An Efficient Method on Trajectory Privacy Preservation

Zhiqiang Zhang(✉), Yue Sun, Xiaoqin Xie, and Haiwei Pan

College of Computer Science and Technology, Harbin Engineering University, Harbin, China
zqzhang@hrbeu.edu.cn

Abstract. Traditional trajectory *k*-anonymity method might lead to a serious information distortion of trajectory and reduce the data quality. This paper proposes an efficient method to protect trajectory privacy by protecting points of interest, and improve the data quality.

Keywords: Trajectory *k*-anonymity · Point of interest · Privacy requirement · Data quality

1 Introduction

Trajectories contain a lot of valuable information. However, since consecutive points are dependent on each other in a trajectory, the problem of anonymization on trajectories is more difficult. This paper puts forward a novel trajectory *k*-anonymity privacy protection method based on the points of interest, and a new way to compute similarity of trajectories is proposed.

Location *k*-anonymity requires each location to be indistinguishable with other at least *k*-1 locations [1]. However, researchers found that only protecting objects' locations is not enough for protecting trace information. So the concept of trajectory privacy protection is proposed. Abul et al. proposed (k, δ)-anonymity model based on the inherent uncertainty of positioning systems [2]. Nergiz et al. proposed another *k*-anonymity method based on notion [3]. In [4], *k*-anonymity is defined that an original trajectory T is generalized into a trajectory g(T) without time information. g(T) is a subset of the generalizations of at least *k*-1 other original trajectories. In [5], the authors propose the concept of adapted trajectory *k*-anonymity based on the bipartite attack graph about original and anonymized trajectories. In [6], authors propose trajectory *k*-anonymity based trajectory graph on the basis of linkage attack and observation attack.

When compute the trajectories' distance, the trajectories need to be synchronized through adding or removing some segments in order to unify trajectories in both the time and the space. This process has a lot of distortion before the trajectories were anonymized. Besides, traditional trajectory *k*-anonymity methods consider the privacy requirements of all points as the same, and need process all points during anonymization. However, processing all points in a whole trajectory is not only increasing time but also leading to serious distortion and reducing the data quality. Huo Zheng et al. [7] argue that background knowledge is more relevant to stay points. However we

Y. Wang et al. (Eds.): BigCom 2015, LNCS 9196, pp. 231–240, 2015.
DOI: 10.1007/978-3-319-22047-5_19

observe that some specific points besides stay points such as the start, the end, and the turning points are also important though further investigation. And adversaries' background knowledge is also relevant to these important points. Therefore only protecting stay points is not enough to protect the whole trajectory.

2 Preliminary

Definition 1 (Trajectory). A trajectory is a sequence of spatio-temporal points. It is represented as T={ID, (x_1 , y_1 , t_1), …, (x_n, y_n, t_n)} ($t_1 < \ldots < t_n$), where ID represents a trajectory, (x_i , y_i , t_i) ($1 \le i \le n$) represents a spatio-temporal point.

Definition 2 (Point of Interest, POI). POI means some points that users may be interested in, including the start, the end, stay points and turning points. POI is a 3-tuple (ID, *loc*, *t*), where ID represents the ID of trajectory which this POI belongs to, *loc*=(latitude, longitude) represents the geography coordinates of this POI, *t* represents the time when object passes by the point.

The start and the end in a trajectory are the source and destination of the moving object. Stay points are some locations where moving object stays for a long time and exceeds a certain threshold. Turning points can reflect the routes of a trajectory. It can be confirmed by the angle between two adjacent track fragments.

Definition 3 (Region of Interest, ROI). ROI is a region including some closer POIs. So each ROI is represented as (*id*, *centerPoint*, *radius*, Δt), where *id* represents an ROI, *centerPoint* represents the center point of the region, *radius* represents the radius of the region, Δt represents the time difference of region.

Definition 4 (Trajectory Similarity). Suppose the number of common ROIs of two trajectories is M, and the number of all ROIs of these two trajectories is N. Trajectory Similarity of two trajectories is M/N, signified by *Sim*. The value of *Sim* ranges from [0,1]. The formula is as follows.

$$\mathrm{Sim}\,(\mathrm{tr}_1, \mathrm{tr}_2) = \frac{|\,\mathrm{R}_1 \bigcap \mathrm{R}_2\,|}{|\,\mathrm{R}_1 \bigcup \mathrm{R}_2\,|} \tag{1}$$

Where R_1 and R_2 represent ROIs of two trajectories respectively.

Definition 5 (Trajectory *k*-anonymization Set). If a set consists at least *k* trajectories and the similarity of any two trajectories is 1, so we called these *k* trajectories as a *k*-anonymization set.

3 POI-Based Trajectory *k*-anonymity Privacy Protection Algorithms

The data released should guarantee that adversaries cannot re-identify moving objects' information using background knowledge, and meanwhile the quality of data

will not affect utility. Our proposed method consists of three main phases: Phase 1, extracting POIs of all trajectories in the database; Phase 2, clustering all POIs and forming ROIs, then partitioning trajectories on the basis of ROIs; Phase 3, anonymizing trajectories in each k-anonymization set separately and publication. Algorithm 1 gives a brief description about the method we proposed.

Algorithm 1. POI-based trajectory k-anonymity privacy protection algorithm

```
Input: Original trajectory database D
Output: Anonymous trajectory database D*
1.  for all tr in D do
2.      C = C ∪ ExtractPOI(tr); //Extracting POIs
3.  ROI = ClusterPOI(C); //Clustering POIs
4.  G = GroupTr(D,ROI); //Grouping Trajectories
5.  for all group in G do
6.      D* = D* ∪ Anonymity(group);
7.  return D*;
```

3.1 Extracting POIs

In this paper, our purpose is completely different from [8]. Hence, we consider POIs from two aspects: privacy requirements of moving objects and background knowledge of adversaries. POIs in this paper consist of the start and the end, stay points and turning points. The goal of the privacy protection of entire trajectory is achieved by the recognition and protection of these sensitive points.

The start and the end of a trajectory are added into the set directly during extracting POIs. If the angle of two adjacent segments exceeds a threshold which is set beforehand, the point which connects these two segments is a turning point. Then add the turning point into the set. Stay points are locations or approximate locations which the stay time exceeds a threshold. There are two cases: One is stopping for a long time at one location; the other is wandering for a long time near a location. For the first situation, we just need to judge the stay time between a point and the next one. If the stay time is larger than the predefined threshold, the point would be regarded as a stay point, and be added to the set. For the second situation, points which are near the location need to find first. If the spatial distance between two points is less than MinDist, these points will be looked as one location. A represented point is the nearest point to the center point. The stay time of this place is the time span between the first point and the last point. If the stay time exceeds the threshold MinStayTime, this represented point of this place is added into the set, the other points of this place are removed finally. Algorithm 2 gives a description about the POI extracting.

Algorithm 2. POI Extracting algorithm

Input: A trajectory T, MinAngle, MinDist, MinStayTime
Output: the POI set of the trajectory C={POIs}

```
1. initialize C = {point_1}, cur = 2;  //adding the start
2. while cur<n do     //n is the number of points in T
3.    Find the next point which the spatial distance not
less than MinDist with the cur point.
4.    Compute the time span Δt between the cur and the
next
5.    if Δt >= MinStayTime then
6.       if next == cur+1 then
7.         C = C ∪ {point_cur};
8.       else
9.       compute the center of points from cur to next;
10.      find the nearest point point_m with center;
11.        C = C ∪ {point_m};
12.    else
13.      compute angle;
14.      if angle >= MinAngle then
15.          C = C ∪ {point_cur};
16.    cur = next;
17.  C = C ∪ {point_n};   //adding the end
18.  return C;
```

3.2 POI Clustering and Trajectories Group

POIs which are near each other in space and time are put into the same cluster. When searching spatial and temporal neighborhood of all unvisited points in the set, if the number of POIs in the neighborhood is smaller than the density threshold, then mark the point as candidate noise preliminary. Otherwise, the point is a core point; it needs to build a new cluster. After all points in the set are visited, the clustering process is completed preliminary. At last, further process of POIs which are marked as "noise-Candidate" is done. In order to protect POIs in trajectories, points which are not belong to any cluster need to be removed. It is obvious that the more noise points, the larger information distortion is. In order to reduce information distortion, it needs that candidate noise points are added into the formed clusters as much as possible. So, this paper improves the ST-DBSCAN [9] method to cluster POIs. The noise points generated by original ST-DBSCAN which we called candidate noise are added into closest clusters as far as possible for reducing information loss. The detail of the improved algorithm is as follows.

Algorithm 3. POI Clustering algorithm

```
Input: POI set C, Spatial neighborhood radius Es,
Temporal neighborhood radius Et, Density Threshold MinPts
Output: Cluster = {cluster1, cluster2,..., clusterm}
1.  id = 0;
2.  mark all points in C as 'unvisited';
3.  for each point p in C do
4.     if (p is unvisited) then
5.        mark p as 'visited';
6.        N = getNeighbours(p, Es, Et);
7.        if (|N| < MinPts) then
8.           mark p as 'noiseCandidate';
9.        else
10.          id++;
11.          expandCluster(p, N, id, Es, Et, MinPts);
12.  for all noiseCandidate nc do
13.     ncN = getNeighbours(nc, Es, Et);
14.     if (|ncN| > 0) then
15.         add nc to the nearest cluster,
16.     else
17.         mark nc as 'noise';

expandCluster(p, N, id,  Es , Et, MinPts)
1.  add p to cluster id;
2.  for each point q in N do
3.    if (q is unvisited) then
4.        mark q as 'visited';
5.        M = getNeighbours(q, Es, Et);
6.        if (|M| >= MinPts) then
7.            N = N ∪ M;
8.    if (q is not yet member of any cluster) then
9.        add q to cluster id;
```

The candidate noise points which do not belong to any clusters finally mark as noise points and are removed at the published trajectories. After Clustering of POIs, a series of clusters which each one produces an ROI are formed. The core point of a cluster is the center point of the corresponding ROI. A ROI's radius and time difference of the region are the maximum space and time difference of center point and other points in the cluster respectively.

After partition ROIs, we find trajectory *k*-anonymization set using trajectory similarity defined in previous section. For achieving the requirements of *k*-anonymity, the size of each trajectory group should between *k* and 2*k*-1. In addition, in order to reduce information distortion, trajectories which do not belong to any groups after partition should be added into existing groups as far as possible. If the similarity with

any trajectory in a group is more than α, then it can be added into this group. The value of α is set to 1. Finally, as same as traditional *k*-anonymity, trajectories in a group which the size is less than *k* need to be removed at the published data. The trajectory grouping algorithm is as follows.

Algorithm 4. Trajectory Grouping Algorithm

Input: POI set C, k
Output: Trajectory Group = {G_1, G_2, ..., G_m}
1. Cluster POIs in C by calling Algorithm 3,
2. let R_1, R_2, ..., R_m are ROIs generated after clustering,
3. for all tr in D do
4. Compute similarity *sim* between tr and other trajectories, and put trajectories into one group if their *sim*=1;
5. for all trn not in any group do
6. Compute similarity *sim* between trn and existed groups;
7. if (*sim* >= α) do
8. Add trn into this group,
9. put all groups which size no less than *k* into Group;
10. return Group;

3.3 Trajectory Anonymity Publication

The trajectories have been clustered, aiming at trajectories in each group indistinguishable through anonymous methods, such as perturbing, generalization or characteristic publication. Perturbing means anonymous trajectories cannot match with original ones by reconstructing adjacent points or segments of trajectories in each group. After anonymity, if the trajectory group size is at least k, so since these k trajectories are indistinguishable, the re-identify probability of every trajectory is no less than 1/k, and this is trajectory k-anonymity.

For protecting the reality of locations, some researchers propose two methods [10]: SwapLocations and ReachLocations. The former adopts the way to find the near point set of every point in each *k*-anonymization set and swapping randomly. This method guarantees trajectory *k*-anonymity. The latter adopts swapping randomly among near locations in the whole dataset and only spatial information is swapped, and this method satisfies only location l-diversity [11]. Based on the above idea, this paper applies swapping POIs for anonymity, so the original locations are held in the released dada. Since we protect trajectories through POIs protection, only the real locations of each ROI in each group will be swapped randomly.

When extracting stay points in the first stage, some stay points are represented point of some near points; in this condition, other non-represented ones are removed and only hold the represented one in the released trajectories. After swapping

randomly, the reconstructed trajectories are generated. The purpose of trajectory privacy protection is achieved.

Our algorithm firstly marks POIs of all ROIs as un-swapped, and then swaps un-swapped POIs in each ROI randomly until all points are swapped at least once. After swapped, the reconstructed trajectories are added into the anonymous trajectory set finally. The anonymity of a trajectory group has finished.

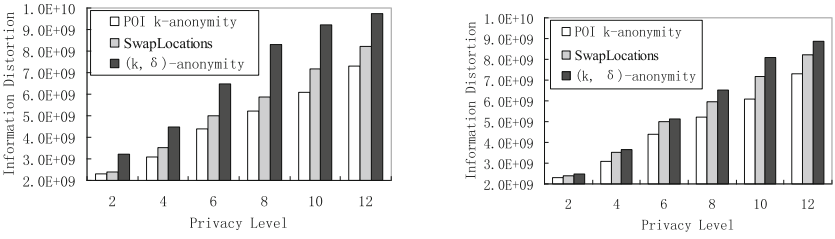
4 Experiments

Our experiments are run on an Intel Core 2 Quad 2.66HZ, Windows XP machine equipped with 2GB main memory, JDK1.7. The dataset we used is real trajectories of volunteers collected by GeoLife project, which contains 18670 trajectories of 182 users from April, 2007 to August, 2012. We adopt the same parameter values with the [8], MinDist = 200 meters and MinStayTime = 20 minutes. The value of MinAngle is set 30 degree. Finally, we extract 270,666 POIs from all 24,876,978 points, and then cluster these POIs and divide ROIs.

In this paper, we measure the data utility from two aspects: information distortion and spatio-temporal range queries. In order to verify the availability of our proposed method, we compare with the typical method (k, δ)-anonymity (NWA) [3] and the SwapLocations method proposed in [11]. Trajectories are clustered using greedy cluster method and finally each cluster is converted to (k, δ)-anonymity set, where δ represents the radius of set. Due to the rapid development of technology, the accuracy of some positioning devices can reach centimeter-level, even millimeter. So we set the values of δ are set to 0, and 2000.

4.1 Information Distortion

This paper uses the same equation to calculate information distortion with [3]. Two different attributes are applied to compare, which are total space distortion and the percentage of removed points. When computing total space distortion, the value of Ω is 0. The percentage of removed points is considered alone. So the value of Ω can be set based on the different applications for computing the whole distortion.

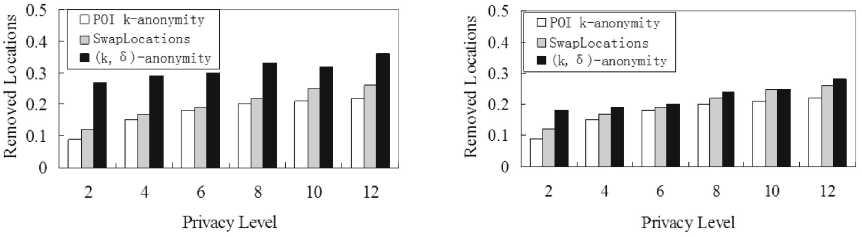


(a) δ = 0 (b) δ = 2000

Fig. 1. Information distortion measurement

Fig.1 describes the information distortion contrast with NWA and SwapLocations when Ω=0, which means without considering the distortion caused by points removing. From the figure we can see our method leads to less space distortion than the other methods. This is due to our method only swap POIs of every *k*-anonymity set in the same ROI for achieving anonymity. SwapLocations method swaps all points in each *k*-anonymity set for cloaking, which leads to relatively large information distortion. For (*k*, *δ*)-anonymity model, when δ>0, the distortion considers not only the space distortion of points within uncertainty range, but also the distortion caused by space translation which moves some points not in uncertainty range to another locations from original locations. This greatly increases the information distortion.

From Fig.2, the POI-based trajectory *k*-anonymity method discards less points than SwapLocations and (*k*, δ)-anonymity. This is because the distance of trajectories is not considered when grouping trajectories; as long as these trajectories pass through the same ROIs, they will be classified into the same group. SwapLocations clusters trajectories through calculating distance between trajectories.



(a) δ = 0 (b) δ = 2000

Fig. 2. Percentage of removed locations

In addition, we can also find from Fig.1 and Fig.2, the gap of space distortion and the removed points' size among (*k*, δ)-anonymity with other two methods decreases gradually as the value of δ increases. This is because, with δ increasing, the number of points lied out of uncertainty range is few, which means the space distortion caused by space translation decreasing. Meanwhile, the number of trajectories not in the anonymous set is decreasing as δ increasing, and the number of discarded points is also reduced, so the gap between these methods is also becoming small.

4.2 Spatio-Temporal Range Queries

There are six types of spatio-temporal range queries introduced in [12], aiming at evaluating the relative position of a moving object with respect to a region *R* in a time interval [t_b, t_e]. In particular, when evaluating data utility we are more interested in whether a trajectory t_r is in the region *R* which mark as inside(*R*, t_r). Like [3], we do experiments on original dataset D and anonymous dataset D* separately focusing on two cases: Possibly_Sometimes_Inside (PSI) and Definitely_Always_Inside (DAI), and then compute the distortion of query results. The way to calculate Range Query Distortion is shown as equation (2)

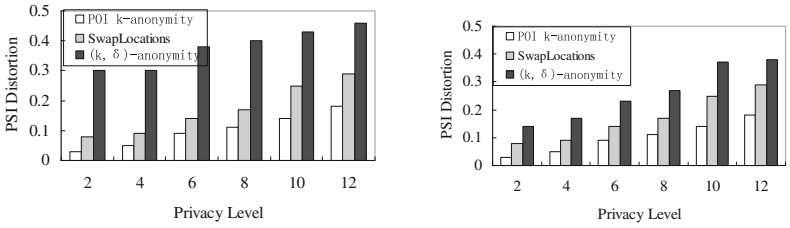
$$RQD(D,D^*)=\frac{1}{n}\sum\frac{|Q(D)-Q(D^*)|}{\max(Q(D),Q(D^*))} \tag{2}$$

In the above formula, query Q represents PSI or DAI defined above, n is the number of executed queries for each type of query.

We adopt the same parameters with [3], and the queries are generated by randomly chosen circular regions having radius between 500 and 5000, and randomly chosen time interval [t_b, t_e] with duration ranging from 2 to 8 hours. Finally 1000 different queries run in anonymous dataset generated by both methods, and then calculate the average query distortion.

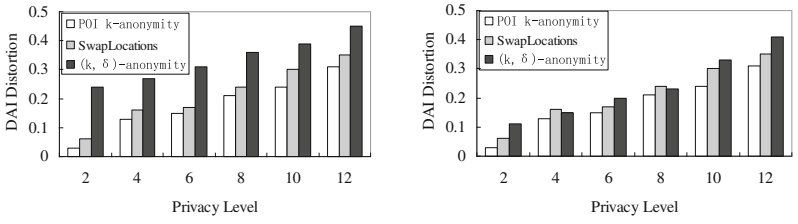
The DAI query distortion of our method is less than SwapLocations method and (k, δ)-anonymity. This is because only POIs are swapped for anonymity in our method and other points keep the same except the removed ones. SwapLocations method adopts cloaking all sample points of every anonymous set, which lead to large query distortion. However, since (k, δ)-anonymity adopts space translation, many sampling points are translated to totally different ones, making the largest loss of query results.

From Fig.3 and Fig4, we can also find that the gap of query distortion among (k, δ)-anonymity with the other two methods is less and less as δ increasing. In fact, when the value of δ tends to infinity, (k, δ)-anonymity means no trajectories needs to be anonymized, and the anonymous database is as same as the original one. Therefore, the impact on anonymous results will be small with large δ.



(a) δ = 0 (b) δ = 2000

Fig. 3. PSI Query Distortion



(a) δ = 0 (b) δ = 2000

Fig. 4. DAI Query Distortion

5 Conclusions and Future Work

This paper proposes POI-based trajectory *k*-anonymity method for solving privacy leakage problem on trajectory publication. We differ from the traditional trajectory *k*-anonymity on this point that our method mainly aims at *k*-anonymity of trajectories reconstructed by points of interest. Finally, location swapped method is adopted to release anonymous trajectories. The experiments proved our method have improved the quality of the released data.

Acknowledgement. This paper is supported by the National Natural Science Foundation of China under grant No. 61370084, 61202090, 61272184, the Program for New Century Excellent Talents in University No. NCET-11-0829, the Natural Science Foundation of Heilongjiang Province under grant F201130, and the Fundamental Research Funds for the Central Universities under grant No. HEUCF100609, HEUCFT1202.

References

1. Gruteser, M., Grunwald, D.: Anonymous usage of location-based services through spatial and temporal cloaking. In: Proceedings of the 1st International Conference on Mobile Systems, Applications and Services, pp. 31–42. ACM (2003)
2. Abul, O., Bonchi, F., Nanni, M.: Anonymization of moving objects databases by clustering and perturbation. Information Systems **35**(8), 884–910 (2010)
3. Nergiz, M.E., Atzori, M., Saygin, Y.: Towards trajectory anonymization: a generalization-based approach. In: Proceedings of the SIGSPATIAL ACM GIS International Workshop on Security and Privacy in GIS and LBS, pp. 52–61. ACM (2008)
4. Monreale, A., Andrienko, G.L., Andrienko, N.V., et al.: Movement Data Anonymity through Generalization. Transactions on Data Privacy **3**(2), 91–121 (2010)
5. Yarovoy, R., Bonchi, F., Lakshmanan, L.V.S., et al.: Anonymizing moving objects: how to hide a MOB in a crowd?. In: Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology, pp. 72–83. ACM (2009)
6. Huo, Z, Huang, Y, Meng, X.: History trajectory privacy-preserving through graph partition. In: Proceedings of the 1st International Workshop on Mobile Location-Based Service, pp. 71–78. ACM (2011)
7. Huo, Z., Meng, X., Hu, H., Huang, Y.: *Y*ou *c*an *w*alk *a*lone: trajectory privacy-preserving through significant stays protection. In: Lee, S.-g., Peng, Z., Zhou, X., Moon, Y.-S., Unland, R., Yoo, J. (eds.) DASFAA 2012, Part I. LNCS, vol. 7238, pp. 351–366. Springer, Heidelberg (2012)
8. Zheng, Y, Zhang, L, Xie, X, et al.: Mining interesting locations and travel sequences from GPS trajectories. In: Proceedings of the 18th International Conference on World Wide Web, pp. 791–800. ACM (2009)
9. Birant, D., Kut, A.: ST-DBSCAN: An algorithm for clustering spatial–temporal data. Data & Knowledge Engineering **60**(1), 208–221 (2007)
10. Domingo-Ferrer, J, Sramka, M, Trujillo-Rasúa, R.: Privacy-preserving publication of trajectories using microaggregation. In: Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Security and Privacy in GIS and LBS, pp. 26–33. ACM (2010)
11. Machanavajjhala, A., Kifer, D., Gehrke, J., et al.: L-diversity: Privacy beyond k-anonymity. ACM Transactions on Knowledge Discovery from Data (TKDD) **1**(1), 3 (2007)
12. Trajcevski, G., Wolfson, O., Hinrichs, K., et al.: Managing uncertainty in moving objects databases. ACM Transactions on Database Systems (TODS) **29**(3), 463–507 (2004)