Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 2067047, 9 pages
https://doi.org/10.1155/2018/2067047

Research Article

A Privacy-Preserving Spatial Index for Spatial Query Processing

Doohee Song[1], **Moonbae Song**[2] **and Kwangjin Park**[1]

[1]*Department of Information Communication Engineering, Wonkwang University, Iksan-shi, Republic of Korea*
[2]*Samsung Electronics, Suwon, Republic of Korea*

Correspondence should be addressed to Kwangjin Park; kjpark@wku.ac.kr

Received 16 October 2018; Accepted 27 November 2018; Published 16 December 2018

Academic Editor: Laurie Cuthbert

An increasing amount of active research is being conducted to protect the locations of mobile device users. Users must tune to more data than they would like to in order to hide their location. In particular, if a user requests a query over *k*NN, the number of objects the user must receive may increase. Several studies have been proposed to solve these problems. However, problems have been identified during the course of query processing, such as errors and increased query processing times. When the tuning time is increased, the amount of data to download and the battery consumption of the client also increase. In this study, we propose the Privacy-preserving Spatial Index (PSI), an index that allows users to reduce their tuning time while being satisfied with the results of their queries. The querier (q) requests the object in the area protecting his/her location from the server. The server sends the requested data of points of interest (POIs) (DPOIs) in the Privacy-preserving Region (PR) to q. Finally, q reduces tuning time by selectively tuning to the desired data of POIs (Dw) through PSI. The superiority of PSI over previous techniques is experimentally proven.

1. Introduction

As the use of mobile devices has recently increased rapidly, the use of location-based services (LBS) based on GPS has also increased. LBS refers to various information services provided by the LBS server based on the location of mobile users, such as finding nearby points of interest (POIs), navigation, location tracking, and maps [1–5].

However, users must reveal their location information in order to access LBS. When the location information of a user is sent to the server, the server can precisely identify the location of the user. If the server is hacked or otherwise abused, the location of the user can be revealed, potentially causing serious damage. Active research is therefore being conducted on the protection of user locations [6–10]. The D-Bcast model was proposed to enable clients that have not received data from the main server or clients that have moved from other main servers to effectively listen to data [6]. The method proposed by [7] can store the received data in cache and reuse them in order to minimize the exposure of user query service data to the unreliable LBS server. Reference [9] proposes a method that can make the user location ambiguous through the location anonymity. However, [10] points out that location anonymity is also an extension of server and cannot be trusted. The cloaking method can prevent the exposure of the specific location of a user such as building information, because queries are sent from a generalized area including the user instead of the specific location of the user. If the location of a user is continuously revealed to the server, the movement path of that user can be exposed [11–16]. For example, let us assume that a mobile user sends queries in a certain path from a starting point to a destination. The server can predict the moving path of the user by connecting the locations of the query points from the starting point to the destination. If this path is revealed to a malicious attacker, the living pattern as well as the home and work addresses of the user can be revealed, and other pieces of information such as the hospitals that the user has visited before can also be revealed. Thus, this can lead to privacy issues. Therefore, it is critical to protect this trajectory information, which is a set of location data, as well as the location information in general when using LBS. However, in order to hide their location, users must increase the number of areas or paths involved in query processing

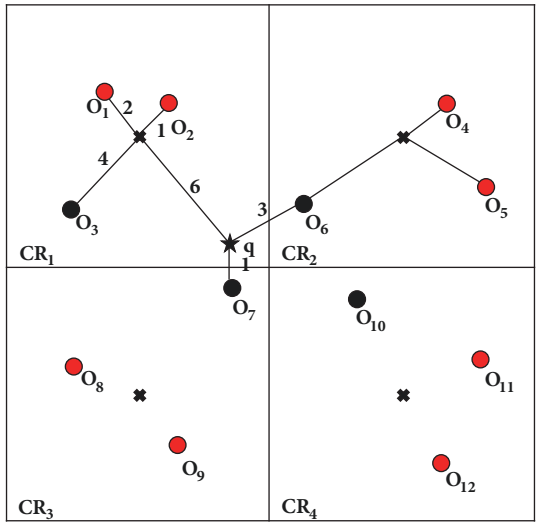

Figure 1: Error when objects are searched according to the center of CR.

and tune to more data than they would like to. If the tuning time increases, the data to be downloaded by the client and the battery consumption increase. Recently, a method that decreases the number of objects that need to be checked while protecting the location and query of a mobile user has been proposed [17–20]. Reference [18] proposes a method of solving the problems of users and servers. For example, users want to hide their location and the server must not want to process the queries of all users. Therefore, a system model that satisfies the needs of users and servers was developed by proposing a mobile service provider that gathers the queries of multiple users. Reference [20] considered various types of objects as a method for supporting effective approximate *k*NN queries. This method does not have to reveal the accurate location of the querier (q) to the server because the query is requested from the center of a grid of a map divided by the server instead of the precise location of the q. While the query should be requested from the center of a grid, however, an error may occur such that a query is made based on the location of the q. When the q provides cloaking region (CR); CR_1, CR_2, and q requests a query of the server; Q instead of his/her location, the server provides the q with objects corresponding to Response Generation (RG); RG (Q, $CR_{1,2}$), thereby protecting the location of the q. However, when the objects are searched according to the center of CR, an error may occur as shown in Figure 1.

Figure 1 shows that the entire map has been quartered into CR_1 ~CR_4 and that three objects exist in each CR_i. It is assumed that to protect his/her location, the q selects CR_1 and CR_2 and finds two objects that are closest to himself/herself. Through the method outlined in [20], when two objects are searched from the center of CR_i, the resultant values of CR_1 and CR_2 are $\{O_1, O_2\}$ and $\{O_4, O_5\}$, respectively. However, if two objects are searched based on the actual q, the resultant value is $\{O_6, O_7\}$. Furthermore, the q who receives RG from the server additionally requires a decryption process for the encrypted data. Furthermore, the probability of exposing the q's location increases with *k*-anonymity and the grid size. In addition, as anonymous servers are removed, users should create anonymous zones to protect their location, because users' smart devices have improved in performance of calculating in recent years. In addition, it is considered more important to remove middleware under the circumstances where privacy is becoming an issue and to be 100% reliable in terms of location protection. On the other hand, if the number of candidate objects users receive from the server increases, the cost of exploring the objects of the query results may increase. Thus, in this study, we propose the Privacy-preserving Spatial Index (PSI), which allows for the selective tuning of required object data while protecting the locations of users. As far as we know, this is the first study that enables the tuning of object data in a location protection method. The key contributions of this study are as follows:

(i) There are many methods to protect user location. We propose PSI, which is a general index structure that can support the existing methods.

(ii) Since the user directly makes an anonymous request, no third party can expose his or her information.

(iii) If users set a large query region to protect their location, they must receive data for the number of objects in the set range. On the other hand, if the query range is narrowed in order to reduce the amount of received data, the probability of the user location being exposed can increase. Therefore, we reduce the tuning time by selectively tuning to the object data that must be received from the server.

(iv) We have proven through experimentation that the proposed method exhibits better performance than the existing location protection methods.

This paper is organized as follows: Section 2 describes related works on the protection of trajectory. Section 3 describes our model and Section 4 proposes various queries using the PSI index. Section 5 compares the performance between the PSI and the existing method through experimental results. Finally, Section 6 outlines the conclusions.

2. Related Work

LBS queries are generally classified as either snapshot queries or continuous queries [1–14]. The query process using snapshot queries is as follows. Methods using *k*-anonymity to protect user privacy have been recently proposed [5–8]. Reference [8] constructs a CR by combining the q with other *k*-1 users and then sends the CR to the LBS server instead of the actual location of the q. Reference [19] proposes a dynamic grid system (DGS) that allows users to protect their personal information. Through the DGS, users can protect their location for the grid radius from unreliable servers through the process of sending encrypted queries to the query server and transmitting the content of the queries to the LBs server. However, the encryption and decryption between users and servers can increase the query processing time. Reference [20] improved the problems that could occur when clients are grouped by k and moved to a technique

that protects the locations of users. However, this method has a problem because users must obtain consent from surrounding clients and the movement time and direction need to be considered. Reference [21] proposes a method for protecting the locations of users by using dummies using an enhanced-dummy location selection scenario. However, it has limitation in applying to continuous techniques because it considers snapshots. Reference [22] proposes a method of efficiently placing k dummies to protect the locations of users. However, the dummies may be concentrated on the center if they are placed only by angles depending on the number of dummies. Reference [23] suggests a method of preventing the generation of dummies in arbitrary directions while users move in certain directions if dummies are created randomly while users are moving. The proposed method prohibits the users from moving out of a specific range using the radius d. However, even if dummies are generated within the radius of d, they are likely to be generated in zigzags in contrast to the moving path of the users, and there is a possibility of exposing the locations of users. Reference [24] protects the user information in continuous LBS based on the method of [13]. However, it has a possibility that the user location protection probability will decrease because it does not consider various situations (obstacles) during the generation of dummies.

Aside from that method, there has been research into methods using dummies [20–23] as well as into the encryption of user information [24, 25]. However, the above studies require middleware (hereinafter referred to as an "anonymous server").

Because the anonymous server called k-anonymity exists, client information can be revealed if a third party attacks the anonymous server. To address this, the k-anonymity method was proposed, which uses a peer to peer (P2P) process instead of the anonymous server [26]. Although the privacy level is high because users communicate among themselves without the use of an anonymous server, personal privacy can still be compromised because other users cannot be trusted fully.

Reference [20] proposed a method for supporting effective approximate kNN queries. The query process of this method consists of three steps: Query Generation (QG), Response Generation (RG), and Response Retrieval (RR). In QG, the q requests a query of the server. QG is equal to (Q, s) where Q includes CR, $n \times n$ cells, m POI types (t), the location of q (i, j), and the number of objects to be found and s is for protecting Q. In RG, the server receives (Q, s) from the q and the objects that satisfy the query are sent (R) to the q from the database (D) in which POIs are stored, and this is referred to as RG(Q, D). Finally, RR outputs k objects from the RG(Q, D) received from the server considering k and t requested by the q, and this is referred to as kNN=RR(R, s).

Continuous queries refer to queries continuously sent to the LBS server in real time to the destination. They consist of multiple snapshots, creating a trajectory of the user by connecting the locations of snapshots.

Cloaking methods used to protect continuous queries or the trajectory of the user include the k-anonymity method and the dummy trajectory creation method. The proposed trajectory k-anonymity method receives a similar trajectory as the trajectory of the q in the database which is stored in the anonymous server, and the k-1 locations of other users are grouped together. Queries are then randomly made. However, this method requires an anonymous server and there must be other users near the query location. If a user is somewhat far away, the CR becomes large and the amount of searched data increases, lowering the query processing efficiency.

3. Background

PSI = $\{\alpha, \beta, B_{map}$ (or $C_{map}), T_D\}$. α and β denote the numbers of divisions of the x-axis and y-axis, respectively. B_{map} is a bitmap in the $\alpha * \beta$ grid. C_{map} indicates the existence of object cell coordinates and objects. If B_{map} is larger than C_{map}, the server can provide C_{map}. The sizes of C_{map} and B_{map} can be measured by

$$B_{map} = 2^{(\alpha * \beta)} \tag{1}$$

$$C_{map} = \{2\log_2(\alpha * \beta) + 1\} * D_{POIs} \tag{2}$$

In (1) and (2), Privacy-preserving Region (PR) is the range requested by the user and D_{POIs} is the number of POIs in the PR. The server provides information about B_{map} or C_{map} to the q based on the size of the PR and the number of D_{POIs}.

The purpose of our study is to protect the location of users from the server and to enable effective data tuning. The query process is divided into three steps as follows.

(1) Spatial query (SQ): to protect his/her location, the user sets a PR based on his/her current location and the map data that he/she has. The q requests SQ_{PR}, which are the POIs included in the PR, from the server.

(2) Privacy-preserving Spatial Index (PSI): the server manages the dataset (D) of every POI in the map. The server also divides the PR by n for the x-axis and by m for the y-axis (α=β depending on the distribution of objects). The cell coordinate ($C_{i\text{-}j}$) is set for each divided grid. Each grid can have one object, and bit 1 is saved if it has an object or bit 0 is saved if otherwise. Figure 2 shows the setting of the sequence of bitmaps (B_{map}) based on $C_{i\text{-}j}$.

$$\text{Order of } B_{map} = i * \alpha + j \tag{3}$$

The ranges of i and j are as follows: $0 \le i \le \alpha$, $0 \le j \le \beta$.

If the data sizes of the POIs are identical, the data arrival time can be confirmed through B_{map}. If the data sizes of the POIs differ, the T_D is further configured. The data arrival time size of T_D is assumed to be identical. Finally, the data of all POIs in the PR are sent to the q.

(3) Dataset to the SQ (DSQ): the q first receives the PSI and selects the desired objects through the PSI. The q can then confirm the locations and sending times of the desired POIs through the PSI. Thus, the q selectively tunes to only the data corresponding to D_W among the D_{POIs}. $D \supset D_{POIs} \ni D_W$.

4. Our Model

4.1. Our System Model. As shown in Figure 3, the basic system model is composed of a movement device, a positioning system, and a single LBS server. If an attacker attacks

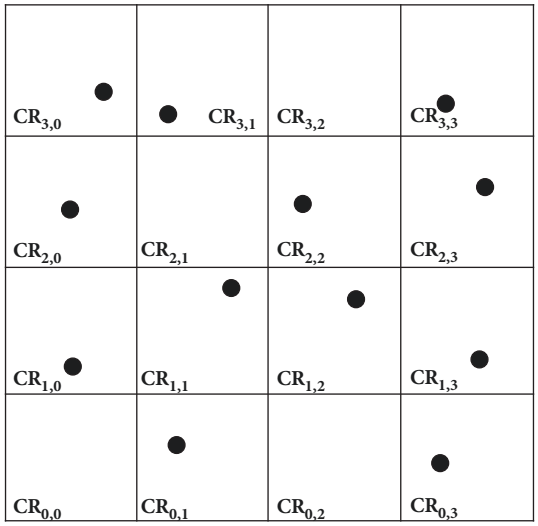

Figure 2: Example of sequence of bitmaps based on C_{i-j}.

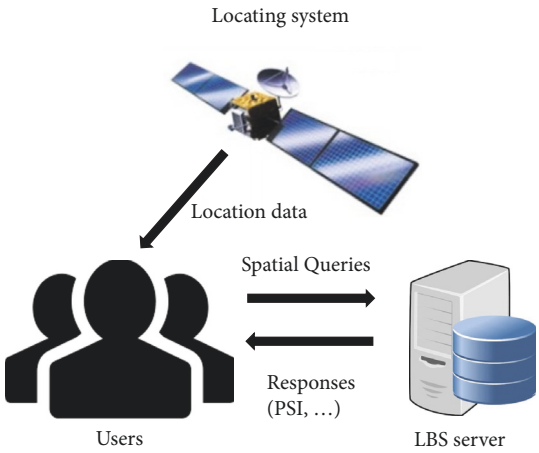

Figure 3: PSI system model.

the LBS server or if the LBS server is unreliable, various pieces of information about the q can be exposed. Therefore, protecting the location information is critical in LBS.

The existing system is composed of an LBS server, an anonymous server, and mobile users. However, the anonymous server cannot be trusted. The anonymous server is a single point of failure, and if is attacked, some or all services will fail. In general, the q sends his/her location information to the LBS server to receive information, and this causes the problem of location exposure. Therefore, we assumed that the q acquires map information through the broadcast method from the LBS server. The advantage of the broadcast method is that the client can obtain map information without exposing one's location information.

The server manages the locations and other information (e.g., price, discount, advertisement) of objects that the q does not have (e.g., gas station, hotel, restaurants). For example, the q creates a PR based on his/her own location through the map information and satellites stored in the terminal. Then he or she requests the location and price of a nearby gas station after

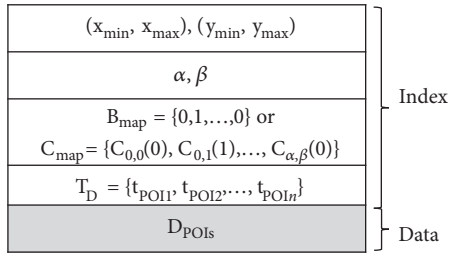

Figure 4: PSI structure.

creating a PR based on his or her location, confirmed through the map and satellite. The server provides the location and other information of gas stations (D_{POIs}) that exist in the PR requested by the q. If the q selects only one nearest gas station, he or she can only tune to the data of one gas station among the ten.

4.2. PSI Index Structure and Query Process. The PSI structure is composed of α, β, B_{map}, or C_{map} (varies according to the number of objects) and the data arrival time table (T_D), as shown in Figure 4.

5. Various Queries Using the PSI Index

In this chapter, we introduce the process of querying after applying PSI to the existing method for protecting the user location. There are three existing methods mainly used, which are defined as follows.

Definition 1 (cloaking-based spatial query (CSQ)). In general, users set an area that is equal to or greater than their desired area as the PR in order to protect their location. Users request information about the objects in the PR without providing their location to the server. The server cannot verify the location of the user because it only receives information about the PR from the user and sends only information on the objects in the PR to the user. The users have the advantage of not revealing their location, but they do have to check all objects in the PR. Meanwhile, the server incurs no additional costs (e.g., searching for the object that is closest to the user) because it does not know the user's precise location.

Figure 5 shows an example of processing the cloaking-based spatial query using PSI.

The CSQ process is as follows:

Step 1. The q requests query results from the server via SQ. The structural elements of SQ in CSQ are as follows: First, the PR is set in a rectangular shape (this shape can vary by the request of the q). The PR of the CSQ (PR_{CSQ}) sets the minimum of x coordinate (x_{min}), the minimum of y coordinate (y_{min}), the maximum of x coordinate (x_{max}), and the maximum of y coordinate (y_{max}) and then requests the D_{POIs} that exists in PR_{CSQ} from the server.

$$PR_{CSQ} = \{(x_{min}, y_{min}), (x_{max}, y_{max})\} \quad (4)$$

Step 2. The server searches requested D_{POIs} in the PR_{CSQ} in the location-based D under its control. After checking the

Input: SQ(e.g., CQS, p-AQS, s-TrQS) of q
Output: PSI, D_{POIs}
Procedure:
01: The server check $PR_{QS} = \{(x_{min}, y_{min}), (x_{max}, y_{max})\}$
02: PR is divided by α for the x axis and by β for the y axis
03: Bit 1 is saved if it has an object or bit 0 is saved if otherwise
04: The server computes PSI and sends to q
05: The q check PSI
06: Checks the location of the objects through B_{map}
07: The q can check the POI number by adding the sequence of B_{map} and bit 1
08: The q selectively tunes to T_D and D_W

Algorithm 1: SQ processing using PSI.

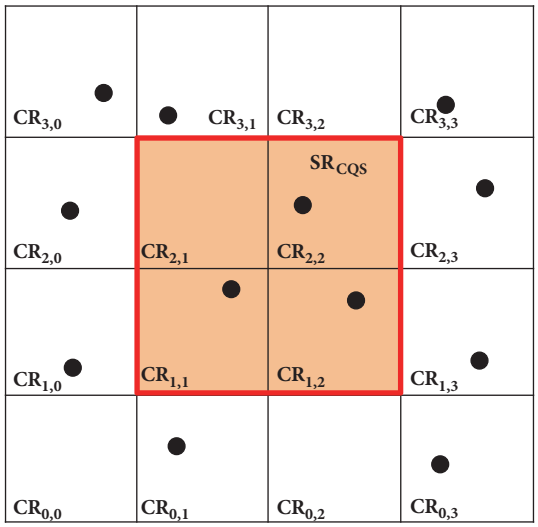

Figure 5: Example of processing the CQS using PSI.

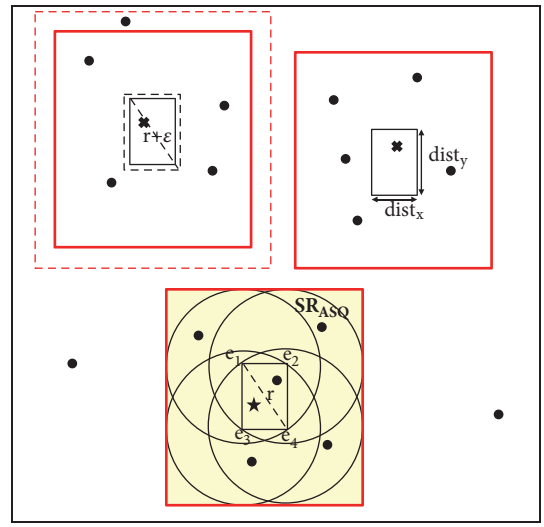

Figure 6: Example of processing the p-ASQ using PSI.

distribution of D_{POIs}, the PR is divided by α for the x-axis and by β for the y-axis so that only one POI will exist in each grid (α=β depending on the distribution of objects). B_{map} is configured through (3). Figure 4 shows that B_{map} is configured as "0101111110111101" according to the distribution of POIs. Finally, the server sends the PSI = $\{\alpha, \beta, B_{map}, T_D\}$ and D_{POIs} to the q.

Step 3. The q divides the map using the PR_{CSQ} that he/she requested as well as the α and β values of PSI and checks the location of the objects through B_{map}. The POIs included in the search region (SR) that the user wants to search are checked and the frame number of POI is checked through B_{map}. Figure 5 shows that the POIs included in the SR are $\{C_{1,1}, C_{1,2}, C_{2,1}, C_{2,2}\}$. The q can determine the POI number by adding the sequences of B_{map} and bit 1. Finally, the q selectively tunes to T_D and D_W only (Algorithm 1).

Definition 2 (p-Anonymity-based Spatial Query (p-ASQ)). We define p-anonymity in order to prevent confusion with k in kNN and k-anonymity. p is a virtual q that the q provides the server with to obfuscate his/her location, and the server cannot distinguish between the location of the q and the location of p-1. As proposed in [19], we also assume that the query is sent to the server with the location of q and the location of p set in the grid area. The size of the area needed to guarantee the accuracy of query result when a query is requested based on the grid is expressed as

$$\text{maxdistance} = r = \sqrt{\left(\text{dist}_x^{\,2} + \text{dist}_y^{\,2}\right)} \quad (5)$$

In (5), r generates a circle based on the longer length between x-axis and y-axis $\{(x_{min}\text{-}r), (x_{max}+r), (y_{min}\text{-}r), (y_{max}+r)\}$. All the grids included in this circle ($r + \varepsilon$) form an area where the POI that the q wants will exist.

Figure 6 shows an example of p-ASQ processing using PSI.

The process of p-ASQ is as follows.

Step 1. The q requests query results from the server via SQ. The structural elements of SQ in p-ASQ are as follows: First, the locations of the q and p-1 virtual points (PRs) are specified and the PR is set in a rectangular shape. The PR of p-ASQ ($P_{p\text{-ASQ}}$) consists of dist_x, which is the distance of

the x-coordinates and $dist_y$, which is the distance of the y-coordinates. After $dist_{(x,y)}$ is randomly set based on p points requested by the q, p PRs are created and k POIs are requested.

Step 2. Among the location-based Ds under its control, the server verifies the $PR_{p\text{-ASQ}}$ requested from the q. Then, for the accuracy of the query result, adds δ to the ($dist_x$, $dist_y$) of $PR_{p\text{-ASQ}}$. Then, k POIs (D_{POIs}) are searched for based on p grids.

$$D_{POIs} = p * k + \lambda - \sigma \tag{6}$$

In (6), λ denotes the number of additional POIs included in $PR_{p\text{-}\delta ASQ}$. If k includes λ, λ=λ-k. σ denotes k POIs that are overlapped among the k POIs of $PR_{p\text{-}\delta ASQ}$.

The server sets an area that includes D_{POIs} and divides the map according to the distribution of POIs (same process as for B_{map}). The server finally sends PSI = {(x_{min}, y_{min}), (x_{max}, y_{max}), α, β, C_{map}, T_D} and D_{POIs} to the q.

Step 3. The q verifies (x_{min}, y_{min}) and (y_{max}, y_{max}) through the PSI received from the server and divides the corresponding map by α and β values. The locations of objects are verified through B_{map}. Finally, after k POIs are verified based on one's own location, the frame number of POI is verified through B_{map}. Finally, the q selectively tunes to T_D and D_W only (Algorithm 1).

Definition 3 (*s*-Trajectory based Spatial Query (*s*-TrSQ)). *s*-TrSQ sets the path from the starting location (L_S) to the ending location (L_E) in which the user will query. The trajectory distance of tr_q is defined as Tr_{dist} and it is assumed that the distance of tr_q and the trajectory distance of tr_i are all identical. To prevent the exposure of his/her tr_q, the q additionally creates s-1 tr_i and then sends a query to the server. trs are connected to ω nodes (n).

$$tr = \{n_1, n_2, \ldots, n_{\omega-1}, n_\omega\} \quad (2 \le \omega < \infty) \tag{7}$$

The server cannot distinguish between tr_q and tr_i. Therefore, the server sends the query result to the q based on Tr that the q requested.

Figure 7 shows an example of *s*-TrSQ processing using PSI.

The *s*-TrSQ process is as follows.

Step 1. The q requests query results from the server via SQ. The structural elements of SQ in *s*-TrSQ are as follows: First, tr_q is set. To create s-1 trs excluding tr_q, the q sets the creation range {(x_{min}, y_{min}), (x_{max}, y_{max})} and randomly sets s-1 trs in this creation range. As shown in Figure 7, the q sets the search range based on the Tr that he/she created and sends it to the server, and then requests D_{POIs} in the search range of this Tr.

Step 2. The server searches requested D_{POIs} in the $PR_{s\text{-TrSQ}}$ among the location-based Ds under its control. After checking the distribution of D_{POIs}, the PR is divided by α for the x-axis and by β for the y-axis so that only one POI will exist in each grid. The server configures the overlapping area between the divided grid and $PR_{s\text{-TrSQ}}$ as C_{map}. Finally, the server sends the PSI = {α, β, C_{map}, T_D} and D_{POIs} to the q.

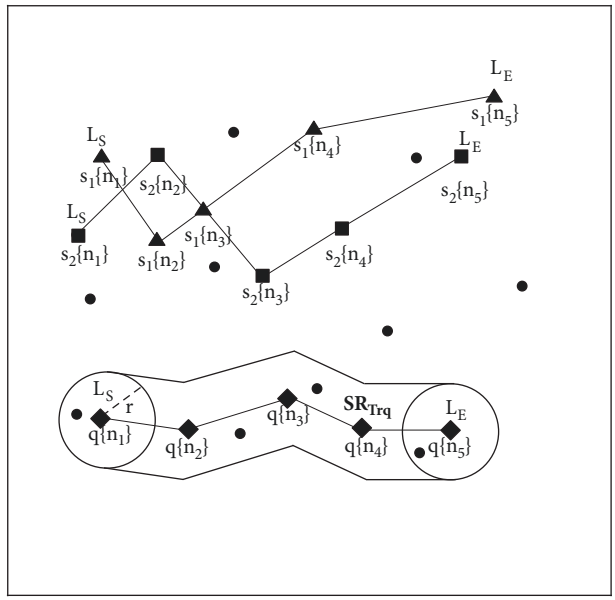

Figure 7: Example of processing the *s*-TrSQ using PSI.

Step 3. The q divides the map using the $PR_{s\text{-TrSQ}}$ that he/she requested and the α and β values of PSI, then checks the location of the objects through C_{map}. The POIs included in the search region (SR_{Trq}) that the user wants to search are checked and the frame number of POI is checked through C_{map}. Figure 7 shows the POIs included in the SR_{Trq}. The q can check the POI number by adding the sequences of C_{map} and bit 1. Finally, the q selectively tunes to T_D and D_W only.

6. Experimental Results

6.1. Experimental Environment. In this section, we discuss the experiments conducted for CSQ, *p*-ASQ, and *s*-TrSQ using PSI. We also compare them with the Original (Ori) CQS, *p*-ASQ, and *s*-TrSQ. In the experiments, the C++ programming language was used to actualize the algorithms on a 3.3-GHz CPU with 8 GB of main memory. We assumed the basic parameter setting values shown in Table 1 in order to evaluate the performance. We also discuss experiments conducted for CSQ, *p*-ASQ, and *s*-TrSQ using only the indexes of each method. To conduct these experiments, we set variables as their default values, except for the variables expressed as the values in parentheses in Table 1. Furthermore, the values of B_{map} and C_{map} are configured by (1) and (2) because they vary by query type. The size of a single grid is assumed to be $10m^2$. The experimental environment comprised a server, a client in 2D space, and a wireless broadcasting channel used by the client to obtain information. Tuning time can differ depending on bandwidth and transfer rate, so the data size was expressed as a graphical result (y-axis) in the experiment.

6.2. Experimental Results of CSQ. PR_{CQS} is 10% of the total map, and SR_{CQS} is set as 20% of PR_{CQS}.

Table 1: Experimental dataset values.

Parameter	Set values
grid	$10^5 * 10^5$
POIs size	10^8
PR_{CQS}	$10^4 * 10^4$
SQ_{CQS}	PR_{CQS} *10%, 20%, 30%, 50%
P	50, 100, 200, 300
K	10, 20, 30,50
S	10, 20, 30, 50
Tr_{dist} (km)	10, 50, 200, 300
Data size (K bytes)	128, 256, 512, 1024

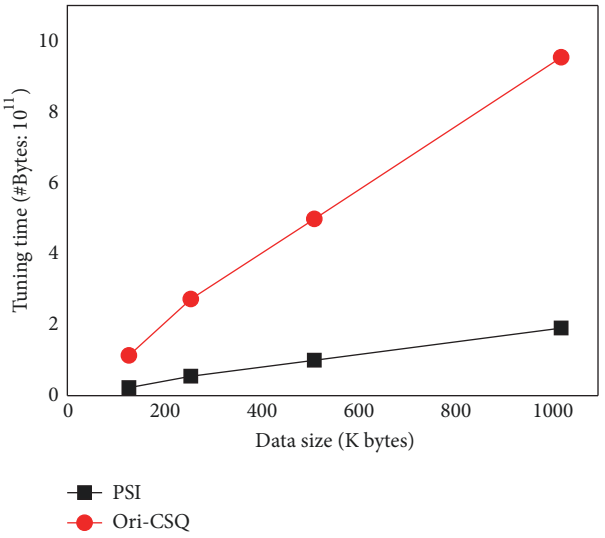

Figure 8: CSQ with different number of data size.

In Figure 8, the x-axis variable is divided into the data sizes of 128, 256, 512, and 1024 K bytes for comparison.

Figure 8 shows the variations in tuning time according to the data size. We can see that the performance of PSI improved by 80% more than that of Ori-CSQ. This is because the number of D_W that the PSI must search is smaller than the number of D_{POIs} that the Ori-CSQ must search. Therefore, as the data size increases, the difference in tuning time increases.

Figure 9 shows the variations in tuning time according to the size of the search range SR_{CQS} desired by the q. The default settings are shown in parentheses in Table 1. The variable SR_{CQS} was set as 0%, 20%, 30%, and 50% of the size of PR_{CQS}. We can see that the performance of PSI improved by 71.5% on average compared to that of Ori-CSQ. This is because as the SR_{CQS} increases, the number of D_W in the SR_{CQS} increased when the tuning time of PSI also increases.

6.3. Experimental Results of p-ASQ. To process p-ASQ, we set the default values listed in Table 1. In Figure 10, the default value p is 100, the data size is 256 K bytes, and the variable of the x-axis is k POIs that are closest to the q, which are divided into 10, 20, 30, and 50 for comparison.

Figure 10 shows the variations in tuning time according to the size of k. The performance of PSI is higher by 98.8%

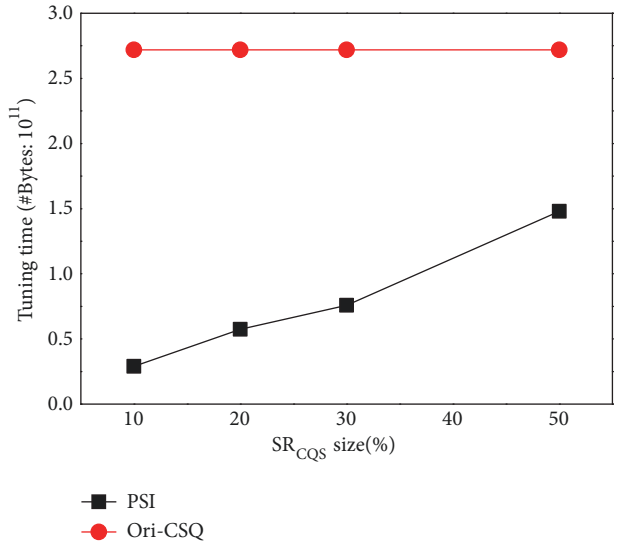

Figure 9: CSQ with different number of SR_{CQS} size.

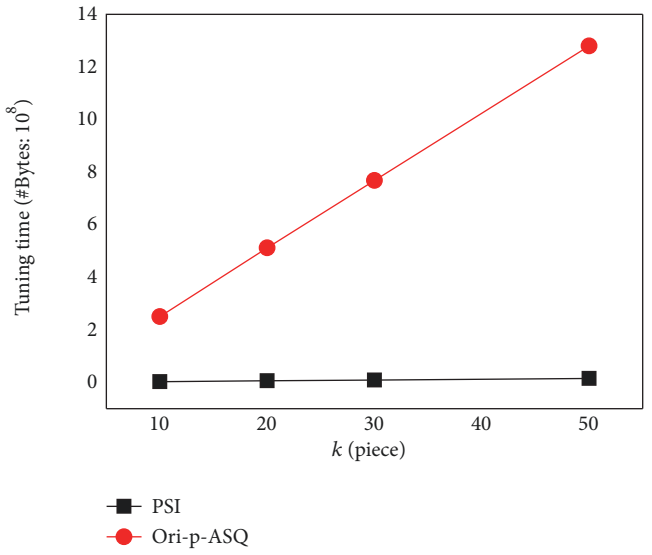

Figure 10: p-ASQ with different values of k.

than that of Ori-p-ASQ, this is because Ori-p-ASQ must tune to all k POIs corresponding to p's locations.

Figure 11 shows the variations in tuning time according to the size of p. The variable of the x-axis is p, and the p size is set to 50, 100, 200, and 300. As the p size increases the tuning time of PSI stays constant, but the tuning time of Ori-p-ASQ greatly increases. In the case of PSI, only k POIs need to be calculated because the location of the q is already known. However, as the Ori-p-ASQ increases, the POIs corresponding to (6) must be tuned, greatly increasing the tuning time. Therefore, the performance of PSI improved by 99.2% on average more than that of Ori-CSQ.

6.4. Experimental Results of s-TrSQ. To process s-TrSQ, we set the default values listed in Table 1. The default value Tr_{dist} in Figure 12 is 50km and the data size is 256 K bytes. The

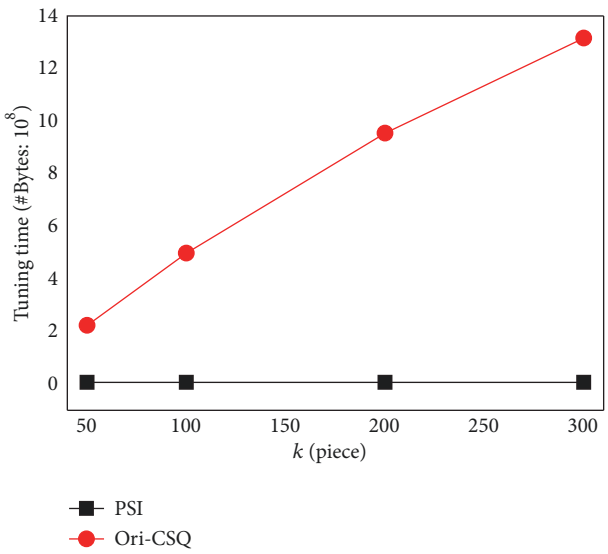

Figure 11: p-ASQ with different values of p.

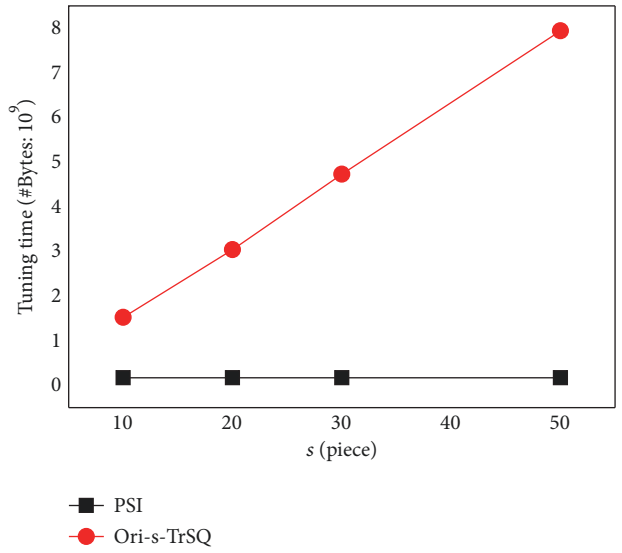

Figure 12: s-TrSQ with different values of s.

variable of the x-axis is the number of s trajectories including the trajectory of the q, which is set to 10, 20, 30, and 50 for comparison.

Figure 12 shows the variations in tuning time according to the size of s. The performance of PSI is higher by 96.1% than that of Ori-s-TrSQ. This is because the Ori-s-TrSQ must tune to all grids included in s paths.

Figure 13 shows the variations in tuning time according to the length of Tr_{dist}. The variable of the x-axis is Tr_{dist}, and the length of Tr_{dist} is set to 10, 50, 200, and 300km. As the length of Tr_{dist} increases, the tuning time of PSI increases at a fixed low rate, whereas the tuning of the Ori-s-TrSQ sharply increases. In the case of PSI, only the POIs in the grids included in the path of the q need to be received. Therefore, the performance of PSI improved by 94.9% on average compared that of Ori-s-TrSQ.

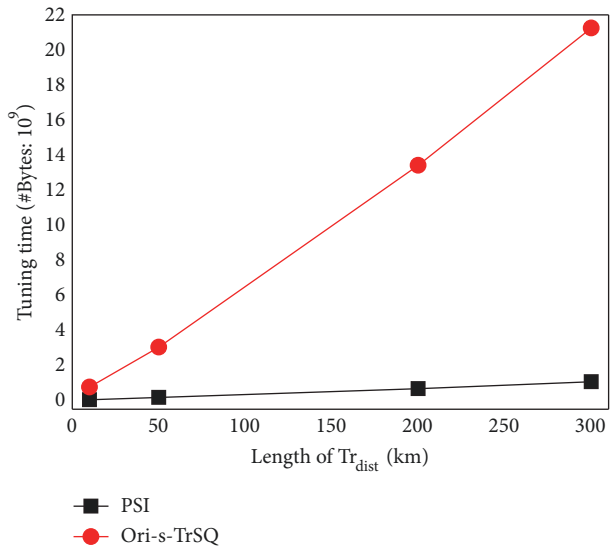

Figure 13: s-TrSQ with different lengths of Tr_{dist}.

7. Conclusions

In this study, we proposed PSI which can selectively tune to only the data desired by the q while protecting the location of the q. Furthermore, we proposed a general index structure applicable to the conventional location protection method for PSI. Finally, the tuning of unnecessary data and the battery consumption of the device were experimentally reduced by selectively tuning to the data of the objects to be received by the server, compared to the conventional method. In the future, we plan to research a space query processing method considering both the type and location of POI.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

Doohee Song, Moonbae Song, and Kwangjin Park declare that there are no conflicts of interest regarding the publication of this manuscript.

Acknowledgments

This paper was supported by Wonkwang University in 2018.

References

[1] C.-Y. Chow and M. F. Mokbel, "Trajectory privacy in location-based services and data publication," *ACM SIGKDD Explorations Newsletter*, vol. 13, no. 1, pp. 19–29, 2011.

[2] A. R. Beresford and F. Stajano, "Location privacy in pervasive computing," *IEEE Pervasive Computing*, vol. 2, no. 1, pp. 46–55, 2003.

[3] K. Park and P. Valduriez, "A hierarchical grid index (HGI), spatial queries in wireless data broadcasting," *Distributed and Parallel Databases*, vol. 31, no. 3, pp. 413–446, 2013.

[4] K. G. Shin, X. Ju, Z. Chen, and X. Hu, "Privacy protection for users of location-based services," *IEEE Wireless Communications Magazine*, vol. 19, no. 1, pp. 30–39, 2012.

[5] B. Niu, X. Zhu, W. Li, H. Li, Y. Wang, and Z. Lu, "A personalized two-tier cloaking scheme for privacy-aware location-based services," in *Proceedings of the 2015 International Conference on Computing, Networking and Communications, ICNC 2015*, pp. 94–98, Garden Grove, CA, USA, 2015.

[6] D. Song and K. Park, "A partial index for distributed broadcasting in wireless mobile networks," *Information Sciences*, vol. 348, no. 20, pp. 142–152, 2016.

[7] B. Niu, Q. Li, X. Zhu, G. Cao, and H. Li, "Enhancing privacy through caching in location-based services," in *Proceedings of the 34th IEEE Annual Conference on Computer Communications (IEEE INFOCOM '15)*, pp. 1017–1025, IEEE, Kowloon, Hong Kong, May 2015.

[8] B. Niu, X. Zhu, H. Chi, and H. Li, "3PLUS: Privacy-preserving pseudo-location updating system in location-based services," in *Proceedings of the 2013 IEEE Wireless Communications and Networking Conference, WCNC 2013*, pp. 4564–4569, April 2013.

[9] M. F. Mokbel, C. Y. Chow, and W. G. Aref, "The new Casper: query processing for location services without compromising privacy," in *Proceedings of the 32nd International Conference on Very Large Data Bases*, pp. 763–774, 2006.

[10] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan, "Private queries in location based services: anonymizers are not necessary," in *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD '08)*, pp. 121–132, 2008.

[11] R. Schlegel, C.-Y. Chow, Q. Huang, and D. S. Wong, "User-defined privacy grid system for continuous location-based services," *IEEE Transactions on Mobile Computing*, vol. 14, no. 10, pp. 2158–2172, 2015.

[12] D. Song, J. Sim, K. Park, and M. Song, "A privacy-preserving continuous location monitoring system for location-based services," *International Journal of Distributed Sensor Networks*, Article ID 815613, pp. 1–10, 2015.

[13] B. Niu, Q. Li, X. Zhu, G. Cao, and H. Li, "Achieving k-anonymity in privacy-aware location-based services," in *Proceedings of the IEEE International Conference on Computing, Networking and Communications*, pp. 754–762, IEEE, Toronto, Canada, 2014.

[14] H. Zhao, J. Wan, and Z. Chen, "A novel dummy-based KNN query anonymization method in mobile services," *International Journal of Smart Home*, vol. 10, no. 6, pp. 137–154, 2016.

[15] F. Li, S. Wan, B. Niu, H. Li, and Y. He, "Time obfuscation-based privacy-preserving scheme for location-based services," in *Proceedings of the 2016 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, Doha, Qatar, April 2016.

[16] B. Niu, S. Gao, F. Li, H. Li, and Z. Lu, "Protection of location privacy in continuous LBSs against adversaries with background information," in *Proceedings of the International Conference on Computing, Networking and Communications, ICNC 2016*, pp. 1–6, February 2016.

[17] R. Paulet, M. G. Kaosar, X. Yi, and E. Bertino, "Privacy-preserving and content-protecting location based queries," *IEEE International Conference on Data Engineering*, vol. 26, pp. 44–53, 2012.

[18] R. Paulet, M. G. Kaosar, X. Yi, and E. Bertino, "Privacy-preserving and content-protecting location based queries," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 5, pp. 1200–1210, 2014.

[19] X. Yi, R. Paulet, E. Bertino, and V. Varadharajan, "Practical k nearest neighbor queries with location privacy," in *Proceedings of the 30th IEEE International Conference on Data Engineering (ICDE '14)*, pp. 640–651, IEEE, Chicago, Ill, USA, April 2014.

[20] X. Yi, R. Paulet, E. Bertino, and V. Varadharajan, "Practical Approximate k Nearest Neighbor Queries with Location and Query Privacy," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 6, pp. 1546–1559, 2016.

[21] B. Niu, Z. Zhang, X. Li, and H. Li, "Privacy-area aware dummy generation algorithms for location-based services," in *Proceedings of the IEEE International Conference on Communications*, pp. 957–962, Sydney, Australia, June 2014.

[22] P.-R. Lei, W.-C. Peng, I.-J. Su, and C.-P. Chang, "Dummy-based schemes for protecting movement trajectories," *Journal of Information Science and Engineering*, vol. 28, no. 2, pp. 335–350, 2012.

[23] T. Hara, A. Suzuki, M. Iwata, Y. Arase, and X. Xie, "Dummy-Based User Location Anonymization under Real-World Constraints," *IEEE Access*, vol. 4, pp. 673–687, 2016.

[24] Y. Elmehdwi, B. K. Samanthula, and W. Jiang, "Secure k-nearest neighbor query over encrypted data in outsourced environments," in *Proceedings of the 30th IEEE International Conference on Data Engineering (ICDE '14)*, pp. 664–675, April 2014.

[25] W. K. Wong, D. W. Cheung, B. Kao, and N. Mamoulis, "Secure kNN computation on encrypted databases," in *Proceedings of the ACM SIGMOD Int. Conf. Manage. Data Eng*, pp. 139–152, July 2014.

[26] C.-Y. Chow, M. F. Mokbel, and X. Liu, "A peer-to-peer spatial cloaking algorithm for anonymous location-based services," in *Proceedings of the 14th Annual ACM International Symposium on Advances in Geographic Information Systems (ACM-GIS '06)*, pp. 171–178, ACM, November 2006.