Practical Approximate k Nearest Neighbor Queries with Location and Query Privacy

Xun Yi, Russell Paulet, Elisa Bertino, *Fellow, IEEE*, and Vijay Varadharajan

Abstract—In mobile communication, spatial queries pose a serious threat to user location privacy because the location of a query may reveal sensitive information about the mobile user. In this paper, we study approximate k nearest neighbor (kNN) queries where the mobile user queries the location-based service (LBS) provider about approximate k nearest points of interest (POIs) on the basis of his current location. We propose a basic solution and a generic solution for the mobile user to preserve his location and query privacy in approximate kNN queries. The proposed solutions are mainly built on the Paillier public-key cryptosystem and can provide both location and query privacy. To preserve query privacy, our basic solution allows the mobile user to retrieve one type of POIs, for example, approximate k nearest car parks, without revealing to the LBS provider what type of points is retrieved. Our generic solution can be applied to multiple discrete type attributes of private location-based queries. Compared with existing solutions for kNN queries with location privacy, our solution is more efficient. Experiments have shown that our solution is practical for kNN queries.

Index Terms—Location based query, location and query privacy, private information retrieval, Paillier cryptosystem, RSA

1 Introduction

The embedding of positioning capabilities (e.g., GPS) in mobile devices facilitates the emergence of location-based services (LBS), which is considered as the next "killer application" in the wireless data market. LBS allows clients to query a service provider (such as Google or Bing Maps) in a ubiquitous manner, in order to retrieve detailed information about points of interest (POIs) in their vicinity (e.g., restaurants, hospitals, etc.).

The LBS provider processes spatial queries on the basis of the location of the mobile user. Location information collected from mobile users, knowingly and unknowingly, can reveal far more than just a user's latitude and longitude. Knowing where a mobile user is can mean knowing what he/she is doing: attending a religious service or a support meeting, visiting a doctor's office, shopping for an engagement ring, carrying out non-work related activities in office, or spending an evening at the corner bar. It might reveal that he is interviewing for a new job or "out" him as a participant at a gun rally or a peace protest. It can mean knowing with whom he/she spends time, and how often. When location data are aggregated it can reveal his/her regular habits and routines - and when he deviates from them.

- *X. Yi is with the School of Computer Science and IT, RMIT University, Melbourne, VIC 3001, Australia. E-mail: xun.yi@rmit.edu.au.*
- *R. Paulet is with the College of Engineering and Science, Victoria University, Melbourne, VIC 8001, Australia. E-mail: Russell.Paulet@vu.edu.au.*
- *E. Bertino is with the Department of Computer Science and Cyber Center, Purdue University, West Lafayette, IN 47907. E-mail: bertino@purdue.edu.*
- *V. Varadharajan is with the Department of Computing, Faculty of Science, Macquarie University, NSW 2109, Australia. E-mail: vijay.varadharajan@mq.edu.au.*

Manuscript received 21 Jan. 2015; revised 1 Jan. 2016; accepted 13 Jan. 2016. Date of publication 21 Jan. 2016; date of current version 27 Apr. 2016. Recommended for acceptance by J. Xu.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TKDE.2016.2520473

A 2010 survey conducted for Microsoft in the United Kingdom, Germany, Japan, the United States, and Canada found that 94 percent of consumers who had used location-based services considered them valuable, but the same survey found that 52 percent were concerned about potential loss of privacy.[1]

In this paper, we study approximate k nearest neighbor (kNN) queries where the mobile user queries the location-based service provider about approximate k nearest points of interest on the basis of his current location. In general, the mobile user needs to submit his location to the LBS provider which then finds out and returns to the user the k nearest POIs by comparing the distances between the mobile user's location and POIs nearby. This reveals the mobile user's location to the LBS provider.

There have been numerous techniques that can provide a certain degree of location privacy. These techniques mainly include

- Information access control [16], [35];
- Mix zone [3];
- k-anonymity [2], [4], [15]
- "Dummy" locations [13], [27], [34];
- Geographic data transformation [11], [12], [31], [32];
- Private Information Retrieval (PIR) [8], [21], [22].
- Two LBS servers [6], [26].

LBS queries based on access control, mix zone and k-anonymity require the service provider or the middleware that maintains all user locations. They are vulnerable to misbehavior of the third party. They offer little protection when the service provider/middleware is owned by an untrusted party. There have been private data inadvertently disclosed over the Internet in the past.

k-anonymity is initially used for identity privacy protection. It is generally inadequate for location privacy

1. http://www.microsoft.com/en-us/download/details.aspx?id=3250

protections, where the notion of distance between locations is important (unlike distances between identities). The effect of LBS queries based on k-anonymity depends heavily on the distribution and density of the mobile users, which, however, are beyond the control of the location privacy technique.

LBS queries based on dummy locations require the mobile user randomly to choose a set of fake locations, to send the fake locations to the LBS and to receive the false reports from the LBS over the mobile network. This incurs both computation and communication overhead in mobile devices. For the purpose of efficiency, the mobile user may choose less fake locations, but the LBS provider can restrict the user in a small sub space of the total domain, leading to weak privacy.

LBS queries based geographic data transformation are prone to access pattern attacks [30] because the same query always returns the same encoded results. For example, the LBS may observe the frequencies of the returned ciphertexts. Having knowledge about the context of the database, it can match the most popular plaintext POI with the most frequently returned ciphertext and, thus, unravel information about the query.

LBS queries based on PIR provide strong cryptographic guarantees, but are often computationally and communicationally expensive. To improve efficiency, trusted hardware was employed to perform PIR for LBS queries [21]. This technique is built on hardware-aided PIR [29], which assume that a trusted third party (TTP) initializes the system by setting the secret key and the permutation of the database. Like LBS queries based on access control, mix zone and k-anonymity, this technique is vulnerable to misbehavior of the third party.

It is a challenge to give practical solutions for kNN queries with location privacy on the basis of PIR.

In this paper, we extend our work [33] presented in ICDE 2014. We construct solutions for kNN queries on the basis of PIR with the Paillier public-key cryptosystem. We have four main contributions as follows:

- Current PIR-based LBS queries [8], [9], [22], [23] usually require two stages. In the first stage, the mobile user retrieves the index of his location from the LBS provider. In the second stage, the mobile user retrieves the POIs according to the index from the LBS provider. The mobile user and the LBS provider need to run two PIR protocols succeedingly. To simplify the process, we give a solution for kNN queries which needs one PIR only, i.e., the mobile user sends his location (encrypted) to the LBS provider and receives the k nearest POIs (encrypted) from the LBS provider.
- Current PIR-based LBS queries only allow the mobile user to find out k nearest POIs regardless of the type of POIs. For the first time, we take into account the type of POIs in kNN queries. We give a solution for the mobile user to preserve query privacy, i.e., finding out k nearest PIOs of the same type without revealing to LBS provider what type of POIs he is interested in. For example, our solution allows the mobile user to find out k nearest car parks from the LBS provider without revealing to LBS provider that the type of POIs is car park.
- Current PIR-based LBS queries [8], [9], [22], [23], [33] allow the mobile user to retrieve only one POI after a protocol execution. For the first time, we take into account sequential queries. We give a solution for the mobile user to query a sequence of POIs without need of multiple executions of the whole protocol. This greatly improves the efficiency of sequential queries.
- Current PIR-based LBS solutions [33] allow LBS queries according to location and single POI type attribute only. They do not support LBS queries with multiple POI type attributes, e.g., car park and daily parking fee (which can be categorized into discrete data values, such as "Low" ($<\$10$), "Middle" ($\$10$-$\$30$) and "High" ($>\$30$)). For the first time, we give a generic solution which can be applied to multiple discrete type attributes of private queries.

To analyze the security of our solutions, we define a security model for private kNN queries. The security analysis has shown that our solutions ensures both location privacy in the sense that the user does not reveal any information about his location to the LBS provider and query privacy in the sense that the user does not reveal what type of POIs he is interested in to the LBS provider. In addition, our solutions have data privacy in the sense that the LBS provider releases to the user only k nearest POIs per query.

We have implemented our solutions on an example of location-based database and experiments have shown that our solutions are practical.

The main differences between our previous work [33] and our current paper are: (1) The previous work fixed the number of nearest neighbors k. The current work allows any number of nearest neighbors k up to K, where K is a constant; (2) The previous work defined location privacy which implied query privacy. The current work defines location and query privacy separately; (3) The previous work used the Rabin cryptosystem [24] to prevent the mobile user to retrieve more than one data per query and did not allow sequential queries without multiple executions of the whole protocol. The current work uses RSA [25] to achieve the data privacy and support sequential queries; (4) The current work adds a generic solution for multiple discrete type attributes of private location-based queries; In addition, (5) we have added some experiments for variable k.

The rest of the paper is arranged as follows. Related works are surveyed in Section 2. Backgrounds are introduced in Section 3. We define our model in Section 4 and describe our solutions in Sections 5 and 6. The security and performance analysis is carried out in Sections 7 and 8. Experiment results are shown in Section 9. Conclusions are drawn in the last section.

2 Related Works

Current main techniques to preserve location privacy for LBS are as follows.

Information access control [16], [35]: User locations are sent to the LBS provider as usual. This technique relies on the LBS provider to restrict access to stored location data through rule-based polices. It supports three types of location-based queries: 1) user location queries (querying the location of a specific user or users, identified by their unique

identifiers); 2) enumeration queries (querying lists of users at specific locations, expressed either in terms of geographic or symbolic attributes); 3) asynchronous queries (querying "event" information, such as when users enter or leave specific areas). This technique requires the LBS provider to maintain all user locations. It is vulnerable to misbehavior of the LBS provider.

Mix zone [3]: A trusted middleware relays between the mobile users and the LBS provider. Before forwarding the location-based queries of the users to the LBS, the middleware anonymizes their locations by pseudonyms. The basic idea is: when a user enters a mix zone, the middleware assigns him a pseudonym, by which the user queries LBS. The communication between the user and the LBS is through the middleware and the pseudonym changes whenever the user enters the mix zone. Recently, the mix-zone has been applied to road networks [20]. This technique requires the middleware to anonymize user locations. It is vulnerable to misbehavior of the middleware.

k-anonymity [28]: This technique ensures that a record could not be distinguished from k-1 other records. Instead of sending a single user's exact location to the LBS, k-anonymity based schemes collect k user locations and send a corresponding (minimum) bounding region to the LBS as the query parameter. The collection of different mobile user locations is done either by a trusted third-party [2], [15] between the users and the LBS, or via a peer-to-peer collaboration [4] among users. Because k-anonymity is achieved, an adversary can only identify a location's user with probability no higher than $1/k$. This technique relies on the third party or a peer user to collect different mobile user locations. It is vulnerable to misbehavior of the third party or the peer user.

"Dummy" locations [13], [27]: The basic idea is when the mobile user queries the LBS, he sends many random other locations along with his location to the LBS provider to confuse his location such that the server cannot distinguish the actual location from the fake locations. Different from k-anonymity based schemes, this approach include fake or fixed locations, rather than those of other mobile users, as parameters of queries sent to the LBS provider. Fake dummy locations are generated at random, and fixed locations are chosen from special ones such as road intersections. Either way, the exact user locations are hidden from the service provider. Although this technique does not rely on any third party, the LBS provider can restrict the user in a small sub space of the total domain, leading to weak privacy.

Private Information Retrieval [18]: This technique allows a user to retrieve a record from a database server without revealing which record he is retrieving. PIR-based protocols [8], [9], [22], [23] are proposed for POI queries and composed of two stages. In the first stage, the user privately determines the index of his location through the service provider without disclosing his coordinates to it. In the second stage, the user runs a PIR protocol with the service provider to retrieve the POIs corresponding to the index. The difference between Ghinita et al. [8], [9] and Paulet et al. [22], [23] PIR-based protocols is in the first stage, where Ghinita et al. approach is based on homomorphic encryption [19] while the technique of Paulet et al. is based on oblivious transfer [17]. In addition, trusted hardware was employed to perform PIR for LBS queries [21]. Their technique is built on hardware-aided PIR [29], which relies on a trusted third party to set the secret key and the permutation of the database. Like LBS queries based on access control, mix zone and k-anonymity, this technique is vulnerable to misbehavior of the third party.

Geographic data transformation [11], [31], [32]: This technique involves three parties: 1) A data owner who has a database D of points, and would like to outsource D to a server (i.e., cloud service provider) that cannot be fully trusted. 2) A user who wants to access and pose queries to the database D. 3) A server that is honest but potentially curious in the tuples in D and/or the queries from the users. A server could be curious either because he is just curious or he has been compromised to become curious on the behalf of a third party without his explicit knowledge. In this setting the data owner is different from the LBS. The owner transforms the database (using some encoding methodology) prior to transmitting it to the LBS. An authorized user that possesses the secret transformation keys issues an encoded query to the LBS. Both the database and the queries are unreadable by the LBS and, thus, location privacy is protected. The goal is to provide the LBS with searching capabilities over the encoded data. Wong et al. [31] propose a secure point transformation, which preserves the relative distances of all the database POIs to any query point. In another solution [32], given only the encryption of location point $E(q)$ and the encryption of database $E(D)$, the server can return a relevant (encrypted) partition $E(G)$ from $E(D)$, such that that $E(G)$ is guaranteed to contain the answer for the NN query. These techniques allow approximate NN search directly on the transformed points. They are prone to access pattern attacks [30] because the same query always returns the same encoded results.

Two LBS Servers: To overcome the access pattern attacks, Elmehdwi et al. [6] gave a solution for kNN query based on the semantically secure Paillier encryption [19], assuming two LBS servers exist, one having the encrypted data and another having the decryption key. Similarly, Schlegel et al. [26] proposed a solution for continuous location-based services, assuming a query server and a service provider exist, where a query server holds the encrypted location while the service provider has the decryption key. These solutions have to assume that two LBS servers never collude.

Recently, Ghinita and Rughinis [10] proposed an interesting location-based alert system, where a mobile user keeps sending the encryption of his location to a LBS server and only when he enters a disaster area, the server is able to know his location and send an alert to him.

3 Backgrounds

3.1 Paillier Public-Key Cryptosystem

Paillier public-key cryptosystem [19] is composed of three algorithms as follows.

- *Key Generation*: A user randomly chooses two large distinct primes p, q and an element g of $\mathbb{Z}^*_{N^2}$ whose order is a nonzero multiple of $N = pq$, publishes the public key $pk = (N, g)$, and keeps the private key $sk = (p, q)$ secret.
- *Encryption*: Given the public key pk of the user, one can encrypt a message m where m is a positive

integer less than N by randomly choosing r from $\mathbb{Z}^*_{N^2}$ and computing

$$c = \mathsf{E}(m, pk) = g^m r^N \ (mod\ N^2), \tag{1}$$

where c is the ciphertext of m. Since r is randomly chosen, the ciphertext c of a message m is random. Therefore, Paillier cryptosystem is a probabilistic encryption.

- *Decryption*: The user can decrypt the ciphertext c with the private key sk by computing

$$m = \mathsf{D}(c, sk) = \frac{(c^\lambda (mod\ N^2) - 1)/N}{(g^\lambda (mod\ N^2) - 1)/N} (mod\ N), \tag{2}$$

where $\lambda = lcm(p-1, q-1)$.

Homomorphic Properties: Paillier cryptosystem has two homomorphic encryption properties as follows:

$$\mathsf{E}(m_1)\mathsf{E}(m_2) = \mathsf{E}(m_1 + m_2), \tag{3}$$

$$\mathsf{E}(m_1)^a = \mathsf{E}(am_1)\ , \tag{4}$$

for any $m_1, m_2, m, a \in \mathbb{Z}_N$.

Suppose that $\mathsf{E}(m_i) = g^{m_i} r_i^N (mod\ N^2)$ for i=1, 2, it is easy to verify (3) and (4) because

$$\mathsf{E}(m_1)\mathsf{E}(m_2) = g^{m_1+m_2}(r_1 r_2)^N (mod\ N^2) = \mathsf{E}(m_1 + m_2),$$

$$\mathsf{E}(m_1)^a = g^{am_1}(r_1^a)^N (mod\ N^2) = \mathsf{E}(am_1).$$

3.2 RSA

RSA [25] is a public-key cryptosystem, composed of three algorithms as follows.

- *Key Generation*: A user randomly chooses two large distinct primes p, q and computes $N = pq$ and $\varphi(N) = (p-1)(q-1)$. Next, he chooses an integer e such that $1 < e < \varphi(N)$ and $gcd(e, \varphi(N)) = 1$, i.e., e and $\varphi(N)$ are coprime, and determines d such that $e \cdot d = 1\ (mod\ \varphi(N))$ using the extended Euclidean algorithm. Then, he publishes the public key $pk = (e, N)$ and keeps the private key $sk = d$ secret. In addition, p, q, and $\varphi(N)$ must also be kept secret because they can be used to calculate d.
- *Encryption*: Given the public key (e, N) of the user, one can encrypt a message m where m is a positive integer less than N by computing

$$c = \mathsf{E}(m, pk) = m^e\ (mod\ N)\ , \tag{5}$$

where c is the ciphertext of m.

- *Decryption*: The user can decrypt the ciphertext c with the private key d by computing

$$m = \mathsf{D}(c, sk) = c^d\ (mod\ N)\ . \tag{6}$$

RSA is not a probabilistic encryption scheme. To transform RSA to a probabilistic encryption scheme, we need to add some random bits into the message m before encrypting m with RSA. Optimal Asymmetric Encryption Padding (OAEP) [1] is a padding scheme often used together with RSA encryption.

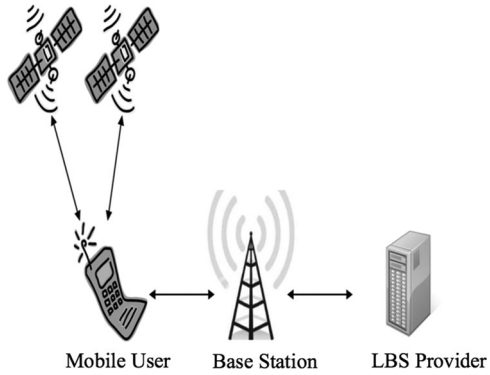

Fig. 1. Location-based service.

4 Our Model

Our model considers a location-based service scenario in mobile environments, as shown in Fig. 1, where there exist the mobile user, the location-based service provider, the base station and satellites, each playing a different role.

- The mobile user sends location-based queries to the LBS provider (or called the LBS server) and receives location-based service from the provider.
- The LBS provider provides location-based services to the mobile user.
- The base station bridges the mobile communications between the mobile user and the LBS provider.
- Satellites provide the location information to the mobile user.

We assume that the mobile user can acquire his location from satellites anonymously, and the base station and the LBS provider do not collude to comprise the user location privacy or there exists an anonymous channel such as Tor[2] for the mobile user to send queries to and receive services from the LBS provider. Our model focuses on user location and query privacy protection against the LBS provider and a kNN query protocol (where $k \leq K$ and K is a constant) is composed of three algorithms as follows.

1) Query Generation (QG): Takes as input a cloaking region CR with $n \times n$ cells and m distinct types of POIs, the location (i, j) of the mobile user, the type t of POIs, and the number of nearest neighbors k, (the mobile user) outputs a query Q (containing CR) and a secret s, denoted as $(Q, s) = \mathsf{QG}(CR, n, m, (i, j), t, k)$.
2) Response Generation (RG): Takes as input the query Q and the location-based database D of POIs, (the LBS provider) outputs a response R, denoted as $R = \mathsf{RG}(Q, D)$.
3) Response Retrieval: Takes as input the response R and the secret s of the mobile user, (the mobile user) outputs k nearest POIs of the type t, denoted as $kNN = \mathsf{RR}(R, s)$.

A private kNN query protocol can be illustrated in Fig. 2 and is correct if $kNN = \mathsf{RR}(R, s)$ outputs k nearest POIs of the type t corresponding the cell at (i, j), where $(Q, s) = \mathsf{QG}(CR, n, m, (i, j), t, k)$ and $R = \mathsf{RG}(Q, D)$.

The security of a private kNN query protocol involves location privacy. Intuitively, the mobile user $\mathcal{U}$ does not

2. https://www.torproject.org/

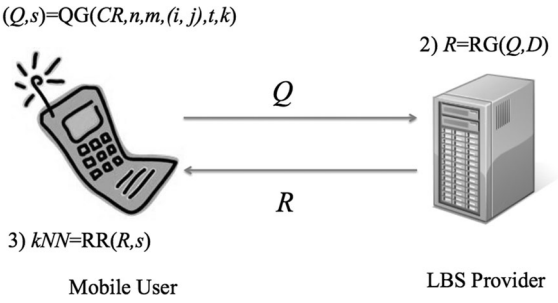

Fig. 2. Private kNN query.

wish to reveal to the LBS provider his location (i, j) to the LBS server which is considered as an adversary.

Now, we formally define location privacy with a game (Game 1) as follows.

Given a cloaking region CR with $n \times n$ cells, m types of POIs and the number of nearest neighbors $k \leq K$ where K is a constant, consider the following game between an adversary (the LBS provider) $\mathcal{A}$, and a challenger $\mathcal{C}$. The game consists of the following steps:

1) For any given type of POIs, t, the adversary $\mathcal{A}$ chooses two distinct tuples (i_0, j_0, t, k) and (i_1, j_1, t, k), where (i_b, j_b) represents the cell, from the cloaking region CR and sends them to the challenger $\mathcal{C}$.
2) The challenger $\mathcal{C}$ chooses a random bit $b \in \{0, 1\}$, and executes the Query Generation (QG) to obtain $(Q_b, s) = \mathsf{QG}(CR, n, m, (i_b, j_b), t, k)$ and then sends Q_b back to the adversary $\mathcal{A}$.
3) The adversary $\mathcal{A}$ can experiment with the code of Q_b in an arbitrary non-black-box way, and finally outputs a bit $b' \in \{0, 1\}$.

The adversary wins the game if $b' = b$ and loses otherwise. We define the adversary $\mathcal{A}$'s advantage in this game to be

$$\mathsf{Adv}_{\mathcal{A}}(\kappa) = |\mathsf{Pr}(b' = b) - 1/2|,$$

where κ is the security parameter.

Definition 1 (Location Privacy Definition). *In a kNN query protocol, the user has location privacy if for any probabilistic polynomial time (PPT) adversary $\mathcal{A}$, we have that $\mathsf{Adv}_{\mathcal{A}}(\kappa)$ is a negligible function, where the probability is taken over coin-tosses of the challenger and the adversary.*

Remark. Location privacy ensures that the server cannot determine the location of the mobile user in the cloaking region CR.

The security of a private kNN query protocol also involves query privacy. Intuitively, the mobile user $\mathcal{U}$ does not wish to reveal to the LBS provider the type t of POIs he is interested in to the LBS server which is considered as an adversary.

Now, we formally define query privacy with a game (Game 2) as follows.

Given a cloaking region CR with $n \times n$ cells, m types of POIs and the number of nearest neighbors $k \leq K$, consider the following game between an adversary (the LBS provider) $\mathcal{A}$, and a challenger $\mathcal{C}$. The game consists of the following steps:

1) For any given location (i, j), the adversary $\mathcal{A}$ chooses two distinct tuples (i, j, t_0, k) and (i, j, t_1, k), where (i, j) represents the cell and t_b stands for the type of POIs, from the cloaking region CR and sends them to the challenger $\mathcal{C}$.
2) The challenger $\mathcal{C}$ chooses a random bit $b \in \{0, 1\}$, and executes the Query Generation (QG) to obtain $(Q_b, s) = \mathsf{QG}(CR, n, m, (i, j), t_b, k)$ and then sends Q_b back to the adversary $\mathcal{A}$.
3) The adversary $\mathcal{A}$ can experiment with the code of Q_b in an arbitrary non-black-box way, and finally outputs a bit $b' \in \{0, 1\}$.

The adversary wins the game if $b' = b$ and loses otherwise. We define the adversary $\mathcal{A}$'s advantage in this game to be

$$\mathsf{Adv}_{\mathcal{A}}(\kappa) = |\mathsf{Pr}(b' = b) - 1/2|,$$

where κ is the security parameter.

Definition 2 (Query Privacy Definition). *In a kNN query protocol, the user has query privacy if for any probabilistic polynomial time adversary $\mathcal{A}$, we have that $\mathsf{Adv}_{\mathcal{A}}(\kappa)$ is a negligible function, where the probability is taken over coin-tosses of the challenger and the adversary.*

Remark. Query privacy ensures that the server cannot determine the type of POIs with the kNN query from the mobile user.

The security of a private kNN query protocol also involves data privacy. Intuitively, the LBS provider S wishes to release only the k nearest POIs of one type to the mobile user $\mathcal{U}$ each time when the user sends a kNN query. In this scenario, the mobile user is considered as an adversary.

Formally, data privacy can be defined with a game (Game 3) as follows.

Given a user location (i, j) where $1 \leq i, j \leq n$, one type t of POIs and the number of nearest neighbors $k \leq K$, consider the following game between an adversary (the user) $\mathcal{A}$, and a challenger $\mathcal{C}$. The game consists of the following steps:

1) The adversary $\mathcal{A}$ chooses any two distinct cloaking regions CR_0 and CR_1 with $n \times n$ cells such that k nearest POIs of the type t in the cell (i, j) are same. The adversary generates a query Q to retrieve the k nearest POIs of the type t in the cell (i, j) and sends Q, CR_0, CR_1 to the challenger $\mathcal{C}$.
2) The challenger $\mathcal{C}$ chooses a random bit $b \in \{0, 1\}$, and runs the response generation algorithm RG to obtain $R_b = \mathsf{RG}(Q, CR_b)$, and then sends R_b back to $\mathcal{A}$.
3) The adversary $\mathcal{A}$ can experiment with the code of R_b in an arbitrary non-black-box way. If the adversary can retrieve the k nearest POIs of the type t in the cell (i, j) from R_b, he outputs his guess $b' \in \{0, 1\}$.

The adversary wins the game if $b' = b$ and loses otherwise. We define the adversary $\mathcal{A}$'s advantage in this game to be

$$\mathsf{Adv}_{\mathcal{A}}(\kappa) = |\mathsf{Pr}(b' = b) - 1/2|,$$

where κ is the security parameter.

Definition 3 (Data Privacy Definition). *In a kNN query protocol, the LBS provider has data privacy if for any probabilistic*

polynomial time adversary $\mathcal{A}$, we have that $\mathsf{Adv}_{\mathcal{A}}(\kappa)$ is a negligible function, where the probability is taken over coin-tosses of the challenger and the adversary.

Remark. Data privacy ensures that the response distributions on the user's view are computationally indistinguishable for any two cloaking regions CR_1 and CR_2 such that the k nearest POI of the type t in the cell (i, j) in the two cloaking regions are the same. This means that a computationally bounded user does not receive information about more than one cell in the cloaking region CR.

Unlike the model defined in [33], we assume that the LBS server encrypts all k nearest POIs with its public key pk. Therefore, in the Response Retrieval (RR) algorithm, after obtaining the encrypted k nearest POIs, the mobile user needs the help of the LBS server with the decryption of the k nearest POIs. The purpose of doing so is to ensure that the mobile user can obtain only one kNN POIs per query. In addition, if the mobile user can obtain a sequence of encrypted k nearest POIs in the response from the LBS server, he can repeatedly run the RR algorithm only with the LBS server to get a sequence of k nearest POIs without need of query generation and response generation. This will greatly improve the efficiency of private queries.

In the decryption process of the RR algorithm, the LBS server must not know the decrypted result. Otherwise, the LBS server can determine the location and query of the mobile user. To hide the decrypted result from the LBS server, the mobile user runs a blind decryption algorithm with the LBS server as follows.

1) Given an encrypted record containing k nearest POIs, denoted as $C = \mathsf{E}(m, pk)$, the user chooses a random number r and computes the blinded ciphertext $C' = \mathsf{F}(C, r)$, where F is the blinding operation. Then he sends C' to the LBS server.
2) Given C', the LBS server decrypts it and replies to the mobile user the blinded plaintext $m' = \mathsf{D}(C', sk)$, where D is the decryption algorithm and sk is the private key of the LBS server.
3) Given m', the mobile user computes the unblinded plaintext $m = \mathsf{G}(m', r)$, where G is the unblinding operation.

The security of the blind decryption algorithm involves blindness. Intuitively, the LBS server provides a decryption service to the mobile user in an encoded form without knowing either the real input or the real output.

Formally, blindness can be defined with a game (Game 4) as follows.

1) The adversary $\mathcal{A}$ (the server) chooses any two distinct plaintexts m_0 and m_1 and encrypts them. Then he sends the ciphertexts $C_0 = \mathsf{E}(m_0, pk)$ and $C_1 = \mathsf{E}(m_1, pk)$ to the challenger $\mathcal{C}$.
2) The challenger $\mathcal{C}$ (the user) chooses a random bit $b \in \{0, 1\}$ and a random number r and computes the blinded ciphertext $C'_b = \mathsf{F}(C_b, r)$. Then he sends C'_b back to the adversary $\mathcal{A}$.
3) The adversary $\mathcal{A}$ can experiment with the code of C'_b in an arbitrary non-black-box way. In the end, he outputs his guess $b' \in \{0, 1\}$.

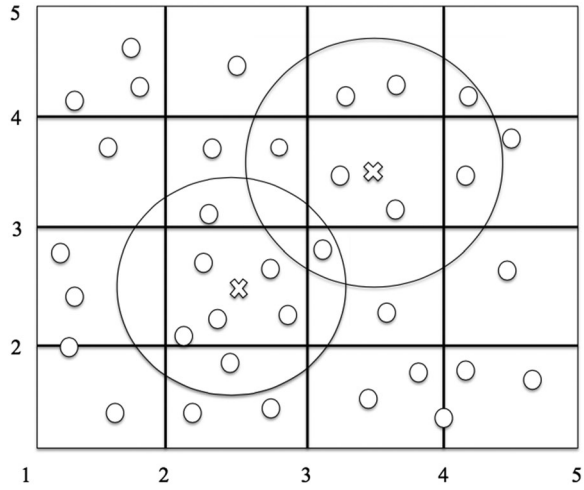

Fig. 3. k nearest POIs for cells.

The adversary wins the game if $b' = b$ and loses otherwise. We define the adversary $\mathcal{A}$'s advantage in this game to be

$$\mathsf{Adv}_{\mathcal{A}}(\kappa) = |\mathsf{Pr}(b' = b) - 1/2|,$$

where κ is the security parameter.

Definition 4 (Blindness Definition). *The Response Retrieval algorithm has blindness if for any probabilistic polynomial time adversary $\mathcal{A}$, we have that $\mathsf{Adv}_{\mathcal{A}}(\kappa)$ is a negligible function, where the probability is taken over coin-tosses of the challenger and the adversary.*

5 Basic Private k Nearest Neighbor Queries

Based on our model, we give a basic construction of private kNN query protocol in this section. Our basic solution is built on the Paillier scheme [19] and RSA [25].

5.1 Initialization

Before execution of any private kNN protocol, an initialization occurs in the LBS server.

First of all, the LBS server divides the location-based database D (a geographic map) into cells with the same size, for example, 1 km width and 1 km length, denoted as grid $= 1$ km. Based on the center of each cell, given a type of POIs, the LBS server collects K nearest POIs of the type, $P_1, P_2, \ldots, P_K$, as shown in Fig. 3, where $K = 8$ and each point is represented by a tuple (x, y), where x and y are the latitude and longitude of the point, respectively.

We assume that POI types are coded into $1, 2, \ldots, m$ which is published to the public. Examples of POI types includes: Churches, Schools, Post offices / postboxes, Telephone boxes, Restaurants, Pubs, Car parks, Speed cameras, Tourist attractions and etc.

For each cell (i, j) and each POI type t, the LBS server keeps K (e.g, $K = 20$) nearest POIs of type t, represented by a stream of bits, denoted as an integer $d^K_{i,j,t}$, where the points are ordered according to the distance to the center of the cell such that the first k (where $k \leq K$) points, denoted as $d^k_{i,j,t}$, are the k nearest POIs. Each cell contains m integers for different types of POIs. We assume $M(k) = max(d^k_{i,j,t})$, i.e., the longest record.

Remark. The LBS server can build different POI databases for different grids. This does not affect privacy for the mobile user and the LBS server. The smaller the grid is, the more accurate the result of kNN query is, but the less efficient the response generation for given cloaking region is. A mobile user can choose a grid accordingly when he queries the LBS.

Because the LBS provider collects K nearest POIs according to the center of each cell (i.e., the cross points shown in Fig. 3), it responses the same k (where $k \leq K$) nearest POIs to the two mobile users within the same cell no matter where the two mobile users are in the cell. For the mobile user locating near the border of two cells, he may query two cells around his location and then find out k nearest POIs among the query responses. The purpose of our method is to avoid privately comparing distances, which is hard to do without revealing the location of the user.

Next, the LBS server generates the RSA public and private key pair (pk, sk), where $pk = \{e, N\}$ and $sk = \{d\}$. Depending on k specified by the user, LBS server will encrypt $d^k_{i,j,t}$ to ${d^k_{i,j,t}}'$ for all i, j, t according to RSA encryption algorithm (described in Section 3.2) and Optimal Asymmetric Encryption Padding [1] as

$$ {d^k_{i,j,t}}' = \mathsf{E}\big(d^k_{i,j,t}, pk\big) = \big(d^k_{i,j,t}\big)^e (mod\ N)\ . \tag{7} $$

Remark. The public key $pk = \{e, N\}$ of the LBS server is published and known to all mobile users. It is required that $\log_2 N > 2\log_2 M(k)$ for kNN query. For different k, the LBS server can publish different public key. Assume the location of a POI can be represented by 32 bits, $d^k_{i,j,t}$ is less than 1,024 bits when $k = 5, 10, 20, 30$, and less than 2,048 bits when $k = 40, 50$ unusually. The location of a POI can be represented by less bits if we introduce latitude and longitude relative to a reference point. For simplicity, we also use $d^k_{i,j,t}$ to denote the k nearest POIs after OAEP.

5.2 Basic Private kNN Query Protocol

We assume that POI types are coded into $1, 2, \ldots, m$ which is published to the public and the mobile user $\mathcal{U}$ wishes to find k nearest POIs of type t around his location. The user $\mathcal{U}$ chooses a cloaking region CR with $n \times n$ cells, where $\mathcal{U}$ is located in the cell (i, j), and runs the kNN query protocol with the LBS provider $\mathcal{S}$, composed of Algorithms 1-3.

Remark. The CR may be specified by the coordinates (x, y) of an origin point and the order n of a square grid. The cell which contains the origin point is labelled as (1, 1). The CR covers the square grid from the cell (1, 1) to the cell (n, n).

Remark. By slightly modifications of Algorithms 1-3, our protocol can protect location or query privacy only.

In case that a user wishes to keep location privacy only (i.e., keep (i, j) private in the query), given the type t of POIs, he runs Steps 1, 3, 5 in Algorithm 1 and submits $Q = \{CR, n, m, k, c_1, c_2, \ldots, c_n, t, pk_1\}$ to the LBS server. Because t is given, the LBS server runs Steps 1 and 3 in Algorithm 2, where $C_{\alpha,\beta} = {d^k_{\alpha,\beta,t}}'$ for $\beta = 1, 2, \ldots, n$ and returns $R = \{C_1, C_2, \ldots, C_n\}$ to the user. In response retrieval, the user computes $w = r^e \mathsf{D}_1(C_j, sk_1)(mod\ N)$ where r is a random integer and runs Steps 2 and 3 to retrieve kNN of the type t.

Algorithm 1. Query Generation (User)

Input: $CR, n, m, (i, j), t, k, pk = \{e, N\}$
Output: Q, s
1: Randomly choose two large primes p_1, q_1 such that $N_1 = p_1 q_1 > N$.
2: Randomly choose two large primes p_2, q_2 such that $N_2 = p_2 q_2 > N$, where $N_2^2 < N_1$.
3: Let $sk_1 = \{p_1, q_1\}, pk_1 = \{g_1, N_1\}$.
4: Let $sk_2 = \{p_2, q_2\}, pk_2 = \{g_2, N_2\}$.
5: For each $\ell \in \{1, 2, \ldots, n\}$, pick a random integer $r_\ell \in \mathbb{Z}^*_{N_1^2}$, compute

$$ c_\ell = \begin{cases} \mathsf{E}_1(1, pk_1) = {g_1}^1 {r_\ell}^{N_1} (mod\ {N_1}^2) & \text{if } \ell = i \\ \mathsf{E}_1(0, pk_1) = {g_1}^0 {r_\ell}^{N_1} (mod\ {N_1}^2) & \text{otherwise} \end{cases} $$

where E_1 denotes the Paillier encryption algorithm with public key $pk_1 = \{g_1, N_1\}$ as described in Section 3.1.
6: For each $\ell \in \{1, 2, \ldots, m\}$, pick a random integer $r'_\ell \in \mathbb{Z}^*_{N_2^2}$, compute

$$ c'_\ell = \begin{cases} \mathsf{E}_2(1, pk_2) = {g_2}^1 {r'_\ell}^{N_2} (mod\ {N_2}^2) & \text{if } \ell = t \\ \mathsf{E}_2(0, pk_2) = {g_2}^0 {r'_\ell}^{N_2} (mod\ {N_2}^2) & \text{otherwise} \end{cases} $$

where E_2 denotes the Paillier encryption algorithm with public key $pk_2 = \{g_2, N_2\}$.
7: Let $Q = \{CR, n, m, k, c_1, c_2, \ldots, c_n, c'_1, c'_2, \ldots, c'_m, pk_1, pk_2\}$, $s = \{sk_1, sk_2\}$.
8: **return** Q, s

Algorithm 2. Response Generation RG (Server)

Input: $D, Q = \{CR, n, m, k, c_1, c_2, \ldots, c_n, c'_1, c'_2, \ldots, c'_m, pk_1, pk_2)$
Output: $R = \{C_1, C_2, \ldots, C_n\}$
1: Based on CR, n, k, for any t in the cell (i, j), take the first k points of $d^K_{i,j,t}$, denoted as $d^k_{i,j,t}$, and encrypt it according to RSA encryption algorithm described in Section 3.2. The result is denoted as ${d^k_{i,j,t}}'$.
2: Based on CR, n, m, for each cell (α, β) in CR, compute

$$ C_{\alpha,\beta} = \prod_{\ell=1}^{m} {c'_\ell}^{{d^k_{\alpha,\beta,\ell}}'} (mod\ {N_2}^2) $$

3: Based on CR and n, compute $R = \{C_1, C_2, \ldots, C_n\}$, where for $\beta \in \{1, 2, \ldots, n\}$,

$$ C_\beta = \prod_{\alpha=1}^{n} {c_\alpha}^{C_{\alpha,\beta}} (mod\ {N_1}^2). $$

4: **return** R

In case that a user wishes to keep query privacy only (i.e., keep the type t private in the query), given the location (i, j) of the user, he runs Steps 2, 4, 6 in Algorithm 1 and submits $Q = \{CR, n, m, k, (i, j), c'_1, c'_2, \ldots, c'_m, pk_2\}$ to the LBS server. Because (i, j) is given, the LBS server runs Steps 1 and 2 in Algorithm 2, where $\alpha = i, \beta = j$ and returns $R = \{C_{i,j}\}$ to the user. In response retrieval, the user computes $w = r^e \mathsf{D}_2$

$(C_{i,j}, sk_2)(mod\ N)$ where r is a random integer and runs Steps 2 and 3 to retrieve kNN of the type t.

Algorithm 3. Response Retrieval RR (User)

Input: $R = \{C_1, C_2, \ldots, C_n\}, s = \{sk_1, sk_2\}, sk = \{d\}$
Output: z

1: The user randomly chooses an integer $r < N$ and computes and sends to the server

$$w = r^e \mathsf{D}_2(\mathsf{D}_1(C_j, sk_1), sk_2)(mod\ N)$$

where $\mathsf{D}_1, \mathsf{D}_2$ are the Paillier decryption algorithm as described in Section 3.1.

2: The server computes and replies to the user

$$v = \mathsf{D}(w, sk) = w^d (mod\ N)$$

where D denotes the RSA decryption algorithm as described in Section 3.2.

3: The user computes

$$z = r^{-1} v (mod\ N)$$

4: **return** z

Remark. In Algorithm 3, when the mobile user receives the response, he can ignore C_ℓ $(\ell \neq j)$ and receive C_j only because only C_j contains the information about the k nearest POIs in the cell (i, j). In fact, the mobile user is able to retrieval the k nearest POIs of type t in any cell (i, γ) $(\gamma = 1, 2, \ldots, n)$ by running Algorithm 3 without need of query generation and response generation. This feature makes private queries very efficient when the mobile user moves from the cell (i, j) to another cell such as the cell $(i, j+1)$ (where $j+1 \leq n$) or $(i, j-1)$ (where $j-1 \geq 1$). In addition, the LBS server can easily control the data release by decryption because one response retrieval invocation releases one data only.

Theorem 1 (Correctness). *Our basic kNN query protocol (Algorithms 1-3) is correct. In other words, for any cloaking region CR with $n \times n$ and m types of POIs, and the index i, j of any cell $(1 \leq i, j \leq n)$, any type t of POIs, and any number of nearest neighbor $k \leq K$, we have*

$$d^k_{i,j,t} = \mathsf{RR}(R, s),$$

holds, where $d^k_{i,j,t}$ stands for k nearest POIs of the type t referring to the center of the cell (i, j), and $(Q, sk) = \mathsf{QG}(CR, n, m, (i, j), t, k)$, $R = \mathsf{RG}(Q, D)$.

Proof. Based on Algorithms 1-2, we have

$$C_j = \prod_{\alpha=1}^{n} c_\alpha^{C_{\alpha,j}} = g_1^{C_{i,j}} \left(\prod_{\alpha=1}^{n} r_\alpha^{C_{\alpha,j}} \right)^{N_1} (mod\ {N_1}^2),$$

which is a Paillier encryption of $C_{i,j}$. Therefore, $\mathsf{D}_1(C_j, sk_1) = C_{i,j}$.

Based on Algorithms 1-2, we also have

$$C_{i,,j} = \prod_{\ell=1}^{m} {c'_\ell}^{d^k_{i,j,\ell}{}'} = {g_2}^{d^k_{i,j,t}{}'} \left(\prod_{\ell=1}^{m} {r'_\ell}^{d^k_{i,j,\ell}{}'} \right)^{N_2} (mod\ {N_2}^2),$$

which is a Paillier encryption of $d^k_{i,j,t}{}'$. Therefore, $\mathsf{D}_2(\mathsf{D}_1(C_j, sk_1), sk_2) = d^k_{i,j,t}{}'$.

Based on Algorithm 3, we have

$$\begin{aligned} z &= r^{-1} v = r^{-1} w^d \\ &= r^{-1} (r^e d^k_{i,j,t}{}')^d = r^{-1} r^{ed} {d^k_{i,j,t}{}'}^d \\ &= r^{-1} r d^k_{i,j,t} = d^k_{i,j,t} (mod\ N). \end{aligned}$$

Therefore, we have $d^k_{i,j,t} = \mathsf{RR}(R, s)$ and the theorem is proved. □

6 Genetic Private k Nearest Neighbor Queries

In this section, we extend our basic solution to a generic construction of private kNN query protocol. Our generic solution considers a multi-dimension space where each POI is defined with location attributes (i, j) (where $1 \leq i, j \leq n$) and multiple discrete type attributes $(t_1, t_2, \ldots, t_T)$ (where t_λ $(1 \leq \lambda \leq T)$ is an integer and $1 \leq t_\lambda \leq m_\lambda$). For example, a car park (encoded as 3) located at $(9, 4)$ in the cell $(8, 5)$ with "Mid" daily parking fee (encoded as 1) may be represented as $d^1_{8,5,3,1} = \text{POI}\ (9, 4)$, where daily parking fee can be categorized into "Low" ($<$ \$10), "Mid" (\$10-\$30) and "High" ($>$ \$30)).

6.1 Initialization

Like our basic solution, first of all, the LBS server divides the database D (a geographic map) into cells with the same size. Based on the center of each cell (i, j), given type attributes $(t_1, t_2, \ldots, t_T)$, the LBS server collects K nearest POIs of the type, $P_1, P_2, \ldots, P_K$, each point is represented by a tuple (x, y).

For each cell (i, j) and each POI type attributes $(t_1, t_2, \ldots, t_T)$, the LBS server keeps K (e.g, $K = 20$) nearest POIs, represented by a stream of bits, denoted as an integer $d^K_{i,j,t_1,t_2,\ldots,t_T}$, where the points are ordered according to the distance to the center of the cell such that the first k (where $k \leq K$) points, denoted as $d^k_{i,j,t_1,t_2,\ldots,t_T}$, are the k nearest POIs. Each cell contains $\prod_{\lambda=1}^{T} m_\lambda$ integers for different type attributes of POIs. We assume $M(k) = max(d^k_{i,j,t_1,t_2,\ldots,t_T})$, i.e., the longest record.

Next, the LBS server generates the RSA public and private key pair (pk, sk), where $pk = \{e, N\}$ and $sk = \{d\}$. Depending on k specified by the user, LBS server will encrypt $d^k_{i,j,t_1,t_2,\ldots,t_T}$ to ${d^k_{i,j,t_1,t_2,\ldots,t_T}}'$ for all $i, j, t_1, t_2, \ldots, t_T$ according to RSA encryption algorithm (described in Section 3.2) and Optimal Asymmetric Encryption Padding [1] as

$$\begin{aligned} {d^k_{i,j,t_1,t_2,\ldots,t_T}}' &= \mathsf{E}\big(d^k_{i,j,t_1,t_2,\ldots,t_T}, pk\big) \\ &= \big(d^k_{i,j,t_1,t_2,\ldots,t_T}\big)^e (mod\ N)\ . \end{aligned} \tag{8}$$

6.2 Generic Private kNN Query Protocol

We assume that the mobile user $\mathcal{U}$ wishes to find k nearest POIs of type attributes $t_1, t_2, \ldots, t_T$ around his location. The user $\mathcal{U}$ chooses a cloaking region CR with $n \times n$ cells, where $\mathcal{U}$ is located in the cell (i, j), and runs the kNN query protocol with the LBS provider $\mathcal{S}$, composed of Algorithms 4-6.

Algorithm 4. Query Generation (User)

Input: $CR, n, m_1, m_2, \ldots, m_T, k, (i,j), (t_1, t_2, \ldots, t_T), pk = \{e, N\}$
Output: Q, s

1: Randomly choose two large distinct primes p_1, q_1 such that $N_1 = p_1 q_1 > N$.
2: For each $\lambda \in \{1, 2, \ldots, T\}$, randomly choose two large distinct primes $p_{2\lambda}, q_{2\lambda}$ such that $N_{2\lambda} = p_{2\lambda} q_{2\lambda} > N$, where ${N_{2\lambda}}^2 < N_{2(\lambda-1)}$ and $N_{20} = N_1$.
3: Let $sk_1 = \{p_1, q_1\}, pk_1 = \{g_1, N_1\}$.
4: For each $\lambda \in \{1, 2, \ldots, T\}$, let $sk_{2\lambda} = \{p_{2\lambda}, q_{2\lambda}\}, pk_{2\lambda} = \{g_{2\lambda}, N_{2\lambda}\}$.
5: For each $\ell \in \{1, 2, \ldots, n\}$, pick a random integer $r_\ell \in \mathbb{Z}^*_{{N_1}^2}$, compute

$$c_\ell = \begin{cases} \mathsf{E}_1(1) = {g_1}^1 {r_\ell}^{N_1} (mod\ {N_1}^2) & \text{if } \ell = i \\ \mathsf{E}_1(0) = {g_1}^0 {r_\ell}^{N_1} (mod\ {N_1}^2) & \text{otherwise} \end{cases}$$

where E_1 denotes the Paillier encryption algorithm with public key $pk_1 = \{g_1, N_1\}$ as described in Section 3.1.
6: For each $\lambda \in \{1, 2, \ldots, T\}$ and each $\ell \in \{1, 2, \cdots, m_T\}$, pick a random integer $r'_{\lambda\ell} \in \mathbb{Z}^*_{{N_{2\lambda}}^2}$, compute

$$c'_{\lambda\ell} = \begin{cases} \mathsf{E}_{2\lambda}(1) = {g_{2\lambda}}^1 {r'_{\lambda\ell}}^{N_{2\lambda}} (mod\ {N_{2\lambda}}^2) & \text{if } \ell = t_\lambda \\ \mathsf{E}_{2\lambda}(0) = {g_{2\lambda}}^0 {r'_{\lambda\ell}}^{N_{2\lambda}} (mod\ {N_{2\lambda}}^2) & \text{otherwise} \end{cases}$$

where $\mathsf{E}_{2\lambda}$ denotes the Paillier encryption algorithm with public key $pk_{2\lambda} = \{g_{2\lambda}, N_{2\lambda}\}$ as described in Section 3.1.
7: Let $Q = \{CR, n, m_1, m_2, \ldots, m_T, k, c_1, c_2, \ldots, c_n, c'_{11}, c'_{12}, \ldots, c'_{1m_1}, c'_{21}, c'_{22}, \ldots, c'_{2m_2}, \ldots, c'_{T1}, c'_{T2}, \ldots, c'_{Tm_T}, pk_1, pk_{21}, pk_{22}, \ldots, pk_{2T}\}$, $s = \{sk_1, sk_{21}, sk_{22}, \ldots, sk_{2T}\}$.
8: **return** Q, s

Algorithm 5. Response Generation RG (Server)

Input: $D, Q = \{CR, n, m_1, m_2, \ldots, m_T, k, c_1, c_2, \ldots, c_n, c'_{11}, c'_{12}, \ldots, c'_{1m_1}, c'_{21}, c'_{22}, \ldots, c'_{2m_2}, \ldots, c'_{T1}, c'_{T2}, \ldots, c'_{Tm_T}, pk_1, pk_{21}, pk_{22}, \ldots, pk_{2T}\}$
Output: $R = \{C_1, C_2, \ldots, C_n\}$

1: Based on CR, $n, m_1, m_2, \ldots, m_T, k$, for each cell (α, β) $(1 \le \alpha, \beta \le n)$ in CR {
2: Let $C_{\alpha,\beta,\ell_1,\ldots,\ell_T} = {d^k_{\alpha,\beta,\ell_1,\ldots,\ell_{T-1},\ell_T}}'$ for all possible $\ell_1, \ell_2, \ldots, \ell_T$.
3: For $\lambda = 1$ to $T-1$ {
4: For all possible $\ell_1, \ell_2, \ldots, \ell_{T-\lambda}$, compute

$$C_{\alpha,\beta,\ell_1,\ldots,\ell_{T-\lambda}} = \prod_{\ell=1}^{m_{T-\lambda+1}} {c'_{(T-\lambda+1)\ell}}^{C_{\alpha,\beta,\ell_1,\ldots,\ell_{T-\lambda},\ell}} (mod\ {N_{2(T-\lambda+1)}}^2)\ \}$$

5: Compute

$$C_{\alpha,\beta} = \prod_{\ell=1}^{m_1} {c'_{1\ell}}^{C_{\alpha,\beta,\ell}} (mod\ {N_{21}}^2)\ \}$$

6: Based on CR and n, compute $R = \{C_1, C_2, \ldots, C_n\}$, where for $\beta \in \{1, 2, \ldots, n\}$,

$$C_\beta = \prod_{\alpha=1}^{n} {c_\alpha}^{C_{\alpha,\beta}} (mod\ {N_1}^2).$$

7: **return** R

Algorithm 6. Response Retrieval RR (User)

Input: $R = \{C_1, C_2, \ldots, C_n\}, s = \{sk_1, sk_{21}, sk_{22}, \ldots, sk_{2T}\}, sk = \{d\}$
Output: z

1: The user randomly chooses an integer $r < N$ and computes and sends to the server

$$w = r^e \mathsf{D}_{2T}(\cdots(\mathsf{D}_{21}(\mathsf{D}_1(C_j, sk_1), sk_{21})\cdots), sk_{2T}) (mod\ N)$$

where $\mathsf{D}_1, \mathsf{D}_{21}, \ldots, \mathsf{D}_{2T}$ are the Paillier decryption algorithms as described in Section 3.1.
2: The server computes and replies to the user

$$v = \mathsf{D}(w, sk) = w^d (mod\ N)$$

where D denotes the RSA decryption algorithm as described in Section 3.2.
3: The user computes

$$z = r^{-1} v (mod\ N)$$

4: **return** z

Theorem 2 (Correctness). *Our generic kNN query protocol (Algorithms 4-6) is correct. In other words, for any cloaking region CR with $n \times n$, and the index i, j of any cell ($1 \le i, j \le n$), any type attributes $(t_1, t_2, \ldots, t_T)$ of POIs, and any number of nearest neighbor $k \le K$, we have*

$$d^k_{i,j,t_1,t_2,\ldots,t_T} = \mathsf{RR}(R, s),$$

holds, where $d^k_{i,j,t_1,t_2,\ldots,t_T}$ stands for k nearest POIs of the type attributes $(t_1, t_2, \ldots, t_T)$ referring to the center of the cell (i, j), and $(Q, s) = \mathsf{QG}(CR, n, m_1, m_2, \ldots, m_T, (i,j), (t_1, t_2, \ldots, t_T), k), R = \mathsf{RG}(Q, D)$.

Proof. Based on Algorithms 4-5, we have

$$\begin{aligned} C_{\alpha,\beta,\ell_1,\ldots,\ell_{T-1}} &= \prod_{\ell=1}^{m_T} {c'_{T\ell}}^{{d^k_{\alpha,\beta,\ell_1,\ldots,\ell_{T-1},\ell}}'} \\ &= {c'_{Tt_T}}^{{d^k_{\alpha,\beta,\ell_1,\ldots,\ell_{T-1},t_T}}'} (mod\ {N_{2T}}^2) \\ &= \mathsf{E}_{2T}({d^k_{\alpha,\beta,\ell_1,\ldots,\ell_{T-1},t_T}}'), \end{aligned}$$

for all possible $\ell_1, \ell_2, \ldots, \ell_{T-1}$.

$$\begin{aligned} C_{\alpha,\beta,\ell_1,\ldots,\ell_{T-2}} &= \prod_{\ell=1}^{m_{T-1}} {c'_{(T-1)\ell}}^{C_{\alpha,\beta,\ell_1,\ldots,\ell_{T-2},\ell}} \\ &= {c'_{(T-1)t_{T-1}}}^{C_{\alpha,\beta,\ell_1,\ldots,t_{T-1}}} (mod\ {N_{2(T-1)}}^2) \\ &= \mathsf{E}_{2(T-1)}(\mathsf{E}_{2T}({d^k_{\alpha,\beta,\ell_1,\ldots,\ell_{T-2},t_{T-1},t_T}}')), \end{aligned}$$

for all possible $\ell_1, \ell_2, \ldots, \ell_{T-2}$.

. . . . . .

$$\begin{aligned} C_{\alpha,\beta,\ell_1} &= \prod_{\ell=1}^{m_2} {c'_{2\ell}}^{C_{\alpha,\beta,\ell_1,\ell}} \\ &= {c'_{2t_2}}^{C_{\alpha,\beta,\ell_1,t_2}} (mod\ {N_{22}}^2) \\ &= \mathsf{E}_{22}(\mathsf{E}_{23}(\cdots(\mathsf{E}_{2T}({d^k_{\alpha,\beta,\ell_1,t_2,\ldots,t_T}}')))), \end{aligned}$$

for all possible ℓ_1.

$$\begin{aligned} C_{\alpha,\beta} &= \prod_{\ell=1}^{m_1} {c'_{1\ell}}^{C_{\alpha,\beta,\ell}} \\ &= {c'_{1t_1}}^{C_{\alpha,\beta,t_1}} (mod\ {N_{21}}^2) \\ &= \mathsf{E}_{21}(\mathsf{E}_{22}(\cdots(\mathsf{E}_{2T}({d^k_{\alpha,\beta,t_1,t_2,\ldots,t_T}}')))). \end{aligned}$$

In addition, we have

$$C_j = \prod_{\alpha=1}^{n} c_\alpha^{C_{\alpha,j}} = g_1^{C_{i,j}} \left(\prod_{\alpha=1}^{n} r_\alpha^{C_{\alpha,j}} \right)^{N_1} (mod\ N_1^2),$$

which is a Paillier encryption of $C_{i,j}$. Therefore, $\mathsf{D}_1(C_j, sk_1) = C_{i,j} = \mathsf{E}_{21}(\mathsf{E}_{22}(\cdots(\mathsf{E}_{2T}({d^k_{i,j,t_1,t_2,\ldots,t_T}}'))))$.

Based on Algorithm 6, we have

$$w = r^e {d^k_{\alpha,\beta,t_1,t_2,\ldots,t_T}}'.$$

Therefore, we have

$$\begin{aligned} z &= r^{-1}v = r^{-1}w^d \\ &= r^{-1}(r^e {d^k_{i,j,t_1,t_2,\ldots,t_T}}')^d = r^{-1}r^{ed}{d^k_{i,j,t_1,t_2,\ldots,t_T}}'^d \\ &= r^{-1}r d^k_{i,j,t_1,t_2,\ldots,t_T} = d^k_{i,j,t_1,t_2,\ldots,t_T} (mod\ N). \end{aligned}$$

Therefore, we have $d^k_{i,j,t_1,t_2,\ldots,t_T} = \mathsf{RR}(R,s)$ and the theorem is proved. □

Remark. Like our basic solution, our generic solution can protect location or query privacy only by slightly modifications of Algorithms 4-6.

In particular, our generic solution can be modified to keep query privacy for partial type attributes. For example, a user wishes to find k nearest POIs around his location (i,j) with the type attribute $(t_1, t_2, \ldots, t_S, t_{S+1}, \ldots, t_T)$, where the location (i,j) and partial type attributes $(t_1, t_2, \ldots, t_S)$ should be kept private to the LBS server. He runs Algorithm 1 in which T is replaced with S and sends to the LBS provider $Q = \{CR, n, m_1, m_2, \ldots, m_S, k, c_1, c_2, \ldots, c_n, c'_{11}, c'_{12}, \ldots, c'_{1m_1}, c'_{21}, c'_{22}, \ldots, c'_{2m_2}, \ldots, c'_{S1}, c'_{S2}, \ldots, c'_{Sm_S}, t_{S+1}, \ldots, t_T, pk_1, pk_{21}, pk_{22}, \ldots, pk_{2S}\}$. Based on Q, the LBS server computes $C_{\alpha,\beta,\ell_1,\ldots,\ell_{S-1}} = \prod_{\ell=1}^{m_S} {c'_{S\ell}}^{{d^k_{\alpha,\beta,\ell_1,\ldots,\ell_{S-1},\ell,t_{S+1},\ldots,t_T}}'} = {c'_{St_S}}^{{d^k_{\alpha,\beta,\ell_1,\ldots,\ell_{S-1},t_S,\ldots,t_T}}'}$ $(mod\ {N_{2S}}^2), \ldots, C_{\alpha,\beta} = \prod_{\ell=1}^{m_1} {c'_{1\ell}}^{C_{\alpha,\beta,\ell}}, C_\beta = \prod_{\alpha=1}^{n} {c_\alpha}^{C_{\alpha,\beta}} (mod\ {N_1}^2)$, and returns $(C_1, C_2, \ldots, C_n)$ to the user, who computes $w = r^e \mathsf{D}_{2S}(\ldots(\mathsf{D}_{21}(\mathsf{D}_1(C_j, sk_1), sk_{21})\ldots sk_{2S}) (mod\ N)$ and executes Steps 2 and 3 of Algorithm 6. In the end, the user obtains $d^k_{i,j,t_1,t_2,\ldots,t_T}$.

7 Security Analysis

We have proposed a basic and a generic kNN query protocols. we now analyze their security on the basis of our security model given in Section 4.

7.1 Location Privacy of Our Protocols

We analyze the location privacy of our basic kNN query protocol at first. In this protocol, Steps 1, 3, and 5 of Algorithm 1 and Step 1 of Algorithm 3 are related to the location (i,j) of the mobile user. Now we prove that this protocol has location privacy based on Definition 1 in Section 4. The location privacy of this protocol is built on the security of the Paillier public-key cryptosystem.

Theorem 3. *If the Paillier public-key cryptosystem is semantically secure, then our basic kNN query protocol described in Section 5 has location privacy according to Definition 1.*

Proof. Suppose our basic kNN query protocol does not have location privacy, then there is a PPT adversary $\mathcal{A}$ (the LBS provider) who has non-negligible advantage ϵ to break the location privacy of our protocol, i.e., winning Game 1. Now we use $\mathcal{A}$ to break the semantic security of the Paillier cryptosystem.

Given the public key pk_1 of the Paillier cryptosystem, we challenge two plaintexts 0 and 1 and the challenger randomly chooses $b \in \{0,1\}$ and returns $\mathsf{E}_1(b)$ to us. We construct our query $Q = \{CR, n, m, k, c_1, c_2, \ldots, c_n, c'_1, c'_2, \ldots, c'_n, pk_1, pk_2\}$ where $c_i = \mathsf{E}_1(b)$ and $c_\ell = \mathsf{E}_1(0)$) $(\ell \neq i)$ and send Q to the adversary $\mathcal{A}$, who gives us his guess i'. If $i' = i$, we output our guess $b' = 1$ and 0 otherwise.

When $b = 1$, Q is a real query and the adversary $\mathcal{A}$ wins the game with the probability $1/2 + \epsilon$, where ϵ is non-negligible. When $b = 0$, Q is independent of i and the adversary $\mathcal{A}$ can only win the game with $1/2$. Therefore, our advantage to break the Paillier cryptosystem (i.e., our guess $b' = b$) is $1/2 \cdot 1/2 + 1/2(1/2 + \epsilon) = 1/2 + \epsilon/2$, which is non-negligible. We break the semantic security of the Paillier scheme. It is in contradiction with the assumption of the theorem. Based on Definition 1, the protocol has location privacy and the theorem is true. □

Remark. In our basic kNN query protocol, our query Q is independent of j. Thus there is no way for the LBS provider to guess j with Q. The theorem ensure location privacy of the mobile user in the query generation.

Now, we analyze location privacy of the mobile user in the response retrieval according to Definition 4 in Section 4. We have

Theorem 4. *The Response Retrieval algorithm (Algorithm 3) of our basic kNN query protocol described in Section 5 has blindness according to Definition 4.*

Proof. With reference to Game 4 in Section 4, the adversary $\mathcal{A}$ (the LBS server) chooses any two distinct plaintexts m_0 and m_1 and encrypts them with RSA, i.e., $C_0 = {m_0}^e$ $(mod\ N)$ and $C_1 = {m_1}^e$ $(mod\ N)$. The challenger (the mobile user), given C_0, C_1, randomly chooses a bit $b \in \{0,1\}$ and a random number r $(0 < r < N)$ and computes the blinded ciphertext $C'_b = r^e C_b$ $(mod\ N)$. Because of the existence of the random number r, the adversary $\mathcal{A}$, given C'_b, cannot guess b chosen by the challenger correctly with a non-negligible advantage. According to Definition 4, the theorem is proved. □

Next, we analyze the location privacy of our generic kNN query protocol. The definition of the location privacy for our generic protocol can be obtained by replacing m and t with $(m_1, m_2, \ldots, m_T)$ and $(t_1, t_2, \ldots, t_T)$, respectively, in Game 1. Following the proof of Theorem 3, we can show

Theorem 5. *If the Paillier cryptosystem is semantically secure, our generic kNN query protocol described in Section 6 has location privacy and blindness.*

7.2 Query Privacy of Our Protocols

In our basic and generic kNN query protocols, the Paillier cryptosystem is used to hide the type t or the type attributes $(t_1, t_2, \ldots, t_T)$ of POIs the mobile user is interested in from the LBS server. Query privacy is also built on the security of the Paillier public key cryptosystem. According to Definition 2 in Section 4, we have

Theorem 6. *If the Paillier cryptosystem is semantically secure, our basic and generic kNN query protocols described in Section 5 and 6 have query privacy according to Definition 2.*

Following the proof of Theorem 3, we can prove Theorem 6.

7.3 Data Privacy of Our Protocols

Next, we analyze data privacy of the LBS server according to Definition 3 in Section 4. Data privacy of our protocols is built on the security of RSA with OAEP.

Theorem 7. *If RSA with OAEP is semantically secure, our basic and generic kNN query protocols described in Sections 5 and 6 have data privacy for the LBS server.*

Proof. With reference to Game 3 in Section 4, the adversary $\mathcal{A}$ (the mobile user) chooses any two distinct cloaking regions CR_0 and CR_1 with $n \times n$ cells such that k nearest POIs of the type t or the type attributes $(t_1, t_2, \ldots, t_T)$ in the cell (i, j) are same. The adversary generates a query Q to retrieve the k nearest POIs of the type t or the type attributes $(t_1, t_2, \ldots, t_T)$ in the cell (i, j) and sends Q, CR_0, CR_1 to the challenger $\mathcal{C}$ (the LBS server). The challenger $\mathcal{C}$ chooses a random bit $b \in \{0, 1\}$, encrypts CR_b with RSA and OAEP, and runs the Response Generation algorithm RG to obtain $R_b = \mathsf{RG}(Q, \mathsf{E}(CR_b))$, and then sends R_b back to $\mathcal{A}$, where $\mathsf{E}(CR_b)$ denotes the encryptions of all data in CR_b.

Since RSA with OAEP has semantic security [1], the adversary cannot distinguish $\mathsf{E}(CR_0)$ from $\mathsf{E}(CR_1)$ and he cannot distinguish R_0 from R_1. Therefore, the adversary $\mathcal{A}$, given R_b, cannot guess b chosen by the challenger correctly with a non-negligible advantage. Based on Definition 3, the theorem is proved. □

8 Performance Analysis

Now we analyze the performance of our basic and generic kNN query protocols and compare our basic protocol with some existing protocols.

8.1 Performance of Our Protocols

In this performance analysis, we consider the computation of modular exponentiations (Exp.) and ignore the computation of modular multiplications because the latter is much cheaper than the former. We also ignore the process of key generation because it can be pre-computed.

TABLE 1
Performance of Our Protocols ($|N| = \log_2 N$)

Component	*Algorithms 1-3*	*Algorithms 4-6*
User Comp.	$O(n+m)$	$O(n+\sum_{\lambda=1}^{T} m_\lambda + T)$
Server Comp.	$O(n^2 m)$	$O(n^2 \prod_{\lambda=1}^{T} m_\lambda)$
Comm.	$O((2n+m)\|N\|)$	$O(2n+\sum_{\lambda=1}^{T} m_\lambda)\|N\|)$

In our basic kNN query protocol (Algorithms 1-3), the mobile user needs to compute $n+m$ Paillier encryptions (about $n+m$ Exp.) in Algorithm 1, and 2 Paillier decryptions (about 2 Exp.) and 1 RSA encryption (about 1 Exp.) in Algorithm 3. So the total computation complexity of the mobile user is about $O(n+m)$ Exp. In Algorithm 2, the LBS provider needs to compute $n^2 m$ Exp. and the total computation complexity of the LBS provider is $O(n^2 m)$ Exp. In addition, the communication complexity is $O(2n+m)\log_2 N$ bits, where N is the RSA modulus.

In our generic kNN query protocol (Algorithms 4-6), the mobile user needs to compute $n + \sum_{\lambda=1}^{T} m_\lambda$ Paillier encryptions (about $n + \sum_{\lambda=1}^{T} m_\lambda$ Exp.) in Algorithm 4, and $T+1$ Paillier decryptions and 1 RSA encryption (about 1 Exp.) in Algorithm 6. So the total comp. complexity of the user is about $O(n + \sum_{\lambda=1}^{T} m_\lambda + T)$ Exp. In Algorithm 5, the LBS provider needs to compute about $n^2 \prod_{\lambda=1}^{T} m_\lambda$ Exp. and the total comp. complexity of the LBS provider is $O(n^2 \prod_{\lambda=1}^{T} m_\lambda)$ Exp. In addition, the comm. complexity is $O(2n + \sum_{\lambda=1}^{T} m_\lambda)\log_2 N$ bits.

Table 1 shows the performance of our basic and generic protocols.

8.2 Performance Comparison

We now compare our basic kNN query protocol described in Section 5 with PIR-based LBS query protocols [8], [9], [22], [23]. Because previous protocols do not consider the type of POIs, we set $m = 1$ and only consider Steps 1, 3, 5 in Algorithm 1 and Steps 1, 3 in Algorithm 2 in the comparison.

Ghinita et al.'s protocol based on [8], [9] has two stages: (1) retrieving the index of the cell where the mobile user is located using the Paillier cryptosystem [19]; (2) retrieving the POIs of the cell using the Kushilevitz-Ostrovsky PIR protocol [14]. If the cell where the mobile user is located has been already known to the mobile user, the first stage is unnecessary. We only consider the second stage in the comparison.

Paulet et al.'s protocol based on [22], [23] also has two stages: (1) retrieving the index and encryption key of the cell where the mobile user is located using the ElGamal cryptosystem [5]; (2) retrieving the POIs of the cell using the Gentry-Ramzan PIR protocol [7].

TABLE 2
Performance Comparison

Component	*Ghinita et al.*	*Paulet et al.*	*Our Basic Protocol ($m = 1$)*
User Comp.	$O(n)$ multi. + $O(R)$ exp.	generate (G, g), compute discrete log	$O(n)$ Exp.
Server Comp.	$O(Rn^2)$ multi.	$O(n^2)$ exp. + $O(Rn^2)$ multi.	$O(n^2)$ Exp.
Comm.	$O(Rn\|N\|)$	$O(n\|N\|)$	$O(n\|N\|)$

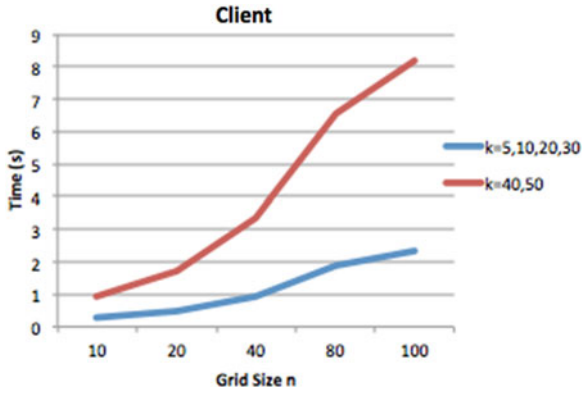

Fig. 4. Performance of our basic protocol for client (where $m = 10$).

We assume the cloaking region has $n \times n$ cells, the performance of Ghinita et al.'s protocol and Paulet et al.'s protocol against our basic protocol ($m = 1$) is shown in Table 2, where we assume the size of the modulus is 1,024 bits (at the same security level) and the size of the record in a cell is R. In addition, multi., exp. and Exp. denote a modular multiplication, a modular exponentiation with small modulus (e.g., 512 bits or 1,024 bits) and a modular exponentiation with large exponents (e.g., 2,048 bits), respectively.

By comparing the three protocols, we have found

1) In Ghinita et al.'s protocol and our protocol, the computation complexities for the mobile user are about the same. Paulet et al.'s protocol needs to generate a group (G, g) and compute a discrete logarithm. This process needs more time than other two protocols. But Paulet et al.'s protocol is independent of the size of the cloaking region.
2) The computation complexities for the server in the three protocols are about the same. Both Ghinita et al.'s protocol and our protocol allow parallel computation. Paulet et al.'s protocol needs to compute single modular exponentiation g^e which cannot be computed in parallel.
3) The communication overhead of both Paulet et al.'s protocol and our protocol are about the same. The communication overhead of Ghinita et al.'s protocol is about R times that of our protocol.

When the size of the record $R = 1{,}024$ bits, our protocol performs better than other two protocols in terms of both

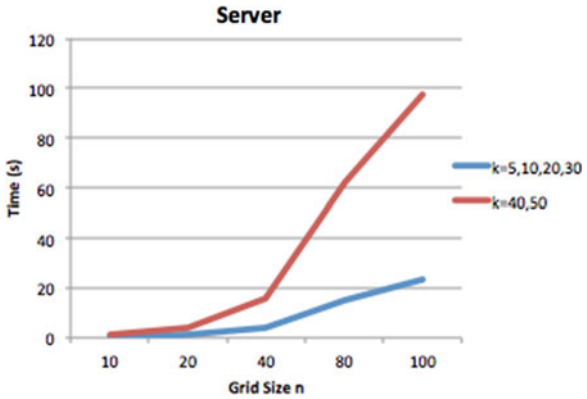

Fig. 5. Performance of our basic protocol for server with 20 computers running in parallel (where $m = 10$).

Fig. 6. Performance of our basic protocol for communication (where $m = 10$).

parallel computation and communication overhead. In particular, our protocol allows the mobile user to retrieve a type of POIs without revealing the type to the LBS provider.

9 Experimental Evaluation

We implemented our basic protocol and test its performance. The implementation was executed on a machine with an Intel Core i7-2600 processor at a clock speed of 3.40 GHz, and with 16 GB of RAM. The experiment used Linux as the operating system and is written using the C programming language. We used the GMP library for computations using large integers.

According the POI dataset[3] which contains 62,556 California place names, we construct our kNN database (grid = 1 km) with 10 types of POIs (school, lake, bridge, creek, hotel, farm, mine, golf course, hospital, and campground) for $k = 5, 10, 20, 30, 40, 50$, respectively, as described in the initialization.

The running times of our basic protocol in different settings for client and server are shown in Figs. 4 and 5 while the communication overhead is shown in Fig. 6.

We ignored public key initialisation and RSA encryptions of all data in the database D because these variables can be precomputed. In Fig. 4, the size of the RSA modulus is 1,024 bits when $k = 5, 10, 20, 30$ and 2,048 bits when $k = 40, 50$. Usually, $k = 20$ is sufficient big. In addition, when $n = 100$, the cloaking region covers a sufficient large area $10{,}000\ km^2$.

10 Conclusion

In this paper, we have presented a basic and a generic approximate kNN query protocols. Security analysis has shown that our protocols have location privacy, query privacy and data privacy. Performance has shown that our basic protocol performs better than the existing PIR-based LBS query protocols in terms of both parallel computation and communication overhead. Experiment evaluation has shown that our basic protocol is practical. Our future work is to implement our protocol on mobile devices.

3. http://chorochronos.datastories.org/?q=node/58

ACKNOWLEDGMENTS

The authors would like to thank the Editor, the Associate Editor, and anonymous reviewers for their valuable comments, which are very helpful for us to enhance our paper.

REFERENCES

[1] M. Bellare and P. Rogaway, "Optimal asymmetric encryption - how to encrypt with RSA," in *Proc. Eurocrypt*, 1994, pp. 92–111.
[2] B. Bamba, L. Liu, P. Pesti, and T. Wang, "Supporting anonymous location queries in mobile environments with privacygrid," in *Proc. 17th Int. Conf. World Wide Web*, 2008, pp. 237–246.
[3] A. R. Beresford and F. Stajano, "Location privacy in pervasive computing," *IEEE Pervasive Comput.*, vol. 2, no. 1, pp. 46–55, 2003.
[4] C. Y. Chow, M. F. Mokbel, and X. Liu, "A peer-to-peer spatial cloaking algorithm for anonymous location-based services," in *Proc. 14th Annu. ACM Int. Symp. Adv. Geograph. Inform. Syst.*, 2006, pp. 171–178.
[5] T. ElGamal, "A public-key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Trans. Inform. Theory*, vol. 31, no. 4, pp. 469–472, Jul. 1985.
[6] Y. Elmehdwi, B. K. Samanthula, and W. Jiang, "Secure k-nearest neighbor query over encrypted data in outsourced environments," in *Proc. IEEE 30th Int. Conf. Data Eng.*, 2014, pp. 664–675.
[7] C. Gentry and Z. Ramzan, "Single-database private information retrieval with constant communication rate," in *Proc. 32nd Int. Conf. Automata, Lang. Program.*, 2005, pp. 803–815.
[8] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan, "Private queries in location-based services: Anonymizers are not necessary," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2008, pp. 121–132.
[9] G. Ghinita, P. Kalnis, M. Kantarcioglu, and E. Bertino, "Approximate and exact hybrid algorithms for private nearest-neighbor queries with database protection," *GeoInformatica*, vol. 15, no. 14, pp. 699–726, 2010.
[10] G. Ghinita and R. Rughinis, "An efficient privacy-reserving system for monitoring mobile users: Making searchable encryption practical," in *Proc. 4th ACM Conf. Data Appl. Security Privacy*, 2014, pp. 321–332.
[11] H. Hu, J. Xu, C. Ren, and B. Choi, "Processing private queries over untrusted data cloud through privacy homomorphism," in *Proc. IEEE 27th Int. Conf. Data Eng.*, 2011, pp. 601–612.
[12] A. Khoshgozaran and C. Shahabi, "Blind evaluation of nearest neighbor queries using space transformation to preserve location privacy," in *Proc. 10th Int. Conf. Adv. Spatial Temporal Databases*, 2007, pp. 239–257.
[13] H. Kido, Y. Yanagisawa, and T. Satoh, "An anonymous communication technique using dummies for location-based services," in *Proc. Int. Conf. Pervasive Serv.*, 2005, pp. 88–97.
[14] E. Kushilevitz and R. Ostrovsky, "Replication is not needed: Single database, computationally-private information retrieval," in *Proc. 38th Annu. Symp. Found. Comput. Sci.*, 1997, pp. 364–373.
[15] M. F. Mokbel, C.-Y. Chow, and W. G. Aref, "The new casper: Query processing for location services without compromising privacy," in *Proc. 32nd Int. Conf. Very Large Data Bases*, 2006, 763–774.
[16] G. Myles, A. Friday, and N. Davies, "Preserving privacy in environments with location-based applications," *IEEE Pervasive Comput.*, vol. 2, no. 1, pp. 56–64, 2003.
[17] M. Naor and B. Pinkas, "Oblivious transfer with adaptive queries," in *Proc. 19th Annu. Int. Cryptol. Conf. Adv. Cryptol.*, 1999, pp. 791–791.
[18] R. Ostrovsky and W. Skeith, "A survey of single-database private information retrieval: Techniques and applications," in *Proc. 10th Int. Conf. Practice Theory Public-Key Cryptograph.*, 2007, pp. 393–411.
[19] P. Paillier, "Public key cryptosystems based on composite degree residue classes," in *Proc. 17th Int. Conf. Theory Appl. Cryptograph. Techn.*, 1999, pp. 223–238.
[20] B. Palanisamy and L. Liu, "Mobimix: Protecting location privacy with mix-zones over road networks," in *Proc. IEEE 27th Int. Conf. Data Eng.*, 2011, pp. 494–505.
[21] S. Papadopoulos, S. Bakiras, and D. Papadias, "Nearest neighbor search with strong location privacy," in *Proc. Very Large Data Bases*, 2010, pp. 619–629.
[22] R. Paulet, M. Golam Kaosar, X. Yi, and E. Bertino, "Privacy-preserving and content-protecting location based queries," in *Proc. IEEE Int. Conf. Data Eng.*, 2012, pp. 44–53.
[23] R. Paulet, M. Golam Kaosar, X. Yi, and E. Bertino, "Privacy-preserving and content-protecting location based queries," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 5, pp. 1200–1210, May 2014.
[24] R. Michael, "Digitalized signatures and public-key functions as intractable as factorization," MIT Lab. Comput. Sci., Cambridge, MA, US, Tech. Rep. MIT-LCS-TR-212, Jan. 1979.
[25] R. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 21, no. 2, pp. 120–126, 1978.
[26] R. Schlegel, C. Chow, Q. Huang, and D. Wong, "User-defined privacy grid system for continuous location-based services," *IEEE Trans. Mobile Comput.*, vol. 14, no. 10, pp. 2158–2172, Jan. 2015.
[27] P. Shankar, V. Ganapathy, and L. Iftode, "Privately querying location-based services with SybilQuery," in *Proc. 11th Int. Conf. Ubiquitous Comput.*, 2009, pp. 31–40.
[28] L. Sweeney, "k-anonymity: A model for protecting privacy," *Int. J. Uncertain. Fuzziness Knowl.-Based Syst*, vol. 10, pp. 557–570, 2002.
[29] S. Wang, X. Ding, R. H. Deng, and F. Bao, "Private information retrieval using trusted hardware," in *Proc. 11th Eur. Symp. Res. Comput. Security*, 2006, pp. 49–64.
[30] P. Williams and R. Sion, Usable PIR, in *Proc. NDSS*, 2008.
[31] W. K. Wong, D. W. Cheung, B. Kao, and N. Mamoulis, "Secure kNN computation on encrypted databases," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2009, pp. 139–152.
[32] B. Yao, F. Li, and X. Xiao, "Secure nearest neighbor revisited," in *Proc. IEEE Int. Conf. Data Eng.*, 2013, pp. 733–744.
[33] X. Yi, R. Paulet, E. Bertino, and V. Varadharajan, "Practical k nearest neighbor queries with location privacy," in *Proc. IEEE Int. Conf. Data Eng.*, 2014, pp. 640–651.
[34] M. L. Yiu, C. Jensen, X. Huang, and H. Lu, "SpaceTwist: Managing the trade-offs among location privacy, query performance, and query accuracy in mobile systems," in *Proc. IEEE Int. Conf. Data Eng.*, 2008, pp. 366–375.
[35] M. Youssef, V. Atluri, and N. R. Adam, "Preserving mobile customer privacy: An access control system for moving objects and custom proles," in *Proc. 6th Int. Conf. Mobile Data Manage.*, 2005, pp. 67–76.

Xun Yi is a professor with the School of Computer Science and IT, RMIT University, Australia. His research interests include applied cryptography, computer and network security, mobile and wireless communication security, and privacy-preserving data mining. He has published more than 150 research papers in international journals, such as *IEEE Transactions Knowledge and Data Engineering*, *IEEE Transactions Wireless Communication*, *IEEE Transactions Dependable and Secure Computing*, *IEEE Transactions Circuit and Systems*, *IEEE Transactions Vehicular Technologies*, *IEEE Communication Letters*, *IEE Electronic Letters*, and conference proceedings. He has undertaken program committee members for more than 20 international conferences. Recently, he has led a few of the Australia Research Council (ARC) Discovery Projects.

Russell Paulet received the bachelor's degree, the honour's degree, and the PhD degree from the Victoria University, Melbourne, Australia. His honour's thesis was in the field of image processing, where he analyzed the performance of edge detectors in the presence of various types of noise. The subject of the PhD degree was the design and analysis of cryptographic protocols in applications like private location based query. His research interests include applied cryptography, data mining and privacy-preserving data mining, applied mathematics, and computer algorithms.

Elisa Bertino is a professor of the Computer Science Department, Purdue University, and serves as research director of CERIAS and director of Cyber Center, Purdue University. Her main research interests include security, privacy, digital identity management systems, database systems, distributed systems, and multimedia systems. She received the 2002 IEEE Computer Society Technical Achievement Award for outstanding contributions to database systems and database security and advanced data management systems and the 2005 IEEE Computer Society Tsutomu Kanai Award for pioneering and innovative research contributions to secure distributed systems. She is a fellow of the IEEE and ACM.

Vijay Varadharajan is currently professor and Microsoft chair in Innovation in Computing at the Macquarie University. He has published more than 380 papers in international journals and conferences, has coauthored and edited eight books on information technology, security, networks and distributed systems, and holds two patents. His current areas of research interest include Web services security, cloud computing security, secure distributed applications, Trusted computing, security policies and management in distributed systems, internet security, secure mobile agents, security in Mobile networks, wireless security, secure E-commerce, security policies, models and architectures and protocols. He is a fellow of the British Computer Society (FBCS), a fellow of the IEEE, United Kingdom (FIEE), a fellow of the Institute of Mathematics and Applications, United Kingdom (FIMA), a fellow of the Australian Institute of Engineers (FIEAust) and a fellow of the Australian Computer Society (FACS).