Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2017) 25: 1612 – 1628
© TÜBİTAK
doi:10.3906/elk-1511-190

Assignment as a location-based service in outsourced databases

Ahmet Salih BÜYÜKKAYHAN, Taflan İmre GÜNDEM*
Department of Computer Engineering, Faculty of Engineering, Boğaziçi University, İstanbul, Turkey

Received: 21.11.2015 • **Accepted/Published Online:** 05.06.2016 • **Final Version:** 29.05.2017

Abstract: Location-based services provide opportunities to organizations for tracking and utilizing their resources. Moreover, some corporations prefer to outsource their database management services. In the literature, privacy preservation techniques in outsourced spatial databases for several query types, such as nearest neighbor, K-nearest neighbor, and proximity search, have been examined. In this paper, we present an efficient application of the capacity and coverage-constrained assignment query on outsourced spatial databases. In the proposed technique, we introduce a novel spatial transformation strategy (square spiral encoding) to achieve privacy efficiently for approximate assignment query results. To process the assignment query, we use the parallel auction algorithm. We implemented and tested the assignment query with the proposed technique for correctness and efficiency.

Key words: Location anonymization, location-based services, location privacy, assignment query, outsourced databases

1. Introduction

There are various location-based services for corporations. Outsourcing database services are becoming increasingly popular [1–3]. In outsourcing database services, privacy preservation is an important issue. In the literature, there exist various location privacy-preserving techniques based on anonymization, transformation, encryption, and private information retrieval (PIR). Anonymization techniques [4–6] depend on a trusted third party. Traditional encryption-based techniques [7–9] suffer from high computation and communication costs for spatial query processing. Finally, PIR-based techniques [10,11] suffer from high computational complexity and communication costs.

In this paper, our main contribution is a novel and efficient privacy-aware technique that does not need a trusted third party. We name the technique 'square spiral encoding'. The square spiral technique transforms the data before sending them to the outsourced database. This technique enables the service provider to change the amount of location privacy and the accuracy of the result by changing a parameter of the transformation key. Next, the transformed values are sent to the outsourced database. Since the outsourced database server is not aware of the real locations or the transformation keys, the data privacy is preserved. The outsourced database server processes the queries blindly over the transformed data.

The private information retrieval techniques proposed for the nearest neighbor (NN) [11] and K-nearest neighbor (KNN) [10] methods cannot be readily applied to assignment queries. A recent work [12] proposed using both Moore curves and asymmetric cryptography for KNN queries. This technique also suffers from high communication and computation costs and is, again, only suitable for KNN queries. Assignment query processing requires global data, since a small local change could change the result entirely.

*Correspondence: gundem@boun.edu.tr

In this paper, we apply our proposed technique to the capacity and coverage-constrained assignment query on spatial databases. In our application, we use a corporate location-based service that assigns corporate parking lots at different locations to corporate vehicles by considering all parking requests and all the available spaces in the parking lots. This service applies the capacity-constrained assignment to reduce the overall costs (distance traveled by vehicles, total amount of gasoline used, etc.). Since it may not be economical to assign a parking lot to a vehicle that is far away, the system also considers the maximum coverage distance for a parking lot.

In the next section, we present the relevant literature. In Section 3, we explain our implementation of the assignment query for outsourced databases. In Section 4, we present the experimental results for testing the performance of the proposed methodology, and in Section 5 we give the conclusions.

2. Related work

In this section, we first summarize the scientific literature that is related to assignment query and the algorithms for solving the assignment problem. We then review the location privacy-aware techniques for spatial databases.

2.1. Assignment query

In the literature, the assignment problem is also recognized as a bipartite matching problem [13,14]. The goal is to assign N items to M other items by maximizing or minimizing an objective function. If assigning an item to another item has a cost, then the problem is referred to as weighted assignment problem. The Hungarian [15,16] and the successive shortest path (SSP) [17] algorithms are the most well-known algorithms for solving the assignment problem. The lowest worst-case performance of these algorithms is O (V (E + V log V)), where V is the number of vertexes and E is the number of edges. The most efficient algorithm for sparse flow networks is the auction algorithm [18]. It is shown that the worst-case execution time of the auction algorithm is O (N A log(N C)), where N is the total number of bidders, A is the total number of all person–object pairs, and C is the maximum benefit [18]. Furthermore, the auction algorithm could be easily implemented in parallel, synchronously, or asynchronously [19].

Konan et al. [20] proposed the assignment query to solve the assignment problem in moving object databases. The proposed query should be used only for one-to-one matching, and the approximation algorithm produces, in the worst case, 33% higher total cost when compared to the exact algorithms. Other authors [21] extended the assignment query by including capacity constraints. In [21], the authors focused on static data stored on the disks, assuming that the services have infinite coverage and solve the problem for fully connected graphs. A recent study [22], presenting coverage and capacity-constrained optimal assignment query, used a more realistic scenario, where points of interest have limited service regions. However, it used the SSP algorithm with QuickMatch enhancements [23] to find the optimal assignment. It was experimentally shown that the QuickMatch algorithm is slower than the auction algorithm [23].

None of the approaches mentioned above consider the location privacy or outsourced databases. To the best of our knowledge, there is no previous work that processes the assignment query blindly over the transformed data.

2.2. Location privacy

2.2.1. Anonymity-based approaches

Identity protection and identification of seemingly anonymous users have been examined in various studies in the literature [24,25]. Location [26,27] and trajectory [28] K-anonymity is a major privacy preservation paradigm for the protection of identity. K-anonymity protects the location information of a user by hiding it among other K-1 user locations (which may include dummies). In the literature, many approaches based on cloaking, K-anonymity, and dummies have been proposed to reduce the possibility of identifying a user's location [4,5,24,26–29].

Most of the anonymization approaches rely on a trusted intermediary party. This means that each query should be directed to an anonymizer. Anonymizers increase the implementation and maintenance costs; additionally, they may become a potential point of failure or security breach. Instead of using an anonymizer, queries could be anonymized among the users in a decentralized manner [4,29]. In such an approach, each user should trust all other users in the system. Therefore, anonymization-based approaches require a trusted party: either a trusted intermediary or a trusted group of users. In general, any increase in the degree of location privacy reduces service quality or accuracy. Furthermore, K-anonymity does not always work. For instance, in an unpopulated area, the size of the cloaked region can be very large in order to include other K-1 users. Even worse, there may not be enough subscribed users to the service to provide K-anonymity. On the other hand, anonymization-based techniques are considerably efficient in terms of communication and computation overhead [26].

2.2.2. Cryptographic-based approaches

Encryption-based approaches utilize secure multiparty computation schemes in order to blind the untrusted party (i.e. the server or another user). The main advantage of encryption is its strong privacy guarantee. Encryption algorithms use one-way functions, and it is almost impossible to decrypt the data without the key. Cryptographic techniques do not suffer from privacy leaks of anonymization and transformation. The challenge here is to execute the query efficiently over the encrypted data. Hacıgümüş et al. [7] proposed the bucketing technique to overcome this problem. However, the communication and computational costs make existing approaches unsuitable for location-based services in spatial databases. For example, in [8], the distance between a query point and each point of interest (POI) must be both transferred and computed on the client.

2.2.3. PIR-based approaches

The aim of private information retrieval (PIR) is to get the ith record from an untrusted database, without revealing i to the database. PIR requires private spatial indexes to be used by PIR operations for efficient spatial query processing; therefore, the PIR scheme guarantees privacy. PIR methods can be divided into hardware-based [10] and computational [11] methods. In general, R partitions the location space in order to index the location data. In the literature, there are various one-dimensional or two-dimensional portioning techniques to execute NN queries. For instance, [11] proposed Varonoi diagrams to support exact NN queries. Similar to cryptographic approaches, PIR-based approaches do not suffer from the privacy leaks of the anonymization- or transformation-based approaches. However, existing PIR protocols are still expensive in terms of computation and communication costs, and they require a significant amount of resources. It has been argued that sending the entire database to the client is more efficient when compared to the cost of privately retrieving items from the database using existing PIR techniques [30].

2.2.4. Transformation-based approaches

Transformation-based approaches transform the queries and data to prevent the database server from learning the location information of the user. In [31], Hilbert space-filling curves are utilized as one-way transformations.

A transformation is one-way if the data can be easily transformed into single direction and cannot be reverse-transformed in a reasonable time period without any additional information [31]. Both the users and POIs are encoded using this transformation; consequently, the queries are evaluated in the transformed space. The database server provides the transformed query results to the users, and then the users reverse the transformation efficiently using a trapdoor. This trapdoor information is only known to the user and is hidden from the database server. Although this method provides approximate results, the query efficiency is very high. Unfortunately, the proposed transformation function does not apply to queries other than NN and KNN. Lin et al. [32] proposed a distance-preserving transformation schema, but this requires trusted agents to transform the data or query. A recent work [33] proposed a framework called SpaceTwist. This blinds the database server by incrementally retrieving POIs from a fake location, which is called the anchor point. In [34,35], the space is partitioned, and then a linear distinct transformation is applied to each partition. The proposed techniques are applied only to range and KNN queries of the outsourced data and filter the false positives in the clients. Therefore, these techniques incur high communication and computation costs to the clients. Transformation-based approaches generally do not require a trusted third party for query processing. Furthermore, they can utilize existing indexes to use nonprivacy-aware database servers, which makes them readily applicable to existing systems. The main challenge for transformation-based techniques is maintaining distance properties while preventing reverse transformations by the untrusted database server or an attacker. For this purpose, we propose a novel transformation approach to encode the space by utilizing square spirals as one-way transformations. Our technique does not use the processing power of the clients for query evaluation or for filtering the false positives. Only a single result is sent to the client in order to minimize the communication and computation costs.

2.2.5. Square spirals and HG-tree

Square spirals start from the center of the squares and traverse all the outer squares while moving away. After each loop or cycle, the spiral level increases by one. Square spirals are geometric entities used in the scientific literature for various purposes. Our use of square spirals for privacy preservations and the associated algorithms is unique, as explained in Section 3.

To store square spiral values, we use the SS-tree, as explained in Section 3. The SS-tree is a one-dimensional R-tree. In the literature, we have such a tree in the Hilbert R-tree to store the largest Hilbert value (LHV). A slight modification of the Hilbert R-tree is the HG-tree [36], which stores two types of Hilbert values, the LHV and the smallest Hilbert value (SHV). The SS-tree is similar to the HG-tree, but stores SS values rather than Hilbert values.

3. System model

In this section, we present the proposed privacy preservation technique and its application in outsourced capacity and coverage-constrained assignment query processing when assigning vehicles (users) to the parking lots of an organization.

In Figure 1, we have the event trace diagram of the complete system. Here rectangles represent the various components of our proposed system. Directed horizontal lines represent the messages passed and their directions. Time increases as you move down in the figure. In the offline phase, the locations and capacities of all parking lots are transformed using two-dimensional to one-dimensional mapping with a transformation key. The transformed data are then uploaded to the outsourced database management system (processing the assignment query execution), which does not know the decryption key. All users (e.g., vehicle drivers) have tamper-resistant devices (clients), which store the decryption key but do not know the key themselves. The users send the encrypted location, their ID, and their coverage region to the outsourced database management system to request parking lots. For identity privacy, the clients may use random or fake ID numbers, as long as they are unique in the system. The POIs send the encrypted location and available capacity updates to the outsourced database management system; only updated values are sent to the system to reduce the communication cost. The users that are located close to each other may have the same transformed location. In this case, those that have the same transformed location are presented as a single user, which has a capacity that is equal to the number of users in this location. The assignment query is executed when needed in the outsourced database management system to assign users to the POIs. Each assigned user receives the encrypted location of the assigned POI. The client then decrypts the encrypted location of the POI and displays it to the user.

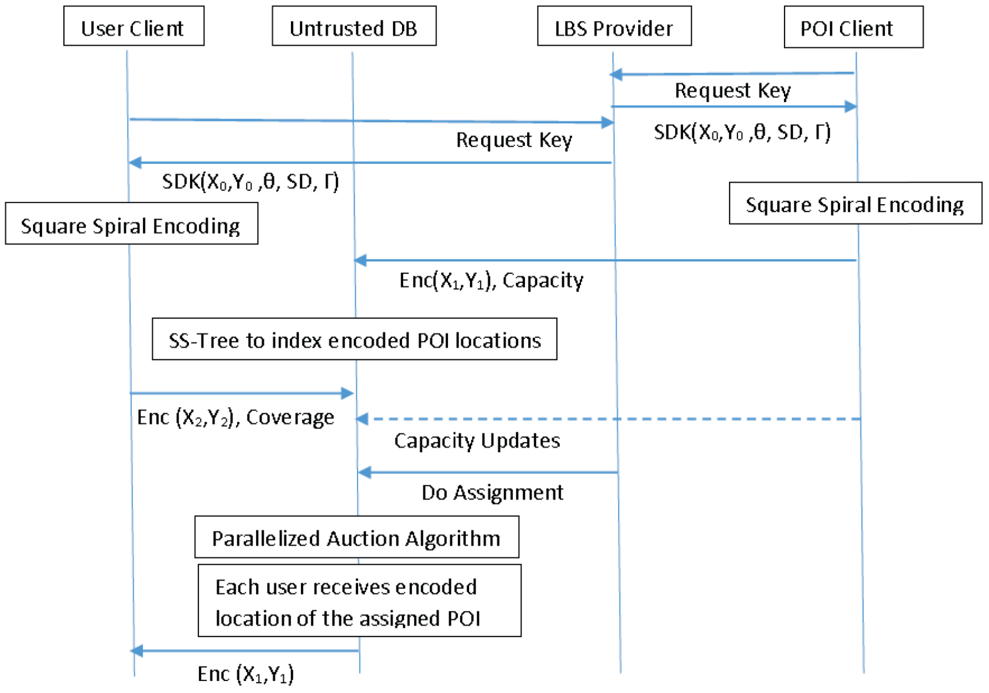


Figure 1. System event trace diagram.

3.1. Assignment query

In this section, we will first define the problem associated with the assignment query.

Problem definition: We have a set of users $U = \{u_1 \ldots u_n\}$ and a set of POIs $P = \{p_1 \ldots p_m\}$. The cost of assigning $u_i \in U$ to $p_j \in$ P is $COSTu_ip_j$, which is the Euclidean distance between u_i and p_j. For each u_i and p_j, the coverage radius is equal and represented as R. The capacity of a user u_i is Cu_i and the capacity of a POI p_j is Cp_j. As explained earlier, if there are users that are in the same encrypted location, they are represented as a single user. The capacity of such a virtual user is equal to the total number of users in its encrypted location.

Given:U, P = n users and m POIs, Cu_i, Cp_j = capacity of user u_i and POI p_j, and R = maximum coverage radius for the POI or user.

Calculated: $COSTu_ip_j$ = cost of assigning a single capacity from user u_i to POI p_j. Cu_ip_j = assigned capacity from u_i to p_j. A = set of assigned pairs.

Objective:

Minimize the total cost: $\sum\limits_{\forall(u_i,p_j)\in A}^{Cu_ip_jCOSTu_ip_j}$

Maximize the assigned capacity: $\sum\limits_{\forall(u_i,p_j)\in A}^{Cu_ip_j}$

Subject to:

User capacity limit: $\sum\limits_{j=1}^{m} Cu_ip_j \leq Cu_i \forall(u_i,p_j) \in A$ (1)

POI capacity limit: $\sum\limits_{i=1}^{n} Cu_ip_j \leq Cp_j \forall(u_i,p_j) \in A$ (2)

Assigned capacity limit: $0 \leq Cu_ip_j \leq \min(Cu_i, Cp_j)$ (3)

Coverage constraint: $0 \leq COSTu_ip_j \leq COST_R$ (4)

Coverage radius is positive: $0 \leq R$ (5)

This problem is a variant of the minimum cost flow (MCF) problem, which contains only integer flows. Due to the coverage constraint, the bipartite graph becomes a sparse graph. The solution to this problem is the answer to the capacity-constrained assignment query, which we refer to as 'assignment query' in the remainder of this paper. The proposed assignment query for the outsource databases is represented as ASSIGN (U = set of users, P = set of POIs, and A = output).

When the assignment query is started using the current database, the input candidate pairs for the assignment are determined with respect to the coverage and capacity constraints. The assignment cost between each candidate is calculated without revealing the locations. The following sections will provide further explanation.

3.2. Square spiral encoding

For the square spiral (SS) encoding, we identify a two-dimensional to one-dimensional mapping function as a one-way transformation and define the parameters of our mapping function as the trapdoor so that the query results are decrypted quickly for the users. The trapdoor will only be provided to the clients to reverse the encoded query result and retrieve the response in its original form. The users and POIs use the transformation function to report their encrypted locations to the outsourced database server. Figure 2 illustrates the SS and sequential encoding according to the traversal order of each square.

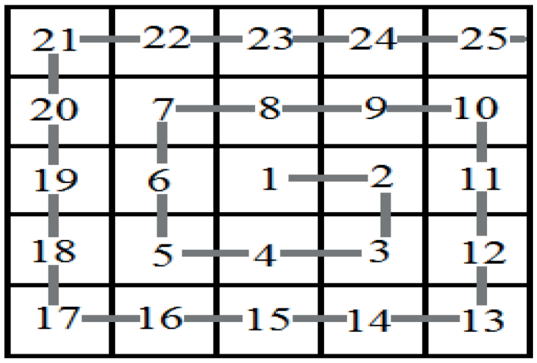
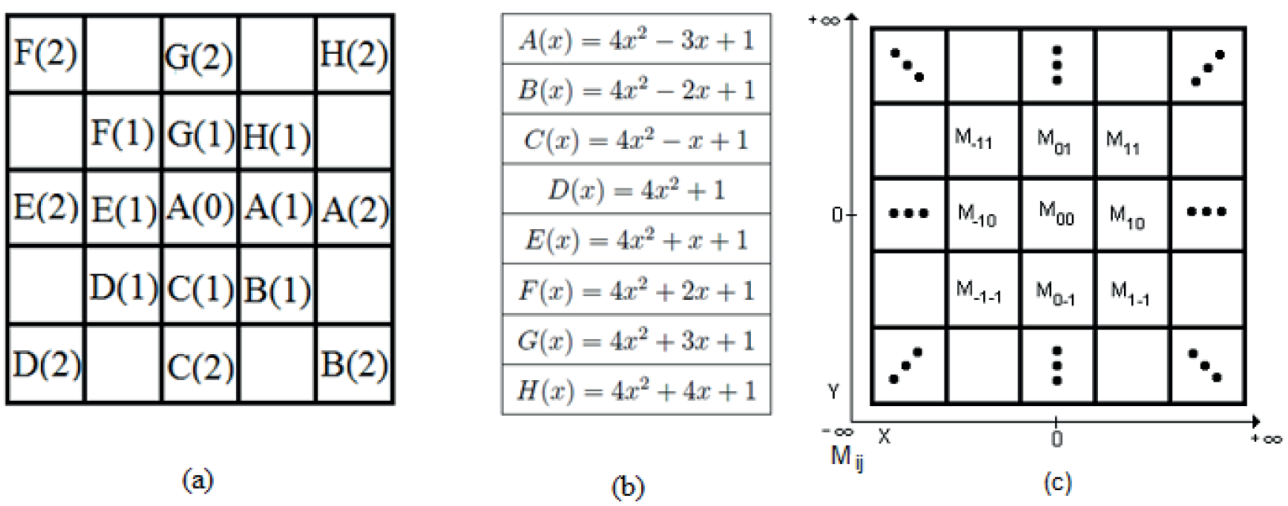


Figure 2. Square spiral space encoding.

In the literature, the spirals take the form of curves, which start from the center and draw progressively farther away as they revolve around the center. Square spirals start from the center of the squares and traverse all the outer squares while drawing away. After each loop or cycle, the spiral level increases by one. This level is similar to the radius of a square, which is centered at the starting point of the square spiral.

A square spiral where only prime numbers are visualized is called a Ulam spiral. Ulam used a square spiral to visualize the prime numbers and discovered that they form straight lines in some areas [37]. For example, $4x^2 - 2x + 41$ forms a straight line in a square spiral and is equivalent to Euler's prime generating polynomial $x^2 - x + 41$ when x is replaced by $2x$.

The most interesting feature of square spirals is how they preserve the relative distance after transformations with satisfactory values when used in location-based services. This property is well suited to our problem, since we are interested in addressing the assignment query in a transformed space. Furthermore, there is a relationship between the square spiral values (SS-values) and their geometric layout. The encoded values of the squares that have the same direction according to the starting point satisfy the same function as shown in Figure 3a. The equations of those eight functions for eight different directions are given in Figure 3b.



$A(x) = 4x^2 - 3x + 1$
$B(x) = 4x^2 - 2x + 1$
$C(x) = 4x^2 - x + 1$
$D(x) = 4x^2 + 1$
$E(x) = 4x^2 + x + 1$
$F(x) = 4x^2 + 2x + 1$
$G(x) = 4x^2 + 3x + 1$
$H(x) = 4x^2 + 4x + 1$

Figure 3. Square spiral functions: a) geometric layout, b) functions, c) M$_{ij}$ and coordinates.

Let M$_{00}$ be the starting square and the center of the square spiral, as illustrated in Figure 3c. Let other squares be labeled as M$_{ij}$ with respect to M$_{00}$. Subscript i is the horizontal index and j is the vertical index in the coordinate system.

$$M_{ij}(SS-\text{value}) \;=\; \begin{cases} A(0) = B(0) = \ldots = H(0) = 1 & i = j = 0 \\ A(i) & j = 0,\ 0 < i \\ B(i) & i = -j > 0 \\ C(-j) & i = 0, 0 > j \\ D(-i) & i = j < 0 \\ E(-i) & j = 0, 0 > i \\ F(j) & i = -j < 0 \\ G(j) & i = 0, 0 < j \\ H(j) & i = j > 0 \end{cases}$$

For example, squares encoded 2 and 11 in Figure 2 are in the right direction of the starting square encoded 1 and satisfy the function $A(x) = 4x^2 - 3x + 1$, where x is the level of the square spiral. The level of a square spiral is defined as $x = \max(|\mathrm{i}|, |\mathrm{j}|)$ for the square M_{ij}. Number 2 is in the first level and is obtained by substituting x with 1 in function $A(x)$. Similarly, number 11 is in the second level and is found by substituting x with 2 in function $A(x)$. Figure 3 shows the square spiral functions and the geometric layout.

Square spiral transformation is a one-way function and cannot be reversed without the decryption key. The square spiral starting point (X_0, Y_0), spiral orientation θ, spiral direction (SD), and spiral scale factor Γ jointly form the transformation key. We name the key 'spiral decryption key' (SDK), where SDK = $\{\mathrm{X}_0, Y_0, \theta, SD, \Gamma\}$. Thus, an attacker who does not know this key must try all possible combinations of the square spiral parameters. The reverse transformation of square spiral values to obtain the original points is computationally very difficult, as shown in Theorem 1.

Theorem 1 *If each square spiral parameter is defined by p number of bits, then the algorithm complexity of an exhaustive search to reveal the transformation key is $O(2^{4p})$.*

Proof The approach we use in this proof is similar to that given in [6] for the Hilbert curve transformation. We first consider the square spiral's starting point (X_0, Y_0). The exact values of both X_0 andY_0 are required to determine the key accurately. However, finding the exact values of both X_0 and Y_0 in a continuous space is theoretically impossible.

On the other hand, in practice, the attacker could generate a very fine grid to guess X_0 and Y_0. Let us assume that an attacker's best guess (X_0', Y_0') is located very close to (X_0, Y_0). In such a case, it is possible to generate the same SS-values for the known POIs using (X_0', Y_0') as the starting point. However, the entire coordinate space should be exhaustively searched to obtain a very close approximate value of the starting point. If a location is defined by p bit coordinates, then 2^p different values can be produced on each axis. This means $2^p \times 2^p$ values should be evaluated by the attacker to find a very close approximate value of the starting point. Finding the square spiral orientation (θ) is similar to finding the starting point. If θ is defined by q bits, 2^q different possible values of θ could be generated as a square spiral orientation. The square spiral scale factor Γ is a continuous number in a range from 0 to 1. Similarly, when $\Gamma$is defined by r bits, 2^r values should be generated between 0 and 1 to guess Γ. The square spiral direction determines the traversal order of the squares. For N different square spiral direction values, $2^p \times 2^p \times 2^q \times 2^r \times N$ elements become candidates. At first, $N << 2^p$, and if each parameter is represented by p bits, then the brute force search complexity is $O(2^{4p})$. □

Theorem 2 *The difference between the largest SS-value of a level n and the largest SS-value of one level below $(n-1)$ is $8n$.*

Proof The largest SS-value of the square spiral can be formulated as $(2n+1)^2 = 4n^2 + 4n + 1$, for each $n = 0, 1, 2....$ The largest SS-value in the lower level is calculated for $n = n - 1$, and the difference gives the result $(2n+1)^2 - (2(n-1)+1)^2 = (4n^2 + 4n + 1) - (4n^2 - 4n + 1) = 8n$. As is clearly seen, the difference between the largest SS-values is $8n$. This is used in the algorithm to calculate the distance to the center of the square spiral and the angle with the center point for any given SS-value. □

3.3. Square spiral encoding algorithm phases

As explained earlier, the complexity of a brute-force attack to find the transformation key of the square spiral is $O(2^{4p})$, where each parameter is represented by p number of bits. The square spiral encryption is strong enough; however, the SDK should be updated regularly to increase security and prevent brute force attacks. When clients obtain these parameters, they start to construct the square spiral from (X_0, Y_0). After the square spiral is constructed, each visited point can be encrypted using the encryption function E, which maps points (P_x, P_y) to SS-values (SS-value) $= E(P_x, P_y)$. Similarly, any received data can be decrypted using the decryption function D, which returns the geometric center point (P_x, P_y) of the region, which is represented by the SS-value $(P_xP_y) = D$(SS-value). The outsourced database management system does not have any information about the square spiral parameters. The encryption and decryption operations can be performed only in clients using the SDK. Only the SS-values of the POIs are indexed for efficient retrieval. For this reason, the size of the database is not dependent on the number of users, but rather only on the number of POIs. The SS-values of the POIs are stored in a B+-like tree due to efficient insertion, retrieval, and removal capabilities. In the proposed structure, all keys are intervals, as in an HG-tree [36]. We adapted the HG-tree to square spirals to store SS-values and named it the SS-tree.

The SS-tree is composed of leaf and internal nodes. A leaf or internal node may have a variable number of entries. However, the maximum number of entries is constant. Leaf node entries are in the form of (oid, SS-value), where oid is the primary key in the database and the SS-value is the square spiral encoded value for the spatial data. Internal node entries are in the form of (ptr, I), where ptr points to the child node, and I is the smallest interval on the square spiral that covers all regions of the lower node and is called the minimum bounding interval (MBI). In order to define the MBI, two SS-values are required as the start and end points: I = (SS-value1, SS-value2), where SS-value1 and SS-value2 are the start and end points, respectively.

Figure 4 illustrates the SS-tree data structure and geometric layout. The root node of the SS-tree is node n, which has two internal child nodes, n_1 and n_2. Node n_1 contains a pointer to the inner leaf nodes and interval I_2, which starts with the minimum SS-value (20) and ends with the maximum SS-value (23) of the child leaf nodes. Each leaf node contains an oid and the corresponding SS-value.

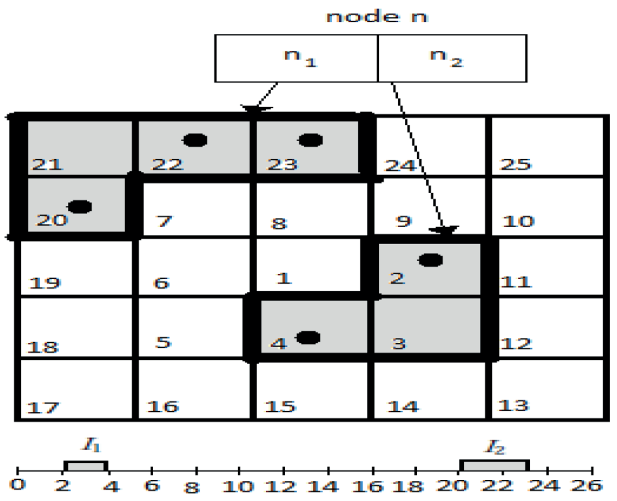


Figure 4. SS-tree data structure.

The online query processing phase has three steps: determining candidate pairs, calculating the assignment costs, and assigning users to POIs. At first, the outsourced database management system determines the POIs that have a user in their coverage region by searching the SS-tree index. If a POI is in the coverage region of a user, then this user and POI is an assignment candidate pair. Some users may not take place in a candidate pair if there is no POI in their coverage. After the candidate pairs are determined, the next step is to calculate the assignment costs of the candidate pairs. Various cost computation algorithms are used in an assignment query. In our implementation, we use the Euclidean distance between the user and POI as a cost function. In order to determine the Euclidean distance between SS-values, the radius and angle to the center point must be computed first. Therefore, the outsourced database management system computes the radius and angle to the center of the spiral for the POIs and users. The database is able to perform this computation for any given SS-value without any knowledge related to SDK = $\{X_0, Y_0, \theta, SD, \Gamma\}$, using the following Algorithm 1. Please note that the radius is not the distance to the center but rather the level that the square spiral value belongs to.

Algorithm 1 Calculate the radius and angle of a given SS-value.

```
1: calculateRadiusAndAngle (x = SS-value) {
2: if (x $\leq$ 1) then return (0, 0);
3: y = 1; radius = 1;
4: while $x > y$ loop $y+ = 8\times$ radius; radius++; end loop;
5: angle = 45 $-(y - x)\times$ (45 / radius);
6: return (radius, angle);    }
```

When the angle and radius of two SS-values are known, then the distance is calculated using the cosine theorem. In a triangle, if we know the length of two sides and the angle between them, then we can calculate the length of the third side using the cosine theorem.

In Figure 5, the radius of point A is R_A and the radius of point B is R_B. The angle between the central x-axis and $|OA|$ is $\alpha + \beta$, where β is the angle between the central x-axis and $|OB|$. The distance to the starting point O is determined with the radius and the angle with the central x-axis. This distance is represented by L and is maximized at the corners when the angle is 45°, 135°, 225°, and 315°.

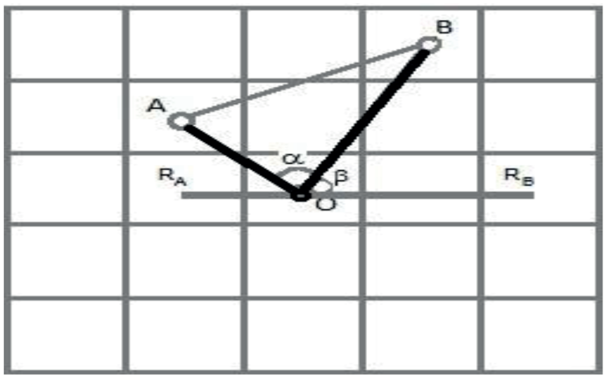


Figure 5. Distance calculation.

$$L = \begin{cases} \frac{R}{|\cos\theta|}, & if\ |\tan\theta| \le 1 \\ \frac{R}{|\sin\theta|}, & if\ |\tan\theta| > 1 \end{cases}$$

Distance D between points A and B is calculated with the cosine theorem as $D(A,B)=\sqrt{L_A^2+L_B^2-2L_AL_B\cos\alpha}$.

$$\sqrt{\left(\frac{r_1}{\cos(A)}\right)^2+\left(\frac{r_2}{\cos(B)}\right)^2-2\left(\frac{r_1}{\cos(A)}\right)\left(\frac{r_2}{\cos(B)}\right)\cos(|A-B|)}$$

For example, in order to find the distance between SS-value 3 and SS-value 13, the calculateRadiusAndAngle function is executed with arguments 3 and 13 and it returns (1, –225) and (2, –225), respectively. The distance between them, according to the cosine theorem, is 1.

Once we get the assignment costs, encrypted locations, and identities of the users and POIs, the auction algorithm computes the optimal assignment in parallel. The details of the parallel auction algorithm are described in Section 3.5. At the end, we list the assigned pairs with the SS-values. Each client receives the corresponding POI identity and SS-value. SS-values are decrypted by the client using SDK as the trapdoor. Finally, users obtain the location of the POI without revealing their locations to the untrusted database server.

3.4. Approximation

The approximation is performed only in the transformation phase. The scale of the smallest square can be changed according to the square spiral parameters. The assignment algorithm always uses the center of the square in distance calculations. Thus, we are able to control the approximation with parameter Γ. The upper bound of the error in fully connected assignment graphs, where the assignment leaves unassigned users, if every POI is fully utilized, is given in [15]. In our case, we additionally consider the service coverage distance. Thus, the number of assigned pairs is at most A, where $A \le min(\sum_{\forall u\in U} C_u, \sum_{\forall p\in P} C_p)$.

Figure 6 illustrates the user or POI locations in the squares. As defined in Section 3.2, Γ is the scale factor of the square spiral, which is the length of each square. The diagonal of each square is fixed and equal to δ, which is equal to $\Gamma\sqrt{2}$. The approximation is a result of the square spiral encoding. This is because, after the encoding, the user or POI locations are represented at the center of the square and the square capacity equals the total capacity of the objects inside.

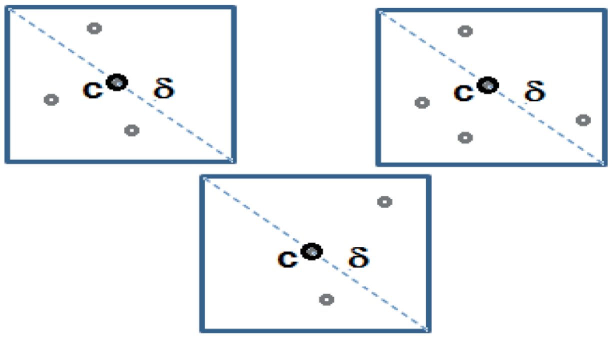


Figure 6. Approximation in square spiral encoding.

Let us assume that M_{opt} is the optimum assignment when all points are in their exact positions. Let us assume $\gamma = \min(\sum_{\forall u\in U} C_u, \sum_{\forall p\in P} C_p)$. For all assigned pairs, the assignment cost will increase at most by $\gamma.\delta$. This gives us $M' \le M_{opt} + \gamma.\delta$. However, M' may not be the optimal matching for U' and P', which gives us $M'_{opt} \leq M' \leq M_{opt} + \gamma.\delta$. When we replace all the points to the center of the small squares, the assignment cost for all assigned pairs will increase at most by $\gamma.\delta$. Thus, we come up with $M \leq M_{opt} + 2.\gamma.\delta$. As a result, the upper bound of the assignment error is bounded by $2.\gamma.\delta$.

3.5. Assignment algorithm

We say that a user is in the coverage of the POI if the user is in the square region centered in the POI and the radius is equal to or smaller than the coverage distance. In the real world, the coverage distances are limited, such that the bipartite matching graph turns into a sparse graph.

There are several algorithms proposed for capacity-constrained assignment problems, and the auction algorithm is the most efficient algorithm for sparse graphs. Furthermore, it may be adapted to efficient parallel or distributed execution. For these reasons, we adopted the auction algorithm to execute the optimal assignment query.

In the auction algorithm, there are bidders and objects. Each object has its own price, which increases with the bids. The benefit b_{ij} is the result of matching bidder $_i$ to object $_j$, with the aim of maximizing the total benefit $\sum_{k=1}^{n} b_{ijk}$. In the first round, the price of all objects is 0, and each bidder makes bids for all the objects. The winner of the round sets the new price of this object to make it less attractive to other competitors. In the next round, bidders increase their bids and compete for the object. If there is a new winner, then it sets the new price. This iterative process continues until all objects or bidders are assigned.

In our implementation, we defined the user as the bidder, the POI as the object, and benefit b_{ij} as the inverse value of the distance between the user and POI. Moreover, we considered the coverage constraint to ensure that users only make bids to the POIs that are in their coverage. We also eliminated the users and points that cannot be assigned in the beginning to increase efficiency by reducing the total number of elements in the assignment problem.

Our algorithm (Algorithm 2) has two phases. The first is the preparation phase, and the second is the bidding phase for the capacity-constrained parallel auction algorithm. The original algorithm maximizes the assignment cost. However, by reversing the costs and prices, we changed the algorithm to minimize the assignment cost. Moreover, we eliminated the users and points that cannot be assigned in the assignment preparation phase at lines 1 and 2. This pruning strategy increases efficiency by reducing the total number of elements in the assignment problem.

We used the Jacobi parallelization [19], where each processor is responsible for handling the bidding calculations for one unassigned user at a time. Therefore, a thread-safe global FIFO queue (Q) was used. All unassigned users were added to Q at line 5 of the assignment preparation phase. In the bidding phase, each thread polls the head of the Q(line 2) and handles the bidding calculations at lines 8 and 9. The price of the POIs and their winners queue are in the shared memory. However, the bid calculations can be made on out-of-date POI prices in the parallel auction algorithm. When there is more than one thread that makes a bid to the same POI, then they are evaluated sequentially at line 8. The auction algorithm does not use any approximation; it returns the exact result with an average performance of O(NAlog(N)) [28], where A is the number of assigned pairs and N is the number of users.

4. Experimental results

In the experiments, we generated sets of mobile users U and stationary POIs P with different cardinalities such as N and M. The locations of users and POIs are randomly generated in [0: 1000]^2 space, according to uniform or Gaussian distributions with several standard deviations. In default settings, the location of the

Algorithm 2 Capacity and coverage constraint assignment parallel algorithm.

Variables: U, P = n users and m POIs; Cu_i, Cp_j = capacity of user u_i and POI p_j; Cu_ip_j = assigned capacity from u_i to p_j; Q = thread-safe global FIFO queue that has all unassigned users.

Assignment preparation phase:

```
1: If (C$p_i$ = 0) then eliminate($p_i$);//Eliminate all points with no capacity.
        //Eliminate all users with no capacity or no point in their coverage.
2: If (C$n_i$ = 0 OR $n_i$.getNumberOfCandidates() = 0) then eliminate($u_i$);
        3: For all users u ∈ U do in parallel//Calculate assignment costs for each point in the coverage.
3:    $n_i$.calculateAssignmentCostsWithCandidatePOIs()
4:    add $n_i$ to Q
5: loop;
6: For all POI p ∈P do
7:    $p_j$.initializePrice();
8: loop;
        //Wait for all threads to complete the preparation phase.
        Bidding phase:
9: While (Q ≠ ∅) do in parallel threads.
10:    $n_i$ = Q.pollHead();
11:    if (C$n_i$ = 0) {eliminate($n_i$); Go to line 2;} // No user to assign, continue with next.
12:    $p_j$ = $n_i$.findMaxBenefitPOI();//Find the lowest cost + price POI in the coverage.
13:    If ($p_j$ == ∅) {eliminate($n_i$); Go to line 2;} // No POI to assign, continue with next.
14:    //for each unassigned capacity of $n_i$
15:    For(k=0;$k < Cu_i - \sum_{j=1}^{m} Cu_ip_j$;k++) do
16:        Bid$_k$ = $u_i$.makeBid() ;
17:        $p_j$.processBids(Bid$_k$, $n_i$);
18:    loop;
19:    loop;
```

Processing bids:

```
1: processBids(Bid$_k$, $n_i$, $p_{j.}$){
2: //Assume that Bid$_l$ is the lowest bid in the winners queue and $n_l$ is the owner of Bid$_l$
3:    if ($\sum_{i=1}^{n} Cu_ip_j < Cp_j$) {//if $p_j$ has available capacity.
4:        Add Bid$_k$ of $n_i$ to winners queue of $p_j$ ; C$n_i$ –;    }
5:    if ($\sum_{i=1}^{n} Cu_ip_j == Cp_j$) {//if $p_j$ is full, add Bid$_k$ of $n_i$ to winners queue of $p_j$ ;
6:        $p_j$.setPrice(Bid$_l$ + $\varepsilon$); C$n_i$ –;    }
7:    if ($\sum_{i=1}^{n} Cu_ip_j > Cp_j$) {//If capacity limit of $p_j$ is exceeded.
8:        Add Bid$_k$ of $n_i$ to winners queue of $p_j$; Poll Bid$_l$ from the winners queue of $p_j$ ;
9:        $p_j$.setPrice(Bid$_l$ + $\varepsilon$); C$n_i$ –; Re-add the user $n_l$ to Q; C$n_l$ ++;    }
10:    }
```

users is generated by a Gaussian distribution, which has standard deviation σ that is equal to 0.2, multiplied by the shortest path length to the furthest node from the center and mean at the center of the coordinate system. This model may simulate a crowded city center during working hours. M POIs are generated by a uniform distribution with equal POI capacities C_p and radius R. A user should be in the coverage radius of POI in order to get assigned. More than one user can be assigned to the POI as long as its capacity C_p is not exceeded. C_u is used to model a system where multiple users are at the same location.

Unless otherwise stated, the number of users N is 100 (K) and the number of POIs M is 1000. We repeated each simulation 50 times and reported the average of the observed values. All experiments were implemented in Java and executed on an Intel Core 2 Duo P7550 dual-core 2.26 GHz machine with 4 GB of memory. For comparison purposes, we executed a multicore simulation in an Intel Xeon L5410 two quad-core 2.33 GHz machine with 8 GB of memory. In what follows, we provide a summary of our experimental results.

In Figure 7, we measure the CPU time used by the assignment under various user cardinalities N. The processing time increases with an increase in the number of users. However, the increase is not linear, and it slows down because a lower M/N ratio implies that most assignments are performed locally.

In Figure 8, we vary the number of POIs and M, and we plot the CPU time graph. The number of alternatives for a user increases with an increase in the total number of POIs. Searching for a maximum benefit POI in a large candidate list is relatively slow, so the CPU time increases with an increase in M.

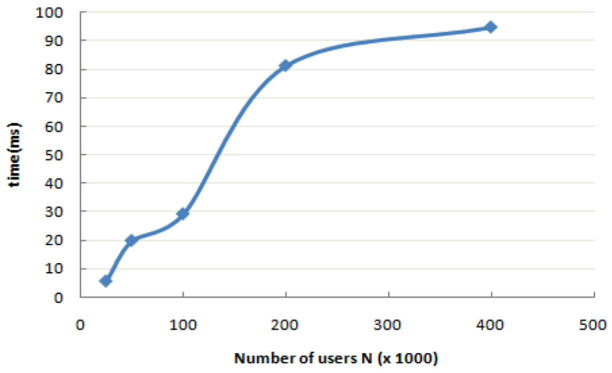


Figure 7. CPU time vs. user cardinality N.

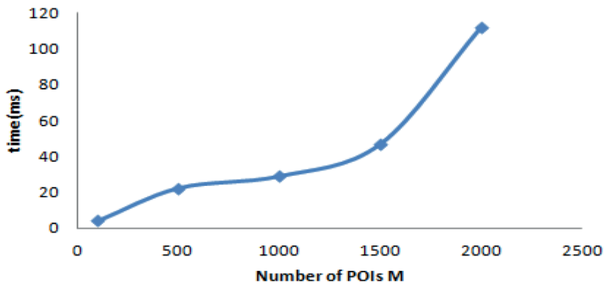


Figure 8. CPU time vs. POI cardinality M.

The user capacity (the number of users at the same location represented as a single user, instead of different users) increases the performance more than 100% for a total number of users greater than 300,000.

As for the effect of the coverage radius R on the performance, when the coverage radius increases, the number of potential matches increases and the problem becomes harder.

To measure the performance of the algorithm when there are multiple processors, we performed the test in two separate machines: Core 2 Duo, which has a single dual core processor, and Xeon Quad-Core, which has two quad-core processors. We obtained the best result (40% increase in performance) with Core 2 Duo. Xeon has a high-performance increase when there are two processors, because two cores are selected from the same physical processor in order to use the shared cache. When there are 3 processors, the cache mismatches and context-switching increases the running time. However, the performance continually decreases when there are more than three processors, since the benefit of parallelism is higher than the synchronization cost.

Memory usage increases linearly with the number of POIs. For example, for 200 POIs, the SS-tree uses 250 KB of storage space.

We examined the displacement due to the unit square spiral length parameter Γ and calculated the average distance of the users to the center of the square they belong to. The displacement increases linearly with the unit square spiral length Γ.

We also compared the efficiency of the square spiral transformation with a widely used and efficient encryption scheme, the data encryption standard (DES). In the experiments, DES caused 52% delay in execution time and required 75% more space for the encrypted data. Furthermore, when the database could not execute the location query over the encrypted data, then the database sent all encrypted locations of POIs to the client. This multiplies the communication cost by M times, where M is the number of POIs. In this case, the client should decrypt the location of POIs and then execute the query by themselves; this multiplies the decryption cost by M times again. As a result, the fully encrypted method suffers from high communication and computation costs and is much less efficient than our proposed method.

5. Conclusions

In this paper, we proposed a novel method to protect location privacy not only for static local queries but also for dynamic global queries such as assignment queries. Our main contribution is the introduction of a location privacy-aware method. The proposed method does not require any trusted third party or intermediary. It uses square spiral encoding to provide privacy and to preserve some of the spatial properties. Our contributions may be summarized as follows: efficient implementation of capacity and coverage-constrained assignment query on outsourced databases; a novel spatial transformation strategy (square spiral encoding) to achieve privacy efficiently for approximate query results; and adapting the idea of the HG-tree [36], which is implemented for the Hilbert curves to the square spirals, and introducing the SS-tree. In future research, the proposed method can be adapted to other location-based queries such as KNN or range queries.

References

[1] Çokpınar S, Gündem Tİ. Positive and negative association rule mining on XML data streams in database as a service concept. Expert Syst Appl 2012; 39: 7503-7511.

[2] Unay O, Gundem TI. Parallel processing of encrypted xml documents in database as a service concept. Inf Technol Control 2010; 39: 301-309.

[3] Unay O, Gundem TI. A survey on querying encrypted XML documents for databases as a service. SIGMOD Record 2008; 37: 12-20.

[4] Chow CY, Mokbel M, Liu X. Spatial cloaking for anonymous location-based services in mobile peer-to-peer environments. GeoInformatica 2011; 15: 351-380.

[5] Gedik B, Ling L. Protecting Location privacy with personalized k-anonymity: architecture and algorithms. IEEE T Mobile Comput 2008; 7: 1-18.

[6] Khoshgozaran A, Shirani-Mehr H, Shahabi C. Blind evaluation of nearest neighbor queries using space transformation to preserve location privacy. GeoInformatica 2013; 17: 599-634.

[7] Hacıgümüş H, Iyer B, Li C, Mehrotra S. Executing SQL over encrypted data in the database-service-provider model. In: Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data; 3–6 June 2002; Madison, WI, USA. New York, NY, USA: ACM. pp. 216-227.

[8] Indyk P, Woodruff D. Polylogarithmic private approximations and efficient matching. In: Halevi S, Rabin T, editors. Theory of Cryptography. Berlin, Germany: Springer, 2006. pp. 245-264.

[9] Bilogrevic I, Jadliwala M, Joneja V, Kalkan K, Hubaux J-P, Aad I. Privacy-preserving optimal meeting location determination on mobile devices. IEEE T Inf Foren Sec 2014; 9: 1141-1156.

[10] Khoshgozaran A, Shahabi C, Shirani-Mehr H. Location privacy: going beyond K-anonymity, cloaking and anonymizers. Knowl Inf Syst 2011; 26: 435-465.

[11] Ghinita G, Kalnis P, Khoshgozaran A, Shahabi C, Tan K-L. Private queries in location based services: anonymizers are not necessary. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of data; 9–12 June 2008; Vancouver, Canada. New York, NY, USA: ACM. pp. 121-132.

[12] Lien I, Lin Y, Shieh J, Wu J. A novel privacy preserving location-based service protocol with secret circular shift for k-NN search. IEEE T Inf Foren Sec 2013; 8: 863-873.

[13] Cormen TH. Introduction to Algorithms. 2nd ed. Boston, MA, USA: MIT Press, 2001.

[14] Karp RM, Vazirani UV, Vazirani VV. An optimal algorithm for on-line bipartite matching. In: Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing; 13–17 May 1990; Baltimore, MD, USA. New York, NY, USA: ACM. pp. 352-358.

[15] Kuhn HW. The Hungarian method for the assignment problem. Nav Res Logist Q 1955; 2: 83-97.

[16] Munkres J. Algorithms for the assignment and transportation problems. J Soc Ind Appl Math 1957; 5: 32-38.

[17] Derigs U. The shortest augmenting path method for solving assignment problems — motivation and computational experience. Ann Oper Res 1985; 4: 57-102.

[18] Bertsekas DP. Auction algorithms for network flow problems: a tutorial introduction. Comput Optim Appl 1992; 1: 7-66.

[19] Bertsekas DP, Castañon DA. Parallel synchronous and asynchronous implementations of the auction algorithm. Parallel Comput 1991; 17: 707-732.

[20] Konan AR, Gündem Tİ, Kaya ME. Assignment query and its implementation in moving object databases. Int J Inf Tech Decis 2010; 9: 349-372.

[21] U LH, Yiu ML, Mouratidis K, Mamoulis N. Capacity constrained assignment in spatial databases. In: Proceedings of the 2008 ACM SIGMOD international conference on Management of Data; 9–12 June 2008; Vancouver, Canada. New York, NY, USA: ACM. pp. 15-28.

[22] U LH, Mouratidis K, Mamoulis N. Continuous spatial assignment of moving users. VLDB J 2010; 19: 141-160.

[23] Orlin JB, Lee Y. QuickMatch: A Very Fast Algorithm for the Assignment Problem. Cambridge, MA, USA: MIT Press, 1993.

[24] Görlach A, Heinemann A, Terpstra W. Survey on location privacy in pervasive computing. In: Robinson P, Vogt H, Wagealla W, editors. Privacy, Security and Trust within the Context of Pervasive Computing. New York, NY, USA: Springer, 2005. pp. 23-34.

[25] Kalnis P, Ghinita G, Mouratidis K, Papadias D. Preventing location-based identity inference in anonymous spatial queries. IEEE T Knowl Data En 2007; 19: 1719-1733.

[26] Ghinita G, Kalnis P, Kantarcioglu M, Bertino E. A hybrid technique for private location-based queries with database protection. In: Mamoulis N, Seidl T, Pedersen T, Torp K, Assent I, editors. Advances in Spatial and Temporal Databases. Berlin, Germany: Springer, 2009. pp. 98-116.

[27] Mokbel MF, Chow CY, Aref WG. The new Casper: query processing for location services without compromising privacy. In: Proceedings of the 32nd International Conference on Very Large Databases; 12–15 September 2006; Seoul, Korea. pp. 763-774.

[28] Gruteser M, Grunwald D. Anonymous usage of location-based services through spatial and temporal cloaking. In: Proceedings of the 1st International Conference on Mobile Systems, Applications and Services; 5–8 May 2003; San Francisco, CA, USA. New York, NY, USA: ACM. pp. 31-42.

[29] Zhang C, Huang Y. Cloaking locations for anonymous location based services: a hybrid approach. GeoInformatica 2009; 13: 159-182.

[30] Radu S, Carbunar B. On the computational practicality of private information retrieval. In: Proceedings of Network and Distributed System Security Symposium; 28 February–2 March 2007; San Diego, CA, USA.

[31] Khoshgozaran A, Shahabi C. Blind evaluation of nearest neighbor queries using space transformation to preserve location privacy. In: Proceedings of the 10th International Conference on Advances in Spatial and Temporal Databases; 16–18 July 2007; Boston, MA, USA. Berlin, Germany: Springer-Verlag. pp. 239-257.

[32] Lin D, Bertino E, Cheng R, Prabhakar S. Location privacy in moving-object environments. T Data Privacy 2009; 2: 21-46.

[33] Yiu ML, Christian SJ, Xuegang H, Hua L. SpaceTwist: managing the trade-offs among location privacy, query performance, and query accuracy in mobile services. In: Proceedings of the IEEE 24th International Conference on Data Engineering; 7–12 April 2008; Cancun, Mexico. New York, NY, USA: IEEE. pp. 366-375.

[34] Yiu ML, Ghinita G, Jensen C, Kalnis P. Enabling search services on outsourced private spatial data. VLDB J 2010; 19: 363-384.

[35] Yiu ML, Ghinita G, Jensen CS, Kalnis P. Outsourcing search services on private spatial data. In: Proceedings of the IEEE 25th International Conference on Data Engineering; 29 March–2 April 2009; Shanghai, China. New York, NY, USA: IEEE. pp. 1140-1143.

[36] Cha GH, Chung CW. A new indexing scheme for content-based image retrieval. Multimed Tools Appl 1998; 6: 263-288.

[37] Stein M, Ulam S, Wells M. A visual display of some properties of the distribution of primes. Am Math Mon 1964: 516-520.