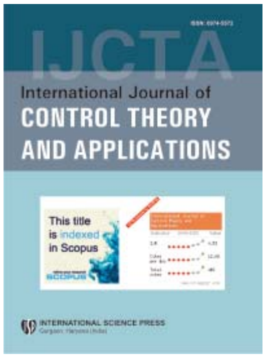


International Journal of Control Theory and Applications

ISSN : 0974–5572

© International Science Press

Volume 9 • Number 42 • 2016

An Effective Approach to Top-*k* Spatial Query Processing Authentication

Ambili T[a] Jisha P. Abraham[a] and Neethu Subash[a]

[a]*Department of Computer Science and Engineering, M.A College of Engineering, Kothamangalam, Kerala, India*

E-mail: ambili10892@gmail.com, jishaanil@gmail.com, neethu.subash@gmail.com

Abstract: With the development of location-based services in mobile internet, spatial query processing is becoming an integral part of many new applications. However, ensuring the query integrity has become a major concern. The system for query processing consists of a data collector, data contributors, location-based service providers (LBSPs), and system users. The query results are provided to the users by LBSP, after purchasing point of interest (POI) dataset from the collector. But practically, LBSPs are untrusted and may return fake query results. LBSPs may modify their data sets by deleting some reviews or adding fake reviews and return tampered query results. As a solution for this, the proposed system presents novel schemes for users to detect fake spatial snapshot and moving top-k query results. Even though there are several existing systems none of the schemes consider spatial top-k queries. Unlike the existing system that uses a Merkle hash tree for authentication, the proposed system presents a MIR tree that minimizes the communication cost and computation overhead. The system also draws a comparison between the two authenticated data structures. The key idea behind the scheme is that the data collector pre-computes and authenticates some auxiliary information about its data set. The main objective of the system is to enable the user to verify the authenticity and correctness of the query result returned by the LBSP.

Keyword: *Spatial Databases; Query Processing; Authentication; Point of Interest(POI); Location Based Services; Verification Object; Authenticated Data Structure(ADS).*

1. INTRODUCTION

With the popularity of the positioning enabled devices and with the rise of the mobile internet, location-based services (LBSs) have become an important part in our daily activities in recent years. Such services provide the users with location aware query experiences based on their locations. Also owing to the growing popularity of social networks, it has become more and more convenient and motivating for mobile users to share with others their experience of all kinds of points of interests (POIs) such as bars, restaurants, accommodation, coffee shops, and hotels. Meanwhile, it has become a commonplace for people to perform various spatial POI queries at online location-based service providers (LBSPs) such as Google and Yelp. As probably the most familiar type of spatial queries, a spatial (or location-based) top-*k* query asks for the POIs in a certain region and with the highest k ratings for a given POI attribute.

Spatial queries are being supported in real-life applications in a variety of ways, such as Google Maps where points of interest can be retrieved, Foursquare where geo-tagged documents can be retrieved, and Twitter where tweets can be retrieved. Spatial keyword querying is also receiving an increasing interest in the research community where a range of techniques have been proposed for efficiently processing spatial keyword queries. Since many of the real-world applications have requirements to support the top-*k* spatial query, it has attracted attention from both academia and industry communities.

However a current top-*k* query service addresses two essential drawbacks. First, individual LBSPs have small data sets comprising POI reviews. In this case, suppose a user queries a particular POI within a query region and if the dataset is limited, it does not provide complete query result to the user. Second, LBSPs may alter their data sets by deleting a few reviews or including fake audits and return customized query results based on the POI"s that are willing to pay or against those that refuse to pay. Even if LBSPs are not malicious, they may even return unfaithful query results under the influence of various attacks such as the Sybil attack, where the same attacker can submit many fake reviews for the same POI. The above mentioned reasons necessitate the development of mechanisms that will allow users to authenticate the top-*k* spatial query results that the LBSP returns. The users need to verify the authenticity and correctness of query results through a proof, called verification object (VO) returned by the LBSP. In particular, the authenticity means that the original spatial textual data in the result set is not tampered with, while the correctness implies that no valid result set is missing.

A basic approach for tackling this problem is to use an authenticated data structure (ADS) called Merkle hash tree (MHT). Based on Merkle hash tree, a best-first traversal algorithm can be employed to process the top spatial queries. Meanwhile, VO is generated based on the nodes which have been visited. After receiving query results and VO, the hash value of the root of Merkle hash tree is reconstructed by the user in a bottom-up manner to verify the authenticity and correctness of query results. However, this approach is highly inefficient as it generates large verification set in the construction of VO. Thus, it will result in a tremendous communication overhead between the LBSP and the user. In addition, it also leads to excessive computation cost at the user side. Therefore, the top-k spatial query processing still remains a very challenging problem. To reduce the VO size and make VO more suitable for the authentication, the proposed system presents another data structure called Merkle-IR tree (MIR). MIR tree combines the concepts of Merkle hash tree (MHT) and IR tree. To verify the authenticity of query results, the user needs to scan VO to reconstruct the hash value of the root of MIR-tree and compare it against the root signature using the DO's public key.

2. RELATED WORK

In the proposed system, work is most related to data outsourcing [4], for which we can only review representative schemes due to space constraints. The framework of data outsourcing was first introduced in [4], in which a data owner outsources its data to a third-party service provider who is responsible for answering the data queries from either the data owner or other users. In general, there are two security concerns in data outsourcing, data privacy and query integrity [5]. Ensuring data privacy requires the data owner to outsource encrypted data to the service provider, and efficient techniques are needed to support querying encrypted data. The seminal paper [4] proposed executing SQL queries over encrypted data using bucketization. Their strategy is to process as much of a query as possible by the service providers, without having to decrypt the data. Decryption and the remainder of the query processing are performed at the client side. Since then, a number of works have appeared on executing various queries over encrypted data.

Another line of research has been devoted to ensuring query integrity, *i.e.*, that a query result is indeed generated from the outsourced data (the authenticity requirement) and contains all the data satisfying the query (the correctness requirement). Query authentication was first studied in the Cryptography literature. Various types of query have been studied, including range queries, *k*NN queries, shortest-path queries, spatial skyline queries etc. Another line of research was to ensure data privacy against untrusted service providers. A common approach is to encrypt the dataset before outsourcing it to the third-party service provider, and various techniques have been proposed to enable efficient query processing over encrypted data. Early research focuses on one-dimensional range queries as well as multi-dimensional range queries.

More recent work targets secure ranked keyword search, access control, and circular range query over encrypted data. The Merkle Hash Tree (MH-tree) [6] is a main-memory binary tree that hierarchically organizes hash values. After building the tree, the data owner signs the hash value stored in the root of the MH-tree, using a public key digital signature scheme. To authenticate one-dimensional range queries, Devanbu et al. [7] sort the database records on the query attribute and index them by a MH-tree. A combination of the MH-tree and the range search tree [8] is exploited in [7] to authenticate multidimensional range queries. Martel et al. [9] extend the MH-tree concept to arbitrary search DAGs (Directed Acyclic Graphs), including dictionaries, tries, and optimized range search trees. Goodrich et al. [10] present ADSs for graph and geometric searching. These techniques, however, focus on main-memory and are highly theoretical in nature. The first disk-based ADS in the Database literature is the VB-tree [11], which authenticates the soundness, but not the completeness, of 1D range results. A subsequent signature chaining approach [12, 13] authenticates both soundness and completeness.

3. PROPOSED WORK

The proposed system presents authentication schemes for secure query processing. To faithfully answer a top-*k* query, a LBSP need return the correct top-*k* POI data records as well as proper authenticity and correctness proofs constructed from authenticated hints. The authenticity proof allows the query user to confirm that the query result only consists of authentic data records from the trusted data collector's data set, and the correctness proof enables the user to verify that the returned top-*k* POIs are the true ones satisfying the query. The system considers two types of queries, snapshot top-*k* query and moving top-*k* query. A snapshot top-*k* query includes the interested POI category, a query region R, and an integer *k*. In contrast, a moving top-*k* query can be viewed as a continuous sequence of snapshot top-*k* queries, where the user is interested in the top-*k* POIs in a moving region defined with respect to the user's current location.

3.1. System Architecture

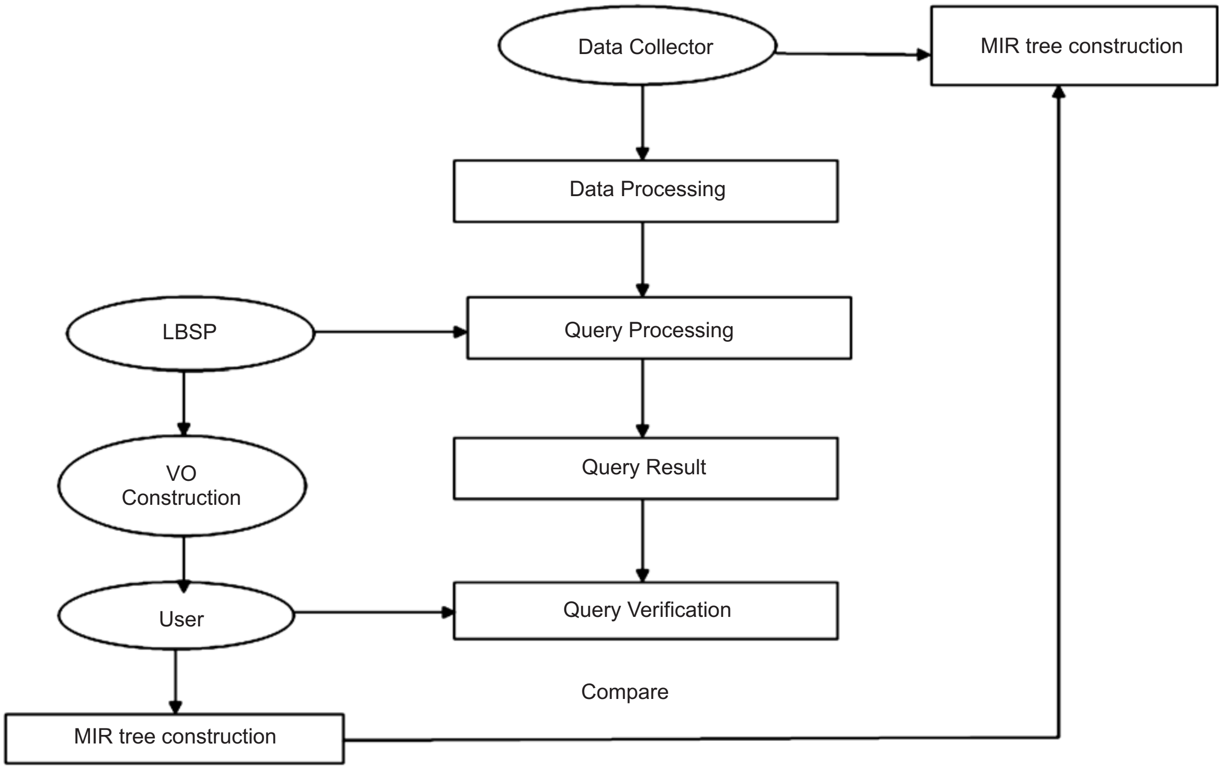


Figure 1: System Architecture

The system for query processing comprises a data collector, data contributors, LBSPs, and query users. Data contributors are common people who submit POI reviews to the data collector. The data collector aggregates the review submissions and also employs necessary measures to filter out fake reviews from malicious data contributors. The data collector sells POI reviews in the form of a location-based data set to individual LBSPs. Each LBSP operates a website for users to perform queries over the purchased data set. The users submit their queries through these websites and obtain the required query result. The data set is classified according to POI categories such as restaurants, accommodation, banks and ATMs etc. Each POI category contains a unique record which contains the information regarding each POI category such as its name, location, ratings etc. The geographic area covered by the data collector is partitioned into M $\geq$ 1 equally-sized non-overlapping zones. For every zone *i*, *ni* denote the number of POIs, and POIi, *j* and Di, *j* denote the *jth* POI and its corresponding data record respectively. The system includes three phases, data pre-processing, query processing, query result and verification. The system also proposes an ADS for generating the VO, which can be used to verify the query result returned by the LBSP.

3.2. MIR Tree

The Merkle-IR-tree (MIR-tree) is developed based on the IR-tree. It is a combination of Merkle hash tree and IR tree. IR tree is a tree data structure mainly used to handle location based queries. In IR tree each leaf node is associated with an inverted file (IF). Each entry summarizes the spatial distances and text relevancies of the entries in its child node. The leaf nodes of MIR-tree are identical to those of IR-tree. Each entry in the leaf nodes of MIR-tree is represented by an object while each entry in the internal nodes of MIR-tree is represented by a tuple, (R, H(R)), where R is an MBR and H(R) is the hash value of R. The hash value of each entry in the internal nodes of MIR-tree is computed by the binary concatenation of the entries and inverted file included in its child node. To process the top-k query over MIR-tree, a best-first traversal algorithm is employed to retrieve the ranking list of k objects. The VO is generated based on the entries and inverted files included in the nodes which have been visited. After the query processing, VO is returned to the user together with the query results.

3.3. System Design

The proposed system involves creating authenticated hints by chaining ordered POIs in every zone via cryptographic hash functions and then combining the POIs in different zones through a MIR tree. It involves the following phases:

3.3.1. Data Preprocessing

The pre-processing of data is carried by the data collector after obtaining the review submissions from the data contributor. Initially the geographic area covered by the data collector is partitioned into equally-sized nonoverlapping zones. Each zone includes a number of POI's and each of the POI contains a data record. The data record for POI includes its name, location and reviews. The data collector preprocesses its data set before selling it to LBSPs. For each zone the data collector sorts the data records corresponding to POI's according to the attribute rating to generate an ordered list of data records. It then computes an index for every data record. The index is computed using a combination of the location information, attribute rating and hash value of data record.

$$\Phi_{i,j} \;=\; <l'_{i,j}, \mathrm{A}'_{i,j,q}, \mathrm{H}(\mathrm{D}'_{i,j})>$$

The data collector chains index function using a cryptographic hash function to ensure the correct order among them. The collector recursively computes hash function as follows:

$$h_{i,j} \;=\; \begin{cases} \mathrm{H}(\chi), & j = n_i + 1, \\ \mathrm{H}(h_{i,j+1} \,||\, \Phi_{i,j}), & 1 \leq j \leq n_i, \end{cases}$$

Finally, the data collector builds a MIR tree over index function. The data collector builds a separate MIR tree for every attribute in each of the POI category and signs every root using its private key.

3.3.2. Query Processing

The LBSP purchases the data sets of interested POI categories from the data collector. For every POI category chosen by the LBSP, the data collector returns the original data set and the signatures on root of MIR tree, and all the intermediate results for constructing the MIR tree. Initially the LBSP need to process the query submitted by the user. The query processing needs to consider both the spatial proximity and textual relevance of the query. This involves generating a ranking list of the objects combining both the aforementioned features. To process a query it also considers candidate zones, zone either completely or partially covered by the query region. The LBSP returns the top-*k* results based on the ranking list of the objects. The LBSP also returns a verification object comprising the authenticity and correctness proofs to the user along with the query result. The LBSP determines the set of candidate zones, to identify the number of POIs in zone with attribute ratings higher than lowest attribute rating. For each candidate zone:

$$\mathrm{X}_{i,j} = \begin{cases} \mathrm{D}'_{i,j}, & \text{if } l'_{i,j} \in \mathrm{R}, \\ \varnothing_{i,j}, & o.w., \end{cases}$$

Then it generates zone information for each of the candidate zones based on the number of POI‟s and attribute ratings. The zone informations are generated as follows:

1. **Case 1:** if $n_i = 0$, $\mathrm{S}_i = < i >$
2. **Case 2:** if $n_i = 1$, $\mathrm{S}_i = < i, \mathrm{X}_i, 1 >$
3. **Case 3:** if $n_i \geq 2$ and $\tau_i = 0$, $\mathrm{S}_i = <i, \Phi_{i,1}, h_{i,2} >$
4. **Case 4:** if $n_i \geq 2$ and $n_i > \tau_i \geq 1$,
   $\mathrm{S}_i = <i, \mathrm{X}_{i,1}, \ldots\ldots \mathrm{X}_{i,\tau i}, \Phi_{i,\tau i+1}, h_{i,\tau i+2}>$
5. **Case 5:** if $n_i = \tau_i \geq 2$, $\mathrm{S}_i = <i, \mathrm{X}_{i,1}, \ldots\ldots \mathrm{X}_i, \tau_i>$

Finally it returns the query result along with the verification object (VO) to the user. The verification object (VO) comprises the necessary information required by the user to reconstruct the root of MIR tree.

3.3.3. Query-Result Verification

For authenticity verification, the user checks if every piece of information in the query result can lead to the same root of MIR tree matching the data collector's signature. The authenticity and correctness verification is done by a plugin developed by data collector at his web browser. For this the user first extracts the POI information from the query results and locates the candidate zones. Based on the POI information the zone information which is being generated by the LBSP is retrieved. The index values are extracted and reconstruct the root of MIR tree. It then verifies whether it matches the MIR tree root generated by the data collector. To perform correctness verification, the user first checks if zones encloses the query region. Then it checks whether there are exactly *k* data records in the query result with POI locations all in query region, which correspond to the top-*k* POIs.

3.3.4. Authentication Framework

The challenge of authenticating a top-*k* result involves the design of a compact VO that achieves low communication cost and less authentication time and computation overhead. The user extracts the top-*k* result from the VO and authenticates it. This involves computing the ranking scores of the objects in the VO and obtains the top-*k* objects. The user then re computes the root of the MIR tree using an auxiliary set and VO and verifies them against the decrypted root spatial and word signatures. The main aim is to re-construct the MIR tree traversal, *i.e*., guaranteeing the entries in the VO are from the original MIR-tree. Then, the user verifies the correctness of the top-*k* result by checking whether the ranking score computed is no worse than those of all the other entries in the VO.

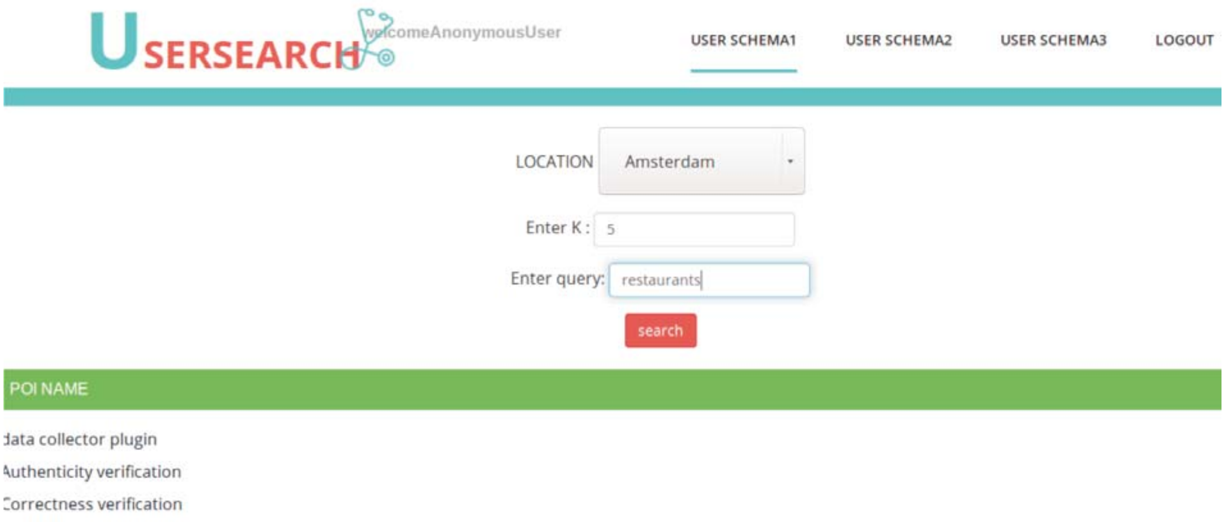
3.3.5. VO Construction

The proposed system presents ADS, the MIR-tree. The VO is constructed based on the fact that only a minimum number of objects and MIR tree entries need to be inserted into the VO. It also involves constructing a verification set covering the objects with ranking scores smaller than the maximum score in the top-*k* result. The VO is computed using a depth first traversal of the MIR-tree. The construction of a VO, involves computing a traversal string, which tracks the search in the MIR-tree. The traversal string is composed of the identifiers of the tree entries and objects added to the VO. The traversal string is required in order to avoid having duplicate entries in the VO.

3.3.6. Query Processing in MIR tree

```
1. Put each entry $R_i$ in the root in to a priority queue
2. Initialize VO with “’[’, each $R_i$ and $H(R_i)$, inverted file associated with root, ‘]’ ”
3. While $k$ objects not found do
   a) Pick entry with smallest score from priority queue
   b) If picked entry is an internal node $R_i$ then
      • Put each entry $R_j$ in $R_i$’s child node into priority queue
      • Replace $R_i$ and $H(R_i)$ with “’[’, each $R_j$ and $H(R_j)$, inverted file associated with $R_i$’s child node, ‘]’”
   c) Else
      • If picked entry is an object
      • Put into the result set
4. Return result set and VO
```

4. RESULT



LOCATION Amsterdam

Enter K : 5

Enter query: restaurants

POI NAME

data collector plugin

Authenticity verification

Correctness verification

Figure 2: Query Result

The proposed method is being evaluated using a dataset from POI plaza. The method also presents a comparison between the Merkle hash tree and MIR tree based on authentication time, communication cost and computation time. The proposed method is also studied under varying settings, including the performance on data sets, varying number of requested results *k*, varying numbers of query keywords and varying speeds of moving queries.

4.1. Cost at LBSP

It is evaluated based on running time. The running time is computed as a function of number of requested results. In all cases tested, the running time of MIR tree is lower than Merkle hash tree due to procedure of VO generation and multiple trees accessed.

4.2. Cost between LBSP and User

It is evaluated based on VO size. The length of verification object largely affects communication cost and computation overhead between LBSP and user. In all cases tested, it has been found that MIR tree generates a compact VO than Merkle hash tree

4.3. Cost at User

The last performance metric is based on the authentication time at the user-side. The authentication operation consists of two aspects. One involves hashing operations, and the other is based on re-computing the ranking scores of returned entries in VO. Based on the tests performed it has been analyzed that the authentication time of MIR tree is lower than Merkle hash tree. Since the leaf nodes of MIR tree contains inverted file the cost of authentication is minimized in MIR tree.

5. CONCLUSION

The proposed system considers a novel distributed system for collaborative location-based information generation and sharing. This paper proposes a new authenticated data structure, the MIR-tree, to efficiently authenticate top-*k* spatial query results, thus guaranteeing the authenticity and correctness of a top-*k* result. A verification object for authenticating the top- k results queries is also being proposed. Unlike the existing system for authentication such as Merkle hash tree the proposed system can minimize the communication cost and computation overhead in generating the verification object. Thus it can enable efficient spatial query processing. Moreover, the schemes proposed in this paper can also be applied to a broader set of query types, such as collective spatial keyword query, etc. This work may open up many promising directions for future work.

REFERENCES

[1] R R. Zhang, Y. Zhang, and C. Zhang, "Secure Top-k Query Processing via Untrusted Location-Based Service Providers," Proc. IEEE INFOCOM '12, Mar. 2012.

[2] H. Hacigumus, S. Mehrotra, and B. Iyer, "Providing Database as a Service," Proc. IEEE 18th Int'l Conf Data Eng. (ICDE), Feb. 2002.

[3] W.-S. Ku, L. Hu, C. Shahabi, and H. Wang, "Query Integrity Assurance of Location-Based Services Accessing Outsourced Spatial Databases," Proc. Int'l Symp. Advances in Spatial and Temporal Databases, July 2009.

[4] Merkle, R. A Certified Digital Signature. CRYPTO, 1989.

[5] Devanbu P., Gertz M., Martel C., Stubblebine S., Authentic Data Publication over the Internet Journal of Computer Security 11(3): 291-314, 2003.

[6] Berg M., van Kreveld M., Overmar M., Schwarzkopf, Computational Geometry: Algorithms and Applications. Springer, 1997.

[7] Martel C., Nuckolls G, Devanbu P, Gertz M, Kwong, A, Stubblebine S, A General Model for Authenticated Data Structures. Algorithmica, 21-41, 2004.

[8] Goodrich M, Tamassia R, Triandopoulos N, Cohen R, Authenticated Data Structures for Graph and Geometric Searching, CT-RSA, 2003.

[9] Pang H, Tan K.. L, Authenticating Query Results in Edge Computing, ICDE, 2004.

[10] Pang H, Jain A, Ramamritham K, Tan K. L, Verifying Completeness of Relational Query Results in Data Publishing. SIGMOD, 2005.

[11] Narasimha M, Tsudik G, Authentication of Outsourced Databases Using Signature Aggregation and Chaining, DASFAA, 2006.

[12] F. Chen and A. Liu, SafeQ: Secure and Efficient Query Processing in Sensor Networks, Proc. IEEE INFOCOM‟10, pp. 1-9, Mar. 2010.

[13] R. Zhang, J. Shi, Y. Liu, and Y. Zhang, Verifiable Fine-Grained Top-K Queries in Tiered Sensor Networks, Proc. IEEE INFOCOM‟10, Mar. 2010.

[14] R. Merkle, Protocols for Public Key Cryptosystems, Proc. IEEE Symp. Security and Privacy (S&P‟80), pp. 122- 134, Apr. 1980.

[15] B. Sheng and Q. Li, Verifiable Privacy-Preserving Range Query in Sensor Networks, Proc. IEEE INFOCOM‟08, pp. 46-50, Apr. 2008.

[16] B. Hore, S. Mehrotra, and G. Tsudik, A Privacy-Preserving Index for Range Queries, Proc. 30th Int‟l Conf. Very Large Data Bases (VLDB‟04), pp. 720-731, Aug. 2004.

[17] M. L. Yiu, Y. Lin, and K. Mouratidis. Efficient verification of shortest path search via authenticated hints. In ICDE, pages 237–248, 2010.

[18] [18] Y. Yang, D. Papadias, S. Papadopoulos, and P. Kalnis. Authenticated join processing in outsourced databases. In SIGMOD, pages 5–18, 2009.