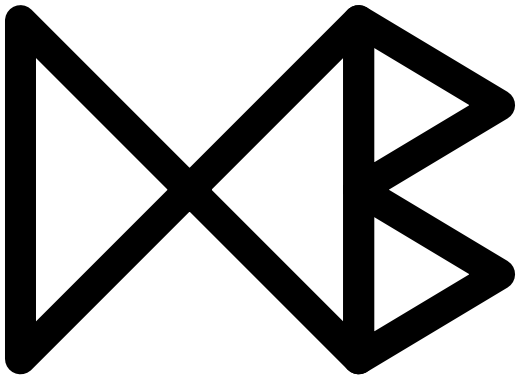


Privacy-Preserving Data Mining on Moving Object Trajectories

Győző Gidófalvi, Xuegang Huang, Torben Bach Pedersen

May 1, 2007

TR-20

A DB Technical Report

Title	Privacy-Preserving Data Mining on Moving Object Trajectories
Author(s)	Győző Gidófalvi, Xuegang Huang, Torben Bach Pedersen
Publication History	May 2007. A DB Technical Report. Extended version of: Győző Gidófalvi, Xuegang Huang, Torben Bach Pedersen, "Privacy-Preserving Data Mining on Moving Object Trajectories," in *Proceedings of the 8th International Conference on Mobile Data Management*, Mannheim, Germany, May 7–11, 2007.

The DB TECH REPORTS icon is made from two letters in an early version of the Rune alphabet, which was used by the Vikings, among others. Runes have angular shapes and lack horizontal lines because the primary storage medium was wood, although they may also be found on jewelry, tools, and weapons. Runes were perceived as having magic, hidden powers. The first letter in the logo is "Dagaz," the rune for day or daylight and the phonetic equivalent of "d." Its meanings include happiness, activity, and satisfaction. The second letter is "Berkano," which is associated with the birch tree. Its divinatory meanings include health, new beginnings, growth, plenty, and clearance. It is associated with Idun, goddess of Spring, and with fertility. It is the phonetic equivalent of "b."

Abstract

The popularity of embedded positioning technologies in mobile devices and the development of mobile communication technology have paved the way for powerful location-based services (LBSs). To make LBSs useful and user–friendly, heavy use is made of context information, including patterns in user location data which are extracted by data mining methods. However, there is a potential conflict of interest: the data mining methods want as precise data as possible, while the users want to protect their privacy by not disclosing their exact movements. This paper aims to resolve this conflict by proposing a general framework that allows user location data to be anonymized, thus preserving privacy, while still allowing interesting patterns to be discovered. The framework allows users to specify individual desired levels of privacy that the data collection and mining system will then meet. Privacy-preserving methods are proposed for two core data mining tasks, namely *finding dense spatio–temporal regions* and *finding frequent routes*. An extensive set of experiments evaluate the methods, comparing them to their non-privacy-preserving equivalents. The experiments show that the framework still allows most patterns to be found, even when privacy is preserved.

1 Introduction

The efficient management of moving object databases has gained much interest in recent years due to the development of mobile communication and positioning technologies. A typical way of representing moving objects is to use the trajectories. Much work has focused on the topics of indexing, query processing and data mining of moving object trajectories, but little attention has been paid to the preservation of privacy in this setting. In many applications such as intelligent transport systems (ITS) and fleet management, floating car data (FCD), i.e., tracked vehicle locations, are collected, and used for mining traffic patterns. For instance, mining vehicle trajectories in urban transportation networks over time can easily identify dense areas (roads, junctions, etc.), and use this for predicting traffic congestion. By data mining the periodic movement patterns (objects follow similar routes at similar times) for individual drivers, personalized, context-aware services can be delivered. However, exposing location/trajectory data of moving objects to application servers can cause threats to the *location privacy* of individual users. For example, a service provider with access to trajectory data can study a user's personal habits. It is not enough to keep the user ID secret, since common locations such as the home and office address can be found by correlating historical trajectories, followed by cross-referencing these locations with, e.g., Yellow Pages, to reveal user identity. Privacy–preserving data mining of moving object trajectories has not been addressed in the literature. The challenge of obtaining detailed, accurate patterns from anonymized location and trajectory data is the motivation for this paper.

This paper makes a number of novel contributions that together constitute an effective method for trajectory data collection and mining that preserves user location privacy. First, the paper proposes a novel *anonymization model* for preservation of location privacy on moving object trajectories. Here, the users specify their requirements of location privacy, based on the notions of *anonymization rectangles* and *location probabilities*, intuitively saying how precisely they want to be located in which areas. Second, the paper shows a *common problem* with existing methods based on the notion of *k–anonymity*. This problem allows an adversary to infer a commonly occurring location of a user, e.g., the home address, by correlating several observations. Third, the paper presents an effective *grid-based framework* for data collection and mining over the anonymized trajectory data. The framework is based on the notions of *anonymization grids* and *anonymization partitionings* which allow effective management of both the user-specified location privacy requirements and the anonymized trajectory data. Along with the framework, three *policies* for constructing *anonymization rectangles*, called *common regular partitioning*, *individual regular partitioning*, and *individual irregular partitioning* are presented. These policies avoid the problems in existing methods. Fourth, the paper presents a *client-server architecture* for an efficient implementation of the system. A distinguishing feature of the architecture is that anonymization is performed solely on the client, thus removing the need for trusted middleware. Fifth, the paper presents techniques for solving two basic trajectory data mining operations, namely *finding dense spatio–temporal areas* and *finding frequent routes*. The techniques are based on probabilistic counting. Finally, *extensive experiments* with a prototype implementation show the effectiveness of the approach, by comparing the presented solutions to their non-privacy-preserving equivalents. The experiments show that the framework still allows most patterns to be found, even when privacy is preserved. In summary, we believe this paper to be the first to consider the topic of data mining on anonymized trajectory data.

The rest of this paper is organized as follows. Section 2 explores related work. Section 3 discusses anonymization models of trajectory data. Section 4 presents the grid-based framework, while Section 5 presents an empirical

evaluation. Finally, Section 6 concludes and points out future directions for research.

2 Related Work

Privacy protection in databases has been a core area in the database research community and many related topics have appeared in the literature, such as access control, inference control and statistical databases. To protect the privacy of LBSs users, three existing solutions [7, 8, 13] propose to use a trusted middleware (an anonymizer) that maintains location updates and queries between the LBS users and LBS server. Each time a query request is sent from a LBS user, the anonymizer, in the spirit of *k–anonymity* [14], encloses the query location in a "cloaking" rectangle that includes both the query location and the locations of $k-1$ other users, and sends the query to the LBS server with the cloaking rectangle. The LBS server returns a superset of the results and the final results are filtered by the anonymizer and sent back to each LBS user.

This method for anonymizing locations and trajectories has several problems. First, it requires trusted middleware. Second, while [13] provides an effective solution for finding locations of the other $k-1$ users in the presence of such trusted middleware, a solution to the same task in an environment that contains only untrusted components is unknown and likely to be computationally prohibitive. Third, the notion of location privacy that is guaranteed by *k–anonymity* may not be satisfactory in the case where a large number of moving objects stay in a small area where users do not want to be observed (such as a red light district). This problem can be eliminated by requiring cloaking rectangles to have a minimum area [13]. Fourth, the cloaking rectangles calculated for the same user for the same location at different times depends on locations of the other $k-1$ users, and hence may vary in extent and location. This, in a sense *non-deterministic* or *probabilistic* nature of cloaking rectangles sacrifices location privacy, as demonstrated later. Finally, traditional mining methods cannot be easily and effectively adapted to the anonymized location or trajectory.

This paper does not consider *k-anonymity* and does *not* assume the existence of trusted middleware for providing the *k-anonymity* rectangles. Instead, we focus on novel ways to conceal the actual moving object trajectories while still allow the data mining algorithms on the LBS server to extract detailed, accurate traffic patterns and rules from the anonymized trajectory data. Note that our solution does *not even aim* to provide k-anonymity. The reason is that for some applications, e.g., traffic services in remote areas, even a rather small k will cause the reported rectangles to become extremely large, and thus worthless for the purpose of mining. Instead, our solution will perform a *spatial anonymization* that meets the user's requirements for location privacy.

Spatio–temporal data mining is an on-going topic in the database community. Approaches have appeared for finding dense areas of moving objects [11, 12, 15] and extracting spatio–temporal rules and patterns [9, 16]. Our paper is focused on discovering areas with potential traffic jams and roads that are frequently used by drivers. Two very related papers [11, 12] study the querying of spatio–temporal regions with a high concentration of moving objects. The first paper [11] divides the data space into a uniform grid so that the density query is simplified as reporting cells that satisfy the density conditions. This solution provides fast answers, but can lead to *answer loss* (as termed in the second paper [12]), such as regions that cover boundaries of several cells with a high density of objects (but each individual cell does not contain enough number of objects to be dense). The second paper [12] provides a new definition of density query that eliminates answer loss and proposes a two-phase filter-and-refinement algorithm for computing the density queries. A method to provide approximate answers to *distinct* spatio-temporal aggregation is proposed in [15], where aggregation is grid–based, and the distinct criterion is time– and space–effectively solved by combining a spatio–temporal index (aRB-tree) and sketches. Finding frequently travelled routes taken by moving objects has many applications in telematics, ITS and LBS. This has been recognized recently in [10], where frequent route mining is mapped to frequent itemset mining, after trajectories are transformed to a set of spatio–temporal grid cells. We take a similar approach to finding frequent routes, but focus on the privacy preserving aspect of the data mining.

A lot of recent research work has focused on techniques for privacy–preserving data mining [2]. This topic has appeared due to the advances in data collection and dissemination technologies which force existing data mining algorithms to be reconsidered from the point of view of privacy preservation. Various papers have recently addressed privacy–preserving data mining. Important techniques include perturbation, condensation, and data hiding with conceptual reconstruction. Paper [17] presents a good review of these techniques. The techniques proposed in this paper follow the spirit of a common strategy used for privacy–preserving data mining, namely *generalization*.

3 Spatio-Temporal Anonymization

For the simplicity of our discussion, we assume that the time domain $\mathbb{T}$ is totally ordered and use the non-negative numbers as the time domain. We define the trajectory of a moving object in 2-dimensional (2D) space as a sequence of tuples $S = \langle (loc_1, t_1), \ldots, (loc_n, t_n) \rangle$ where $loc_i \in \mathbb{R}^2$ ($i = 1, \ldots, n$) describe locations, and $t_1 < t_2 < \ldots < t_n \in \mathbb{T}$ are irregularly spaced but temporally ordered time instances, i.e., gaps are allowed.

We consider to anonymize the trajectory by reducing the spatio-temporal resolution of the 2D space. One basic method is to enclose the trajectory into one or more space-time rectangles, denoted as *anonymization rectangles*. A formal definition is as follows:

Definition 3.1. Given an area size $areasize \in \mathbb{R}^+$ and a probability threshold $maxLocProb \in [0, 1]$, an **anonymization rectangle** satisfying ($areasize$,$maxLocProb$) for a moving object o is a three-tuple (R, t_s, t_e), where $t_s < t_e \in \mathbb{T}$ are two time instances, and R is a 2D rectangle such that the maximum probability that can be *inferred* about o being in any subregion A of size $areasize$ in R during the period $[t_s, t_e]$ is at most `maxLocProb`.

Definition 3.2. Given an area size $areasize \in \mathbb{R}^+$, we term this maximum probability that can be *inferred* about the whereabouts of object o inside R as the **location probability** of R and denote it as $R.LocProb$.

Privacy preservation in spatio-temporal data sets is challenging because spatio-temporal data sets are so rich in correlations, allowing many "privacy attack" strategies that are difficult to counteract and sometimes even to anticipate. We believe to protect against a few obvious threats, namely, 1) detection of frequent private/personal/individual locations due to self-correlations in historical spatio-temporal (trajectory) data sets, 2) detection of the current position due to physical mobility constraints on objects (maximum speed, road network, spatio-temporal restrictions in general).

In our definitions we emphasize *inferred*, because the straight-forward, uniform spatio-temporal probability distribution for the location of an object o does not hold for any rectangle $R \in \mathbb{R}^+$. By relating external spatial and/or temporal data sources, which put limitations on the possible locations of o, more specific distributions can be derived that sacrifice the privacy of o. This is illustrated in Figure 1, where anonymization rectangle R of o is composed of 4 unit-area cells (c_1, c_2, c_4, c_5). Not combining any external data sources, $R.LocProb = 1/4$. Knowing that cells c_1 and c_4 are covered by water, $R.LocProb = 1/2$. Finally, knowing about the location and opening hours of the Nature Resort Park in cell c_2 and the current time (8am), $R.LocProb = 1$. Clearly, relating

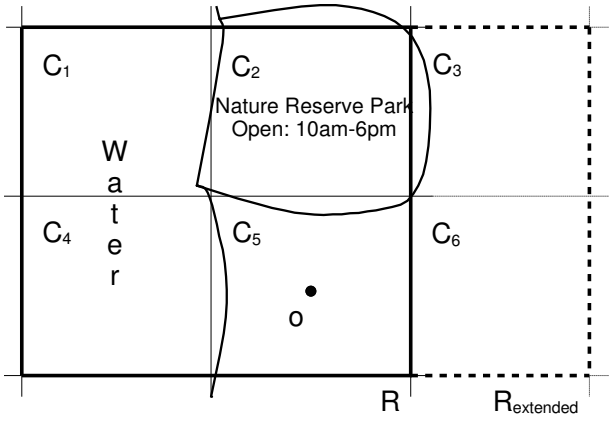


Figure 1: Location Privacy

more and more spatio-temporal, external data sources to R raises the location probability of it, and guarantees less privacy for o. One natural way to guarantee a location probability of at most `maxLocProb`, is to spatially, or temporally, extend R to $R_{extended}$, such that $R_{extended}.LocProb \leq$ `maxLocProb`. In Section 4.2, we will describe how to do this in practice.

If we denote the currently known spatio-temporal probability distribution for the location of an object o as PD_o, then any kind of "extra" external spatio-temporal information can be modeled as a function $F(PD_o)$ that returns a new spatio-temporal probability distribution PD'_o. If the location probability of *o* at certain locations

is then over the threshold `maxLocProb` with the new distribution, there is a problem that needs to be handled somehow, most often by enlarging the area partitions.

Intuitively, we can enclose the whole trajectory of a moving object into a single rectangle so that the anonymity of the trajectory is preserved. However, as the trajectories are often very long, the rectangles can be very big so that it becomes impossible for the data mining algorithms to return any useful results. Our proposal is to provide an **anonymized format** of the trajectory by cutting a long trajectory into pieces and enclosing each piece in an anonymization rectangle. This format can give opportunities for doing data mining without sacrificing location privacy.

3.1 Practical "Cut-Enclose" Implementation

The "cut-enclose" procedure splits the whole trajectory of a moving object o into a set of polylines which correspond to a set of time periods $\{[t_1, t_2], [t_2, t_3], [t_3, t_4], \ldots, [t_{k-1}, t_k]\}$, such that at any time instance $t_i \in \{t_2, t_3, \ldots, t_{k-1}\}$ o's trajectory crosses an edge between two neighboring anonymization rectangles R_i and R_{i+1}. Since around this instance t_i, o is more likely to be close to the edge between R_i and R_{i+1}, $R_{i+1}.LocProb$ will temporarily be higher, which might sacrifice the location privacy of o. More specifically, from the times spent in the previous anonymization rectangles, their sizes, and relative locations to each other, a malicious server can easily maintain a linear movement model of o. Using this movement model, when o sends the anonymization rectangle R_{i+1}, the malicious server can *deduce* a *possible location range* R^* of o, such that $R^*.LocProb >$ `maxLocProb`.

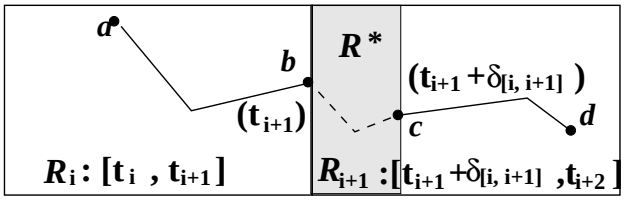


Figure 2: Time Delay Factor

To avoid this situation and preserve the location privacy of o, we introduce a **time delay factor** $\delta_{[i,i+1]}$ for delaying the sending of the anonymous rectangle R_{i+1} after leaving R_i. The factor $\delta_{[i,i+1]}$ can be calculated as follows. Object o can maintain the same linear movement model about its own movement as the malicious server can. Hence, at any time instance $t^* > t_i$, having entered R_{i+1}, o can calculate R^* and $R^*.LocProb$. As time progresses, the size of R^* is monotonically increasing and $R^*.LocProb$ is monotonically decreasing. Hence, at some time point $t^s > t_i$, when the associated $R^*.LocProb \leq$ `maxLocProb` it is *safe* for o to send R_{i+1} to the server. The time delay factor is then $\delta_{[i,i+1]} = t^s - t_i$.

Most moving objects are confined to road networks. In the presence of road networks, more sophisticated movement models are possible. Actual values for the time delay factor have been investigated for a number of network–based movement models on real–world datasets in [5], but this work had a different aim, namely to aid tracking.

3.2 Problems with Existing Methods

To construct an anonymization rectangle for a given piece of trajectory, one naive method is to randomly choose a location in the vicinity of the trajectory and use this location as the center to build the anonymization rectangle based on a pre-defined size. Another method, motivated from the discussion of *location k-anonymity* in the literature [7, 8, 13], is to build the anonymization rectangle that enclose this piece with trajectory pieces of $k-1$ other moving objects.

However, these two methods can lead to an undesired *loss of location privacy*. Sensitive locations that need to be kept private, or trajectory pieces that lead to these, are often re-visited by the objects many times, at a similar time of day. For example, objects (users), in the evening hours return to their *home* using the same path (trajectory piece). If on different occasions the anonymization rectangles for this trajectory piece are constructed in a *non-deterministic* way, the location of the trajectory piece can be narrowed down to the intersection of these

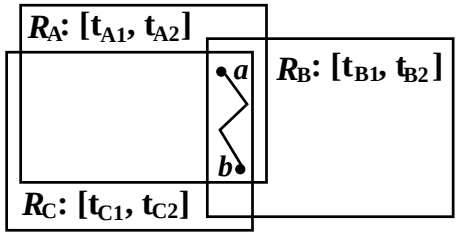


Figure 3: Overlapping Area

anonymization rectangles. This leads to an undesirable loss of privacy. In the example on Figure 3, object o returns to its *home* b using the same trajectory piece $[a, b]$ on three different occasion at the same time of the day. On the three occasions, three anonymization rectangles R_A, R_B, and R_C are constructed, such that they contain both the trajectory piece $[a, b]$ and the location b. Based on the multiple visits, the location of $[a, b]$ can be narrowed down to the small overlapping area of the anonymization rectangles.

In the next section, we present a grid-based solution and several methods for constructing anonymization rectangles in a *deterministic* way on this grid, thereby avoiding the privacy loss described above. The grid-based framework also allows for an efficient implementation of the "cut-enclose" procedure described in Section 3.1.

4 A Grid-Based Solution

A basic method to anonymize location is to reduce the spatial resolution. Thus, instead of randomly constructing the anonymization rectangles or building the rectangles based on trajectories of other moving objects, we consider building all moving objects' anonymization rectangles based on a single, pre-defined 2D grid. We proceed to discuss the solution in detail.

4.1 Grid-Based Anonymization

We denote the whole 2D Euclidean space as $\mathbb{R}^2$ and proceed to define an anonymization grid and anonymization partitioning as follows.

Definition 4.1. An **anonymization grid** (briefly, a grid) G is a uniform grid of $\mathbb{R}^2$ with a pre-defined $O \in \mathbb{R}^2$ as the starting point and a side length l. An **anonymization partitioning** (briefly, a partitioning) is a set of pairwise disjoint sets of grid cells covering all of G.

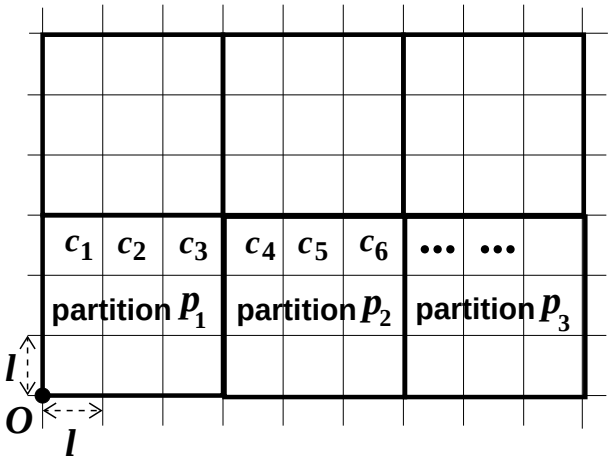


Figure 4: Anonymization Grid and Partitioning

As illustrated in Figure 4, given a starting point $O \in \mathbb{R}^2$, the anonymization grid (briefly, the grid) G uniformly divides the whole space into square-shaped **grid cells**, each of which has side length l. Each grid cell has an ID value, such as $c_1, c_2, \cdots$ in Figure 4. A *partition* of a partitioning that is defined on the grid is a set of grid cells.

Next, we develop several methods for constructing **anonymization partitionings** based on the anonymization grid. All of the partitionings are constructed *deterministically*, thereby avoiding the privacy loss due to overlapping partitions.

Common Regular Partitioning (CRP): The simplest method is to define a single, regular partitioning that is used by all the objects. We call a partitioning **regular** if all the partitions are rectangles with side lengths $i_x \times l$ and $i_y \times l$, where i_x and i_y are integers.

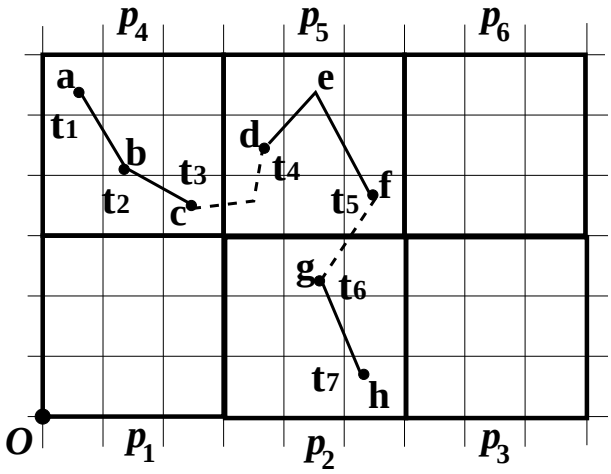


Figure 5: Anonymized Trajectory

Such a regular partitioning can be seen as a coarser grid on the 2D space. As illustrated in Figure 4, given the grid (the grid of thin lines), the partitioning (the grid of thick lines) is defined by an origin O and $i_x = 3, i_y = 3$. In the example the grid cells c_1, c_2, c_3 belong to the partition p_1. With the grid and partitioning, we are able to transform a moving object trajectory to a set of non-overlapping anonymization rectangles to preserve anonymity. For instance, given the trajectory $\langle (a, t_1), (b, t_2), \cdots, (h, t_7) \rangle$ in Figure 5, we build a grid on the 2D space and make the partitioning on the grid. The partitions are denoted as $p_1, \cdots, p_6$ in the figure and they are non-overlapping rectangles. As described in Section 3.1, given the time delay factor δ, the whole trajectory is cut into several pieces with $t_4 - t_3 = \delta$ and $t_6 - t_5 = \delta$. Then, the whole trajectory is transformed into a list of anonymization rectangles $\langle (p_4, t_1, t_3), (p_5, t_4, t_5), (p_2, t_6, t_7) \rangle$.

The above described partitioning guarantees the same minimal level of privacy for all users in any region of the space. This method of partitioning is termed Common Regular Partitioning (CRP).

Fundamental spatio–temporal data mining tasks, like finding dense spatio–temporal regions, and frequent routes, are based on simple counts or identities of the users that are present in a given spatio–temporal region. Since in the CRP model all users report the same set of grid cells for the same location, the spatio–temporal granularity of any pattern found is lower bounded by the size of a partition. In the example in Figure 5, the size of the common partition is 9 grid cells, hence the smallest dense ST-region that can be found will be 9 grid cells.

Individual Regular Partitioning (IRP): Not all objects require the same level of location privacy. This requirement of individual objects can easily be accommodated in our anonymization grid-based framework. Objects requiring higher levels of privacy construct and use a regular partitioning with larger partitions, while objects requiring lower levels of privacy define and use a regular partitioning with smaller partitions. This method of partitioning is termed Individual Regular Partitioning (IRP).

Besides being more flexible in terms of the objects' privacy requirements, the IRP method allows the discovery of patterns of spatio–temporal granularity that is equal to the size of a single grid cell (if enough data is present).

Individual Irregular Partitioning (IIP): Objects may have different location privacy requirements in different regions of space. For example, most objects (users) desire a higher level of location privacy when being at *home* or the *work place* than when being in transition or when being in other general areas of the city. This requirement of individual objects can again be easily accommodated in our anonymization-grid-based framework. Objects

can be allowed to individually define privacy levels for regions in space that reflect their needs. The definition of these regions can be either manual, or can be aided by discovering frequent (presumably sensitive) locations of individual objects. Since the selection or discovery of these sensitive locations can be accomplished on the client side, it can be kept private. This method of partitioning is termed as Individual Irregular Partitioning (IIP).

The IIP method also allows the discovery of patterns of spatio–temporal granularity that is equal to the size of a single grid cell. The additional ability to define spatially varying privacy levels not only adds more privacy control, but it is also expected to allow the discovery of more patterns with finer spatio–temporal granularity. This is due to the fact that most objects are expected to require higher levels of location privacy in relatively small subregions. The more detailed patterns are expected to be more useful for ITS applications.

With our grid-based framework, the knowledge one can infer about the whereabouts of a user does not depend on the number of samples collected. The certainty of the inference only depends on the amount of external spatio-temporal information available for the anonymous rectangle.

4.2 System Architecture

We implement the grid-based solution based on a client/server architecture. As illustrated in Figure 6, the server side has three components, the *anonymity component* which defines one or more grids and communicates them to the client, the *storage component* which collects the anonymization rectangles sent from the clients and stores the data on disk, and the *data mining component* which discovers certain patterns and rules either directly from the incoming data stream or from the historical data retrieved from the storage component.

The clients are responsible for accepting an anonymization grid and developing a partitioning based on the grid. In practice, the partitioning will be made in one of two ways: a) the user selects among a small number of pre-computed partitionings to find one that meets their privacy requirements, or b) the partitioning is computed by a dedicated program on the client, based on user input about privacy requirements. Both a) and b) take available

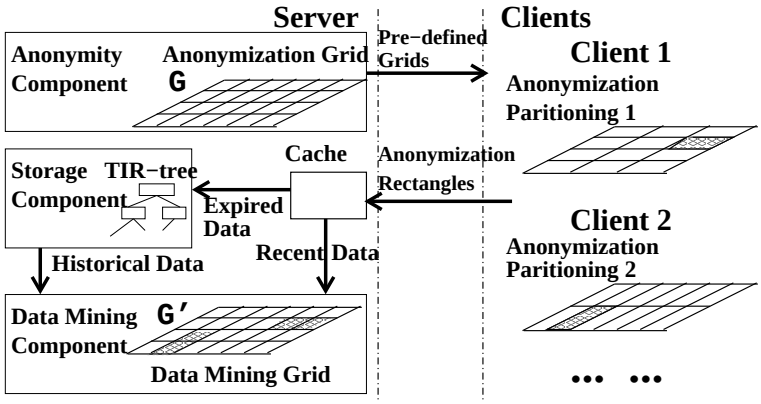


Figure 6: System Architecture

background knowledge into account. The framework can also handle the presence of road networks. If road networks are dense compared to the partition size, the framework can be used without modification. If not, the partitions have to be enlarged so that each partition contains enough road to get a location probability that is comparable to those of the other partitions.

These client- and grid-specific partitionings are stored on the clients and only anonymization rectangles (in the form of sets of grid cells), which are computed *at the clients*, are transmitted to the server. We do assume that the client has a fair amount of storage and CPU power, but not more than what can be found in most currently available smartphones or PDAs.

Saving a partitioning at the client side does not take much space. For a regular partitioning, where partitions form a regular grid, it is enough to store the starting point and the side length of the partitioning. Finding the partition that corresponds to a location is a matter of simple arithmetic. For a non-regular partitioning, where partitions do not form a regular grid, i.e., are of different size and/or shape, partitions can be kept in an R-tree. Finding the partition that corresponds to a location can be done by issuing a stabbing query on the R-tree for the location. The communication cost between the clients and the server is very low since the grids can be described

with the starting point and side length l of the grid, and the anonymization rectangles only involves a few data fields (i.e., coordinates of the client's current partition and the time instances).

The clients always send their current anonymization rectangle, i.e., partition, to the server. When the anonymized data is transmitted to the server, it is stored it in two places. To be able to perform data mining on historical data, the data is first stored in a time-interval R-tree (TIR-tree in Figure 6) on disk. The TIR-tree is a 1-dimensional R-tree that indexes the data on the time intervals. To be able to perform online data mining on the current data, the data is also stored in a *cache*, with a FIFO replacement policy as follows. According to the size of the cache, when a new anonymization rectangle of a moving object arrives, either the previous anonymization rectangle of this moving object (if in the cache) or the oldest data in the cache is deleted.

The system architecture in Figure 6 supports data mining on both historical trajectories and recent data. Each anonymization grid in the anonymity component corresponds to an in-memory instance of the same grid in the data mining component. For instance, the anonymization grid `G` in Figure 6 corresponds to the data mining grid `G'` (we assume the data mining component has enough memory to store `G'`). Based on this architecture, we proceed to present algorithms for discovering *dense ST-areas* and *frequent routes* on the anonymized trajectory data.

4.3 Finding Dense Spatio-Temporal Areas

Discovering dense areas is one of the most common topics for spatial and spatio-temporal data mining. Existing research work has explored density clustering [6], spatio-temporal dense area discovery [16], and density queries [12]. For dense area discovery on the anonymized trajectory data, the most basic operation is to find those grid cells that contain a large amount of moving objects during specified time intervals. In the anonymized format, objects are present in a grid cell with some *probability* only. Hence, we propose a **time interval *probabilistically* dense spatio-temporal area query**, or *dense ST-area query* for short, which can be seen as a basic, atomic operation for advanced dense area mining algorithms over the anonymization grid. Such advanced and complex data mining algorithms can be made by assembling this operations with other basic query types.

Specifically, suppose a moving object o corresponds to a partition P on a given anonymization grid `G`, a partition cell $p \in P$ contains o's trajectory during time interval $[t_s, t_e]$, and p includes grid cells $c_1, c_2, \ldots, c_k$. We use p as the anonymization rectangle for o's trajectory and each grid cell $c_i \in p$ has the location probability $c_i^o.LocProb = 1/k$ for o at any time instance during $[t_s, t_e]$. Let O^{c_i} be the set of moving objects whose anonymization rectangles include the grid cell c_i in at least one time instance during the time interval $[t_s, t_e]$. Then $c_i.count = |O^{c_i}|$ and $c_i.prob = \sum_{o \in O^{c_i}} c_i^o.LocProb/|O^{c_i}|$. Intuitively, $c_i.count$ is the *maximum* number of objects that *can* be inside c_i during $[t_s, t_e]$, while $c_i.prob$ is the *average* location probability of the objects that can be inside c_i during $[t_s, t_e]$. Consequently, $c_i.prob \times c_i.count$ is the *expected* number of objects inside c_i during $[t_s, t_e]$. Furthermore, we define the *pattern certainty* $c_i.cert = \prod_{o \in O^{c_i}} c_i^o.LocProb$ as the probability of *actually* having $c_i.count$ number of moving objects inside of c_i during $[t_s, t_e]$.

We say that a grid cell c_i is **probabilistically dense** during $[t_s, t_e]$ if $c_i.count \geq$ `min_count` and $c_i.prob \geq$ `min_prob`, for some given threshold values `min_count` and `min_prob`. Thus, we formulate the **dense ST-area query** as follows:

Definition 4.2. A **dense ST-area query** $Q = ([t_s, t_e], \texttt{min\_count}, \texttt{min\_prob})$ retrieves all the grid cells whose corresponding $count$ and $prob$ values during $[t_s, t_e]$ are greater than or equal to `min_count` and `min_prob`, respectively.

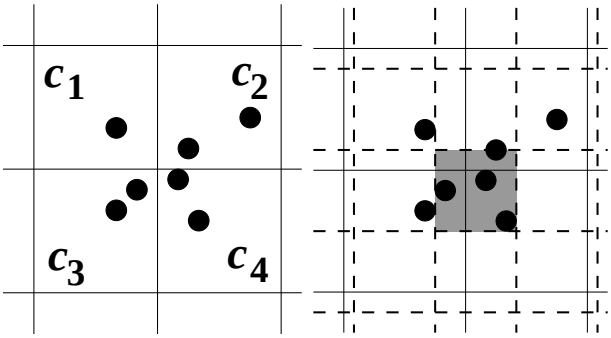
To process a dense ST-area query, the first step is to compute the $count$ and $prob$ values for each grid cell c_i for the specified time interval $[t_s, t_e]$. Based on the system architecture in Figure 6, we need to issue a range query over the TIR-tree to find all the anonymization rectangles whose time periods have intersections with $[t_s, t_e]$. Results of the range query are used to fill in the $count$ and $prob$ values for each cell c_i of the data mining grid `G'`. Then the set of dense ST-grid cells is:

$$D = \{c_i : c_i.count \geq \texttt{min\_count} \wedge c_i.prob \geq \texttt{min\_prob}\}$$

During the query time interval $[t_s, t_e]$ a moving object can leave and later reenter a given grid cell c_i. To avoid counting such an object multiple times for c_i, we maintain a hash array of object IDs and only update values for $c_i.count$ and $c_i.prob$ when an object ID is encountered for c_i for the first time. If we consider only approximate counts, these can be more effectively obtained using the methods from [15].

As we will see in Section 5, the cut-off criteria for dense areas presented above is in some cases not strict enough, thus generating too many dense areas (false positives). To remedy this, we introduce the alternative *steepest slope* cut-off criteria, which is calculated by first sorting the expected counts for dense areas passing the first criteria in descending order, finding the deltas between any two consecutive values, and making the cut-off where the (negative) delta is the smallest, i.e., where the "slope" is steepest.

The above method is simple to implement but can not discover all the dense areas that have the size of a single grid cell. As illustrated in Figure 7(a), grid cells c_1, c_2, c_3, c_4 have several moving objects (big dots in the figure) and the threshold `count`$= 4$. None of the four cells are reported as dense since the $count$ value of each is less than 4.



(a) Loss of Result (b) Using the Second Grid

Figure 7: Multi-Grid to Overcome Answer Loss

To overcome such loss of results, we can let the server provide several anonymization grids with different starting points (and perhaps side length values) and distribute these grids to the moving objects so that there are in equal amount of moving objects that build their partitioning based on each of the anonymization grids. This *Multi-Grid* approach can capture the answer loss with a single grid. As shown in Figure 7(b), the dense junction area of the four cells can be captured by the dark cell belonging to another grid. We leave this *Multi-Grid* extension for future work.

The time interval dense ST-area query can be seen as an atomic operation over the anonymized trajectory data. Advanced and complex data mining functions can be made by assembling this operations with other basic query types. We proceed to introduce another atomic operation, namely the **time interval *probabilistically* frequent route query**.

4.4 Frequent Route Mining

Next, we introduce another operation, namely the **time interval *probabilistically* frequent route query**, or *frequent route query* for short, which can again be seen as a basic, atomic operation for advanced frequent route mining algorithms over the anonymization grid.

Finding frequently travelled routes taken by moving objects has many applications in telematics, ITS and LBS. One such application, an intelligent rideshare application, was recently proposed in [10]. Here, similar to our work, trajectories are represented as a set of spatio–temporal grid cells, and the task of finding sharable long patterns in trajectories is posed as finding closed frequent itemsets, a common task in data mining.

We adopt a similar approach to answer a frequent route query in our anonymization grid based framework. Similarly to Section 4.3, we assume that for any historical time interval $[t_s, t_e]$ the locations of moving objects can be retrieved from the server side storage component in form of anonymization rectangles, which are stored as a set of grid cells, with corresponding location probabilities. To construct a spatio–temporal grid cell representation of trajectories, we split trajectories inside the query time interval into m sub–trajectories of duration $d = (t_e - t_e)/m$, by issuing m d–long time interval queries to the server side storage component at time instances $t_s, t_s + d, t_s + 2d, \ldots, t_e - 2d, t_e - d$. For each query result, we combine the returned grid cell IDs with the starting time of the query to form spatio-temporal grid cells. As a result of this transformation, a trajectory that falls inside the query time interval is represented as a set of spatio-temporal grid cell IDs, each of which is associated with a location

probability. After this transformation, we modify the traditional frequent itemset framework in [1] and map the problem of finding frequent routes to the problem of finding maximal frequent itemsets as follows.

Conforming to the naming convention used in the traditional frequent itemset mining framework, a spatio–temporal grid cell ID is equivalently referred to as an *item*. Then, a *transformed trajectory* t inside the query time interval, equivalently referred to as a *transaction* is a two–tuple (X, P), where X is an *itemset* and P are the corresponding probabilities of the items in X. For a given transaction $t = (X, P)$, we denote the probability of an item $i \in X$ as $i.prob$. Given a user–defined threshold `min_prob`, a transaction $t = (X, P)$ *probabilistically satisfies* an itemset Y if $Y \subseteq X$ and $\forall i \in Y \cap X$, $i.prob \geq$ `min_prob`. Given a set of transactions, the *probabilistic support* of an item i, denoted as $i.count$ is the number of transactions that probabilistically satisfy the itemset $Y = \{i\}$. Similarly, the probabilistic support of an itemset Y, denoted as $Y.count$ is the number of transactions that probabilistically satisfy all items $i \in Y$. Given a user–defined threshold `min_count`, an item i is *probabilistically frequent* if $i.count \geq$ `min_count`. Similarly, an itemset Y is probabilistically frequent if $Y.count \geq$ `min_count`. An itemset Y is a *maximal probabilistically frequent itemset* if there does not exists a probabilistically frequent itemset X such that $Y \subset X$.

Using this modified frequent itemset framework, referred to as the *probabilistically frequent itemset framework*, we define a **frequent route** to be a maximal probabilistically frequent itemset. Consequently, we formulate the **frequent route query** as follows:

Definition 4.3. A **frequent route query** $Q = ([t_s, t_e],$ `min_count`, `min_prob`$)$ retrieves all the maximal probabilistically frequent itemsets during $[t_s, t_e]$.

It is important to note that frequent routes returned by the query are not necessarily continuous in space or time. For example, a large set of objects may share the beginning and end of a route, but may make different detours in the middle of their trajectories. Such discontinuous, frequent routes can still be of great interest to telematics and ITS.

We adopt a prominent itemset mining algorithm, MAFIA [4], to our probabilistic frequent itemset framework as follows. A transaction t can probabilistically satisfy an itemset $Y = \{i_k\}$ iff $i_k \in t$ and $i_k.prob \geq$ `min_prob`. Hence, before the mining, items having a location probability less than `min_prob` are deleted from transactions. This preprocessing guarantees that all maximal frequent items mined in the traditional framework, will be maximal probabilistically frequent itemsets. Such a preprocessing of transformed transactions is a straight forward, linear time operation, which not only allows a simple, but also an efficient mapping between the two frameworks and tasks. Since items that have location probabilities less than `min_prob` cannot appear in a maximal probabilistically frequent itemset, eliminating them from the transactions before mining, reduces the search lattice and ultimately computation cost.

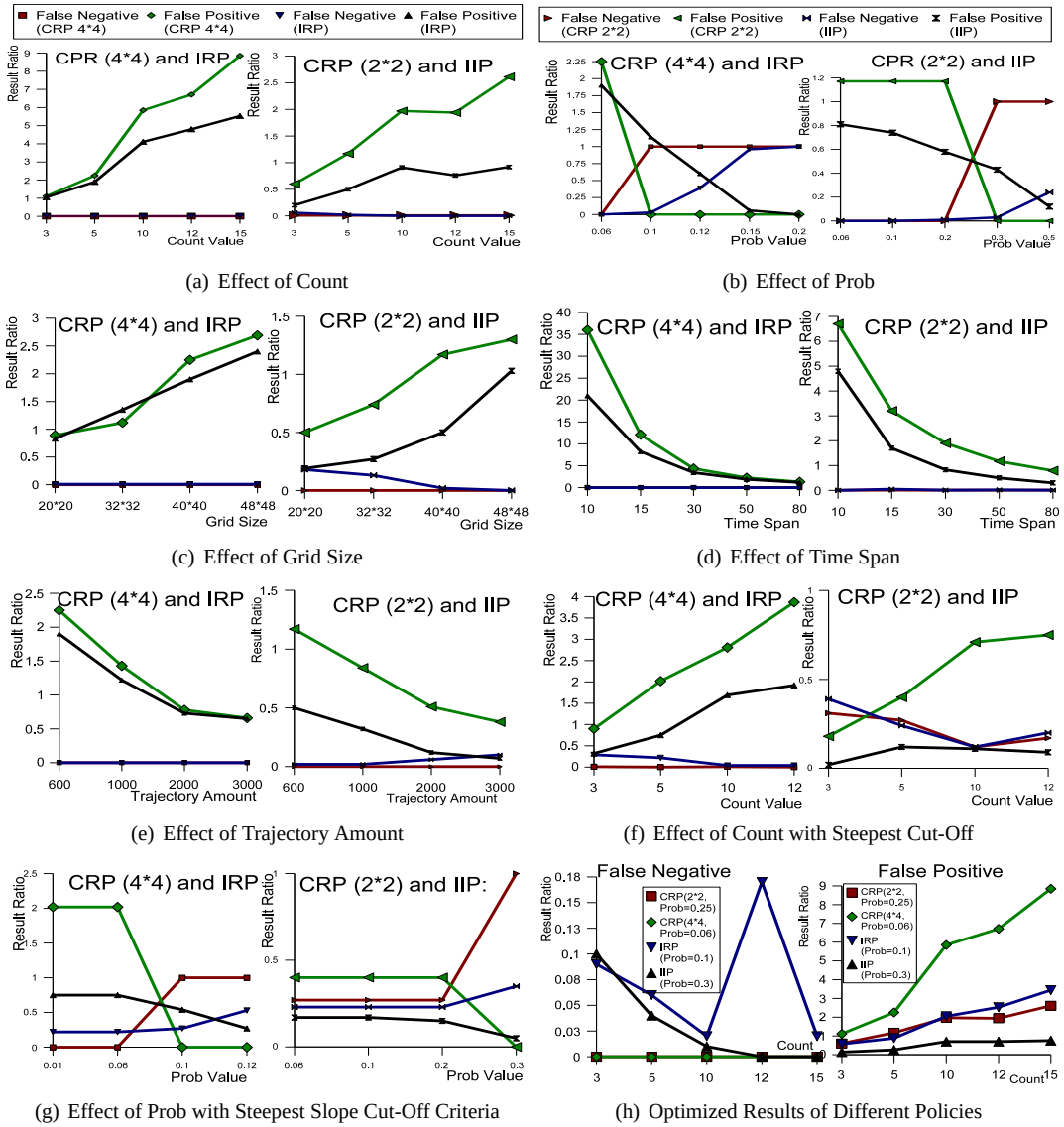
5 Evaluation

To evaluate the algorithms, we use Brinkhoff's network-based generator of moving objects [3] to generate trajectories on the Oldenburg network. We use integer as unit time instance and set the whole time period from 0 to 100, and generate 600 to 3000 trajectories. To capture the real world time span between two consecutive time instances, we calculate, for all the trajectories, the average distance between every two subsequent reported locations. The average distance is $234.96m$, which is about 14 seconds travel time for a $60km/hour$ moving object. Thus, the actual time span between two consecutive time instances is about 14 seconds. The default time span for all the queries are 50 time instances.

To implement the grid-based solution, the anonymization grid is generated based on the minimum bounding rectangle (MBR) of the Oldenburg network. We assign each generated trajectory a randomly-chosen anonymization partitioning based on the grid.

The default grid is a 40×40 partitioning on the MBR of the Oldenburg network. Based on the Oldenburg network data, the size of each grid cell is $589m \times 672.9m$. In the experiments, we also tune the grid partitioning from 20×20 to 50×50 to observe the performance. We apply the three policies on the trajectories with the anonymization grid. To implement the CRP policy, we make two fixed partitionings, where each user has 2×2 or 4×4 grid cells. In the IRP policy, every user partition contains at most 4×4 grid cells. In the IIP policy, we set each moving object to use the anonymization partition (each partition contains at most 4×4 grid cells) that covers the start location of the moving object. After the object is out of this partition, it uses the lowest level of privacy so that each partition equals to a grid cell.

In the experiments, we are focused on evaluating the accuracy of the algorithms, i.e., the amount of false positives and false negatives. A false negative, is the error of not finding a pattern that does exist in the data. A false positive, is the error of finding a "pattern" that does not exist in the data. To compare the algorithms, we also apply the algorithms on an ideal case, where the partitioning of every user equals the anonymization grid, and use the results of this case as the evaluation target. Suppose the actual amount of dense grid cells or frequent routes is D, we collect the number of false positives P, false negatives N for every algorithm and report the ratio between these values and D, called the *false positive rate* (FPR) and *false negative rate* (FNR), respectively. The choice of these measures over the precision and recall measures used in information retrieval is because of conceptual simplicity. We relate different kinds of errors to the same reference set (D), as opposed to relating the same set of correctly retrieved patterns to the set of all true patterns (recall) and to the set of retrieved patterns (precision). Hence, more accurate results are characterized by lower error rates rather than by higher recall and precision. However, it holds that Recall=1-FNR and Precision=(1-FNR)/(1-FNR+FPR).



(a) Effect of Count

(b) Effect of Prob

(c) Effect of Grid Size

(d) Effect of Time Span

(e) Effect of Trajectory Amount

(f) Effect of Count with Steepest Cut-Off

(g) Effect of Prob with Steepest Slope Cut-Off Criteria

(h) Optimized Results of Different Policies

Figure 8: Experiments on Dense ST-area Query

5.1 Dense ST-area Query

In the experiments of *dense ST-areas* query, we tuned the *count* and *prob* values to observe the amount of false positives and false negatives. Experiments have also been conducted to test the effect of grid size, time span and amount of trajectories on the accuracy. As seen in Figure 8(a) to Figure 8(e), there are very few false negatives and the amount of false positives grows in certain cases. In particular, based on Figure 8(a), with the growth of *count* values, more false positives appear. With the experiment on the *prob* value (Figure 8(b)), it is possible to reach an optimal situation by tuning this *prob* value for each policy. For instance, the IRP policy has fewer false positives when *prob*$= 0.1$ and so has IIP when *prob*$= 0.3$. An observation from Figure 8(c) is that the amount of false positives grows with the grid size. Our explanation is that when the grid becomes denser, there are fewer really dense grid cells but the amount of dense cells found through the three policies does not decrease very much so that the reported ratio value becomes larger. In Figure 8(d) and Figure 8(e), we observe that increase of the time span and the amount of trajectories reduces the amount of false positives for all the policies.

To test how the steepest slope cut-off criteria influences the algorithms, we tune the *count* and *prob* values to observe the amount of false positives and false negatives on the different policies. As illustrated in Figure 8(f) and Figure 8(g), the cut-off criteria decreases the amount of false positives but, compared to the same settings in Figure 8(a) and Figure 8(b), brings more false negatives. Thus, considering all the parameters for the three policies, we have the following **recommendation** settings for the dense ST-area query:

The IIP is the most effective policy for doing dense ST-area query with privacy protection. The second and third best choice is the CRP policy with 2×2 *partitioning and the IRP policy. For all the policies, certain optimal situation on the amount of false positives and false negatives can be reached by tuning the prob value. To increase the time span and amount of trajectories will improve the performance of all approaches.*

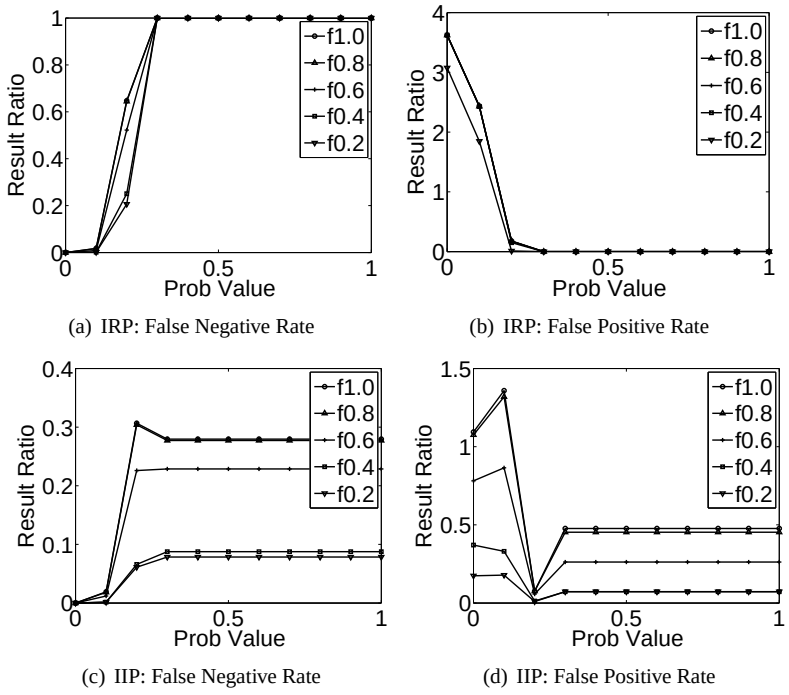
Based on the **recommendation**, we have an experiment to compare the different policies with their optimal settings. We increase the amount of trajectories to 1000 and use optimal *prob* values for each policy. Figure 8(h) presents the results. The CRP policy with each partition containing 2×2 cells and the IIP policy shows the most promising performance. These two policies guarantee a precision level that makes them useful for most applications.

5.2 Frequent Route Query

In the simulated moving object data sets, objects move between a source and a destination location, obeying the limitations of the road network and traffic conditions. However, there is little regularity in the selection choice for the source and destination locations. Hence, we expect to find longer frequent routes, only if there are a large number of objects on the road network at the same time. Thus in the frequent route query experiments we use the data sets that contain 3000 objects, and a query time span of 120.

For a frequent route query, a pattern is a maximal set of spatio-temporal grid cells. Since two patterns are very unlikely to be exactly ythe same, to be able to evaluate false positive and false negative errors, we first need to define an approximate matching criteria between two patterns. We say that a pattern p_1 is *f-contained* in a pattern p_2, if at least f fraction of the items in p_1 are also in p_2.

In the experiments, for `min_count` $= 5$, we evaluate the accuracy of the three anonymization policies for various prob values and f-containment. The results of the experiments for the IRP and IIP policies are shown in Figure 9. Results for the CRP (2 x 2) policy have been excluded from the figure, because the varied parameters had little to no effect on the results. For all f-containments for `min_prob` ≥ 0.3 the false positive rate was 0, while the false negative rate was 1. For `min_prob` < 3, the opposite was true. The reason for this behavior is that in the CRP (2 x 2) policy all grid cells have a location probability of 0.25. The false positive rate of 0 for `min_prob` ≥ 0.3 is due to the fact that under this setting no patterns (out of which none are false) are found in the anonymized data. A similar behavior is observable for the IRC policy, except the false positive rate, for sufficiently low `min_prob` values, is between 3 and 3.5. This is due to the fact the the average partition size for the IRP policy is 9 cells, which gives rise to a lot more patterns. As expected, the best performance is obtained by the IIR policy, where for `min_prob` $= 0.3$ false negative is 0.37 and the false positive rate is 0.47. *The experiments show that for low* `min_prob` *values the IRP policy, but even more so, the IIR policy provides accurate and useful frequent route mining results.*



(a) IRP: False Negative Rate (b) IRP: False Positive Rate

(c) IIP: False Negative Rate (d) IIP: False Positive Rate

Figure 9: Experiments on Frequent Route Query

6 Conclusions and Future Work

Motivated by the possible loss of location privacy for LBS users, this paper proposed a general grid-based framework that allowed user location data to be anonymized. Thus, privacy is preserved, but interesting patterns could still be discovered. The framework allowed users to specify individual desired levels of privacy and developed three policies for implementing that. Privacy-preserving methods were proposed for two core data mining tasks, namely *finding dense spatio–temporal regions* and *finding frequent routes*. An extensive set of experiments evaluated the methods and showed that the framework still allowed most patterns to be found, even when privacy was preserved.

Future work will be along three paths. First, we will further investigate the *Multi-Grid* approach as it offers a direction for getting more detailed data mining results without violating the privacy. Second, in addition to the CRP, IRP and IIP policies, it is possible to develop more policies for creating anonymization rectangles suitable for different real world situations. Third, since the grid-based solution can be seen as a simple and general framework for privacy preserving data mining on moving object trajectories, we will extend this framework to support more kinds of spatio-temporal data mining algorithms.

References

[1] R. Agrawal, T. Imilienski, and A. Swami. Mining Association Rules between Sets of Items in Large Databases. In *Proc. SIGMOD*, pp. 207–216, 1993.

[2] R. Agrawal and R. Srikant. Privacy-Preserving Data Mining. In *Proc. SIGMOD*, pp. 439–450, 2000.

[3] T. Brinkhoff. A Framework for Generating Network-Based Moving Objects. In *GeoInformatica*, (6)2, pp. 153–180, 2002.

[4] D. Burdick, M. Calimlim, and J. Gehrke. MAFIA: A Maximal Frequent Itemset Mining Algorithm for Transactional Databases. In *Proc. ICDE*, 2001.

[5] A. Civilis, C. S. Jensen, and S. Pakalnis. Techniques for Efficient Road-Network-Based Tracking of Moving Objects. In *TKDE,* 17(5), pp. 698–712, 2005.

[6] M. Ester, H. P. Kriegel, J. Sander, X. Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proc. KDD,* pp. 226–231, 1996.

[7] M. Gruteser and D. Grunwald. Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking. In *Proc. USENIX Mobisys,* 2003.

[8] B. Gedik and L. Liu. Location Privacy in Mobile Systems: A Personalized Anonymization Model. In *Proc. ICDCS,* pp. 620–629, 2005.

[9] G. Gidófalvi, T. B. Pedersen. Spatio-temporal Rule Mining: Issues and Techniques. In *Proc. DaWaK,* pp. 275–284, 2005.

[10] G. Gidófalvi, T. B. Pedersen. Mining Long, Sharable Patterns in Trajectories of Moving Objects. In *Proc. STDBM,* pp. 49-58, 2006.

[11] M. Hadjieleftheriou, G. Kollios, D. Gunopulos, and V. J. Tsotras. On-Line Discovery of Dense Areas in Spatio-Temporal Databases. In *Proc. SSTD,* pp. 306–324, 2004.

[12] C. S. Jensen, D. Lin, B. C. Ooi, and R. Zhang. Effective Density Queries on Continuously Moving Objects. In *ICDE,* 2006.

[13] M. F. Mokbel, C. -Y. Chow, and W. G. Aref. The New Casper: Query Processing for Location Services without Compromising Privacy. In *Proc. VLDB,* pp. 763–774, 2006.

[14] L. Sweeney. K-Anonymity: A Model for Protecting Privacy. In *Int. J. of Uncertainty, Fuzziness and Knowledge-Based Systems,* 10(5), pp. 557–570, 2002.

[15] Y. Tao, G. Kollios, et. al. Spatio–Temporal Aggregation Using Sketches. In *Proc. ICDE,* pp. 214–226, 2004.

[16] I. Tsoukatos and D. Gunopulos. Efficient Mining of Spatiotemporal Patterns. In *Proc. SSTD,* pp. 425–442, 2001.

[17] V. S. Verykios, E. Bertino, et. al. State-of-the-art in Privacy Preserving Data Mining. In *SIGMOD Record,* 33(1), pp. 50–57, 2004.