维度上的距离接近. 在评测空间距离度量时,考虑大多数位置服务应用场景中轨迹都是双向的(如道路由两个相反方向的多个行车道构成),本文使用了文献[7]中改进的 Hausdorff 距离. 即

$$SD(L_i,L_j)=\max(sd(L_i,L_j),sd(L_j,L_i)),$$
$$sd(L_i,L_j)=\max(\min(\|p_{i_s},p_{j_s}\|,\|p_{i_s},p_{i_e}\|),\min(\|p_{i_e},p_{j_s}\|,\|p_{i_e},p_{j_e}\|)),$$

其中,p_{i_s}(或 p_{j_s})以及 p_{i_e}(或 p_{j_e})分别表示两条轨迹线段的开始及结束位置.

为确保轨迹数据的实时聚类,需要尽可能减少远程节点与协调者节点间的通信开销(减少数据传输量),即远程节点传输给协调者节点的局部聚类结果大小. 本文定义了一种紧凑的聚类摘要数据结构——分布式时序轨迹聚类特征的指数直方图(DF)以提取各簇的摘要信息(聚类特征). 可根据轨迹的到达时刻将 DF 划分为若干个桶,每一个桶存储单个时刻提取的不同簇轨迹集的聚类特征,即分布式时序轨迹聚类特征(TF).

本文采用最小包围盒(MBR)描述 TF 包含的轨迹线段集所覆盖的空间区域,考虑将其中所包含的轨迹线段集的中心点与角度的线性和等统计量以及轨迹数与到达时间等作为代表各轨迹簇的聚类特征. 与文献[11]不同,鉴于聚类结果容易受到较长轨迹的影响,结合轨迹线段的长度提取了线段角度与长度乘积的线性和,通过计算其与线段长度线性和的商获得轨迹簇的角度特征.

定义 4. 分布式时序轨迹聚类特征 TF(Distributed Temporal trajectory cluster Feature). 给定当前时刻一个远程节点上到达的轨迹线段集合 $C=\{L_1,L_2,\cdots,L_n\}$,$TF=\{LS_{cen},LS_p,LS_{len},BL,TR,n,t\}$. 其中,

LS_{cen} 是所有轨迹线段中心点的线性和;

LS_p 是所有轨迹线段角度与长度乘积的线性和;

LS_{len} 是所有轨迹线段长度的线性和;

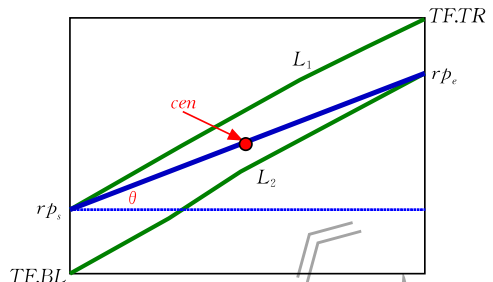
BL 是包含轨迹线段集的 MBR 的左下角;

TR 是包含轨迹线段集的 MBR 的右上角;

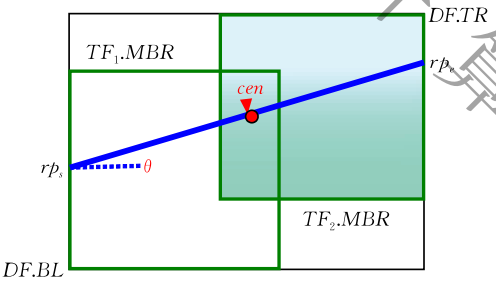
n 是轨迹线段集的线段数;

t 是轨迹线段集中线段的最晚到达时刻.

以图 1 中(a)图为例,一个 TF 包含两条轨迹线段(L_1 和 L_2),以 MBR 表示. 使用代表轨迹(rp)描述 TF 中轨迹线段集的共同移动行为,其计算办法与文献[11]中一致. 根据 TF 计算得到的中心点($cen=LS_{cen}/n$)和角度($\theta=LS_p/LS_{cen}$)绘制了一条粗的蓝色直线,与 MBR 边界相交的两个交界点作为代表轨迹的起点 s 和终点 e,记为($TF.rp_s$, $TF.rp_e$).



(a) TF及其代表轨迹线段



(b) DF及其代表轨迹线段

图 1　代表轨迹线段示例

性质 1(综合性).　给定两个轨迹线段集 C_1、C_2 的分布式时序轨迹聚类特征 $TF(C_1)$ 和 $TF(C_2)$,满足 $C_1\cap C_2=\varnothing$,则 $TF(C_1\cup C_2)$ 可由 $TF(C_1)$ 和 $TF(C_2)$ 构建.

证明.　根据定义 4,$TF(C_1\cup C_2)$ 中 LS_{cen},LS_p、LS_{len} 和 n 可以分别由 $TF(C_1)$ 和 $TF(C_2)$ 中的对应项累加得到,BL 可由 $\min\{TF(C_1).BL, TF(C_2).BL\}$ 获得,而 TR 和 t 分别由 $\max\{TF(C_1).TR, TF(C_2).TR\}$ 以及 $\max\{TF(C_1).t, TF(C_2).t\}$ 计算得到.

分布式时序轨迹聚类特征指数直方图可以近似地获取各个时刻到达的不同簇轨迹的聚类特征,确保不满足当前窗口时间范围的 TF 将被有效删除. 考虑直方图内过时轨迹会导致聚类结果产生误差,将分布式时序轨迹聚类特征的指数直方图定义如下.

定义 5.　分布式时序轨迹聚类特征指数直方图 DF(Exponential Histogram of Distributed Temporal Trajectory Cluster Feature). 给定误差阈值 ε,DF 是远程节点上的一个轨迹簇根据轨迹到达时刻产生的不同子集$\{C_1,\cdots,C_i,C_j,\cdots\}$($1\le i<j$)所提取的 TF 集合,满足以下约束条件:

(1) 令 $|C_i|$ 为轨迹线段集合 C_i 的大小,则 $|C_1|=1$,对于任意 $i\ge 2$,$|C_i|=|C_{i-1}|$ 或 $|C_i|=2|C_{i-1}|$;

(2) 当 $|C_i|=2^l$ 时,称 C_i 的级别为 l,所对应的 TF 称为 l 级 TF,各级 TF 数至多为 $\left\lceil\frac{1}{\varepsilon}\right\rceil+1$.

引理 1.　给定一个包含 n_i 个轨迹线段的 DF,过时的轨迹线段数满足 $[0,\varepsilon n_i]$,该 DF 至多包含 $\left(\frac{1}{\varepsilon}+1\right)(\log(\varepsilon n_i+1)+1)$ 个 TF.

证明.　DF 中只有最高级 TF(即最早创建的 TF)中包含过时轨迹,令 n_s 表示其包含的轨迹线段数,根据定义 5,有 $\frac{1}{\varepsilon}(1+2+4+\cdots+n_s)\le n_i$,则 $n_s\le\varepsilon n_i$ 得证. 此外,根据给定的滑动窗口大小 n_i 和误差阈值 ε 构造一个指数直方图,每个 TF 与指数直方图中的一个桶一一对应,由文献[23]可知,指数直方图结构可用 $\left(\frac{1}{\varepsilon}+1\right)(\log(\varepsilon n_i+1)+1)$ 个桶维护一个 ε 误差的概要,因此,一个 DF 至多包含 $\left(\frac{1}{\varepsilon}+1\right)(\log(\varepsilon n_i+1)+1)$ 个 TF.

在 DF 并入新的轨迹线段时,采用以下方式进行增量式维护. 对于新加入的轨迹线段根据定义 4 为其生成一个新的 0 级 TF 加入到 DF,如果存在超过 $\left\lceil\frac{1}{\varepsilon}\right\rceil+1$ 个 0 级 TF,根据性质 1 将最早产生的两个 0 级 TF 合并生成一个新的 1 级 TF. 随后,对于较高级别 $l\ge 1$,继续合并 l 级中最早创建的两个 TF,直到该级 TF 数不超过 $\left\lceil\frac{1}{\varepsilon}\right\rceil+1$ 为止. 图 2 中的示例描述了一个 DF 在并入 L_1 到 L_{16} 等 16 个轨迹线段时各级 TF 的维护过程. 设定 $\varepsilon=1/4$,表示 DF 中各级至多保持 5 个 TF. 在并入新到达的 L_6 时,为其创建一个 0 级 $TF(TF_0(L_6))$,此时 0 级 TF 的个数已增至 6,导致 $TF_0(L_1)$ 和 $TF_0(L_2)$ 合并产生一个新的 1 级 $TF(TF_1(L_1,L_2))$. 在并入 L_{16} 时也有类似的 TF 合并过程. L_{16} 的加入不仅触发了 $TF_0(L_{11})$ 和 $TF_0(L_{12})$ 的合并生成 $TF_1(L_{11},L_{12})$,而且进而引发 $TF_1(L_1,L_2)$ 和 $TF_1(L_3,L_4)$ 合并生成 $TF_2(L_1,L_2,L_3,L_4)$.

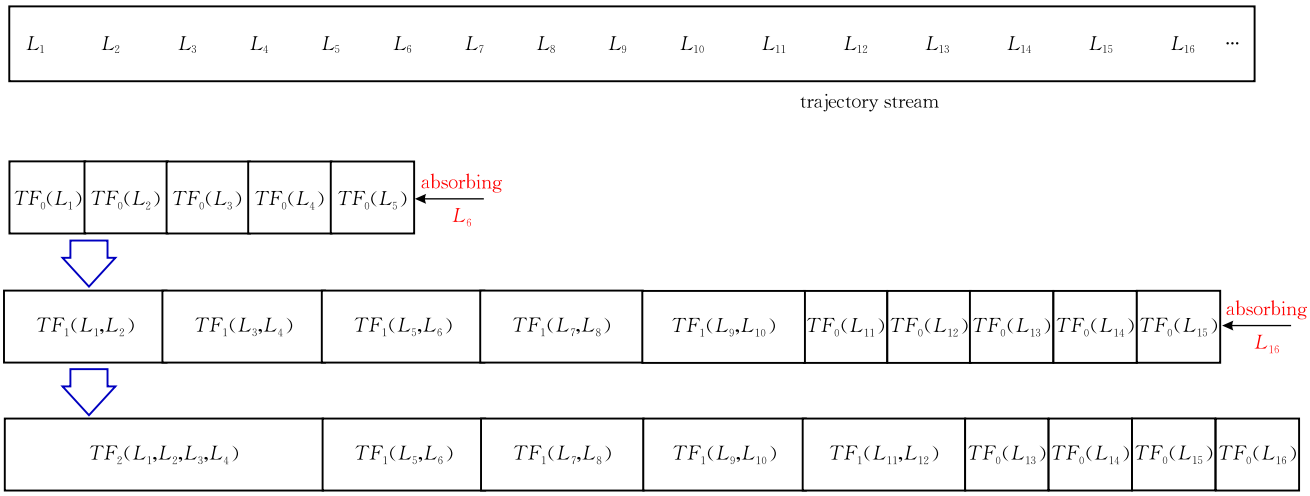


图 2 一个 DF 的合并过程示例 $\left(\varepsilon=\frac{1}{4}\right)$

对于 DF 代表轨迹线段(表示为 $DF.rp$)的计算,类似于 TF 代表轨迹计算,通过分别计算它包含的所有 TF 的中心点和角度的加权平均值获得,即

$$DF.cen=\frac{\sum_i TF_i.LS_{cen}\cdot TF_i.n}{\sum_i TF_i.n},$$

$$DF.\theta=\frac{\sum_i TF_i.LS_p}{\sum_i TF_i.LS_{len}}.$$

以图 1 中(b)图为例,根据性质 1,由 DF 中两个 TF(TF_1 和 TF_2)的 MBR 获得 DF 的 MBR. 随后,根据 $DF.cen$ 和 $DF.\theta$ 画出一条蓝色的粗线段并延伸至与 DF 的 MBR 边界相交,获得的两个交界点视为 DF 的起点和终点,记为($DF.rp_s$,$DF.rp_e$).

TF 与 DF 的概要数据结构提取原始轨迹簇的统计信息,可以满足分布式轨迹流数据处理及传输局部聚类结果的紧凑型要求,从而减少网络节点间的数据传输量. 它们不仅能有效维护远程节点上滑动窗口内各个时刻轨迹簇的聚类特征,DF 概要结构还能在聚类轨迹的同时删除过时轨迹线段的影响,确保各个簇不会因为并入新轨迹而逐渐扩大,避免簇中心的迁移(即概念漂移). 此外,远程节点上的每个 DF 维护自己的指数直方图,仅当新到达轨迹线段不能并入该节点的任何 DF 时,才会为其产生一个只包含一个 0 级 TF 的 DF. 如果该轨迹线段为噪声数据,在当前时间窗口内不会有新的轨迹线段并入其所在的 DF,而包含它的 DF 一旦过时就会被删除,也不会对其后的聚类造成影响.

定义 6. 问题定义:在一个由 M 个远程节点与 1 个协调者节点构成的分布式网络中,面对所有远程节点持续收到的局部轨迹流集合 $S=\bigcup_{i=1}^{M}S_i$,给定滑动窗口大小 W,设计较小通信与计算开销的在线分布式轨道流聚类算法,将各远程节点上当前时间窗口内各时刻到达的所有轨迹线段集根据轨迹相似性准则划分成不同簇,确保获取与集中式轨迹流聚类相同精度的全局聚类结果.

4 基于滑动窗口模型的分布式轨迹流在线聚类算法

4.1 在线分布式轨迹流聚类框架

本文研究分布式轨迹流的在线聚类问题,考虑一个由 M 个远程节点和 1 个协调者节点组成的分布式计算环境,如图 3 所示. 移动对象产生的轨迹流数据在各个远程节点上持续到达,协调者节点负责接受用户的聚类需求以及输出全局聚类结果. 由于直接从各远程节点将局部轨迹流数据发送给协调者节点再集中聚类的方法会带来大量的计算与通信开销,采用各远程节点对最新到达轨迹数据执行并行聚类的方式,可提高在线轨迹流聚类的效率. 本文提出了一个分布式轨迹流在线聚类算法 OCluDTS,主要包含两个基本处理过程,即各远程节点的并行局部轨迹流聚类与协调者节点的全局合并聚类. 考虑到第二阶段各远程节点将局部聚类结果发送给协调者节点可能引致较大的通信开销,本文通过减少传输的局部聚类结果大小以减少通信时间. 具体地,远程节点与协调者节点使用相同的概要数据结构 DF(定义 5)描述其生成的聚类结果. 各个时刻协调者节点仅对远程节点传输的 DF 集合进行合并聚类,

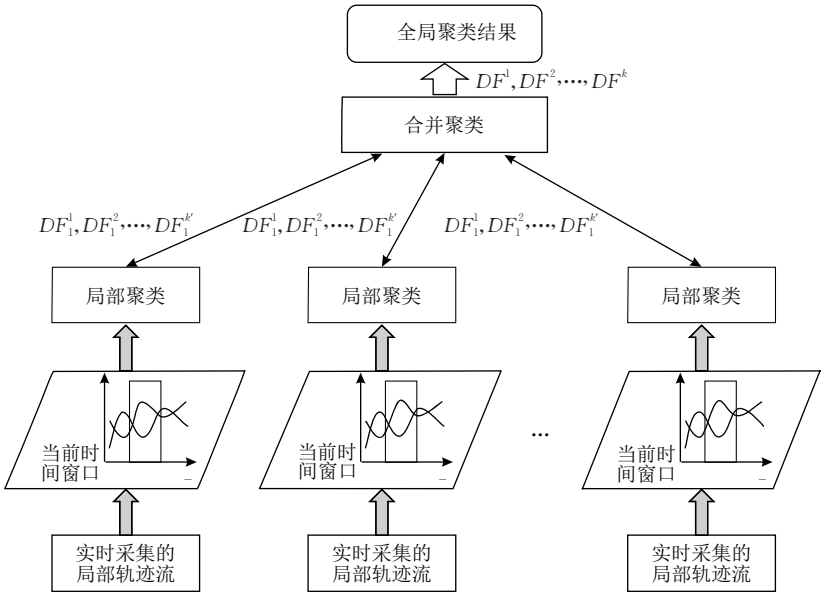


图 3　分布式轨迹流聚类框架

在保证较好聚类精度的同时能节省大量的计算开销与通信开销.

首先,对于各个远程节点持续到达的局部轨迹流数据,使用滑动窗口模型将其切分成有限时段内的轨迹数据集进行局部聚类分析. 各个远程节点将实时产生的局部聚类结果(k'个 DF,$k'<k$)发送给协调者节点,随后,由协调者节点对收到的来自 M 个远程节点的局部聚类结果(即 $M\cdot k'$个 DF)执行合并聚类,产生 k 个 DF 作为全局聚类结果输出. 采用先局部聚类再合并聚类结果的两阶段聚类模式,可以确保获得与集中式轨迹流聚类相同的聚类精度. 此外,由于分布式网络中只涉及远程节点向协调者节点传输局部聚类结果(k'个 DF),相比直接传输轨迹数据给协调者节点再集中聚类而言,大大减少了分布式算法中远程节点与协调者节点间的通信开销,提高了分布式算法的执行效能.

4.2　远程节点的局部轨迹流聚类算法

远程节点将待处理的局部轨迹流数据按时间窗口进行分批处理,基于滑动窗口模型的方法确保实时处理当前时间窗口中持续到达的轨迹流数据. 各远程节点在聚类最新到达数据的同时删除过时轨迹的影响. 随后,将本节点聚类产生的局部聚类结果(k'个 DF)传送给协调者节点. 在滑动窗口内的局部轨迹流聚类实质上是对最近到达的轨迹线段集进行在线增量聚类,具体过程参见算法 1 第 1～12 行. 令 Z_i 为第 i 个远程节点聚类产生的 DF 集,则 Z_i 的大小最大为 k',表示任何时候远程节点的内存中最多保存 k'个 DF.

算法 1.　OCluDTS 算法.

输入:局部轨迹流集合$\{S_1, S_2, \cdots, S_M\}$,滑动窗口大小 W,协调者节点上的聚类数 k,当前时刻 t_c

输出:t_c 时刻产生的 DF 集 Z

远程节点 i: //对 t_c 时刻到达的局部轨迹线段流数据进行聚类,发送局部聚类结果(k'个 DF)给协调者节点

1. $Z_i\leftarrow\varnothing$ //Z_i 表示远程节点 i 上的 DF 集
2. FOR 局部流 S_i 中的每条轨迹线段 L_q DO
3. 搜索与 L_q 最相似簇的 DF(表示为 h);
4. IF $Dist(h.rp, L_q)<d_{\min}$ THEN //$d_{\min}$ 表示将任意轨迹线段并入已有簇的最小差异度阈值
5. $Absorb(L_q, h)$; //将轨迹线段 L_q 并入 h
6. ELSE IF $|Z_i|<k'$ THEN
7. 创建一个包含 $TF_0(\{L_q\})$ 的 DF (h_{new});
8. $Z_i\leftarrow Z_i\cup\{h_{\text{new}}\}$;
9. ELSE
10. 检查 Z_i 中是否包含过时的 DF,如果有则删除,否则合并 Z_i 中最为相似的簇,确保聚类数不超过 k';
11. END IF
12. END FOR
13. 发送产生的 DF 集 Z_i 给协调者节点;

协调者节点: //接收 t_c 时刻 M 个远程节点产生的局部流聚类结果(至多($M\times k'$)个 DF),执行合并聚类输出结果 Z(k 个 DF).

14. WHILE 收到任意远程节点 j 的聚类结果 Z_j DO
15. $Z\leftarrow Z\cup Z_j$;
16. $j\leftarrow j+1$;
17. IF $|Z|>k$ THEN
18. 合并 Z 中最为相似的簇
19. END IF

20. IF $j=M$ THEN
21. break;
22. END IF
23. END WHILE
24. $j\leftarrow 0$
25. RETURN Z;

当收到一条新轨迹线段 L_q 时，根据相似性度量(定义 3)在 Z_i 中为 L_q 寻找最为相似的簇，即在最近时间间隔内空间距离 L_q 最近的 DF，如果找到该簇(以 h 表示)．通过调用函数 $Absorb(L_q,h)$ 将 L_q 并入 h，详细过程见算法 2．首先为 L_q 创建一个 $TF_0(\{L_q\})$ 加入 h(算法 2 第 1 行)，根据定义 5，如果 h 中 0 级 TF 的数量超过 $\left\lceil\frac{1}{\varepsilon}\right\rceil+1$，则 0 级 TF 中最早创建的两个 TF 被合并产生一个 1 级 TF．这类合并低一级 TF 产生高一级 TF 的操作将会反复执行，直到某一级 TF 的数量不超过 $\left\lceil\frac{1}{\varepsilon}\right\rceil+1$ 为止(算法 2 第 3～6 行)．为了避免过时轨迹对最新聚类结果的影响，在将 L_q 并入 h 的过程中需要检查 h 是否包含已过时的 TF，也就是判定 h 的最高级 TF 中最早创建的 TF 是否已过时并予以删除(算法 2 第 7～9 行)．

算法 2. Absorb 算法．

输入：新到轨迹线段 L_q，与线段 L_q 最相似簇的 DF h
输出：h

1. $h\leftarrow h\bigcup\{TF_0(L_q)\}$;
2. $l\leftarrow 0$;
3. WHILE $n_l>\left\lceil\frac{1}{\varepsilon}\right\rceil+1$ DO //n_l 表示 l 级 TF 的数量
4. 合并 h 的 l 级 TF 中两个最早创建的 TF 成为一个($l+1$)级 TF;
5. $l\leftarrow l+1$;
6. END WHILE
7. IF h 中最早建立的 TF(表示为 TF_{lst})过时 THEN
8. 从 h 中删除 TF_{lst};
9. END IF
10. RETURN h;

如果在 Z_i 中没有找到与 L_q 最为相似的簇，判定当前时刻聚类数是否超过 k'，在类数未超过 k' 的前提下，直接创建一个包含 $TF_0(L_q)$ 的新 $DF(h_{\text{new}})$ 加入 Z_i(算法 1 第 6～8 行)．如果已超过 k'，先检查是否存在过时的 DF 并删除它(算法 1 第 10 行)．通过删除过时的 DF 可以消除过时轨迹对聚类结果的影响，从而避免概念漂移．在无过时的 DF 可删除的情况下选择合并最为相似的 DF(算法 1 第 10 行)．在合并 DF 时考虑到即使两个最相似的 DF 也面临对应级 TF 时间不一致的情况，本文首先根据两个 DF 各级 TF 的创建时间执行对齐操作，再根据定义 5 分别合并相应级 TF，类似于 Absorb 算法，一旦两个 DF 中对应级 TF 的数量超过 $\left\lceil\frac{1}{\varepsilon}\right\rceil+1$，则相应级 TF 合并产生高一级 TF，以此类推，直到某一级 TF 的数量不超过 $\left\lceil\frac{1}{\varepsilon}\right\rceil+1$ 为止．

实际应用中各个远程节点到达的轨迹数据是偏态分布的．少数远程节点可能存在一段时间内没有收到新轨迹数据的情况，如果不及时删除该类过时数据，也会影响最新时刻轨迹聚类结果的精度．因此，本文对各个远程节点根据预设的时间阈值周期性检查并删除较长时间未更新的 DF，进一步减少过时数据对聚类有效性的影响．

4.3 协调者节点的全局轨迹流聚类算法

由于远程节点仅对各自采集到的局部轨迹流数据进行聚类，可能导致处于数据采集交叉区域的同一类轨迹数据被划分到多个节点的不同簇中，因此，考虑将各远程节点的分布式局部聚类结果进行合并聚类产生全局聚类结果．由协调者节点对于当前时刻收到的多个远程节点的局部轨迹流聚类结果($M\cdot k'$ 个 DF)执行合并聚类，产生最终的全局聚类结果(k 个 DF)，在存入本地磁盘的同时将其计算得到的 k 个代表轨迹作为结果输出(算法 1 第 14～25 行)．

当协调者节点收到所有局部轨迹流聚类结果时，将各簇进行两两比较，以满足给定距离阈值且距离最短为相似性准则寻找相似簇，如果找到则按照性质 1 合并两个代表轨迹所表示的簇．任何时刻协调者节点至多保留 k 个簇，k 值的确定根据协调者节点最大内存处理能力而定．以 DF 提取聚类特征的各簇可由所产生的代表轨迹线段表示，因此，对局部聚类结果($M\cdot k'$ 个 DF)的合并聚类实质上是执行 $M\cdot k'$ 个代表轨迹线段的聚类，这里的聚类技术可以是任意加权聚类方法，包括基于划分的方法、基于层次的方法以及基于密度的方法等．

4.4 分布式轨迹流聚类算法的时间开销分析

相比直接处理所有新到达轨迹数据(令 n 表示数据量)的集中式聚类，分布式聚类将其计算负载划分到多个远程节点并行执行，显著降低了聚类处理开销．OCluDTS 算法的执行开销由各远程节点的并行局部聚类开销 Cp_{l_i}($1\leq i\leq M$)、协调者节点的合

并聚类开销 Cp_c 以及各远程节点与协调者节点之间的通信开销 $Cm_{l_i}(1\leq i\leq M)$ 组成. 令 $n'(n'>n/M)$、$n''(n''\geq 0)$ 分别表示各远程节点当前时刻到达的最大、最小轨迹线段数，各远程节点在当前时刻的最大局部聚类开销包括为新到轨迹线段寻找最为相似的轨迹簇的开销 $O(n'\cdot k')$、合并最相似轨迹簇的开销 $O(k'^2)$、删除过时轨迹簇的开销 $O(k')$，则 Cp_{l_i} 至多为 $O(n'\cdot k'+k'^2+k')$. 当远程节点收到的轨迹数据量比较大时，$n'\gg k'$，则远程节点当前时刻的最大局部聚类开销可近似为 $O(n'\cdot k')$. 对于协调者节点的合并聚类计算开销 Cp_c，主要是为收到的每个轨迹簇寻找最相似簇的开销. 在协调者节点陆续收到各个远程节点的局部聚类结果（k' 个 DF）后，根据轨迹相似性度量对相似的 DF 进行合并，Cp_c 至多为 $O(M^2k'^2)$.

由于轨迹数据的不均匀分布性以及各远程节点不同的处理能力，协调者节点采用多线程收集多个远程节点传送的局部聚类结果，因此所有远程节点向协调者节点发送局部聚类结果的时间开销小于其串行传输时间总和 $\sum_{i=1}^{M}Cm_{l_i}$. 各远程节点与协调者节点间的通信开销主要是用于传输局部聚类结果（k' 个 DF）的时间，其值可以通过最早完成并行聚类的远程节点传输结果的开始时刻与所有远程节点传输结果的结束时刻之间的差值计算得到. 令 $|DF|$ 代表 DF 的大小，则当前时刻由远程节点向协调者节点传输局部聚类结果的最大传输开销为 $O((n'-n''+|DF|)\cdot k')$. 分布式轨迹流聚类算法总的执行开销包括远程节点的最大局部聚类开销、传输局部聚类结果的最大开销与协调者节点合并聚类的开销.

4.5　算法优化策略

为提高 OCluDTS 算法的性能，考虑进一步减少算法的计算开销以及协调者节点与远程节点间的通信开销. 首先，在协调者节点对局部聚类结果（$M\cdot k'$ 个 DF）进行合并聚类的过程中，需要执行 $(M\cdot k')^2$ 次簇间（即 DF）相似性比较，随着远程节点数的增加，会导致较高的计算开销. 对于协调者节点上的合并聚类考虑通过减少簇间相似性的比较次数以降低计算开销. 实际的位置服务应用中，分布式轨迹数据流由地理上分散的不同区域中心服务器独立采集得到，同一时刻距离较远的区域所采集的轨迹不可能属于相同簇. 因此，在协调者节点选择需要合并的轨迹簇时，只考虑比较各区域聚类产生的轨迹簇与其邻近区域的轨迹簇间的相似性，即比较各个远程节点与其邻近节点间的簇间相似性，以减少协调者节点的合并聚类开销. 令 $M'(M'<M/2)$ 表示各远程节点的最多邻近节点数，则协调者节点的合并聚类计算开销 Cp_c 至多为 $O(M\cdot M'\cdot k'^2)$.

其次，分布式轨迹流聚类算法因受限于网络带宽，通信开销成为算法的瓶颈. OCluDTS 算法在协调者节点与远程节点间只传输以概要数据结构（DF）提取的局部聚类结果，相比传输局部轨迹数据流本身而言明显降低了通信开销. 但随着远程节点数的增多，仍然存在较多的传输开销. 为进一步提升分布式算法的性能，考虑尽可能地减少各远程节点向协调者节点传输局部聚类结果的时间 Cm_{l_i}. 轨迹数据具有偏态分布性，以城市路网为例，部分路段短时间内有大量车通过产生较多轨迹，而一些路段较长时段内却只有少数轨迹发生. 鉴于路网中大部分路段区域在短时间内保持轨迹的局部性，即无明显的聚类分布变化，本文考虑当前时刻聚类分布没有变化的远程节点无需传输聚类结果给协调者节点. 对于各个远程节点在聚类新到达轨迹数据集产生的聚类结果，通过计算当前时刻与上一时刻聚类结果距离平方和的平均值（表示为 $AverageSSQ$）之间的差值以判定当前聚类结果是否有变化. 具体定义如下：对于以 DF_i 代表的各类，首先计算其包含的所有轨迹线段 $L_j^i(0<i\leq k,0<j\leq n_i)$ 与 DF_i 的代表轨迹线段之间的距离平方和，再获得所有轨迹线段的平均的 SSQ 值. 表示为 $AverageSSQ=\frac{1}{n}\sum_{i=1}^{k}\sum_{j=1}^{n_i}SD^2(L_j^i,DF_i.rp)$. 以 $AverageSSQ_C$ 和 $AverageSSQ_P$ 分别代表当前时刻和上一时刻聚类结果产生的平均 SSQ 值，当 $|AverageSSQ_C-AverageSSQ_P|<\delta$（$\delta\leq 0.0001$）时，视其为聚类分布无变化无需传输聚类结果给协调者节点. 在处理大数据时，随着远程节点数的线性增加可明显减少传输的数据量. 此外，通过减少远程节点传输的局部聚类结果集，继而降低了协调者节点的合并聚类开销. 因此，基于协调者节点相似性计算剪枝策略及远程节点先测试再传输局部聚类结果的优化策略，OCluDTS 算法的性能得到显著提升.

5　实验结果与分析

5.1　实验设置

算法进行了有效性与可伸缩性的综合实验评估，并与集中式轨迹流聚类算法 TSCluWin[11] 进行

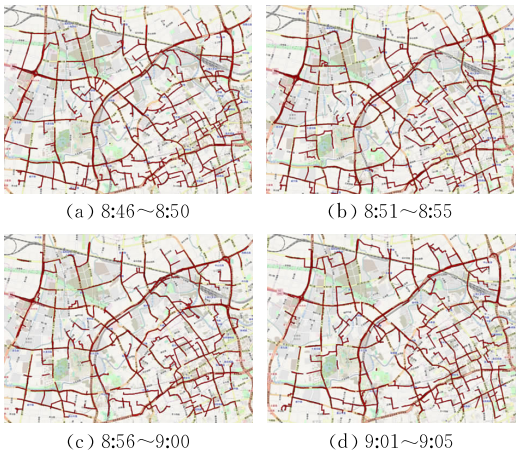
了对比，算法用 Java 实现. 实验采用 5 台服务器搭建了 spark 集群. 节点配置如下：CPU：Xeon E5-2630 v4 × 2(2.20 GHz)，内存：16 GB × 14，硬盘：480 GB SSD，操作系统：centos 7.2，各节点上运行 spark-2.1.0-bin-hadoop2.7 版本.

实验数据采用来自第三方企业的真实出租车轨迹数据集(TaxiShanghai13)，数据集中的每条记录由时间戳、车辆 ID、经度、纬度、速度、角度、运营状态等字段组成. 整个数据集包含上海市约 13 400 辆出租车在 2013 年第四季度产生的 GPS 轨迹采样点数据，数据集大小约 200 GB.

由于出租车出行频繁，在实际行驶过程中通常采用降低采样频率的方式以节省能源消耗及获得合理的响应时间. 因此，不同车辆的位置信息采集时间间隔不同. 本文将滑动窗口内每个时刻设为 1 分钟，确保实时获取大多数移动对象的 GPS 位置点数据. 通过在分布式平台上各个远程节点以固定时间间隔连续输入轨迹数据集，模拟海量高速轨迹数据流的产生过程以形成时间窗口数据. 在 TaxiShanghai13 数据集上评估了 OCluDTS 算法的聚类有效性以及执行开销. 在不指定参数的情况下，算法参数默认设置如下：$\varepsilon=\frac{1}{3}$，$M=16$，$k=300$，$W=5$ min.

5.2 有效性评估

本文在 TaxiShanghai13 数据集上对 OCluDTS 算法进行了聚类效果(聚类演化分析)的测试. 具体采用 2013 年 10 月 9 日上午 8:46～9:05(时间窗口大小为 5 min)时间范围内的上海出租车轨迹数据执行 OCluDTS 算法聚类，获得 4 个不同时刻的聚类结果($k=300$)，其可视化聚类结果能揭示特定时空区域内的车辆密集路段，如图 4(a)～(d)所示. 从实验结果可以看出，OCluDTS 算法能对轨迹数据流进行实时演化聚类分析，4 个不同时刻的聚类结果(以棕红色标注的各簇代表轨迹)能够很好地代表实时交通流，反映了道路网络的持续性.



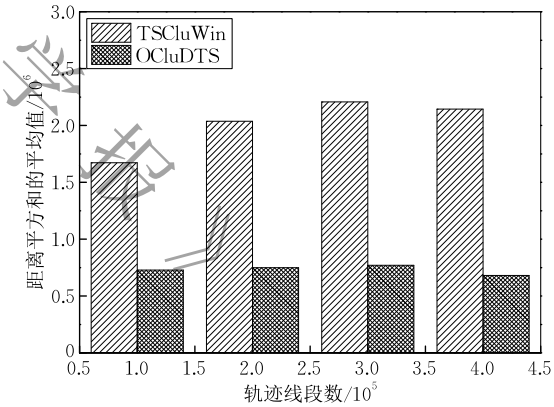
(a) 8:46～8:50 (b) 8:51～8:55

(c) 8:56～9:00 (d) 9:01～9:05

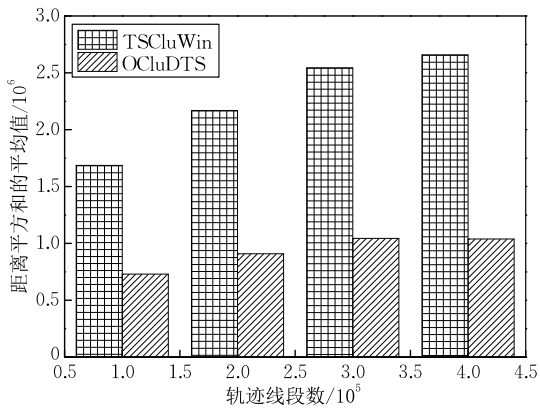
图 4 上海出租车轨迹实时聚类效果

随后，为了验证 OCluDTS 算法的有效性，本文设置不同的窗口大小对 OCluDTS 算法(采用优化策略)的聚类结果与 TSCluWin 算法的微聚类结果进行了对比. 使用平均距离平方和(*AverageSSQ*)作为聚类有效性的评价准则对集中式轨迹流聚类(TSCluWin 算法)与 OCluDTS 算法进行聚类有效性比较. 一般来说，*AverageSSQ* 值越小表示聚类效果越好.

图 5 显示了时间窗口大小分别为 5 min 和 10 min 时随时间推移两种算法的 *AverageSSQ* 值的变化. 由实验结果可以看出，窗口大小对聚类效果影响不大，同时，随着数据的持续流入，OCluDTS 算法的实时聚类效果比 TSCluWin 算法更好. 这是由于 OCluDTS 算法的概要数据结构 TF 在提取各簇角度特征时考虑了簇内较长线段对于代表轨迹的影响，相比 TSCluWin 算法，OCluDTS 算法获得的代表轨迹线段能更好代表各簇的数据分布特征. 此外，TSCluWin 算法在聚类时为了满足给定的聚类数阈



(a) $W=5$ min



(b) $W=10$ min

图 5 聚类有效性比较

值，一旦聚类数设置不合理，原本距离较远的微类可能合并为一簇，导致少数微类边界持续增大（聚类中心不断偏移）降低了算法的总体性能，从而产生较大的 *AverageSSQ* 值. 相比较而言，OCluDTS 算法中，各个远程节点仅对其对应区域内的数据聚类，局部聚类结果相对紧密，即使协调者节点为满足给定聚类数执行合并聚类，也能有效避免将距离较远的轨迹簇合并的情况. 因此上述实验验证了 OCluDTS 算法的有效性.

5.3　性能评估

本文在 TaxiShanghai13 数据集上测试了 OCluDTS 算法的聚类效率以及可伸缩性. 根据处理数据量的变化、节点数的变化以及 k 值的变化，对 OCluDTS 算法的执行时间进行了评测. 首先，本文将 OCluDTS 算法与 TSCluWin 算法进行了处理时间开销的对比. 由于 OCluDTS 算法采用远程节点先并行局部聚类协调者节点再合并聚类的方式，OCluDTS 算法的执行时间包括远程节点的最大并行局部聚类时间、各远程节点向协调者节点传输局部聚类结果的最大时间和协调者节点执行合并聚类的时间.

如图 6 所示，两种算法的处理时间随着数据的不断流入（数据量由 50 000 增加到 400 000）呈线性增长趋势，而 OCluDTS 算法总的时间开销（total time）始终优于 TSCluWin 算法. 此外，远程节点进行局部聚类的最大时间（time to cluster）随着数据量增加轻微地线性增长，但是由远程节点向协调者节点传输局部聚类结果的最大时间（time to send）变化不大（平均 3.8 s）. 这是因为分布式算法只涉及远程节点向协调者节点间传输局部聚类结果（k' 个 DF），其传输数据量大小远远小于轨迹流数据本身，通信开销较小. 因此，OCluDTS 算法的性能相较于 TSCluWin 算法得到大幅度提升，继而验证了 OCluDTS 算法相对于轨迹数据量的可扩展性.

随后，本文通过调整远程节点的个数（$M=4,8,16$）对 OCluDTS 算法随数据持续增加（数据量由 50 000 增加到 1 000 000）时每条记录的处理时间（可伸缩性）进行了评测. 由图 7 可以看出，随着数据的不断流入，OCluDTS 算法每条数据的平均处理时间在 $M=4$ 时为 0.003 26 s，$M=8$ 时为 0.0006 s，在 $M=16$ 时为 0.000 17 s. 这是因为采用远程节点的并行局部聚类以及协调者节点的合并聚类模式，用于聚类处理（计算）的时间开销随远程节点数的增加而减少；与此同时各远程节点只需要向协调者节点传输局部聚类结果，其传输开销并没有随着节点数的增加线性增长. 随着远程节点数增长而增加的通信开销远远小于并行聚类节省的处理开销，因此，算法整体执行效率在远程节点数最多（$M=16$）时表现最好.

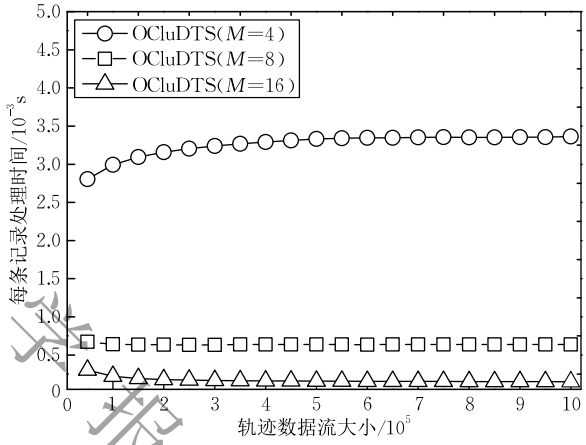


图 7　不同节点数的分布式执行开销

最后，为了验证 OCluDTS 算法的鲁棒性，本文研究了 OCluDTS 算法对聚类数 k 的敏感性. 图 8 显示了在使用不同 k 值（从 200 至 500 变化阈值 k）时执行 OCluDTS 算法（$M=16$）的每条轨迹数据的

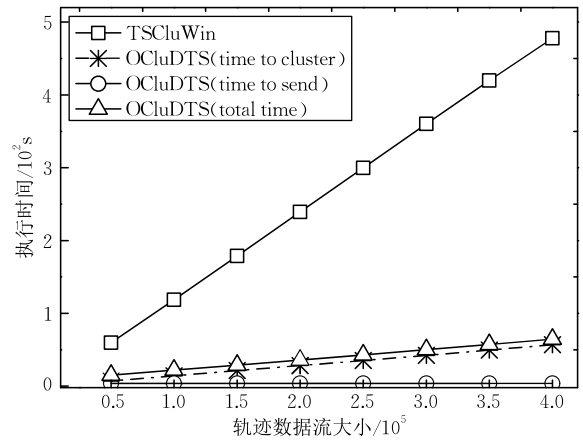


图 6　集中式与分布式算法执行开销比较

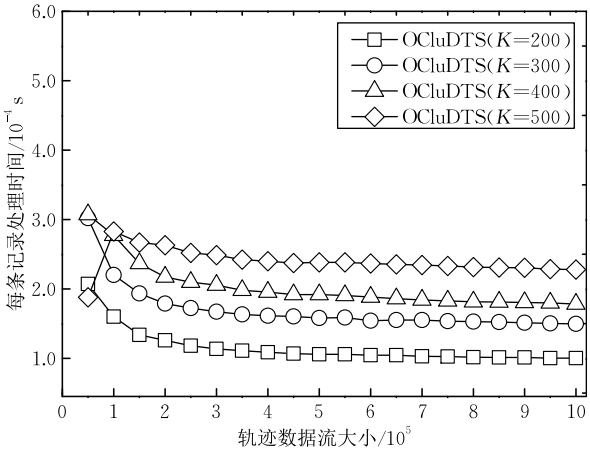


图 8　不同 k 值的分布式执行开销

处理开销.从结果可以看出,随着数据的不断流入(由 50000 到 1000000),OCluDTS 算法在不同 k 值时总的执行开销差别不大,始终保持在 0.0001 s～0.0003 s 之间.随着 k 值增加,OCluDTS 算法的时间开销仅略有增加,在 $k=200$ 时最小(平均 0.000115 s),$k=500$ 时稍高(平均 0.00024 s).因此,得益于较低的通信开销,当远程节点数一定($M=16$)时,k 值的增长对 OCluDTS 算法影响不大.

综合有效性与性能评估实验,本文所提 OCluDTS 方法能够以较小时间开销处理大规模分布式轨迹流的聚类问题,同时获得比集中式轨迹流聚类算法更好的精度提升.

6 结束语

本文以减少通信开销提高分布式轨迹流聚类效率为目标,基于 Spark Streaming 计算框架首次提出了一个两阶段的分布式轨迹流在线聚类算法 OCluDTS,解决了地理上分散采集轨迹流数据的高效聚类问题.在 OCluDTS 算法中,各远程节点负责增量维护其采集获得的局部轨迹流的概要数据结构.通过各远程节点并行聚类局部轨迹流以及协调者节点合并聚类的方式,保证分布式聚类获得与集中式聚类相同的精度.此外,仅限于聚类更新的远程节点传输聚类结果给协调者节点以及协调者节点相似性计算剪枝等优化策略,进一步降低了 OCluDTS 算法的总执行开销.理论分析与基于真实轨迹数据集的实验结果证明了 OCluDTS 算法的高效性以及可扩展性.今后的工作包括研究根据数据集自适应调整滑动窗口大小以及进一步减少节点间的通信开销等,确保分布式轨迹流聚类的有效性和实时性.

参 考 文 献

[1] Pan Bei, Zheng Yu, Wilkie D, Shahabi C. Crowd sensing of traffic anomalies based on human mobility and social media//Proceedings of the 21st SIGSPATIAL International Conference on Advances in Geographic Information Systems. Orlando, USA, 2013: 334-343

[2] Liu Hui-Ping, Jin Che-Qing, Zhou Ao-Ying. Popular route planning with travel cost estimation//Proceedings of the 21st International Conference on Database Systems for Advanced Applications. Dallas, USA, 2016: 403-418

[3] Zheng Bo-Long, Su Han, Zheng Kai, Zhou Xiao-Fang. Landmark-based route recommendation with crowd intelligence. Data Science and Engineering, 2016, 1(2): 86-100

[4] Duan Xiao-Yi, Jin Che-Qing, Wang Xiao-Ling, et al. Real-time personalized taxi-sharing//Proceedings of the 21st International Conference on Database Systems for Advanced Applications. Dallas, USA, 2016: 451-465

[5] Wu Hao, Tu Chuan-Chuan, Sun Wei-Wei, et al. GLUE: A parameter-tuning-free map updating system//Proceedings of the 24th ACM International Conference on Information and Knowledge Management. Melbourne, Australia, 2015: 683-692

[6] Li Zhen-Hui, Lee Jae-Gil, Li Xiao-Lei, Han Jia-Wei. Incremental clustering for trajectories//Proceedings of the 15th International Conference on Database Systems for Advanced Applications. Tsukuba, Japan, 2010: 32-46

[7] Roh Gook-Pil, Hwang Seung-Won. NNcluster: An efficient clustering algorithm for road network trajectories//Proceedings of the 15th International Conference on Database Systems for Advanced Applications. Tsukuba, Japan, 2010: 47-61

[8] Lee Jae-Gil, Han Jia-Wei, Whang Kyu-Young. Trajectory clustering: A partition-and-group framework//Proceedings of the ACM SIGMOD International Conference on Management of Data. Beijing, China, 2007: 593-604

[9] Gaffney S, Smyth P. Trajectory clustering with mixtures of regression models//Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. San Diego, USA, 1999:63-72

[10] Han B, Liu L, Omiecinski E. Road-network aware trajectory clustering: Integrating locality, flow, and density. IEEE Transactions on Mobile Computing, 2015, 14(2): 416-429

[11] Mao Jia-Li, Song Qiu-Ge, Jin Che-Qing, et al. TSCluWin: Trajectory stream clustering over sliding window//Proceedings of the 21st International Conference on Database Systems for Advanced Applications, Dallas, USA, 2016: 133-148

[12] Costa G, Manco G, Masciari E. Dealing with trajectory streams by clustering and mathematical transforms. Journal of Intelligent Information Systems, 2014, 42(1): 155-177

[13] Yu Yan-Wei, Wang Qin, Wang Xiao-Dong, et al. Online clustering for trajectory data stream of moving objects. Computer Science and Information Systems, 2013, 10(3): 1293-1317

[14] Zhou Ao-Ying, Cao Feng, Yan Ying, et al. Distributed data stream clustering: A fast EM-based approach//Proceedings of the 23rd International Conference on Data Engineering. Istanbul, Turkey, 2007: 736-745

[15] Cormode G, Muthukrishnan S, Zhuang Wei. Conquering the divide: Continuous clustering of distributed data streams//Proceedings of the 23rd International Conference on Data Engineering. Istanbul, Turkey, 2007: 1036-1045

[16] Zhang Qi, Liu Jin-Ze, Wang Wei. Approximate clustering on distributed data streams//Proceedings of the 24th International Conference on Data Engineering, Cancún, México, 2008: 1131-1139

[17] Yin Jie, Gaber M M. Clustering distributed time series in sensor networks//Proceedings of the 8th IEEE International Conference on Data Mining. Pisa, Italy, 2008: 678-687

[18] Jensen C S, Lin Dan, Ooi B C. Continuous clustering of moving objects. IEEE Transactions on Knowledge and Data Engineering, 2007, 19(9): 1161-1174

[19] Li Yi-Fan, Han Jia-Wei, Yang Jiong. Clustering moving objects //Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Seattle, USA, 2004: 617-622

[20] Deng Ze, Hu Yang-Yang, Zhu Mao, et al. A scalable and fast OPTICS for clustering trajectory big data. Cluster Computing, 2015, 18(2): 549-562

[21] Zhang Zhi-Gang, Wang Yi-Lin, Mao Jia-Li, et al. DT-KST: Distributed top-k similarity query on big trajectory streams// Proceedings of the 22nd International Conference on Database Systems for Advanced Applications. Suzhou, China, 2017: 199-214

[22] Galic Z, Meskovic E, Osmanovic D. Distributed processing of big mobility data as spatio-temporal data streams. GeoInformatica, 2017, 21(2): 263-291

[23] Datar M, Gionis A, Indyk P, Motwani R. Maintaining stream statistics over sliding windows//Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms, San Francisco, USA., 2002: 635-644

MAO Jia-Li, born in 1979, Ph. D., associate professor. Her research interests include location based services and spatio-temporal data mining.

CHEN He, born in 1992, M. S. candidate. His research interests include mobile data management and encrypted database.

SONG Qiu-Ge, born in 1992, M. S. candidate. Her research interests include data mining and location-based services.

JIN Che-Qing, born in 1977, Ph. D., professor. His main research interests include streaming data management, location-based services, uncertain data management, data quality, and database benchmarking.

ZHOU Ao-Ying, born in 1965, Ph. D., professor. His research interests include data management for data-intensive computing, memory cluster computing.

Background

This paper mainly focuses on the research on distributed trajectory stream clustering upon sliding window model. As a typical class of moving pattern discovery approach, clustering aims to group a large amount of trajectories into some comparatively homogeneous clusters to extract the representative paths or common moving trend shared by various objects. Although distributed data stream clustering and clustering on static trajectory data set have been studied extensively in the past decade, little effort has been devoted to dealing with distributed trajectory streams. The main challenge comes from the strict space- and time-complexities of processing the continuously arrived trajectory data on distributed remote sites, combined with the difficulty of concept drift. To address this issue, we present a novel synopsis structure (DF) to extract clustering characteristic of local trajectory stream on each remote site, and develop an incremental algorithm for online clustering distributed trajectory streams (called OCluDTS). It contains a local clustering phase on multiple remote sites and a global clustering phase on a coordinate site. During the process of local clustering, each remote site aims to cluster and summarize the most recent trajectory line segments of its received local stream at each time instant. Subsequently, after the coordinate site receives the local clustering results of all the remote sites, it will merge them using a certain cluster method to obtain the global clustering result, that is, a set of DF.

Since OCluDTS approach clusters new arrived trajectories while eliminating the obsolete records, it supports continuously clustering massive amounts of trajectory data collected by geographical disperse sites. We conduct comprehensive experiments on the real taxi trajectory dataset. The experimental results show that our proposal not only attains the same accuracy as the centralized trajectory stream clustering method, but also has good scalability and efficiency.

This paper offers an extensive view on clustering distributed streaming trajectories, and also provides a new perspective that will be helpful to the future researches in this field.

This Work is supported by the NSFC (Nos. 61702423, 61370101, 61532021, U1501252, U1401256 and 61402180), the Natural Science Foundation of the Education Department of Sichuan Province (No. 17ZA0381), the Special Foundation of National Programme Cultivation (No. 16C005) and the Meritocracy Research Funds of China West Normal University (No. 17YC158).