Computer Communications 190 (2022) 57–68

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Decentralized k-anonymization of trajectories via privacy-preserving tit-for-tat

Josep Domingo-Ferrer *, Sergio Martínez, David Sánchez

Universitat Rovira i Virgili, Department of Computer Engineering and Mathematics, CYBERCAT-Center for Cybersecurity Research of Catalonia, Av. Països Catalans 26, 43007 Tarragona, Catalonia, Spain

ARTICLE INFO

Keywords:
Privacy
P2P
k-anonymity
Decentralized anonymization

ABSTRACT

Mobility data, and specifically trajectories, are used to monitor the mobility of the population and are crucial to improve public health, transportation, urban planning, economic planning, etc. However, trajectories are personally identifiable information and hence they should be anonymized before releasing them for secondary use. Anonymization cannot be limited to suppressing the metadata containing the subject's identity, because the origin, the destination and even the intermediate points of a trajectory may allow re-identifying the subject who followed it. Proper anonymization requires masking detailed spatiotemporal information. The standard approach to build anonymized data sets is centralized: the subjects send their original movement data to a controller, who takes care of producing an anonymized mobility data set. This requires subjects to blindly trust the controller. In this paper, we empower subjects with the ability to anonymize their trajectories locally by adhering to a privacy model in order to achieve formal privacy guarantees. After reviewing the state of the art, we motivate our choice of k-anonymity as a privacy model. We then set out to decentralize k-anonymity in a *rational* setting: a subject k-anonymizes her *completed* trajectory by aggregating with $k-1$ similar trajectories obtained from other (unknown) subjects. The latter trajectories are gathered via an anonymous and privacy-preserving tit-for-tat data exchange protocol, which runs on a fully decentralized peer-to-peer network. Experiments show that, without relying on a (trusted) data controller and while ensuring privacy w.r.t. other peers, our approach yields k-anonymized mobility data sets that are still reasonably useful compared to the near-optimal data sets obtained in the centralized approach.

1. Introduction

Knowing how people move is very useful for researchers and planners in order to study human behavior and thereby improve public health, transportation, urban planning, economic planning, etc. See [1] for a survey of applications. Quite recently, we have also seen how mobility data derived from cellphones can be extensively used to fight the spread of Covid-19 [2,3].

However, trajectories, where a trajectory is an ordered sequence of points and a point is a time-stamped location, are personally identifiable information. Hence, according to the EU General Data Protection Regulation [4], they cannot be released for secondary use unless they have been anonymized (or all moving subjects have given their consent for such a secondary use, which is unlikely). Anonymization must go beyond suppressing metadata containing the subject's identity and must involve masking the spatio-temporal points of the trajectory. Indeed, the origin, the destination and even the intermediate points of a trajectory may allow re-identification of the subject. Think of a person living in a suburban villa and working in an industrial estate: the origin and the destination of that person when driving to her work every morning are most likely unique to her and, hence, may allow re-identification. When this happens, the remaining points of that person's trajectory, which may be confidential or otherwise sensitive, will be unequivocally attributed to her.

According to a recent state of the art on anonymization of mobility data [5], all the methods in the literature turn out to be centralized: a central data controller gathers all original trajectories and anonymizes them, in order to obtain an anonymized data set that can be released or shared. Thus, although the released anonymized data set may not disclose personal information to third parties, the moving subjects must reveal their original mobility patterns to the controller. This is a serious privacy threat (*e.g.*, due to possible attacks and data leakages) and requires subjects to blindly *trust* the controller.

1.1. Contribution and plan of this paper

To tackle these issues, in this paper we empower subjects with the ability to anonymize their *completed trajectories* locally in a *rational* decentralized setting. In this way, they do not need to release their

* Corresponding author.
E-mail addresses: josep.domingo@urv.cat (J. Domingo-Ferrer), sergio.martinezl@urv.cat (S. Martínez), david.sanchez@urv.cat (D. Sánchez).

https://doi.org/10.1016/j.comcom.2022.04.011
Received 11 January 2022; Received in revised form 6 April 2022; Accepted 8 April 2022
Available online 16 April 2022

original trajectories to a central data controller. Since we focus on producing general-purpose anonymized data sets with formal privacy guarantees, we look for a privacy model that does not limit the uses of the released data. Based on the review of the state of the art that we conduct in Section 2, we justify our choice of the k-anonymity privacy model. Among the computational methods to achieve trajectory k-anonymity, we opt for microaggregation [6] because it can preserve the utility of the protected data better than methods based on generalization, and we set out to craft a decentralized version of it. In a nutshell, in our approach a subject k-anonymizes her trajectory by aggregating it with a set of $k-1$ similar trajectories obtained from other (unknown) subjects. The latter trajectories are gathered via an anonymous tit-for-tat data exchange protocol, which runs on a fully decentralized peer-to-peer network and ensures that no private information is leaked to other peers.

Experiments show that, without relying on a central data controller, our approach yields k-anonymous mobility data sets that are still reasonably useful compared to the near-optimal data sets obtained in a centralized manner.

The rest of this paper is organized as follows. Section 2 reviews the state of the art in trajectory anonymization and justifies our adoption of k-anonymity as a privacy model. Section 3 provides background on k-anonymity via microaggregation. Section 4 describes our decentralized trajectory anonymization method. Experimental work is presented in Section 5. Conclusions and future work are gathered in Section 6.

2. Related work

The methods in the literature use one of the three following principles to protect mobility data [5]: (i) mitigation, that is, cloaking, segmentation or swapping, (ii) indistinguishability, based on the k-anonymity privacy model, and (iii) uninformativeness, based on differential privacy.

Mitigation methods reduce privacy risks in data sets without following a well-defined privacy principle, that is, without adhering to a specific privacy model. Several heuristic solutions predicated on different data transformations have been proposed to reduce the privacy risk associated with trajectory data. *Cloaking* consists in reducing the granularity of the space and/or time dimensions of trajectory data. For example, in [7] the authors randomly remove a percentage of locations from the original trajectories to reduce the risk of re-identification. In [8] the authors reduce the granularity of mobility data by decreasing the number of decimals in latitude and longitude values. *Segmentation* is a simple technique that attenuates trajectory uniqueness by slicing original trajectories into several trajectories [9]. In this case, the utility of the data is significantly reduced, without nonetheless guaranteeing elimination of uniqueness. *Swapping* is a method where parts of trajectories are swapped between users [10]. To reduce information loss, trajectories are selected for swapping based on their similarity. Since all the aforementioned mitigation methods leverage predefined data transformations, they do not completely eliminate the possibility of unequivocal re-identification, because the trajectories of some subjects may still be unique in the data set [11]. As a result, they cannot offer privacy guarantees.

Indistinguishability methods are based on the k-anonymity privacy model [12]. In the context of mobility data, k-anonymity means that each trajectory in the data set is modified for it to be confused with at least $k-1$ other trajectories, thereby forming a k-anonymous class. Trajectories within a k-anonymous class are indistinguishable from each other. Compared to mitigation approaches, k-anonymity based methods completely prevent unequivocal re-identifications in data releases. Methods employing k-anonymity to protect mobility data are the following. In [13], k-anonymity is achieved by grouping nearby users at the start of their trajectories and always reporting the region containing the k users as their location. The authors in [14] propose a group-based approach by defining a distortion function founded on the velocity of each user. When creating the k-sized groups, sets are sought to minimize the distortion function along the trajectory. The GLOVE algorithm proposed in [15] aims to protect the set of trajectories via k-anonymity by using generalization and (possibly) suppression of spatiotemporal points. The authors define a metric for anonymizability of a given trajectory named k-gap, which estimates how difficult it is to hide the trajectory in a data set, according to the accuracy loss required to make a trajectory indistinguishable from other $k-1$ trajectories. In this method, regions of space containing points of the trajectories are stretched so that they are k-anonymous; a similar procedure is followed for the time dimension. In [16], the authors propose an anonymization method based on k-anonymity through the generalization and suppression of spatial (but not temporal) data. They assume that an attacker may know a sub-trajectory (a trajectory contained in another trajectory) of her victim. The generalization method consists in extracting characteristic points of the trajectories in the original dataset, clustering these characteristic points, and obtaining the centroid of each of the clusters. The centroids are then used as generating points for a Voronoi tessellation of the area (using the Euclidean distance), and the trajectories are converted into visits to each of the cells, using the centroid as location. To ensure that each of the Voronoi cells contains at least k visits, less frequently visited adjacent cells are iteratively merged by recomputing the centroids of these areas and repeating the tessellation process. In [17], the authors propose a k-anonymous method based on clustering. Clusters of trajectories are anonymized by generalizing matching points into minimum bounding boxes. Unmatched points are suppressed, and synthetic trajectories are generated by sampling the bounding boxes. In [6], k-anonymity is enforced via microaggregation of trajectories. Microaggregation consists in grouping trajectories according to their similarity and replacing them by group representatives. To this end, the authors introduce a *synchronized trajectory distance*, which is computed in two steps: (i) synchronization of trajectories, which results in time-overlapped trajectories, and (ii) computation of the Euclidean distance between contemporary points. All non-overlapping points are suppressed.

Uninformativeness methods are based on ϵ-differential privacy (DP, [18]), which was initially introduced for privacy-preserving data mining. Differential privacy is effective at protecting the answers to numerical or categorical queries by adding Laplace noise or other forms of randomization calibrated according to ϵ and the sensitivity of the query to changes in the data of a subject. In the context of mobility data, DP has been successfully employed to protect the answers to interactive queries [19,20]. Other works aggregate and publish differentially private statistics from trajectory databases [21,22], or from streamed mobility data [23]. However, these works do not allow general-purpose privacy-preserving data publishing (PPDP) of trajectory data sets. Indeed, since by definition differential privacy attempts to make the presence or absence of any single record unnoticeable from the DP-protected output, it is difficult to use it to protect releases of microdata (records corresponding to individual subjects) without either (i) severely hampering data utility, or (ii) significantly relaxing the offered privacy guarantee [24].

As mentioned in [25], k-anonymity is the most widely used definition of privacy for location-based systems in the literature, whereas DP is only successful when applied to *aggregate* information on several users. This is why, rather than directly applying DP to trajectory data, a mix of DP and k-anonymity has been suggested (first k-anonymize and then use DP on the k-anonymous data, [26]). In fact, [25] propose to avoid the need for prior aggregation via k-anonymity by replacing differential privacy with a variant called geo-indistinguishability (which is no longer DP proper). This variant, rather than real DP, is used in recent works such as [27].

To obtain precise privacy guarantees, we wish to adopt a principled approach based on a privacy model. This leaves us with a choice between k-anonymity and DP. As explained in the above state of the art, using strict DP only is not practical, so we select k-anonymity, which

has the additional advantage of being the most common privacy model for trajectories. Our focus will be on achieving trajectory k-anonymity in a decentralized way. We leave it for future research to combine our approach with DP.

3. Background on k-anonymity and microaggregation of trajectories

k-Anonymity [12] limits the capability of an attacker who knows a set of features on a subject (some locations in the case of trajectory data) to perform successful re-identifications in a released data set. A data set is said to satisfy k-anonymity if, for each combination of values (locations), at least k records (trajectories) share that combination. Thus, the probability of correct re-identification is, at most, $1/k$.

In conventional microdata, with one record per subject, one can distinguish between (quasi-)identifying attributes, whose values may be publicly known for a certain subject, and confidential attributes. For example, *date-of-birth* is a quasi-identifying attribute whereas *income* is typically confidential. Therefore, for conventional microdata, anonymization is usually performed on (quasi-)identifying attributes only (to avoid re-identification) while leaving confidential attributes untouched (to preserve data utility), because the latter cannot be unequivocally attributed to single subjects. However, in trajectory microdata, with one trajectory per subject, the points reporting the subject's successive locations (latitude, longitude and time) play the role of attributes, but they cannot be split *ex ante* into quasi-identifying and confidential. If an attacker knows as background knowledge a few points of his target's trajectory, then these points are quasi-identifying; otherwise, they are confidential. Thus, to play it safe, all points must be subjected to anonymization.

Even though k-anonymity has usually been enforced via generalization of values, this entails a large information loss for high-dimensional and spread data such as trajectories [6]. A more utility-preserving alternative to generalization is *microaggregation* [6,28]. Trajectory microaggregation is based on partitioning the data set into disjoint clusters containing each at least k trajectories. After the partition stage is complete, the aggregation stage aggregates the trajectories within each cluster by replacing them with the cluster centroid.

Algorithm 1 gives a heuristic for k-anonymizing a set of trajectories via *centralized* microaggregation. Given a set T with $|T|$ trajectories, the algorithm first calculates the centroid trajectory of T, say c, which is defined as the trajectory in the data set that minimizes the sum of distances to the rest of trajectories. Then, the algorithm constructs a cluster around the most distant trajectory from c within T, say trajectory r, by placing in the cluster trajectory r and the $k-1$ closest trajectories to r. After that, it takes the most distant trajectory from c among the remaining unclustered trajectories, say r', and it creates another cluster around r' including r' and the $k-1$ unclustered trajectories closest to r'. The algorithm iterates until fewer than $2k$ unclustered trajectories remain, and it forms a final cluster with them. Finally, it replaces each original trajectory with the centroid trajectory of the cluster to which the trajectory belongs. Since clusters are built by grouping the closest trajectories together, the resulting microaggregated data set minimizes the information loss incurred when enforcing k-anonymity.

Algorithm 1: Centralized k-anonymization of trajectories based on microaggregation

input : T: data set of trajectories
k: the desired level of privacy
output: T': k-anonymized version of T

```
Calculate the centroid c of the complete data set T;
n = |T|;
while n ≥ 2k do
    Select the most distant trajectory r from the centroid c;
    Generate a cluster formed by trajectory r and the k − 1
      trajectories closest to r among those that remain
      unclustered;
    n = n − k;
end
Form a cluster with the remaining unclustered trajectories;
Replace each original trajectory by the centroid of the cluster it
  belongs to;
```

4. Trajectory anonymization based on tit-for-tat

In the following, we propose a decentralized method in order to generate k-anonymous sets of trajectories that avoids the need for each of the moving subjects to reveal her exact trajectory to a central controller. With our method, each individual subject can k-anonymize her trajectory locally. To that end, each subject should be able to find other subjects with similar trajectories, and engage in a trajectory exchange with them in order to achieve k-anonymity via microaggregation. We design such exchange so that it is anonymous and no private information is leaked to other peers.

We assume a pre-existing peer-to-peer (P2P) network, in which peers are rational and are equipped with digital signatures. More specifically, we rely on a fully decentralized P2P network in which users holding their own trajectories act as peers that dynamically enter the network to anonymously gather similar trajectories from other peers.

There are several alternatives to implement decentralized P2P networks such as Chord [29] or Pastry [30]. Pastry is especially interesting due to its proximity-based routing. It consists in an overlay network that provides a fault-tolerant and load-balanced distributed hash table (DHT) for large-scale P2P applications. Each P2P user is identified by a unique *nodeId*, which acts as a pseudonym that hides the real identity of the user. The DHT is locally used to map a *nodeId* to its associated IP address.

4.1. Trajectory anonymization protocol

We assume a decentralized setting in which the peers willing to anonymize their completed trajectories are *rational*. To k-anonymize trajectories while minimizing the information loss caused by microaggregation, similar trajectories should be gathered and microaggregated together. For this to be possible, peers must anonymously exchange their trajectories with other peers whose trajectories (partially) coincide with or, at least, are similar enough to their own. Within a decentralized network, a straightforward way to find peers with similar trajectories is to broadcast to all active peers a message with the request. However, this incurs severe privacy disclosure. Moreover, broadcasting is very inefficient since it floods the network with messages that are useless to most peers. Instead, we use multicast group communications, based on a topic subscription system implemented by the network in which (i) users only exchange data with peers having similar trajectories and (ii) their detailed spatio-temporal locations are only disclosed to matching peers in an anonymous way. In short, a peer who wants to find trajectories similar to her own *subscribes* to *topics* associated with her own trajectory features, and then *publishes* her generalized trajectory in the appropriate topic in the subscription system. Published trajectories are generalized so that (i) they do not disclose the specific locations of the publisher's trajectory and (ii) they can still be used as a model to find similar trajectories from other peers.

Using multicast group communication, peers can subscribe to any topic by defining a *topicId*. If the topic does not exist, it is automatically created. The key here is to create the *topicId* as a generalization of the trajectory by means of a fixed criterion, so that peers with similar spatio-temporal trajectories generate the same *topicId*. In this way, matches between similar trajectories can be found. In our system, the spatial dimension of the trajectory is generalized by replacing specific locations by the coordinates of a *square* containing the location, and the

temporal dimension is generalized by replacing specific times by time intervals. Then, peers can send messages to the *topicId* in such a way that only peers having subscribed to the *topicId* (and thus, with similar trajectories that produce the same *topicId*) receive the messages.

Once a peer finds a matching between her local trajectory and a (generalized) trajectory published by another peer (the publisher), the former peer sends her local trajectory to the latter peer. After the publisher gathers at least $k-1$ matching trajectories, he will microaggregate them with his own trajectory to generate and release a k-anonymous data set. By running this in parallel for all peers, a k-anonymous data set encompassing all the trajectories of all peers in the network is generated in a decentralized way via multiple local microaggregations.

Since matching local trajectories are sent in the clear to the publisher, we need a mechanism to ensure the anonymity of the sender. To do this, instead of having the sender directly send her local trajectory to the publisher, we allow her to forward her trajectory to a third peer. In turn, this third peer will decide randomly whether he sends the trajectory to the publisher or rather forwards the trajectory to another peer. This process is iterated until a peer sends the trajectory to the publisher. Since the sending process may involve several hops, a publisher receiving a trajectory does not know whether the peer sending it is its owner (the peer who followed that trajectory) or she is simply forwarding a trajectory she has received from another peer. In this manner, we effectively break the link between the trajectory and her owner, thereby achieving anonymity.

To motivate intermediate peers to forward trajectories (rather than just drop them) and therefore make the protocol sustainable, we propose a tit-for-tat mechanism by which the peers that do not fulfill the forwarding requests by other peers will be blacklisted. Blacklisted peers will not get answers from other peers when trying to forward their own trajectories. Therefore, it is in the interest of rational peers to avoid being blacklisted, which requires them to adhere to the protocol and fulfill requests from other peers.

Finally, since anonymous trajectories from individuals may still allow re-identification and disclose confidential information (*i.e.*, if a subset of the trajectory locations are unique and known to belong to an individual and this can be matched to the remaining sensitive points of the trajectory), we allow peers to send trajectory fragments or even individual trajectory locations rather than complete trajectories via the anonymous communication introduced above. In the extreme case, where just anonymous individual points are received, neither re-identification nor disclosure of sensitive locations would be possible.

Protocol 1 describes the overall process. The protocol uses the following notation and parameters:

- $GridSize$ indicates the size of the squares used to generalize the spatial dimension of a trajectory. $MinGridSize$ indicates the minimum value of $GridSize$. Initially, $GridSize = MinGridSize$.
- $TimeInterval$ indicates the time interval used to generalize the temporal dimension of a trajectory. $MinTimeInterval$ indicates the minimum value of $TimeInterval$. Initially, $TimeInterval = MinTimeInterval$.
- (lat_{ini}, lon_{ini}) and (lat_{end}, lon_{end}) correspond, respectively, to the upper-left and lower-right map coordinates of an area (*e.g.* a city) including all the trajectories to be anonymized.
- $T_i = \{(t_1^i, x_1^i, y_1^i), \ldots, (t_n^i, x_n^i, y_n^i)\}$ represents a trajectory completed by peer P_i, where $(t_\ell^i, x_\ell^i, y_\ell^i)$ is a *point* consisting of a time-stamp t_ℓ^i and a location specified as latitude x_ℓ^i and longitude y_ℓ^i.

$MinGridSize$ and $MinTimeInterval$ should be large enough so that peers have no privacy concerns to publish generalized versions of their trajectories under those parameters. In our experiments, we have used $MinGridSize = 100$ meters and $MinTimeInterval = 3600$ seconds, but larger values could be chosen.

Protocol 1 (*Local k-anonymization of Trajectories Based on Tit-for-tat*)**.**

1. *P_i generates a $Grid$ of squares with square size $GridSize$ starting at coordinate (lat_{ini}, lon_{ini}) and ending at coordinate (lat_{end}, lon_{end}).*
2. *P_i checks whether a generalized trajectory T_g that matches her own T_i has been published by some other peer in the network. To do this, she calculates topicId as described in Algorithm 2 with arguments $Grid$ and $TimeInterval$, and she subscribes to the resulting topicId.*
3. *If topicId already exists in the network and was created by a peer P_j not blacklisted by P_i then:*
   (a) *P_i publishes in topic topicId an ack message and she waits for an ack reply message from P_j.*
   (b) *If P_i receives an ack reply message from P_j, P_i sends her trajectory T_i to P_j with one of the following options:*
      - *Either P_i sends to P_j her entire trajectory T_i by calling $Receipt$=Send-tit-for-tat$(P_i, T_i, P_j, true)$ (see Protocol 2);*
      - *Or P_i sends to P_j her trajectory T_i by separately sending the centroids of the locations in T_i following Algorithm 3.*
   (c) *Else (After a timeout, P_i has not received an ack reply message):*
      i. *$GridSize = GridSize + MinGridSize$. If the number of squares is less than 2 then: $TimeInterval = TimeInterval + MinTimeInterval$, and $GridSize = MinGridSize$.*
      ii. *Go back to Step 1.*

   Else (topicId generated by P_i does not exist in the network):
   (a) *P_i creates topicId and waits for ack messages from other peers.*
   (b) *If P_i receives at least $k-1$ ack messages then:*
      i. *P_i sends ack reply messages to the peers that got in touch and waits for their trajectories.*
      ii. *Once P_i has received at least $k-1$ trajectories from other peers, P_i microaggregates the closest $k-1$ trajectories with her own T_i by using Algorithm 5. P_i publishes the microaggregated trajectory on behalf of the $k-1$ contributing peers.*

   Else (After a timeout, P_i has not received $k-1$ ack messages):
      i. *$GridSize = GridSize + MinGridSize$. If the number of squares in $Grid$ is less than 2 then $TimeInterval = TimeInterval + MinTimeInterval$, and $GridSize = MinGridSize$.*
      ii. *Go back to Step 1.*

Protocol 2 (*Send-tit-for-tat$(P_i, T_i, P_j, first_hop)$*)**.**

1. *If $first\_hop$ then: $p_{forward} = 1$ else: $p_{forward} = p$, where $p \in [0, 1]$ is a parameter.*
2. *If Bernoulli$(p_{forward}) = 1$ then:*
   (a) *P_i randomly chooses a non-blacklisted peer P'.*
   (b) *P_i sends T_i to P'.*
   (c) *If P_i is blacklisted by P', then P' exits the protocol.*
   (d) *If $P' \neq P_j$ then:*
      i. *P' calls $Receipt$ = Send-tit-for-tat$(P', T_i, P_j, false)$.*
      ii. *If P' obtains $Receipt = Sign_{P_j}(T_i)$ then: P' returns $Receipt$ via the reverse path.*
      Else: P' blacklists P_j;

   Else:
      i. *P_j computes $Receipt = Sign_{P_j}(T_i)$.*

ii. P_j *returns* $Receipt$ *via the reverse path.*

(e) *If* P_i *obtains* $Receipt = Sign_{P_j}(T_i)$ *then:* P_i *returns* $Receipt$ *via the reverse path.*
Else: P_i *blacklists* P'.

Else:

(a) P_i *sends* T_i *to* P_j.
(b) *If* P_i *is blacklisted by* P_j *then* P_j *exits the protocol.*
(c) *If* P_i *obtains* $Receipt = Sign_{P_j}(T_i)$ *then:* P_i *returns* $Receipt$ *via reverse path.*
Else: P_i *blacklists* P_j.

The generation of a $Grid$ in Step 1 of Protocol 1, common to all peers in the system, is straightforward by starting from the global upper-left coordinates (lat_{ini}, lon_{ini}) and finishing at the lower-right coordinates (lat_{end}, lon_{end}) with a square size of $GridSize$.

In Step 2 of Protocol 1, P_i creates a *topicId* using Algorithm 2. This algorithm computes *topicId* by constructing a generalized trajectory T_g from P_i's trajectory T_i.

Algorithm 2: *topicId* CONSTRUCTION

input : trajectory: $T_i = \{(t_1^i, x_1^i, y_1^i), \ldots, (t_n^i, x_n^i, y_n^i)\}$
$Grid$
$TimeInterval$
output: the *topicId*

1 $T_i' = interpolation(T_i)$;
2 $T_g = \emptyset$;
3 $(q_x^a, q_y^a) = (-1, -1)$;
4 **for** $j \leftarrow 1$ **to** $|T_i'|$ **do**
5 $\quad c_j = (x_j^i, y_j^i) \in T_i'$;
6 $\quad (q_x, q_y)$ = square in $Grid$ where c_j falls;
7 $\quad$ **if** $(q_x, q_y) \neq (q_x^a, q_y^a)$ **then**
8 $\quad\quad T_g = T_g \cup (q_x, q_y)$;
9 $\quad$ **end**
10 $\quad (q_x^a, q_y^a) = (q_x, q_y)$;
11 **end**
12 $\bar{t}^i = \frac{t_1^i + t_n^i}{2}$;
13 $I_g = \text{round}(\frac{\bar{t}^i}{TimeInterval})$;
14 $topicId = T_g \cup I_g$;
15 **return** $topicId$

In Step 1 of Algorithm 2, the input trajectory T_i is interpolated so that the successive locations of the resulting trajectory T_i' are at most at distance $GridSize/2$. To this end, we first compute the number of new points that need to be added between each two consecutive original points in the trajectory to fulfill the condition on the maximum distance. Then, we calculate the new points by linear interpolation on the straight line between the two original points, as described in [31]. As a result, T_i' has locations in all the squares of the $Grid$ traversed by T_i. In Steps 4 to 11, a generalized trajectory T_g is created by replacing each location in T_i' by the (x_grid, y_grid) indices of the $Grid$ square containing the location (see Fig. 1). The condition of Step 7 avoids duplicating square indices in T_g in case more than one location falls into the same square of $Grid$. In Steps 12 and 13, time-stamps of locations in T_i are generalized into a fixed time interval I_g. Finally, *topicId* is defined as the string resulting from concatenating T_g and I_g.

Fig. 1 depicts an example trajectory T_i placed on a grid. Following Algorithm 2, its generalized trajectory T_g is (1, 2), (2, 2), (3, 1), (4, 1), (5, 1), (6, 1), (7, 1), (8, 2). Regarding time, assume the time stamps at the origin and the destination are t_1^i=1213082518 and t_n^i=1213083210, and the value of $TimeInterval$ is 1 h (3600 s); then, $\bar{t}^i = (t_1^i + t_n^i)/2$=1213082864 and thus $I_g = \bar{t}^i / TimeInterval$=336967. The resulting *topicId* is, therefore, (1, 2), (2, 2), (3, 1), (4, 1), (5, 1), (6, 1), (7, 1), (8, 2), 336967.

In Step 3b of Protocol 1, peers send their trajectories to be microaggregated to P_j by either (i) sending the entire trajectory via tit-for-tat to P_j, or (ii) sending the centroids of trajectory locations individually. The first case is straightforward and incurs minimum network overhead but, as discussed above, may result in disclosure if (parts) of the trajectory can be used to re-identify the sender. To avoid re-identification, the sender may opt to send trajectory points individually, each one through a separate SEND-TIT-FOR-TAT call. This option provides maximum privacy (because re-identification is completely prevented) at the cost of increased network overhead due to the extra forwarding messages required to send each individual point. Notice that sending an entire trajectory or just individual points are extreme scenarios; we may also consider intermediate cases where fragments of the trajectory, rather than individual points, are sent.

For the receiver, getting individual points rather than trajectories is not problematic because, as we detail below, she just needs $k-1$ points, each one from a different sending peer, for each of her own points. However, if the amounts of points of the trajectories of the $k-1$ sending peers differ from the amount of points of the receiver's trajectory, the receiver does not get $k-1$ points for each of her own points. To solve this issue, peers send trajectory centroids created with the same fixed criterion employed to create the *topicId* described above. The process is formalized in Algorithm 3. Steps 3 to 10 calculate the ordered set of different squares where the points in the trajectory fall. Then, in Steps 12 to 20, the centroid of each square is calculated and sent via tit-for-tat. To calculate the centroid of a square (Step 19), we use the $calculate\_centroid$ function of Algorithm 5, which returns a point that is the centroid of the set of points received as argument. Notice that sending centroids instead of exact points or complete trajectories already incurs some information loss. Thus, this strategy provides maximum privacy at the cost of both increased network overhead and some information loss. Both are quantified in the experimental section.

Algorithm 3: SENDING TRAJECTORY CENTROIDS

input : trajectory: $T_i = \{(t_1^i, x_1^i, y_1^i), \ldots, (t_n^i, x_n^i, y_n^i)\}$
P_i: the sending peer
P_j: the destination peer
$Grid$

1 $squares = \emptyset$;
2 $(q_x^a, q_y^a) = (-1, -1)$;
3 **for** $k \leftarrow 1$ **to** n **do**
4 $\quad c_k = (x_k^i, y_k^i) \in T_i$;
5 $\quad (q_x, q_y)$ = indices of the square in the $Grid$ where c_k falls;
6 $\quad$ **if** $(q_x, q_y) \neq (q_x^a, q_y^a)$ **then**
7 $\quad\quad squares = squares \cup (q_x, q_y)$;
8 $\quad$ **end**
9 $\quad (q_x^a, q_y^a) = (q_x, q_y)$;
10 **end**
11 $Q = \emptyset$;
12 **for** $k \leftarrow 1$ **to** $|squares|$ **do**
13 $\quad (q_x, q_y) = squares_k$;
14 $\quad points = \emptyset$;
15 $\quad$ **for** $l \leftarrow 1$ **to** n **do**
16 $\quad\quad$ **if** (t_l^i, x_l^i, y_l^i) *falls in square* (q_x, q_y) **then**
17 $\quad\quad\quad points = points \cup (t_l^i, x_l^i, y_l^i)$;
18 $\quad\quad$ **end**
19 $\quad\quad Q = Q \cup calculate\_centroid(points)$;
20 $\quad$ **end**
21 **end**
22 **for** $k \leftarrow 1$ **to** $|Q|$ **do**
23 $\quad$ SEND-TIT-FOR-TAT$(P_i, Q_k, P_j, true)$;
24 **end**

Finally, let us return to the end of Protocol 1. If, after a timeout, P_i has not received $k-1$ acknowledgment messages (*ack*), this means that there are not enough peers whose trajectories match the generalized

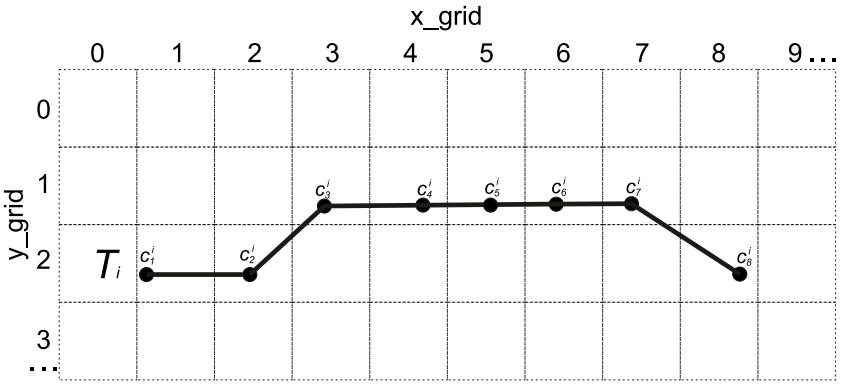

Fig. 1. Example trajectory within a grid. Each square of the grid has sides of length $GridSize$. The upper-left coordinate corresponds to (lat_{ini}, lon_{ini}) and the lower-right coordinate corresponds to (lat_{end}, lon_{end}). The ordered set of grid coordinates where each location falls is the generalization T_g of the trajectory T_i.

trajectory. In such case, P_i increases $GridSize$ to obtain a yet more generalized trajectory, for which there are more chances of finding matching trajectories. If increasing $GridSize$ is not enough, the time interval is also increased and $GridSize$ is reset. Note that enlarging the grid and/or the time interval will result in greater information loss due to microaggregation, because less similar trajectories will be matched and aggregated. This is why the protocol starts by trying to find enough matches for the least generalized trajectory that is considered privacy-preserving enough.

4.2. Measuring the distance between trajectories

Trajectory microaggregation relies on a distance measure to quantify the resemblance of the trajectories to be grouped and aggregated. This distance must consider both space and time. Most spatial distances, such as the Euclidean distance, can be extended into spatio-temporal distances by adding a time coordinate to spatial points and balancing the weight of the spatial and temporal dimensions [32]. Also, since each trajectory may have a different number of points, a method to sample, match and compare individual trajectory points is needed.

Dynamic time warping (DTW) [33] has become one of the most popular algorithms to measure the distance between trajectories [32]. The DTW algorithm searches through all locations in two trajectories for a pair of points at a minimum distance. If T_i and T_j are two trajectories with lengths n and m, respectively, the computational cost of DTW is $O(mn)$. Piece-wise dynamic warping (PDTW) [34] speeds up DTW by a parameter c, which depends on the data. The computational cost of PDTW is $O(NM)$, where $N = n/c$ and $M = m/c$. In [35], the authors propose the *spatio-temporal linear combine* (STLC) distance, which combines spatial and temporal distances between trajectories. STLC uses the Euclidean distance for the spatial dimension and the Manhattan distance for the temporal dimension; then, spatial and temporal similarities are linearly combined according to a parameter λ that weights both similarities. The computational cost of STLC is quadratic $O(mn)$.

We define a distance measure based on PDTW and STLC that tries to find the best match between pairs of trajectory points with a low computational cost. The distance calculation is described in Algorithm 4. Its computational cost is just $O(h)$, where h is the average number of points in the two trajectories, which makes this distance suitable for large data sets. To calculate the distance, the algorithm selects a list of h representative pairs of points, proportionally to the number of points in each trajectory. Only these points will be considered during the distance calculation. Fig. 2 shows an example of the pairs of points selected to compare trajectories T_a and T_b. Note that the points corresponding to the origin and the destination of a trajectory are always selected.

Once a pair of points, c_i^a and c_j^b, have been matched, Step 9 of Algorithm 4 calculates the distance between them. Similar to STLC, we define a distance that linearly combines spatial and temporal distances between pairs of points, as per Expression (1):

$$dist(c_i, c_j) = dist_{spa}((x_i^a, y_i^a), (x_j^b, y_j^b)) + \lambda \cdot (|t_i^a - t_j^b| \cdot V_{ab}), \quad (1)$$

where $dist_{spa}$ is the spatial distance between coordinates (x_i^a, y_i^a) and (x_j^b, y_j^b), and V_{ab} is the mean velocity of trajectories T^a and T^b. The temporal distance $|t_i^a - t_j^b|$ is multiplied by the mean velocity of trajectories V_{ab} to convert it to a spatial distance that can be added to $dist_{spa}$. However, converting the temporal distance to a spatial distance can lead to an excessive weight of the temporal distance, because this latter distance implicitly assumes that the subjects are constantly moving away from each other (whereas they may actually move in winding trajectories or even in circles). To mitigate this problem, we weight the temporal component with a parameter $\lambda = \frac{D}{V \cdot T}$, where D is the maximum distance between points in the data set, V is the mean velocity of the trajectories in the data set and T is the maximum time difference between points in the data set. Note that the denominator of λ is the maximum distance that would be attained by a subject constantly moving away from another subject at the mean speed V for the maximum time T, whereas the numerator D is the actual maximum distance that two subjects in the data set achieve.

Algorithm 4: Distance between trajectories

input : trajectories
$T_a = \{(t_1^a, x_1^a, y_1^a), \ldots, (t_n^a, x_n^a, y_n^a)\}$
$T_b = \{(t_1^b, x_1^b, y_1^b), \ldots, (t_m^b, x_m^b, y_m^b)\}$
output: distance between trajectories T_a and T_b

1 $h = \text{round}(\frac{n+m}{2})$;
2 $gap^a = \frac{n}{h}$; $gap^b = \frac{m}{h}$;
3 $index^a = 0$; $index^b = 0$;
4 $i = 0$; $j = 0$;
5 $distance = 0$;
6 **for** $k \leftarrow 1$ **to** h **do**
7 $\quad c_i^a = (t_i^a, x_i^a, y_i^a) \in T_a$;
8 $\quad c_j^b = (t_j^b, x_j^b, y_j^b) \in T_b$;
9 $\quad distance = distance + dist(c_i^a, c_j^b)^2$;
10 $\quad index^a = index^a + gap^a$;
11 $\quad index^b = index^b + gap^b$;
12 $\quad i = \text{round}(index^a)$;
13 $\quad j = \text{round}(index^b)$;
14 **end**
15 $distance = \sqrt{\frac{distance}{h}}$;
16 **return** $distance$

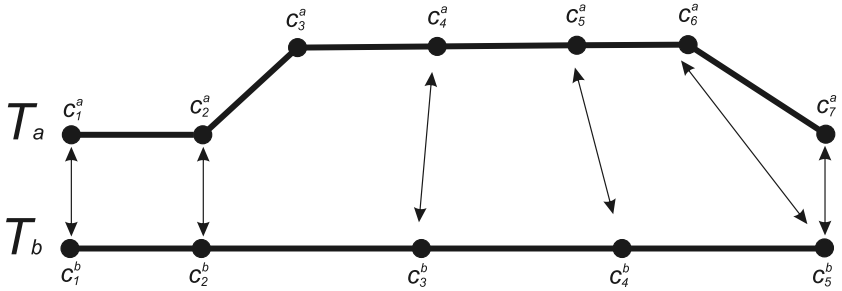

Fig. 2. Trajectory distance calculation. Arrows show the pairs of points selected to calculate the distance between trajectories T_a and T_b.

4.3. Aggregating trajectories

Trajectory aggregation consists in replacing a set of trajectories by a single representative, namely the centroid of the set. Since the replacement causes information loss, calculating accurate centroid trajectories is crucial to retain the utility of the anonymized data set as much as possible. Algorithm 5 formalizes trajectory aggregation.

Algorithm 5: Trajectory aggregation

input : a subset Q of trajectories $\in T$
output: the centroid trajectory Q_c of Q

1 $Q_c = \emptyset$;
2 $n = |Q|$;
3 $|Q_i|$ = number of points in trajectory $Q_i \in Q$;
4 $h = \text{round}(\frac{1}{n}\sum_{i=1}^{n}|Q_i|)$;
5 $gap[1 \dots n]$;
6 **for** $i \leftarrow 1$ **to** n **do**
7 $\quad gap[i] = \frac{|Q_i|}{h}$;
8 **end**
9 $index[1 \dots n]$; initialized to 0
10 **for** $k \leftarrow 1$ **to** h **do**
11 $\quad points[1 \dots n]$;
12 $\quad$ **for** $j \leftarrow 1$ **to** n **do**
13 $\quad\quad i = \text{round}(index[j])$;
14 $\quad\quad points[j] = (t_i, x_i, y_i) \in T_j$;
15 $\quad$ **end**
16 $\quad Q_c = Q_c \cup calculate\_centroid(points)$;
17 $\quad$ **for** $j \leftarrow 1$ **to** n **do**
18 $\quad\quad index[j] = index[j] + gap[j]$;
19 $\quad$ **end**
20 **end**
21 **return** Qc
22 **Function** *calculate_centroid(points: set)* : (t_c, x_c, y_c) **is**
23 $\quad n = |points|$;
24 $\quad t_c = 0$; $x_c = 0$; $y_c = 0$;
25 $\quad$ **for** $i \leftarrow 1$ **to** n **do**
26 $\quad\quad (t_i, x_i, y_i) = points[i]$;
27 $\quad\quad t_c = t_c + t_i$;
28 $\quad\quad x_c = x_c + x_i$;
29 $\quad\quad y_c = y_c + y_i$;
30 $\quad$ **end**
31 $\quad t_c = \frac{t_c}{n}$; $x_c = \frac{x_c}{n}$; $y_c = \frac{y_c}{n}$;
32 $\quad$ **return** (t_c, x_c, y_c)
33 **end**

Step 4 of Algorithm 5 calculates h as the mean number of points in the trajectories included in the input set Q. Analogously to what we did in the distance calculation of Algorithm 4, we select a sample h points from each trajectory in Q proportionally to its number of points. This yields h sets of $|Q|$ points each. The centroid (t^c, x^c, y^c) of each of the h sets is calculated as the component-wise mean of the $|Q|$ points in the set. The centroid trajectory Q_c is obtained as the concatenation of the h centroids, that is, $Q_c = \{(t_1^c, x_1^c, y_1^c), \dots, (t_h^c, x_h^c, y_h^c)\}$.

Fig. 3 shows an example of aggregation of two trajectories T_a and T_b. The arrows show the sample points selected from each trajectory. The resulting centroid trajectory Q_c is depicted with a dotted line.

4.4. Privacy analysis

After the proposed protocol suite is completed, the privacy of the moving subjects vs observers of the anonymized trajectories is guaranteed by k-anonymity: the trajectory of any subject is confused with at least $k-1$ other trajectories.

As to the privacy a moving subject vs peers who collaborate in the tit-for-tat trajectory exchange protocol, it rests on (i) the unlinkability of the senders and their trajectories, and (ii) the possibility of sending trajectory fragments (or individual points) rather than complete trajectories. Specifically, we can state the following propositions.

Proposition 1. *If a malicious peer participates in the tit-for-tat protocol (Protocol 2) and has no background knowledge about the originator of a certain trajectory she receives (that is, about who is the moving subject who followed the trajectory), she cannot unequivocally link the trajectory with its originator.*

Proof. If P' is a malicious intermediate peer who receives a trajectory T from another peer P in Protocol 2 and has no background knowledge about who followed trajectory T, P' does not know whether P is the originator of the trajectory or just a forwarder. On the other hand, if P_j is a malicious destination peer (who has published a *topicId* and receives trajectories for that topic), P_j knows that she will never receive trajectories directly from their respective originators, because the forwarding probability in the first hop of Protocol 2 is 1; thus, unless P_j has background knowledge about who followed the received trajectory, P_j cannot guess who it is. Hence, without background knowledge, no peer receiving a trajectory can unequivocally link the trajectory with its originator. □

However, if in Protocol 2 a malicious peer knows as background knowledge some points of an originator's trajectory and such points happen to be unique in the community of moving subjects — this can occur, because trajectories exchanged in Protocol 2 are not yet k-anonymized —, both re-identification and attribute disclosure are possible. On the one hand, the unique points re-identify the trajectory's originator. On the other hand, the remaining points in the trajectory, which may be confidential or sensitive, will be unequivocally attributed to the originator, thereby yielding attribute disclosure.

To prevent this, the sender can send trajectory points (centroids) individually, as described in Step 3b of Protocol 1, rather than the complete trajectory. In this case, we can state the following proposition.

Proposition 2. *If in Protocol 1 the originator sends the points of the trajectory separately, no malicious peer in Protocol 2 can unequivocally reconstruct the originator's trajectory. This holds even if the malicious peer knows some of the originator's points as background knowledge.*

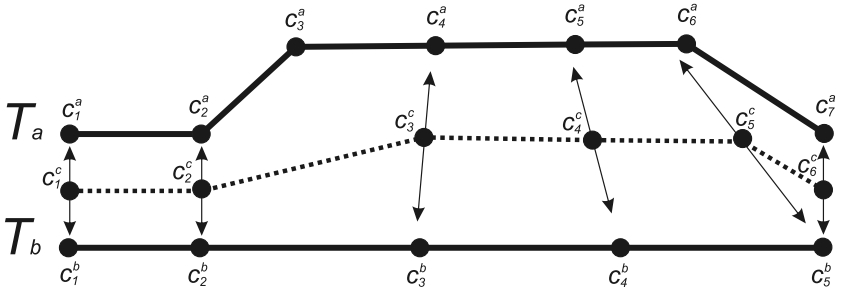

Fig. 3. Aggregation of trajectories. The arrows show the points selected to aggregate the trajectories T_a and T_b. The dotted line represents the centroid trajectory taken as output of the aggregation.

Proof. By definition, a point cannot be (quasi-)identifying and confidential at the same time, because quasi-identifying points are known by other peers but confidential points are known only to the originator who visited them. Hence, sending individual points via separate and anonymous tit-for-tat exchanges completely prevents attribute disclosure, because the malicious peer cannot associate any of the originator's confidential points with any of the originator's quasi-identifying points that the malicious peers knows as background knowledge. Thus, the malicious peer enjoys no advantage. □

5. Experimental results

We ran a simulation of our decentralized anonymization protocol on a set of real trajectories extracted from the cab mobility GPS data provided by the Exploratorium museum within the Cabspotting project.[1] The data set contains the trajectories of approximately 500 taxi cabs in the San Francisco Bay Area recorded during May 2008 [36]. The data capture the usual features of realistic trajectories (*i.e.*, short trajectories in areas with dense population). Each record contains the GPS coordinates and absolute times of all trajectory points. The experiments were run on an AMD Ryzen 7 3.59 GHz 8-core processor with 64 GB RAM. Our code and data are publicly available to facilitate reproducibility.[2]

5.1. Configuration of the experiments

We have used the mobility records corresponding to a specific weekday, namely Monday. We joined all trajectories recorded on all Mondays of the month, by respecting the hours, in order to obtain a large and dense data set. We considered only those trajectories corresponding to cabs that were occupied by a customer. This resulted in realistic trajectories with meaningful and precise origins, paths and destinations, rather than seemingly random routes of cabs wandering or waiting for customers. We omitted very short trajectories of fewer than 500 meters or with fewer than 4 locations. The resulting data set contains 43,628 trajectories, which we assigned to different virtual subjects whose privacy had to be protected. Fig. 4 (top) shows that the traces are concentrated during the working hours ([8h–18h]) and the leisure time ([19h–1h]). Likewise, Fig. 4 (bottom) shows that the number of GPS positions recorded for most trajectories is 20 or less.

We then evaluated Protocol 1 on the above-described data set according to the following metrics:

- Utility of the anonymized data, measured as the reciprocal of the information loss resulting from the microaggregation of trajectories. Information loss accounts for the differences between the original and the anonymized data sets, which we measured as the root mean square error (RMSE):

$$RMSE = \frac{1}{n}\sqrt{\sum_{i=1}^{n} dist(T_i, T_i')^2}, \tag{2}$$

  where n is the number of trajectories in the data set, T_i is a trajectory in the original data set, T_i' is the anonymized version of T_i and $dist$ is the distance between trajectories computed with Algorithm 4.
- Network overhead, measured as the number of messages sent to the network during the execution of the protocol.

The first metric allowed us to compare decentralized anonymization with its centralized counterpart, in which a single data controller gathers all trajectories and anonymizes them with the same microaggregation method.

We assumed rational behavior for all 43,628 peers in the system. As initial parameters to compare the scenarios, we took values of k in the range 2–100 (which covers the most usual k-anonymity values); a forwarding probability $p = 0.5$ (that is, same probability of sending the location to the recipient or to an intermediate peer, see Protocol 2); $MinGridSize$ initialized to 100 meters and $MinTimeInterval$ to 3600 s. The top-left map coordinate was taken to be $(lat_{ini}, lon_{ini}) = (37.810791, -122.515053)$ and the bottom-right coordinate $(lat_{end}, lon_{end}) = (37.601877, -122.343929)$, which included all trajectories in the data set (see the parameters in Protocol 1). For the sake of stability, for each test we took the average result of 5 runs.

As to the network overhead, we evaluated it under the two extreme scenarios defined in Protocol 1:

- *Sending trajectories*: Peers send their entire trajectories to the peer who performs the anonymization. This scenario offers the least privacy (because a complete trajectory may contain (quasi-)identifying and confidential locations, thereby incurring confidential attribute disclosure), but the best efficiency (because the number of messages sent over the network is minimized).
- *Sending centroids*: Peers separately send the centroids of points falling in each square of the grid to the peer who performs the anonymization. This scenario offers the best privacy (because anonymously sent individual centroids do not disclose the whereabouts of the sender) but the worst efficiency (because a message is needed for each trajectory point).

Note that, to balance the trade-off between privacy and efficiency/overhead, intermediate scenarios between the two above extreme cases could be considered in which trajectory fragments are sent.

5.2. Results

In Fig. 5, we show the RMSE information loss as a function of the microaggregation parameter k for the two aforementioned scenarios and the centralized microaggregation baseline.

As expected, RMSE grows with k, due to the fact that the higher k, the more trajectories are grouped and aggregated to become indistinguishable. Also, we observe that centralized microaggregation yields substantially less information loss. In the decentralized setting, a peer forms a cluster with the first $k-1$ trajectories she receives that match her published generalized trajectory. In contrast, if a single

[1] http://www.exploratorium.edu/id/cab.html.
[2] https://github.com/anonymProjects/decentralized_k-anom/.

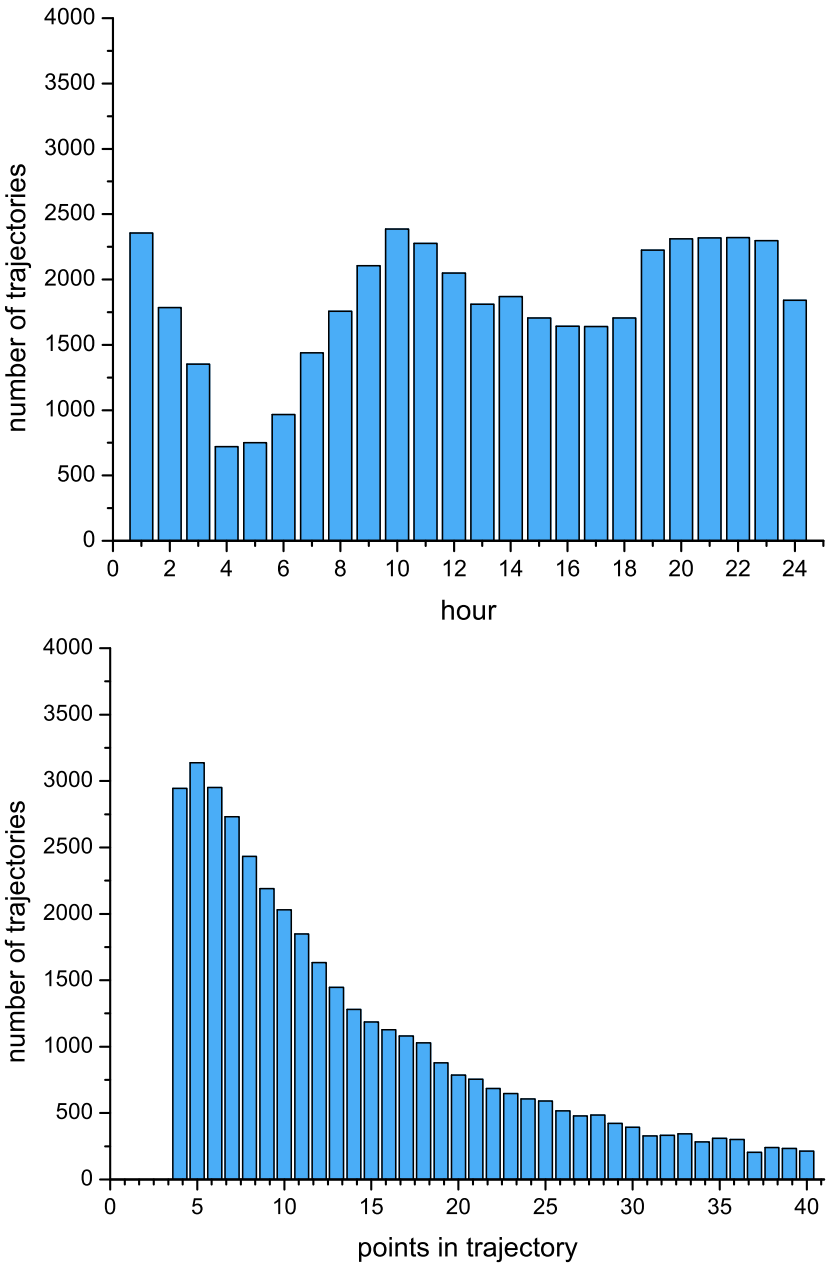

Fig. 4. Distribution of the mobility traces used in our experiments: by hours (top) and by number of points in the trajectory (bottom).

party (the data controller) receives all original trajectories, that party can microaggregate them into (near) maximally homogeneous clusters, thereby incurring less information loss.

If we compare the two decentralized scenarios, we see that separately sending centroids during the anonymization process results in a higher information loss than sending the entire trajectory (around 10% higher for all k values). The reason is that, to send centroids, they must first be computed, which already entails some information loss: all the trajectory locations falling inside each square in the trajectory must be aggregated into a single centroid.

The network overhead for the two decentralized scenarios is depicted in Fig. 6. The results follow a similar pattern as for information loss: the higher k, the greater the network overhead. This is due to the increasing difficulty of finding $k-1$ matching trajectories for a generalized trajectory when k increases: successive topic generalizations may be needed. If we compare the two scenarios, we can see that sending centroids rather than complete trajectories takes 40% more messages on average, because peers have to send one message per square of the generalized trajectory, rather than a single message per trajectory. In any case, the absolute number of messages per peer stays in the range 5–50, which is a reasonable number that can be easily accommodated in a decentralized P2P network, where the bandwidth load is balanced among peers.

In the previous simulations, we have assumed that all peers strictly follow the protocol. A more realistic situation is to consider some *bad* peers who ignore forwarding requests of a trajectory or a part of a trajectory during the tit-for-tat of Protocol 2. This behavior may occur due to peers maliciously deviating to break the protocol, or due to selfishness, if they find no incentives to contribute. In our protocol, these bad peers are easily identified because they are unable to return a correctly signed receipt in Protocol 2, which causes them to be blacklisted. However, the presence of bad peers generates some overhead in the system due to the need to re-execute incomplete exchanges.

We report below the results of our protocol when the percentage of bad peers is between 0 and 10%. The network overhead is shown in Fig. 7. The average number of messages sent by each peer grows with

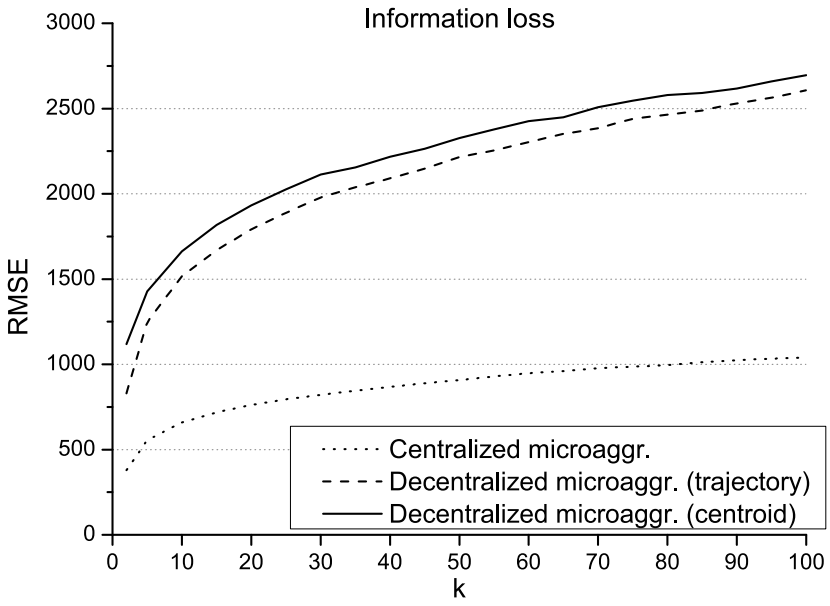

Fig. 5. Information loss for centralized and decentralized microaggregation ($p = 0.5$).

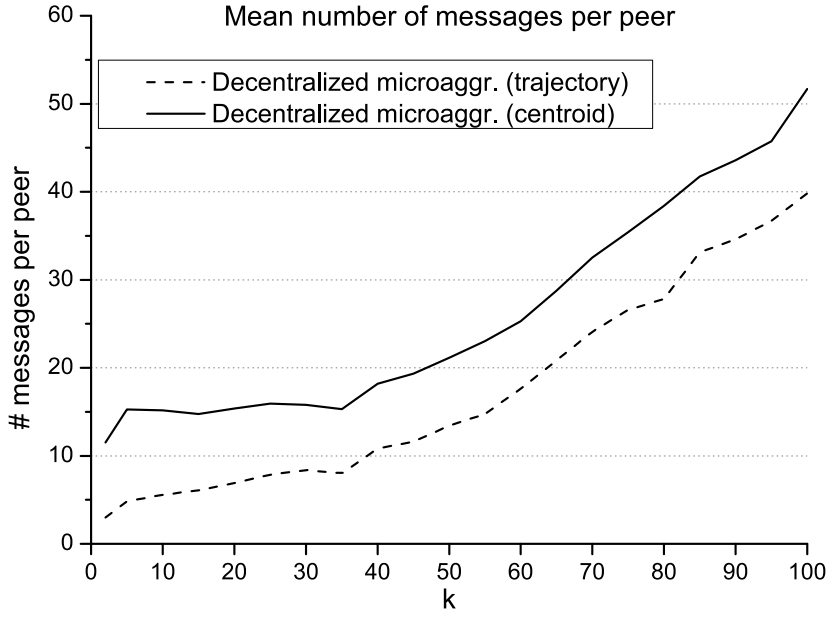

Fig. 6. Average number of messages per peer sent during the matching process for the two scenarios ($p = 0.5$).

the percentage of bad peers. As expected, this experiment shows that the presence of bad peers introduces a significant network overhead (on average 800 messages are required for each peer in the worst case). The network overhead increases in the presence of bad peers because the good peers have to re-send messages that have been ignored by bad peers and, on the other hand, bad peers have to re-send messages because they are blacklisted by a greater number of peers. However, given the small size of the messages sent, these figures are still manageable in a decentralized setting, especially when compared with file-sharing P2P networks. It can also be observed that, the larger k, the slower is the overhead growth with the percentage of bad peers. The reason is that, as k increases, fewer trajectory clusters are formed, and hence fewer messages are needed.

On the other hand, the fact that bad peers are identified and blacklisted can lead to a worse anonymization quality (more information loss) for those peers. To measure this, we separately quantified the information loss derived from the anonymization for the bad and good peers. Fig. 8 shows that bad peers incur more information loss than good peers. This loss of anonymization quality is due to the fact that, once a bad peer is identified and blacklisted, there is a higher probability that her requests are rejected. This forces the bad peer to try other, worse matches with more distant trajectories from other peers. In contrast, good peers always get their requests accepted and, due to the blacklisting of bad peers, they also increase further their chance of finding better matching trajectories. Yet, the RSME penalty for bad peers decreases as k increases: the reason is that for larger k, larger clusters are formed, which have a worse RSME due to their size. Thus, for larger clusters the information loss is also large for good peers.

Putting together the results of Figs. 7 and 8, we conclude that, among the three values of k we considered, the intermediate value $k = 10$ achieved the best trade-off between high overhead penalty and high information loss penalty for bad peers. Note that the objective is that the overall penalty is high enough for rational peers to be motivated to

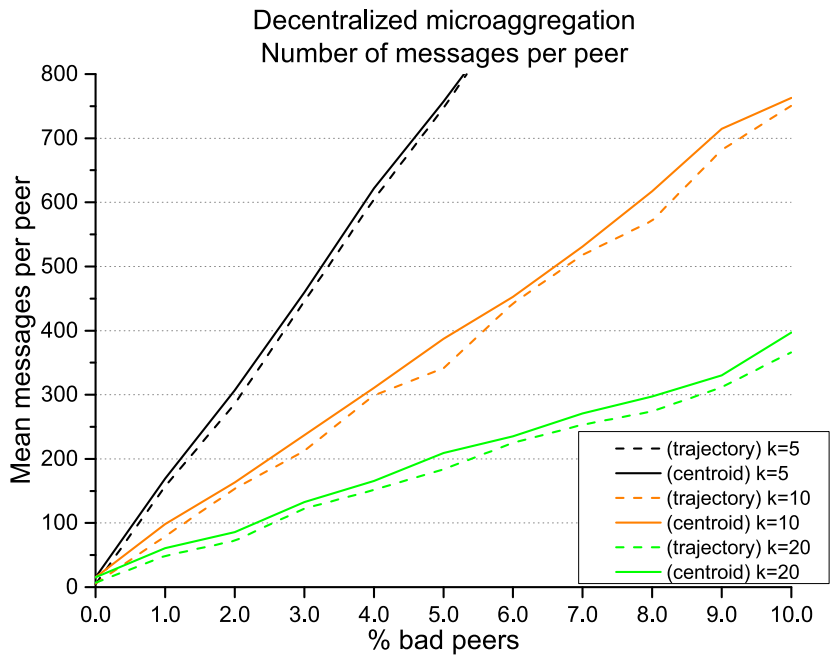

Fig. 7. Average number of messages per peer for the two decentralized scenarios with 0% to 10% bad peers ($p = 0.5$ and $k = 5, 10, 20$).

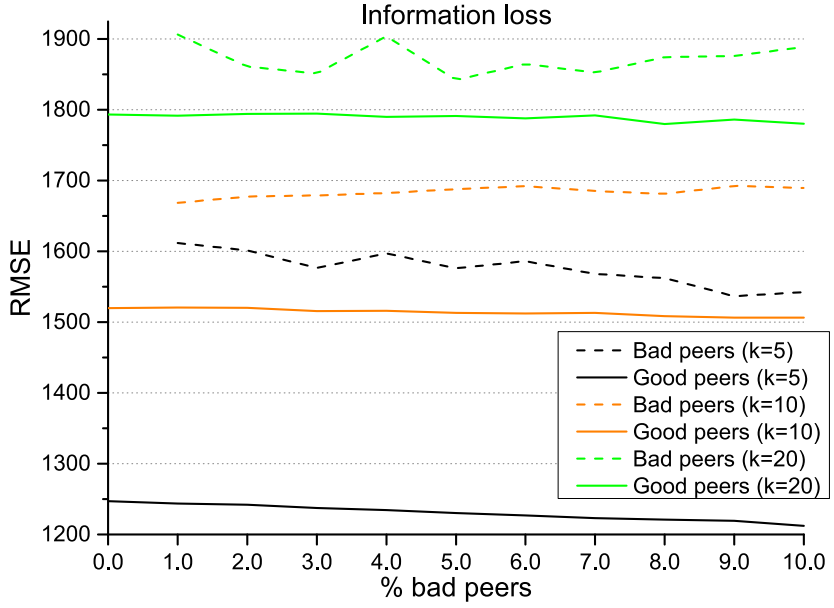

Fig. 8. Information loss for bad and good peers as a function of the proportion of bad peers ($p = 0.5$ and $k = 5, 10, 20$).

follow the protocol. This ensures the sustainability of the protocol even in a fully decentralized setting.

6. Conclusions and future research

State-of-the-art methods for privacy-preserving trajectory data release are all based on centralized k-anonymity, which causes privacy and security concerns w.r.t. the data controller enforcing the anonymization algorithm. In this paper, we have tackled this issue by means of a decentralized P2P network that avoids relying on a central data controller. Our method offers formal privacy guarantees according to the k-anonymity privacy model. The resulting sets of anonymized trajectories effectively prevent unequivocal re-identification while keeping information loss reasonably low (and hence utility reasonably high). Our decentralized anonymization protocol has been designed so that anonymity and attribute non-disclosure are also ensured w.r.t. the other peers in the network. Adherence to the protocol rules by peers is incentivized by a punishment mechanism based on black lists.

Directions for future research could include designing ways to automatize the fragmentation of trajectories prior to sending them to other peers. The idea would be to fragment only (quasi-)identifying sets of points, instead of sending each point individually. In this way, we can optimize the trade-off between privacy and efficiency. We will

also investigate how to combine our decentralized k-anonymity method with DP.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments and disclaimer

We acknowledge support from the European Commission (projects H2020-871042 "SoBigData++" and H2020-101006879 "MobiDataLab") and the Government of Catalonia, Spain (ICREA Acadèmia Prizes to J. Domingo-Ferrer and D. Sánchez).

References

[1] X. Kong, M. Li, K. Ma, K. Tian, M. Wang, Z. Ning, F. Xia, Big trajectory data: a survey of applications and services, IEEE Access 6 (2018) 58295–58306.
[2] N. Oliver, B. Lepri, H. Sterly, R. Lambiotte, S. Deletaille, M. De Nadai, E. Letouzé, A. Ali Salah, R. Benjamins, C. Cattuto, V. Colizza, N. de Cordes, S.P. Fraiberger, T. Koebe, S. Lehmann, J. Murillo, A. Pentland, P.N. Pham, F. Pivetta, J. Saramäki, S.V. Scarpino, M. Tizzoni, S. Verhulst, P. Vinck, Mobile phone data for informing public health actions across the COVID-19 pandemic life cycle, Sci. Adv. 6 (23) (2020).
[3] S. Latif, M. Usman, S. Manzoor, W. Iqbal, J. Qadir, G. Tyson, I. Castro, A. Razi, M.N. Kamel Boulos, A. Weller, J. Crowcroft, Leveraging data science to combat COVID-19: a comprehensive review, IEEE Trans. Artif. Intel. 1 (1) (2020) 85–103.
[4] General data protection regulation, 2016, Regulation (EU) 2016/679 of the European Parliament and of the Council, April 27, 2016.
[5] M. Fiore, P. Katsikouli, E. Zavou, M. Cunche, F. Fessant, D. Le Hello, U. Matchi Aivodji, B. Olivier, T. Quertier, R. Stanica, Privacy in trajectory micro-data publishing: a survey, Trans. Data Priv. 13 (2020) 91–149.
[6] J. Domingo-Ferrer, R. Trujillo-Rasua, Microaggregation and permutation-based anonymization of movement data, Inform. Sci. 208 (2012) 55–80.
[7] T. Murakami, A. Kanemura, H. Hino, Group sparsity tensor factorization for re-identification of open mobility traces, IEEE Trans. Inf. Forensics Secur. 12 (3) (2017) 689–704.
[8] L. Rossi, J. Walker, M. Musolesi, Spatio-temporal techniques for user identification by means of GPS mobility data, EPJ Data Sci. 4 (1) (2015).
[9] Y. Song, D. Dahlmeier, S. Bressan, Not so unique in the crowd: A simple and effective algorithm for anonymizing location data, in: International Workshop on Privacy Preserving IR, PIR 2014, 2014, pp. 19–24.
[10] J. Salas, D. Megías, V. Torra, Swapmob: Swapping trajectories for mobility anonymization, in: Josep Domingo-Ferrer, Francisco Montes (Eds.), Privacy in Statistical Databases, Springer, 2018, pp. 331–346.
[11] D. Sánchez, S. Martínez, J. Domingo-Ferrer, Comment on 'unique in the shopping mall: On the reidentifiability of credit card metadata', Science 351 (2016) 1274.
[12] P. Samarati, L. Sweeney, Protecting Privacy when Disclosing Information: K-Anonymity and Its Enforcement Through Generalization and Suppression, Technical Report, SRI International, 1998.
[13] C.Y. Chow, M.F. Mokbel, Enabling private continuous queries for revealed user locations, in: International Symposium on Spatial and Temporal Databases, 2007, pp. 258–275.
[14] X. Pan, X. Meng, J. Xu, Mobimix: Protecting location privacy with mix-zones over road networks, in: Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, 2009, pp. 256–265.
[15] M. Gramaglia, M. Fiore, Hiding mobile traffic fingerprints with glove, in: Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies, 2015, pp. 1–13.
[16] A. Monreale, G.L. Adrienko, N.V. Adrienko, F. Giannotti, F.D. Pedreschi, S. Rinzivillo, S. Wrobel, Movement data anonymity through generalization, Trans. Data Priv. 3 (2) (2010) 91–121.
[17] M.E. Nergiz, M. Atzori, Y. Saygin, B. Guc, Towards trajectory anonymization: a generalization-based approach, Trans. Data Priv. 2 (1) (2009) 47–75.
[18] C. Dwork, F. McSherry, K. Nissim, A. Smith, Calibrating noise to sensitivity in private data analysis, in: Proceedings of the Third Conference on Theory of Cryptography, in: TCC, vol. 06, Springer, 2006, pp. 265–284.
[19] H. Ngo, J. Kim, Location privacy via differential private perturbation of cloaking area, in: 2015 IEEE 28th Computer Security Foundations Symposium, 2015, pp. 63–74.
[20] Y. Xiao, L. Xiong, Protecting locations with differential privacy under temporal correlations, in: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, 2015, pp. 1298–1309.
[21] S. Brunet, S. Canard, S. Gambs, B. Olivier, Novel differentially private mechanisms for graphs, IEEE Trans. Inf. Forensics Secur. (2016) 745, IACR Cryptology EPrint Archive, 2016.
[22] M. Hay, A. Machanavajjhala, G. Miklau, Y. Chen, D. Zhang, Principled evaluation of differentially private algorithms using dpbench, in: Proceedings of the 2016 International Conference on Management of Data, in: SIGMOD, vol. 16, 2016, pp. 139–154.
[23] Y. Cao, M. Yoshikawa, Differentially private real-time data release over infinite trajectory streams, in: 2015 16th IEEE International Conference on Mobile Data Management, Vol. 2, 2015, pp. 68–73.
[24] J. Domingo-Ferrer, D. Sánchez, A. Blanco-Justicia, The limits of differential privacy (and its misuse in data release and machine learning), Commun. ACM 64 (7) (2021) 34–36.
[25] M.E. Andrés, N.E. Bordenabe, K. Chatzikokolakis, C. Palamidessi, Geo-indistinguishability: differential privacy for location-based systems, ACM CCS 2013 (2021) 901–914.
[26] R. Dewri, Local differential perturbations: location privacy under approximate knowledge attackers, IEEE Trans. Mob. Comput. 12 (12) (2012) 2360–2372.
[27] W. Cheng, R. Wen, H. Huang, W. Miao, C. Wang, OPTDP: Towards optimal personalized trajectory differential privacy for trajectory data publishing, Neurocomputing 472 (2022) 201–211.
[28] J. Domingo-Ferrer, V. Torra, Ordinal, continuous and heterogeneous k-anonymity through microaggregation, Data Min. Knowl. Discov. 11 (2) (2005) 195–212.
[29] I. Stoica, R. Morris, D. Karger, M. Frans Kaashoek, H. Balakrishnan, Chord: A scalable peer-to-peer lookup service for internet applications, in: Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, in: SIGCOMM, vol. 01, 2001, pp. 149–160.
[30] M. Castro, P. Druschel, Y.C. Hu, A. Rowstron, Exploiting Network Proximity in Peer-to-Peer Overlay Networks, Technical report MSR-TR2002-82, 2002.
[31] E. Meijering, A chronology of interpolation: from ancient astronomy to modern signal and image processing, Proc. IEEE 90 (3) (2002) 319–342.
[32] H. Su, S. Liu, B. Zheng, X. Zhou, K. Zheng, A survey of trajectory distance measures and performance evaluation, VLDB J. 29 (1) (2020) 3–32.
[33] A. Gasmelseed, N. Mahmood, Study of hand preferences on signature for right-handed and left-handed peoples, Int. J. Adv. Eng. Technol. 1 (5) (1963) 41–46.
[34] F.K. Chan, A.W. Fu, C. Yu, Haar wavelets for efficient similarity search of time-series: with and without time warping, in: IEEE Trans. Knowl. Data Eng., 15 (3) (2003) 686–705.
[35] S. Shang, L. Chen, Z. Wei, C.S. Jensen, K. Zheng, P. Kalnis, Trajectory similarity join in spatial networks, Proc. VLDB Endow. 10 (11) (2017) 1178–1189.
[36] M. Piorkowski, N. Sarafijanovic-Djukic, M. Grossglauser, Dataset epfl/mobility (v. 2009-02-24), 2022, CRAWDAD Repository, Feb. 24, 2009. https://crawdad.org/epfl/mobility/20090224. (Accessed 10 April 2022).