一种基于单点收益的轨迹隐私保护方法

陈传明,林文诗,俞庆英,罗永龙
(1. 安徽师范大学计算机与信息学院,安徽芜湖 241002;2. 网络与信息安全安徽省重点实验室,安徽芜湖 241002)

摘 要: 如何在轨迹数据发布时保护用户隐私信息并且最大程度地减少数据损失是隐私保护研究领域的一个重要课题. 本文提出一种基于单点收益的轨迹隐私保护方法,在满足用户隐私要求的情况下,根据收益计算结果,在轨迹数据集中抑制位置点或者添加假轨迹,保证每次处理轨迹数据集时能达到最大收益,从而减少信息损失. 理论分析和实验结果表明,在隐私容忍度要求较高或者攻击者数量较多的情况下,本文方法能在保证隐私保护强度前提下有效降低数据损失率.

关键词: 假轨迹;隐私保护;单点收益;轨迹发布;轨迹抑制;问题点对
中图分类号: TP309.2 **文献标识码:** A **文章编号:** 0372-2112 (2020)01-0143-10
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2020.01.018

A Trajectory Privacy-Preserving Method Based on Single Point Gain

CHEN Chuan-ming, LIN Wen-shi, YU Qing-ying, LUO Yong-long
(1. *School of Computer and Information, Anhui Normal University, Wuhu, Anhui* 241002, *China*;
2. *Anhui Provincial Key Laboratory of Network and Information Security, Wuhu, Anhui* 241002, *China*)

Abstract: How to preserve users' privacy information and minimize the loss of information when publishing trajectory data has become an important topic in the research field of privacy preservation. In this paper, we propose a trajectory privacy-preserving method based on single point gain, which satisfies the privacy requirements of users. According to the single point gain values, we suppress location points or add dummy trajectories into the trajectory dataset to ensure that the maximum benefit can be achieved at each iteration, thus reducing the loss of information. Theoretical analysis and experimental results show that, in the case of high privacy tolerance or a large number of attackers, the proposed method effectively reduces the information loss rate while guaranteeing the intensity of privacy preserving.

Key words: dummy trajectory; privacy preservation; single point gain; trajectory publication; trajectory suppression; problematic point pair

1 引言

直接发布轨迹数据会对用户的隐私构成不可忽视的威胁[1~3],常用轨迹隐私保护策略会降低数据的可用性,如何在用户隐私容忍范围内提升数据效用是轨迹隐私保护研究中亟待解决的重要课题之一[4~7]. 轨迹数据发布中的隐私保护研究主要有基于扰动的方法和基于抑制的方法[8~11]. Lei 等人[12]提出的 k-交叉轨迹生成算法考虑了用户移动模式,使假轨迹与真实轨迹更加难以区分,但是在满足用户个性化需求方面还有不足;针对此问题,Shen 等人[13]以用户为中心,将攻防问题形式化为贝叶斯 Stackelberg 博弈并进行定量分析,使隐私保护模型更好地满足用户需求. 赵等人[14]采用基于频率的局部与全局位置抑制方法对轨迹数据集进行匿名处理,Chen 等人[15]使用局部抑制法保留轨迹数据集中位置、时间以及频繁序列,都提升了数据利用率. 上述研究主要通过单一的轨迹匿名处理方式实现轨迹数据发布时的隐私保护,数据损失率有待进一步降低,同时还需要根据用户的隐私需求,平衡隐私保护强度和数据效用之间的关系. 因此,为了在满足用户隐私需求的同时提高匿名数据的可用性,本文结合假轨迹法和抑制法提出一种基于单点收益的轨迹隐私保护方法 SPG (privacy-preserving method based on Single Point Gain),主要解决以下两个问题:

收稿日期:2018-04-18;修回日期:2019-09-19;责任编辑:蓝红杰
基金项目:国家自然科学基金(No. 61702010, No. 61972439, No. 61672039)

(1)如何更好地满足用户隐私需求;

(2)如何提高匿名数据的可用性.

首先,根据用户需求找到可能泄露隐私的问题轨迹集合;其次,比较抑制和扰动两种匿名处理方法能获得的单点收益大小;最后,选择单点收益较大的匿名处理方法进行轨迹匿名处理.在大规模用户合成轨迹数据集上的实验结果表明,本文所提方法一方面能够在满足用户隐私容忍度的情况下,实现匿名轨迹集合中不存在泄露隐私的轨迹;另一方面能在确保隐私安全的情况下降低数据损失率.

2 问题定义

定义 1 轨迹记录:设 L 是所有位置点的集合,定义 L 中的位置点以时间顺序组成的一个序列为一条轨迹记录(简称为轨迹).设 t_i 是轨迹集合 T 中的第 i 条轨迹记录,其长度为 len,可表示为 $t_i=\mathrm{loc}_1^i\rightarrow\mathrm{loc}_2^i\rightarrow\cdots\rightarrow\mathrm{loc}_{\mathrm{len}}^i$,其中 $\mathrm{loc}_j^i\in L(j=1,\cdots,\mathrm{len})$.

表 1 列出由 8 条轨迹记录构成的轨迹集合 T,定义在 $L=\{a_1,a_2,a_3,b_1,b_2,b_3\}$ 上,其中 a_1,a_2,a_3 表示同一类场所的签到信息,b_1,b_2,b_3 表示另一类场所的签到信息.本例中轨迹记录 t_1 的长度为 3.

表 1 轨迹集合 T

编号	轨迹
t_1	$a_1\rightarrow b_1\rightarrow b_2$
t_2	$a_1\rightarrow b_1\rightarrow a_2\rightarrow b_2\rightarrow a_3$
t_3	$a_1\rightarrow b_2\rightarrow b_3\rightarrow a_3$
t_4	$a_2\rightarrow b_2\rightarrow b_3\rightarrow a_3$
t_5	$a_3\rightarrow a_1\rightarrow b_1$
t_6	$b_2\rightarrow b_1\rightarrow a_1\rightarrow a_3$
t_7	$b_1\rightarrow b_2\rightarrow a_2\rightarrow a_3$
t_8	$a_2\rightarrow a_3\rightarrow b_1\rightarrow b_3$

定义 2 子轨迹:假设存在两条定义在 L 上的轨迹记录 $t_1=\mathrm{loc}_1^1\rightarrow\cdots\rightarrow\mathrm{loc}_m^1$ 和 $t_2=\mathrm{loc}_1^2\rightarrow\cdots\rightarrow\mathrm{loc}_k^2$,若 $m>k$,并且存在一种映射 g 使得 $\mathrm{loc}_1^2=\mathrm{loc}_{g(1)}^1,\cdots,\mathrm{loc}_k^2=\mathrm{loc}_{g(k)}^1$,$g(1)<g(2)<\cdots<g(k)$,则称 t_2 为 t_1 的一条子轨迹.

例如,对于轨迹记录 $t=a_1\rightarrow b_1\rightarrow b_2$,它的 6 条子轨迹分别为 $a_1,b_1,b_2,a_1\rightarrow b_1,b_1\rightarrow b_2,a_1\rightarrow b_2$.

定义 3 掌握位置:攻击者可以追踪到的所有位置信息,即攻击者所拥有的背景知识.设有攻击者集合 $\mathrm{Advs}=\{\mathrm{Adv}_1,\mathrm{Adv}_2,\cdots,\mathrm{Adv}_m\}$,$\forall\,\mathrm{Adv}_i,\mathrm{Adv}_j\in\mathrm{Advs}(i\neq j)$,攻击者 Adv_i 可以追踪到的位置信息集合记为 L^{Adv_i},即 Adv_i 的掌握位置,其中 $L^{\mathrm{Adv}_i}\cap L^{\mathrm{Adv}_j}=\varnothing,\bigcup_{i=1}^{m}L^{\mathrm{Adv}_i}=L$.

例如,定义 1 中 $L=\{a_1,a_2,a_3,b_1,b_2,b_3\}$ 包含两类位置信息,攻击者 A 的掌握位置 $L^A=\{a_1,a_2,a_3\}$,攻击者 B 的掌握位置 $L^B=\{b_1,b_2,b_3\}$,满足.$L^A\cap L^B=\varnothing,L^A\cup L^B=L$.

定义 4 投影 $\mathrm{PR}_{\mathrm{Adv}}(t)$:已知攻击者 Adv 的掌握位置 L^{Adv},轨迹记录 $t=\mathrm{loc}_1\rightarrow\cdots\rightarrow\mathrm{loc}_{\mathrm{len}}$,$(1\leqslant i_1\leqslant\cdots\leqslant i_s\leqslant\cdots\leqslant\mathrm{len})$.若存在一条 t 的子轨迹 $\mathrm{loc}_{i_1}\rightarrow\cdots\rightarrow\mathrm{loc}_{i_s}$,满足 $\forall\,\mathrm{loc}_{i_r}\in L^{\mathrm{Adv}}(r=1,\cdots,s)$,则定义该条子轨迹为 t 相对于攻击者 Adv 的投影,记为 $\mathrm{PR}_{\mathrm{Adv}}(t)$.

例如,表 1 中 $t_1=a_1\rightarrow b_1\rightarrow b_2$,则 $PR_A(t_1)=a_1$,$\mathrm{PR}_B(t_1)=b_1\rightarrow b_2$.

定义 5 投影知识 T_{Adv}:设有轨迹集合 T 以及一个攻击者 Adv,则 T 相对于攻击者 Adv 的投影知识是指 T 中每条轨迹记录相对于 Adv 的投影构成的集合,记为 T_{Adv},即 $T_{\mathrm{Adv}}=\{\mathrm{PR}_{\mathrm{Adv}}(t)\mid t\in T\}$.

表 2 列出了表 1 所示轨迹集合 T 相对于攻击者 A 的投影知识 T_A.

表 2 轨迹集合 T 相对于攻击者 A 的投影知识 T_A

编号	$PR_A(t_i)$
t_1	a_1
t_2	$a_1\rightarrow a_2\rightarrow a_3$
t_3	$a_1\rightarrow a_3$
t_4	$a_2\rightarrow a_3$
t_5	$a_3\rightarrow a_1$
t_6	$a_1\rightarrow a_3$
t_7	$a_2\rightarrow a_3$
t_8	$a_2\rightarrow a_3$

定义 6 投影相同的轨迹集合 $S(t^A)$:设 t^A 是轨迹集合 T 中的某一条轨迹相对于攻击者 A 的投影,那么,投影相同的轨迹集合指的是投影与 t^A 相同的轨迹集合,可表示为 $S(t^A)=\{t\mid t\in T,\mathrm{PR}_A(t)=t^A\}$.

例如,当 $t^A=a_1\rightarrow a_3$ 时,$S(a_1\rightarrow a_3)=\{t_3,t_6\}$.

根据 t^A,攻击者 A 能够以一定概率推测出 A 不掌握的位置点信息.例如,当 $t^A=a_1$ 时,攻击者 A 能够根据该投影 a_1 以一定的概率 P 推测出位置点 b_1,b_2,概率 P 的计算公式如定义 7 所示.

定义 7 推断概率 $P(x,t^{\mathrm{Adv}})$:攻击者 A 根据 $S(t^{\mathrm{Adv}})$ 推断出其他位置 x 的概率,定义为

$$P(x,t^{\mathrm{Adv}})=\frac{\mathrm{S\_ack}(x,t^{\mathrm{Adv}})}{|S(t^{\mathrm{Adv}})|}\tag{1}$$

其中,$\mathrm{S\_ack}(x,t^{\mathrm{Adv}})=|\{x\mid x\notin L^{\mathrm{Adv}},x\in t,t=S(t^{\mathrm{Adv}})\}|$ 指的是投影相同的轨迹集合 $S(t^{\mathrm{Adv}})$ 所含轨迹记录中不属于 L^{Adv} 的位置点 x 的个数.

为了使匿名处理后的数据能在一定程度上具有隐私保护性,给出以下定义.

定义 8　隐私容忍度 P_{br}[11]:轨迹隐私被披露的概率,记为 P_{br}. 根据用户隐私需求,由用户自己定义. 攻击者不能以超过 P_{br}的概率推断出不属于攻击者知识背景的信息.

定义 9　问题点对:设有轨迹集合 T,x 是 T 中的一个位置点,t^A是 T 中某一条轨迹 t 相对于攻击者 A 的投影,则称 x 和 t^A的组合为一个点对,记作 $\mathrm{pp}(x,t^A)$. 如果攻击者 A 根据 $S(t^A)$推断出其他位置的概率 $P(x,t^A)$大于隐私容忍度 P_{br},则称 $\mathrm{pp}(x,t^A)$为一个问题点对.

定义 10　可安全发布的轨迹数据集:对于指定的隐私容忍度 P_{br},如果经过匿名处理后的轨迹数据集中不存在任何问题点对,则称该匿名轨迹数据集是可安全发布的.

3　基于单点收益的轨迹隐私保护算法

3.1　算法思想

本文提出一种基于单点收益的轨迹隐私保护算法 SPG,具体包含两个步骤:(1)问题点对筛选,即找出原始轨迹数据集中潜在的违反用户隐私容忍度的问题点对;(2)基于单点收益的轨迹匿名处理,对即将发布的轨迹数据,按照特定方法(添加假轨迹或者抑制轨迹)计算问题点对的收益值,选择收益最大的问题点对并记录其对应方法,使用该方法进行轨迹匿名处理,保证每次处理的都是最佳问题点对,从而最大程度地减小数据损失. 3.2 节和 3.3 节分别实现这两个步骤,SPG 算法整体流程如图 1 所示.

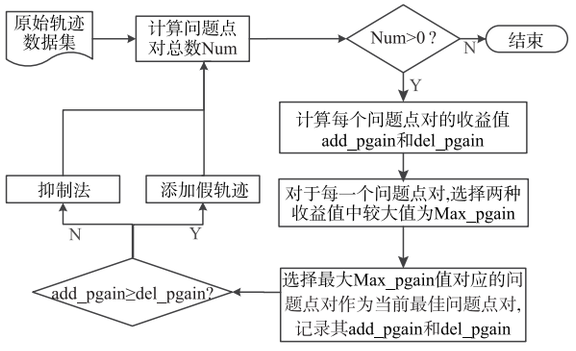


图1　SPG算法流程图

3.2　问题点对筛选

找出原始数据集中潜在的违反用户隐私容忍度的问题点对,通过算法 1 实现,该算法思想参考了 Terrovitis 等人[10]提出的 THREATID 算法. 输入包括原始轨迹集合 T,隐私容忍度 P_{br}以及攻击者所具备的背景知识;输出包括轨迹集合中所有问题点对的集合 Q_p(一张二维表)以及问题点对总数 Num. 问题点对筛选算法伪代码如算法 1 所示.

算法 1　问题点对筛选算法 AGPBR

输入:T,P_{br},Advs
输出:Q_p,Num
1:Initialize $Q_p=\varnothing$,Num = 0;
2:for each $t\in T$ do
3:　for each adversary Adv do
4:　　if(~isempty($\mathrm{PR}_{Adv}(t)$)) then
5:　　Calculate the number of $x\in t\backslash \mathrm{PR}_{Adv}(t)$;
6:　　Save as S_ack(x,t^{Adv});
7:　　end if
8:　end for
9:end for
10:for each adversary Adv do
11:　for each $x\in L\backslash L^{Adv}$ and each $t^{Adv}\in T_{Adv}$ do
12:　　if S_ack(x,t^{Adv})/|$S(t^{Adv})$| > P_{br} then
13:　　Push pp(x,t^{Adv}) into list Q_p;
14:　　Num = Num + S_ack(x,t^{Adv});
15:　　end if
16:end for
17:end for
返回:Q_p,Num;

假定表 1 所示的轨迹集合 T 为算法 1 的输入,P_{br} = 0.5. 按照算法执行顺序,首先考虑 $t^{Adv}=a_1$,轨迹 $t_1=a_1\rightarrow b_1\rightarrow b_2$,$t_1$相对于攻击者 A 的投影为 a_1,即 $\mathrm{PR}_A(t_1)=a_1$. t_1中 b_1和 b_2不属于 L^A,所以 S_ack(b_1,a_1) = 1,S_ack(b_2,a_1) = 1. 同理,考虑轨迹 t_2,t_3等直到获得所有 S_ack(x,t^{Adv})的值. 然后根据式(1)计算 $P(x,t^{Adv})$的值,例如第一个点对 pp(b_1,a_1),有:

$$P(b_1,a_1)=\frac{\mathrm{S\_ack}(b_1,a_1)}{|S(a_1)|}=1>P_{br}$$

所以,pp(b_1,a_1)是问题点对,将它加入 Q_p,Num 加上 S_ack(b_1,a_1)的值 1. T 中的问题点对如表 3 所示.

表 3　T 中的问题点对(P_{br} = 0.5)

pp(x,t^{Adv})	S_ack(x,t^{Adv})	|$S(t^{Adv})$|	$P(x,t^{Adv})$
(b_1,a_1)	1	1	1
(b_2,a_1)	1	1	1
($b_1,a_1\rightarrow a_2\rightarrow a_3$)	1	1	1
($b_2,a_1\rightarrow a_2\rightarrow a_3$)	1	1	1
($b_2,a_1\rightarrow a_3$)	2	2	1
($b_1,a_2\rightarrow a_3$)	2	3	0.66
($b_2,a_2\rightarrow a_3$)	2	3	0.66
($b_3,a_2\rightarrow a_3$)	2	3	0.66
($b_1,a_3\rightarrow a_1$)	1	1	1
(a_1,b_1)	1	1	1
(a_3,b_1)	1	1	1
($a_1,b_1\rightarrow b_2$)	2	3	0.66

续表

pp(x,t^{Adv})	S_ack(x,t^{Adv})	$\|S(t^{Adv})\|$	$P(x,t^{Adv})$
(a_2,$b_1\to b_2$)	2	3	0.66
(a_3,$b_1\to b_2$)	2	3	0.66
(a_2,$b_1\to b_3$)	1	1	1
(a_3,$b_1\to b_3$)	1	1	1
(a_1,$b_2\to b_1$)	1	1	1
(a_3,$b_2\to b_1$)	1	1	1
(a_3,$b_2\to b_3$)	2	2	1

3.3 基于单点收益的轨迹匿名处理

按照特定方法(添加假轨迹法或者抑制法)计算问题点对的收益值,选择收益最大的问题点对,并使用获取该收益值的对应方法进行轨迹匿名处理.

定义 11 收益 gain:收益指的是处理当前轨迹能消除的问题点对数 Num′与处理之前的问题点对数 Num 的比值,定义为

$$\text{gain}=\frac{\text{Num}'}{\text{Num}} \tag{2}$$

定义 12 损失 cost:损失是指添加轨迹或者删除点之后所产生的位置信息的丢失量,记为 cost.

例如,如果添加一条轨迹投影 $a_1\to a_2\to a_3$,那么损失值 cost = 1 × 3 = 3;如果将轨迹投影 $a_1\to a_2\to a_3$ 与 a_1 统一,需要抑制 a_2 和 a_3,cost = 2 × $|S(a_1\to a_2\to a_3)|$.

为了平衡收益和信息损失,使得当前处理的问题点对是本次处理的最优问题点对,本文提出了单点收益的概念.

定义 13 单点收益 pgain:处理每一个点带来的收益称为单点收益,记为 pgain,计算方法如式(3)所示.

$$\text{pgain}=\frac{\text{gain}}{\text{cost}} \tag{3}$$

基于损失和收益的计算,轨迹匿名处理具体包含 3 个部分:(1)基于假轨迹方法的单点收益计算;(2)基于抑制法的单点收益计算;(3)基于 pgain 值的轨迹匿名处理.

3.3.1 添加假轨迹算法

根据用户所需的隐私容忍度 P_{br},以及 3.2 节得到的问题点对,在原轨迹集合中添加假轨迹使得 $P(x,t^{Adv})=P_{br}$,将需要添加的最少轨迹条数记为 addline. 计算添加 addline 条投影轨迹后能使其余问题点对消除的数量,即添加的轨迹能使其余问题点对满足 $P(x,t^{Adv})\leqslant P_{br}$. 需要消除的问题点对数量由两个部分组成:(1)自身的问题点对数量;(2)能消除的其他问题点对数量. 保存能消除的问题点对总数.

算法 2 添加假轨迹算法 Addgain

输入:t^{Adv},P_{br},pp(x,t^{Adv})的总个数 n,Num
输出:add_pgain

```
1:Initialize add_pgain = 0;
2:addline = ceil(n/P_br) - |S(t^Adv)|;//需要添加的轨迹数
3:add_gain = n;
4:for each t^Adv in Q_p do // Q_p 中有相同投影的问题点对
5:   p = S_ack(x,t^Adv)/(|S(t^Adv)| + addline);
6:   if p≤P_br then
7:      add_gain = add_gain + S_ack(x,t^Adv);
8:   end if
9:end for
10:add_gain = add_gain/Num;
11:add_cost = length(t^Adv) × addline;
12:add_pgain = add_gain/add_cost;
返回:add_pgain;
```

以 Q_p 中第一个问题点对(b_1,a_1)为例,$t^A=a_1$,$n=1$,$|S(t^A)|=1$,addline = ceil(n/P_{br}) − $|S(t^A)|=1$;点对(b_2,a_1)具有相同的投影,当增加 1 条投影 t^A 时,$P(b_2,a_1)=1/2=0.5=P_{br}$,该问题点对也被消除,add_gain = 1 + 1 = 2,Num = 27,最后返回收益值 add_gain = 2/27 = 0.0741;添加该投影带来的损失 add_cost = 1 × 1 = 1;可得单点收益 add_pgain = add_gain/add_cost = 0.0741.

3.3.2 抑制算法

对于每一个问题点对对应的轨迹投影 t_R^{Adv},如果存在子轨迹 t_r^{Adv},则将 t_r^{Adv} 与 t_R^{Adv} 统一为 t_r^{Adv},记录 $D=\{x\,|\,x\in t_R^{Adv}$ 且 $x\notin t_r^{Adv}\}$ 为统一后删除的点. 同样地,计算删除 D 后能使其余问题点对消除的数量,即删除的点能使得其余问题点对满足 $P(x,t^{Adv})\leqslant P_{br}$. 消除的所有问题点对数包含三个部分:(1)自身的问题点对数;(2)子轨迹中的问题点对数;(3)其他攻击者对应的问题点对数.

如果当前处理的投影有多条子轨迹,选择收益最大的子轨迹进行统一. 如果不存在子轨迹,则计算删除当前投影的收益值.

算法 3 轨迹抑制算法 Delgain

输入:t^{Adv},P_{br},Num
输出:del_pgain,D

```
1:Initialize del_pgain = 0, max_deln = 0, D = ∅;
2:for each t^Adv in Q_p do
3:   Find all sub-trajectory projections of t^Adv and insert them into a dataset R_p;
4:end for
5:if (isempty(R_p)) then
6:   D = t^Adv;
7:   Calculate the number deln of problematic point pairs that have been solved;
8:   max_deln = deln;
9:else
10:   for each sub-trajectory projection t_r^Adv of t^Adv do
11:      Unify t^Adv with t_r^Adv;
12:      Calculate the number deln of problematic point pairs that have
```

```
        been solved;
13:     if deln > max_deln
14:         max_deln = deln;
15:       D = {x | x ∈ t^Adv 且 x ∉ t_r^Adv};//x 代表位置点
16:     end if
17:  end for
18:end if
19:del_gain = max_deln/Num;
20:del_cost = length(D) × the number of influenced projections;
21:del_pgain = del_gain/del_cost;
返回:del_pgain,D;
```

例如,对于第一个问题点对,$t^{Adv}=a_1$,Q_p中不存在a_1的子轨迹,则将a_1加入集合D中. 抑制a_1后可以发现问题点对(b_1,a_1)和(b_2,a_1)均被消除,另外点对$(a_1,b_1\rightarrow b_2)$中,原a_1个数为2,消除一个后为1,$P(a_1,b_1\rightarrow b_2)=1/3=0.33<0.5=P_{br}$,问题点对$(a_1,b_1\rightarrow b_2)$消除. 易得 $del\_gain=4/27=0.1481$, $del\_cost=1$, $del\_pgain=0.1481$.

3.3.3　基于最大单点收益的轨迹匿名算法

根据添加假轨迹收益算法和轨迹抑制收益算法,得到每个问题点对添加假轨迹和删除位置点的单点收益,选择两者中数值大的值作为该点对的最大收益值Max_pgain,计算Q_p中每个问题点对的最大收益值Max_pgain[pp(x,t^{Adv})],选择Q_p中Max_pgain最大的点对作为本次处理的点对. 如果add_pgain≥del_pgain,选择添加假轨迹的方法,否则,选择抑制法.

算法4　单点收益最大匿名算法 PgainMax

```
输入:Q_p,Num,P_br
输出:匿名后轨迹数据集 T
1:(Q_p,Num) = AGPBR(T,P_br);
2:while Num > 0
3:   for each t^Adv of problematic point pairs pp in Q_p do
4:   n = the number of problematic point pair pp;
5:    add_pgain = Addgain(t^Adv,P_br,n,Num);
6:    (del_pgain,D) = Delgain(t^Adv,P_br,Num);
7:    Max_pgain(pair_i) = max(add_pgain,del_pgain);
8:  end for
9:Choose problematic point pair with maximum value of Max_pgain as current optimal processing point pair;
10:if add_pgain≥del_pgain then
11:      Insert into T with t^Adv of the number addline;
12:else
13:      Delete D of the trajectories from T;
14:end if
15:(Q_p,Num) = AGPBR(T,P_br);
16:end while
返回:匿名后轨迹数据集 T;
```

通过算法1得到问题点对集合Q_p和问题点对数Num,每次选择单点收益最大的问题点对进行处理,Num=0时算法结束. 每次迭代选择Max_pgain最大的问题点对进行匿名处理,当add_pgain≥del_pgain时,选择添加假轨迹的方法,反之,选择抑制法. 例如,通过算法2和算法3,第一次迭代后每个问题点对的各项值如表4所示;选择Max_pgain最大的点对$(a_2,b_1\rightarrow b_3)$处理,$b_1\rightarrow b_3$与投影b_1统一,add_pgain<del_pgain,故抑制点b_3,抑制后的结果如表5所示;最终得到的匿名轨迹集合如表6所示.

表4　第一次迭代后问题点对各项值

pp(x,t^{Adv})	add_pgain	del_pgain	Max_pgain
(b_1,a_1)	0.0741	**0.1481**	0.1481
(b_2,a_1)	0.0741	**0.1852**	0.1852
$(b_1,a_1\rightarrow a_2\rightarrow a_3)$	0.0247	**0.1481**	0.1481
$(b_2,a_1\rightarrow a_2\rightarrow a_3)$	0.0247	**0.1481**	0.1481
$(b_2,a_1\rightarrow a_3)$	0.0185	**0.0926**	0.0926
$(b_1,a_2\rightarrow a_3)$	**0.1111**	0.0864	0.1111
$(b_2,a_2\rightarrow a_3)$	**0.1111**	0.0988	0.1111
$(b_3,a_2\rightarrow a_3)$	**0.1111**	0.1111	0.1111
$(b_1,a_3\rightarrow a_1)$	0.0185	**0.1111**	0.1111
(a_1,b_1)	0.0741	**0.1111**	0.1111
(a_3,b_1)	0.0741	**0.1481**	0.1481
$(a_1,b_1\rightarrow b_2)$	0.1111	**0.1481**	0.1481
$(a_2,b_1\rightarrow b_2)$	0.1111	**0.1481**	0.1481
$(a_3,b_1\rightarrow b_2)$	0.1111	**0.1481**	0.1481
$(\boldsymbol{a_2,b_1\rightarrow b_3})$	**0.0370**	**0.2222**	**0.2222**
$(a_3,b_1\rightarrow b_3)$	0.0370	**0.2222**	0.2222
$(a_1,b_2\rightarrow b_1)$	0.0370	**0.2222**	0.2222
$(a_3,b_2\rightarrow b_1)$	0.0370	**0.2222**	0.2222
$(a_3,b_2\rightarrow b_3)$	0.0185	**0.0556**	0.0556

表5　第一次迭代后的轨迹集合

编号	轨迹
t_1	$a_1\rightarrow b_1\rightarrow b_2$
t_2	$a_1\rightarrow b_1\rightarrow a_2\rightarrow b_2\rightarrow a_3$
t_3	$a_1\rightarrow b_2\rightarrow b_3\rightarrow a_3$
t_4	$a_2\rightarrow b_2\rightarrow b_3\rightarrow a_3$
t_5	$a_3\rightarrow a_1\rightarrow b_1$
t_6	$b_2\rightarrow b_1\rightarrow a_1\rightarrow a_3$
t_7	$b_1\rightarrow b_2\rightarrow a_2\rightarrow a_3$
$\boldsymbol{t_8}$	$\boldsymbol{a_2\rightarrow a_3\rightarrow b_1}$

表 6 匿名轨迹集合

编号	轨迹
t_1'	$b_1 \rightarrow b_2$
t_2'	$a_1 \rightarrow b_1 \rightarrow b_2 \rightarrow a_3$
t_3'	$a_1 \rightarrow b_2 \rightarrow b_3 \rightarrow a_3$
t_4'	$a_2 \rightarrow b_2 \rightarrow b_3 \rightarrow a_3$
t_5'	$a_3 \rightarrow a_1$
t_6'	$a_1 \rightarrow a_3$
t_7'	$b_1 \rightarrow b_2 \rightarrow a_2 \rightarrow a_3$
t_8'	$a_2 \rightarrow a_3$
t_9'	$a_1 \rightarrow a_3$
t_{10}'	$a_2 \rightarrow a_3$
t_{11}'	$b_1 \rightarrow b_2$
t_{12}'	$b_2 \rightarrow b_3$
t_{13}'	$b_2 \rightarrow b_3$

3.4 算法分析

3.4.1 算法复杂度分析

算法 1 的时间复杂度主要依赖于:(1)计算 S_ack 的代价,查找每条轨迹中不属于攻击者知识背景的位置,得到 S_ack;(2)计算问题点对 pp 的代价. 两次扫描轨迹集合的时间复杂度为 $O(2\times m\times n)$,m 为攻击者个数,是常数项,因此算法 1 时间复杂度为 $O(n)$.

算法 2 的时间复杂度主要依赖于计算添加假数据后单点收益值的代价. 扫描一遍轨迹投影集得到添加假数据的收益值,时间复杂度为 $O(m\times n)$,m 为攻击者个数,因此算法 2 时间复杂度为 $O(n)$.

算法 3 的时间复杂度主要依赖于搜索最合适的子轨迹统一轨迹投影. 扫描一遍投影集找到所有子轨迹,再对每一条子轨迹进行分析,算法 3 时间复杂度为 $O(2\times n)$,即 $O(n)$.

算法 4 的时间复杂度主要依赖于问题点对数以及对上述各算法的调用. 算法 4 时间复杂度为 $O(n^2)$.

3.4.2 算法安全性分析

定理 匿名后的轨迹数据集不存在问题点对.

证明 首先,算法 1 根据用户隐私需求得到问题点对集合和问题点对总数,而算法 4 的迭代结束条件是问题点对总数 Num = 0. 每次迭代后都会重新计算当前轨迹集合中存在的问题点对总数,直到不存在问题点对. 所以算法结束后,轨迹集合中不存在能被攻击者利用的问题点对. 针对上文所举的例子,通过 SPG 算法最后得到的匿名轨迹集合如表 6 所示. 在 $P_{\mathrm{br}}=0.5$ 时,根据算法 1,对表 6 轨迹集合进行问题点对计算,结果如表 7 所示. 从表中结果可以看出不存在 $P(x,t^{\mathrm{Adv}})>P_{\mathrm{br}}$. 由此可推,在轨迹数据增多的情况下,本算法仍可确保匿名后的轨迹数据集中不存在问题点对.

表 7 匿名轨迹集合问题点对计算

$\mathrm{pp}(x,t^{\mathrm{Adv}})$	$\mathrm{S\_ack}(x,t^{\mathrm{Adv}})$	$\lvert S(t^{\mathrm{Adv}})\rvert$	$P(x,t^{\mathrm{Adv}})$
$(b_1,a_1\rightarrow a_3)$	1	4	0.25
$(b_2,a_1\rightarrow a_3)$	2	4	0.5
$(b_3,a_1\rightarrow a_3)$	1	4	0.25
$(a_1,b_1\rightarrow b_2)$	1	4	0.25
$(a_2,b_1\rightarrow b_2)$	1	4	0.25
$(a_3,b_1\rightarrow b_2)$	2	4	0.5
$(b_1,a_2\rightarrow a_3)$	1	4	0.25
$(b_2,a_2\rightarrow a_3)$	2	4	0.5
$(b_3,a_2\rightarrow a_3)$	1	4	0.25
$(a_1,b_2\rightarrow b_3)$	1	4	0.25
$(a_2,b_2\rightarrow b_3)$	1	4	0.25
$(a_3,b_2\rightarrow b_3)$	2	4	0.5

4 实验

4.1 实验平台和数据集

实验环境为 Intel core i7 处理器,CPU 3.3GHz,操作平台为 Windows 10. 通过 Matlab 2012 编程实现匿名算法. 实验选用 city80 数据集[11],该数据集的格式如表 1 所示,轨迹总数为 80000 条. 我们进行了两组实验,一组采用随机选取的 30000 条轨迹组成的轨迹集合 DS1;另一组采用随机选取的 15000 条轨迹组成的轨迹集合 DS2,DS1 和 DS2 中的轨迹平均长度均为 5.

4.2 效用评价指标

4.2.1 基于正确序列的保留评价

定义 14 轨迹保留比率 TR:逐点比对轨迹集合中每条轨迹匿名前后的位置点保留情况,将位置点保留比率记作 TR. 对于第 i 条轨迹,用 TR_{t_i} 表示其保留比率,计算方法如式(4)所示.

$$\mathrm{TR}_{t_i}=\frac{|\text{locations retained with original order in } t_i'|}{|t_i|}\times 100\% \tag{4}$$

定义 15 轨迹集合效用 STR:STR 用于评估轨迹数据集的整体轨迹保留比率. 对于轨迹集合中的每条匿名轨迹,STR 表示轨迹集合中满足 $\mathrm{TR}_{t_i}>\theta$ 的轨迹条数占轨迹总数的比率,计算方法如式(5)所示.

$$\mathrm{STR}=\frac{|\{\mathrm{TR}\,|\,\mathrm{TR}_{t_i}>\theta,0\leqslant i\leqslant |T|,0\leqslant\theta\leqslant 1\}|}{|T|}\times 100\% \tag{5}$$

定义 16 平均相对误差 AREL[10]:AREL 是指用于测量由于匿名化而错误查询的平均轨迹数.

令 μ_{be}^{i} 为原始轨迹集合中能检索到的包含第 i 条频

繁有序子序列轨迹的数量,μ_{af}^{i}是匿名轨迹集合中能检索到的包含第 i 条频繁有序子序列轨迹的数量. 对前 N 个最频繁的有序子序列进行查询,AREL 计算方法如式(6)所示.

$$\mathrm{AREL}=\frac{\sum_{i=1}^{N}(\mu_{be}^{i}-\mu_{af}^{i})/\mu_{be}^{i}}{N}\times 100\% \tag{6}$$

4.2.2　数据损失率

定义 17　数据损失率 TL:原始轨迹集合 T 中每条轨迹 t_i包含位置点数为$|t_i|$,其中 $1\leqslant i\leqslant|T|$,匿名轨迹集合 T'中每条轨迹 t_j包含位置点数$|t_j'|$,其中,$1\leqslant i\leqslant|T'|$. 数据损失率 TL 计算公式定义为

$$\mathrm{TL}=\frac{\left|\sum_{i=1}^{|T|}|t_i|-\sum_{j=1}^{|T|}|t_j'|\right|}{\sum_{i=1}^{|T|}|t_i|}\times 100\% \tag{7}$$

TL 值越小说明数据损失率越小.

4.2.3　平均位置出现比率

定义 18　平均位置出现比率 ξ[10]:设 L 是轨迹集合 T 中所有位置点的集合,对 $\forall l\in L$,r_o^l 为原始轨迹集合中 l 出现的次数,r_a^l 为匿名轨迹集合中 l 出现的次数,则平均位置出现比率 ξ 计算方法如式(8)所示.

$$\xi(l)=\frac{1}{|L|}\sum_{l\in L}\frac{r_a^l}{r_o^l}\times 100\% \tag{8}$$

4.3　实验结果分析

4.3.1　保留评价

影响匿名算法的因素包括隐私容忍度 P_{br}、攻击者数目|Adv|以及轨迹集合大小$|T|$. 攻击者数目|Adv|和隐私容忍度 P_{br}对 STR 的影响如表 8 和表 9 所示,从中可以看出 STR 随着 P_{br}增加而增加,随着攻击者数量的增加而减少.

表 8　隐私容忍度 P_{br}对 STR 的影响(%)

数据集	P_{br} / STR / θ	0.3	0.4	0.5	0.6	0.7
DS1 (|Adv|=4)	0.7	99.64	99.91	99.99	99.89	99.99
	0.75	99.19	99.83	99.97	99.89	99.99
	0.8	97.16	99.58	99.89	99.87	99.98
	0.85	95.55	99.52	99.83	99.86	99.95
DS2 (|Adv|=4)	0.7	99.04	99.72	99.75	99.93	99.97
	0.75	98.41	99.52	99.72	99.92	99.97
	0.8	96.84	98.86	99.53	99.82	99.91
	0.85	94.13	97.82	99.43	99.78	99.89

图 2(*a*)、2(*b*)展示了本文算法在 DS1 与 DS2 上的实验结果,其中设置参数 $\theta=0.85$,图 2(*a*)和图 2(*b*)分别显示了 STR 随不同隐私容忍度和不同攻击者数目的变化情况;图 2(*c*)、2(*d*)展示了本文算法与文献[14]算法在 DS1 和 DS2 上的对比实验结果,反映了 STR 随不同隐私容忍度的变化情况.

表 9　攻击者数目|Adv|对 STR 的影响(%)

数据集	|Adv| / STR / θ	2	3	4
DS1 ($P_{br}=0.5$)	0.7	99.99	99.99	99.99
	0.75	99.98	99.97	99.97
	0.8	99.95	99.90	99.89
	0.85	99.79	99.84	99.83
DS2 ($P_{br}=0.5$)	0.7	99.84	99.81	99.75
	0.75	99.83	99.79	99.72
	0.8	99.77	99.63	99.53
	0.85	99.72	99.55	99.43

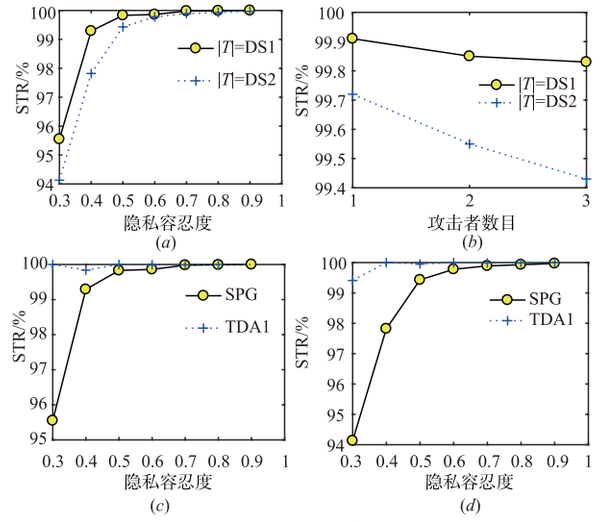


图2　STR变化曲线图(θ=0.85)

由图 2(*a*)可以发现,STR 随 P_{br}的增大而增大,说明用户隐私容忍度要求越严格,匿名数据处理后的损失程度越大. 而对于同等数值的 P_{br},DS1 曲线位于 DS2 曲线上方,说明在相同的隐私容忍度要求下,原数据集中轨迹数越多,通过匿名算法处理后的损失程度越小. 而当 $P_{br}>0.5$ 时,STR 值变化程度趋于平缓.

由图 2(*b*)可以发现在相同的隐私容忍度($P_{br}=0.5$)下,改变攻击者的数目,STR 值随攻击者数目增大而减小. 同样的,DS1 曲线位于 DS2 曲线上方,说明在其他变量相同情况下,原始轨迹集合越大,损失程度越小.

由图 2(*c*)和图 2(*d*)对比发现,文献[14]的算法 TDA1 曲线趋于平缓,接近 100%,STR 值没有明显随着隐私容忍度的改变而变化. 为使轨迹集合中的问题点对都

得到处理,该方案在绝大多数情况下选择了添加假轨迹法. 本文算法 SPG 在选择匿名处理方式时有效地综合了抑制法和假轨迹法,每次匿名处理时选择了当前收益最大的处理方式,提升了数据效用.

本文算法 SPG 与文献[10]提出的四个算法(全局抑制算法 GSUP、局部抑制算法 LSUP、轨迹分裂算法 SPLIT 和混合算法 MIX)的 AREL 得分结果对比情况如图 3 所示. 为了更好地说明实验结果,将$|Adv|$的值从 2 变化到 4,并将 P_{br}从 0.2 上升到 0.5. 从图 3 中可以看出,在所有情况下,SPG 算法的 AREL 值均低于文献[10]中四种算法在对应参数下的 AREL 值. 图 3(a)中,$P_{br}=0.5$,SPG 的 AREL 值接近于 0. 随着攻击者数量的增加,AREL 值逐渐增加. 实验结果验证了所提算法 SPG 的优越性,SPG 的 AREL 得分比 GSUP、LSUP、SPLIT、MIX 分别低了 66.28%、55.68%、35.95%和 32%. 图 3(b)中,$|Adv|=4$,随着 P_{br}的增加,AREL 值逐渐减小. SPG 将投影作为假轨迹添加到轨迹集合中,保留了频繁模式.

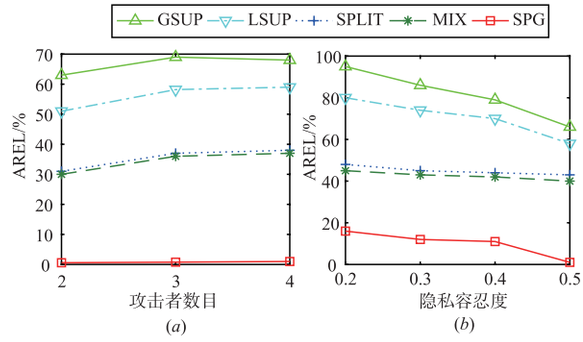


图3 AREL变化曲线(top-200) (DS2)

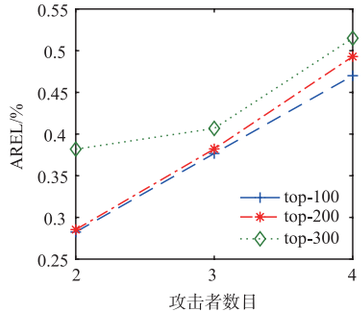


图4 AREL评估(DS1)

在 DS1 上取不同的 N 值对 SPG 进行 AREL 测试,实验结果如图 4 所示. 随着攻击者数量的增加,AREL 得分逐渐增加. 这三条曲线非常接近,并且在 $N=300$ 的情况下 AREL 结果高于其他两种情况下的结果,这是因为查询了更多的频繁模式.

4.3.2 数据损失率评价

文献[14]提出的第一个方案同样采用抑制和添加假数据相结合的匿名算法,采用式(7)对文献[14]算法 TDA1 与本文算法结果进行对比. 在 DS1 上进行实验,对比了不同隐私容忍度、不同攻击者数目下的 TL 变化情况. 实验结果如图 5(a)和(b)所示.

实验选取的隐私容忍度范围为 0.3 至 0.8. 从图 5(a)中可以看出,在隐私容忍度需求较大,即 P_{br}数值较小时,本文算法呈现的 TL 值占优势. TDA1 算法的 TL 值随着 P_{br}值的增大呈递减趋势,而本文算法的 TL 值与隐私容忍度大小不成线性关系,在隐私容忍度小于 0.5 时,选择本文算法能使匿名轨迹集合的数据损失率减少.

图 5(b)对比了隐私容忍度为 0.5 时,不同攻击者数量对于 TL 值的影响. 从图中可以看出,本文算法取得的 TL 值较文献[14]算法小,整体效果较好. 当攻击者数量变多的时候,TDA1 算法曲线呈线性上升,TL 值越来越大,即数据损失程度越来越大. 本文算法曲线变化平缓,当攻击者数量变多时可以有效减少数据损失.

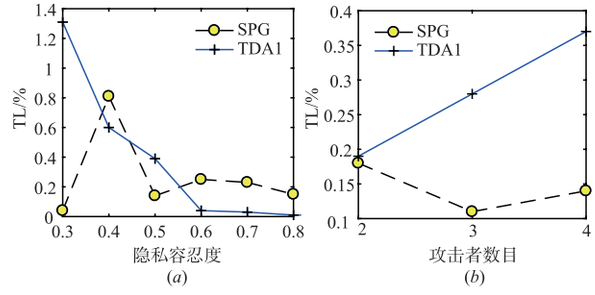


图5 TL变化曲线(DS1)

4.3.3 平均位置出现比率评价

本文统计了匿名轨迹集合中不同位置的出现次数,如图 6 所示,横坐标表示不同位置,纵坐标为相应出现次数. 设置$|Adv|=4$,$P_{br}=0.5$,轨迹集合位置保留结果如表 10 所示,可以看出匿名前后位置点没有减少,保留了 32 个位置点. 此外,DS1 和 DS2 中不同位置出现的平均保留率分别达到 99.74%和 99.46%.

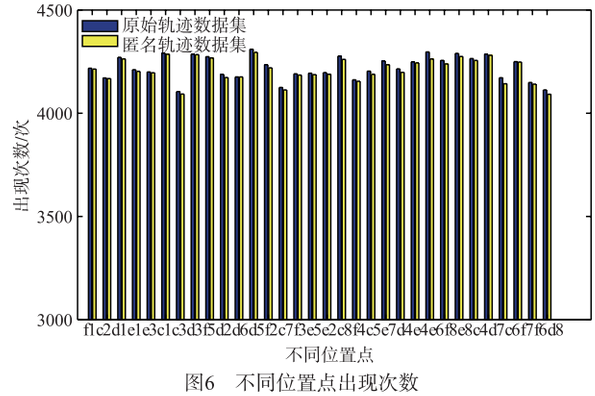


图6 不同位置点出现次数

将 GSUP、LSUP、SPLIT 和 MIX 分别与 SPG 算法的 ξ 值进行比较,实验结果如图 7 所示. 在图 7(a)中,$|Adv|$在 2 和 4 之间变化,$P_{br}=0.5$. 可以观察到,SPG 的曲线高

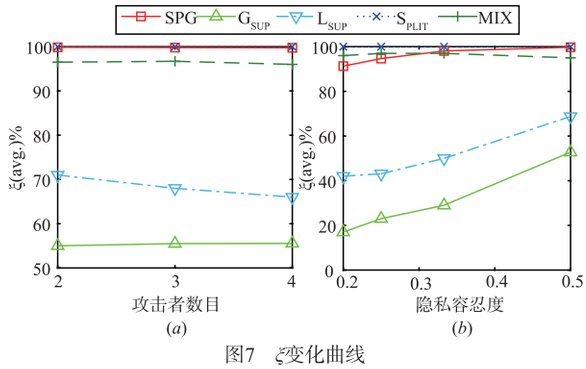


图7　ξ变化曲线

于 GSUP、LSUP 和 MIX 的曲线,略低于 SPLIT 的曲线. 然而,应该注意,SPLIT 算法不涉及位置点改变,仅涉及轨迹的分离. 当 SPLIT 算法执行时,原始轨迹被分成若干轨迹段,破坏了轨迹的原始运动模式. SPG 的位置损失率比 GSUP、LSUP 和 MIX 分别低了 44%、29% 和 4%.

在图 7(*b*)中,P_{br}在 0.2 和 0.5 之间变化,其中|Adv|=4. $P_{br}>0.35$ 时,SPG 的曲线高于 GSUP、LSUP 和 MIX 的曲线. 对于所有算法,ξ 随着 P_{br}的增加而增加,所有曲线都处于上升趋势,并且 SPG 接近于 1. P_{br}增大使得问题点对数量减少,数据损失率变小.

表 10　位置保留情况

数据集	DS1		DS2	
位置点保留结果	不同位置点数	出现次数	不同位置点数	出现次数
匿名前	32	135,081	32	67,832
匿名后	32	134,733	32	67,465
保留百分比	100%	99.74%	100%	99.46%

5　结论

本文提出了一种基于单点收益的轨迹隐私保护方法,用于保护用户的隐私不被具有部分背景知识的攻击者攻击. 分析了轨迹数据集大小、隐私容忍度大小以及攻击者数量对算法的影响,得出以下结论:

(1)即使用户隐私要求和攻击者数量增加,所提出的算法仍然保持较低的数据损失率;

(2)通过所提算法获得的匿名轨迹集合在用户隐私容忍度范围内不会泄露轨迹隐私信息.

本文所提算法在一定程度上提升了轨迹发布时的隐私保护能力,在隐私要求较高以及攻击者数量增加的情况下能有效提升数据效用. 在下一步工作中,将进一步研究各类轨迹隐私保护算法,综合考虑算法的优缺点,使算法在处理轨迹数据发布问题时更高效.

参考文献

[1] Zhang W,Wu L,et al. A trajectory privacy model for radio-frequency[J]. Wireless Personal Communications,2016,90(3):1121 – 1134.

[2] Memon I,Memon H,Chen G. Pseudonym changing strategy with multiple mix zones for trajectory privacy protection in road networks[J]. International Journal of Communication Systems,2018,31(1):1 – 44.

[3] Ye H,Cheng X,Yuan M,et al. A survey of security and privacy in big data[A]. Proceedings of the International Symposium on Communications and Information Technologies[C]. Piscataway:IEEE,2016. 268 – 272.

[4] 霍峥,孟小峰. 轨迹隐私保护技术研究[J]. 计算机学报,2011,34(10):1820 – 1830.
HUO Zheng,MENG Xiao-feng. A survey of trajectory privacy-preserving techniques[J]. Chinese Journal of Computers,2011,34(10):1820 – 1830. (in Chinese)

[5] Hu Z,Yang J,Zhang J. Trajectory privacy protection method based on the time interval divided[J]. Computers & Security,2018,77:488 – 499.

[6] Hwang R H,Hsueh Y L,Chung H W. A novel time-obfuscated algorithm for trajectory privacy protection[J]. IEEE Transactions on Services Computing,2013,7(2):126 – 139.

[7] Huo Z,Meng X,Hu H,et al. You can walk alone:Trajectory privacy-preserving through significant stays protection[A]. Proceedings of the International Conference on Database Systems for Advanced Applications[C]. Berlin:Springer,2012. 351 – 366.

[8] Li M,Zhu L,Zhang Z,et al. Differentially private publication scheme for trajectory data[A]. Proceedings of the 1st International Conference on Data Science in Cyberspace[C]. Piscataway:IEEE,2016. 596 – 601.

[9] 王丽娜,彭瑞卿,赵雨辰,等. 个人移动数据收集中的多维轨迹匿名方法[J]. 电子学报,2013,41(8):1653 – 1659.
WANG Li-na,PENG Rui-qing,ZHAO Yu-chen,et al. Multi-dimensional trajectory anonymity in collecting personal mobility data[J]. Acta Electronica Sinica,2013,41(8):1653 – 1659. (in Chinese)

[10] Terrovitis M,Poulis G,Mamoulis N,et al. Local suppression and splitting techniques for privacy preserving publication of trajectories[J]. IEEE Transactions on Knowledge and Data Engineering,2017,29(7):1466 – 1479.

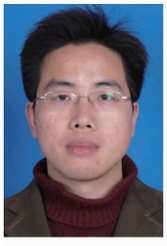
[11] Komishani E G,Abadi M,Deldar F. PPTD:Preserving personalized privacy in trajectory data publishing by sensitive attribute generalization and trajectory local suppression[J]. Knowledge-Based Systems,2016,94:43 – 59.

[12] Lei P R,Peng W C,Su I J,et al. Dummy-based schemes for protecting movement trajectories[J]. Journal of Information Science and Engineering,2012,28(2):335 – 350.

[13] Shen H,Bai G,Yang M,et al. Protecting trajectory privacy:A user-centric analysis[J]. Journal of Network and Com-

puter Applications,2017,82:128 – 139.
[14] 赵婧,张渊,李兴华,等. 基于轨迹频率抑制的轨迹隐私保护方法[J]. 计算机学报,2014,37(10):2096 – 2106.
ZHAO Jing,ZHANG Yuan,LI Xing-hua,et al. A trajectory privacy protection approach via trajectory frequency suppression[J]. Chinese Journal of Computers,2014,37(10):2096 – 2106. (in Chinese)
[15] Chen R,Fung B C M,Mohammed N,et al. Privacy-preserving trajectory data publishing by local suppression[J]. Information Sciences,2013,231:83 – 97.

作者简介



陈传明 男,1981 年 4 月出生于安徽六安. 现为安徽师范大学计算机与信息学院副教授、硕士生导师. 主要研究方向为数据挖掘、隐私保护.
E-mail:ccm_0@163.com

林文诗 女,1994 年 12 月出生于浙江台州. 现为安徽师范大学计算机与信息学院硕士研究生. 主要研究方向为网络与信息安全、隐私保护.
E-mail:lws_lin@163.com

俞庆英 女,1980 年 10 月出生于安徽黄山. 现为安徽师范大学计算机与信息学院副教授、硕士生导师. 主要研究方向为空间数据处理、信息安全.
E-mail:ahnuyuq@ahnu.edu.cn

罗永龙(通讯作者) 男,1972 年 4 月出生于安徽安庆. 现为安徽师范大学计算机与信息学院教授、博士生导师. 主要研究方向为空间数据处理、信息安全.
E-mail:ylluo@ustc.edu.cn