doi:10.11959/j.issn.1000-436x.2015043

基于轨迹位置形状相似性的隐私保护算法

王超，杨静，张健沛
(哈尔滨工程大学 计算机科学与技术学院，黑龙江 哈尔滨 150001)

摘 要：为了降低轨迹数据发布产生的隐私泄露风险，提出了多种轨迹匿名算法。然而，现有的轨迹匿名算法在计算轨迹相似性时忽略了轨迹的形状因素对轨迹相似性的影响，因此产生的匿名轨迹集合的可用性相对较低。针对这一问题，提出了一种新的轨迹相似性度量模型，在考虑轨迹的时间和空间要素的同时，加入了轨迹的形状因素，可以在多项式时间内计算定义在不同时间跨度上的轨迹的距离，能够更加准确、快速地度量轨迹之间的相似性；在此基础上，提出了一种基于轨迹位置形状相似性的隐私保护算法，最大限度地提高了聚类内部轨迹的相似性，并且使用真实的原始位置信息形成数据“面罩”，满足了轨迹 *k*-匿名，在有效地保护轨迹数据的同时，提高了轨迹数据的可用性；最后，在合成轨迹数据集和真实轨迹数据集上的实验结果表明，本算法花费更少的时间代价，具有更高的数据可用性。
关键词：时空轨迹数据；轨迹数据发布；贪婪聚类；数据面罩；轨迹匿名
中图分类号：TP391.7 **文献标识码**：A

Privacy preserving algorithm based on trajectory location and shape similarity

WANG Chao, YANG Jing, ZHANG Jian-pei
(College of Computer Science and Technology, Harbin Engineering University, Harbin 150001, China)

Abstract: In order to reduce the privacy disclosure risks when trajectory data is released, a variety of trajectories anonymity methods were proposed. However, while calculating similarity of trajectories, the existing methods ignore the impact that the shape factor of trajectory has on similarity of trajectories, and therefore the produced set of trajectory anonymity has a lower utility. To solve this problem, a trajectory similarity measure model was presented, considered not only the time and space elements of the trajectory, but also the shape factor of trajectory. It is computable in polynomial time, and can calculate the distance of trajectories not defined over the same time span. On this basis, a greedy clustering and data mask based trajectory anonymization algorithm was presented, which maximized the trajectory similarity in the clusters, and formed data "mask" which is formed by fully accurate true original locations information to meet the trajectory *k*-anonymity. Finally, experimental results on a synthetic data set and a real-life data set were presented; our method offer better utility and cost less time than comparable previous proposals in the literature.
Key words: spatio-tempporal trajectory data; publication of trajectory data; greedy clustering; data mask; trajectory anonymization

1 引言

移动社会网络的兴起及移动智能终端的发展带来了大量的新数据，尤其是个人位置和轨迹数据，而且存储技术的迅速发展，使在时空数据库中存储移动对象的位置和轨迹数据成为可能。由于位

收稿日期：2013-08-30；**修回日期**：2013-11-29
基金项目：国家自然科学基金资助项目(61370083，61073043，61073041)；高等学校博士学科点专项科研基金资助项目(20112304110011，20122304110012);黑龙江省自然科学基金资助项目(F200901)；哈尔滨市科技创新人才研究专项基金资助项目(2011RFXXG015)
Foundation Items: The National Natural Science Foundation of China (61370083, 61073043, 61073041); The Research Fund for the Doctoral Program of Higher Education of China (20112304110011, 20122304110012); The Natural Science Foundation of Heilongjiang Province (F200901); The Special Funds for Technological Innovation Research of Harbin (2011RFXXG015)

置和轨迹数据含有丰富的时空信息，对其进行分析和挖掘可以支持多种与移动对象相关的应用，比如智能交通、交通监控、城市及道路规划、供应链管理等。因此，分析并发布这样的时空数据是非常有必要的。

然而，只要有数据，就必然存在安全与隐私的问题，个人的隐私信息很可能会随着轨迹数据的发布而受到威胁。在轨迹数据中，最大的隐私威胁就是“敏感位置泄露”，如果攻击者能够了解某人在哪些时间访问了哪些位置，那么攻击者就能够确定此人在发布数据库中的真实记录，并且能够了解此人的其他轨迹信息，进而推理得到此人的兴趣爱好、行为模式、社会习惯等隐私信息，造成个人隐私信息的泄露。因此，面向移动社会网络轨迹发布的隐私保护方法是一个亟待解决的问题。

对于传统的关系型数据的隐私保护，学术界已经进行了广泛的研究[1~7]，但是和传统的关系型数据的隐私保护方法相比，轨迹数据的隐私保护方法有着显著的不同，因为包含时空信息的轨迹数据与关系型数据有很大的差异，传统关系型数据的隐私保护不能直接应用到包含时空数据的轨迹隐私保护方法中。因此，有必要设计特定的隐私保护算法来降低轨迹数据发布时的隐私泄露风险。

目标是实现轨迹数据发布的隐私保护。首先，引入了轨迹 *k*-匿名的概念，然后针对现有的轨迹匿名算法在计算轨迹相似性时忽略轨迹形状因素，产生的匿名轨迹集合可用性相对较低这一问题，提出了一种新的轨迹相似性度量模型，不仅考虑了轨迹的时间和空间要素，更加入了轨迹的形状因素，可以在多项式时间内计算轨迹距离，并且可以计算定义在不同时间跨度上的轨迹距离，在此基础上，提出了一种基于轨迹位置形状相似性的隐私保护算法，在轨迹聚类中使用真实的原始位置信息形成数据“面罩”，满足轨迹 *k*-匿名，在有效地保护轨迹数据的同时，显著地提高了轨迹数据的可用性，最后使用合成的和真实的轨迹数据集，验证了该方法的有效性和合理性。

2　相关工作

2.1　关系型数据匿名

针对传统的关系型数据，最常见的攻击方式是“链接攻击”[8]。针对链接攻击，Sweeney[9]提出了 *k*-匿名模型（*k*-anonymity），使概化后的数据集中的任意一个元组至少有 *k*−1 个其他元组与其在准标识符上完全相同，使攻击者不能唯一确定某个个体的敏感属性值，从匿名集合中识别出某个体敏感属性值的概率不超过 1/*k*。

然而，*k*-匿名不能直接应用于时空轨迹数据，因为轨迹中任何时空点或者时空点的组合都可以被视为一个准标识符属性。在轨迹数据上直接进行 *k*-匿名操作，对于任意一个轨迹，都需要至少 *k*−1 个其他原始轨迹被转换成完全相同的匿名轨迹来构成一个匿名轨迹集合，这样会造成巨大的信息损失。

2.2　轨迹匿名

针对轨迹数据发布可能导致的隐私泄露问题，研究人员做了大量的研究[10~14]。Abul 等[12]提出了(NWA, never walk alone)方法，该方法提出基于共定位的(k, δ)-匿名模型，将轨迹集合划分成不相交的子集，通过聚类形成满足 *k*-匿名的轨迹集合；针对欧式距离可能导致的大量离群点，Abul 等[13]提出了(W4M, wait for me)的方法，使用编辑距离来度量轨迹上采样点之间的距离。

Nergiz 等[14,15]提出基于泛化的算法，首先，通过轨迹点匹配的方式将轨迹聚集到聚类中。然后，将同一类中的轨迹的对应点泛化为最小边界矩形。最后，对匿名的轨迹数据进行重构，并发布重构后的原子轨迹。

Huo 等[16]将轨迹 *k*-匿名集合的选择问题转化为一个图划分的问题，通过最小化划分代价，来降低匿名过程中的信息损失；为了进一步降低信息损失，Huo 等[17]提出了(YCWA, you can walk alone)方法，该方法提取轨迹中重要的停留点，并使用基于网格和基于聚类的方法对停留点进行匿名。

Josep 等[18]提出了一个基于微聚集和排列的匿名算法，通过微聚集算法对轨迹进行聚类，然后使用位置排列算法，对轨迹数据进行重构。

本方法与以上方法有较大不同：首先，使用的轨迹相似性度量标准，不仅考虑了时间和空间因素，还考虑了轨迹的形状因素，能更准确地衡量轨迹的相似性；其次，使用了贪婪的轨迹聚类策略，最大限度地提高了聚类内部轨迹的相似性；最后，将数据“面罩”的方法与聚类技术相结合，在保证轨迹隐私保护程度的基础上，显著地提高了轨迹数

据的可用性。

3 轨迹可用性度量

为了准确地度量匿名轨迹的可用性，使用几个可用性度量标准，如下所述。

1) 聚类和匿名过程中，删除轨迹的比例。

2) 聚类和匿名过程中，删除位置的比例。

3) 轨迹的时空信息损失。

定义 1 轨迹。轨迹指的是时间位置点的一个有序集合，可以表示如下

$$T=\{(t_1,x_1,y_1),(t_2,x_2,y_2),\cdots,(t_n,x_n,y_n)\} \quad (1)$$

定义 2 子轨迹。轨迹 $S=\{(t_1',x_1',y_1'),(t_2',x_2',y_2'),\cdots,(t_m',x_m',y_m')\}$是式(1)中轨迹 T 的子轨迹，如果存在整数 $1\leqslant i_1<\cdots<i_m\leqslant n$ 满足$(t_j',x_j',y_j')=(t_{i_j},x_{i_j},y_{i_j}),1\leqslant j\leqslant m$，可以表示为 $S\leqslant T$。

为了衡量轨迹的时空信息损失，在文献[12]的基础上，定义如下时空信息损失，不仅考虑了轨迹的空间损失，还考虑了轨迹的时间损失。

定义 3 时空信息损失。原始轨迹 T 在 t 时刻的三元组(t,x,y)经匿名操作形成匿名轨迹 T^*在对应时间 t^*时刻的三元组(t^*,x^*,y^*)，则此三元组匿名产生的时空信息损失可以表示为

$$SD_t(T,T^*)=\begin{cases}\sqrt{(1+(t^*-t)^2)((x^*-x)^2+(y^*-y)^2)},\\(x^*,y^*)在t^*时刻有意义\\\Omega,其他\end{cases} \quad (2)$$

其中，Ω 是一个常数，表示对删除位置的惩罚。那么，整条匿名轨迹 T^*相对于原始轨迹 T 的空间失真为

$$SD(T,T^*)=\begin{cases}n\Omega,如果T在匿名过程中被删除\\\sum_{t\in TS}SD_t(T,T^*),其他\end{cases} \quad (3)$$

其中，TS 为轨迹 T 中包含的所有时间点，$n=|TS|$。那么，匿名轨迹集合 GT^*相对于原始轨迹集合 GT 的空间失真为

$$\text{Total}\,SD(GT,GT^*)=\sum_{T\in GT}SD(T,T^*) \quad (4)$$

4 轨迹隐私模型

4.1 攻击模型

轨迹数据发布时个人隐私泄露的风险与攻击者所掌握的背景知识有关，攻击者掌握的背景知识越多，个人的隐私泄露风险越大，反之亦然。假设攻击者掌握如下所述的背景知识。

1) 攻击者可以访问发布的匿名轨迹集合 GT^*，并且知道匿名轨迹集合 GT^*中的任意一个位置 λ 所包含的信息(即(t, x, y)中的元素)都存在于原始的轨迹集合中。

2) 攻击者知道原始的目标轨迹 T 的子轨迹 S。

在此基础上，定义攻击模型如下。

定义 4 攻击模型。对于已知的子轨迹 $S(S\leqslant T)$和匿名轨迹集合 GT^*，如果攻击者能够确定匿名轨迹 $T^*(T^*\in GT^*)$是 T 的匿名轨迹，则认为用户的个人隐私遭到攻击。

4.2 轨迹隐私模型描述

为了能准确地衡量轨迹隐私保护程度，引入以下轨迹隐私模型[18]。

给定子轨迹 $S(S\leqslant T)$，$\Pr_{T^*}[T|S]$表示攻击者能够确定匿名轨迹 $T^*(T^*\in GT^*)$是 T 的匿名轨迹的概率。

定义 5 轨迹 p-隐私。如果 $\Pr_{T^*}[T|S]$ $(\forall T\in GT, S\leqslant T)$，那么就说匿名轨迹满足轨迹 p-隐私。

定义 6 轨迹 k-匿名。满足轨迹 k-匿名当且仅当满足轨迹 $1/k$-隐私。

5 轨迹相似性度量

聚类轨迹需要定义轨迹相似性度量标准来衡量 2 个轨迹之间的距离，然而现在的轨迹相似性度量只考虑了轨迹的时间和空间因素，忽略了轨迹的形状因素对轨迹间相似性的影响。图 1 显示了 3 条轨迹，每条轨迹分别对应 5 个采样点，且每个对应的采样点时间 t 相同（另假设与未标出的 y 轴坐标相同），只有 x 轴坐标不同。通过分别计算 5 个对应的采样点之间的欧式距离，很容易得到，轨迹 2 和轨迹 3 到轨迹 1 的距离完全相同；但是由图 1 可以发现，轨迹 2 与轨迹 1 的相似性明显大于轨迹 3 与轨迹 1 的相似性，因为上述只考虑轨迹对应采样点距离的相似性计算方法，忽略了轨迹的形状因素对轨迹间相似性的影响。

因此，为了更准确地衡量轨迹之间的相似性，在已有研究的基础上，加入了轨迹的形状因素，提出了如下轨迹相似性度量模型。

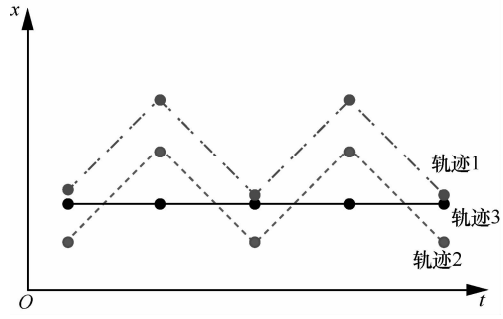


图 1　3 条轨迹间距离对比

5.1　轨迹相交和同步轨迹

定义 7　相交轨迹。2 条轨迹 T_i= {(t_1^i, x_1^i, y_1^i), (t_2^i, x_2^i, y_2^i), …, (t_n^i, x_n^i, y_n^i)} 和 T_j={(t_1^j, x_1^j, y_1^j), (t_2^j, x_2^j, y_2^j),…, (t_m^j, x_m^j, y_m^j)}, I = max(min(t_n^i, t_m^j)−max(t_1^i, t_1^j), 0)。如果 I>0，那么轨迹 T_i 和 T_j 是相交轨迹；否则，轨迹 T_i 和 T_j 不是相交轨迹。

定义 8　轨迹 p%-相交[18]。2 条轨迹 T_i= {(t_1^i, x_1^i, y_1^i), (t_2^i, x_2^i, y_2^i),…,(t_n^i, x_n^i, y_n^i)}和 T_j= {(t_1^j, x_1^j, y_1^j), (t_2^j, x_2^j, y_2^j), …, (t_m^j, x_m^j, y_m^j)} 相交，I=max(min(t_n^i, t_m^j)−max(t_1^i, t_1^j), 0), 如果满足 $p = 100\left(\frac{I}{\max(t_n^i, t_m^j) - \min(t_1^i, t_1^j)}\right)$，那么 T_i 和 T_j 轨迹 p%-相交。

2 条轨迹 100%-相交，当且仅当这 2 条轨迹有相同的开始时间和相同的结束时间，2 条轨迹 0%-相交，当且仅当 2 条轨迹没有重叠的时间间隔。

定义 9　同步轨迹。2 条轨迹 T_i 和 T_j 轨迹 p%-相交，p>0，如果在重叠时间间隔 $ot(T_i, T_j)$内，轨迹 T_i 和 T_j 有相同数量的位置点，并且位置点对应的时间点对应相同，那么轨迹 T_i 和 T_j 是同步轨迹。

定义 10　同步轨迹集合。轨迹集合为同步轨迹集合当且仅当集合内任意 2 条 p%-相交（p>0）的轨迹为同步轨迹。

为了计算轨迹相似性，必须设法将不满足同步轨迹条件的轨迹转换为同步轨迹。假设移动对象在任意 2 个相邻采样点之间都做匀速直线运动，如果轨迹 T_i 和轨迹 T_j p%-相交，p>0，轨迹 T_i 在 t 时刻 ($t \in ot(T_i, T_j)$)有一个采样点 l_i，轨迹 T_j 在 t 时刻没有采样点，那么可以直接在轨迹 T_j 中 t 时刻插入一个采样点 l_j，并满足假设，反之亦然。通过重复以上操作，即可将轨迹 T_i 和 T_j 转换为同步轨迹，具体的算法见文献[18]中的算法 1。

5.2　轨迹形状距离

为了衡量轨迹形状的相似性，使用轨迹段间斜率的大小作为度量轨迹间形状相似性的标准，定义如下轨迹形状距离。

定义 11　轨迹形状距离。给定同步轨迹集合 GT= {T_1, T_2,…, T_n},轨迹 T_i={(t_1^i, x_1^i, y_1^i), (t_2^i, x_2^i, y_2^i),…, (t_n^i, x_n^i, y_n^i)} 和 T_j={(t_1^j, x_1^j, y_1^j), (t_2^j, x_2^j, y_2^j), …, (t_m^j, x_m^j, y_m^j)}均属于 GT，并且 T_i 和 T_j 轨迹 p%-相交，p>0，则轨迹 T_i 和 T_j 之间的形状距离可以表示为

$$d_{\text{shape}}(T_i, T_j) = \frac{1}{p}\sqrt{\sum_{t_s, t_{s+1} \in ot(T_i, T_j)} \left(\frac{(x_{s+1}^i - x_s^i)}{(t_{s+1}^i - t_s^i)} - \frac{(x_{s+1}^j - x_s^j)}{(t_{s+1}^j - t_s^j)}\right)^2 + \left(\frac{(y_{s+1}^i - y_s^i)}{(t_{s+1}^i - t_s^i)} - \frac{(y_{s+1}^j - y_s^j)}{(t_{s+1}^j - t_s^j)}\right)^2} \tag{5}$$

5.3　轨迹位置距离

定义 12　轨迹位置距离。给定同步轨迹集合 GT= {T_1,T_2,…, T_n},轨迹 T_i= {(t_1^i, x_1^i, y_1^i), (t_2^i, x_2^i, y_2^i),…, (t_n^i, x_n^i, y_n^i)} 和 T_j={(t_1^j, x_1^j, y_1^j), (t_2^j, x_2^j, y_2^j), …, (t_m^j, x_m^j, y_m^j)}均属于 GT，并且 T_i 和 T_j 轨迹 p%-相交，p>0，则轨迹 T_i 和 T_j 之间的位置距离可以表示为

$$d_{\text{loc}}(T_i, T_j) = \frac{1}{p}\sqrt{\sum_{t_s \in ot(T_i, T_j)} \frac{(x_s^i - x_s^j)^2 + (y_s^i - y_s^j)^2}{\left|ot(T_i, T_j)\right|^2}} \tag{6}$$

5.4　轨迹相似性计算

使用轨迹距离来度量轨迹之间的相似性，轨迹间的距离越大，轨迹间相似性越小，反之亦然。下面，在定义了轨迹形状距离和轨迹位置距离的基础上，定义轨迹距离。

定义 13　轨迹距离。给定同步轨迹集合 GT={T_1, T_2, … ,T_n}, 轨迹 T_i={(t_1^i, x_1^i, y_1^i), (t_2^i, x_2^i, y_2^i), …, (t_n^i, x_n^i, y_n^i)} 和 T_j={(t_1^j, x_1^j, y_1^j), (t_2^j, x_2^j, y_2^j), …, (t_m^j, x_m^j, y_m^j)}均属于 GT，并且轨迹 T_i 和轨迹 T_j p%-相交，p>0，则轨迹 T_i 和轨迹 T_j 之间的距离可以表示为

$$d(T_i, T_j) = \alpha d_{\text{shape}}(T_i, T_j) + (1-\alpha) d_{\text{loc}}(T_i, T_j) \tag{7}$$

其中，α 为衡量轨迹形状距离和轨迹位置距离的一个权重，$\alpha \in [0,1]$，可以根据实际情况指定，使用 α=0.5。

如果 T_i 和 T_j 轨迹 0%-相交，即轨迹 T_i 和 T_j 不是相交轨迹，则判断同步轨迹集合中是否存在轨迹 T_k，满足 T_i 和 T_k 是相交轨迹，并且 T_k 和 T_j 是相交轨迹。如果存在这样的轨迹 T_k，则轨迹 T_i 和 T_j 之间的距离可以表示为

$$d(T_i,T_j)=\sum_{\min T_k} d(T_i,T_k)+d(T_k,T_j) \tag{8}$$

否则，不能度量轨迹 T_i 和 T_j 之间的距离，记为 $d(T_i, T_j)=\infty$。

为了能够直观地理解、高效地计算轨迹距离，使用一个无向带权图来描述轨迹间的距离。

定义 14 轨迹距离图。轨迹距离图是一个无向带权图，它满足的条件如下。

1) 节点代表轨迹。

2) 节点 T_i 和 T_j 直接相连当且仅当对应的轨迹 p%-相交，$p>0$。

3) T_i 和 T_j 的边的权重代表了对应轨迹间的距离。

因此，给定一个轨迹集合，首先将其转化为同步轨迹集合，然后计算其中任意 2 条轨迹的距离，并据此构造轨迹距离图，轨迹距离图中任意两点间的最短距离即为对应 2 条轨迹的距离。根据以上定义，就可以计算任意 2 条轨迹间的距离，无论它们是否定义在相同的时间跨度上。使用 Floyd-Warshall 算法[19]来计算轨迹距离图中任意两点之间的最短距离。

6 轨迹匿名算法

6.1 GC-DM 轨迹匿名算法

为了保证发布的轨迹数据的隐私不被泄露，提出了基于轨迹位置形状相似性的隐私保护算法，它包含 2 个阶段(GC-DM, greedy clustering-data mask)，描述如下。

1) 贪婪聚类：根据贪婪聚类思想，将轨迹集合聚类为若干类集合，保证聚类内部轨迹相似性尽量大，聚类半径尽量小，删除的轨迹尽量少；如果删除的轨迹数超过阈值，则放宽聚类限制，直到满足阈值条件。

2) 数据“面罩”：所谓的数据“面罩”即由“时间 t”，“位置坐标 x”，“位置坐标 y”随机组合形成的新的三元组。根据数据“面罩”的思想，将同一个聚类中的轨迹进行位置元素重构，形成满足条件的 k 条原子轨迹，以提高数据的可用性。

下面分别介绍算法的 2 个阶段，首先是贪婪聚类算法，如算法 1 所示。

算法 1 Greedy Clustering

输入：轨迹集合 GT，轨迹 k-匿名的参数 k，最大损失轨迹数量 Max-Trash

输出：轨迹等价类集合 Q，轨迹等价类中心点集合 *pivots*

Begin

1) *initialize*(*maxradius*);// 初始化参数 *maxradius*，默认为数据集地图半径的 0.5%，也可以手动输入

2) Repeat

3) $Q=\Phi$;//Q 表示轨迹等价类的集合

4) *trash* $=\Phi$;//删除的轨迹集合

5) *pivots* $=\Phi$;//轨迹等价类中心点集合

6) active=GT;//标记轨迹集合中的轨迹为 active 状态

7) T_p 为轨迹集合 GT 中的平均轨迹；

8) while active $\neq\Phi$ && |active| $\geqslant k$ do

9) T_p=arg $\max_{t\in \text{active}}$ $d(T_p,t)$;

10) C_p={T_p}$\cup${$k-1$ 个到 T_p 距离最近的轨迹，并且在集合 $GT\backslash Q$ 中}

11) If $\max_{t\in Cp}$ $d(T_p,t)\leqslant$ *maxradius* then

12) active = active$\backslash C_p$;

13) $Q=Q\cup C_p$;

14) *pivots* = *pivots*$\cup T_p$;

15) Else

16) active = active$\backslash T_p$;

17) End if

18) End while

19) For $T\in GT\backslash Q$ do

20) T_p=arg $\min_{T'\in pivots}$ $d(T',T)$

21) If $d(T,T_p)\leqslant$ *maxradius* then

22) $C_p=C_p\cup${T};

23) Else

24) *trash*=*trash*$\cup${T}

25) End If

26) End For

27) *increase*(*maxradius*); // 迭代增加参数 *maxradius*, 每次增加 0.5 倍

28) until | take care,rash| $\leqslant$Max-Trash;

29) Return Q

End

算法 1 是一个贪婪的聚类算法，它首先选择一系列轨迹作为聚类等价类的中心轨迹，每一个选择的中心轨迹都是距离上一个中心轨迹最远的轨迹，第一个选取的中心轨迹是整个轨迹集合的平均轨迹(7~9 行)。然后，将每个中心轨迹和与之距离最近的 k–1 条轨迹（均不属于其他等价类）形成一个包含 k 条轨迹的等价类（10 行），最后剩下的轨迹将被分配到距离其最近的中心集合所在的等价类集合中（19~22 行）。

但是，本算法在聚类过程中，增加了其他限制条件，即聚类半径不能超过某阈值 *maxradius*（11~14 行，21~22 行）。参数 *maxradius* 默认为数据集地图半径的 0.5%，也可以设置为其他阈值。当不能以中心轨迹形成等价类时，该轨迹被移除 active 类，即该轨迹不能再作为中心轨迹，但是仍可以作为轨迹等价类中的成员（15~16 行）。当最后剩下的轨迹不能够加入其他轨迹等价类时，它们被直接删除（23~24 行）。如果本次处理过程导致最大损失的轨迹数量大于 max-trash（28 行），可以调整参数 *maxradius* 的值（27 行），得到一个合适的聚类结果。

为了使聚类的结果满足引用的轨迹 k-匿名模型，将聚类技术与数据“面罩”技术相结合，通过聚类集合内部轨迹的时间和位置坐标信息的变换，起到保护轨迹隐私的效果，并且提高发布的轨迹数据的可用性。数据“面罩”算法的具体步骤如算法 2 所示。

算法 2　Data Mask

输入：轨迹等价类集合 $Q=\{C_1,C_2,\cdots,C_m\}$，时间阈值 Rt，空间阈值 Rs，轨迹等价类中心点集合 *pivots*

输出：匿名轨迹等价类集合 Q^*

Begin

1) For each cluster $C\in Q$ do//轨迹等价类集合中的任意一个等价类
2) 标记轨迹等价类 C 中的轨迹的轨迹点为 active 状态
3) T_c 为等价类 C 中的轨迹中心点
4) For triples $l=(t_l, x_l, y_l)\in T_c$ 并且 l 为 active 状态 do
5) $U=\{l\}$;
6) For $T\in C$ 且 $T\neq T_c$ do
7) 寻找 T 内状态为 active，满足到位置 l 距离最短的位置 l'，且满足：

$$\left|t_{l'}-t_l\right|\leqslant Rt, 0\leqslant\sqrt{(x_{l'}-x_l)^2+(y_{l'}-y_l)^2}\leqslant Rs$$

8) If 这样的位置 l'存在 then
9) $U=U\cup\{l'\}$;
10) Else
11) 删除位置 l，并回到第 4 步继续执行
12) End If
13) End For
14) If $|U|\geqslant k$ then
15) 随机置换集合 U 中各个轨迹点的位置和时间
//位置坐标信息间相互置换，时间相互置换;
16) 标记集合 U 内的元组状态为 unactive;
17) Else
18) 删除位置 l，并回到第 4 步继续执行
19) End If
20) End for
21) 删除等价类 C 中所有状态为 active 的轨迹点
22) End For
23) Return Q^*

End

算法 2 结合聚类技术和数据“面罩”技术，将聚类后形成的轨迹等价类集合转换为满足轨迹 k-匿名的轨迹等价类集合。它首先选取等价类集合中的任意一个轨迹等价类，并将其中的轨迹点标记为 active 状态（1~2 行），然后选取等价类中的中心轨迹，并以其包含的轨迹点为基础，从该等价类的其他轨迹中选择到其距离最近的轨迹点，并且满足一定的时间和空间限制（4~13 行），如果能找到 k–1 个这样的轨迹点(14 行)，则通过数据“面罩”技术，置换轨迹点内的时间和位置元素，并修改状态为 unactive（15~16 行），否则，直接将轨迹点删除（10~11 行，17~18 行，21 行）。

6.2　算法时间复杂性分析

在给出具体算法的基础上，分析算法的时间复杂性。假设第 5 节中提到的轨迹距离图作为已知，因为在数据集一定的情况下，轨迹距离图只计算一次。

算法 1 是一个贪婪算法，为了追求最好的效果，

可能需要反复执行，执行次数为 $O(m)$，每一次执行的算法时间度为 $O(N^2)$，N 表示轨迹集合中轨迹的数量。

算法 2 是在算法 1 的基础上，针对算法 1 产生的(N/k)个轨迹等价类进行操作，轨迹的平均长度为 $O(n)$，对于每个轨迹点，需要在其他 $k-1$ 条轨迹中寻找对应的匿名轨迹点，候选的轨迹点共有 $O((k-1)n)$，因此，算法 2 的时间复杂度为 $O((k-1)n^2)$。

因此，一般情况下算法总的时间复杂度为 $O(m)O(N^2)+O(N/k)O((k-1)n^2)=O(mN^2)+O(Nn^2)$, 最好的情况下，贪婪算法只执行一次，此时算法的时间复杂度为 $O(N^2)+O(Nn^2)$。

对比 Josep 等在文献[18]提出的算法，提出的 GC-DM 算法在计算轨迹相似性时加入了轨迹的形状因素，能够更加准确地度量轨迹间的相似性，但是时间复杂度和原算法相当，充分体现了本算法的效率，实验部分的结果给出了有力的佐证。

7 轨迹隐私保障

命题 1 假设子轨迹 S 是攻击者掌握的关于原始目标轨迹 T_s 的背景知识，满足 $S\leqslant T_s$，$\lambda_1,\lambda_2,\cdots,\lambda_{|S|}$ 是子轨迹 S 的全部轨迹点，对于任意一个轨迹 T_i，子轨迹 S 中的轨迹点 λ 出现在轨迹 T_i 的匿名轨迹 T_i^* 中的概率为

$$\Pr(\lambda\in T_i^*\,|\,\lambda\in S)=\begin{cases}P<1/k, T_s\text{和}T_i\text{在同一个轨迹}\\ \qquad\text{等价类中,且未被删除}\\ 0,\text{其他}\end{cases}\tag{9}$$

证明 通过 Data Mask 算法可以看到，如果 T_s 和 T_i 不在同一个轨迹等价类中，就不可能对 T_s 和 T_i 中的轨迹点进行交换操作，因此 $\Pr(\lambda\in T_i^*\,|\,\lambda\in S)=0$。

设经过 Data Mask 算法匿名操作后，轨迹 $T_1^*,T_2^*,\cdots,T_k^*$ 被聚集在同一个轨迹等价类集合中，不失一般性，假设 $T_s=T_1$。通过 Data Mask 算法可以看到，对于任意的 $i(1\leqslant i\leqslant k)$，$j(1\leqslant j\leqslant|S|)$，如果轨迹点 λ_j 被删除，$\Pr(\lambda_j\in T_i^*\,|\,\lambda_j\in S)=0$；否则，$\Pr(\lambda_j\in T_i^*\,|\,\lambda_j\in S)=P<1/k$。这是因为，轨迹等价类集合中的轨迹在置换轨迹点的时候不是简单的轨迹点之间的置换，而是位置坐标元素和时间的同时置换，因此 $P<1/k$。综上所述，可以得到命题 1 是正确的。

定理 1 Data Mask 算法实现了轨迹 k-匿名

证明 由命题 1 可知，对任意子轨迹 $S\leqslant T_i$，在轨迹等价类集合 $T_1^*,T_2^*,\cdots,T_k^*$ 中，S 成为 $T_1^*,T_2^*,\cdots,T_k^*$ 子轨迹的概率相等，并都小于 $1/k$，即对于子轨迹 S，攻击者不能以超过 $1/k$ 的概率确定轨迹 T_i 是轨迹 T_i^* 的原始轨迹，根据定义 6，Data Mask 算法实现了轨迹 k-匿名。

8 实验及结果分析

在实验部分，使用了 2 个数据集对算法的可用性进行分析。

1) 合成数据集。使用 Brinkhoff 轨迹生成器生成了 1 000 条合成轨迹，共包含德国奥尔登堡市的 46 906 个位置。

2) 真实数据集。使用一个收集自美国旧金山市的出租车移动轨迹作为本实验的真实数据集[20]，经过过滤操作，得到 2 393 条轨迹信息，其中每条轨迹平均包含 74.9 个位置。使用该数据集的优势在于它包含更大量的轨迹信息，而且这些轨迹信息都是真实的。

实验的硬件环境为：Intel(R) Core(TM)2 Quad CPU Q8 400 @2.66 GHz 2.67 GHz, 2.00 GB 内存，操作系统为 Microsoft Windows 7，算法均在 Matlab R2012a 下实现。

8.1 合成数据集实验结果

考虑到实验结果的可再现性，将生成合成轨迹数据的 Brinkhoff 轨迹生成器的参数公布如下：6 个移动对象类，3 个外部对象类；每个时间点生成 10 个移动对象和 1 个外部对象；共 100 个时间点；移动速度 250；“probability”为 1 000。这种设置生成了包含 46 906 个位置的 1 000 条合成轨迹，最长的轨迹包含 100 个位置信息，每条轨迹平均包含 46.9 个位置。

为了方便提出的算法(GC-DM)的可用性进行比较，实现了文献[18]提出的算法(MDAV)，并对算法及其可用性进行了分析。

8.1.1 GC-DM 轨迹匿名算法分析

最主要的目标就是在提供足够的隐私保护的基础上，最大化轨迹数据的可用性。轨迹数据的可用性用轨迹数据信息损失的多少来衡量，信息损失越大，轨迹数据的可用性越小，反之亦然。

第 2 节中提出了轨迹可用性度量标准，为了

获得准确的信息损失，需要选择合适的 Ω。使用位置置换所产生的信息损失的最大值来近似的估计 Ω。取 Rs（space threshold）为 10^9，Rt（time threshold）为 100 时，k 值变化下位置置换信息损失的最大值作为 Ω 的取值。实验结果如图 2 和图 3 所示。

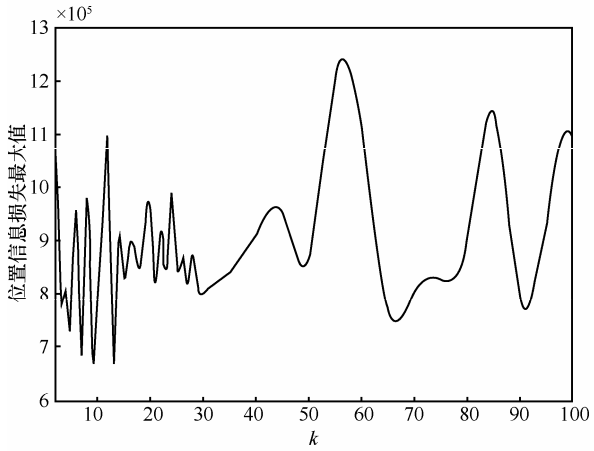


图 2　MDAV 算法中 k 值变化下位置置换信息损失的最大值

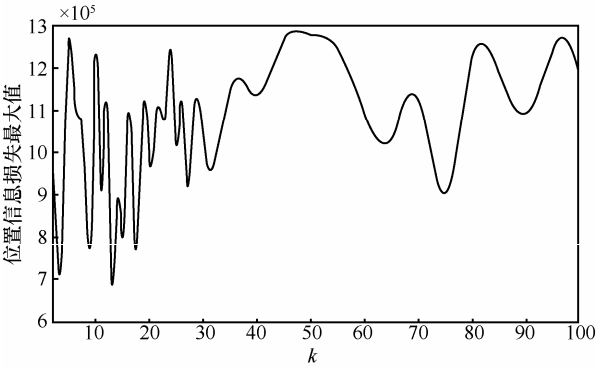


图 3　GC-DM 算法中 k 值变化下位置置换信息损失的最大值

由图 2 和图 3 可知，位置置换产生的信息损失的最大值随着 k 值的增加波动比较大，但是 k 在 40~60 之间取到最大值。因此，将两图中最大的位置置换信息损失作为 Ω 的取值，即 Ω=1.267 5e^6。实验中 Rt 取 100，Rs 取 10^9，是为了确保时间阈值和空间阈值足够大，不会由于阈值过小造成额外的信息损失。

图 4 描述了 GC-DM 算法在 Rs=100 000，*maxradius* (聚类最大半径)=178.95，*Max-Trash*(聚类最大删除轨迹数量)=10 时，随 k 值和 Rt 变化的信息损失变化情况。

图 5 描述了 GC-DM 算法在 Rs=100 000，*maxradius*=178.95，Rt=100 时，随 k 值和 *Max-Trash* 变化的信息损失变化情况。

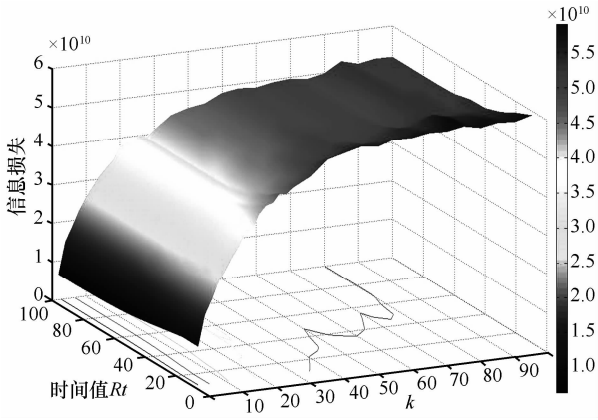


图 4　GC-DM 算法中 k 值和 Rt 变化下的信息损失

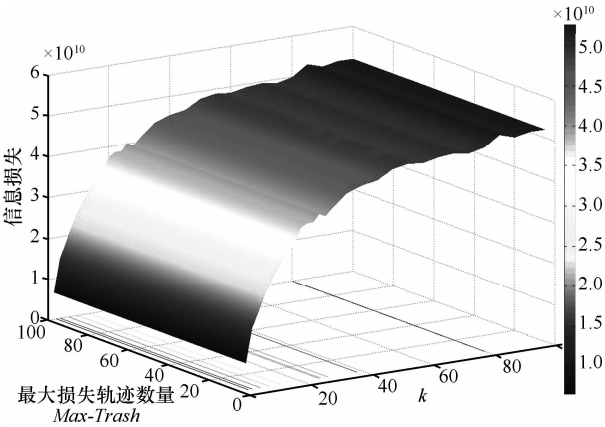


图 5　GC-DM 算法中 k 值和 *Max-Trash* 变化下的信息损失

图 6 描述了 GC-DM 算法在 Rs=100 000，Rt=100，*Max-Trash*=10 时，随 k 值和 *maxradius* 变化的信息损失变化情况。

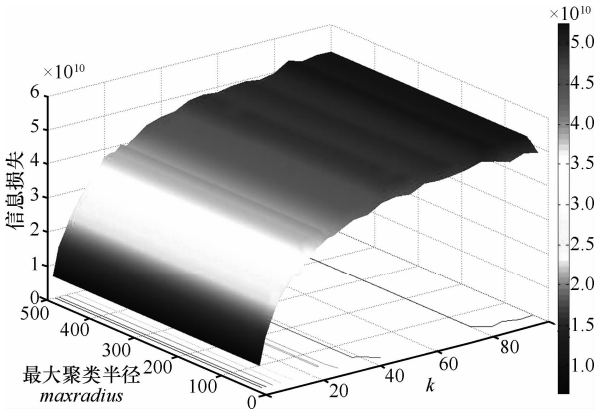


图 6　GC-DM 算法中 k 值和 *maxradius* 变化下的信息损失

通过观察图 4~图 6 可以发现，当 Rt, *Max-Trash*, *maxradius* 一定时，随着 k 值的增加，信息损失在增加，这是因为 k 值增加，使用贪婪聚类算法形成轨迹等价类以及使用数据“面罩”技术形成位置等价类时，需要包含更多的轨迹信息和位置信息，因此

会删除更多的轨迹和位置，造成较大的信息损失；当 *k* 值一定时，随着 *Rt*, *Max-Trash*, *maxradius* 的增加，信息损失基本保持不变，这是因为 *Rt* 不变时，轨迹的时间点分布的比较集中；而 *Max-Trash* 和 *maxradius* 主要应用于轨迹聚类阶段，对之后位置等价类的形成影响比较小。所以，当 *Rs* 取较大值时，相对于参数 *k*, *Rt*, *Max-Trash* 和 *maxradius* 对实验结果的影响较小。

8.1.2 可用性比较

为了分析本算法的可用性，从以下 3 个方面来衡量：删除的轨迹数据的比例；删除的位置信息的比例；总的信息损失。其中，总的信息损失包括删除轨迹和位置带来的信息损失以及位置置换带来的信息损失。

图 7 和图 8 分别描述了 2 个算法在 *Rt*=100，*maxradius*=178.95，*Max-Trash*=10 时，MDAV 算法和 GC-DM 算法在 *k* 值和 *Rs* 变化下删除轨迹的比例。

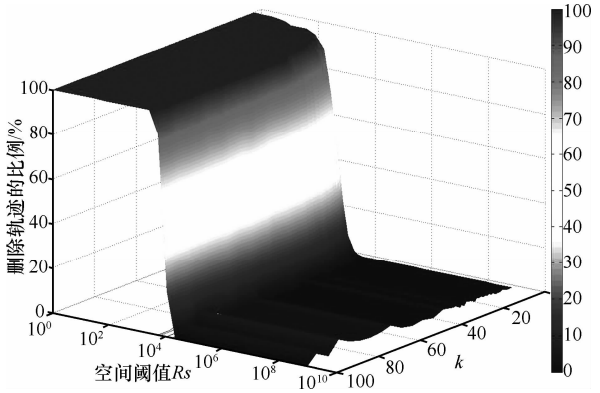


图 7 MDAV 算法中 *k* 值和 *Rs* 变化下删除轨迹的比例

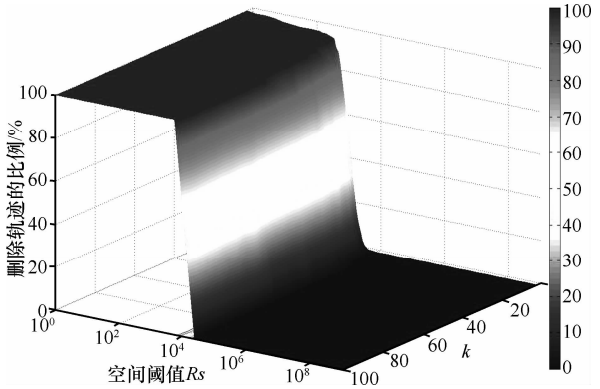


图 8 GC-DM 算法中 *k* 值和 *Rs* 变化下删除轨迹的比例

通过观察图 7 和图 8 可以发现，当 *k* 值一定时，随着 *Rs* 的增加，轨迹删除比例从 100%逐渐下降，但是下降的速度十分缓慢，然后 *Rs* 取值在 10^4 左右时，开始迅速下降，直到接近 0%，然后继续增加 *Rs*，轨迹删除比例基本保持不变。产生这样的变化曲线是因为在 *Rs* 取值较小时，通过数据“面罩”技术不能形成满足阈值条件的位置等价类，不能进行位置置换操作，故删除的轨迹数据很大，甚至会删除全部轨迹；当 *Rs* 增加至 10^4 左右时，由于空间阈值足够大，通过数据“面罩”技术能够形成一定数量的位置等价类，删除的轨迹数据开始减小；随着 *Rs* 继续增加，删除的轨迹数据迅速减少甚至为 0，说明在此 *Rs* 阈值下，几乎所有的轨迹都有位置加入到位置等价类中，所以删除的轨迹数量极少。

图 9 和图 10 分别描述了 2 个算法在 *Rt*=100，*maxradius*=178.95，*Max-Trash*=10 时，MDAV 算法和 GC-DM 算法在 *k* 值和 *Rs* 变化下删除位置的比例。

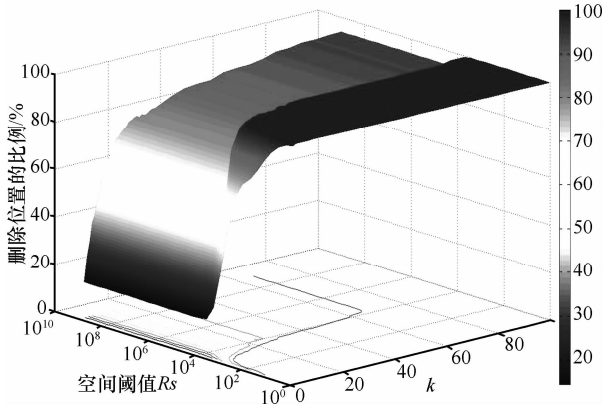


图 9 MDAV 算法中 *k* 值和 *Rs* 变化下删除位置的比例

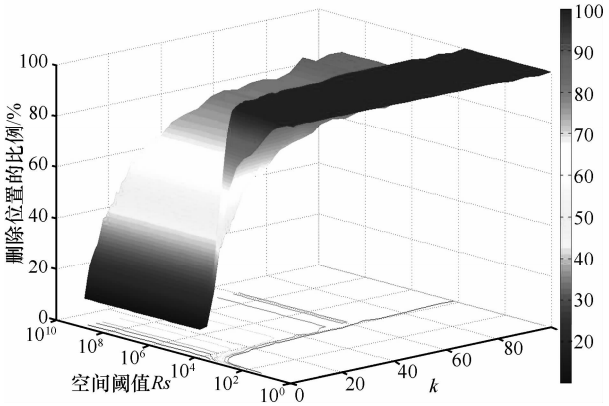


图 10 GC-DM 算法中 *k* 值和 *Rs* 变化下删除位置的比例

通过观察图 9 和图 10 可以发现，当 *k* 值一定时，随着 *Rs* 的增加，删除位置的比例从 100%开始逐渐减小，但是减小的速度十分缓慢，在 *k* 取值较

大时甚至保持 100%不变；当 *Rs* 增加至接近 10^4 时，删除位置的比例开始减小，在增加 *Rs* 至 10^4 时，删除位置的比例不再减小，即增加 *Rs* 也不再改变。之所以产生这样的变化曲线是因为在 *Rs* 取值较小时，通过数据“面罩”技术不能形成满足阈值条件的位置等价类，不能进行位置置换操作，故删除的位置数据很大，甚至删除全部位置；随着 *Rs* 的增加，当空间阈值足够大时，通过数据“面罩”技术能够形成一定数量的位置等价类，删除的位置数据开始减小；当空间阈值增加到一定程度时，空间阈值对位置等价类不再产生限制，删除的位置数量基本保持不变。

当 *Rs* 取值较小时，随着 *k* 值的增加，删除位置的比例在增加，但速度比较缓慢，并且接近 100%，这是因为在 *Rs* 取值较小时，通过数据“面罩”技术很难形成满足阈值条件的位置等价类，只有在 *k* 值较小时，才可能形成位置等价类，*k* 值较大时，不能形成满足空间阈值条件的位置等价类；当 *Rs* 取值较大时，随着 *k* 值的增加，删除位置的比例也在增加，且增加速度较快。因为当 *Rs* 足够大时，空间阈值对位置等价类的生成限制很小，随着 *k* 值的增加，位置等价类形成过程中考虑的轨迹数量越多，造成删除的位置数量越多。

通过图 9 和图 10 的对比不难发现，对于同样的 *k* 值和 *Rs* 值，GC-DM 算法产生的删除位置比例要小于 MDAV 算法，即 GC-DM 算法要比 MDAV 算法产生更小的信息损失，更高的可用性。因为 GC-DM 算法使用的轨迹距离度量标准同时考虑了轨迹的位置距离和形状距离，能够更加贴切地衡量 2 个轨迹之间的距离，而且使用的贪婪聚类技术使轨迹等价类内部轨迹相似性更大，类间轨迹相似性更小，之后使用数据“面罩”技术形成位置等价类时，删除的位置数量减少。

图 11 和图 12 分别描述了在 *Rt*=100，*maxradius*=178.95，*Max-Trash*=10 时，MDAV 算法和 GC-DM 算法在 *k* 值和 *Rs* 变化下的信息损失。

通过观察图 11 和图 12 可以发现，当 *k* 值一定时，随着 *Rs* 增加，信息损失逐渐减小，但减小的速度十分缓慢，在 *k* 值较大时甚至不变；当 *Rs* 增加至接近 10^4 时，信息损失开始减小，在增加 *Rs* 至 10^4 时，信息损失不再减小，即使增加 *Rs* 也不再改变。产生这样的变化曲线是因为在 *Rs* 取值较小时，通过数据“面罩”技术不能形成满足阈值条件的位置等价类，不能进行位置置换，故删除的位置和轨迹数据很大，甚至会删除全部位置和轨迹数据；随着 *Rs* 增加，当空间阈值足够大时，通过数据“面罩”技术能形成一定数量的位置等价类，删除的位置数据开始减小，信息损失随之减少；当空间阈值增加到一定程度时，空间阈值对位置等价类不在产生限制，删除的位置数据基本保持不变，信息损失也保持不变。

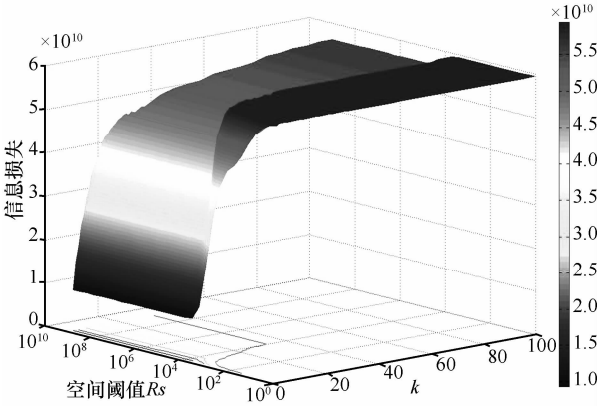


图 11　MDAV 算法中 *k* 值和 *Rs* 变化下的信息损失

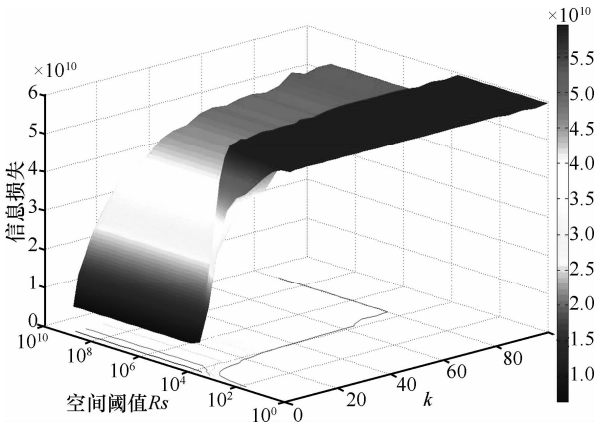


图 12　GC-DM 算法中 *k* 值和 *Rs* 变化下的信息损失

当 *Rs* 值较小时，随着 *k* 值增加，信息损失增加，但速度比较缓慢，因为在 *Rs* 值较小时，通过数据“面罩”技术很难形成满足阈值条件的位置等价类，只有在 *k* 取较小值时，才可能形成位置等价类，*k* 取较大值时，不能形成满足空间阈值条件的位置等价类，造成大的信息损失；当 *Rs* 值较大时，随着 *k* 值增加，信息损失也在增加，并且速度较快。因为当 *Rs* 足够大时，空间阈值对位置等价类的生成限制很小，随着 *k* 值的增加，位置等价类形成过程中考虑的轨迹数量越多，造成删除的位置数量越

多，信息损失越大。

通过图 11 和图 12 的对比发现，对于同样的 k 值和 Rs 值，GC-DM 算法产生的信息损失小于 MDAV 算法，即 GC-DM 算法比 MDAV 算法产生更高的可用性。因为 GC-DM 算法使用的轨迹距离度量标准同时考虑了轨迹的位置距离和形状距离，能够更加贴切地衡量 2 个轨迹之间的距离，而且使用的贪婪聚类技术使轨迹等价类内部轨迹相似性更大，类间轨迹相似性更小，之后使用数据“面罩”技术形成位置等价类时，删除的位置数量减少，因此产生较小的信息损失。

通过以上分析可知，在 Rt，*Max-Trash* 和 *maxradius* 确定时，Rs 在接近 10^4 时，信息损失可取到最小值。而随着 k 值增加，信息损失会一直增加。

8.1.3 执行时间比较

图 13 和图 14 分别描述了在 Rt=100，*maxradius*=178.95，*Max-Trash*=10 时，MDAV 算法和 GC-DM 算法在 k 值和 Rs 变化下的执行时间。

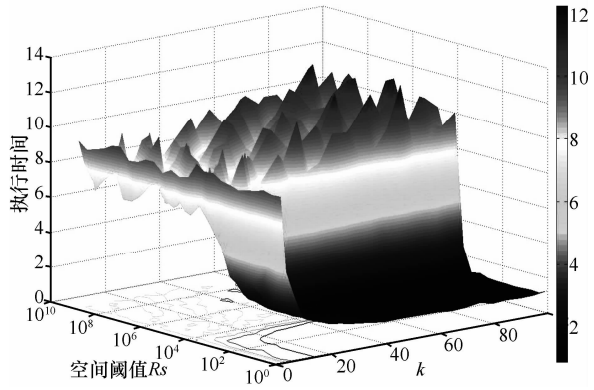


图 13 MDAV 算法中 k 值和 Rs 变化下的执行时间

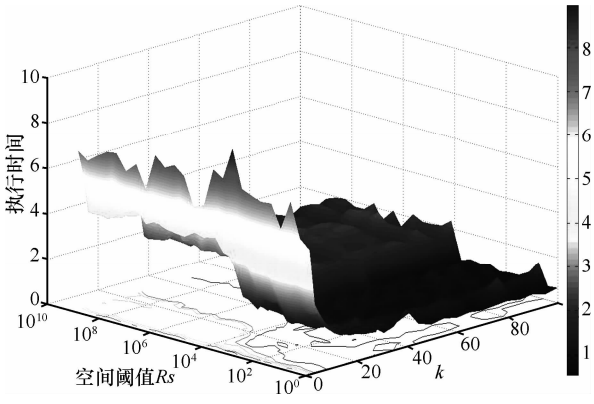


图 14 GC-DM 算法中 k 值和 Rs 变化下的执行时间

通过观察图 13 和图 14 可以发现，同样的实验条件下，GC-DM 算法比 MDAV 算法花费更少的执行时间，尤其在 k 取值增加时，执行效率的体现尤为明显。这主要是因为随着 k 值的增加，满足 k 匿名需要的轨迹增加，造成迭代次数的降低，执行时间减少。

8.2 真实数据集实验结果

使用的出租车数据集[20]由若干个文件构成，每个文件都包含一个出租车在 2008 年 5 月和 6 月的 GPS 信息，文件中的每一行都包含着出租车在给定时间的坐标（经度和纬度）。然而，出租车在一个月内的移动轨迹不可能被看作一条轨迹，因此，设置最大时间间隔，将一条移动轨迹划分成多个轨迹。

为了方便实验，仅选取一天的轨迹数据作为数据集合，由于 2008 年 5 月 25 日 12:04 至 5 月 26 日 12:04 这一天的轨迹信息最多，将其作为数据集合。在该数据集合中，连续位置的平均时间间隔为 92 s，因此设置 3 min（2 倍的平均时间间隔）为最大时间间隔，并选取采样点数量在 20~200 之间的轨迹，得到 2 393 条轨迹信息，每条轨迹平均包含 74.9 个位置。

8.2.1 可用性比较

与第 7 节实验相同，为了获得总的信息损失，需要选择一个合适的 Ω，取 Rs=10^{12}，Rt=100 000 时，k 值变化下位置置换信息损失的最大值作为 Ω 的取值。实验结果如图 15 和图 16 所示。

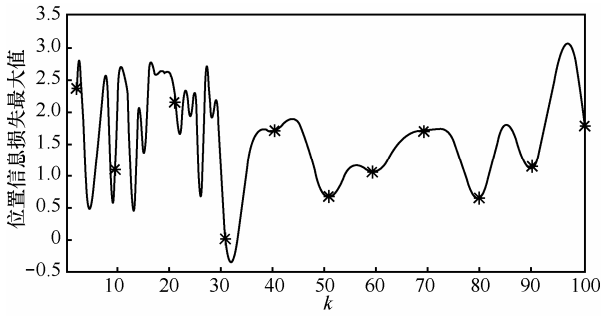


图 15 MDAV 算法中 k 值变化下位置置换信息损失的最大值

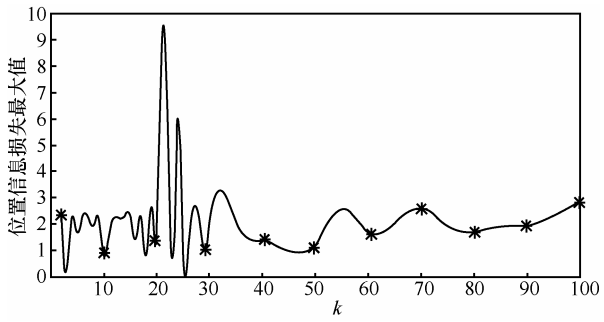


图 16 GC-DM 算法中 k 值变化下位置置换信息损失的最大值

由图 15 和图 16 可知，位置置换产生的信息损失

的最大值随 k 值的增加波动较大，将两图中最大的位置置换信息损失作为 Ω 的取值，即 Ω=9.867 5e^9。实验中 Rt 取 100 000，Rs 取 10^{12}，是为了确保时间阈值和空间阈值足够大，不会由于阈值过小造成额外的信息损失。

图 17~图 22 分别描述了 2 个算法在 Rt=100 000，$maxradius$=178.95，$Max\text{-}Trash$=10 时，MDAV 算法和 GC-DM 算法在 k 值和 Rs 变化下删除轨迹的比例、删除位置的比例以及总的信息损失。

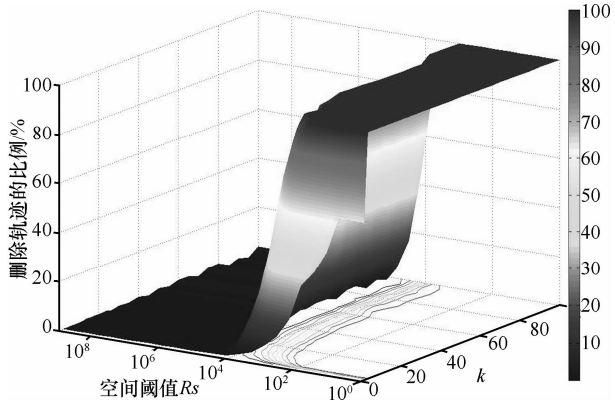


图 17　MDAV 算法中 k 值和 Rs 变化下删除轨迹的比例

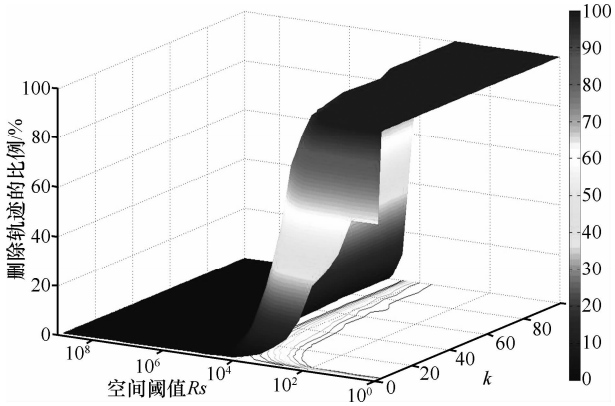


图 18　GC-DM 算法中 k 值和 Rs 变化下删除轨迹的比例

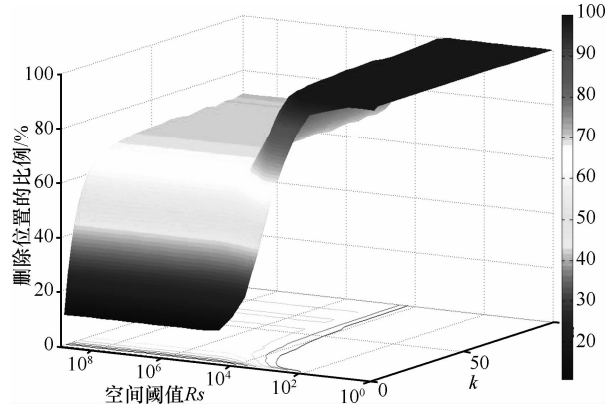


图 19　MDAV 算法中 k 值和 Rs 变化下删除位置的比例

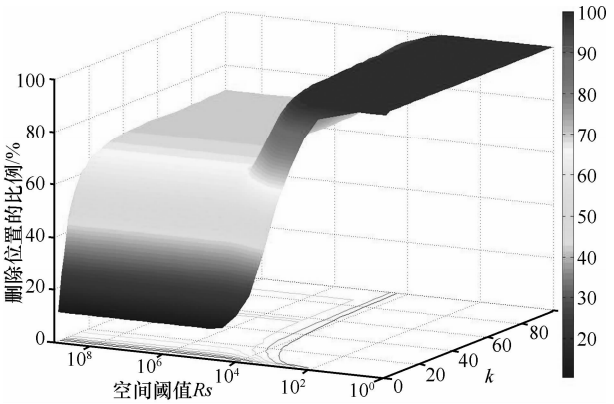


图 20　GC-DM 算法中 k 值和 Rs 变化下删除位置的比例

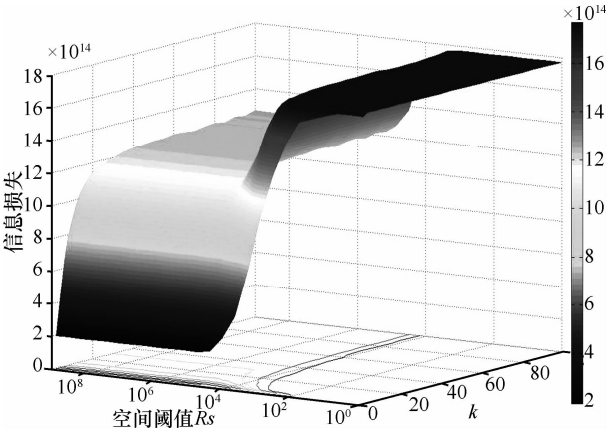


图 21　MDAV 算法中 k 值和 Rs 变化下的信息损失

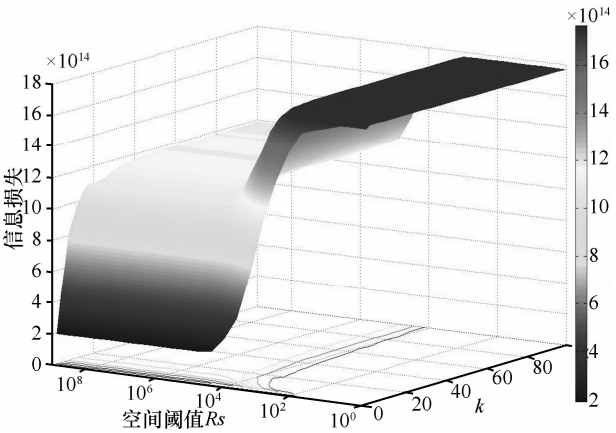


图 22　GCDM 算法中 k 值和 Rs 变化下的信息损失

通过观察图 17~图 22 可以发现，对于同样的 k 和 Rs，GC-DM 产生的信息损失小于 MDAV 算法，即 GC-DM 比 MDAV 算法产生更高的可用性。因为 GC-DM 算法的轨迹距离度量标准同时考虑了轨迹的位置距离和形状距离，能够更加贴切地衡量 2 个轨迹间的距离，而且使用的贪婪聚类技术使轨迹等价类内部轨迹相似性更大，类间轨迹相似性更小，之后使用数据“面罩”技术形成位置等价类时，删

除的位置数量减少，因此产生较小的信息损失。

8.2.2 执行时间比较

图 23 和图 24 分别描述了在 *Rt*=100 000，*maxradius*= 178.95，*Max-Trash*=10 时，MDAV 算法和 GC-DM 算法在 *k* 值和 *Rs* 变化下的执行时间。

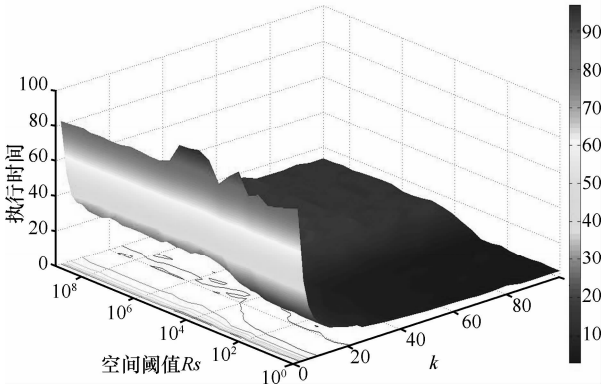


图 23 MDAV 算法中 *k* 值和 *Rs* 变化下的执行时间

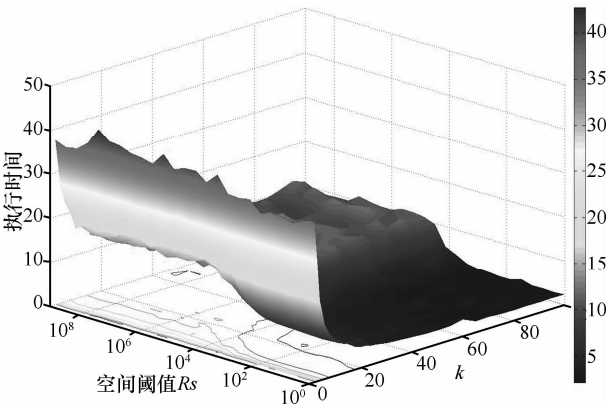


图 24 GC-DM 算法中 *k* 值和 *Rs* 变化下的执行时间

通过观察图 23 和图 24 可以发现，在同样的实验条件下，GC-DM 算法比 MDAV 算法花费更少的执行时间，尤其在 *k* 取值增加时，执行效率的体现尤为明显。主要是因为随着 *k* 值的增加，满足 *k* 匿名需要的轨迹增加，造成迭代次数降低，减少了执行时间。

通过在合成数据集和真实数据集上的实验可以发现，提出的 GC-DM 算法在可用性和执行时间上均要优于 MDAV 算法，因此 GC-DM 算法是可行的。

9 结束语

针对发布的时空轨迹数据可能遭遇的隐私保护问题，首先引入了轨迹 *k*-匿名的概念，并提出了一种新的轨迹相似性度量模型，不仅考虑了轨迹的时间和空间要素，更加入了轨迹的形状因素，可以在多项式时间内计算轨迹距离，并且可以计算定义在不同时间跨度上的轨迹距离，能够更加准确、快速地度量轨迹之间的相似性；在此基础上，提出了一种基于轨迹位置形状相似性的隐私保护算法，在轨迹聚类中使用真实的原始位置信息形成数据“面罩”，并满足轨迹 *k*-匿名，在有效地保护轨迹数据的同时，显著地提高了轨迹数据的可用性；最后，在合成轨迹数据集和真实轨迹数据集上的实验结果表明，本文提出的算法花费了更少的时间代价，具有更高的数据可用性。

参考文献：

[1] 韩建民,于娟, 虞慧群等. 面向数值型敏感属性的分级 *l*-多样性模型[J].计算机研究与发展, 2011,48(1):147-158.
HAN J M, YU J, YU H Q, *et al*. A multi-level *l*-diversity model for numerical sensitive attributes[J]. Journal of Computer Research and Development, 2011, 48(1): 147-158.

[2] 韩建民, 岑婷婷, 虞慧群. 数据表 *k*-匿名化的微聚集算法研究[J]. 电子学报, 2008,36(11) : 2021-2029.
HAN J M, CEN T T, YU H Q. Research in micro aggregation algorithm for *k*-anonymization[J]. Chinese Journal of Electronics, 2008, 36(11) : 2021-2029.

[3] 杨高明, 杨静, 张健沛. 半监督聚类的匿名数据发布[J]. 电子学报, 2011, 32(11):1489-1494.
YANG G M, YANG J, ZHANG J P. Semi-supervised clustering-based anonymous data publishing[J]. Chinese Journal of Electronics, 2011, 32(11):1489-1494.

[4] 杨静, 王波. 一种基于最小选择度优先的多敏感属性个性化 *l*-多样性算法[J]. 计算机研究与发展, 2012, 49(9): 2603-2610.
YANG J, WANG B. Personalized *l*-diversity algorithm for multiple sensetive attributes based on minimum selected degree first[J]. Journal of Computer Research and Development, 2012, 49(9): 2603-2610.

[5] 王波, 杨静. 一种基于逆聚类的个性化隐私匿名方法[J]. 电子学报, 2012, 40(5):883-890.
WANG B, YANG J. A personalized privacy anonymous method based on inverse clustering[J]. Chinese Journal of Electronics, 2012, 40(5): 883-890.

[6] 周水庚, 李丰, 陶宇飞等. 面向数据库应用的隐私保护研究综述[J]. 计算机学报, 2009, 32(5): 847-861.
ZHOU S G, LI F, TAO Y F, *et al*. Privacy preservation in database applications: a survey[J]. Chinese Journal of Computers, 2009, 32(5): 847-861.

[7] 熊平, 朱天清. 基于杂度增益与层次聚类的数据匿名方法[J].计算机研究与发展, 2012,49(7):1545-1552.
XIONG P, ZHU T Q. A data anonymization approach based on impurity gain and hierarchical clustering[J]. Journal of Computer Research and Development, 2012,49(7):1545-1552.

[8] SAMARATI P, SWEENEY L. Protecting privacy when disclosing information: *k*-anonymity and its enforcement through generalization

and suppression[A]. Proceedings of the IEEE Symposium on Research in Security and Privacy[C]. Paloalto, CA: IEEE, 1998.1-19.

[9] SWEENEY L. *k*-anonymity: a model for protecting privacy[J]. International Journal on Uncertainty, Fuzziness and Knowledge-Based Systems, 2002, 10(5): 557-570.

[10] DOMINGO-FERRER J, SRAMKA M, TRUJILLO-RASÚA R. Privacy-preserving publication of trajectories using microaggregation[A]. Proceedings of the SIGSPATIAL ACM GIS 2010 International Workshop on Security and Privacy in GIS and LBS, SPRINGL 2010[C]. San Jose, California, USA, ACM, 2010.

[11] 袁冠, 夏士雄, 张磊等. 基于结构相似度的轨迹聚类算法[J]. 通信学报, 2011, 32(9): 103-110.
YUAN G, XIA S X, ZHANG L, *et al*. Trajectory clustering algorithm based on structural similarity[J]. Journal on Communications, 2011, 32(9): 103-110.

[12] ABUL O, BONCHI F, NANNI M. Never walk alone: uncertainty for anonymity in moving objects databases[A]. Proceedings of the IEEE International Conference on Data Engineering[C]. Cancun: IEEE, 2008.376-385.

[13] ABUL O, BONCHI F, NANNI M. Anonymization of moving objects databases by clustering and perturbation[J]. Information Systems, 2010, 35(8): 884-910.

[14] NERGIZ M E, ATZORI M, SAYGIN Y, *et al*. Towards trajectory anonymization: a generalization-based approach[J]. Transactions on Data Privacy, 2009, 2(1):47-75.

[15] NERGIZ M E, ATZORI M, SAYGIN Y. Towards trajectory anonymization: a generalization-based approach[A]. Proceedings of the SIGSPATIAL ACM GIS 2008 International Workshop on Security and Privacy in GIS and LBS[C]. California, USA, ACM, 2008. 52-61.

[16] HUO Z, HUANG Y, MENG X. History trajectory privacy-preserving through graph partition[A]. Proceedings of the First International Workshop on Mobile Location-Based Service[C]. Beijing: China, ACM, 2011.71-78.

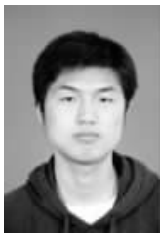
[17] HUO Z, MENG X, HU H, *et al*. You can walk alone: trajectory privacy-preserving through significant stays protection[A]. Proceedings of the 17th International Conference on Database Systems for Advanced Applications (DASFAA2012)[C]. Busan, South Korea, 2012. 351-366.

[18] JOSEP D F, ROLANDO T R. Micro aggregation and permutation-based anonymization of movement data[J]. Information Sciences 2012, 208: 55-80.

[19] FLOYD R W. Algorithm 97: shortest path[J]. Communications of the ACM, 1962, 5(6):345-350.

[20] PIORKOWSKI M, SARAFIJANOVOC-DJUKIC N, GROSSGLAUSER M. A parsimonious model of mobile partitioned networks with clustering[A]. The First International Conference on Communication Systems and Network (COMSNETS)[C]. Bangalore, India, 2009.

作者简介：



王超（1988-），男，河北沧州人，哈尔滨工程大学博士生，主要研究方向为数据库与知识工程、数据挖掘、隐私保护。

杨静（1962-），女，黑龙江哈尔滨人，哈尔滨工程大学教授、博士生导师，主要研究方向为数据库与知识工程、数据挖掘、隐私保护、软件理论等。

张健沛（1956-），男，黑龙江哈尔滨人，哈尔滨工程大学教授、博士生导师，主要研究方向为企业智能计算、数据库与知识工程、数据挖掘、社会网络、软件理论等。