*Y*ou *C*an *W*alk *A*lone: Trajectory Privacy-Preserving through Significant Stays Protection

Zheng Huo[1], Xiaofeng Meng[1], Haibo Hu[2], and Yi Huang[1]

[1] School of Information, Renmin University of China, Beijing, China
{huozheng,xfmeng,westyi}@ruc.edu.cn
[2] Department of Computer Science, Hong Kong Baptist University, Hong Kong
haibo@comp.hkbu.edu.hk

Abstract. Publication of moving objects' everyday life trajectories may cause serious personal privacy leakage. Existing trajectory privacy-preserving methods try to anonymize k whole trajectories together, which may result in complicated algorithms and extra information loss. We observe that, background information are more relevant to where the moving objects really visit rather than where they just pass by. In this paper, we propose an approach called *Y*ou *C*an *W*alk *A*lone (*YCWA*) to protect trajectory privacy through generalization of stay points on trajectories. By protecting stay points, sensitive information is protected, while the probability of whole trajectories' exposure is reduced. Moreover, the information loss caused by the privacy-preserving process is reduced. To the best of our knowledge, this is the first research that protects trajectory privacy through protecting significant stays or similar concepts. At last, we conduct a set of comparative experimental study on real-world dataset, the results show advantages of our approach.

Keywords: Privacy-preserving, Trajectory data publication, Stay points extraction.

1 Introduction

Recent years, positioning techniques and location-aware devices have made numerous locations and traces of moving objects (MOBs) collected and published. Mining and analyzing trajectories is beneficial to multiple novel applications. For example, analyzing trajectories of passengers in an area can help people to make commercial decisions, such as where to build a restaurant; while, analyzing trajectories of vehicles in a city may help government to optimize traffic management systems. Although publishing trajectories is beneficial to mobility-related decision making processes, it still causes serious threats to personal privacy: spatio-temporal information contained in trajectories may reveal individuals personal information, such as, living habits, health conditions, social customs, work and home addresses, etc. When we say *trajectory privacy-preserving*, we mean to protect both whole trajectories not to be re-identified and the frequent/sensitive location samples not to be exposed. To address these problems, trajectory k-anonymity is proposed to anonymize k trajectories together in a broader similar time span [1, 2, 3]. But we argue that, it is not necessary to involve all location samples into privacy strategy.

S.-g. Lee et al. (Eds.): DASFAA 2012, Part I, LNCS 7238, pp. 351–366, 2012.

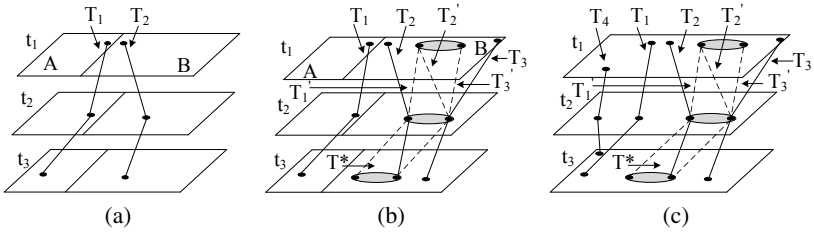


Fig. 1. An example of trajectory k-anonymity

Most of the trajectory k-anonymity methods try to anonymize k whole trajectories together, which may lead to serious information loss. Examples of trajectory k-anonymity are shown in Fig.1, where k=3. Without loss of generality, number of trajectories n_t in each sub figure is set to 2, 3, 4. They stand for n_t is less than k, exactly equals to k and larger than k respectively. We take (k,δ)-anonymity [2] as an example. When n_t=2 in Fig.1(a), trajectory 3-anonymity cannot be achieved, both trajectories should be deleted for privacy-preserving purpose. Note that, the deletion happens for all $n_t < k$. In Fig.1(b), $n_t = 3$. For a given radius δ, trajectory 3-anonymity can be achieved by trajectory clustering and space translation. Thus, original trajectories T_1, T_2 and T_3 (represented by the solid lines) are translated to T_1', T_2' and T_3' (represented by the dotted lines) respectively, then each location sample is generalized in T^* (represented as the cylinder in gray). Besides generalization, space translation also causes information loss. For example, given a query *Find me trajectories in area A*, it returns nothing if executes on T^*. While in fact, T_1 is in A, thus the query result is totally lost. If trajectory (k, δ)-anonymity tries to avoid space translation, anonymity region should be expanded, which may also cause information loss. In Fig.1(c), T_4 is added. However, the radius of the 4 trajectories is too large to be anonymized together, T_4 should be deleted. Then T_1, T_2 and T_3 are anonymized in T^*, the same as in Fig.1(b). Compared with original trajectories, deletion, space translation and generalization are adopted to achieve trajectory k-anonymity, while each of them may lead to information loss. Thus, the total information loss of trajectory k-anonymity is high.

Instead of treating location samples equally in trajectory k-anonymity, our key observation is that real trajectories are not randomly sampled spatio-temporal points, they have semantics, such as *stay points*. We therefore propose to protect trajectory privacy through protecting stay points, which may avoid serious information loss as well as providing high privacy guarantee. This idea is feasible for two main reasons: firstly, most background information is relevant to stay points (*e.g., check-ins at semantic places or credit card transactions at shopping malls*). Thus, protecting stay points may reduce the probability of whole trajectory exposure; secondly, protection of stay points can prevent leakage of sensitive information on trajectories, since stay points contain more sensitive information than ordinary location samples (*e.g., if a MOB visits or stays at a hospital, adversaries may infer the person has a health problem; while the adversary may infer nothing if the MOB just passes through in front of a hospital*). Moreover, trajectory k-anonymity highly depends on distributions of MOBs. If the distribution is too sparse to satisfy k-anonymity, trajectories may be deleted for privacy-preserving

purpose (as shown in Fig.1); while if the distribution is too dense, locations or trajectories may be exposed. This is because people always gather in semantic places, if we anonymize k MOBs at a semantic place, their locations are exposed. Our proposal can avoid this by generalizing stay points into *zones*, each zone contains at least l semantic places. Thus, adversaries cannot distinguish MOBs$'$ exact locations.

In this paper, we study the problem of protecting trajectory privacy in a data publication perspective. The key challenges of our proposal are how to extract stay points efficiently on people's trajectories and how to generate zones with minimized size and diversified contents. The contributions of this paper are as follows:

- We propose to depersonalize significant stay points on trajectories instead of current whole trajectory anonymization.
- We then implement this notion in our proposed method *Y*ou *C*an *W*alk *A*lone (*YCWA*) through splitting trajectories into {move, stay} sequences, and generalizing each stay point into a territory based on a generated split map.
- Two approaches are proposed to generate the split map. One of which is grid-based approach; the other one is clustering-based approach. The latter takes both spatial distance and semantic similarities into consideration.
- We experimentally evaluate the proposed approach on a real-world dataset. Experiment results show that information loss caused by *YCWA* exhibits lower than 20%, which obviously dominates trajectory k-anonymity method.

The rest of the paper is organized as follows. Section 2 summarizes related work. Section 3 formally defines concepts that we study in this paper. In section 4, we present our proposed approach. Section 5 analyzes the privacy guarantee and data utility. Experiment results are shown in section 6. Finally, section 7 concludes the paper.

2 Related Work

Trajectory privacy-preserving is a new research area that has received lots of concerns recent years. Several approaches have been proposed to tackle the problem in a data publication perspective in an off-line manner, while some have been proposed in the context of location-based services in an online manner.

We first introduce privacy-preserving techniques in trajectory data publication. In [2], Abul et al. propose a concept called (k, δ)-anonymity due to the imprecision of GPS devices, where δ represents the possible location imprecision. Then an approach called Never Walk Alone (*NWA*) is proposed to achieve (k, δ)-anonymity through trajectory clustering and space translation. In [3], Yarovoy et al. observe the fact that there does not exist a fixed set of QID attributes for all the moving objects, and the anonymity groups may not be disjoint. A notion of attack graph-based k-anonymity is proposed. Yarovoy et al. propose two algorithms called Extreme Union and Symmetric Anonymization to generate anonymity groups which satisfy the novel k-anonymity. In [1], Nergiz et al. argue that since the trajectories are published for research purpose, it is useful to publish atomic trajectories rather than anonymized regions. So the authors design a method to publish atomic trajectories in an anonymization-reconstruction manner: first, enforce

k-anonymity by clustering trajectories together based on *log cost distance*, then reconstruct trajectories by randomly selecting location samples from anonymized regions. Privacy strategies in [4] is based on the assumption that different adversaries may have different parts of MOBs' trajectories, while the data publisher knows what the attackers own. Then a suppression-based method is proposed to suppress trajectory segments which may reduce the probability of disclosing whole trajectories. In [6], the authors propose a new trajectory privacy-preserving method which is implemented through spatial generalization and *k*-anonymity.

Recently, several approaches have been proposed to protect MOBs' trajectory privacy in an online manner. In [10], a suppression-based method is proposed to protect users' online trajectory privacy. In this paper, areas are classified as either *sensitive* or *insensitive* based on the proportion of visitors and the whole population of that area. Location updates are suppressed when users enter a sensitive area. In [13], Toby et al. propose to anonymize historical trajectories with users' current trajectories. This proposal helps to reduce the area size of the cloaking region. In [5] Gyozo et al. propose a notion of trajectory privacy-preserving data collection, then implement it based on a server-client architecture. Each client's trajectory is split, exchanged and anonymized by the server before collected by the service provider.

The main difference between these works and our proposal is that they do not account for any difference of location samples, while in fact, stay points are more important and more sensitive than ordinary location samples. As far as we have investigated, we are the first to propose protecting trajectory privacy through protecting significant stay points. Another important concern in this paper is the diversity of sensitive attributes. If the places contained in a zone are almost the same type, the visitors' personal privacy may be exposed. *e.g., if someone visits a zone that only contains a certain type of sensitive locations, such as clubs, even the stay point is generalized to an area, adversaries may still discover he has visited a club no matter which one it is*. We solve this problem by using a mixed distance in the clustering algorithm to enforce diversified places into a zone.

3 Problem Statements

Trajectories of moving objects are collected and stored in moving object databases (MOD). For a moving object O_i, its trajectory $\mathbb{T}$ is a set of discrete locations at sampling time, represented as: $\mathbb{T} = \{q_i, (x_1, y_1, t_1), (x_2, y_2, t_2), \ldots (x_n, y_n, t_n)\}$, where q_i is the identifier of the trajectory; (x_i, y_i) represents MOB's position at sampling time t_i, (x_i, y_i, t_i) is a location sample on trajectories. Raw trajectories consist of GPS record, which is defined in [8]. In the next, we give some definitions in our study.

Definition 1 (Location). *A location* $\mathbb{L}$ *is a two-tuple* $<\boldsymbol{x}, \boldsymbol{y}>$, *which represents the latitude and longitude of the location.*

Each GPS record corresponds to a location at sampling time. Location samples are basically in two categories: pass-by points and stay points. Pass-by points are locations where MOBs just go through with a non-zero speed, while stay points stand for locations where the MOB stays over a certain time interval.

Definition 2 (Stay point). *A stay point $\mathbb{L}_{sp}$ is a four tuple <**sID**, **x**, **y**, **Δt**>, where sID is the identifier of the stay point; <x, y> is the coordinate of the stay point, Δt is the duration of the stay.*

Each time a user stays at somewhere over a time interval, we can obtain a corresponding stay point. Stay points of a real-world place may have different coordinates. Suppose a person visits a shopping mall from the front gate, while another person visits the same mall from the back door. Although they visit the same shopping mall, the stay points we extract are different. On the other hand, the imprecision of GPS devices may also result in different stay points for a real-world place. In order to obtain the real-world places where the MOBs visit, we define the notion of *place*.

Definition 3 (Place). *A place $\mathbb{P}$ is a set of stay points, it is represented as <**pID**, **loc**, **add**, **sem**>, where pID and loc represent the identifier and the centroid coordinate of $\mathbb{P}$ respectively, add represents the address of $\mathbb{P}$, while sem represents the semantic characteristics of $\mathbb{P}$, which consists of three parameters $<\overrightarrow{v}, \Delta t_{avg}, t_{enter}>$, they represent the visitors, average visit duration and average enter time respectively.*

Places we define here correspond to real-world places, such as shopping malls, clubs, restaurants, etc. Each place is available for all MOBs to visit or stay.

Definition 4 (Zone). *A zone $\mathbb{Z}$ is an area consists of at least l places, it is represented as <**zID**, **bl**, **ur**, **pn**>, where zID is the identifier of the zone, bl and ur represent the coordinate of the bottom-left corner and the upper-right corner respectively, pn represents the number of places included by the zone.*

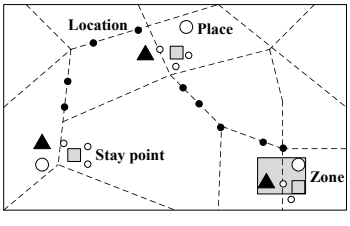


Fig. 2. Locations, stay points, places and zones

An example of these definitions can be seen in Fig.2, where solid back points are locations, hollow points represent stay points. Squares, circulares and triangles are real-world places, different shapes stands for different types of places. The shaded rectangle is a zone which contains three places. Zones are derived from places, places are derived from stay points. Thus, a zone is a generalized version of users' stay points.

4 Proposed Solutions

4.1 Solutions Overview

We assume adversaries have access to all published trajectories and the public background information. They know the distribution of real-world places on the map, but

they do not know the movement parameters of MOBs. Before our method, we assume the traces are already anonymized by replacing the true identifier with a random and unique pseudonym. Our goal in this paper is to anonymize original trajectory database D to a published version D^*, in which stay points cannot be exposed in a probability larger than $1/l$. The procedure of *YCWA* is as follows (also shown in Fig.3):

- Split map generation. First, we extract stay points from raw trajectories, then reconstruct semantic places using a *reverse geocoder* [9]. After that, we construct zones containing l places through a grid-based and a clustering-based method respectively.
- Trajectory anonymization. We divide trajectories into {move, stay} sequences, where stay points are replaced by corresponding zones. Pass-by points are either deleted or un-processed, depending on wether it locates inside a zone or not. At last, D is transformed to D^* in this step.
- Information loss measure. We measure information loss of D^* in this step. Since D^* is always published for analysis purpose, the utility of D^* should be kept high. Here we adopt an information loss measure in [3], which is represented as the reduction of the probability with which people can accurately determine the position of a MOB.

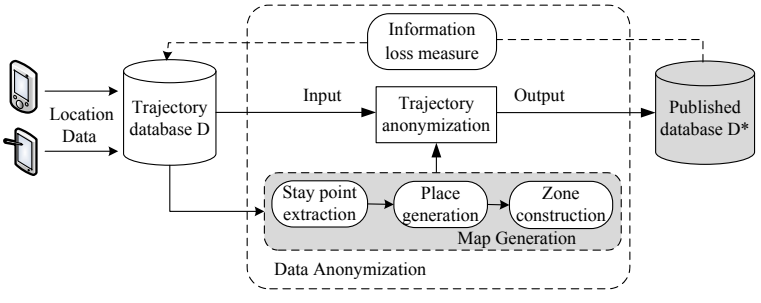


Fig. 3. Procedure of *YCWA*

4.2 Stay Points Extraction

We adopt stay points extraction strategies in [7] with improvements. Stay points are in two basic categories: stop and wondering. Accordingly, stay points occur in the following two situations. One of which is when a person equipped with a GPS logger gets into a building, the GPS logger loses signals and stops logging. Or, if a GPS-enabled car driver stops his car, the GPS device is turned off. The other one is when a GPS carrier is wondering around an outside sign, the GPS device is still logging and the velocity is not zero. In this case, the sign should also be regarded as a stay point [7].

For the first situation, we adopt a duration-based strategy. A parameter δt is introduced in order to avoid mistaking occasional stays for stay points, such as waiting for traffic lights. If a MOB stays at a location exceeding the time threshold δt, the location is regarded as a stay point. That is to say, given a trajectory $\mathbb{T}=\{(L_1, t_1), (L_2, t_2), \cdots,$

$(L_n, t_n)\}$, if $|t_{i+1}\text{-}t_i| > th_{time}$, L_i is regarded as a stay point, and the duration of this stay Δt is set to $|t_{i+1}\text{-}t_i|$. After that, all the stay points are put into D_{stays}. Thus, starts, ends and long stays are regarded as stay points.

For the second situation, we adopt a density-based strategy. Given a distance threshold th_{dist} and a time threshold th_{time}, if $distance(L_{i+1}, L_i) < th_{dist}$ and $|\ t_{i+1}\text{-}t_i\ | > th_{time}$, the MBR consists of L_i and L_{i+1} is called a dense area A_{dense}. Generally speaking, A_{dense} can be regarded as an outdoor stay point. However, a more complicated problem arises. When a GPS-enabled car meets traffic jams, the congestion area may be mistaken as an outdoor stay point. We solve this problem by recognizing whether the dense area A_{dense} is a road segment or not, since traffic jams always happen on road, while most outdoor signs are not. We put all the dense areas (represented by their geographic centers) as stay point candidates into D_{sc} and take D_{sc} into the next procedure.

4.3 Places Reconstruction

We recall Google Maps API to reverse-geocode coordinate of each stay point and stay point candidate. Places we reconstruct are put into D_{places}, which is initialized to empty. For each stay point L_i in D_{stays}, compare its reverse-geocoded address $L_i.add$ with each place′s address in D_{places}. If $L_i.add$ equals to $P_i.add$, merge L_i with P_i. Visitors $\overrightarrow{v}$, average visit duration Δt_{avg} and average enter time of the place t_{enter} of P_i are updated. While if $L_i.add$ does not equal to any addresses of existing places in D_{places}, set L_i as a new place in D_{places}. For each stay point candidate A_{dense} in D_{sc}, we reverse-geocode it at first. If the obtained address is a road segment, A_{dense} is probably caused by traffic jams, it should be deleted from D_{sc}. If the $A_{dense}.add$ is not a road segment, A_{dense} is regarded as an outdoor stay point, subsequent processing follows the same procedure as L_i. At last, we merge D_{sc} into D_{stay} and return D_{place}.

Real-world places are in different types, such as apartments, shopping malls, clubs, and office buildings,etc. In privacy-preserving literature, the most ideal situation is to enforce diversified places into a zone, but it is too hard to tag each place with a type. Therefore, we adopt a notion of similarity between places called place similarity [8] to solve this problem. We define similarity of places according to three parameters: *visitors*, *average visit duration* and the *average enter time*. Since places of different types usually exhibit different features. *e.g., office buildings may have an average enter time during 8.AM and 10.AM, average visit duration ranging from 7 to 9 hours, while a night club may be significantly different*. The three parameters can be used to capture these features and measure the similarities between places, as shown in equation (1).

Definition 5 (Place similarity). *Given two places* $< P_i$, *loc, add, sem*$>$ *and* $< P_j$, *loc, add, sem*$>$, *where sem is represented as* $< \overrightarrow{v}, \Delta t_{avg}, t_{enter} >$. *Place similarity can be computed by:*

$$sim(P_i, P_j) = \frac{\overrightarrow{v_i} \cdot \overrightarrow{v_j}}{|\ \overrightarrow{v_i}\ ||\ \overrightarrow{v_j}\ |} + \frac{min(\Delta t_{avgi}, \Delta t_{avgj})}{max(\Delta t_{avgi}, \Delta t_{avgj})} + \frac{min(t_{enteri}, t_{enterj})}{max(t_{enteri}, t_{enterj})} \quad (1)$$

The vector space model is used to compute the similarity between two visitor lists. The similarity of average visit duration is computed as the smaller one divided by the larger one, the same is done for average enter time. Sim(P_i,P_j) is computed by linear combination of the three scores. The higher the sim(P_i, P_j) value, more similar they are.

4.4 Zones Construction

In this section, we turn places into zones in order to generate a split map. Two different approaches are proposed, one is grid-based approach, called GridPartition; the other one is clustering-based approach called DiverseClus.

GridPartition. In GridPartition, the whole 2D Euclidean space is uniformly divided into square cells. Each cell is called a grid. Obviously, places are located in different grids, and the number of places in each grid is different. $G_i.num$ denotes number of places contained by G_i. For a user specified privacy level l, not every grid contains enough places to be a zone. We design an enlarging strategy to enforce at least l places into a zone, as shown in Algorithm 1. Each grid G_i is scanned in a spatial order. If $G_i.num > l$, G_i is tagged as a zone, and put G_i into D_{zones} (line 3-6). If $0 < G_i.num < l$, we try to merge it with its *neighbors*. For any grid or zone G' near G_i, if $0 < G'.num < l$, it is regarded as G_i's grid neighbor, and then put it into NGB_g; while G_i's zone neighbors are put it into NGB_z (line 7-8). An example can be seen in Fig.4(a), where l is set to 5. When $G_i.num < l$, G_i enlarges itself by merging with its grid neighbors in NGB_g (line 10-12). If $NGB_g = \Phi$, G_i enlarges itself by merging with its zone neighbors in NGB_z (line 13-15). After merging, $G_i.num$, $G_i.ur$ and $G_i.bl$ should be updated, and G_i is put into D_{zones}. Fig.4(b) shows the resulted two zones using the enlarging strategy.

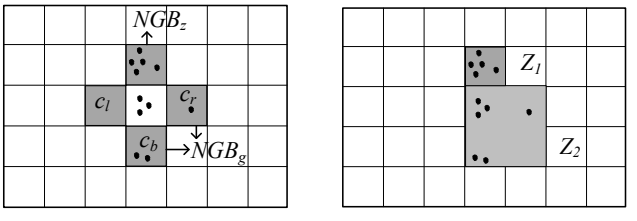


Fig. 4. GridPartition

GridPartition guarantees each zone contains more than l places, but it doesn't take places' semantic meanings into consideration. If a zone contains l places of the same type, it may also lead to privacy leakage, as we have previously mentioned. DiverseClus is proposed to address this problem.

DivserseClus. Given two places P_i and P_j, based on the spatial distance and place similarity, we introduce a mixed distance measure defined in equation (2).

$$Dist_{mix}(P_i, P_j) = \frac{Dist(P_i, P_j)}{sim(P_i, P_j) + \alpha} \tag{2}$$

Algorithm 1: GridPartition (D_{places}, l)

```
   Input  : D_places, minimum place numbers in each zone l
   Output: D_zones
 1 D_zones ← Φ;
 2 Divide the space into grids;
 3 for each grid G_i do
 4     if G_i.num > l then
 5         D_zones ← G_i;
 6         continue;
 7     NGB_g ← G_i's grid neighbors;
 8     NGB_z ← G_i's zone neighbors;
 9     while G_i.num < l do
10         if NGB_g ≠ Φ then
11             randomly select a grid g_i in NGB_g;
12             merge g_i into G_i;
13             else
14                 randomly select a zone z_i in NGB_z;
15                 merge z_i into G_i;
16         update G_i.num, G_i.ur and G_i.bl;
17     D_zones ← G_i;
18 return D_zones;
```

Here Dist(P_i, P_j) is a non-zero Euclidean distance, since P_i and P_j are represented by their geographic center, zero value never happens if they are two different places. The primary targets of DiverseClus is to cluster l places into zones with minimized area size, as well as diversified contents. We therefore measure the diversity of places by the dissimilarity of two places, represented as $\frac{1}{sim(P_1,P_2)+\alpha}$. The larger the value, more diverse they are, and they are more likely to be clustered together. α is used to avoid divide-by-zero error and smooth the penalty when the place similarity are very small. In our experiments, α is set as the standard deviation of place similarities, this strategy considers the majority differences of place similarities, and works well in our experiments. The details of the algorithm are represented in Algorithm 2.

At a very general level, the procedure of DiverseClus follows a similar structure of k-mediods [12]. The algorithm begins with a cluster center P_{cen}, which is the centroid place of D_{places}. Then, each cluster center is chosen as the farthest from the last one (except the first one, the *farthest* is measured by mixed distance, line 3-5). Places that near P_{cen} are clustered into *Clus*. $S_{P_{cen}}$ which represents clustering score of using P_{cen} as center is introduced to measure qualities of clusters. $S_{P_{cen}}$ is computed by the sum of distances of each place to P_{cen} in the cluster (line 6-8). In order to get an optimized result, the clusters should be adjusted by replacing P_{cen} with another place. Each place P_i in *Clus* is selected to replace P_{cen}, the clustering score S_{Pi} is computed using P_i as the clustering center (line 9-11). If $S_{P_{cen}} < S_{P_i}$, replace P_{cen} with P_i, until no replacement happens (line 12-14). The clusters are represented by their minimum bounding rectangles (MBRs). At last, *Clus.num*, *Clus.bl* and *Clus.ur* are updated and *Clus* is put into D_{zones}.

Algorithm 2: DiverseClus (D_{places}, l)

Input : D_{places}, minimum place numbers required l
Output: D_{zones}

```
1  P_cen ← the centroid place of D_places;
2  D_zones ← Φ;
3  while N_cen ≤ ⌊ |D_place|/l ⌋ do
4  P_cen ← the farthest of the last one;
5  N_cen++;
6  for each P_cen do
7      Clus ← P_cen ∪ l-1 nearest neighbors of P_cen;
8      S_Pcen ← Σ_j Dist(P_cen,P_j) / (sim(P_cen,P_j)+α);
9      for each P_i in Clus do
10         select P_i to replace P_cen;
11         S_Pi ← Σ_j Dist(P_i,P_j) / (sim(P_i,P_j)+α);
12         if S_Pi < S_Pcen then
13             replace P_i with P_cen;
14             until no changes;
15     update Clus.num, Clus.ur, Clus.bl;
16     D_zones ← Clus;
17 return D_zones;
```

The clusters generated by DiverseClus should be post-processed, since some of the clusters may overlap spatially. This is because two places are dissimilar in semantic meanings while the spatial distance between them is far. We adjust the clusters using the following strategy. For each cluster *Clus* derived from DiverseClus, check if it spatially overlaps with other clusters. If so, merge the overlapped clusters until no overlap exists. After the merging, we need to check the place number *Clus.num* in each cluster. If $l < Clus.num < 2l$, *Clus* is a qualified cluster, and it should be put into D_{zones}. If $Clus.num > 2l$, split *Clus* into two non-overlapped clusters spatially, each cluster contains at least l places. After the adjustment, each cluster's MBR is regarded as a zone, a split map consists of zones is generated. It may be argued that, the adjustment may eliminate the effects of the place similarity, this is undesirable because most clusters are not overlapping or the overlapping region is relatively small, since the nearby places are very likely to be in different types, this can also be proved in our experiments.

4.5 Trajectory Anonymization

Trajectories are then split and anonymized based on the split map. The original trajectory database D is set as input, each location sample is scanned, stay points are replaced by the corresponding zones. For each pass-by point, the published version is kept as the original one, unless the pass-by point is *covered* by a zone. *Cover* is a spatial relationship between a zone and a pass-by point of the same trajectory, as defined in definition 6. If a pass-by point L_j is covered by a zone, L_j is suppressed for

privacy-preserving purpose, since publication of location samples approaching to a zone may cause exposure of a stay point.

Definition 6 (Cover). *Given two location samples (L_i, t_i) and (L_j, t_j) on $\mathbb{T}$, L_i is a stay point, its corresponding zone is Z_i, while L_j is a pass-by point. If L_j locates inside Z_i, then L_j is* ***covered*** *by Z_i.*

5 Privacy and Utility Analysis

In this section we discuss the privacy guarantees and data utilities. We formally show that by applying our methods, the published database D^* will not expose any user's stay points during their travels. Privacy guarantee is always measured by re-identification probability which means the probability of adversaries to identify a stay point or a trajectory from the published database D^*.

Theorem 1. *Given a trajectory database D=$\{\mathbb{T}_1, \mathbb{T}_2, \ldots \mathbb{T}_n\}$ and its published version D^*=$\{\mathbb{T}_1^*, \mathbb{T}_2^*, \ldots \mathbb{T}_n^*\}$ generated by* YCWA*, the average stay points re-identification probability is bounded by 1/l.*

Proof. In the attack model, we assume adversaries have access to all the published trajectories and public knowledge. Adversaries do know the distribution of the places on the map, but they do not know the movement parameters of MOBs. Given a published version D^*, each stay point in D^* is generalized to an area which contains at least l diverse stay-able places. The re-identification probability depends on the number of places in a zone, which is bounded by $1/l$. □

To capture the information loss, we adopt the reduction in the probability with which people can accurately determine the position of an object in [3]. Given a published database D^* of D, the average information loss is defined in equation (3).

$$IL_{avg} = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}(1 - 1/area(zone(O_i,t_j))) + \sum_{d=1}^{h} L_d}{n \times m} \tag{3}$$

IL_{avg} represents the average shrinks of the identify probability of a location in D^*. Where $area(zone(O_i,t_j))$ represents the area size of the corresponding zone of O_i at time t_j when O_i stays. The probability of adversaries can accurately determine the location where the MOB stays shrinks from 1 to 1/area(zone(O_i, t_j)). If a location L_d is deleted, it is totally indistinguishable, so the information loss turns to be 1. $n \times m$ represents the total location samples in D. Obviously, IL_{avg} ranges from 0 to 1.

6 Experiments

In this section we report the empirical evaluation we have conducted in order to assess the performance of our methods, in terms of data utility and the efficiency.

6.1 Experimental Setup

We run our experiments on a real-world dataset. Thanks to the *Geolife* project [11], we get the published real trajectories of volunteers. The dataset contains more than 8000 trajectories of 155 users ranging from May 2007 to May 2010 mainly in Beijing. More than 23 million GPS records are contained. The dataset is represented as BEIJING henceforth. The experiments are run on an Intel Core 2 Quad 2.66HZ, windows 7 machine equipped with 4GB main memory.

Since we use the same dataset as in [7] to extract stay points, we adopt the same parameter values. Specifically, the duration threshold δ_t and th_{time} are set to 20 minutes, while the distance threshold th_{dist} is set to 200m. This results in 75,593 stay points (shown in Fig.5(b)) extracted from the BEIJING databaset (shown in Fig.5(a)). We may avoid bothering the readers with such details, as I believe we can simply clean the dataset first by removing all non-Beijing location points. It can be seen that, location samples distribute all over the city of Beijing, more than 95% of them concentrate within the Fifth Ring Road.

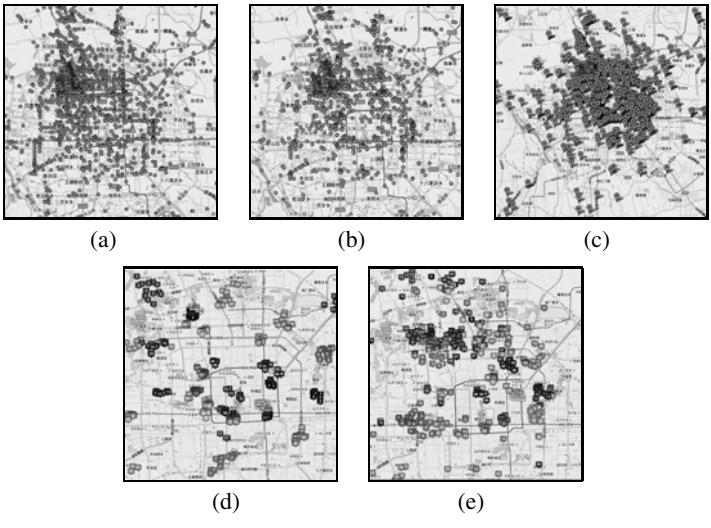


Fig. 5. Data distribution on the map. In (a) we report data distribution in BEIJING, in (b) stay points distribution, in (c) distribution of places, in (d) 50 clusters obtained purely on spatial distance, in (e) 50 clusters on mixed distance.

After extracting stay points in BEIJING, we recall Google Maps API to reverse each stay point to a real-world address. Thanks to Google Maps API, the returned results contain exact addresses and post codes, which make the place generation available and reasonable. After this procedure, 6902 semantic places are found. Fig.5(c) shows the distribution of semantic places in Beijing.

Both GridPartition and DiverseClus are used to generate the split map. In GridPartition, we divide the whole city of Beijing into grid cells of size $0.008^\circ \times 0.008^\circ$ each,

which results 62,408 grid cells. We then implement the enlarging strategies on these grids. In DiverseClus, the parameter α is set as the standard deviation of place similarities, as we have previously mentioned. The clustering results on spatial distance and mixed distance (without post-processing) are sampled in Fig.5(d) and 5(e), respectively. In both figures, 50 clusters are randomly selected to show the results. Places in the same cluster are painted in the same color. It can be seen that, clusters obtained based on mixed distance do overlap in Euclidean space representation before post-processing. Inclusion of place similarities do pose impacts on clustering results, post-processing is necessary in this situation. Based on the clustering results of places, the zones and the whole split map can be generated with various l values, i.e., the privacy levels. In the following experiments, we set l = 2, 4, 6 ,8, 10, and 12.

6.2 Measure of Data Utility

We then run a set of experiments on BEIJING to make a comparison on data quality between our approach and (k,δ)-anonymity [2]. (k,δ)-anonymity only works over trajectories with the same time span, a pre-processing step that partitions trajectories into equivalent classes is needed. Then a greedy clustering method is used to cluster trajectories together. At last, trajectories in each cluster are transformed into a (k,δ)-anonymity set, where δ is given as the radius of the anonymity set. The information loss we measure is computed by equation (3), where the area size is measured in square meters[1]. For (k, δ)-anonymity, the value of δ is set according to [2], ranges from 1000 to 4000, step by 1000. The evaluations shown in our figures are average values on δ. In information loss evaluation of (k, δ)-anonymity, we only account for the generalization and the deletion part, while the information loss caused by space translation can be seen in range query distortion measure.

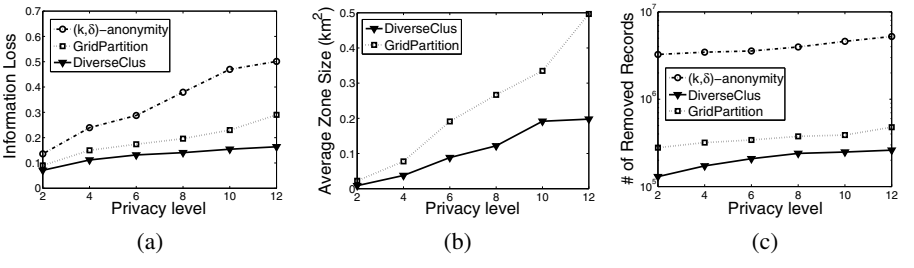
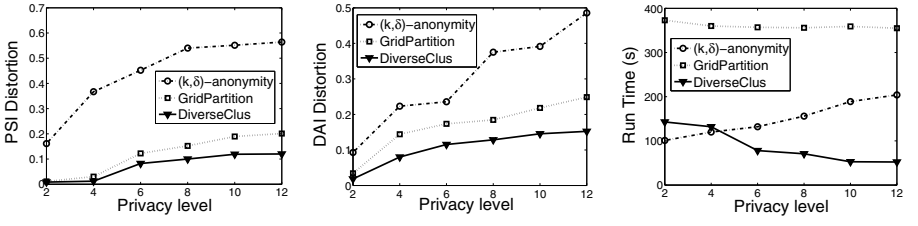


Fig. 6. Data utility measure of DiverseClus. In(a), we report information loss of the three algorithms, in (b) the average zone size, in (c) the number of removed location samples.

Comparison of all the three algorithms are shown in Fig.6(a). DiverseClus leads to less information loss than GridPartition since it adopts a clustering strategy, which

[1] The area size should be normalized by dividing 100, this is because the imprecision of GPS devices ranges from 5 to 15 meters, that is to say if a MOB locates in an approximately $100m^2$ area, the location of the MOB can be identified.

makes the zone size smaller than GridPartition. Information loss of both DiverseClus and GridPartition are less than 20%, which obviously dominate (k,δ)-anonymity. The information loss caused by our proposal is mainly caused by generalization of stay points and deletion of *covered* pass-by points. We therefore measure the average zone size and number of deleted location samples in Fig.6(b) and Fig.6(c). Obviously, DiverseClus performs better than GridPartition on both metrics. Since smaller zone size may cover fewer pass-by points, thus, making the removed location samples reduced. In all three figures, performance decreases as *privacy level* grows. We do not measure anonymized region of (k, δ)-anonymity, since for a given δ, the anonymized region is fixed to $\pi(\frac{\delta}{2})^2$.



(a) (b) (c)

Fig. 7. Performance evaluation of 3 algorithms, in (a) we report *PSI* query distortion comparison, in (b) the comparison of *DAI* distortion, in (c) run time comparison of 3 algorithms

We then measure the actual distortion of range query results on the published dataset D^* from the original dataset D. In particular, given a spatial region R and a time duration $[t_s, t_e]$, we consider two range queries the same as [2]: *Possibly_Sometimes_Inside* and *Definitely_Always_Inside*, represented as *PSI* and *DAI* for short respectively. Range query distortion is measured by $Distor_{rq} = \frac{min(Q(D),Q(D_*))}{max(Q(D),Q(D_*))}$. Where $Q(D)$ represents the number of query results on D, $Q(D^*)$ stands for the number of query results on D^*. We measure query distortion for various l values. We randomly choose circular region R having radius between 500 and 5000, and randomly choose time interval ranging from 2 hours to 8 hours. The parameter settings we use are according to [2]. At last, 1000 queries are generated, each of them have 1000 runs. We run these queries on trajectory database anonymized by our approaches and (k, δ)-anonymity. The average query distortion is shown in Fig.7(a) and Fig.7(b). Both GridPartition and DiverseClus have a range query distortion under 20%, while for (k,δ)-anonymity, the query distortion is larger, almost up to 60%, since it adopts space translation, some location samples are translated to totally different ones, making the query results lost.

6.3 Measure of Efficiency

The running times evaluation of the three algorithms are shown in Fig.7(c). Running time of GridPartition is the longest of the three, since the partition into grids procedure is costly. In both GridPartition and DiverseClus, we exclude the time consumption of stay points extraction and place reconstruction, because Google Maps API contains

restrictions, it is only allowed to reverse one coordinate every 2 seconds. The procedure of stay points extraction and places generation cost about 2951 minutes, but the data set of places can be used on various l values for both methods.

7 Conclusions

Collection and publication of people's everyday trajectories pose serious threats on people's personal privacy. In this paper, we propose to protect trajectory privacy through protecting significant stays on their trajectories, which can avoid unnecessary anonymization of pass-by location samples. Although sometimes the privacy guarantee of *YCWA* is not better than trajectory k-anonymity, in some applications, *YCWA* works well and the information loss is much lower.

In the future, we plan to reinforce our approach for multiple attack models as well as improve the space similarity measure. In *YCWA*, we assume adversaries do not know the moving speed of a MOB. But if they know the moving speed, such as, by knowing the maximal moving speed of a MOB, adversaries may infer the reachability to a zone, thus, the re-identification probability may increase. In another aspect, we plan to extend our approach to an online scenario, which means to dynamically maintain the split map when a new comer enters an area, thus, it can protect people's real-time trajectory privacy efficiently.

Acknowledgments. This research was partially supported by the grants from the Natural Science Foundation of China (No. 61070055, 60833005, 91024032, 91124001), the Fundamental Research Funds for the Central Universities, and the Research Funds of Renmin University of China (No. 11XNL010, 10XNI018), National Science and Technology Major Project (No. 2010ZX01042-002-003).

References

1. Nergiz, M.E., Atzori, M., Saygin, Y., Baris, G.: Towards Trajectory Anonymization: A Generalization-based Approach. IEEE Transactions on Data Privacy 2, 47–75 (2009)
2. Abul, O., Bonchi, F., Nanni, M.: Never Walk Alone: Uncertainty for Anonymity in Moving Objects Databases. In: 24th IEEE International Conference on Data Engineering, pp. 215–226. IEEE Press, Washington (2008)
3. Yarovoy, R., Bonchi, F., Lakshmanan, S., Wang, W.H.: Anonymizing Moving Objects: How to Hide a MOB in a Crowd? In: 12th International Conference on Extending Database Technology, pp. 72–83. ACM Press, New York (2009)
4. Terrovitis, M., Mamoulis, N.: Privacy Preservation in the Publication of Trajectories. In: 9th International Conference on Mobile Data Management, pp. 65–72. IEEE Press, Washington (2008)
5. Gidofalvi, G., Huang, X., Bach, P.T.: Privacy-preserving Trajectory Collection. In: 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, p. 46. ACM Press, New York (2008)
6. Moreale, A., Andrienko, G.L., Andrienko, N.V., Giannotti, F., Pedreschi, D., Rinzivillo, S., Wrobel, S.: Movement Data Anonymity through Generalization. IEEE Transactions on Data Privacy 3, 91–121 (2010)

7. Zheng, Y., Zhang, L., Xie, X., Ma, W.: Mining Interesting Locations and Travel Sequences from GPS Trajectories. In: 18th International Conference on World Wide Web, pp. 791–800. ACM Press, New York (2009)
8. Cao, X., Cong, G., Jensen, C.S.: Mining Significant Semantic Locations From GPS Data. Proceedings of the VLDB Endowment 3(1), 1009–1020 (2010)
9. Google Reverse Geo-coder, http://www.findlatitudeandlongitude.com/find-address-from-latitude-and-longitude.php
10. Gruteser, M., Liu, X.: Protecting Privacy in Continuous Location-tracking Applications. IEEE Security and Privacy 2(2), 28–34 (2004)
11. Microsoft Research Geolife, http://research.microsoft.com/en-us/projects/geolife/
12. Kaufmann, L., Rousseeuw, P.: Clustering by means of medoids. Statistical Data Analysis Based on the L1-Norm and Related Methods, 405–416 (1987)
13. Xu, T., Cai, Y.: Exploring Historical Location Data for Anonymity Preservation in Location-based Services. In: 27th Conference on Computer Communications, pp. 547–555. IEEE Press, Washington (2008)