基于密度峰值的轨迹聚类算法

刘曾超前[1,2] 许光銮[1]
(1. 中国科学院空间信息处理与应用系统技术重点实验室 北京 100190;2. 中国科学院大学 北京 100049)

摘 要:基于分割聚类框架的TRACLUS算法是轨迹聚类领域中具有代表性的方法。但TRACLUS在中心线两侧轨迹点偏离较大时,无法找到最优的分割点,同时又依赖于输入参数的精细调整。针对这些不足,该文提出一种新的基于密度峰值的轨迹聚类算法(trajectory clustering based on density peaks, TCDP)。TCDP包含两个步骤,首先,利用提升的基于最小描述长度的分割算法,将轨迹分割为子轨迹。通过引入平行夹边实现前向探测地分割,提高轨迹分割的准确性。其次,基于子轨迹聚类中心具有较高的局部密度并被低密度的子轨迹所围绕,而不同聚类中心之间存在较远距离的思想,实现了基于密度峰值的子轨迹聚类,以此增强算法对输入参数的鲁棒性。TCDP解决了TRACLUS算法的不足。实验结果表明,TCDP具有更好的轨迹聚类效果。

关键词:轨迹;子轨迹;聚类;分割;密度峰值
中图分类号: TN0 **文献标识码**:A **国家标准学科分类代码**: 510.4030

Trajectory clustering algorithm based on density peaks

Liu Zengchaoqian[1,2] Xu Guangluan[1]
(1. Key Laboratory of Technology in Geo-spatial Information Processing and Application System, Institute of Electronics, Chinese Academy of Sciences, Beijing 100190, China;2. University of Chinese Academy of Sciences, Beijing 100049, China)

Abstract: Trajectory clustering is a useful approach to analyze trajectory data. Numbers of methods have been proposed in this field, of which TRACLUS is a representative trajectory clustering algorithm based on the partition-and-group framework. However, TRACLUS fails to find the optimal partitioning when trajectory points falling on both side of the center line are far away from the center line and is sensitive to the input parameters. Aiming at those vulnerabilities, a new trajectory clustering algorithm TCDP is proposed in this paper. TCDP is composed of two steps. In the first step, trajectories are partitioned into sub-trajectories using the improved MDL partitioning algorithm which is more applicative and has higher precision than the partitioning algorithm used in TRACLUS. In the second step, a new sub-trajectories algorithm is proposed. It is based on the idea that sub-trajectory centers are surrounded by lower density sub-trajectories and far from other sub-trajectory centers. The new sub-trajectories algorithm is robust to input factor. TCDP resolves the defects TRACLUS having. Meanwhile, the experiment shows TCDP performs better in quality of trajectory clustering.

Keywords: trajectory; sub-trajectory; clustering; partitioning; density peak

1 引 言

随着技术的发展,尤其是定位设备的广泛应用,人们积累了大量的轨迹数据,诸如动物迁徙[1],物体运动[2],人类行为[3-4],市场营销[5]等。在这些轨迹数据中,隐含着丰富的信息,在科学领域、商业领域和政府管理领域都极具价值[6]。分析轨迹数据并提取其中所蕴含的有用信息具有重大现实意义。通常,轨迹用一连串包含时空信息的多属性点序列来表示。即 $TR=\{P_1,P_2,P_3,\cdots,P_{\text{len}},\cdots\}$,这里 $P_j(1\leqslant j\leqslant len)$ 是指轨迹点。而子轨迹是指整条轨迹中的部分连续轨迹点构成的子序列。

轨迹聚类源自于聚类分析,用于提取复杂轨迹数据中的潜在信息[7-8]。这一领域中,前人多将整条轨迹作为聚类的最小单元[9]。这样能保留轨迹的整体相似性,但会丢

收稿日期:2017-02

失轨迹中微观信息。在2007年，Lee等人[10]提出了分割聚类框架，并提出基于该框架的一种新轨迹聚类算法TRACLUS。该算法将轨迹聚类的处理单元缩小到子轨迹，充分保留轨迹中的微观信息，得到了广泛应用[11-12]。

TRACLUS包含2个步骤：分割和聚类。在分割步骤中，轨迹被分割为子轨迹段；而在聚类步骤中，相似的子轨迹段被聚为一类。研究这两步后发现，该算法中存在着两个不足。1)在轨迹分割时，当轨迹中心线两侧轨迹点偏离较远时，TRACLUS不能找到最优分割点。2)在子轨迹聚类时，TRACLUS对输入参数十分敏感，参数的轻微波动会导致聚类结果差异巨大。

针对TRACLUS中的2个不足，本文提出了相应的改进方案，形成了一种新的子轨迹聚类算法TCDP。在轨迹分割时，提出了改进的基于最小描述长度(MDL)的分割方法，该分割方法通过引入平行夹边，实现了分割时的前向探测，提升了分割精度；在轨迹聚类时，本文引入了一种基于密度峰值的聚类算法，减少了算法对参数的依赖性。

2 相关工作

轨迹聚类是基于一定准则将相似的轨迹聚为一类的过程。聚类准则通常由不同的特征来进行度量。不同的准则下，轨迹的聚类结果差异明显。轨迹聚类已经发展多年，总结来说，轨迹聚类可以分为以下4类。

1)基于模型的轨迹聚类：该类方法对部分或者整个轨迹数据集进行建模，并通过对模型拟合参数的不同来代表轨迹的轨迹类别。其代表方法由Chamroukhi等人[13]提出。他们首先构建了一种受限于隐马尔可夫链的多项式回归模型，然后通过最大似然法确定轨迹相对于模型中簇的隶属度，从而实现轨迹聚类。然而，该类算法[13-14]建模以及确定拟合度较为困难。

2)基于距离的轨迹聚类：在这类轨迹聚类算法中，轨迹通常被转化为多维度的特性向量。通过转换，轨迹聚类问题简化为对特征向量的聚类问题。通过选择不同的聚类方法(例如k-means)以及不同的距离度量(例如欧式距离)，轨迹被聚为多个类。基于距离的轨迹聚类方法[15-16]易于理解，但轨迹的复杂性使得该类方法的应用受到限制，距离的度量方式也会对结果造成极大影响。

3)基于微聚类的轨迹聚类算法：该类轨迹聚类算法通过识别移动微簇的运动情况来进行轨迹聚类。其典型算法由Hwang等人[17]提出，他们通过找寻所有轨迹距离很近时的最长时间段来实现轨迹聚类。该算法可以检测对象随时间变化的移动规律，并能找出运动的重要时间段。文献[18]也是该类算法的典型代表。但该类算法存在查找时间段困难和算法复杂度高的缺点。

4)基于密度的轨迹聚类：这种聚类算法通过密度阈值来区分有价值数据和噪声，常以DBSCAN和OPTICS为基础算法。ST-DBSCAN[19]是该类算法的代表算法，其可以通过不同的维度值来发现轨迹簇。该算法首先通过阈值确定核心对象，进而通过核心对象进行类扩展，进而得到多个类簇。文献[20-21]也是该类算法中极具代表性的方法。通常情况下，密度阈值的选取对该类聚类算法影响较大。

上述轨迹聚类方法中，大多将整个轨迹视为最小处理单元，子轨迹的相似性被丢弃。然而，在某些情况下，子轨迹的相似性可能更具价值。TRACLUS算法的出现解决了该问题。该算法是一种基于密度的轨迹聚类算法，包含分割和聚类两个步骤。轨迹分割时，TRACLUS采用了MDL准则。但中心线两侧轨迹点偏离较大时，算法无法找到最优分割点。原作者指出，该算法的平均精度约为80%[10]。子轨迹聚类时，聚类算法由DBSCAN演变而来。该算法需要两个参数：半径阈值ε和最小线段数阈值$\min lns$。根据这两个阈值，算法将线段聚类到不同的类别中。聚类过程对参数十分敏感。

3 TCDP算法

本节将详细讨论TCDP。TCDP算法如算法1所示。该算法采用与TRACLUS相同的框架，但基于该框架的具体算法与TRACLUS有所不同，尤其是TCDP将采用全新的聚类算法。第3.2节和第3.3节将分别介绍TCDP的具体步骤。

算法1 TCDP

输入：轨迹集合 $TRs=\{TR_1,TR_2,\cdots,TR_n\}$
输出：聚类集合 $SCs=\{SC_1,SC_2,\cdots,SC_m\}$
算法：
/*第一步：轨迹分割*/
01：使用改进的MDL分割算法分割每条轨迹
02：获得子轨迹数段集合
/*第二步：子轨迹聚类*/
03： 使用基于密度峰值的子轨迹聚类算法对子轨迹聚类
04： 得到子轨迹聚类集合
05： 返回子轨迹聚类集合

3.1 相关定义

定义1：子轨迹段之间的距离是指子轨迹段之间的垂直距离($d_\perp$)、平行距离($d_\parallel$)和角度距离(d_θ)之和。该定义借用文献[10]的定义，其计算公式如式(1)所示。其中ω为3个距离系数，在本文中，其均取1。

$$d_{ij}=\omega_\perp d_\perp(L_i,L_j)+\omega_\parallel d_\parallel(L_i,L_j)+\omega_\theta d_\theta(L_i,L_j) \tag{1}$$

$$d_\perp(L_i,L_j)=\frac{l_{\perp 1}^2+l_{\perp 2}^2}{l_{\perp 1}+l_{\perp 2}} \tag{2}$$

$$d_\parallel(L_i,L_j)=MIN(l_{\parallel 1},l_{\parallel 2}) \tag{3}$$

$$d_\theta(L_i,L_j)=\begin{cases}\|L_j\|\times\sin\theta, 0^\circ\leqslant\theta<90^\circ\\ \|L_j\|, 90^\circ\leqslant\theta\leqslant180^\circ\end{cases} \tag{4}$$

TCDP的分割算法由TRACLUS中的分割算法改进

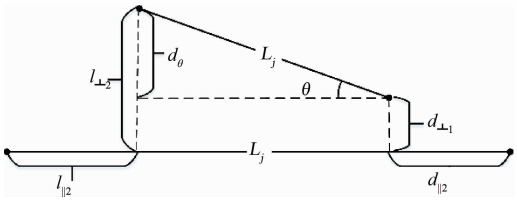


图1 距离计算示意图

而来，也是一种基于 MDL 准则的分割算法。该算法中，需计算 $L(H)+L(D|H)$ 的值，其计算公式如式(5)和(6)所示。

$$L(H)=\sum_{j=1}^{\mathrm{par}_i-1}\log_2(len(p_{c_j}p_{c_{j+1}})) \tag{5}$$

$$L(H)=\sum_{j=1}^{\mathrm{par}_i-1}\sum_{k=c_j}^{\mathrm{par}_{c_{j+1}}-1}\log_2(d_\perp(p_{c_j}p_{c_{j+1}},p_kp_{k+1}))+\log_2(d_\theta(p_{c_j}p_{c_{j+1}},p_kp_{k+1})) \tag{6}$$

式中：len 指两点之间距离，$d_\perp$ 指垂直距离，d_θ 指角度距离，p_k 为轨迹点，p_{c_j} 为分割点，式中 $c_j\leqslant k\leqslant c_{j+1}-1$。

TCDP 的聚类算法为基于密度峰值的子轨迹聚类算法。该子轨迹聚类算法的基础是快速搜索高密点的聚类方法[22]。其被提出后得到了广泛应用[23-24]。该算法是一种基于密度的聚类方法。主要思想是聚类中心点的局部密度将比其环绕点的局部密度高，并且聚类中心点之间的距离比较远。该算法首先计算两个重要特征：每个点的局部密度 ρ_i 及与比其密度高的点的距离 δ_i。这两个因素仅取决于数据点 P_i 和 P_j 之间的距离 d_{ij}。局部密度 ρ_i 计算为：

$$\rho_i=\sum_j x(d_{ij}-dc) \tag{7}$$

式中：如果 $y>0$，则 $X(y)=1$，否则 $X(y)=0$，式中的 dc 是指截断距离。δ_i 是距离比它局部密度高的点的最小距离，其计算为：

$$\delta_i=\min_{j:\rho_j>\rho_i}(d_{ij}) \tag{8}$$

具有最大局部密度的点的距离定义为 $\delta_i=\max_j(d_{ij})$。具有较大局部密度和较大距离的点被认为是聚类中心，其他点将被聚类到离它最近的比它局部密度高的点所在的类簇中。

为了使该聚类思想能应用于子轨迹聚类，部分定义需重写：

定义 2：子轨迹的局部密度 ρ_i 指在截断距离内，其他子轨迹的数目，其计算公式如式(9)所示。

$$\rho_i=\sum_{\mathrm{LS}_j\in(\mathrm{LS}_i-\mathrm{LS}_i)}\exp-\frac{(d_{ij})^2}{dc} \tag{9}$$

式中：dc 是截断距离。该计算式和文献[22]中对应的公式不同，但该计算方法仍能描述截断距离内其他子轨迹的数量，同时该值为连续值，能有效避免产生相同的密度数值，方便后续处理。

定义 3：子轨迹段的距离 δ_i 是指子轨迹段到距离比它局部密度更高的子轨迹段的距离的最小值。其计算如式(10)所示。

$$\delta_{q_i}=\begin{cases}\min\{d_{q_iq_j}\},i\geqslant 2\\ \max\{\delta_{q_i}\},i=1\end{cases} \tag{10}$$

在该式中 $\{q_i\}_{i=1}^N$ 是 $\{\rho_i\}_{i=1}^N$ 的一个降序排列，其满足 $\rho_{q_1}\geqslant\rho_{q_2}\geqslant\cdots\geqslant\rho_N$。

定义 4：子轨迹聚类中心 $\{CC_i\}_{i=1}^k$ 是指具有较大的 δ 和 ρ 的子轨迹段的集合，其中，k 是指该集合的数目。该集合是产生子轨迹类簇的核心。

定义 5：子轨迹噪声为具有较大 ρ 和较小 δ 的子轨迹段。在本文中，如果某子轨迹的局部密度小于平均局部密度且其距离大于平均子轨迹距离，则该子轨迹被认为是噪声子轨迹。

子轨迹聚类时，其关键是找到轨迹聚类中心 $\{CC_i\}_{i=1}^k$。为了方便选取聚类中心，定义了一个新的指标——密距积。

定义 6：密距积 DMD_i 指密度与距离的乘积，用 DMD 表示。由于局部密度和距离两者的尺度相差较大，因此还需对两者进行归一化。于是密距积的计算如式(11)所示。

$$\mathrm{DMD}_i=\left(\frac{\delta_i}{\max(\delta)}\right)\times\left(\frac{\rho_i}{\max(\rho)}\right) \tag{11}$$

密距积同时考虑了局部密度和距离的影响，当且仅当局部密度和距离都具有较大值时，密距积的值才会比较大。即密距积较大者可以被作为子轨迹聚类中心。对密距积进行排序后，由于数据的分布特点，密距积的值会在某个位置有一个明显的跳变，通过该跳变即可确定子轨迹聚类中心集。

在该子轨迹聚类算法中，仅需要一个参数。轨迹聚类时所依赖的属性由数据本身所决定，输入参数对这两个属性影响较小。因此，从理论分析上来说，该输入参数对最终的结果影响比较小。该子轨迹聚类算法对输入参数具有鲁棒性。

3.2 轨迹分割

TCDP 中的分割算法由 TRACLUS 中所提出的分割算法改进而来，是本文的主要创新之一。其具体描述如算法 2 所示。

TRACLUS 中的分割算法是一种近似的 MDL 算法，该算法将局部最优解作为全局最优解，以达到提高算法效率的目的。算法中，当两个点 P_i 和 $P_j(i<j)$ 是这两点之间所有点中仅有的两个分割点时，用 $MDL_{par}(P_i,P_j)$ 代替 MDL 中的 $L(H)+L(D|H)$；而当 P_i 和 P_j 之间(包括这两点)没有任何分割点时，用 $MDL_{nopar}(P_i,P_j)$ 代替。在 P_iP_j 之间，最长轨迹分割将满足 $MDL_{\mathrm{par}}(P_i,P_j)\leqslant MDL_{\mathrm{nopar}}(P_i,P_k)$，其中 $i<k\leqslant j$。通过该思路，可以较快地找到局部最优解。然而，该算法在图 2 所示情况下将失效。图中，$MDL_{\mathrm{par}}(P_1,P_8)\leqslant MDL_{\mathrm{nopar}}(P_1,P_8)$。故在该段中，分割点应在 P_8 以后。但算法将 P_3 和 P_8 作为分割点，原因在于 P_4 和 P_6 处满足 $MDL_{\mathrm{par}}(P_1,P_4)>$

$MDL_{nopar}(P_1,P_4)$ 和 $MDL_{par}(P_3,P_6) > MDL_{nopar}(P_3,P_6)$ 此时，该分割算法无法找到最优分割点。

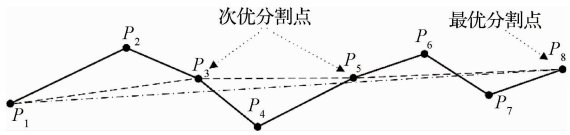


图 2　TRACLUS 中分割算法失效示意图

为了解决 TRACLUS 中分割算法的缺陷，本文提出了改进的 MDL 分割算法。在改进的 MDL 分割算法中，最主要的变化是引入了平行夹边，实现了前向探测。改进的 MDL 算法的出发点是分割后的子轨迹段能被平行夹边所包裹。因此，在分割轨迹时，通过平行夹边可以实现分割点检测时的前向检测。在图 2 所示的情况下，在检测轨迹点 P_3 时，改进的 MDL 分割算法首先通过点 P_1 和 P_3，构建一个平行夹边，如图 3 所示，然后找到在该平行夹边内的点 P_1，P_5，P_8。在平行夹边内的点中，找到第一个满足分割条件的点的前一个点，下一次检索的点为该点的后一点。如果所有点都不满足分割条件（如图中的 P_8），则下一次检索的点为最后一个点的下一点。平行夹边的宽度通过内部设置，该值与用来限定最短子轨迹长度的值一致。改进的 MDL 分割算法的描述如算法 2 所示。

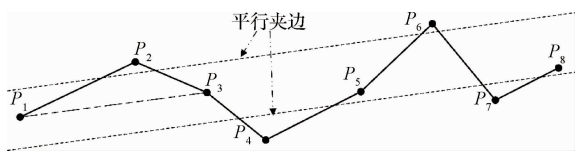


图 3　改进的 ML 分割算法示意图

算法 2 改进的 MDL 分割算法

输入：一条轨迹 $TR = \{P_1,P_2,P_3,\cdots,P_N\}$
输出：子轨迹段 $LSs = \{LS_1,LS_2,\cdots,LS_m\}$
算法：
01：将 P_1 设为起点，将检测点设置为 P_2
02：当检测点不是 P_N 时：
03：　当前检测点是否为分割点，如果是，则建立新 LS 并加入到 LSs，将起始点设置为当前检测点，检测点设置为当前检测点下一点，返回到 02；如果不是则继续。
04：　通过起始位置和检测点建立平行夹边
05：找到位于平行夹边内的所有点，记为 totalPoints
06：找到 totalPoints 第一个满足分割条件的点的前一点，并将检测点改为该点后一点，返回到 02
07：如果 totalPoints 都不满足分割条件，将检测位置改为 totalPoints 中最后一点的下一点，返回到 02。
08：通过起始点与 P_N 建立新 LS 并加入 LSs
09：返回 LSs

3.3　基于密度峰值的子轨迹聚类

基于密度峰值的子轨迹聚类算法的详细描述如算法 3。在算法输入一个参数和子轨迹集后，算法将计算每条子轨迹的局部密度，得到 $\{\rho_i\}_{i=1}^m$。然后，算法对 $\{\rho_i\}_{i=1}^m$ 进行排序，根据排序后的局部密度值，算法进一步计算每条子轨迹的距离，得到 $\{\delta_i\}_{i=1}^m$。通过得到的 $\{\rho_i\}_{i=1}^m$ 和 $\{\delta_i\}_{i=1}^m$，进一步计算 $\{DMD_i\}_{i=1}^k$ 并排序，确定聚类个数，提取出聚类中心。下一步则是通过每条子轨迹与各个聚类中心的距离，将子轨迹聚到不同的类别中。在该过程中，噪声子轨迹将被直接剔除。聚类完成后，输出结果。

算法 3 基于密度峰值的子轨迹聚类算法

输入：子轨迹集 $LSs = \{LS_1,LS_2,\cdots,LS_m\}$
　　截断距离 dc
输出：聚类集合 $SCs = \{SC_1,SC_2,\cdots,SC_k\}$
算法：
01：计算 $\{\rho_i\}_{i=1}^m$
02：对 $\{\rho_i\}_{i=1}^m$ 排序
03：计算 $\{\delta_i\}_{i=1}^m$
04：计算 $\{DMD_i\}_{i=1}^m$
05：对 $\{DMD_i\}_{i=1}^m$ 排序
06：确定聚类个数 k
07：提取聚类中心 $\{CC_i\}_{i=1}^k$
08：对于每条 LS_i：
09：　当其不是噪声时：
将聚类到离它最近的局部密度比它高的聚类中心所在的类。
10：　返回到 06
11：返回 SCs

3.4　启发式的参数选取

TCDP 需要一个输入参数，即截断距离。本文采用熵理论来选择截断距离值。熵是对象不确定性的度量，当所有事件的可能性相同时，熵的值最大。当聚类效果最差时，子轨迹 LS_i 截断距 dc 内的其他轨迹数目 $|N_{dc}(LS_i)|$ 将趋向一致，熵将取得最大值。在聚类效果较好时，熵的值最小。本文中通过式(12)来定义子轨迹熵。通过观察熵随截断距离的变化趋势，来选择截断距离值。

$$H = \sum_{i=1}^{N} p(x_i)\log_2\frac{1}{p(x_i)} \tag{12}$$

式中：$p(x_i) = \dfrac{|N_{dc}(x_i)|}{\sum_{j=1}^{n}|N_{dc}(x_j)|}$，$N$ 是指子轨迹的总数。

4　实验评估

在实验过程中，本文采用了两个真实数据集，鹿数据集和麋鹿数据集。这两个数据集来自美国的斯塔基项目（Starkey project）。该项目使用无线电遥测部分动物如鹿，麋鹿，牛等的相关数据，时间跨度为 1993 年～1996 年。本文采用 1995 年的鹿数据和 1993 年的麋鹿数据进行实验。1995 年的鹿数据集有 32 条轨迹 20 065 个轨迹点；而 1993 年的麋鹿数据集包含 33 条轨迹共计 47 204 个轨迹点。

4.1　评估改进的 MDL 分割算法

在轨迹分割阶段，本文改进了 TRACLUS 中的轨迹分割算法，提出改进的 MDL 分割算法。用原算法和改进

后的算法在数据集上进行了实验，比较了分割算法的精确性和简明性。定义总平均距离(TAD)来度量精确性。该指标通过每个轨迹点与对应子轨迹之间的距离描述轨迹点与子轨迹的差异。其计算如式(13)所示。

$$\mathrm{TAD}=\frac{1}{N}\cdot\sum_{i=1}^{N}\sum_{LS_i\in TR,P\in LS_j}\sum dis(P,LS_j) \tag{13}$$

式中：N 是总的子轨迹数目，$dis(P,LS_j)$ 是指轨迹点 P 与其所在的子轨迹段 LS_i 之间的距离。本文采用子轨迹段的数目描述分割简明性。实验过程中，选取了不同的用来控制子轨迹段长短的参数进行实验。该参数越大，子轨迹段的长度将越长，子轨迹总数将越小。图4所示为在两个数据集上的实验结果。从图中可以看到，在相同的子轨迹段数下，改进的MDL分割算法比原算法有更小的TAD值，即分割的准确性更高。在相同参数下，改进的MDL分割算法与原分割算法对应的子轨迹段的数目如表1所示。改进后的分割算法有更少的子轨迹段数目。因此，改进后的分割算法的简明性更好。综合考虑精确性和简明性后可以得出，改进后的分割算法有更好的效果。

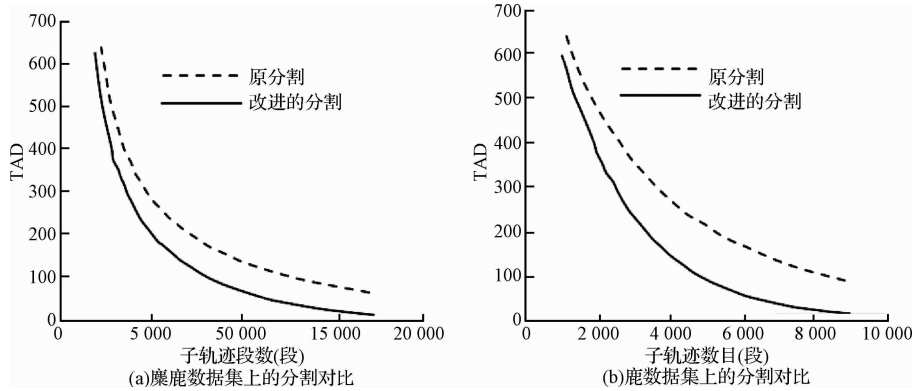


图4 TRACLUS和TCDP中分割算法实验结果

表1 鹿数据集上相同参数下的子轨迹段的数目

参数	TRACLUS中分割算法	TCDP中分割算法	子轨迹变化/%
5	13 809	7 948	−42.4
15	6 134	4 371	−28.7
25	3 539	2 864	−19.1
35	2 483	2 142	−13.7

4.2 子轨迹聚类算法评估

TCDP中的子轨迹聚类算法需要输入一个参数，用以确定计算局部密度的范围。文中提出了一种启发式的确定该参数的方法。通过该方法得到了如图5所示的结果。观察图可以看到，在麋鹿数据集上，最佳的截断距离为25；而在鹿的数据集上，最佳的截断距离值为26。

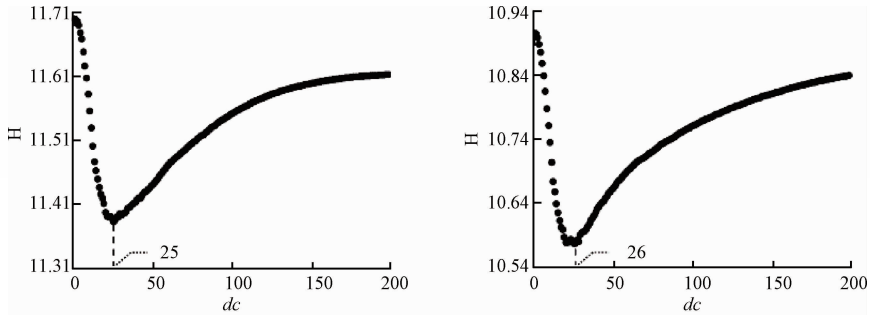
TCDP中，确定聚类的数目需要借助DMD值。算法通过DMD值的突变位置确定聚类数目。将这一过程可视化出来时，得到如图6所示的可视化表示图。图6是麋鹿数据在截断距离为25时，对应的DMD数据排序图和密度距离图。从DMD图中可以看到，麋鹿数据集中，密度和距离值都较大的子轨迹段有7段(对应密度距离图中7个三角形)，故该数据集存在7个类。

为了验证算法对输入参数具有鲁棒性，在不同的截断距离下进行了子轨迹聚类。在麋鹿的数据集上，dc 的取值范围为19～38，而在鹿的数据集上，dc 的取值范围为12～45。在不同的截断距离下，分别记录了对应的聚类数目，其实验数据如图7所示。在TRACLUS算法中，采用不同的邻域半径 ε，聚类的数目变化极大。而采用TCDP算法时，聚类的数目变化很小。因此，TCDP算法对输入参数具有更好的鲁棒性。

本文采用了总平方差(SSE)与噪声影响(NP)两者之和作为评价聚类结果的指标。该指标在文献[10]中被首次提出并使用，文中采用QMeasure来表示该值，其计算公式如式(14)所示。

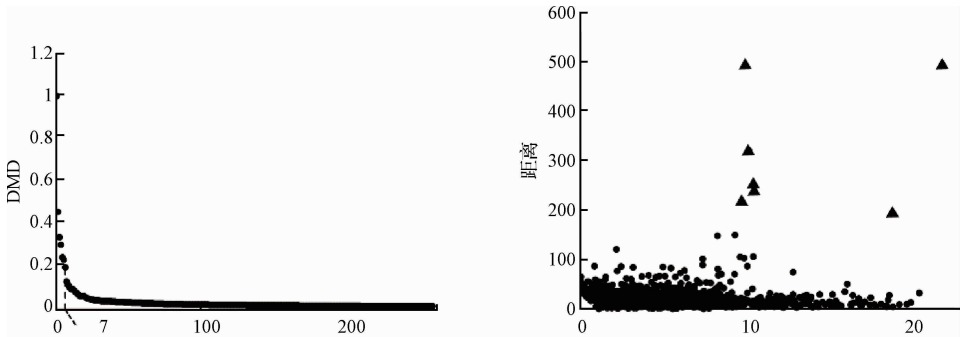
$$QMeasure=SSE+NoisePenalty=\sum_{i=1}^{k}\left(\frac{1}{2\mid C_i\mid}\sum_{x\in C_i}\sum_{y\in C_i}dis\ (x,y)^2\right)+\frac{1}{2\mid N\mid}\sum_{w\in N}\sum_{z\in N}dis(w,z)^2 \tag{14}$$

式中：k 是指聚类的数目，C_i 代表第 i 个聚类，N 代表噪声子轨迹的数目，$dis(x,y)$ 代表两条子轨迹之间的距离。QMeasure值越小，意味着聚类的结果越好。在各自的参数下的实验结果如图8所示。其中，TRACLUS算法在麋鹿数据集上的取值为 $minlns=9$ 和 $\varepsilon=27$；而在鹿的数据集上的取值为 $minlns=8$ 和 $\varepsilon=29$。这两组值为文献[10]中所提供的参数。TCDP中截断距离的值为前文中通过启发式算法确定的数值。可以看到，TCDP聚类算法的QMeasure值比TRACLUS聚类算法的值要小，意味着TCDP的聚类效果更好。



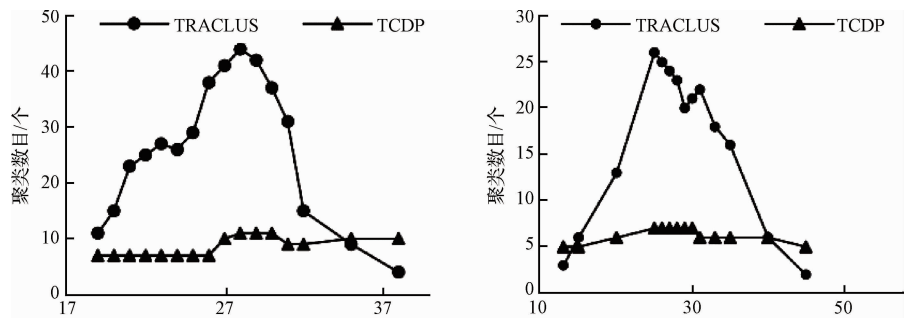
(a) 麋鹿数据集的熵　　(b) 鹿数据集的熵

图 5　两个数据集的熵



(a) DMD可视化图　　(b) 密度距离坐标图

图 6　麋鹿数据聚集上 DMD 排序图与密度距离图(dc=25)



(a) 在麋鹿数据集上的聚类数目对比图　　(b) 在鹿数据集上的聚类数目对比图

图 7　不同参数下聚类数目的变化

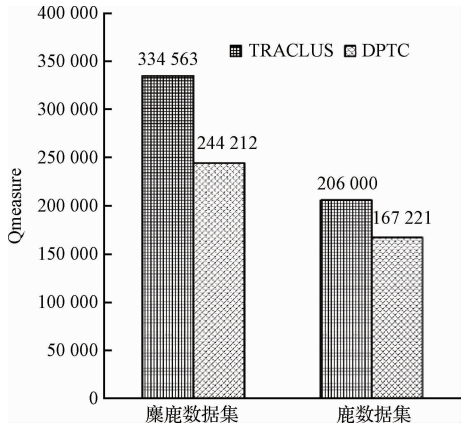


图 8　聚类效果对比

5　结　论

随着轨迹数据的大量积累，轨迹数据挖掘日益重要，轨迹聚类是其中一种有效分析轨迹数据的方法，TRACLUS 作为轨迹聚类领域中应用十分广泛的方法，仍存在两方面不足。即在轨迹分割时，轨迹点偏离中心较大时，无法找到最优解；在子轨迹聚类时，算法对输入参数敏感。针对这两个问题，本文提出一种新的轨迹聚类算法 TCDP。在轨迹分割时，本文提出一种改进的轨迹分割方法，通过引入平行夹边进行前向探测，达到了更好的分割效果。在子轨迹聚类时，本文引入一种基于密度峰值的子轨迹聚类算法，增强算法对输入参数的鲁棒性。实验结果

表明，TCDP改进了TRACLUS中的不足，并达到了更好的效果。

参考文献

[1] 程淑红，刘洁，李雷华. 基于鱼类运动行为的水质异常评价因子研究[J]. 仪器仪表学报，2015，36(8)：1759-1766.

[2] 杨杰，赵敏，苏浩，等. 基于粒子群矢量搜索融合的射流轨迹识别方法[J]. 电子测量与仪器学报，2016，30(5)：803-809.

[3] 丁兵，吴允平，李彬雅. 一种基于C4.5算法的车位识别方法[J]. 电子测量技术，2015，38(8)：64-68.

[4] 肖秦琨，谢艳梅. 融合深度图和三维模型的人体运动捕获[J]. 国外电子测量技术，2015，34(1)：19-22.

[5] SONG A，ZENG H，YANG R，et al. Fundamental problems in rehabilitation robots based on neuro-machine interaction[J]. Instrumentation，2014，1(3)：1-16.

[6] IZAKIAN Z，MESGARI M S，ABRAHAM A. Automated clustering of trajectory data using a particle swarm optimization[J]. Computers Environment & Urban Systems，2016(55)：55-65.

[7] DAHLBOM A，NIKLASSON L. Trajectory clustering for coastal surveillance[C]. 10th International Conference on Information Fusion，2007：1-8.

[8] XU H，ZHOU Y，LIN W，et al. Unsupervised trajectory clustering via adaptive multi-kernel-based shrinkage[C]. IEEE International Conference on Computer Vision，2015：4328-4336.

[9] KISILEVICH S，MANSMANN F，NANNI M，et al. Spatio-Temporal Clustering：A Survey[M]. US：Springer，2009.

[10] LEE J G，HAN J，WHANG K Y. Trajectory clustering：a partition-and-group framework[C]. ACM SIGMOD International Conference on Management of Data，2007：593-604.

[11] BAHBOUH K，WAGNER J R，MORENCY C，et al. Travel demand corridors：Modelling approach and relevance in the planning process[J]. Journal of Transport Geography，2017(58)：196-208.

[12] LEE J G，HAN J，LI X. Trajectory outlier detection：a partition-and-detect framework[C]. IEEE International Conference on Data Engineering，2008：140-149.

[13] CHAMROUKHI F，SAM A，AKNIN P，et al. Model-based clustering with Hidden Markov Model regression for time series with regime changes[C]. International Joint Conference on Neural Networks，2013：2814-2821.

[14] GHASSEMPOUR S，GIROSI F，MAEDER A. clustering multivariate time series using hidden markov models[J]. International Journal of Environmental Research & Public Health，2014，11(3)：2741-2763.

[15] PELEKIS N，KOPANAKIS I，MARKETOS G，et al. Similarity search in trajectory databases[C]. International Symposium on Temporal Representation and Reasoning，2007：129-140.

[16] SANCHEZ I，AYE Z M M，RUBINSTEIN B I，et al. Fast trajectory clustering using Hashing methods[C]. 2016 International Joint Conference on Neural Networks (IJCNN)，2016：3689-3696.

[17] HWANG S Y，LIU Y H，CHIU J K，et al. Mining mobile group patterns：A trajectory-based approach[C]. Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining，2005：713-718.

[18] LI Z，LEE J G，LI X，et al. Incremental clustering for trajectories[C]. International Conference on Database Systems for Advanced Applications，2010：32-46.

[19] BIRANT D，KUT A. ST-DBScan：An algorithm for clustering spatial-temporal data[J]. Data & Knowledge Engineering，2007，60(1)：208-221.

[20] LIU L X，SONG J T，GUAN B，et al. Tra-DBScan：A algorithm of clustering trajectories[J]. Applied Mechanics & Materials，2011，121(126)：4875-4879.

[21] MAI S T，HE X，FENG J，et al. Anytime density-based clustering of complex data[J]. Knowledge and Information Systems，2015，45(2)：319-355.

[22] RODRIGUEZ A，LAIO A. Clustering by fast search and find of density peaks[J]. Science，2014，344(6191)：1492-1496.

[23] SUN K，GENG X，JI L. Exemplar component analysis：A fast band selection method for hyperspectral imagery[J]. Geoscience & Remote Sensing Letters IEEE，2015，12(5)：998-1002.

[24] CHEN Y W，LAI D H，QI H，et al. A new method to estimate ages of facial image for large database[J]. Multimedia Tools and Applications，2016，75(5)：1-19.

作者简介

刘曾超前，1990年出生，工学硕士，主要研究方向为轨迹数据挖掘、机器学习。
E-mail：lzchaoqian@163.com