Private Shortest Path Queries in outsourced database

Draft By:
Abdeltawab M. Hendawi (hendawi@cs.umn.edu)
Joe Naps (naps@cs.umn.edu)

ABSTRACT
Mobile devices equipped with positioning capabilities (e.g. GPS) can ask location-dependent queries to Location Based Services (LBS). In data outsourcing, the database owner (DBO) wishes to get benefits from the service provider computational, storage, and professional capabilities by getting data management services for his/her data with a reasonable price. The main challenge the DBO faces while outsourcing his/her data is privacy, user and/or data privacy. Data privacy refers to the content of the database and user privacy is users’ locations in their queries. The comfort of LBS comes with privacy versus efficiency cost. Various privacy violations caused by sharing sensitive location information with potentially malicious services have highlighted the importance of location privacy research aiming to protect user privacy while interacting with. We contribute in this paper as follows, firstly, we review the existing start-of-the-art literatures for various privacy preserving techniques in LBS. Secondly, we address the problem of how to answer shortest path queries as a (LBS) privately. We formalize two versions of the problem by making variations in our assumptions. Version 1 assumes that the user is trusted and data is valuable while version 2 assumes the user is untrustworthy and might collaborate with the service provider, and the data is not valuable. Both versions deal with untrustworthy service provider. As a solution for version 1, we propose a naïve solution base on shortest paths matrix with cryptographic (classic encryption), while the second solution is based on shortest path quad-tree and Order Preserving Encryption (OPE). To solve version 2, we propose a cloaking solution using shortest path R-Tree based index. Unlike shortest path quad-tree, our shortest path R-Tree based solution doesn’t assume the road network to be planar. It does deal with any kind of road network paths with storage cost reduction from $O(n^3)$ to $O(n^2)$. Finally, we show the effect of our proposed technique by empirical experiment.

1. INTRODUCTION
With the high demand on location based services and the emergency of cloud computing technology, the IT companies that provide such location based services to their users wish to get benefits from the cloud computing facilities by outsourcing their database management to some service provider. Database outsourcing is subcontracting a service of database management to a service provider which includes; DB administration, DBMS migration, Access control,
Maintenance, Performance adaptation, and Answering queries. These services will be provided to the database owner with a reasonable cost computed according to the required computational and storage resources [1], [10], [11]. The motivation behind this research is shown through the following example;

Assume that there is a company that collects data about restaurants, gas stations, supermarkets, hypermarkets, and pharmacies, and this company started in Minnesota. The data includes the location of the entity and information about the services and products offered by this entity. The participated entities give this company special offers, discounts and coupons, so the registered consumers in this company will get the services or/and the products with less price. The users usually need to ask about the shortest path to a certain entity (i.e. a certain pasta restaurant). At this moment, the company has the processing ability to answer all users’ queries with a reasonable response time. So there is no problem till now.

After its succeeding in Minnesota, this company decided to expand its work to cover all the states in the USA. At this moment, the problem appears. The company doesn’t have the capabilities to process and calculate shortest path queries to all users in the US. So it decided to outsource its database management to a database service provider and pay the cost for renting this service. Database outsourcing allows the company to gain the advantages of the service provider (SP) capabilities such as processing, storage, administration, maintenance, and migration with reasonable cost instead of buying expensive H/W devices and S/W packages and hiring professional teams with high cost. However database outsourcing will solve the processing problem for this company, but some troubles will float to the surface.

The first problem is that the user doesn’t wish to reveal his/her data to the service provider as s/he trusts the company only and doesn’t know anything about the service provider. The users’ queries include their start locations and their points of interest, destinations. So the company wishes to serve its users while preserving their location privacy to known by the SP.

The second problem is that the company itself might consider its data as its valuable assets. So it doesn’t want anyone else including the service provider and/or the user to know the content of its data.

2. PROBLEM STATEMENT

We formalize two versions of the problem in this research according to assumptions variations. Version 1: is concerning with how to outsource the spatial database to allow the service provider to answer the users’ shortest path queries while protecting both the spatial database content and users’ privacy. In this version, we assume that the database owner (DBO), the company in the previous example, does not trust the service provider (SP), at which the spatial database (SDB) will be sited, but the user, who sends the queries and receives the results, is trusted which means there is no collaboration between any user and the SP and also means that the service provider can’t be a user.
## PROBLEM STATEMENT VERSION 1

| **Input** | A road network as a connected undirected graph $G=(V,E)$  
|           | Each vertex $v_i$ has coordinates $(x_i, y_i)$  
|           | Each edge connects between two different vertices and has a cost weight, $e_i = (v_j, v_k, w_i)$ |
| **Output** | Exact answer for shortest path queries to user |
| **Objective** | Perform spatial data outsourcing to a service provider while answering users’ shortest path queries  
|           | Preserve the privacy for users’ locations  
|           | The outsourced data is worthy so we need to protect it from the SP  
|           | Minimize the response time  
|           | Minimize the storage cost |
| **Assumptions** | The road network is static  
|           | The service provider is untrustworthy  
|           | The user is trusted and there is no collaboration between user and the SP |

In version 2: we assume that the database owner (DBO) does trust neither the service provider nor the user as there might be a collaboration between the SP and malicious users. But we also assume the data is not worthy. So our aim concern in this version is to protect users’ locations in their queries. To the best of our knowledge, this is the first work that considers both the service provider and the user are not trusted and there might be collaboration between them to attack other users’ privacy.

## PROBLEM STATEMENT VERSION 2

| **Input** | A road network as a connected undirected graph $G=(V,E)$  
|           | Each vertex $v_i$ has coordinates $(x_i, y_i)$  
|           | Each edge connects between two different vertices and has a cost weight, $e_i = (v_j, v_k, w_i)$ |
| **Output** | Exact answer for shortest path queries to user |
| **Objective** | Perform spatial data outsourcing to a service provider while answering users’ shortest path queries |
Preserve the privacy for users’ locations  
Minimize the response time  
Minimize the storage cost

**Assumptions**  
The road network is static  
The service provider is untrustworthy  
The user is also untrustworthy  
There might be a collaboration between the user and the SP  
The data itself is not worthy so we don’t worry about it

3. RELATED WORK

The most related work is [7] in which the authors use dummies to achieve the privacy for users’ locations, however they assume that the user is trusted and the data is not worthy. In [20] the authors introduce shortest paths quad-tree to efficiently store pre-computed all combination of shortest paths in the road network; however they aren’t concerned with the privacy. Also the network has to be planar to be modeled by their index structure. The following figure summaries the classification for the four approaches used to preserve the privacy while answering Location-Based Queries (LBS) in outsourcing architecture. The classification is based on the methods used to achieve the privacy.

![Figure 1 Private LBS Approaches Classification](image-url)
3.1. Anonymization, Dummies and Cloaking-Based Approaches

The earlier work on location privacy focused on protecting a user’s private location information by disguising it among $K - 1$ other user locations or extending it from a point location to an area (spatial extent). With the first approach, user u, as well as $K-1$ other user locations form an anonymity set which is sent to the server instead of user’s precise location. Similarly, with cloaking techniques, the resulting cloaked region (which contains u and several other users) is sent to the server. These techniques try to ensure the user’s location cannot be distinguished from the location of the other $K-1$ real users, K-1 false positions (dummies) or the exact user location within the cloaked region is not revealed to the un-trusted server responding to location queries. Depending on the method used, the un-trusted server executes the query for every object in the anonymity set or for the entire cloaked region. Several techniques based on cloaking, K-anonymity, and dummies have been proposed in the literature to reduce the probability of identifying a user’s location [1], [2], [3], [4], [5], [6], [7].

Limitations

Cloaking, dummies, and K-anonymity approaches have some important limitations: First, the majority of cloaking approaches rely on a trusted intermediate party to “anonymize” user locations which means all queries should involve the anonymizer during the system’s normal mode of operation. The anonymization can also be performed in a decentralized fashion among users which means each user has to trust all other users in the system with her location. In other words, while users do not trust the location server, they either have to trust another third party, or all other users. Second, a limitation of cloaking techniques in general is that either the quality of service or overall system performance degrades significantly as users choose to have more strict privacy preferences. Third, the concept of K-anonymity does not work in all scenarios. For example, in a less populated area, the size of the extended area can be prohibitively large in order to include $K-1$ other users. Even worse, not enough number of users may be subscribed to the service to construct the required cloaked region.

Merits

The methods in this approach are considerably efficient in terms of computational and communication overhead [1].
### 3.2. Transformation-Based Approaches

This class of approaches try to mitigate some of the weaknesses of the techniques discussed above are based on query transformation to prevent the server from learning information about user locations. In [8], space filling curves are utilized as one-way transformations to encode the locations of both users and points of interest into an encrypted space and to evaluate a query in this transformed space. The transformed space maintains the distance properties of the original space which enables efficient evaluation of location queries in the transformed space. Subsequently, upon receiving transformed query results, users can reverse the transformation efficiently using the trapdoor information which is only provided to them and are protected from the server. Recently, Yiu et al. proposed a framework termed SpaceTwist to blind an un-trusted location server by incrementally retrieving points of interest based on their ascending distance from a fake location near the query point termed the anchor point [9]. In [10], and [11], the authors use transformation to protect the data privacy not the user location.

**Limitations**

The result for LBS such as nearest neighbor query of [8] is approximate and [9] suffers from several privacy leaks and costly computation/communication if exact results and strict privacy are required simultaneously.

**Merits**

The key advantage of transformation-based approaches over the anonymization, dummies, and cloaking-based techniques is the elimination of the need for a trusted third party during the query processing. Furthermore, [8] provides very efficient query processing without compromising privacy and [9] utilizes the existing query processing index structures present in non-privacy aware servers which makes it readily applied to existing location servers.

### 3.3. Cryptographic-Based Approaches

This class of techniques blinds the un-trusted party (i.e., the server or another user) by utilizing secure multi-party computation schemes. Zhong, Goldberg and Hengartner propose three solutions to what they define as the “nearby-friend problem” [12]. The problem is defined as allowing users to learn information about their friends’ locations if and only if their friends are actually nearby. The three protocols are all efficient in terms of the amount of computation and communication required by each party. Each protocol is an instance of a multi-party computation scheme with certain strengths and restrictions (in terms of number of messages transferred and the resilience to a malicious party). Finally, Zhong et al. provide two protocols aiming at protecting user locations in LBS. While the first protocol allows a user to share his/her location information with other users via an un-trusted server, the second protocol enables a dating
service where a user learns whether other users with similar profiles (found by the server) are located in the same region she is located [13]. This protocol assumes the entire user profile is known to the server, and the server first finds any potential matches between a user and all other users. The server then sends all matched profiles to the requester so that she can blindly compare their locations with her own location. Similar to the other protocols discussed above, a multi-party computation protocol is proposed which involves the requester, the dating service and any other matched user.

**Limitations**

The fundamental disadvantage of the protocols discussed in this category is their high computation or communication complexity when being used for spatial query processing. For instance, in [14] the distance between query point and each and every point of interest must both be computed or transferred to the client, i.e., $O(n)$ computation or communication complexity where $n$ is the size of the database. This is because the points of interest are treated as vectors with no exploitation of the fact that they are in fact points in space. Therefore, the main limitation of cryptographic-based techniques discussed above is the loss of spatial information via encryption. This loss either results in a linear scan of the entire database if used to evaluate a spatial query as in [14], or makes the protocol unusable for spatial query processing as in [12], and [13].

**Merits**

The main advantage of the three methods in this approach discussed above is their strong privacy guarantees. Building their framework on well-known cryptographic primitives and widely used one-way functions, these protocols do not suffer from privacy leaks of cloaking-anonymization-dummies, and transformation-based methods. Furthermore, their problem-specific designs allow very efficient implementations of the protocols mostly involving low computations and few message transfers.

### 3.4. PIR-Based Approaches

The approaches studied in this category are based on the solutions proposed to the well-known problem of Private Information Retrieval (PIR). These approaches construct private spatial indexes on top of PIR operations to provide efficient spatial query processing, while the underlying PIR scheme guarantees privacy. We discuss two location privacy schemes based on hardware-based [15,16] and computational PIR [17] protocols. The former approach superimposes a regular grid on the data and uses PIR to privately evaluate range and KNN queries. The latter technique supports approximate and exact nearest neighbor query evaluation by utilizing various 1-D and 2-D partitioning to index the data and then restructuring partitions into a matrix that can be privately queried using PIR. The study in [18] proposes employing PIR
to provide location privacy is which presents an architecture that uses PIR and trusted computing to hide location information from an un-trusted server. With this approach, PIR is used to prevent the un-trusted location server from learning user locations and trusted computing is used to ensure users that the PIR algorithm and other services provided by the server are only performing the operations as intended. In fact, similar to hardware-based PIR, [18] places a trusted module as close as possible to the un-trusted host to disguise the selection of records. However, the proposed techniques do not specifically focus on spatial query processing (such as range and KNN) and the proposed architecture is not implemented.

**Limitations**

The proposed PIR protocols are still expensive and require a significant amount of server resources. All database records still have to be processed at the server. Sion et al. argue in [19] that the cost of privately retrieving database items from the server is significantly higher than sending the entire database to the client.

**Merits**

The proposed PIR approaches do not suffer from the privacy vulnerabilities of cloaking-dummies-anonymization and transformation-based approaches or the prohibitive communication and computation costs of cryptographic-based techniques.

**4. PROPOSED SOLUTION**

As we mentioned in section 2, we deal with two different problem versions. Figure 2 shows our proposed solution architecture for addressing version 1 of the problem.

![Figure 2 Proposed Solution architecture for problem version 1](image-url)
All our proposed solutions are based on pre-computing all combinations of shortest paths from each vertex to others in the given road network. With storing the resulted shortest paths in matrix structure will cost $O(n^3)$ of storage. Procedure 1 below shows the steps for our proposed solution for version 1 problem using shortest paths matrix.

**Procedure 1: private shortest paths matrix**

- Compute the shortest path from each vertex to others (using Dijkstra alg.)
- Apply encryption for both the vertices and the pre-computed shortest paths:  
  \[ v'_i = E(v_i), \quad \text{and} \quad v_i = D(e_i) \]
  where $e_i$ is the encrypted value for vertex $v_i$ and $E$ is the encryption function and $D$ is the decryption function
- Store the shortest path in Matrix $M(NXN)$ ($N$: number of vertices)
- Build hash table where \( H(v'_i, v'_j) \implies P'(v'_i, v'_j) \)
  where \( P'(v'_i, v'_j) = E(P(v_i, v_j)) \quad \text{if} \quad P \text{ is the path from } v_i \text{ to } v_j \)
  // we might use different Enc. Func. for paths
- Embed the encryption key inside the user utility that will perform the encryption/decryption at user’s device
- Send the encrypted matrix to the sp
- The user encrypt his/her query using the sent key
- The SP evaluate users’ encrypted queries over the encrypted data and send an encrypted result
- The user decrypt the received encrypted result to get answer for his query

To solve version 1 using shortest paths quad-tree, we use Order Preserving Encryption (OPE) to encrypt the tree index. OPE allows us to navigate the shortest path quad-tree to answer users’ queries with no need to decrypt neither the query nor the tree index. Procedure 2 below shows the steps for our proposed solution for version 1 problem using shortest paths quad-tree.
For version 2 of the problem, we use shortest path R-Tree based index to store the pre-computed shortest paths, then we do some data analysis to get the data distribution (under investigation) so we can generate the cloaking region directly from the resulted distribution. Procedure 3 shows the steps for building the shortest path R-Tree.

**Procedure 3: Building shortest paths R-Tree**

1. Get the vertex with the smallest x value, if there are more than one, start from the one with the smallest y value
   - Output: vertex in hand \((v_i)\)
2. Compute all shortest paths from the \(v_i\) to all other vertices \(v_j \in V\) such that \((v_j.x = v_i.x\) and \(v_j.y > v_i.y\)) or \((v_j.x > v_i.x)\)
   - Output: New shortest paths added to the list of all paths
3. Repeat 1 and 2 until finishing the road network (each time it generates \(n-1, n-2, \ldots 2, 1, 0\) and the total paths added for all vertices in the map = \(n(n+1)/2\)
4. Make a reduction by deleting redundant paths or these are subsets of other paths: which means if a path \(P_1 = <v_1, v_2, v_3, v_4>\) and \(P_2 = <v_2, v_3>\) then we delete \(P_2\) and keep \(P_1\).
   - Output: Final list of longest shortest paths
5. Find the building blocks for each path \(P(v_i , v_j)\) by dividing the path at each node \(v_k\) with \(d > 2\) inside this path into two paths \(P(v_i , v_k)\) and \(P(v_k, v_j)\), i.e. as in figure 3, \(P(v_6, v_9)\) will be divided into \(P(v_6,v_8)\) and \(P(v_8,v_9)\)
   - Output: List of all Building blocks paths
6. Make a reduction on the list of building block as in 4
   - Output: Final list of blinding blocks
7. Store the building blocks in a table \(<\text{Block_id}, \text{Block_Vertices}>\)
   - Output: Building blocks table
8. Redefine each Path in 4 by references to its building blocks in 5 and store the paths into paths table \(<\text{Paths},e, \text{Set_Of_Blocks}>\)
   - Output: Table of paths defined in terms of building blocks
9. Build the shortest path R-Tree to index the table of paths from 8
Figure 3 (a) shows a simple road network divided into a set of MBRs that represent the final list of shortest paths in figure 3 (b), and (c) is the shortest path R-Tree for the set of paths in this example.

![Figure 3](image)

**List of paths**

- P1 = <1,2,3,5>  \[ R1 = \{(1,3), (4,5)\} \]
- P2 = <1,2,3,4>  \[ R2 = \{(1,1), (6,3)\} \]
- P3 = <1,2,3,6>  \[ R3 = \{(1,3), (7,5)\} \]
- P4 = <5,3,6>  \[ R4 = \{(2,4), (7,5)\} \]
- P5 = <5,3,4>  \[ R5 = \{(2,1), (6,5)\} \]
- P6 = <4,3,6>  \[ R6 = \{(4,1), (7,5)\} \]

Figure 3: (a) A simple road network and the final MBRs (b) Final list of paths and their regions (c) is the R-Tree index for the table of shortest paths.
5. EXPERIMENT

Currently, we are implementing our proposed solution using the three different data structures used in this research to privately answer shortest path queries. We use synthetic sample data to test our proposed solutions. Also we will conduct a comparison to differentiate between them in terms of efficiency versus privacy.

6. CONCLUSION AND FUTURE WORK

In this work we propose a solution for the private shortest paths queries problem. We formalized two different versions of the problem and propose a suitable solution for each one. Current work aims at completing the implementation and test the proposed solution.

As a future work, we are working on completing this research, especially for problem version2,

- Build attacking Scenarios as a validation method for testing the privacy
- Use both the synthetic and real data to test both the storage and response time cost.
- Describe in details the modules inside the secure coprocessor and how it could be used to add more security layers by doing some cloaking and cashing for the results.

We believe that there is still work to be done to complete this research and we wish to complete is this summer.

7. REFERENCES

1. Ghinita G., Kalnis P., Kantarcioglu M., Bertino E. A Hybrid Technique for Private Location-Based Queries with Database Protection. SSTD (2009)


18 Hengartner, U.: Hiding location information from location-based services. In:MDM 2007