A Privacy-Aware Location Cloaking Technique Reducing Bandwidth Consumption in Location-Based Services

Miyoung Jang
Dept. of Computer Engineering
Chonbuk National University
Jeon-ju, Republic of Korea
+84 63 270 2418
brilliant@jbnu.ac.kr

Min Yoon
Dept. of Computer Engineering
Chonbuk National University
Jeon-ju, Republic of Korea
+84 63 270 2418
myoon@jbnu.ac.kr

Hyeong-il Kim
Dept. of Computer Engineering
Chonbuk National University
Jeon-ju, Republic of Korea
+84 63 270 2418
melipion@jbnu.ac.kr

Jae-Woo Chang
Dept. of Computer Engineering
Chonbuk National University
Jeon-ju, Republic of Korea
+84 63 270 2418
jwchang@jbnu.ac.kr

ABSTRACT
The explosive growth of location-detection devices, such as GPS (Global Positioning System), continuously increases users' privacy threat in location-based services (LBSs). However, in order to enjoy such services, the user must precisely disclose his/her exact location to the LBS. So, it is a key challenge to efficiently preserve user's privacy while accessing LBS. For this, the existing method employs a 2PASS cloaking scheme hides the actual user location and reduces bandwidth consumption. However, it suffers from privacy attack since the 2PASS does not actually consider user distribution in the cloaking area.. Therefore, it is required to preserve user privacy by utilizing k-anonymity property. So, we, in this paper, propose a weighted adjacency graph based k-anonymous cloaking technique that can ensure users privacy protection and also reduce bandwidth usages. Our cloaking approach efficiently supports k-nearest neighbor queries without revealing the private information of the query initiator. We demonstrate via experimental results that our algorithm yields much better performance than the existing one.

Categories and Subject Descriptors
H.3.3 [**Information Search and Retrieval**]: Search process

General Terms
Algorithms, Performance, Design, Experimentation, Security

Keywords
Location-based services (LBS); location privacy; cloaking; bandwidth; k-anonymity; weighted adjacency graph

ACM SIGSPATIAL QUeST'12, November 6, 2012. Redondo Beach, CA, USA

1. INTRODUCTION
Location-based services allow users to connect with others based on their current locations. In most cases, people use their positioning devices (i.e., iPhone, Android, Blackberry) to find out his/her location like restaurants, bars and stores they visit. However, frequent and continuous accesses to the services expose users to privacy risk. A location-based service provider might be able to collect private and delicate information about a user's choices and habits from the user's location. For example, for a k-nearest neighbor query (kNN), the user (u) requests to LBS about the nearest dental clinic from his/her location and based on u's location LBS server returns nearest dental clinic name to u. For this LBS server can infer user health conditions that might be harmful for user's privacy. Due to an increasing awareness of privacy risks, users might desist from accessing LBSs, which would prevent the proliferation of these services [1, 2].

Current research aligns on developing techniques to elaborate on k-anonymity [3-18] that preserve user privacy during the access of LBSs. At first, L. Sweeny proposed k-anonymity [20] approach which tries to blur a user's location among k-1 users. So location privacy mechanism requires cloaking region (CR) (i.e., basically circular or rectangular region) for achieving k-anonymity requirement. In case of trusted third party based (TTP-based) [3-12] techniques, they use a location cloaker (LC) based on privacy metric (i.e., k-anonymity, the area of this region must exceed a threshold etc.) to calculate CR and to ensure k-anonymity. TTP-free techniques satisfy k-anonymity by using one of two methods. Firstly, collaboration-based methods [13-18] are to satisfy k-anonymity by collaborating with neighboring users and calculate their common CR. Secondly, private information retrieval (PIR) based method [19] uses standard public key cryptography but it tends to incur high computational and communicational overhead.

Most of the cloaking techniques enclose more non-result objects with original result due to the achievement of a user's privacy. As shown in Figure 1, the user sets range with his/her accurate location and requests for the nearest clinic. The LBS server returns non result objects (clinics) C_1, C_2 and C_3 with the actual result C to the user. However, larger result set size preserves more

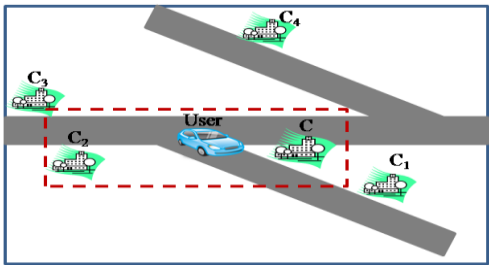

Figure 1. Range Nearest Neighbor Query

privacy, but consumes more network bandwidth and device battery as well. Furthermore, the crucial matter is how to minimize the number of non-result objects during cloaking period. Some researches [3, 6-12] indirectly minimize the bandwidth by minimizing the size of cloaking region for different privacy metrics.

The location cloaking approach called 2PASS (2-Phase Asynchronous Search) [21] has been proposed that minimize the bandwidth usages as well as preserve user privacy. Basically, 2PASS follows client-server model. It is based on a notion of Voronoi cells and each cell contains one object that is the nearest neighbor of any point in its cell. The user can fix the cloaked area to the Voronoi cell of the nearest neighbor object, if he/she knows the Voronoi cells in advance. 2PASS is able to save the bandwidth usages compared to the Range Nearest Neighbor (RNN) [22] approach based on this concept. For this, it processes a kNN query in two steps. In the first step, the user requests the Voronoi cell information corresponding to the query. In the second step, it selects objects to request. Although 2PASS optimizes the bandwidth usages, it suffers from privacy attack.

In this paper, we propose a weighted adjacency graph [21] based k-anonymous cloaking technique that can reduce bandwidth usages and provide protection to user. We follow user-cloaking-server model [3-12] where the trusted third party (location cloaker, LC) performs location cloaking for the user. Our algorithm computes a kNN query in three phases. In the first phase, the user requests the location cloaker (LC) and LC requests the WAG information corresponding to the query. LC selects objects to request in- the second phase and returns the actual result to the user in the third phase. We also include k-anonymity property for enhancing users' privacy while accessing LBSs. Our contributions can be summarized as follows:

- **We propose a weighted adjacency graph based k-anonymous cloaking technique to reduce bandwidth usages of requested services**
- **We use location cloaker between user and LBS server to handle cloaking operation**
- **We also consider k-anonymity property for enhancing user's privacy**

The rest of this paper is organized as follows. In Section 2, we discuss related cloaking methods. We discuss our detailed system architecture and propose a weighted adjacency graph based k-anonymous cloaking technique in Section 3. Section 4 is devoted to experimental results. Finally, we conclude our work with future direction in Section 5.

2. RELATED WORK

Recently, considerable research interest has focused on preventing identity inference in location-based services. The main concern is to allow the mobile user to request services without compromising his/her privacy. We classify these techniques into three main groups: pseudonym, dummy and cloaking

2.1 Pseudonym-Based Technique

Pseudonym combines the location and user identity called Mix zone [23]. For this, the server only receives the location without the user identity. B. Palanisamy, et.al[29] consider multiple factors in the mix-zone approach, such as the geometry of the zones, the statistical behavior of the user population, the spatial constraints on movement patterns of the users, and the temporal and spatial resolution of the location exposure. However, such a technique is limited to those location-based services that do not require the user's identity.

2.2 Dummy-Based Technique

In dummy based technique [24-26], it generates false user locations called dummies and combine them with the actual user location into the request. The user can refer to the list corresponding to his/her actual location, i.e., filter out the other lists. However, the server may infer the actual user location from dummies after monitoring long-term movement patterns of the user. Kido et al. [24] propose dummy-based technique but shows poor performance in a real environment. You et al.[25] improve this technique by generating consistent movement patters for dummies in a long run. Suzuki et al. [26] enhance it by generating scattered dummies around the user based on the user's current location and previous dummies locations. They also consider dummies locations according to the real road information. Ghinita et al. [19] proposed Private Information retrieval (PIR) to process nearest neighbor queries. Based on cryptographic techniques, they guarantee that an adversary cannot infer the user's location within polynomial time of a security parameter (e.g., key length). However, query processing is particularly slow, and the technique is inapplicable to range and kNN queries (with $k > 1$).

2.3 Cloaking-Based Technique

In cloaking based technique, it generates blur area (circular or rectangular) that encloses an actual query issuer with other users based on his/her privacy requirement (e.g., *k-anonymity*, granularity metric etc.). By this, the user can hide his/her location from adversary or LBS server. For example, a user requests LBS with 3-anonymity through by trusted third party (Location cloaker) or group head (i.e., Distributed system) in Figure 2. Then location cloaker (LC) or group head (GH) makes the cloaking area and sends to the LBS for results set. Gruteser et al. [3] introduced a cloaking technique known as Interval Cloak, where they calculate the cloaking region based on a quad tree method. The main idea is

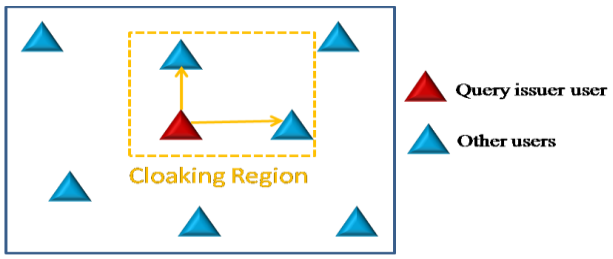

Figure 2. Cloaking based technique(based on k-anonymity)

that it recursively partitions the space into four equally-squared regions until a user fits in a quadrant to satisfy *k-anonymity*. Gedik et al. [4] proposed the CliqueCloak algorithm that relies on the ability to locate a clique in a graph to perform location cloaking. This algorithm is expensive and performs poorly when k is large.

Mokbel et al. [7] proposed Casper that addresses location anonymization using a pyramid data structure and allows the system to quickly locate anonymized areas. However, Casper produces a large cloaking area and yields poor quality of services (QoS). Ghinita et al. [9] proposed HilASR that satisfies a *reciprocity* condition in every k-anonymous region. Bamba et al. [10] presented a grid-based cloaking method that introduced l-diversity concept. Mokbel et al. [13] proposed CLOAKP2P that constructs a cloaking area by considering the neighboring users of the query. However, it fails to provide privacy protection when there are many users. Ghinita et al. [14] proposed PRIVE in which users form a cluster and maintain a hierarchical overlay network resembling a distributed B+ tree. However, it may suffer from a slow response time since the nodes in a root level constitute a potential bottleneck. Liu et al. [11] presented the X-Star based cloaking algorithm that can achieve the optimal query processing cost and high privacy protection for the users. It groups neighboring queries into a cloaking star structure and adjusts the result cloaking stars to satisfy the privacy requirement of users. In addition, it merges neighboring stars into a super-star structure to meet the privacy requirements of each individual user. However, X-star has a low successful anonymization rate. To address this issue, Hossain et al. [12] proposed H-Star based cloaking algorithm that can achieve high successful rate and reduce query processing cost. To the best of our knowledge, there exists only one research on result-aware location cloaking approach (called 2PASS), proposed by H. Hu and J. Xu [21]; based on client-server model that minimizes the number of requested objects directly. For this, they introduced (i) weighted adjacency graph (WAG) that stores Voronoi cell information and (ii) WAG-tree indexing for computing the objects to request from the server.

Basically, 2PASS [21] is based on the notion of Voronoi cells and each cell contains one object that is the nearest neighbor of any point in its cell. For example, Figure 3(a) shows an example of Voronoi diagram with 6 objects such as a, b, c, d, e and f. The solid lines show the borders of the Voronoi cells, and the dotted lines connect the adjacent cells' objects. The dotted lines are called Delaunay triangulation of the space because these lines

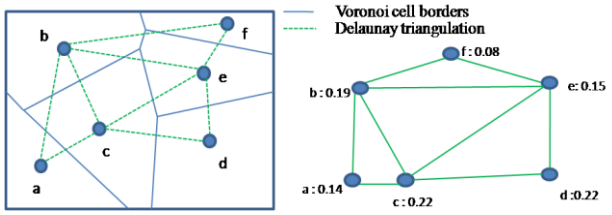

Figure 3. (a) Voronoi Diagram and (b) WAG

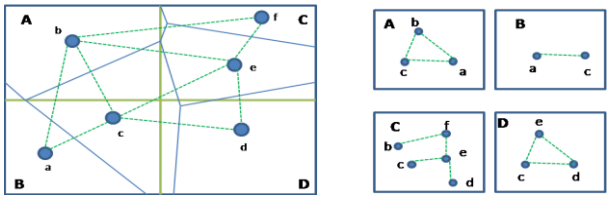

Figure 4. WAG Snippets

divide the space into partitions of spatial shape-triangles. However, if the space is bounded, these dotted lines might not form a closed Delaunay triangulation because two cells might share a border at somewhere beyond the bounded space. For example, in the rectangular space of Figure 3(a), there is no dotted line between two objects a and d, because their Voronoi cells are not adjacent in this space, although they share a border outside the space. If the user knows the Voronoi cells in advance, he/she can set the cloaked region to the Voronoi cell of the nearest neighbor objects.

To access the Voronoi cell information, they develop a weighted adjacency graph (WAG). WAG is a weighted undirected graph that stores the Voronoi diagram and Delaunay triangulation. For example, in Figure 3(b), each vertex in this graph denotes an object, and each edge denotes a line in the Delaunay triangulation. Each vertex is also assigned a nonnegative weight. The specialty of this graph is to notify that the WAG vertices are weighted based on Voronoi cell area size. User can compute out the objects to request from the server by using WAG-tree index. The criteria of object selection are a combination of the following: a) the sum of the areas of Voronoi cells from the selected objects must exceed τ; b) the genuine nearest neighbor o* must be selected; and c) these Voronoi cells must be connected, i.e., no cell is isolated from the rest of the cells.

To reduce computational overhead, 2PASS [21] also proposes to partition the entire WAG into WAG snippets of reasonable size so that the user receives only the snippets surrounding the query location. For example, in Figure 4, the four snippets are obtained by partitioning the space into four sub-spaces A, B, C and D of equal widths and heights and computing their WAG's, respectively. The weight of an object in a WAG snippet is set to its Voronoi cell area that resides in this subspace. WAG snippets can be joined to become the WAG of the union of these subspaces. The join is done by merging the vertices corresponding to the same object and assigning its new weight as the sum of the weights of these vertices. WAG-tree follows a top down recursive fashion. For each node, the algorithm maintains objects whose Voronoi cells in the whole space overlap this sub-space. Since each such object is the nearest neighbor of some point in this sub-space, it is essentially the range nearest neighbor (RNN) of this sub-space. The algorithm recursively computes range nearest neighbor of a child node until satisfying the certain criterion, e.g., the snippet area is larger than user defined threshold and Figure 5 shows a WAG-tree and snippet pointed by it. 2PASS is able to save bandwidth usage compared with others by returning less number of non-result objects. However, it suffers from privacy risk.

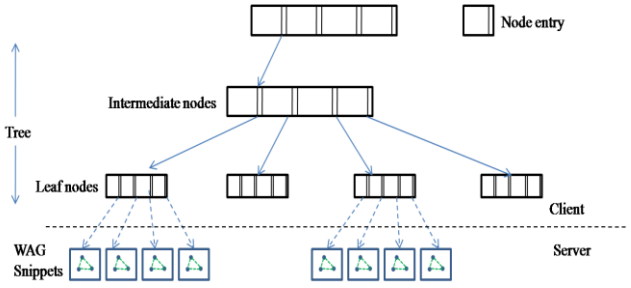

Figure 5. WAG tree

3. K-ANONYMOUS CLOAKING TECHNIQUE BASED ON WEIGHTED ADJACENCY GRAPH

In this section, we first will present system architecture of our work. In addition, we introduce our proposed work, a weighted adjacency graph (WAG) based k-anonymous cloaking technique (k-WAG).

3.1 System Architecture

We first describe our system architecture that is based on client-cloaker-server architecture. Our approach to address the range nearest neighbor queries is based on the weighted adjacency graph (WAG) that encloses Voronoi diagram.

Using WAG we propose k-anonymous cloaking framework that adopt trusted third party model. This choice is made by the following reasons. First, the third party (location cloaker) acts as a mediator between user and the server, and performs location cloaking. Second, the existing anonymous services [28] can able to support trusted third party based system. For this, anonymous services provide anonymous use of the internet/ e-mail based services without revealing the user's real e-mail. Thirdly, the existence of information security techniques and system architectures support trusted third party services. Finally, the existing methods ensure that a third party will honor the user privacy requirements. In this paper, we focus on granularity metric like 2PASS [21], so location cloaking generates a random cloaked region that encloses the user's genuine location and whose area is no less than a user specified threshold τ. Figure 6 shows the system architecture of our method. In the first phase, user sends a query to location cloaker (LC) and LC requests LBS for *iWAG* (Improved WAG) information, where the weight of a vertex is based on the area of the corresponding Voronoi cell and the number of users on that cell. In the second phase, LC selects objects from the *iWAG* (e.g. three restaurants Pizza Hut, MacDonald and KFC) and requests them for their complete contents (e.g., customer reviews and reservation status etc.). In third phase, LC sends the exact answer to the user. In our system architecture, LC is responsible for cloaking procedure instead of user. Thus, the LBS server may not be used to infer a query issuer user.

Now, we describe the basic concepts of our work. Consider that there are a set of n objects and that Voronoi diagram divides a space into disjoint polygons [27]. Each polygon is called a Voronoi cell and corresponds to one object. The nearest neighbor of any point inside a cell is the corresponding object. The boundaries of the Voronoi cells, called Voronoi cell areas, are the set of locations that can be assigned to more than one object. The Voronoi cells that share the same areas are called adjacent Voronoi cells and their objects are called adjacent objects (Figure 7(a)).

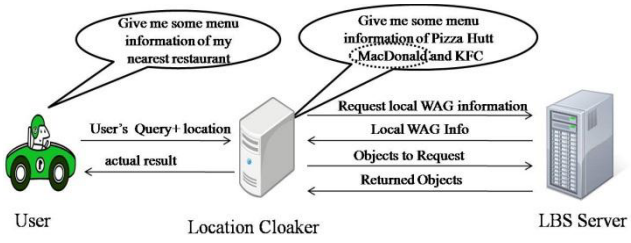

Figure 6. System Architecture

To access the Voronoi cell information, 2PASS [21] present WAG that stores Voronoi diagram and Delaunay triangulation where each vertex v is assigned a non-negative weight wi based on Voronoi cell area size. But in our work, each vertex is also assigned a nonnegative weight. The difference is that the WAG vertices are weighted based on the size of Voronoi cell area and the number of users on that cell. So, we call this improved WAG (*iWAG*).

3.2 K-WAG Algorithm

In this section, we propose weighted adjacency graph based k-anonymous cloaking technique (k-WAG) to solve the problems of the 2PASS [21]. In k-WAG, user sends a query with privacy requirement to location cloaker (LC) and LC requests the objects (including the genuine nearest neighbor (NN) together with other non-result objects) based on the Voronoi cell information to satisfy the privacy requirement on the cloaked region. Our work is unique in that the LC controls what objects to request from the server so that their total number (i.e., the overall bandwidth) is minimized. To minimize the object number while still meeting the privacy threshold τ and k-anonymity requirement, the criteria of object selection are a combination of the following: (ì) the sum of the areas of Voronoi cells from the selected objects must exceed τ and the number of users ≥ k on that cell; (ii) the genuine nearest neighbor o* must be selected; and (iii) these Voronoi cells must be connected, i.e., no cell is isolated from the rest of the cells. The last criterion guarantees that the cloaked region is a single region, which is a common assumption in all existing location cloaking approaches. Besides, the single-region assumption not only adapts to most location-based services which readily accept a single location as the input, but also alleviates some security problems. For example, a single region is more resilient than isolated regions against background or domain knowledge attacks. With the *iWAG*, the object selection is equivalent to finding a sub graph that satisfies the following criteria: ì) the sum of the weights of vertices in the sub graph must exceed τ and the number of user ≥ k on that cell; ìì) o* must be in the sub graph; and ìì) this sub graph must be a connected component. Now, we describe the *iWAG* generation procedure.

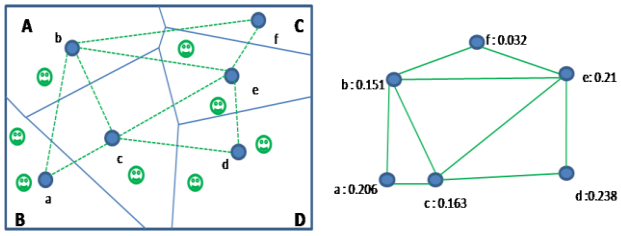

Figure 7. (a) Voronoi Diagram and (b*) iWAG* with user

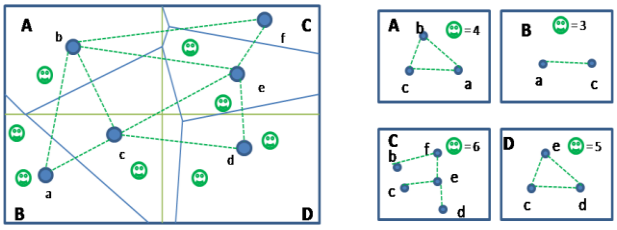

Figure 8. *iWAG* Snippets

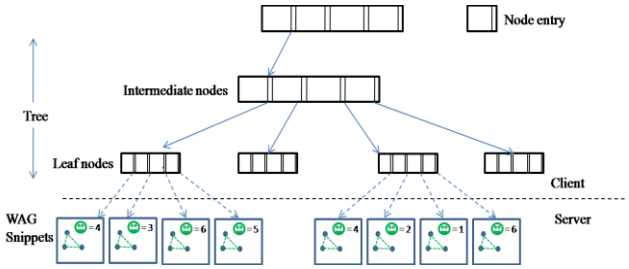

Figure 9. *iWAG* Tree

For this, we give the weight (w) for each object (Vw) based on Voronoi cell area size (Va) and number of user (Un) on that Voronoi cell. We set the priority for Voronoi cell area size and the number of user in that cell. For example, if we consider the total priority, p = (α + β) = 1, then the preference of the number of user (β) is get priority than the preference of Voronoi cell area size (α). Therefore, the following equation holds true,

$$V_w = (V_a \times \alpha) + \left(\frac{U_n}{total\ U_n} \times \beta\right) \text{.....(1)}$$

Figure 7(a) shows Voronoi diagram with eight users. We calculate objects weight based on equation (1). For example, if we consider object a, then weight of aw =0.206. By this, we get all of objects' weight as shown in Figure 7(b). Next, we describe three algorithms. Firstly, Algorithm 1 describes the *iWAG* generation procedure. It calculates each Voronoi cell area size in line 1 and then computes how many users exist on that cell in line 2. Based on (line 1 and 2), it calculates vertex weight by using equation (1) in line 3. It sets all vertex weight in line 4 and line 5 ends the algorithm. Actually, we consider the total vertex weight is 1. Our objective is to find out the valid weight connected component based on *iWAG*. For this, we follow approximate minimum valid weight connected component (MVWCC) algorithm [21].

In the first phase of our work, the client sends a query to LC and LC requests for the *iWAG* of its neighborhood. If the object data set is not huge, the LC can request the *iWAG* of the entire space and cache it to avoid re-request for subsequent queries. However, for a practical data set with thousands or even millions of objects, it is impractical to request and cache the entire *iWAG* due to the following reasons: (1) the WAG is huge in terms of memory footprint; (2) the cached WAG is vulnerable to even a slight change of the dataset, e.g., and object deleted/inserted/moved; (3) the computational cost of the entire WAG is high. As such, we also partition the entire *iWAG* into *iWAG* snippets (i.e., as like [21]) of reasonable sizes so that the LC receives only the snippet(s) surrounding the query location. Actually, an *iWAG* snippet is the *iWAG* of a subspace.

For example, eight users and the four snippets are obtained by partitioning the space into four subspaces A, B, C and D of equal size and computing their *iWAG*'s, as shown in figure. It is noteworthy that an object being outside of a subspace can still appear in the *iWAG* snippet of this subspace, as long as the Voronoi cell of this object in the *iWAG* of the entire space overlaps this subspace, e.g., objects a and c in snippet A. The weight of an object in a *iWAG* snippet is set to its Voronoi cell area and the number of users that resides in this subspace. *iWAG* snippets can be joined to become the *iWAG* of the union of these subspaces. The join is done by merging the vertices corresponding to the same object and assigning its new weight as the sum of the weights of these vertices. In order for the LC to know which a hierarchical index called *iWAG*-tree construction algorithm like a quad tree. This index recursively partitions the space into quadrants until a certain criterion is met. Each entry in its leaf node points to a WAG snippet.

Secondly, **Algorithm 2** shows the pseudo code of the *iWAG*-tree construction process. This tree follows top-down approach. At first, it checks the condition in line 1. We set two conditions: (i) area range and (ii) object's (Point of Interest) number. It builds bounding areas (snippets) if satisfying certain conditions in line 2. Otherwise it partitions the whole space into four parts in line 3-4. The algorithm maintains objects whose Voronoi cells in the whole space overlap this sub-space in line 5-6; it is essentially the k-range nearest neighbor (kRNN) of this space. The algorithm recursively computes range nearest neighbor of a child node until satisfying the certain criterion in line 7. Figure 9 shows the *iWAG*-tree and snippet pointed by it.

***iWAG* Generation Algorithm**

Input: V_a : Voronoi cell

U_n : number of user for each Voronoi cell

Output: V_w : vertex weight

1: compute area size for each Voronoi cell (V_a)
2: calculate number of user on that cell (U_n)
3: give vertex weight (V_w)
4: set WAG (sum of the vertex weight)

Algorithm 1. *iWAG* Generation

***iWAG* tree Construction Algorithm**

Input: R : the set of RRNs

T: the root node of tree

F: the termination condition

Output: T : the Snippet rooted at T

1. If F (T) is true then
2. Build snippet using R
3. else
4. partition T into 4 quadrants t_1-t_4
5. for each t_i do
6. compute the RNN set R_i from R
7. call Snippet generation algorithm (R_i, t_i, F)

Algorithm 2. *iWAG* Tree Construction

***iWAG* tree Construction Algorithm**

Input: R : the set of RRNs

T: the root node of tree

F: the termination condition

Output: T : the Snippet rooted at T

1. If F (T) is true then
2. Build snippet using R
3. else
4. partition T into 4 quadrants t_1-t_4
5. for each t_i do
6. compute the RNN set R_i from R
7. call Snippet generation algorithm (R_i, t_i, F)

Algorithm 3. K-WAG Query Processing

Finally, **Algorithm 3** summarizes the whole procedure of our work, named k-WAG query processing algorithm. The whole *iWAG* tree is sent to LC during the system initialization time. Based on the kNN, the LC traverses the *iWAG* tree and finds out the snippet that contains the query point (line 1-3). And the LC matches the privacy requirements (k-anonymity, the area size (τ) etc) that are sent by the user (line 4-5). If the area of this snippet is still smaller than the user specified requirements, the user will locate the lowest-level child node of this snippet whose area exceed privacy requirements (line 6). And the user requests all snippets rooted at this node, called host snippets (line 7). The LC then adds the received host snippets into a single *iWAG* and calculates the minimum valid connected components by using MVWCC algorithm [21] (line 8-9). In this process, LC does cloaking procedure instead of user and LC does not provide any location information or privacy requirements of user to the server.

4. PERFORMANCE ANALYSIS

In this section, we present the performance results of our location cloaker and query processing algorithm. We implemented our cloaking technique and query processing algorithms to evaluate the performance of our approach. Our main objectives are to observe the influence of performance factors on the system and to test the feasibility of our technique.

4.1 Experimental Setup

Our system consists of three main components; the mobile user, the location cloaker and the LBS server. For the mobile user, we assume that each Voronoi cell encapsulates at least 2 users. We implement the location cloaker as a new module for interacting with mobile users to anonymize queries in the framework. Our privacy protected query approaches are also implemented inside the framework as new functions and play the role of LBS server. We use the real data set of Northern East America (NE) that contains 119,898 point of interest (POIs). For easy presentation, the coordinates of these objects are normalized to a unit square.

Table 1. Experimental environment

CPU	Intel® Xeon® CPU 2.00 GHz
Memory	2 GB
Simulator	Visual Studio 2010
OS	Windows XP

Table 2. Simulation parameter setting

Parameters	**Range**
Total User	239,668
Query Number	70,000
Granularity Threshold (τ)	0.000001, 0.00001, 0.0001, 0.001
Maximum Area of WAG Snippet	0.001
K-Anonymity	2, 4, 6, 8, 10
Average Number of User in each Cell	2

Table 1 represents the experiment environment. The query load consists of 70,000 queries that are uniformly distributed in the unit square. We compare our work with the exiting approach 2PASS [21] in terms of response time and bandwidth size. The response time can be defined as how quickly the server returns the result set after receiving the query. The bandwidth can be defined as page size that encloses objects. The parameter settings are summarized in Table 2.

4.2 Query Processing Time

We vary the user specified threshold (τ) value from 0.000001 to 0.001 and the threshold value reflects the response time that means query processing time. As for query performance, the query processing time of 2PASS and our scheme are near to similar when τ is equivalent to small value. But, the query processing time of 2PASS is much higher than our scheme when τ becomes greater value. It is mainly due to the more number of objects it request. As a consequence, 2PASS also consumes more bandwidth than our scheme. Figure 10 demonstrates the query processing time with different τ value. Since a bigger threshold value usually contains more underlying network area, it takes longer to process. As shown in Figure 10, the increase of former metric is quite moderate until $\tau \leq 0.0001$. On the other hand, the latter metric linearly increases as τ grows. We test the impact of varying the number of *k-anonymity* with query processing time. We alter *k- anonymity* range from 2 to 10. As shown in Figure 11, the query processing time remains the same when we raise *k-anonymity* from 2 to 10. That means the proposed work fulfills the *k-anonymity* requirement without affecting the query processing time.

We also measure the query processing time with different vertex weight. We assume that each Voronoi cell consists of 1 to 3 user and set priority matrix based on number of user in each cell and Voronoi cell area size.

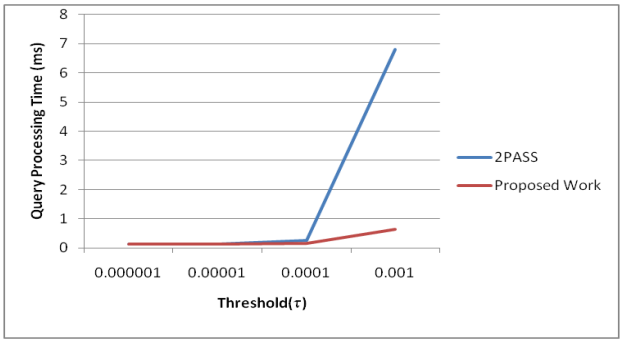

Figure 10. Query Processing Time vs. τ

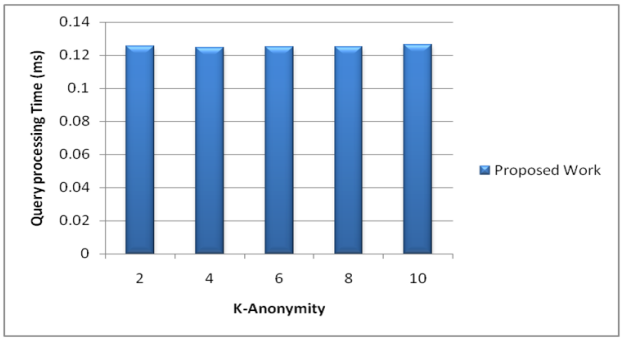

Figure 11. Query Processing Time vs. k-anonymity

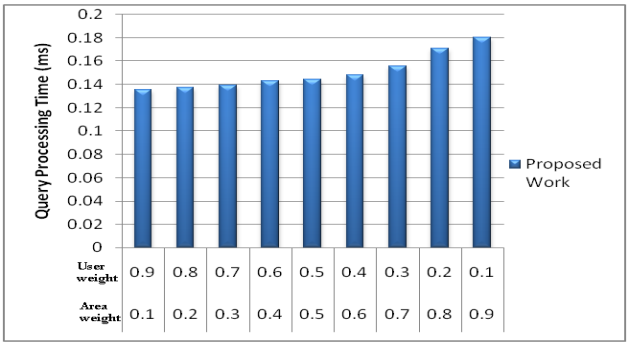

Figure 12. Query Processing time vs. V_w

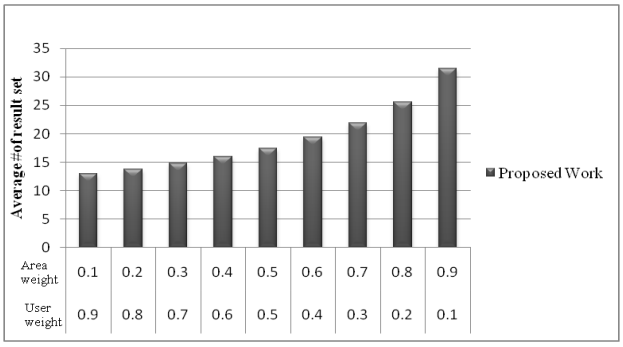

Figure 15. Average Number of Results vs. V_w

We consider equation 1 (Section 3.3) by varying user weight and area weight based on priority for calculating the vertex weight (V_w). Figure 12 depicts the query processing time with varying V_w. We note that the query processing time increases when V_w is based on less user priority. This is because the weight of vertex depends on not only cell area size but also user number in our scheme. If we give more priority to the cell area size, our scheme selects large area for covering the k-anonymity requirement of user during anonymization period. As a result, the query processing time is larger due to the return large number of POIs.

4.3 Bandwidth

The response time of 2PASS is larger than that of our scheme, which is mainly due to the more number of objects in our scheme. As a consequence, 2PASS also consumes more bandwidth than the proposed one. We calculate the bandwidth size based on average number of objects returned by varying with different threshold value (τ). Figure 13 depicts the result set size with different τ. We observe that a bigger τ value generates a larger candidate result set.

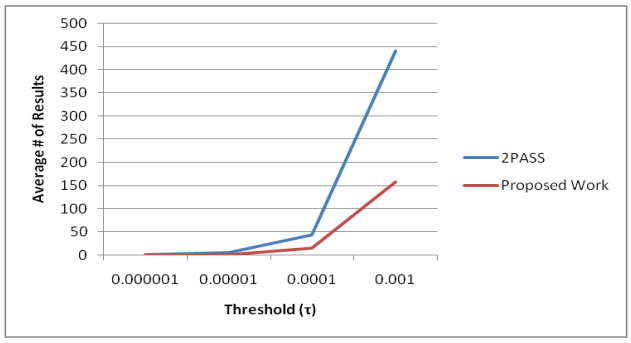

Figure 13. Average Number of Results vs. τ

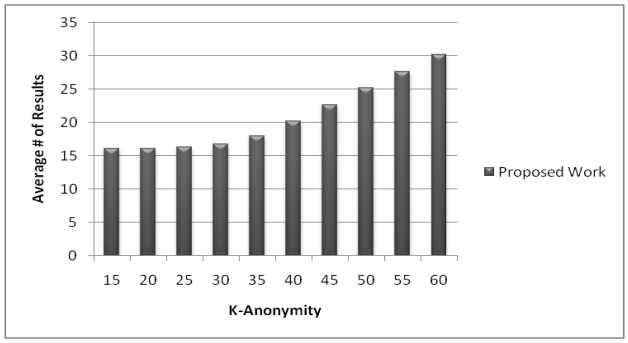

Figure 14. Average Number of Results vs. k-anonymity

Figure 14 shows the result set size with different *k-anonymity*. As expected, our scheme returns more candidate result set as *k-anonymity* increases. In fact, the increase of average number of result is quite similar until *k-anonymity* equals 25. We observe that the result set increases linearly when we raise k from 30 to 60. We analyze the performance of our scheme with regard to various vertex weight (V_w). The result shows that the number of average results set increases as the priority of user number becomes low in Figure 15. This is because the vertex weight is highly depend on the user number than the cell area size.

5. CONLCUSIONS

We identify the limitations of privacy-ware data access in location-based services. We propose a weighted adjacency graph based k-anonymous cloaking technique that allows k-anonymity property for providing the location privacy of all users in the network. Our technique follows third party based approach where the location cloaker handles cloaking process instead of user. We also minimize the bandwidth consumption by using *iWAG*-tree index from which the location cloaker can compute out the objects to request from the server. Through our experimental performance evaluations, we have shown that our cloaking method is much more efficient in terms of both response time and bandwidth consumption than the 2PASS.

In future work, we plan to extend our work beyond a larger geographical area. We also plan to conduct the extensive performance evaluations by our experiments with various data sets and study the behavior of our work when the user issues a series of requests within a short period.

6. ACKNOWLEDGMENTS

This work was supported by Business for Cooperative R&D between Industry, Academy, and Research Institute funded Korea Small and Medium Business Administration in 2012. (Grants No. C0055482) This research was also supported by Basic Science Research program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology(grant number 2012-0023800)

7. REFERENCES

[1] Privacy concerns a major roadblock for location-based services say survey. http//www.Govtech.com/gt/articles/104064, 2007.

[2] W. R. Muntz, T. Barclay, J. Dozier, C. Faloutsos, A. Maceachren, J. Martin, C. Pancake and M. Satyanarayanan, "IT Roadmap to a Geospatial Future," The National Academics Press, 2003.

[3] M. Gruteser and D. Grunwald, “Anonymous usage of Location-Based Services Through Spatial and Temporal Cloaking,” In Proceedings of the First ACM/USENIX International Conference on Mobile Systems, Application and Services (MobiSys), San fransisco, CA, USA, 2003.

[4] B. Gedik and L. Liu, “Protecting Location Privacy with Personalized k-Anonymity: Architecture and Algorithms,” IEEE Trans. Mobile Computing, vol. 7, no. 1, 2008, pp. 1-18.

[5] B. N. Schilit, J. I. Hong and M. Gruteser, “Wireless Location Privacy Protection,” IEEE Computer, 36(12): 135-137, 2003.

[6] B. N. Schilit, J. I. Hong and M. Gruteser, “Wireless Location Privacy Protection,” IEEE Computer, 36(12): 135-137, 2003.

[7] 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 International Conference Very Large Database (VLDB). 763–774, 2006.

[8] M. F. Mokbel, “Towards Privacy-Aware Location Based Database Servers,” In Proceeding of the 22nd IEEE International Conference on Data Engineering (ICDE) Workshop, Atlanta, Georgia, USA, 2006.

[9] P. Kalnis, G. Ghinita, K. Mouratidis and D. Papadias, “Preventing Location-Based Identity Inference in Anonymous Spatial Queries," IEEE Transactions on Knowledge and Data Engineering, v. 19, no. 12, p. 1719-1733, December 2007.

[10] B. Bamba, L. Liu and P. Pesti, T. Wang, “Supporting anonymous location queries in mobile environments with privacygrid,” In Proceeding of the International Conference World Wide Web, pp 237-246, April 2008.

[11] T. Wang and L. Liu, “Location Privacy Over Road Networks,” the International Conference Very Large Database (VLDB), 2009.

[12] A. Hossain, A. A. Hossain and J.W. Chang, “Spatial Cloaking Method Based on Reciprocity Property for Users’ Privacu in Road Network,” IEEE 11th International Conference on Computer and Information technology (CIT), page 487 – 490, 2011.

[13] C. Y. Chow, M.F. Mokbel and X. Liu, “A Peer-to-Peer Spatial Cloaking Algorithm for Anonymous Location-Based Service”, In Proceeding of Ann. ACM Int’l Symp. Advances in Geographic Information Systems (GIS), 2006.

[14] G. Ghinita, P. Kalnis and S. Skiadopoulos, “PRIVÉ: Anonymous Location-Based Queries in Distributed Mobile Systems,” In Proceeding of The International Conference World Wide Web (WWW), pp. 371-380, 2007.

[15] M. Monjur, S. I. Ahamed and S. H. Chowdhury, “ELALPS: A Framework to Eliminate Location Anonymizer from Location Privacy Systems,” In Proceeding of 33rd Annual IEEE International Computer Software and Applications Coference, 2009.

[16] T. Hashem and L. Kulik, “Safeguarding Location Privacy in Wireless Ad-hoc Networks,” In Proceeding of Ubicomp 2007: Ubiquitous Computing, vol 4717, pp. 372-390, 2007.

[17] G. Zhong and U. Hengartner, “Toward a distributed k-anonymity protocol for location privacy,” In Proceedings of the 7th ACM workshop on Privacy in the electronic society, CCS, pp- 33-38, 2008.

[18] A. Solanas and A. Mart´ınez-Ballest´e, “Privacy protection in locationbased services through a public-key privacy homomorphism,” In Proceeding of 4th European PKI Workshop: theory and practice, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 362 – 368, palma de Mallorca, Spain, 2007.

[19] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan, “Private Queries in Location Based Services: Anonymizers Are Not Necessary,” In Proceeding of ACM SIGMOD, 2008.

[20] L. Sweeney, “k-anonimity: A model for protecting privacy,” International Journal of Uncertainty, Fuzziness and Knowledge Based Systems 10(5), 557–570, 2002.

[21] H. Hu and J. Xu, “2PASS: Bandwidth-Optimized Location Cloaking for Anonymous Location-Based Services,” IEEE Transactions and Parallel on Distributed Systems, 2010.

[22] H. Hu and D. Lee, “Range Nearest Neighbor Query,” IEEE Trans. Knowledge and Data Eng., vol. 18, no. 1, pp. 78-91, 2006.

[23] A. Beresford and F. Stajano, “Location privacy in pervasive computing,” IEEE Pervasive Computing, 2(1): 46-55, 2003.

[24] H. Kido, Y. Yanagisawa and T. Satoh, “An anonymous communication technique using dummies for location based services,” In Proceeding of 2nd ICPS, pages 88-97, 2005.

[25] T. You, W. Peng, and W. Lee, “Protect Moving Trajectories with Dummies,” In Proceeding of International Workshop Privacy-Aware Location-Based Mobile Services, 2007.

[26] A. Suzuki, M. Iwata, Y. Arase, T. Hara, X. Xie and S. Nisho, “A user location anonymization method for location based services in a real environment,” In Proceeding of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems, 2010.

[27] M. Berg, M. Kreveld and M. Overmas, “Computational Geometry: Algorithm and Applications,” Springer-Verlag, 1997.

[28] http://www.mailshell.com/, 2009.

[29] B. Palanisamy, L. Liu, “MobiMix: Protecting Location Privacy with Mix-zones over Road Networks,” IEEE 27th International Conference on Data Engineering, 2011.