JCSE Journal of Computing Science and Engineering

Regular Paper

Journal of Computing Science and Engineering,
Vol. 5, No. 3, September 2011, pp. 210-222

Uncertainty for Privacy and 2-Dimensional Range Query Distortion

Spyros Sioutas*, Emmanouil Magkos, Ioannis Karydis
Department of Informatics, Ionian University, Corfu, Greece
sioutas@ionio.gr, emagos@ionio.gr, karydis@ionio.gr

Vassilios S. Verykios
School of Science and Technology, Hellenic Open University, Patras, Greece
verykios@eap.gr

Abstract

In this work, we study the problem of privacy-preservation data publishing in moving objects databases. In particular, the trajectory of a mobile user in a plane is no longer a polyline in a two-dimensional space, instead it is a two-dimensional surface of fixed width $2A_{min}$, where A_{min} defines the semi-diameter of the minimum spatial circular extent that must replace the real location of the mobile user on the *XY*-plane, in the anonymized (kNN) request. The desired anonymity is not achieved and the entire system becomes vulnerable to attackers, since a malicious attacker can observe that during the time, many of the neighbors' ids change, except for a small number of users. Thus, we reinforce the privacy model by clustering the mobile users according to their motion patterns in (u, θ) plane, where u and θ define the velocity measure and the motion direction (angle) respectively. In this case, the anonymized (kNN) request looks up neighbors, who belong to the same cluster with the mobile requester in (u, θ) space: Thus, we know that the trajectory of the k-anonymous mobile user is within this surface, but we do not know exactly where. We transform the surface's boundary poly-lines to dual points and we focus on the information distortion introduced by this space translation. We develop a set of efficient spatiotemporal access methods and we experimentally measure the impact of information distortion by comparing the performance results of the same spatiotemporal range queries executed on the original database and on the anonymized one.

Category: Smart and intelligent computing

Keywords: Uncertainty; Privacy; Anonymity; Moving objects databases; Voronoi clustering

I. INTRODUCTION

Technological advances in sensors and wireless communications have enabled offering of high location accuracy in location tracking at a low cost [1-3]. Such accuracy gave rise to a series of location-based applications that exploit positional data to offer high-end location-based services (LBS) to their subscribers [4]. Conversely, mobile users require the protection of their context information (e.g., location and/or identity information) against privacy adversaries (e.g., Big-brother type threats, users profiling, unsolicited advertising) [5-8]. The privacy issue is amplified by the requirement in modern telematics and location-aware applications for real-time, continuous location updates and accurate location information (e.g., traffic monitoring, asset tracking, location-based advertising, location-based payments, routing directions) [9-11].

We consider a population of mobile users who are supported by some telecommunication infrastructure, owned by a telecom operator. Every user periodically transmits through his/her mobile device a location update to some traffic monitoring sys-

Open Access http://dx.doi.org/10.5626/JCSE.2011.5.3.210 http://jcse.kiise.org

Received 01 February 2011, **Accepted** 20 March 2011
*Corresponding Author

Preliminary version of this work was presented in the International Conference on Privacy in Statistical Databases (PSD2010), and appeared in Vol.6344, pp.85-96 of Springer LNCS series.

 pISSN: 1976-4677 eISSN: 2093-8020

tem residing in a untrusted Internet-connected LBS provider. A trusted server in the telecom operator plays the role of a proxy that mediates between the user and the LBS provider. Our setting involves an LBS provider that is requested to efficiently answer Range Queries (RQs) of mobile users moving on the plane. For example historical queries of the form:

SELECT Count (Mobile-User-Ids)
FROM Moving_Object_Database (MOD)
WHERE spatial-area=MBR('Acropolis') and *t_past* < *time* < *t_now*

or prediction queries of the form:

SELECT Count (Mobile-User-Ids)
FROM Moving_Object_Database (MOD)
WHERE spatial-area=MBR('Acropolis') and *t_now* < *time* < *t_future*

that provide traffic monitoring services to mobile users. This type of query focuses on the problem of indexing mobile users in two dimensions and efficiently answering range queries over the users' locations. This problem is motivated by a set of real-life applications, such as intelligent transportation systems, cellular communications, and meteorology monitoring.

Our privacy requirements, include:

- *Location privacy*: The protocol does not reveal the (exact) user's location information to the LBS provider.
- *Tracking protection* (*Unlinkability*): The LBS provider is not able to link two or more successive user positions.

In our privacy threat model, we assume an attacker has access to the k-anonymized (cloaked) location queries and to the algorithms used by the trusted server to support user privacy. We assume that the proxy server of the telecom operator is trusted not to collude with the LBS provider or other internal/external non-authorized entities to expose the atomic location updates of system users.

Traditionally, the transformation of the exact user location to a spatiotemporal area is achieved by the *k*-anonymity principle for relational data [12, 13]. This requires that each record in a given dataset is indistinguishable from at least $k-1$ other records with respect to a certain set of identifying variables, collectively known as the "quasi-identifier". The *k*-anonymity principle requires that the spatiotemporal area that is generated by the trusted server from the exact location of the mobile user is such that the identity of the user cannot be disclosed with a probability that is larger than $1/k$, among $k-1$ other users. The *k*-anonymity primitive is essential to protect the privacy of the users, starting from the point of request for a RQ service and continuing for as long as the requested service withstands completion.

A. High Level Description of Our Framework

As part of our framework, we deliver the type of *k*-trajectory privacy [14] that identifies $k-1$ users that are close to the requester at the time of request and thus could have issued the request. This includes a minimum circular spatial area A_{min} around the requester, in which the participants of the anonymity set should be searched, so the user is adequately hidden. The proposed privacy framework relies on a user privacy profile that stores the necessary information related to his/her privacy requirements. This includes 1) the preferred value of k (in *k*-anonymity) for each requested RQ service, 2) the minimum circular spatial area A_{min}, around the requester, in which the participants of the anonymity set should be searched, so that the user is adequately hidden. This threshold defines the minimum extent of the spatial area that must replace the real location of the user, in the anonymized request.

The proposed framework deals with the event of failure in the provision of *k*-anonymity, in the case where the number of participants inside this minimum spatial area is less than k - 1. In this case, the trusted server postpones the servicing of the user request for a small period. Then, if the anonymization process fails again, the requester is protected by blocking the servicing of the request.

The proposed privacy framework can be reinforced by clustering the mobile users according to their motion patterns on the (u, θ) plane, where u and θ define the velocity measure and the motion direction (angle) respectively. In this case, the anonymized (kNN) request lookups neighbors, who belong to the same cluster with the mobile requester in (u, θ) space.

Two basic approaches are used when trying to handle RQ services; those that deal with discrete and those that deal with continuous movements. In a discrete environment the problem of dealing with a set of moving objects can be considered to be equivalent to a sequence of database snapshots of the object positions/extents taken at time instants $t_1 < t_2 < \ldots$, with each time instant denoting the moment where a change took place. From this viewpoint, the indexing problems in such environments can be dealt with by suitably extending indexing techniques from the area of temporal [15] or/and spatial databases [16]; in [17] it is elegantly exposed how these indexing techniques can be generalized to handle efficiently queries in a discrete spatiotemporal environment. A plethora of efficient data structures exists that consider continuous movements [18-22]. The common thrust behind these indexing structures lies in the idea of abstracting each object's position as a continuous function $f(t)$ of time and updating the database whenever the function parameters change; accordingly, an object is modeled as a pair consisting of its extent at a reference time (design parameter) and its motion vector. One categorization of the aforementioned structures is according to the family of the underlying access method used. In particular, there are approaches based either on R-trees or on Quad-trees. Conversely, these structures can be also partitioned into 1) those that are based on geometric duality and represent the stored objects in the dual space [19, 21] and 2) those that leave the original representation intact by indexing data in their native *n*-d space [20, 22, 23]. The *geometric duality transformation* is a tool heavily used in the Computational Geometry literature that maps hyper-planes in R^n to points and vice-versa. We implement a framework based on the proposed privacy model that uses the Spatial extensions of MySql 5.x to offer privacy in RQ services.

In this work, we study the problem of privacy-preservation data publishing in moving objects databases. In particular, the trajectory of a mobile user in the plane is no longer a polyline in a two-dimensional space, instead it is a two-dimensional surface: we know that the trajectory of the mobile user is within

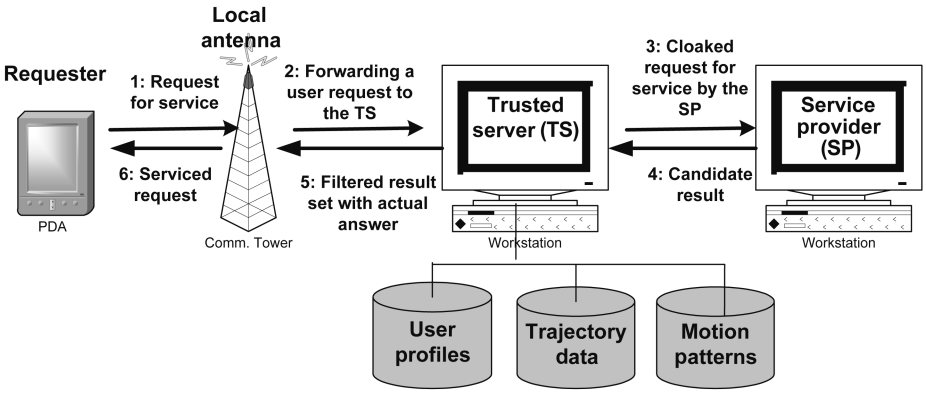


Fig. 1. The system architecture.

this surface, but we do not know exactly where. We transform the surface's boundary polylines to dual points [19, 20] and we focus on the information distortion introduced by this space translation. We develop a set of efficient spatiotemporal access methods and, we experimentally measure the impact of information distortion by comparing the performance results of the same spatiotemporal range queries executed on the original database and on the anonymized one.

The remainder of this paper is organized as follows. Section II lays out the system model for our proposed methodology and gives a formal description of the problem. Section III discusses preliminary notions and results used throughout the paper. Section IV presents the method of transforming the trajectory polylines to two-dimensional surfaces. Section V presents: 1) the duality transformation of the surface's boundary poly-lines, 2) a simple observation, based on which we enforce the privacy model of our method, and 3) the information distortion introduced by this space translation. Section VI presents an extended experimental evaluation. Section VII concludes the paper.

II. SYSTEM MODEL AND PROBLEM DESCRIPTION

This section lays out the system model in which our proposed methodology is applied. The considered system framework interconnects a set of registered LBS providers to a set of subscribed users, to enable the offering of RQSs in a privacy-aware manner. Fig. 1 depicts our system framework that has two main components: the trusted server, which is assigned the task of trajectory privacy preservation, and the service providers, which offer the actual RQSs. As part of its functionality, the trusted server maintains a database that stores the privacy profiles of the users, their history of movement, as well as knowledge of motion patterns.

A system user can be any individual who is equipped with a mobile device and is registered to the trusted server. Upon his/her registration, the user is assigned a unique profile in the system and his/her movement is subsequently monitored by the trusted server. The tracking of user movement is enabled by the transmission of periodic location updates from the mobile device of the user to the trusted server. A privacy profile for a user consists of a set of tuples (sid, K, A_{min}) that define his/her privacy requirements that involve 1) the value of k for each provided service sid, where k indicates that the user wants to remain K-anonymous when using this service, and 2) the minimum area of generalization A_{min}, where A_{min} indicates the minimum spatial area to which the anonymized request should point.

Fig. 1 depicts the regular scenario for the privacy-aware provision of RQSs. Whenever a user requests an RQS (step 1), the trusted server receives the original request (step 2), examines the point of request and determines the appropriate anonymity strategy that needs to be enforced. Next, the original request is transformed to a 'secure' equivalent that preserves the k-anonymity of the requester and is forwarded to the appropriate service provider to be serviced (step 3). After the request has been serviced, the result set, computed by the service provider, is returned to the trusted server (step 4) where it is filtered to contain only the actual answer that is subsequently forwarded to the requester (steps 5 and 6). This concludes the privacy-aware provision of the service.

We consider a database that records the position of moving objects in two dimensions on a finite terrain. We assume that objects move with a constant velocity vector starting from a specific location at a specific time instant. Thus, we can calculate the future object position, if its motion characteristics remain the same. Velocities are bounded by $[u_{min}, u_{max}]$. Objects update their motion information, when their speed or direction changes. The system is dynamic, *i.e.,* objects may be deleted or new objects may be inserted. Let $P_z(t_0) = [x_0, y_0]$ be the initial position at time t_0 of object z. If object z starts moving at time $t > t_0$, its position will be $P_z(t) = [x(t), y(t)] = [x_0 + u_x(t - t_0), y_0 +$

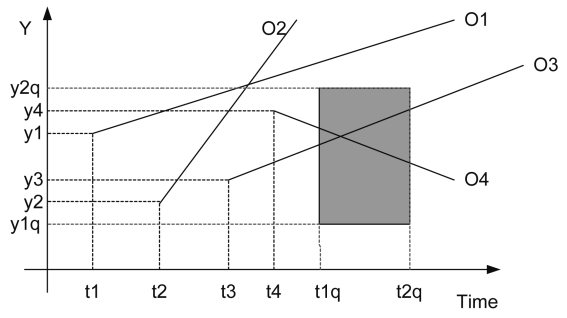


Fig. 2. Trajectories and query in (t, y) plane.

$u_y(t - t_0)$], where $U = (u_x, u_y)$ is its velocity vector. For example, in Fig. 2 the lines depict the objects trajectories on the (t, y) plane. We would like to answer queries of the form:

"Report the objects located inside the rectangle $[x_{1_q}, x_{2_q}] \times [y_{1_q}, y_{2_q}]$ at the time instants between t_{1_q} and t_{2_q} (where $t_{now} \leq t_{1_q} \leq t_{2_q}$), given the current motion information of all objects."

III. PRELIMINARIES

In the following, we briefly describe the most basic methods for indexing mobile objects, as well as the best current clustering algorithms that will be used throughout the paper and specifically for clustering mobile objects with similar motion-pattern.

A. Literature Survey: The Most Basic Methods for Indexing Mobile Objects

In the sequel, let N denote the input size (number of stored objects), B the block size, K the output size and thus $n = N/B$ and $k = K/B$ are the size of the input and output in blocks, respectively.

In [19] a set of indexing techniques was presented, which used the geometric duality transformation at the cost of increasing by one the dimensionality. Hence, for the one-dimensional case they reduced the indexing problem to the two-dimensional simplex range searching problem and they employed external memory partition trees to solve their indexing problem in $O(n)$ space, $O(n^{1/2} + k)$ I/Os query time and $O(\log n)$ I/Os update time.

Partition trees, with a guaranteed worst case performance, are generally considered impractical, since they entail large hidden factors. Thus, in [19] two more structures were presented, one based on k-d-trees and a more complex one based on B^+-trees; both structures used linear space and usually work well. Moreover, they extended their results in the two-dimensional case for two distinct versions of the problem; first, the objects were allowed to move on a network of one-dimensional routes, and, second, the objects were allowed to move arbitrary. The first version reduced to a number of one-dimensional sub-problems that used the previously described structures, whereas the second is equivalent (through geometric duality) to simplex range queries in $\Re^3$, which can be solved in $O(n^{2/3} + k)$ I/Os with the use of external memory partition trees.

In [24] the above results were further refined. A new version of partition tree was introduced to handle the indexing problem in the plane in $O(n)$ space, $O(n^{1/2} + k)$ query time, and $O(\log_B n)$ expected amortized update time; the results could apply in higher-dimensional spaces as well, degrading only the update time (it became $O(\log_B^2 n)$ I/Os). If it is assumed that the queries arrive in chronological order, then the query time can be further reduced to $O(\log_B^2 n / \log_B \log_B n)$ I/Os; this is achieved by employing the kinetic data structures framework at external range trees. Moreover, they developed an indexing scheme with small query time for near-future queries and increased time for distant in the future queries under the bound of $O(n^{1/2+\epsilon} + k)$ I/Os by combining multiversion kinetic data structures with partition trees. Finally, an indexing technique was described to handle $O(\delta)$-approximate queries; the query time was $(n^{1/2+\epsilon}/\delta)$, the expected update time $O(\log_B^2 n/\delta)$ and the space $O(n/\delta)$ disk blocks.

The TPR tree [25] in essence is an R*-tree generalization to store and access linearly moving objects. The leaves of the structure store pairs with the position of the moving point and the moving point ID, whereas internal nodes store pointers to sub-trees with associated rectangles that minimally bound all moving points or other rectangles in the sub-tree. The difference to the classical R*-tree lies in the fact that the bounding rectangles are time-parameterized (their coordinates are functions of time). It is considered that a time-parameterized rectangle bounds all enclosed points or rectangles at all times later than the current time. The algorithms for search and update operations in the TPR tree are straightforward generalizations of the respective algorithms in the R*-tree. Moreover, the various kinds of spatiotemporal queries can be handled uniformly in 1-, 2-, and 3-dimensional spaces.

The TPR-tree constituted the base structure for further developments in the area [26]. An extension to the TPR-tree was proposed in [22], the so-called *TPR*-tree*. The main improvement lies in the update operations, where it is shown that local optimization criteria (at each tree node) may seriously degrade the performance of the structure and more particularly in the use of update rules that are based on global optimization criteria. They proposed a novel probabilistic cost model to validate the performance of a spatiotemporal index and analyzed the optimal performance for any data-partition index with this model.

The STRIPES index [21] is based on the application of the duality transformation and employs disjoint regular space partitions (disk based quadtrees [16]); the authors claim, through the use of a series of implementations, that STRIPES outperforms TPR*-trees for both update and query operations.

Finally, in [27] a new access method (LBTs) was presented for indexing mobile objects that move on the plane to efficiently answer range queries about their future location. It has been proved that its update performance is the most efficient in all cases. The superiority of LBTs method regarding query performance has been shown as far as the query rectangle length remains at realistic levels (greatly outperforming other methods). If the query rectangle length becomes extremely large in relation to the whole terrain, then STRIPES is better than any other solution, however, only to a small margin in comparison to LBT's method.

B. Literature Survey: The Most Basic Clustering Algorithms

Clustering is the method of grouping the data into sets, so data points within the set should have high similarity, while those within different sets are dissimilar [28, 29].

1) K-Means Algorithm: K-means clustering [30] is a simple and widely used clustering algorithm. The K-Means algorithm is based on the squared error minimization method. We discuss the K-Means algorithm as follows:

The time complexity of K-Means algorithm is $O(nkt)$, where n is the number of data objects, k the number of clusters and t the number of iterations.

The obvious shortcomings of the basic K-Means clustering are that the number of clusters needs to be determined in

Algorithm 1. K-Means

1: Randomly choose k data points from the whole data set as initial cluster centers;
2: Calculate Euclidean distance of each data point from each cluster center and assign the data points to its nearest cluster center;
3: Calculate the new cluster center, so the squared error distance of each cluster should be minimized;
4: Repeat step 2 and 3 until the cluster centers do not change;
5: Stop the process;

advance and its computational cost with respect to the number of observations, clusters, and iterations. Although, K-Means is efficient in the sense that only the distance between a point and the k cluster centers –not all other points– needs to be (re)computed in each iteration. Typically, the number of observations n is much larger than k.

Many approaches have tried to alleviate both these problems. Variations of K-Means clustering that are supposed to cope without prior knowledge of the number of cluster centers have been presented [31, 32]. While the proposed solutions to K-Means clustering certainly improve its practical behavior, they do not overcome the fundamental problems associated with the algorithm.

In [33] density-based algorithms have been proposed as a better alternative to identifying the correct number of clusters in the data. These algorithms make use of Voronoi diagrams (as in [34] and [35]) and are statistically significantly more accurate than K-Means clustering.

2) Voronoi Diagram: The Voronoi diagram is one of the most important and useful techniques in the field of computational geometry. Given a set D of n data points $o_1, o_2, o_3, \ldots o_n$ in a plane, the Voronoi diagram (V or (D)) is a subdivision of the plane into Voronoi cells ($V(o_i)$ for o_i) (Fig. 3). The Voronoi cells are the set of points u that are closer to o_i than any other point in D. *i.e.,* $V(o_i) = \{u | d(o_i, u) \leq d(o_j, u) \forall j \neq i\}$, where d is the Euclidian distance. The Voronoi diagram divides the plane into n convex polygon regions (for each o_i), the vertices (P of Fig. 3) of the diagram are the Voronoi vertices, and the borders between two adjacent Voronoi cells are termed Voronoi edges (E of Fig. 3). Note that each Voronoi vertex is the center of a circle touching three or more data points lying in its adjacent Voronoi cells (Fig. 3).

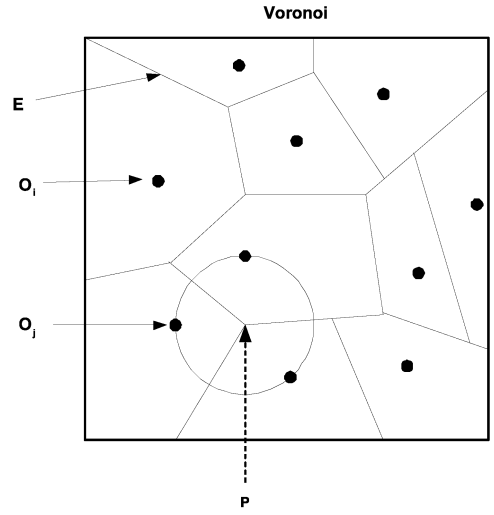


Fig. 3. Voronoi diagram.

3) Voronoi Clustering: The algorithm presented in [33], begins by constructing the Voronoi diagram for S, where $S = s_1, \ldots, s_n \subseteq \Re^d$ is a given data set of n points that is going to be clustered. The computational complexity of Voronoi diagram construction in the general case is $O(n \log n)$ [36], but requires only linear time in some restricted cases [37]. The algorithm is given as an input parameter a threshold value max indicating the maximum volume allowed to a cell that still can be combined into an evolving cluster.

Each point makes up its own cell in the Voronoi diagram for the data. The volumes (areas in case of the plane) of the cells are computed (or approximated) next. This approximation is not very accurate in small dimensions, but it improves, as the number of dimensions grows. Each cell is associated with a class label. The algorithm assigns increasing integer values as class labels. The point with the smallest Voronoi cell is handled first.

Algorithm 2. Voronoi clustering (set S, real max)

1: Construct the Voronoi diagram for the sample $S = s_1, \ldots, s_n$;
2: Approximate the Voronoi cell volumes and order the points accordingly. Without loss of generality, let the obtained order be $s_1, \ldots, s_n$;
3: For i = 1 to n do
4: If the volume of the cell R_i associated with s_i is at most *max* then
5: (a) For j = 1 to i - 1 do
6: If cluster C_j and cell R_i are adjacent then Merge them together;
7: If R_i has no class number yet, then it assumes that of C_j, else both assume the minimum class label among those of C_j and R_i;
8: (b) If cell R_i has no class number, then assign a new one to it;
9: else assign R_i to the closest neighboring cluster;

The cell boundaries in the Voronoi diagram of a point set are by definition equally distant from the relevant points. Thus, the Voronoi diagram naturally gives rise to a kind of maximum margin separation of the data. The algorithm combines existing cells, instead of creating artificial center points as, e.g., K-Means does. Hence, the maximal margins are maintained also in the clustered data. Maximal margin clustering seems a natural choice, since there are no reasons to place the cluster border closer to one or other cluster.

One can view Voronoi clustering as a change of paradigm from the K-Means, whose Euclidean distance minimization can be made autonomous with respect to the number of clusters only by optimizing a parameter that is too hard to learn. Kao et al. [34] proposed a similar pruning technique based on Voronoi

diagrams to reduce the number of expected distance calculations in case the mobile objects draw uncertain locations.

IV. TRAJECTORY POLY-LINES AND TWO-DIMENSIONAL SURFACES

For every mobile user, we calculate a circular range query whose center is its current 2-D location and radius a given value R defined by a minimum circular spatial area A_{min}. If this circular spatial area includes at least $k - 1$ other neighbors, then the mobile user is adequately hidden. Otherwise, if the number of participants inside this minimum spatial area is less than k - 1, the trusted server postpones the servicing of the user request for a small period. Next, if the anonymization process fails again, the requester is protected by blocking the servicing of the request. Thus, consecutive circular spatial areas construct a 2-D buffer defined by its upper and lower boundary poly-lines y' and y'' respectively that anonymize the original trajectory y of mobile user A (Fig. 4). We can create each mobile user as 2-D point using the Spatial extensions of MySql 5.x, as follows:

```
CREATE TABLE Points (
name VARCHAR(20) PRIMARY KEY,
location Point NOT NULL,
description VARCHAR(200),
SPATIAL INDEX(location)
);
```

To obtain points in a circular area as a counted result ordered by distance from the center of the selection area, we write:

```
SELECT COUNT(name), AsText(location), SQRT(POW
(ABS(X(location) - X(@center)), 2) + POW(ABS(Y(location)
- Y(@center)), 2)) AS distance
FROM Points
WHERE Intersects(location, GeomFromText(@bbox))
AND SQRT(POW(ABS(X(location) - X(@center)), 2) + POW
(ABS(Y(location) - Y(@center)), 2)) ≤ @R
ORDER BY distance;
```

If the result returned is less than k-1, the trusted server postpones the servicing of the user request for a small period.

Thus, the RQ service depicted in Fig. 2, was transformed to the Privacy-Aware RQ service depicted in Fig. 5:

If the entire buffer lies inside the query area (the buffers of mobile users O_3 or O_4 in Fig. 5), meaning that both upper and lower boundary poly-lines lie in the query rectangle, then the same holds for the original trajectory. If the entire buffer lies outside the query area (the buffer of mobile user O_2 in Fig. 5), meaning that both upper and lower boundary poly-lines lie outside the query rectangle, then the same holds for the original trajectory. In the worst-case, we face the distortion effect, where one of the two boundary poly-lines lies in the query rectangle (the buffer of mobile user O_1 in Fig. 5). In the later case, TS (Trusted Server) has to check what happens to the original trajectory.

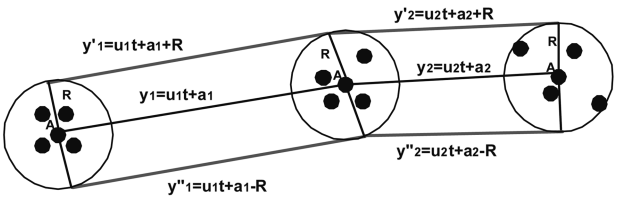


Fig. 4. 2-D buffer and boundary trajectories y' and y'' of mobile user A.

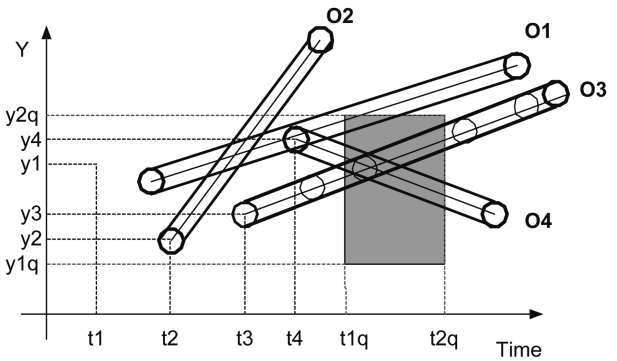


Fig. 5. Boundary trajectories and query in (t, y) plane.

V. DUALITY TRANSFORMATION OF BOUNDARY-TRAJECTORIES AND INFORMATION DISTORTION

The duality transform, in general, maps a hyper-plane h from R^n to a point in R^n and vice-versa. In this subsection, we briefly describe how we can address the problem at hand in more intuitively, using the duality transform in the 1-d case.

A. Hough-X Transform

One duality transform for mapping the line with equation $y(t) = ut + a$ to a point in R^2 is using the dual plane, where one axis represents the slope u of an objects trajectory (i.e., velocity), whereas the other axis represents its intercept a. Thus, we get the dual point (u, a) (this is the so called *Hough-X transform* [19, 20]). Accordingly, the 1-d query $[(t_{1_q}, t_{2_q}), (y_{1_q}, y_{2_q})]$ becomes a polygon in the dual space.

Using a linear constraint query [38], the query in the dual Hough-X plane (Fig. 6) is expressed as:

If $u > 0$, then $Q_{Hough\text{-}X} = A_1 \cap A_2 \cap A_3 \cap A_4$, where A_i is defined

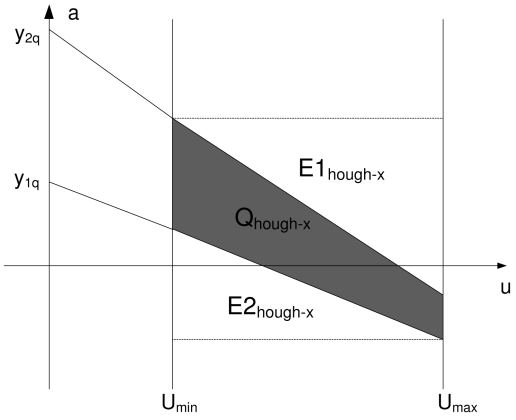


Fig. 6. Query in the Hough-X dual plane.

as follows:

$A_1 = u \geq u_{min}$,
$A_2 = u \leq u_{max}$,
$A_3 = a \geq y_{1q} - t_{2q} u$,
$A_4 = a \leq y_{2q} - t_{1q} u$.

Inequalities of the A_1 and A_2 areas are obvious. The inequalities for A_3 and A_4 can be derived as follows: $\forall t \in [t_{1q}, t_{2q}] \Rightarrow y_{1q} \leq y \leq y_{2q} \Rightarrow y_{1q} - t_{2q} u \leq y_{1q} - tu \leq a \leq y_{2q} - tu \leq y_{2q} - t_{1q} u$, since $t_{1q} \leq t \leq t_{2q}$.

If $u < 0$, then $Q_{Hough\text{-}X} = B_1 \cap B_2 \cap B_3 \cap B_4$, where B_i is defined as follows:

$B_1 = u \leq - u_{min}$,
$B_2 = u \geq - u_{max}$,
$B_3 = a \geq y_{1q} - t_{1q} u$,
$B_4 = a \leq y_{2q} - t_{2q} u$.

Inequalities of the B_1 and B_2 areas are obvious. For B_3 and B_4, we are working in the same way as in the case of A_3 and A_4.

$\forall t \in [t_{1q}, t_{2q}] \Rightarrow y_{1q} \leq y \leq y_{2q} \Rightarrow y_{1q} - t_{1q} u \leq y_{1q} - tu \leq a \leq y_{2q} - tu \leq y_{2q} - t_{2q} u$, since $0 \leq t_{1q} \leq t \leq t_{2q}$ and $u < 0$.

In Fig. 6 the line $a = y_{1q} - t_{1q} u$ for $u = u_{max}$ becomes $a = y_{1q} - t_{1q} u_{max}$ and the line $a = y_{2q} - t_{2q} u$ for $u = u_{min}$ becomes $a = y_{2q} - t_{2q} u_{min}$. Thus, for the Service Provider (SP), the initial query $[(t_{1q}, t_{2q}), (y_{1q}, y_{2q})]$ in the (t, y) plane is transformed to the rectangular query $[(u_{min}, u_{max}), (y_{1q} - t_{1q} u_{max}, y_{2q} - t_{2q} u_{min})]$ in the (u, a) plane.

Similarly, for the upper ($y'(t) = ut + a + R$) and lower ($y''(t) = ut + a - R$) boundary trajectories, SP gets the dual points $(u, a + R)$ and $(u, a - R)$ as well, as the final (transformed) rectangular queries become $[(u_{min}, u_{max}), (y_{1q} - R - t_{1q} u_{max}, y_{2q} - R - t_{2q} u_{min})]$ and $[(u_{min}, u_{max}), (y_{1q} + R - t_{1q} u_{max}, y_{2q} + R - t_{2q} u_{min})]$ respectively in the (u, a) plane.

B. Hough-Y Transform

By rewriting the equation $y = ut + a$ as $t = \frac{1}{u} y - \frac{a}{u}$, we can arrive at a different dual representation (the so called *Hough-Y transform* in [19, 20]. The point in the dual plane has coordinates (b, n), where $b = -\frac{a}{u}$ and $n = \frac{1}{u}$. Coordinate b is the point where the line intersects the line $y = 0$ in the original space. Horizontal lines cannot be represented using this transform. Similarly, the Hough-X transform cannot represent vertical lines. Nevertheless, both transforms are valid, since in our setting lines have a minimum and maximum slope (velocity is bounded by $[u_{min}, u_{max}]$).

The query in the dual Hough-Y plane (Fig. 7) is expressed as follows. If $u > 0$, then $Q_{Hough\text{-}Y} = C_1 \cap C_2 \cap C_3 \cap C_4$, where

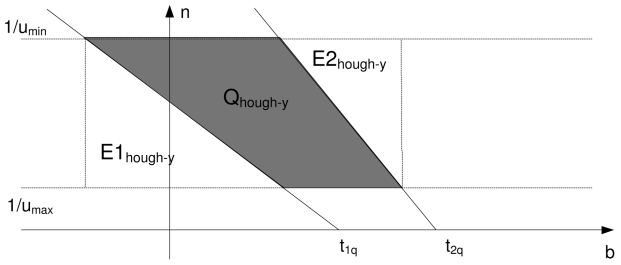


Fig. 7. Query on the Hough-Y dual plane.

$C_1 = n = \frac{1}{u} \geq \frac{1}{u_{max}}$,

$C_2 = n = \frac{1}{u} \leq \frac{1}{u_{min}}$,

$C_3 = n \geq -\frac{1}{y} b + \frac{t_{1q}}{y}$,

$C_4 = n \leq \frac{1}{y} b + \frac{t_{2q}}{y}$.

Inequalities of the C_1 and C_2 areas are obvious. The inequalities for C_3 and C_4 can be derived as follows: $\forall t \in [t_{1q}, t_{2q}] \Rightarrow n = -\frac{1}{y} b + \frac{t}{y} \leq -\frac{1}{y} b + \frac{t_{1q}}{y}$ and $n = -\frac{1}{y} b + \frac{t}{y} \leq -\frac{1}{y} b + \frac{t_{2q}}{y}$. Hence, the two lines in Fig. 7 have negative slope and for $b = 0$ intersect the axis b in t_{1q} and t_{2q}, respectively. The intersection of the four regions C_1, C_2, C_3, and C_4 form the shaded polygon query of Fig. 7.

If $u < 0$, then $Q_{Hough\text{-}Y} = D_1 \cap D_2 \cap D_3 \cap D_4$, where

$D_1 = n = \frac{1}{u} \leq -\frac{1}{u_{max}}$,

$D_2 = n = \frac{1}{u} \geq -\frac{1}{u_{min}}$,

$D_3 = n \geq -\frac{1}{y} b + \frac{t_{1q}}{y}$,

$D_4 = n \leq -\frac{1}{y} b + \frac{t_{2q}}{y}$.

The line $n = -\frac{1}{y} b + \frac{t_{1q}}{y}$ for $n = \frac{1}{u_{min}}$ implies that

$$b = t_{1q} - \frac{y}{u_{min}}. \tag{1}$$

In the same way, the line $n = -\frac{1}{y} b + \frac{t_{2q}}{y}$ for $n = \frac{1}{u_{max}}$ implies that

$$b = t_{2q} - \frac{y}{u_{max}}. \tag{2}$$

However, according to the initial query and equation (1), we have

$$y_{1q} \leq y \leq y_{2q} \Leftrightarrow t_{1q} - \frac{y_{2q}}{u_{min}} \leq b \leq t_{1q} - \frac{y_{1q}}{u_{min}}. \tag{3}$$

Analogously, according to the initial query and equation (2), we have

$$y_{1q} \leq y \leq y_{2q} \Leftrightarrow t_{2q} - \frac{y_{2q}}{u_{max}} \leq b \leq t_{2q} - \frac{y_{1q}}{u_{max}}. \tag{4}$$

According to (3) and (4) the initial query $[(t_{1q}, t_{2q}), (y_{1q}, y_{2q})]$ in the (t, y) plane can be transformed to the following rectangular query in the (b, n) plane: $[(t_{1q} - \frac{y_{2q}}{u_{min}}, t_{2q} - \frac{y_{1q}}{u_{max}}), (\frac{1}{u_{max}}, \frac{1}{u_{min}})]$. Similarly, for the upper ($y'(t) = ut + a + R$) and lower ($y''(t) = ut + a - R$) boundary trajectories, SP gets the transformed rectangular queries $[(t_{1q} - \frac{y_{2q} - R}{u_{min}}, t_{2q} - \frac{y_{1q} - R}{u_{max}}), (\frac{1}{u_{max}}, \frac{1}{u_{min}})]$ and $[(t_{1q} - \frac{y_{2q} + R}{u_{min}},$

$t_{2q} - \frac{y_{1q}+R}{u_{max}}$), ($\frac{1}{u_{max}}$, $\frac{1}{u_{min}}$)] respectively in the (b, n) plane.

Algorithm 3. SP-Index-Building

1: BEGIN_PSEUDOCODE
2: Decompose the 2-d motion into two 1-d motions on the (t, x) and (t, y) planes;
3: For each projection, build the corresponding index structure;
4: END_PSEUDOCODE

Algorithm 4. SP-Mobile-User-Partitioning

1: BEGIN_PSEUDOCODE
2: Users with small velocity are stored using the Hough-X dual transform;
3: The remainder are stored using the Hough-Y dual transform;
4: Motion information about the other projection is also included;
5: END_PSEUDOCODE

Algorithm 5. SP-Privacy-Aware-RQ Query

1: BEGIN_PSEUDOCODE
2: SP decomposes the query into two 1-d queries, for the (t, x) and (t, y) projection;
3: For each projection, SP gets the dual-simplex query;
4: For each projection, SP calculates a specific criterion c (for details see [19, 20]) and chooses the one (say p) that minimizes it;
5: For all dual-points of H_xS or H_yS sets, SP searches in projection p the simplex query of the Hough-X or the MBR of the simplex query of the Hough-Y partition. In the latter case, it performs a refinement or filtering step "on the fly", using the entire motion information;
6: if the dual-points of both upper and lower boundary trajectories ((u_i, $a_i + R$), (u_i, $a_i - R$) or b'_i, b''_i) lie inside the dual-simplex spatial area, then SP detects that the same holds for the dual-point ((u_i, a_i) or b_i) of the original trajectory;
7: else if the dual-points of both upper and lower boundary trajectories lie outside the dual-simplex spatial area, then SP detects that the same holds for the dual-point of the original trajectory;
8: else host=send_RQ(Trusted-Server, Algorithm 6);
9: END_PSEUDOCODE

C. The Proposed Algorithm for Privacy-Aware Indexing

Let $S = \{y_1, y_2, \ldots, y_n\}$ be the initial set of original trajectory equations, and $S' = \{y'_1, y''_1, y'_2, y''_2 \ldots, y'_n, y''_n\}$ the set of boundary trajectory equations defined by the buffer.

Then, let $H_xS = \{(u_1, a_1 + R), (u_1, a_1 - R), \ldots, (u_n, a_i + R), (u_n, a_i - R)\}$ and $H_yS = \{b'_1, b''_1, \ldots, b'_n, b''_n\}$ be the set of dual Hough-X and Hough-Y transforms respectively.

Algorithm 3 runs in SP and depicts the procedure to build the index. Algorithm 4 also runs in SP and presents the procedure to Partition the Mobile Users according to their velocity. Algorithm 5 is running in SP and finally calls Algorithm 6, which is running in Trusted Server (TS) to perform the final filtering step. In particular, Algorithms 5 and 6 outline the privacy-aware algorithm to answer the exact 2-d.

Algorithm 6. TS-Privacy-Aware-RQ Query

1: BEGIN_PSEUDOCODE
2: TS, with knowledge of the value R, searches the simplex query of the Hough-X or Hough-Y partition for the dual-point ((u_i, a_i) or b_i) of the original trajectory;
3: END_PSEUDOCODE

RQ query:

Algorithm 7. SP-Privacy-Aware Indexing of $Q_{Hough\text{-}Y}$ partition with Lazy B-tree

1: BEGIN_PSEUDOCODE
2: Decompose the query into two 1-d queries, for the (t, x) and (t, y) projection;
3: For each projection, get the dual-simplex query;
4: Search the MBR of the simplex query of the Hough-Y partition and perform a filtering step "on the fly", using the entire motion information;
5: Let $B \subset H_yS$ be the answer set of dual parameters that satisfy the query above;
6: *Result* = 0;
7: For all elements of B do
8: Begin_for
9: if ($b'_i \in B$ AND $b''_i \in B$) then
Result = *Result* $\cup$ (*idofuser*$_i$, *neighbors*(*idofuser*$_i$, $k - 1$);
10: return Result;
11: else if ($b'_i \notin B$ AND $b''_i \notin B$) then return Result;
12: else host=send_RQ(Trusted-Server, Algorithm 8);
13: End_for
14: END_PSEUDOCODE

Algorithm 8. TS-Privacy-Aware Indexing of $Q_{Hough\text{-}Y}$ partition with Lazy B-tree

1: BEGIN_PSEUDOCODE
2: if $b_i \in MBR$ of Hough-Y simplex partition then
3: begin
4: *Result* = *Result* $\cup$ (*idofuser*$_i$, *neighbors(idofuser*$_i$, $k - 1$); return Result;
5: else return Result;
6: end
7: END_PSEUDOCODE

In [19, 20], $Q_{Hough\text{-}X}$ is computed by querying a 2-d partition

tree, whereas $Q_{Hough-Y}$ is computed by querying a B^+ - tree that indexes the b parameters. Here, we consider the case where the users are moving with non-small velocities u, meaning that the velocity value distribution is skewed (Zipf) towards umin in some range $[u_{min}, u_{max}]$ and consequently the $Q_{Hough-Y}$ transformation is used (denote that u_{min} is a positive lower threshold). Moreover, our method will incorporate the Lazy B-tree [39] indexing scheme (Source code of Lazy B-tree index is free available at the following URL: http://www.ionio.gr/sioutas/New-Software.htm), since the latter can handle update queries in optimal (constant) number of block-transfers (I/Os). Thus, we get Algorithm 7 and 8. Algorithm 7 is running in SP machine and it calls Algorithm 8, which runs in TS machine, because the latter has the appropriate knowledge of the original trajectory parameter b_i to execute the final filtering step.

Algorithm 9. SP-Robust-Privacy-Aware Indexing

1: BEGIN_PSEUDOCODE
2: Decompose the query into two 1-d queries, for the (t, x) and (t, y) projection;
3: For each projection, get the dual-simplex query;
4: Search the MBR of the simplex query of the Hough-Y partition and perform a filtering step "on the fly", using the entire motion information;
5: Let $B \subset H_yS$ be the answer set of dual parameters that satisfy the query above;
6: *Result* = 0;
7: For all elements of B do
8: Begin_for
9: if ($b_i' \in B$ AND $b_i'' \in B$) then
10: Voronoi-Clustering of (u_i, θ_i) points, where $1 \le i \le n$;
11: *Result* = *Result* $\cup$ (*idofuser*$_i$, *neighbors*(*idofuser*$_i$, $k-1$);
12: return Result;
13: else if ($b_i' \notin B$ AND $b_i'' \notin B$) then return Result;
14: else host=send_RQ(Trusted-Server, Algorithm 10);
15: End_for
16: END_PSEUDOCODE

Algorithm 10. TS-Robust-Privacy-Aware Indexing

1: BEGIN_PSEUDOCODE
2: if $b_i \in$ *MBR* of Hough-Y simplex partition then
3: begin
4: Voronoi-Clustering of (u_i, θ_i) points, where $1 \le i \le n$;
5: *Result* = *Result* $\cup$ (*idofuser*$_i$, *neighbors*(*idofuser*$_i$, $k-1$); return Result;
6: else return Result;
7: end
8: END_PSEUDOCODE

D. Reinforcing the Privacy Model of Our Method Using Voronoi-Clustering of Mobile Users with a Similar Motion-Pattern

In statements 9 and 4 of algorithm 7 and 8 respectively, we call the routine *neighbors*(*idofuser*$_i$, $k-1$) routine, to find the $k-1$ nearest neighbors that will anonymize the original mobile user i. Let a malicious attacker who observes that during the time, many of the neighbors' ids change, except for a small number of users. In this case, the desired anonymity is not achieved and the entire system becomes vulnerable to attackers. Thus and in order to reinforce the entire privacy model, we initially call the *Voronoi − Clustering* routine to cluster the mobile users according to their motion pattern (Fig. 8) and then we call the *neighbors*(*idofuser*$_i$, $k-1$) routine that chooses neighbors who belong to the same cluster as user i. In this case, the neighbors' ids do not change with high frequency and the entire privacy model becomes robust.

We define as the motion-pattern of mobile user i at time instant t_i the point (u_i, θ_i) that represents both velocity value u_i and motion direction θ_i. Note here that we cannot avoid the 2-D clustering method mentioned before, by clustering of 1-D elements, like the parameters b_i or the projections of velocity vector ($u_{ix} = u_i cos\theta$) at X- or ($u_{ix} = u_i sin\theta$) Y-axis. Although these parameters include information about motion direction, these are not at all sufficient. Just think of the case where mobile

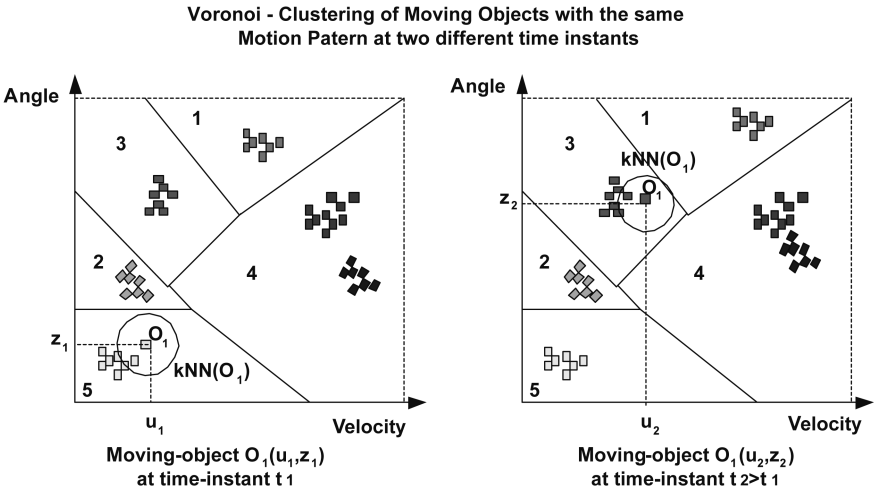


Fig. 8. Voronoi clustering and neighbours of the same cluster.

objects are moving in opposite directions, but also belong to the same cluster, since f.e. the cosine of opposite angles is the same. Thus, we get Algorithm 9 and 10. Similarly, Algorithm 9 is running in SP machine as well as Algorithm 10 executes the final filtering step in TS machine.

1) Distortion and Competitive Ratio: Let K be the number of b_i parameters associated with boundary trajectories of buffers that intersect with the query rectangle. Then, algorithms 9 and 10 require totally $T(n) = O(Cost(LazyB\_tree) + K)$ I/Os or block transfers. Moreover, and according to notations presented in [40], let us say that D is the initial Database that stores the N original trajectories and D' the Privacy-Aware Database that stores the $2N$ boundary-trajectories. Let us also say, $Q(D)$ and $Q(D')$ are the Query Results obtained consuming $T(D)$ and $T(D')$ block-transfers (I/Os) in D and D' database schemes, respectively. We define the *Distortion ratio to be the Distortion* $\_Ratio = \frac{|Q(D) - Q(D')|}{max(Q(D), Q(D'))}$ and the *Competitive ratio* to be the $Competitive\_Ratio = \frac{|T(D) - T(D')|}{max(T(D), T(D'))}$. In all cases, $T(D') > \ldots > T(D)$, thus it is very important to determine, how competitive to the optimal one ($T(D)$) is the privacy-aware method that answers the query in D'. An experimental evaluation of *Competitive_Ratio* vs. K is also presented in the following section, since the distortion effect in D' absolutely depends on parameter K.

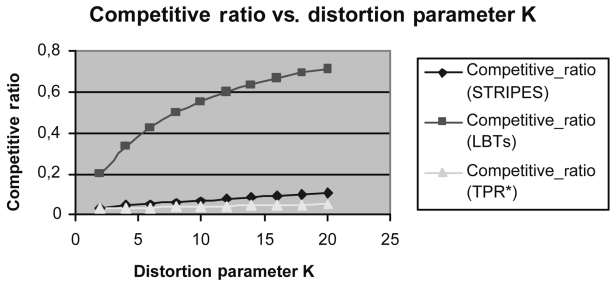
VI. EXPERIMENTAL EVALUATION

This section compares the query performance of our privacy-aware method, when incorporating STRIPES [21] (the best known solution), Lazy B-trees (LBTs) and TPR*-tree, respectively. We deploy spatiotemporal data that contain insertions at a single timestamp 0. In particular, objects' MBRs are taken from the LA spatial dataset (128971 MBRs) (Downloaded from the Tiger website http://www.census.gov/geo/www/tiger/). We want to simulate a situation in which all objects move in a space with dimensions 100 × 100 km. Thus, each axis of the space is normalized to [0,100000]. For the TPR*-tree, each object is associated with a Velocity Bounded Rectangle (VBR), such that 1) the object does not change spatial extents during its movement, 2) the velocity value distribution is skewed (Zipf) towards 30 in range [30,50], and 3) the velocity can be either positive or negative with equal probability. As in [23], we will use a small page size, so the number of index nodes simulates realistic situations.

Thus, for all experiments, the page size is 1 kbyte, the key length is 8 bytes, whereas the pointer length is 4 bytes. Thus, the maximum number of entries (< x > or < y >, respectively) in Lazy B-tree is 1,024 / (8 + 4) = 85. Similarly, the maximum number of entries (2-D rectangles or < $x1$, $y1$, $x2$, $y2$ > tuples) in TPR*- tree is 1,024 / (4*8 + 4) = 27. Conversely, the STRIPES index maps predicted positions to points in a dual transformed space and indexes this space using a disjoint regular partitioning of space. Each of the two dual planes, are equally partitioned into four quads. This partitioning results in $4^2 = 16$ partitions that we term *grids*. Thus, the fanout of each non-leaf node is 16. All indexes except STRIPES have similar sizes for each dataset. Specifically, for LA, each tree has 4 levels and around 6700 leaves apart from the STRIPES index that has a maximum height of seven and consumes about 2.4 times larger disk space. Each query q has three parameters: $q_R len$, $q_v len$, and $q_T len$, such that 1) its MBR q_R is a square, with length $q_R len$, uniformly generated in the data space, 2) its VBR is $q_V = -q_v len/2$, $q_v len/2$, $-q_v len/2$, $q_v len/2$, and 3) its query interval is $q_T = [0, q_T len]$. The query cost is measured as the average number of node accesses in executing a workload of 200 queries with the same parameters. Implementations were performed in C++ including particular libraries from SECONDARY LEDA.

A. Query Cost Comparison

We measured the Competitive Ratio of LBTs method (this method incorporates two Lazy B-trees that index the appropriate b parameters in each projection respectively, and finally combines the two answers by detecting and filtering all the pair permutations), the TPR*-tree and STRIPES presented in [21]



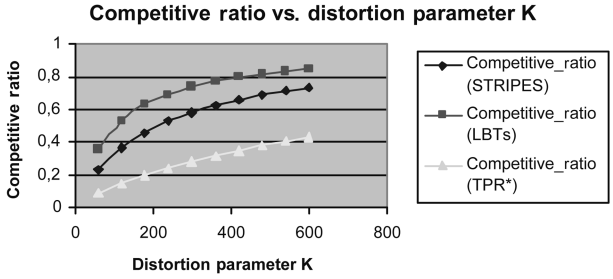


Fig. 9. $q_V len$ = 5, $q_T len$ = 50, $q_R len$ = 100 (top), $q_R len$ = 1,000 (bottom), R_{max} = 50 (top), R_{max} = 200 (bottom).

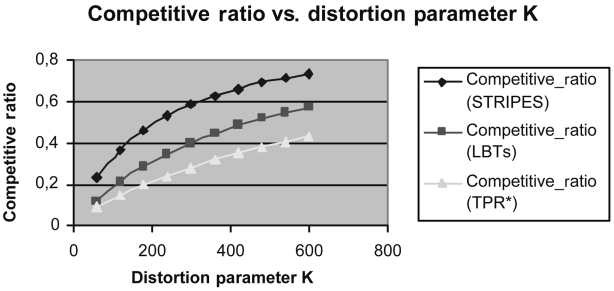
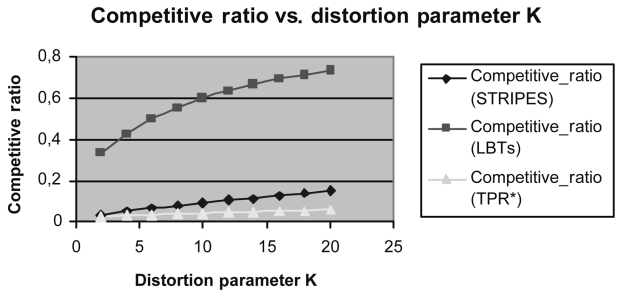


Fig. 10. $q_R len$ = 2,000, $q_V len$ = 5, $q_T len$ = 50, R_{max} = 500.



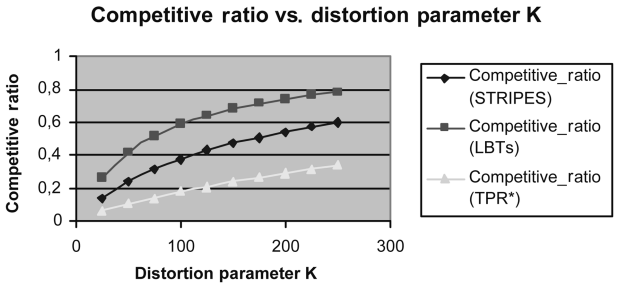
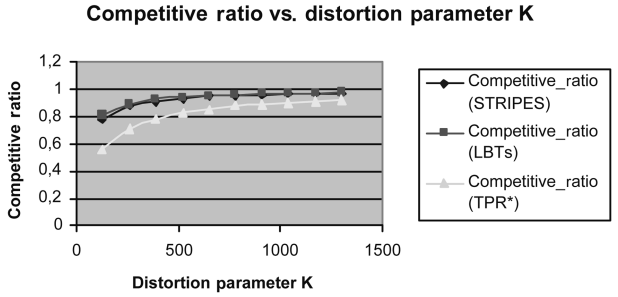


Fig. 11. $q_v len = 10$, $q_T len = 50$, $q_R len = 400$ (top), $q_R len = 1{,}000$ (bottom), $R_{max} = 100$ (top), $R_{max} = 200$ (bottom).



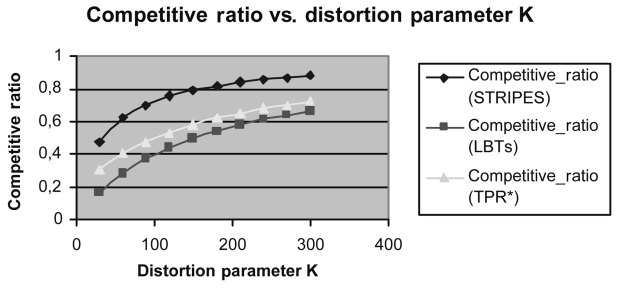


Fig. 12. $q_v len = 5$, $q_T len = 1$, $q_R len = 400$ (top), $q_R len = 1000$ (bottom), $R_{max} = 100$ (top), $R_{max} = 200$ (bottom).

and [22] respectively, using the same query workload, every 10,000 updates.

Figs. 9 to 13 show the Competitive Ratio as a function of K (for datasets generated from LA as described above), using workloads with different parameters. Parameter K represents boundary trajectories of buffers that intersect with the query rectangle, and obviously requires an additional filtering on the fly process. Obviously, the required number of block transfers

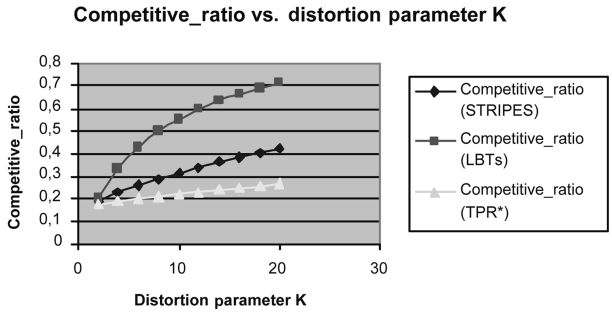


Fig. 13. $q_R len = 400$, $q_v len = 5$, $q_T len = 100$, $R_{max} = 100$.

depends on the answer's size, as well as the size of K.

Fig. 9 depicts how competitive to the optimal solution the LBTs method is, in comparison to TPR*-tree and STRIPES. The Ratio degrades as the query rectangle length grows from 100 to 1,000. When the query rectangle length or equivalently the query surface becomes extremely large (e.g., 2000), then the STRIPES index becomes more competitive (Fig. 10).

Fig. 11 depicts how competitive to the optimal solution the LBTs method is, towards to TPR*-tree and STRIPES, in case the velocity vector grows. The Ratio degrades as the query rectangle length grows from 400 to 1,000.

Fig. 12 depicts the performance of all methods where the time interval length degrades to 1. Even in this case, the LBTs method is more competitive than STRIPES and TPR*-tree. As the query rectangle length grows from 400 to 1,000, the LBTs method advantage decreases; we remark that STRIPES becomes faster, whereas LBTs method has exactly the same performance as the TPR*-trees.

Fig. 13 depicts the efficiency of LBTs solution in comparison to that of TPR*-trees and STRIPES, respectively, in the case where the time interval length increases to 100.

VII. CONCLUSIONS

We presented the problem of anonymity preservation data publishing in moving objects databases. In particular, we studied the case where the trajectory of a mobile user on the plane is no longer a polyline in a two-dimensional space, instead it is a two-dimensional surface, which covers the real location of the mobile user on the XY-plane, in the anonymized (kNN) request. We reinforced the privacy model by clustering the mobile users according to their motion patterns in (u, θ) plane, where u and θ define the velocity measure and the motion direction (angle), respectively.

In this case, the anonymized (kNN) request looks up neighbors, who belong to the same cluster with the mobile requester in (u, θ) space. By transforming the surface's boundary polylines to dual-points, we experimentally focused on the impact of information distortion introduced by this space translation.

REFERENCES

1. R. Bajaj, S. L. Ranaweera, and D. P. Agrawal, "GPS: location-

tracking technology," *Computer*, vol. 35, no. 4, pp. 92-94, Apr. 2002.
2. R. Clarke, "Person location and person tracking: technologies, risks and policy implications," *Information Technology & People*, vol. 14, no. 2, pp. 206-231, 2001.
3. B. Hoh, M. Gruteser, H. Xiong, and A. Alrabady, "Preserving privacy in GPS traces via uncertainty-aware path cloaking," *Proceedings of the 14th ACM Conference on Computer and Communications Security*, Alexandria, VA, 2007, pp. 161-171.
4. T. D'Roza and G. Bilchev, "An overview of location-based services," *BT Technology Journal*, vol. 21, no. 1, pp. 20-27, 2003.
5. C. Hauser and M. Kabatnik, "Towards privacy support in a global location service," *Proceedings of the IFIP Workshop on IP and ATM Traffic Management*, Paris, France, 2001, pp. 81-89.
6. M. Gruteser and D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking," *Proceedings of the 1st International Conference on Mobile Systems, Applications and Services*, San Francisco, CA, 2003, pp. 31-42.
7. M. Duckham and L. Kulik, "Location privacy and location-aware computing," *Dynamic and Mobile GIS: Investigating Changes in Space and Time*, R. Billen, E. Joao, and D. Forrest, Eds., Boca Raton, FL: CRC Press, ch. 3, pp. 34-51, 2006.
8. C. Ardagna, M. Cremonini, S. De Capitani di Vimercati, and P. Samarati, "Access control in location-based services," *Privacy in Location-Based Applications. Lecture Notes in Computer Science Vol. 5599*, C. Bettini, S. Jajodia, P. Samarati, and X. Wang, Eds., Heidelberg: Springer Berlin, pp. 106-126, 2009.
9. M. Gruteser and X. Liu, "Protecting privacy in continuous location-tracking applications," *IEEE Security and Privacy*, vol. 2, no. 2, pp. 28-34, 2004.
10. L. Kulik, "Privacy for real-time location-based services," *SIGSPATIAL Special*, vol. 1, no. 2, pp. 9-14, Jul. 2009.
11. G. Ghinita, "Private queries and trajectory anonymization: a dual perspective on location privacy," *Transactions on Data Privacy*, vol. 2, no. 1, pp. 3-19, 2009.
12. P. Samarati, "Protecting respondents' identities in microdata release," *IEEE Transactions on Knowledge and Data Engineering*, vol. 13, no. 6, pp. 1010-1027, 2001.
13. L. Sweeney, "K-anonymity: a model for protecting privacy," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 5, pp. 557-570, 2002.
14. A. Gkoulalas-Divanis, V. S. Verykios, and P. Bozanis, "A network aware privacy model for online requests in trajectory data," *Data and Knowledge Engineering*, vol. 68, no. 4, pp. 431-452, 2009.
15. B. Salzberg and V. J. Tsotras, "Comparison of access methods for time-evolving data," *ACM Computing Surveys*, vol. 31, no. 2, pp. 217-221, 1999.
16. V. Gaede and O. Günther, "Multidimensional access methods," *ACM Computing Surveys*, vol. 30, no. 2, pp. 170-231, 1998.
17. Y. Manolopoulos, Y. Theodoridis, and V. J. Tsotras, Advanced Database Indexing, Boston, MA: Kluwer Academic Publishers, 2000.
18. C. S. Jensen, D. Lin, and B. C. Ooi, "Query and update efficient b+-tree based indexing of moving objects," *Proceedings of the 30th International Conference on Very Large Data Bases*, Toronto, Canada, 2004, pp. 768-779.
19. G. Kollios, D. Gunopulos, and V. J. Tsotras, "On indexing mobile objects," *Proceedings of 18th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, Philadelphia, PA, 1999, pp. 261-272.
20. D. Papadopoulos, G. Kollios, D. Gunopulos, and V. J. Tsotras, "Indexing mobile objects on the plane," *Proceedings of the 13th International Workshop on Database and Expert Systems Applications*, Aix-en-Provence, France, 2002, pp. 693-697.
21. J. M. Patel, Y. Chen, and V. P. Chakka, "STRIPES: an efficient index for predicted trajectories," *Proceedings of the ACM SIGMOD International Conference on Management of Data*, Paris, France, 2004, pp. 635-646.
22. Y. Tao, D. Papadias, and J. Sun, "The TPR*-tree: an optimized spatio-temporal access method for predictive queries," *Proceedings of the 29th International Conference on Very Large Data Bases*, Berlin, Germany, 2003, pp. 790-801.
23. N. Beckmann, H. P. Kriegel, R. Schneider, and B. Seeger, "The R-tree: an efficient and robust access method for points and rectangles," *Proceedings of the ACM SIGMOD International Conference on Management of Data*, Atlantic City, NJ, 1990, pp. 322-331.
24. P. K. Agarwal, L. Arge, J. Erickson, P. G. Franciosa, and J. S. Vitter, "Efficient searching with linear constraints," *Proceedings of the 17th ACM SIGART-SIGMOD-SIGART Symposium on Principles of Database Systems*, Seattle, WA, 1998, pp. 169-178.
25. S. Saltenis, C. S. Jensen, S. T. Leutenegger, and M. A. Lopez, "Indexing the positions of continuously moving objects," *Proceedings of the ACM SIGMOD International Conference on Management of Data*, Dallas, TX, 2000, pp. 331-342.
26. S. Saltenis and C. S. Jensen, "Indexing of moving objects for location-based services," *Proceedings of the 18th International Conference on Data Engineering*, San Jose, CA, 2002, pp. 463-472.
27. S. Sioutas, K. Tsakalidis, K. Tsichlas, C. Makris, and Y. Manolopoulos, "A new approach on indexing mobile objects on the plane," *Data and Knowledge Engineering*, vol. 67, no. 3, pp. 362-380, 2008.
28. J. Han and M. Kamber, Data Mining: Concepts and Techniques, 2nd ed., San Francisco, CA: Morgan Kaufmann Publishers, 2006.
29. M. H. Dunham, Data Mining: Introductory and Advanced Topics, Upper Saddle River, NJ: Prentice Hall/Pearson Education, 2003.
30. J. MacQueen, "On convergence of k-means and partitions with minimum average variance (abstract)," *Annals of Mathematical Statistics*, vol. 36, no. 3, p. 1084, Jun. 1965.
31. D. Pelleg and A. W. Moore, "X-means: extending k-means with efficient estimation of the number of clusters," *Proceedings of the 17th International Conference on Machine Learning*, Stanford, CA, 2000, pp. 727-734.
32. G. Hamerly and C. Elkan, "Learning the k in k-means," *Advances in Neural Information Processing Systems 16: Proceedings of the 2003 Conference*, S. Thrun, L. K. Saul, and B. Scholkopf, Eds., Cambridge, MA: MIT Press, pp. 281-288, 2004.
33. H. Koivistoinen, M. Ruuska, and T. Elomaa, "A voronoi diagram approach to autonomous clustering," *Discovery Science. Lecture Notes in Computer Science Vol. 4265*, L. Todorovski, N. Lavrac, and K. Jantke, Eds., Heidelberg: Springer Berlin, pp. 149-160, 2006.
34. B. Kao, S. D. Lee, F. K. F. Lee, D. W. Cheung, and W. S. Ho, "Clustering uncertain data using voronoi diagrams and R-tree index," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 9, pp. 1219-1233, 2010.

35. P. S. Bishnu and V. Bhattacherjee, "CTVN: clustering technique using voronoi diagram," *International Journal of Recent Trends in Engineering*, vol. 2, no. 3, pp. 13-15, Nov. 2009.
36. F. Aurenhammer, T. U. Graz, R. Klein, F. Hagen, and P. I. Vi, "Voronoi diagrams," *Handbook of Computational Geometry*, J. R. Sack and J. Urrutia, Eds., Amsterdam, The Netherlands: North-Holland Publishing Co., pp. 201-290, 2000.
37. A. Aggarwal, L. J. Guibas, J. Saxe, and P. W. Shor, "A linear-time algorithm for computing the voronoi diagram of a convex polygon," *Discrete & Computational Geometry*, vol. 4, no. 1, pp. 591-604, 1989.
38. J. Goldstein, U. Shaft, R. Ramakrishnan, and J. B. Yu, "Processing queries by linear constraints," *Proceedings of the 16th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, Tucson, AZ, 1997, pp. 257-267.
39. A. Kaporis, C. Makris, G. Mavritsakis, S. Sioutas, A. Tsakalidis, K. Tsichlas, and C. Zaroliagis, "ISB-tree: a new indexing scheme with efficient expected behaviour," *Algorithms and Computation. Lecture Notes in Computer Science Vol. 3827*, X. Deng and D. Z. Du, Eds., Heidelberg: Springer Berlin, pp. 318-327, 2005.
40. O. Abul, F. Bonchi, and M. Nanni, "Never walk alone: uncertainty for anonymity in moving objects databases," *Proceedings of the 24th International Conference on Data Engineering*, Cancun, Mexico, 2008, pp. 376-385.

Spyros Sioutas

Spyros Sioutas was born in Greece, in 1975. He graduated from the Department of Computer Engineering and Informatics, School of Engineering, University of Patras, in December 1997. He received his Ph.D. degree from the Department of Computer Engineering and Informatics, in 2002. He is now an Assistant Professor in Informatics Department of Ionian University. His research interests include Data Management, Computational Geometry, GIS, Data Structures, Advanced Information Systems, P2P Networks and Web Services. He has published over 60 papers in various scientific journals and refereed conferences.

Emmanouil Magkos

Emmanouil Magkos received his first degree in Computer Science from the University of Piraeus, Greece in 1997. In April 2003, he received a Ph.D. in Security and Cryptography from the University of Piraeus, Greece. Currently he is affiliated with the Department of Informatics, Ionian University, Corfu, Greece, where he holds the position of Lecturer in computer security and cryptography. His current research interests include: security and privacy in distributed information systems and pervasive environments, key management in wireless ad-hoc networks, intrusion detection for malware propagation.

Ioannis Karydis

Ioannis Karydis was born in Athens, Greece in 1979. He received a BEng (2000) in Engineering Science & Technology from Brunel University, UK, an MSc (2001) in Advanced Methods in Computer Science from Queen Mary University, UK and a PhD (2006) in Mining and Retrieval Methods for Acoustic and Symbolic Music Data from the Aristotle University of Thessaloniki, Greece. He has published more than 20 papers in various scientific journals and refereed conferences and currently is a contract lecturer at the Ionian University, Greece. His research interests include music databases, music information retrieval (indexing & searching), music genre classification, musical similarity using contextual information, continuous querying in musical streams, cultural information systems and privacy issues in databases.

Vassilios S. Verykios

Vassilios S. Verykios received the Diploma in Computer Engineering from the University of Patras in Greece, in 1992 and the MS and the PhD degrees from Purdue University in 1997, and 1999, respectively. He has been an associate professor in the School of Science and Technology, at the Hellenic Open University in Patras, Greece since January, 2011. His main research interests include data management, privacy, security and anonymity, data mining, data reconciliation, and privacy preservation record linkage. He has published over 70 papers in major referred journals and in the proceedings of international conferences and workshops. He has coauthored a monograph on Association Rule Hiding for Data Mining by Springer.