Privacy-Preserving Location Publishing under Road-Network Constraints

Dan Lin, Sashi Gurung, Wei Jiang, and Ali Hurson

Missouri University of Science and Technology
{lindan,sgy99,wjiang,hurson}@mst.edu

Abstract. We are experiencing the expanding use of location-based services such as AT&T TeleNav GPS Navigator and Intel's Thing Finder. Existing location-based services have collected a large amount of location data, which have great potential for statistical usage in applications like traffic flow analysis, infrastructure planning and advertisement dissemination. The key challenge is how to wisely use the data without violating each user's location privacy concerns. In this paper, we first identify a new privacy problem, namely *inference route* problem, and then present our anonymization algorithms for privacy-preserving trajectory publishing. The experimental results have shown that our approach outperforms the latest related work in terms of both efficiency and effectiveness.

1 Introduction

The extensive use of location-based services, such as AT&T TeleNav GPS Navigator, Sprint's Family Locator, and Intel's Thing Finder, have collected a large amount of location data. If information like vehicle IDs and moving directions on roads can be published, people in many fields will benefit from it. With respect to the public sector, traffic flow information can be extracted from published IDs and moving directions. Such information will play an important role in infrastructure construction and traffic light control. With respect to the business domain, traffic information can help decide the location of company branches, and also advertisements can be customized and disseminated at the most advantageous locations. With respect to our daily lives, traffic information is certainly useful for detecting and predicting traffic jam, and calculating better routes in an emergency (e.g., for ambulances). However, in the meantime, location privacy concerns [11, 16] may hinder the development of such attractive usage of traffic information. It is well known that using a pseudonym is not sufficient to prevent the linkage of a published location to a real ID [5]. The key challenge is how to wisely use the location data without violating each user's privacy concerns. This problem is termed as *privacy preserving historical location data publishing*.

Historical location data forms a sequence of locations in chronological order, termed as *trajectory*. In general, one's trajectory consists of roads he has visited. For instance, in Figure 1, user u_1's trajectory can be represented as $IABC$ and user u_4's trajectory is ABD. Such road-network based trajectories are valuable in aforementioned applications. In privacy-preserving location publishing, the goal is to prevent adversaries from mapping published locations to a specific individual.

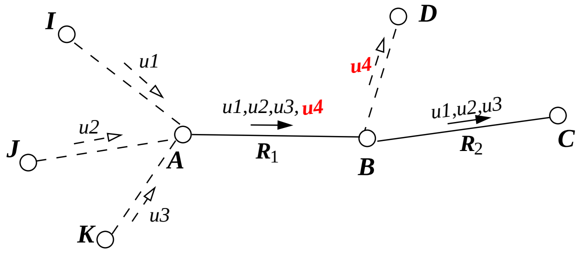


Fig. 1. An Example of Inference Route

One may think that a trajectory resembles a conventional sequential pattern. Hence, a naturally raised question is that if we can directly employ privacy preserving data publishing approaches [3, 4, 13, 20] developed in non-spatial-temporal databases? The answer is negative, and the main reason is that a trajectory distinguishes itself from the conventional sequential patterns due to additional constraints (e.g., road-network information) which do not exist in the traditional sequences. More specifically, elements in traditional sequences are usually independent of one another, while the relationship of elements in the trajectory sequence is fixed under a particular road-network information. Therefore, we cannot use traditional algorithms to arbitrarily remove or replace elements in the sequences because such operations will create unrealistic trajectories consisting of non-connected road segments.

There have been several recent efforts [2, 7, 12, 17] on anonymizing trajectories. Some work [17] considers trajectories as a sequence of landmarks, e.g., stores and museums, which ignore the paths connecting these places. Others [2, 7, 12] consider trajectories as a sequence of coordinates in Euclidean space but ignore the road-network constraints. Very few works considered the road-network constraint. The most recent one is by Pensa et al. [14], who anonymize road-network-based trajectories based on k-anonymity [15]. However, their approach may not preserve trajectory information as much as possible. This can be demonstrated by the example given below.

In [14], trajectories are stored and anonymized by using a prefix tree which may not be an appropriate structure to model the road-network. For instance, consider four users who leave their homes (I, J, K, D) and head for work. When k is 3 and the input to their algorithm is the following four trajectories: $u_1(IABC)$, $u_2(JABC)$, $u_3(KABC)$ and $u_4(ABD)$[1], their anonymization result will be an empty set since the prefix tree treats trajectories with different starting points independently. Such result obviously lost too much useful information. To achieve better information utility, an alternative way is to directly take partial trajectories as input, i.e., consider only busy roads with more than k users. In this case, the input becomes $u_1(ABC)$, $u_2(ABC)$, $u_3(ABC)$ and $u_4(AB)$, and the new anonymization result is : $u_1'(ABC)$, $u_2'(ABC)$, $u_3'(ABC)$ and $u_4'(AB)$, which is more meaningful than the previous empty set.

In addition, since road maps can be found everywhere, in the domain of privacy-preserving location publishing, it is reasonable to assume road-network information

[1] u_1, u_2, u_3 and u_4 can be thought as either a trajectory ID or a person's symbolic ID.

is available to any adversary. Thus, cautions are very much needed when publishing anonymized trajectories. For instance, let us continue from the previous example and assume that the road-network in Figure 1 is accessible to an adversary Bob. When u'_1, u'_2,u'_3 and u'_4 are published, using the road-network, Bob can infer that u'_4 was also travelled on the road segment $\overline{BD}$. Also, if Bob knows that Alice usually travels on $\overline{BD}$, then he can link u'_4 to Alice and consequently track Alice remaining trajectories in the published dataset. This *inference route problem* is caused by the fact that an adversary can infer someone's unpublished infrequent trajectories from the published location dataset. Because the inferred trajectories are infrequent, with high probability, these trajectories, combined with certain external knowledge, can be used to identify a particular individual's trajectory information in the published dataset. In general, given a threshold k, if the attacker can link any anonymous ID to Alice with probability greater than $\frac{1}{k}$ by using the above method, then we say there is an inference route problem.

In this paper, we address the problem of privacy-preserving location data publishing under the assumption that road-network data are public information. Our approach has three main properties: (1) it guarantees k-anonymity of published data, (2) it avoids the inference route problem, and (3) the anonymization results follow the road-network constraints. The basic idea is to employ a clustering-based anonymization algorithm to group similar trajectories and minimize the data distortion caused by anonymization through a careful selection of representative trajectories. We propose a C-Tree (Cluster-Tree) to speed up the clustering process and develop methods to incrementally calculating error rates. The rest of the paper is organized as follows: Section 2 reviews related work, Section 3 presents our proposed approach, Section 4 reports experimental results, and Section 5 concludes the paper with lessons learned and future research directions.

2 Related Work

Privacy-preserving location publishing is a relatively young area in which little research has been carried out. In [7, 12], the spatial-temporal cloaking technique is applied to generate cloaking regions covering segments of trajectories. In [2], Abul et al. consider a trajectory as a cylindrical volume where the radius represents the location imprecision. Then they perturb and cluster trajectories with overlapping volumes to ensure that each released trajectory volume encloses at least $k - 1$ other trajectories. Unlike the previous work which is based on the similarity of trajectories, Yarovoy et al. [19] group trajectories based on so-called quasi-identifiers which is hard to be selected in practice. None of the approaches considers the impacts of road network constraints and hence, their anonymization results are vulnerable to attack when the malicious party knows the road map or holds some other background knowledge. E.g., if a cloaking region covers only one road, the corresponding trajectory can be easily mapped to the road.

In [17], Terrovitis and Mamoulis assume that the adversaries know partial trajectory information of some individuals. They use it as part of input to their anonymization algorithm. Such usage limits the generality and feasibility of their approach. In [1], Abul et al. used a coarsening strategy which removes one or more spatial points in a trajectory to achieve anonymization. An anonymized trajectory may contain disconnected paths. This is different from our approach which preserves continuous trajectories based on road-network information. Two other related works used time confusion and path confusion respectively. The time confusion approach [9] mixes location samples of different trajectories, and the path confusion approach [8] crosses paths in areas where at least two users meet. The main problem of the two approaches is that traffic flows are no longer preserved.

The most related work is by Pensa et al. [14]. They proposed a prefix-tree based anonymization algorithm which guarantees k-anonymity of the published trajectories in a way that no trajectories with support less than k will be published. They defined the support of a trajectory Trj as the number of trajectories containing Trj, which however causes the inference route problem. Here, we can see that how the concept of k-anonymity is applied will affect the quality of the anonymization result.

3 Problem Statement

In general, raw data collected by location-based applications contains user (object) information as a four-tuple $\langle ID, loc, vel, t\rangle$, where ID is the object ID, loc and vel are object location and velocity at timestamp t respectively. The anonymized dataset contains object information in the form of $\langle aid, rid, dir\rangle$, where aid is an anonymized object ID, rid is a road ID and dir is the object's moving direction. Here, for privacy concerns, we replace specific locations and velocities by road ID and moving direction. Such representation is sufficient to derive trajectories or traffic flow information.

The road network is modeled as a directed graph, where each edge corresponds to a road and each node represents an intersection. Specifically, an edge is represented as $\overline{n_i n_j}$, where n_i and n_j denote nodes. We then proceed to define the frequent road and inference route problem.

Definition 1 *Let W be a time interval, and let k be a threshold. We say a road is a frequent road if the number of moving objects moving along one direction on this road is no less than k within time W. We call the number of moving objects the frequency of the road.*

Definition 2 *Let Υ be an intersection of roads r_1, ..., r_m, and let U_i^+, U_i^- be the sets of objects moving toward and outward Υ on road r_i ($1 \leq i \leq m$) during W, respectively. If $\exists$ U_i^+, U_j^-, $|U_i^+| \geq k$, $|U_j^-| \geq k$, and (0< $|U_i^+ - U_j^-| < k$ or 0< $|U_j^- - U_i^+| < k$), then we say Υ has an **inference route** problem.*

To have a better understanding of the above definition, let us revisit the example in Figure 1. Node B is an intersection of three roads. On road $\overline{AB}$, U_{AB}^+ = $\{u_1$,u_2,u_3, $u_4\}$; on road $\overline{BC}$, U_{BC}^-=$\{u_1$,u_2,$u_3\}$. Since $U_{AB}^+ - U_{BC}^- = \{u_4\}$, $|U_{AB}^+ - U_{BC}^-|$ = 1< k, node B has an inference route problem.

Next, we present how to evaluate the quality of the anonymized dataset or trajectories. Intuitively, the less difference between the anonymized dataset and the original dataset, the better quality the anonymized dataset is. Therefore, we use two common metrics: average error rate and standard deviation. Suppose there are N roads (or edges in a road-network graph) and r_i represents road i. Let $original_{r_i}$ and $anonymized_{r_i}$ denote r_i's original frequency and frequency after the trajectories have been anonymized. Then in Equation 1, the error function E is defined as the average difference between $original_{r_i}$ and $anonymized_{r_i}$ (i.e., E_i), and σ is the standard deviation of the error rates. A low standard deviation indicates that the anonymization quality of each road is similar and close to the average error rate.

$$E = \frac{1}{N}\sum_{i=1}^{N} E_i = \frac{1}{N}\sum_{i=1}^{N}\frac{|original_{r_i} - anonymized_{r_i}|}{original_{r_i}} \quad (1)$$

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(E_i - E)^2} \quad (2)$$

4 Our Approach

In this section, we present our anonymization algorithm. It consists of two main steps. First, from the raw dataset D, we remove records associated with infrequent roads, i.e., roads with less than k objects within a given time interval. We denote the obtained dataset as D'. In D', we construct partial trajectories for the remaining objects based on moving directions. Note that one user may have several disconnected partial trajectories because he may visit some infrequent roads. Each partial trajectory will be assigned an anonymous ID. For the rest of the paper, the word "trajectory" and "partial trajectory" are interchangeable.

The second step is the core of the anonymization process. We propose a clustering-based anonymization algorithm which guarantees that by achieving strict k-anonymity (defined in Section 4.1) among partial trajectories, our anonymization result is free of the inference route problem. Compared to traditional k-anonymization approaches, our approach not only needs to minimize errors caused by anonymization but also needs to satisfy some unique requirements. Road-network constraints should be enforced during the entire anonymization process, especially when computing the representative trajectories. The first step is relatively straightforward. Therefore, the following discussion focuses on the anonymization step.

4.1 Clustering-based Anonymization

The essential idea of clustering-based anonymization algorithm is to find clusters of similar trajectories and anonymize them by using a representative trajectory. The details are the following.

First, we need to select a proper way to represent trajectories. Trajectories are initially represented as a sequence of timestamped locations. In our anonymized dataset, we do not disclose exact locations because detailed information increases attackers' chances to link published location to specific individuals. Instead, we report only information about which object passing by which road. There are two options: (i) representing a trajectory by road IDs; or (ii) representing a trajectory by node IDs. As illustrated in Figure 2, trajectories Trj_1, Trj_2 and Trj_3 can be represented as r_4r_2, r_1r_3, and r_1r_5 respectively following the first option. Using the second option, trajectories Trj_1, Trj_2 and Trj_3 can be represented as $n_5n_2n_3$, $n_1n_2n_4$, and $n_1n_2n_6$ respectively. Both types of representations well capture the similarity between trajectories Trj_2 and Trj_3 which share one common road. However, the first option treats Trj_1 and Trj_2 as two irrelevant trajectories even though they intersect. To better reflect relationships among

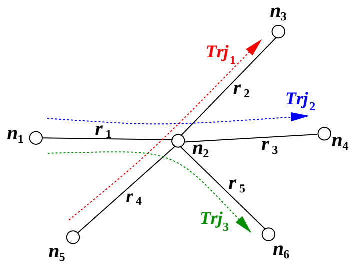


Fig. 2. Trajectory Representation

trajectories, we adopt the second option and represent a trajectory by a sequence of node IDs.

The second issue is to define the distance between trajectories. We employ the *edit distance* [18]. The edit distance between two trajectories is given by the minimum number of operations needed to transform one trajectory into the other, where an operation is an insertion, deletion, or substitution of a node. For example, the edit distance between $Trj_1(n_5n_2n_3)$ and $Trj_2(n_1n_2n_4)$ is 4, while the distance between Trj_2 and $Trj_3(n_1n_2n_6)$ is 2.

Now we are ready to present our clustering-based anonymization algorithm. An outline is given in Figure 3. First, we group same trajectories and count its *support*. Support is defined as the number of users who have the same trajectories (Definition 3).

Definition 3 *Let u be a user's anonymous ID and Trj_u denote his trajectory in D'. We have the support of trajectory Trj as follows: Support(Trj) = $|\{u|Trj_u = Trj$, for every u$\}|$.*

Distinct trajectories are arranged in a descending order of their supports. If a trajectory's support is more than the anonymization threshold k, the trajectory itself forms a cluster. For the remaining trajectories, say Trj, we compare it with existing clusters. If there exists a suitable cluster, we insert the new trajectory into that cluster and update the cluster's information. Otherwise, a new cluster will be created for Trj. After all trajectories have been checked, we translate representative trajectories together with their supports into output format, which contains object anonymious IDs, road IDs, and objects' moving directions. For example, we obtain the following intermediate result after anonymizing the trajectories shown in Figure 1: $u'_1(ABC)$, $u'_2(ABC)$, $u'_3(ABC)$ and $u'_4(ABC)$, where $k = 3$. The published dataset will look like this: $(u'_1, R_1, \overline{AB})$, $(u'_1, R_2, \overline{BC})$, $(u'_2, R_1, \overline{AB})$, $(u'_2, R_2, \overline{BC})$, ..., $(u'_4, R_2, \overline{BC})$. The detailed algorithms for finding candidate clusters, calculation of error rates and selection of representative trajectories will be elaborated in the rest of the section.

Our approach ensures strict k-anonymity (Definition 4) over all trajectories in dataset D'. It is called "strict" because the calculation of trajectory supports is based on an exact match of entire trajectories. In this way, we guarantee that the anonymization result will not contain any inference route. Our proof can be found in [10].

Definition 4 *(Strict k-anonymity over trajectories): Let Trj be a trajectory. We say Trj satisfies strict k-anonymity if Support(Trj) is no less than k.*

```
Clustering-based Anonymization ($TRJ$, $k$)
Input: $TRJ$ is a set of trajectories to be $k$-anonymized
1.  Group same trajectories and form $TRJ'$
2.  Sort trajectories in $TRJ'$ in a descending order of supports
3.  for each $Trj$ in $TRJ'$ do
4.      if $Trj.support \geq k$ then
5.          create a new cluster for $Trj$
6.      else
7.          check existing clusters
8.          if Find_Cluster($Trj$,$C$) then
9.              insert $Trj$ to cluster $C$
10.             Select_Representative_Trajectory($C$,$Trj_r$)
11.             update $C$'s error rate
12.             update $C - tree$
13.         else
14.             create a new cluster for $Trj$
15. for each cluster C in group of clusters
16.     if $C.Total\_TRJ \geq k/2$ then set $C.Total\_TRJ = k$
17.     else remove C from group of clusters
18. Translate representative trajectories into output format
```

Fig. 3. An Outline of Clustering-based Anonymization Algorithm

4.2 Finding Candidate Clusters

In this subsection, we present how to find a candidate cluster for a new trajectory during the clustering-based anonymization. The first step is to check whether a new trajectory can be absorbed by an existing cluster according to the distance metric. As the number of clusters increases, comparing Trj with all clusters becomes very costly. Therefore, we employ an in-memory index structure, C-tree (Cluster-tree), to prune unnecessary comparisons. In particular, each node in the C-tree contains multiple entries and each entry in a node has two fields: a pointer ptr and a set of road IDs (denoted as RID). In leaf nodes, each entry has a pointer to a cluster and the IDs of roads occurring in that cluster. In internal nodes, each entry has a pointer to a child node and the union of roads IDs in its child node. Figure 4 shows an example C-tree.

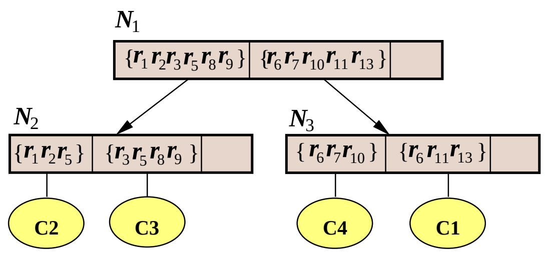


Fig. 4. An Example C-tree

Given a new trajectory Trj, starting from the root of the C-tree, we calculate the similarity between Trj and every entry's RID in the node by using the following similarity function.

$$Sim_c(Trj, RID) = \frac{|S(Trj) \cap RID|}{|S(Trj)|} \tag{3}$$

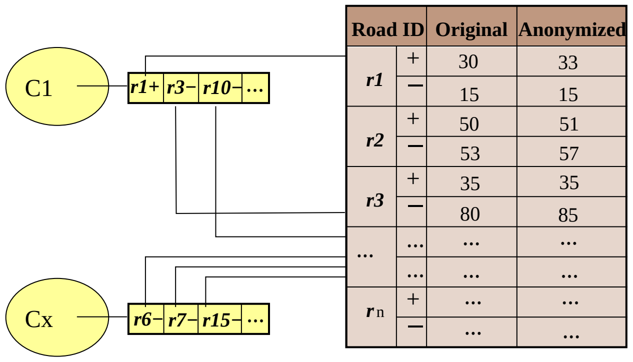
Sim_c computes the percentage of common roads included in Trj and RID, where $S(Trj)$ denotes the set of road IDs in trajectory Trj. If Sim_c is above a threshold ρ, we continue to visit the child node of this entry. This process is repeated until we find all entries in the leaf nodes with Sim_c above the threshold. All the clusters belonging to these entries will be considered as candidate clusters. For example, suppose that a new trajectory contains roads r_2, r_8 and r_9, and the threshold ρ is set to 60%. The similarity Sim_c between the new trajectory and the first and second entries in the root node N_1 are 100% and 0% respectively. The tree below the second entry is pruned and thus we do not need to visit node N_3. We continue to visit the child node N_2 pointed by the first entry. The Sim_c between the trajectory and the first and second entries in N_2 are 33% and 67% respectively. Since the second entry has the similarity score above the threshold, its corresponding cluster $C3$ becomes the candidate cluster for the further consideration.

Among candidate clusters, we further calculate the edit distance between the new trajectory and their representative trajectories. For all clusters which have the shortest edit distance with Trj, we examine the quality of anonymization result (i.e. error rate (E)) by assuming inserting Trj to a cluster. For a cluster C_i, its error rate E_{c_i} is computed based on the roads in this cluster. We select the cluster that satisfies two conditions: (i) it yields the smallest error rate after inserting Trj; (ii) its new error rate is

```
Find_Cluster ($Trj$,$C$)
Input: $Trj$ is a trajectory
Output: $C$ is a cluster

1.  $NODE \leftarrow$ {C-tree.root}
2.  while ($NODE$ is not empty) do
3.      for each node $N$ in $NODE$ do
4.          for each entry $en$ in $N$ do
5.              if $Sim_c(Trj, en.RID) > \rho$ then
6.                  if $N$ is not a leaf node then
7.                      add $en$'s child node to $NODE$
8.                  else add $en$'s cluster to candidate list $L_c$
9.  for all clusters in $L_c$ do
10.     find clusters with shortest edit distance with $Trj$
11.     if more than one clusters found then
12.         find the cluster with the smallest error rate
13.     if error rate after adding $Trj$ does not exceed threshold then
14.         return the cluster found
```

Fig. 5. Algorithm of Finding Clusters



Road ID		Original	Anonymized
r1	+	30	33
	−	15	15
r2	+	50	51
	−	53	57
r3	+	35	35
	−	80	85
...	...	...	...
	...	...	...
*r*n	+	...	...
	−	...	...

Fig. 6. Anonymization Table

below the global threshold Err. Figure 5 summarizes the procedure of finding candidate clusters.

To efficiently and incrementally calculate error rates during clustering, we employ a global data structure, i.e. *anonymization table*. Anonymization table has three fields: *roadID*, *original* and *anonymized*, where "original" is the number of objects before anonymization, and "anonymized" records the latest number of objects on road *roadID* during anonymization. Each cluster only needs to maintain a set of road IDs with pointers referring to the anonymization table. Figure 6 illustrates the data structure.

When actually inserting Trj to C_i, there are three steps: (i) update the representative trajectory; (ii) update the error rate in the anonymization table; and (iii) update the C-tree. The algorithm for selecting the representative trajectory is presented in Section 4.3. Once the representative trajectory is chosen, we recompute the error rate and modify the corresponding field in the anonymization table. Finally, we check whether the node in the C-tree with respect to current cluster needs to be updated. If current cluster contains road IDs which are not included in the road ID list of the corresponding C-tree entry, we will append the new road IDs to the road ID list. This change will be propagated to higher levels of the C-tree until an entry containing all road IDs in current cluster is reached. Consider the C-tree in Figure 4 and suppose that a new trajectory that consists of roads r_2, r_8 and r_9 will be inserted into cluster C_3. We check the road list of C_3's entry in the C-tree, which is $\{r_3r_5r_8r_9\}$ and does not contain r_2. We then add r_2 to the road list. Now the second entry in the C-tree becomes $\{r_2r_3r_5r_8r_9\}$. Next, we check its parent entry, the first entry in N_1. Since r_2 is included in the first entry in N_1, the tree update operation completes.

If no cluster is similar enough to Trj, we create a new cluster for Trj and follow the three similar steps discussed in the previous paragraph. The main difference is that we need to insert a new entry for this new cluster to the C-tree (the insertion algorithm is in Section 4.4).

4.3 Selecting Representative Trajectory

There are two key requirements when selecting a representative trajectory. First, the error rate should be minimized. Second, the representative trajectory must satisfy the road-network constraint. By keeping these in mind, we design the following algorithm.

In a cluster, we find the trajectory with the highest support and then trim the trajectory from both ends to obtain the final representative trajectory. To illustrate it, we use the example in Figure 7. The cluster contains three types of trajectories: Trj_1, Trj_2 and Trj_3. Each trajectory is associated with a number of support, e.g., $support(Trj_1) =$ 10. Numbers on the last line indicates the original numbers of users on each road, e.g., $original(n_1n_2)$=15. Since Trj_1 has the highest support, let us have a further look at it. We compute the error rate E by treating Trj_1 as the representative trajectory. The support of the representative trajectory is the sum of all trajectories in the cluster. The reason behind is to maintain the same amount of trajectories after anonymization. In this example, if we use Trj_1 as the representative trajectory, we will have E = 58%.

Trj_1 (10):	n_1	—	n_2	—	n_4	—	n_7	—	n_8	—	n_9
Trj_2 (5):	n_1	—	n_2	—	n_4	—	n_7				
Trj_3 (6):			n_2	—	n_4	—	n_7	—	n_8		
original:		15		21		21		16		10	

Fig. 7. An Example of Selecting Representative Trajectory

$$E = (E_{n_1n_2} + E_{n_2n_4} + E_{n_4n_7} + E_{n_7n_8} + E_{n_8n_9})/5$$

$$= \frac{(\frac{21-15}{15} + \frac{21-21}{21} + \frac{21-21}{21} + \frac{21-16}{16} + \frac{21-10}{10})}{5} = 58\%$$

Observe that $E_{n_8n_9}$ is higher than 100%. If the road n_8n_9 is excluded from the representative trajectory Trj_1, the overall error can be reduced to 34%. Based on this observation, the second step is to trim the trajectory until the overall error rate cannot be further reduced. Due to the road-network constraint, we can not arbitrarily remove nodes from a trajectory. Our strategy is to remove nodes starting from both ends of the selected trajectory if the road r satisfies the following condition: $original_r < support(Trj_1) - original_r$, i.e., its individual error rate larger than 100%. The process continues until we cannot find such a road at either end of the trajectory. The final representative trajectory for the example case is $n_1n_2n_4n_7n_8$. The algorithm is summarized in Figure 8.

4.4 Construction of the C-tree

In Section 4.2, we have discussed the search and update operations in the C-tree. We now proceed to introduce how to insert a new entry into the C-tree, which occurs when a new cluster is created. Recall that each entry in the node of the C-tree has two fields: (i) a set of road IDs and (ii) a pointer. The maximum number of entries in each node is the same. All insertions start at a leaf node which is identified during the process of finding candidate clusters. We insert the new entry into that node (denoted as N) with the following steps:

1. If the node N contains fewer than the maximum legal number of entries, then there is room for the new entry. Insert the new entry in the node.

```
Select_Representative_Trajectory ($C$,$Trj_r$)
Input: $C$ is a cluster
Output: $Trj_r$ is the representative trajectory
1.  support($Trj_r$)$\leftarrow 0$
2.  for each $Trj$ in $C$ do
3.      if support($Trj$) $>$support($Trj_r$) then
4.          $Trj_r \leftarrow Trj$
5.          support($Trj_r$) $\leftarrow$ support($Trj$)
6.  $i \leftarrow 1$; $j \leftarrow length(Trj_r)$-1
7.  continue $\leftarrow 1$
8.  while ($i < j$ and $continue$) do
9.      continue $\leftarrow 0$
10.     if original($r_i$) $<$support($Trj_r$)-original($r_i$) then
11.         $i \leftarrow i + 1$; continue$\leftarrow 1$
12.     if original($r_j$) $<$support($Trj_r$)-original($r_j$) then
13.         $j \leftarrow j - 1$; continue$\leftarrow 1$
14. $Trj_r \leftarrow$($r_i$...$r_j$)
15. return $Trj_r$
```

Fig. 8. Algorithm of Selecting Representative Trajectory

2. Otherwise N is full, and we evenly split it into two nodes. In particular, we randomly select an entry as seed. Then we compute Sim_c (Equation 3) between other entries and the seed. The average of all Sim_c serves as a separation value. Entries with Sim_c above the average are put in the node N, and the remaining entries are put in the new right node N'.
3. Next, we update the entry pointing to N. The road ID set in the parent is updated to include all roads occur in N. The update may be propagated to the upper levels of the tree. Moreover, if there is a split in the previous step, we need to insert a new entry which includes road IDs in the new node N' to the parent level. This may cause the tree to be split, and so on. If current node has no parent (i.e., the node is the root), a new root will be created above this one.

5 Experimental Study

All the experiments were run on a PC with 2.6G Pentium IV CPU and 3GB RAM. We use both synthetic and real road networks to generate moving objects. In the synthetic datasets, objects are moving on a randomly generated road map which has about 700 roads. Objects can have different speeds which are controlled by the parameter "average trajectory length". As for the real datasets, we use the generator by Brinkhoff [6]. Objects are moving on real road networks. A road consists of multiple segments and each segment is a straight line. An object is initially placed on a randomly selected road segment and then moves along this segment in a randomly selected direction. When the object reaches the end of the segment, an update is issued and a connected segment is selected. Object speeds are varied within a given speed range.

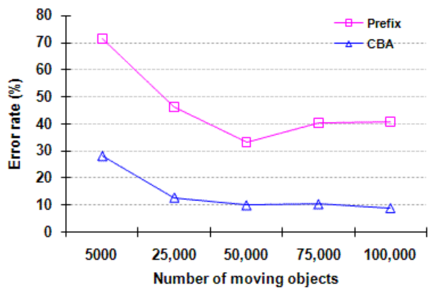
We compare our Clustering-Based Anonymization (CBA) algorithm with the latest work (denoted as Prefix [14]) by Pensa et al. In particular, we examine the existence of the inference route, the error rate, standard deviation and the running time. Unless noted otherwise we use the dataset containing 50,000 moving objects and set k to 30.

5.1 Experimental Results in Synthetic Datasets

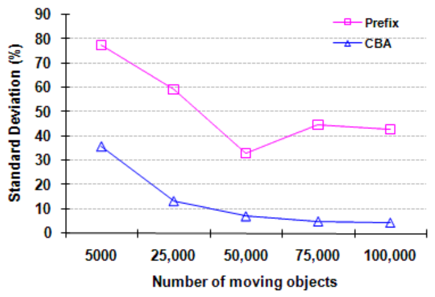
Effect of Data Sizes In the first set of experiments, we study the effect of data sizes by varying the number of moving objects (i.e. number of trajectories) from 5K to 100K. Figure 9(a) shows the average error rate of the anonymization results obtained from Prefix algorithm and our CBA algorithm. We can observe that the CBA algorithm yields much less error rate than the Prefix algorithm in all cases. When the dataset is small (e.g. 5K), the anonymization results obtained from both algorithms have relatively high error rates. This is because the number of objects on each road is few and even a small change of an object trajectory by the anonymization process will have a big impact on the error rate. With the increase of the data sizes, the error rate caused by the CBA algorithm keeps decreasing and it is more than 5 times less compared to that of the Prefix algorithm for 100K dataset. The reason of such behavior is that CBA effectively groups similar trajectories and carefully selects representative trajectories which minimize the overall error rate. Figure 9(b) shows the standard deviation, where we can see that our standard deviation is much lower than that obtained from the Prefix algorithm. This confirms that our anonymization result on each road has similarly good quality.

Figure 10(a) shows the number of nodes having the inference route problem. It is not surprising to see that the anonymization result produced by our CBA algorithm contains 0 inference route. However, the anonymized result obtained from the Prefix algorithm has many road intersections (denoted as node) with the inference route problem caused by their definition of the trajectory support.

We also compare the running time of both approaches. As shown in Figure 10, our CBA algorithm is up to 5 times faster than the Prefix algorithm. This can be attributed to the C-tree that helps prune the clusters to be compared with each new trajectory and hence avoids many unnecessary calculation. The total time is inclusive of the construction and update cost of the C-tree which is almost neglectable compared to the benefits brought by the C-tree.

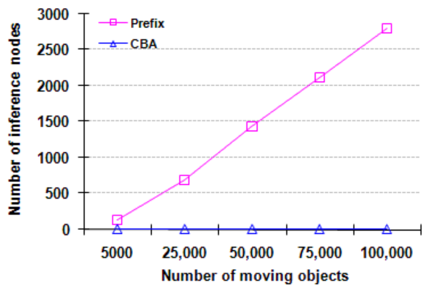


(a) Error rate

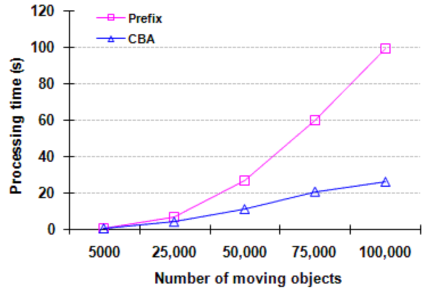


(b) Standard deviation

Fig. 9. Quality of the Anonymized Results



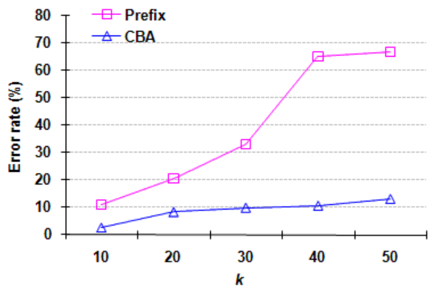
(a) Inference route problem



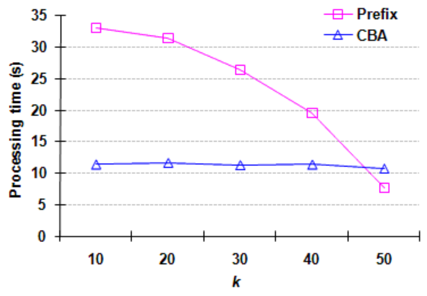
(b) Processing time

Fig. 10. Effect of Data Size

Effect of Parameter k This set of experiments aims to evaluate the performance of both algorithms regarding different values of k. As shown in Figure 11(a), the error rate increases drastically with k by using the Prefix algorithm, while k has only minor effect on our CBA approach. Such behavior can be explained as follows. The Prefix algorithm removes all infrequent trajectories and add their supports to most similar frequent trajectories. When k is large, there are more infrequent trajectories, which thus causes more error. The standard deviation has also demonstrated the similar pattern as the error rate, and the Prefix algorithm again suffers from the inference route problem. Due to the space limit, we do not include the figures here. Regarding processing time (in Figure 11(b)), our CBA approach has a consistent performance while the Prefix approach requires less time for larger k. This is because the Prefix approach needs to deal with less frequent trajectories for a larger k. Note that this results in higher error rates.



(a) Error rate



(b) Processing time

Fig. 11. Varying Parameter k

Effect of the Average Trajectory Length We also investigated the effect of trajectory lengths by testing it up to 50 roads per trajectory. Within the same time interval, a longer trajectory indicates that the object has a faster speed. Our CBA algorithm outperforms the Prefix algorithm in all cases. Please refer to our technical report [**?**] for figures.

5.2 Experimental Results in Real Datasets

In this set of experiments, the datasets are generated based on the road map of Phelps County (Missouri, USA) using the generator [6]. The value of k is 10. From Figure 12, we can observe similar performance patterns as that using synthetic datasets.

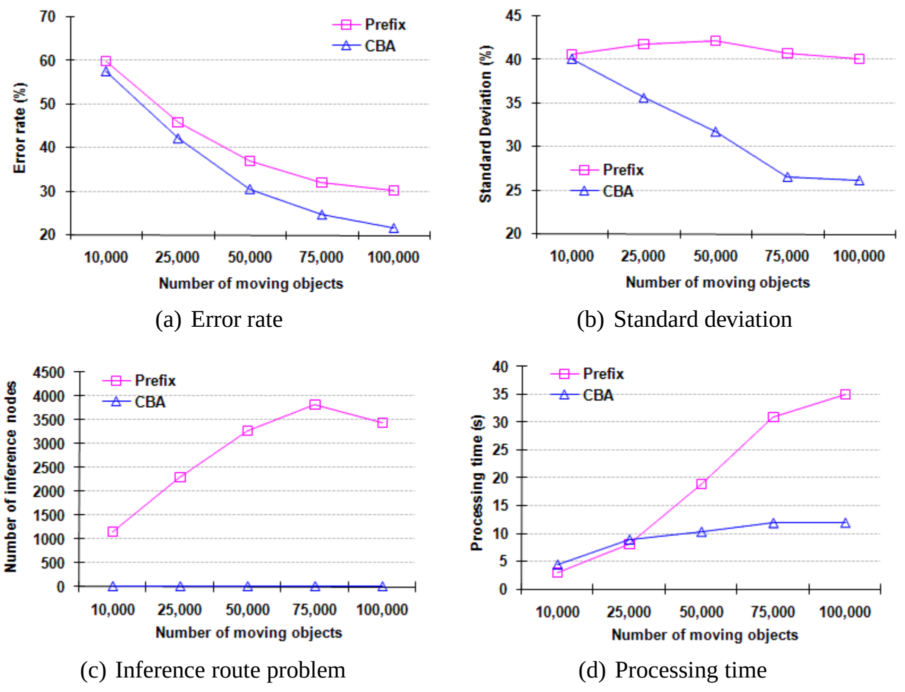


Fig. 12. Performance in Real Road-network

6 Conclusion

Privacy preserving location data publishing has received increasing interest nowadays. In this paper, we address this newly emerging problem by taking into account an important factor, the road network constraint, which has been overlooked by many existing works. We identified and defined a new privacy problem (i.e. the inference route problem), and proposed an efficient and effective clustering-based anonymization algorithm. The clustering-based algorithm guarantees strict k-anonymity of the published dataset and avoids the inference route problem. We compared our approach with the most recent work and the experimental results demonstrate the superiority of our approach.

Acknowledgement

We thank authors of “Pattern-Preserving k-Anonymization of Sequences and its Application to Mobility Data Mining” who share their source code with us. This work was supported in part by the National Science Foundation under the contract IIS-0324835 and UM Research Board under the project ”Preserving Location Privacy in Pervasive Environments”.

References

1. O. Abul, M. Atzori, F. Bonchi, and F. Giannotti. Hiding sensitive trajectory patterns. In *Proc. of ICDM Workshop*, pages 693 – 698, 2007.
2. O. Abul, F. Bonchi, and M. Nanni. Never walk alone: Uncertainty for anonymity in moving objects databases. In *Proc. of the International Conference on Data Engineering*, pages 376–385, 2008.
3. R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Proc. of the ACM SIGMOD International Conference on Management of Data*, pages 439–450, 2000.
4. M. Atzori, F. Bonchi, F. Giannotti, and D. Pedreschi. Anonymity preserving pattern discovery. *The VLDB journal*, 17(4):703–727, 2008.
5. C. Bettini1, X. S. Wang, and S. Jajodia. Protecting privacy against location-based personal identification. In *Proc. of Workshop on Secure Data Management*, pages 185–199, 2005.
6. T. Brinkhoff. A framework for generating network-based moving objects, 2004. `http://www.fh-oow.de/institute/iapg/personen/brinkhoff/generator`.
7. G. Gidofalvi, X. Huang, and T. B. Pedersen. Privacy-preserving data mining on moving object trajectories. In *Proc. of the International Conference on Data Engineering*, pages 60–68, 2007.
8. B. Hoh and M. Gruteser. Protecting location privacy through path confusion. In *Proc. of SecureComm*, pages 194–205, 2005.
9. B. Hoh, M. Gruteser, H. Xiong, and A. Alrabady. Preserving privacy in gps traces via uncertainty-aware path cloaking. In *Proc. of the ACM conference on Computer and Communications Security*, pages 161–171, 2007.
10. D. Lin, S. Gurung, W. Jiang, and A. Hurson. Privacy-preserving location publishing under road-network constraints. *Technical Report,http://web.mst.edu/ lindan/others/trajectory.pdf*.
11. M.F. Mokbel. Privacy in location-based services: State-of-the-art and research directions. In *Proc. of the International Conference on Mobile Data Management*, page 228, 2007.
12. M. E. Nergiz, M. Atzori, Y. Saygin, and B. Guc. Towards trajectory anonymization: a generalization-based approach. *Transactions on Data Privacy*, 2(1):47–75, 2009.
13. J. Pei, J. Han, B. Mortazavi-Asl, J. Wang, H. Pinto, Q. Chen, U. Dayal, and M. Hsu. Mining sequential patterns by pattern-growth: The prefixspan approach. *IEEE Transactions on Knowledge & Data Engineering*, 16(1):1424–1440, 2004.
14. R.G. Pensa, A. Monreale, F. Pinelli, and D. Pedreschi. Pattern-preserving k-anonymization of sequences and its application to mobility data mining. In *Proc. of the International Workshop on Privacy in Location-Based Applications*, 2008.
15. L. Sweeney. Achieving k-anonymity privacy protection using generalization and suppression. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5):571–588, 2002.
16. J.C. Tanner. In search of lbs accountability. In *Telecom Asia*, 2008.
17. M. Terrovitis and N. Mamoulis. Privacy preservation in the publication of trajectories. In *Proc. of the International Conference on Mobile Data Management*, pages 65–72, 2008.
18. R. A. Wagner and M. J. Fischer. The string-to-string correction problem. *Journal of the ACM*, 21(1):168–173, 1974.
19. R. Yarovoy, F. Bonchi, L. V. S. Lakshmanan, and W. H. Wang. Anonymizing moving objects: how to hide a mob in a crowd? In *Proc. of the International Conference on Extending Database Technology*, pages 72–83, 2009.
20. M. J. Zaki. Spade: An efficient algorithm for mining frequent sequences. *Machine Learning*, 42(1/2):31–60, 2001.