International Journal of Database Theory and Application
Vol.9, No.10 (2016), pp.229-240
http://dx.doi.org/10.14257/ijdta.2016.9.10.19

Research on MR-Tree Spatial Query Authenticated Index Introduced Neighbor Relationship

Xiaofu Wei, Shenglin Li, Zuofei Tan, Kaiwen Luo and San Zhang

Department of Information Engineering, Logistical Engineering University, Chongqing, China
E-mail:1514347327@qq.com

Abstract

In database outsourcing, the data owner delegates the tasks of data management to a third-party service provider. As the service provider may be untrusted or susceptible to attacks, query authentication is an essential part. Merkle R-tree (MR-tree) is one of the most efficient authenticated index that combines Merkle hash tree with R-tree. MR-tree can provide an efficient range query authentication, however, as it uses the traditional R*-tree query structure in neighbor queries, a large number of unnecessary nodes may be accessed, and that can affect the efficiency of the query. In this paper, the neighbor relationship is introduced into the construction of MR-tree, and we propose a new index structure, called VMR-tree that incorporates the Voronoi diagram into MR-tree. In order to utilize VMR-tree index structure, we propose algorithms for spatial nearest neighbor queries and experiments to verify it has a better efficiency in spatial neighbor query authentication.*

Keywords*: Database outsourcing, MR-tree, Voronoi diagram, kNN query*

1. Introduction

With the development of geographic information system and satellite positioning technology, there is an increasing demand for the spatial data query. Due to the limitations of resources and technology, spatial data owners are often difficult to deal with a large number of query requests [1]. Therefore, the spatial data owners outsource their spatial database to the third party service providers to provide query services. In the outsourced database model, a query request from customers is executed by the third party server. As the third party server is not trusted and easy to be attacked, the query results returned to the users may have safety problems [2-3]. Therefore, it is necessary to provide a verification mechanism to ensure the effectiveness, correctness and completeness of the data returned to the user. At present, MR-tree [4] is widely used in high dimensional spatial database query authentication, and in the execution of the range query authentication, MR-tree has a good efficiency. However, when the nearest neighbor query is carried out, MR-tree needs to transform the nearest neighbor query into the range query. In the process of transformation, MR-tree needs to make the depth first traversal of the nodes, so as to find the query object. Due to the minimum bounding rectangle (MBR) of MR-Tree nodes in the high dimension space, there are a lot of overlapping areas [5]. In the query traversal algorithm, it is shown that many unnecessary nodes are accessed and the complexity of I/O is increased [6]. In this paper, the Voronoi diagram of spatial data is introduced into the construction of MR-tree, which can reduce the number of nodes accessed by MR-tree in spatial neighbor query, and improve the efficiency of the query.

ISSN: 2005-4270 IJDTA
Copyright ⓒ 2016 SERSC

2. Background

2.1. Outsourced Spatial Database Query Authentication

The concept of outsourcing database was first proposed by Hacigumus *et al.* in 2002[7]. Figure 1 is a framework of database outsourcing service, the data owner (DO) outsources its database to the service provider (SP), and the SP is responsible for the management of the spatial database and provides query services for the client. After obtaining the result from the SP, the client needs to confirm whether the data is from the data provider, and whether the data is correct and complete. So, the authentication mechanism is required to verify the result of the query obtained by the client, and many database query verification techniques are studied [8]. Weiwei Cheng et al propose a spatial digital signature chain to conduct the query authentication of the spatial database [9]. By constructing authentication chain for each point in each partition and all spatial data partitions, the authentication information is added to a spatial data structure. However, this approach generates significant authentication overhead, and it consumes sizeable bandwidth for client server communication. Ling Hu et al propose a multi-dimensional query authenticated structure based on Voronoi diagram named VN-Auth [10], which separates the authentication information from the spatial index. VN-Auth allows efficient query processing and provides smaller verification objects than spatial index based authentication. However, VN-Auth increases the storage overhead of the data owner, and it can only handle point object query.

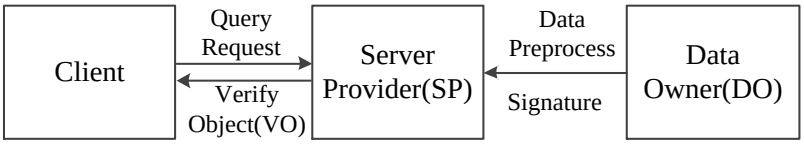


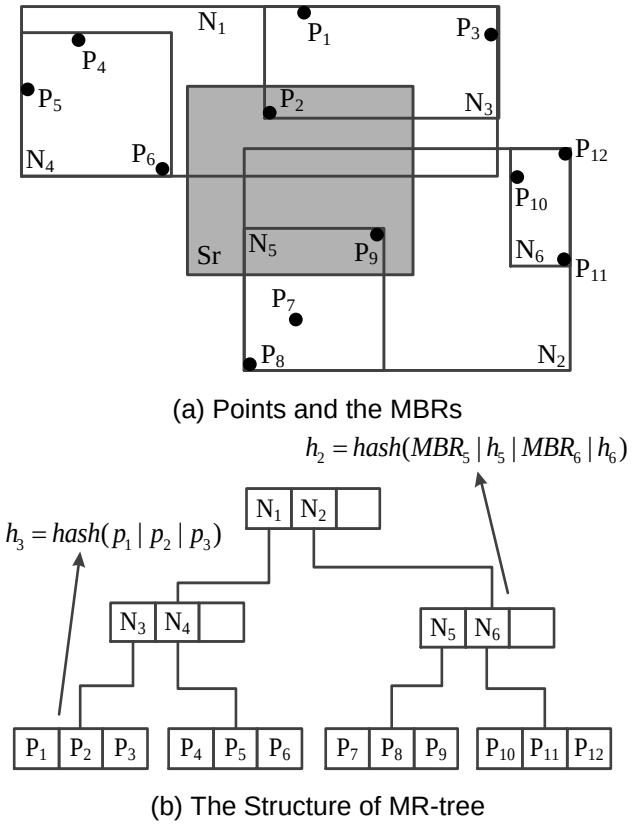
Figure 1. The Framework of Database Outsourcing

MR-tree is developed on the basis of Merkle hash tree [11] and R*-tree [12]. It computes the digests from the leaf node to the root node, and the data owner signed the digest of root node with a private key. The digests include two types, the digest of the leaf node and the digest of the intermediate node. The digest of the leaf node is obtained by computing the hash value of all the object points in the leaf. As shown in Figure 2b, the digest of N_3 is $h_3=hash(P_1|P_2|P_3)$. The digest of the internal node is computed by the MBR information of all child nodes and the digests of all child nodes. As shown in Figure 2b, the digest of N_2 is $h_2=hash(MBR_5|h_5|MBR_6|h_6)$. When a query is executed, the server provider produces *verification objects (VO)* according to the client's query condition. The *VO* mainly consists of two parts: (i) all the leaf nodes that have been visited, (ii) the MBR of all cut nodes and the corresponding digests of them. According to the *VO* returned by the SP, the client can reconstruct the digest of the root node, and then compare with the signature root digest that is decrypted by the public key. If two digests match successfully, the completeness of the query result can be verified by judging whether the query scope is overlapped with the MBR of the non leaf nodes in the *VO*.

When a range query authentication is conducted, for example, querying the shaded Sr in the tree of Figure 2. The SP executes query algorithm, visiting the nodes from root to leaf that enter the shaded Sr, and they are N_1, N_2, N_3, N_5. After executing the query algorithm on the SP, the authentication information is recorded as *VO*. So, *VO*=[[[P_1, P_2, P_3],[N_4_*MBR*,N_4_h]], [[P_7, P_8, P_9],[N_6_*MBR*,N_6_h]]]. After that, the SP sends the *VO* and the signature h_{root} to the client. The client can reconstruct the MR-tree by the obtained *VO*. First the client computes h_3 by [P_1, P_2, P_3], then computes h_1 by [N_4_*MBR*,N_4_h]] and h_3. Similarly, h_2 can be computed, and then we can get h_{root} by h_1 and h_2. Second the client uses the public key of the DO to decrypt the signature h_{root}. Finally, the client compares

 Copyright ⓒ 2016 SERSC

the two h_{root}, and the query result will be correct if they are agree with each other. In addition, the completeness of the query result can be verified by judging that no MBR of non leaf nodes in the *VO* (*e.g.*, N_4, N_6) overlaps the query area Sr.



(a) Points and the MBRs

(b) The Structure of MR-tree

Figure 2. MR-Tree and a Range Query

2.2. Voronoi Diagram

Voronoi diagram, also called Tyson polygon or Dirichlet diagram, it is a set of polygons composed of the vertical split line of two adjacent points [13]. Taking two dimensional space as an example. It is a plane division method based on distance in computational geometry. In the plane there are n non coincident seed points, the plane is divided into n regions, so that the distance that a point to its own seed point is nearer than the distance the point to any other seed points. A given set of points $P=\{p_1,p_2,\ldots,p_n\}\subset R^2$, and $2<n<\infty$. When $i\neq j$, $p_i\neq p_j$ and $i,j\in I_n=\{1,2,\ldots,n\}$, the region of Voronoi can be given as $VP(p_i)=\{p|d(p,p_i)\leq d(p,p_j)\}$ where $d(p,p_i)$ is the nearest distance between p and p_i (the nearest distance refers to minimum Euclidean distance [14]), the region formed by p_i is called the Voronoi polygon(VP).the diagram composed of VP is called Voronoi diagram(VD) that is defined as $VD(P)=\{VP(p_1),VP(p_2),\ldots,VP(p_n)\}$. Voronoi polygons that share the same edge are called adjacent polygons. Their generating points are called adjacency generating points. Figure 3 shows an example of Voronoi diagram.

Copyright ⓒ 2016 SERSC

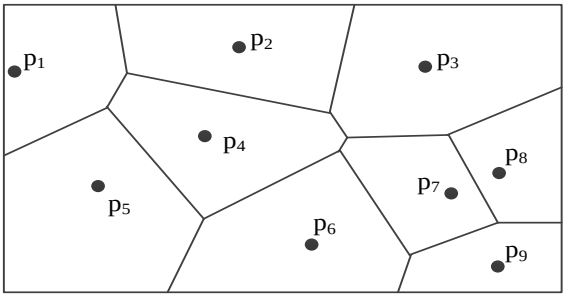


Figure 3. Voronoi Diagram

The Voronoi diagram has three main properties [15].

Property 1: For any given set of points P, the Voronoi diagram of P is unique.

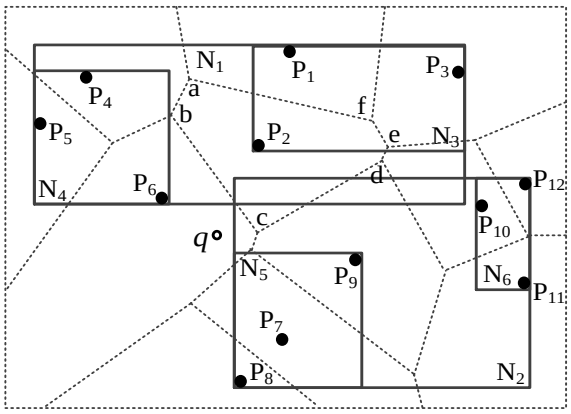
Property 2: In a Voronoi diagram P, for a point $q \in VP(p_i)$, the first nearest neighbor of P to q is p_i, and the second nearest neighbor of P to q is among the Voronoi neighbors of p_i. If $p_m \in VP(p_m)$, and $VP(p_m)$ sharing a edge with $VP(p_i)$, we can call p_m a Voronoi neighbor of p_i.

Property 3: the max number of vertices of a VP is six, so the number of the Voronoi neighbors of p_i cannot exceed six. That is an important property for a kNN query.

3. VMR-Tree Query Authenticated Index

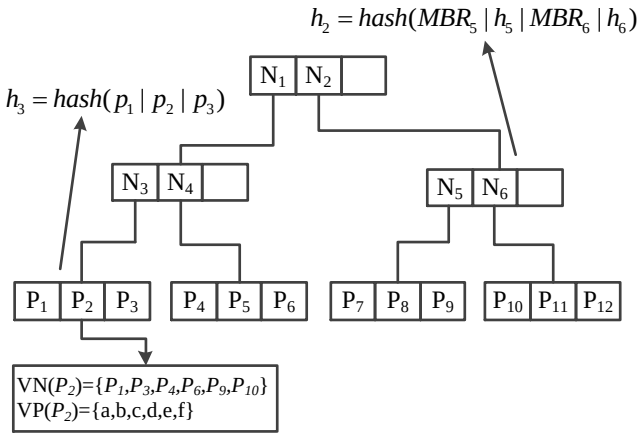
3.1. Structure of VMR-Tree

VMR-tree is based on MR-tree and combined the Voronoi neighbor relationship of the data objects. The Voronoi diagram of the data objects and Mr-tree index are stored on the SP, the creation of Voronoi diagram does not affect the base structure of the MR-tree, and the only need is to add Voronoi neighbor information of the data object to the leaf nodes of the MR-tree. Figure 4 is a schematic diagram of the VMR-tree structure.



(a) Voronoi Diagram and the Point MBRs

 Copyright ⓒ 2016 SERSC



(b) The Structure of VMR-tree

Figure 4. VMR-Tree

As we can see that the Figure 4a is based on the Figure 2a and a layer of Voronoi diagram of P is added, which divides the plane area into non intersecting polygons. Each point $P_i \in P$ is corresponding to one polygon region and each region is corresponding to one point. The distance that any point in the scope of a polygon region to the corresponding point P_i is less than the distance it to any other corresponding points in P. In this way, the non related data objects are related through the Voronoi diagram. Figure 4b shows the structure of VMR-tree, it is based on the MR-tree structure shown in Figure 2b, and it adds Voronoi neighbors(VN) and Voronoi polygon(VP) of P. From the definition of Voronoi diagram, we can know that *VN(P_2)* indicates the collection whose Voronoi polygon shares a common edge with the Voronoi polygon of P_2, that is {P_1,P_3,P_4,P_6,P_9,P_{10}}. *VP(P_2)* indicates the vertex positions of the Voronoi polygon of P_2, that is {a,b,c,d,e,f}. When performing spatial nearest neighbor queries, we can find the adjacent objects of a certain object quickly by its VN, and we can judge whether a certain object is the nearest neighbor of the query object by its VP. That can effectively reduce the number of unnecessary nodes accessed in a query.

3.2. kNN Query Authentication

The k Nearest Neighbor (kNN) Query is that when a dataset P and a point *q* are given, to find the result set P_0 satisfying the following conditions: (i) $P_0=\{p_1,\ldots,p_k\} \in P$, (ii) for any $p_i \in P_0$ and $P \ni p_j \notin P_0$, we have $D(q,p_i) \leq D(q,p_j)$ [16]. Because MR-tree is composed of Merkle hash tree and R*-tree, when a spatial neighbor query is performed in MR-tree, in fact it is accordant with the way of R*-tree to perform. At present, the most efficient kNN algorithm of R*-tree is the Best First Search (BFS) algorithm [17]. The BFS algorithm traverses R*-tree from the root node to the leaf node. First, the root node of R*-tree is inserted into the query queue, and the queue has only one node. Then, we expand the first and only node, sort the resulting sub nodes by the minimum distance between their MBRs and the query object, and the node with minimum distance in the first of the queue. And then continue to expand the first node in the queue, and sort the resulting sub nodes with the remaining nodes of the original queue. In this way, the first node in the queue is continuously expanded, and then the queue is sorted, until the first element in the sorted queue is a specific object (*e.g.*, P_i) rather than a node. Then this object (P_i) is the nearest neighbor of the query object. If k=1 in the kNN query, the object (P_i) is the final result. If k>1, the algorithm will continue to execute, continue to expand the first node in the queue (not the first element), until the first k elements in the queue are all specific objects. Then, the final result is the collection of the first k elements.

Copyright ⓒ 2016 SERSC

After Voronoi diagram introduced, the front part of the step is also to expand the first node of the queue (r_1) continuously by the BFS algorithm when a kNN query is conducted. The difference is that if r_1 is a leaf node, we extract the child (p_i) of r_1 that is nearest to the query object (q), and then judge that whether q is included in the area of VP(p_i). If q is included in the area of VP(p_i), we can judge that p_i is the nearest neighbor of q according to the Property 2 of the Voronoi diagram. Otherwise, If q is not included in the area of VP(p_i), we repeat the previous step, sort the queue, and expand r_1 continuously. When r_1 is a leaf node, the child of r_1 is extracted as the front. After the nearest neighbor of q is found, and if k>1, we will no longer be in accordance with the BFS algorithm to continue to expand the first node of the queue, but extract the VN and VP of the getting nearest neighbor of q. It can be judged by Property 3 that the second nearest neighbor must be in the VN of the first nearest neighbor. So, the second nearest neighbor can be found easily by comparing the VN of the first nearest neighbor. Similarly, the third nearest neighbor must be in the VN of the first and second nearest neighbors, and in this way, the k-th nearest neighbor of q can be found successively. The pseudo code of the algorithm is shown in Figure 5.

```
01 Initialize queue R;
02 Append root to R;
03 while do
04  Expand the first node r1;
05   If（r1 is leaf）then
06    Get nearest object(pi) from the child of r1;
07     if（VP(pi) include q）then
08        for(1:k)  do
09           Compare objects in VN, add to list;
10        end for
11       return list;
12      end if
13   end if
14  sort R in ascending order;
15 end while
```

Figure 5. Pseudo Code of kNN Query

In the following, we take the 2NN query of q in Figure 4 as an example to illustrate the algorithm. The algorithm first initializes the queue, adds the root node to the queue R. Then, expanding the root and sorting R, we can get R={(N_2,3.5),(N_1,6)}. The (N_2,3.5) stands for that the distance from N_2 to p is 3.5. Continuing to expand the first node N_2, we can get R={(N_5,5), (N_1,6), (N_6,47)}. In expanding N_5, as N_5 is the leaf node, we need to compare the distance from P_7, P_8, and P_9 to q respectively. As D(P_7) =24<D(P_9)<D(P_8), we extract the Voronoi polygon $VP(P_7)$, and find that $q \notin VP(P_7)$. Referring back to R, we get R={(N_1,6), (P_7,24), (P_9,26), (P_8,29), (N_6,47)} after sorting. Expanding N_1, we get R={(N_4,12), (N_3,19), (P_7,24), (P_9,26), (P_8,29), (N_6,47)}. In expanding N_4, as N_4 is the leaf node, we find $q \in VP(P_6)$. So, P_6 is the nearest neighbor of q. Extracting the Voronoi neighbor of P_6, we get $VN(P_6)$={P_5,P_4,P_2,P_9,P_7,P_8}. After comparing the distance in $VN(P_6)$ respectively, we can get the minimum distance D(P_2)=19. Thus, the second nearest neighbor of q is P_2, and the 2NN query result is {P_6,P_2}.

After getting the 2NN query result {P_6,P_2}, we need to convert the result into range query, so that to get the VO. We first draw a circle taking the point q as center and D(P_2)

 Copyright ⓒ 2016 SERSC

as radius, then we get the range query region as shown in Figure 6, denoted as *Sr(q)*. After that, the nodes are visited from root to leaf that enter the shaded *Sr(q)*, and they are {N_1, N_2, N_3, N_4, N_5}. So, we can get *VO*=[[[P_1, P_2, P_3], [P_4, P_5, P_6]], [[P_7, P_8, P_9], [N_6_*MBR*,N_6_*h*]]].

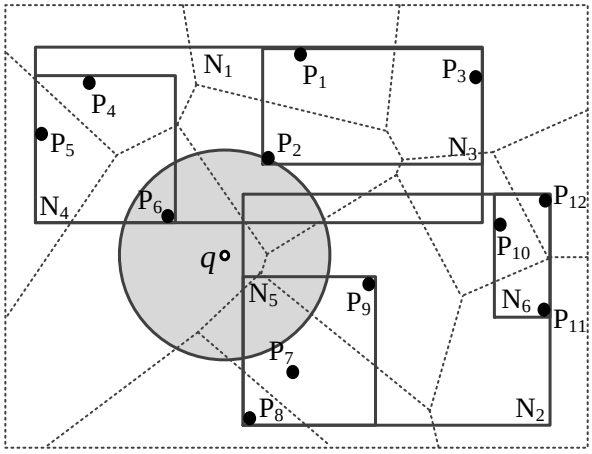


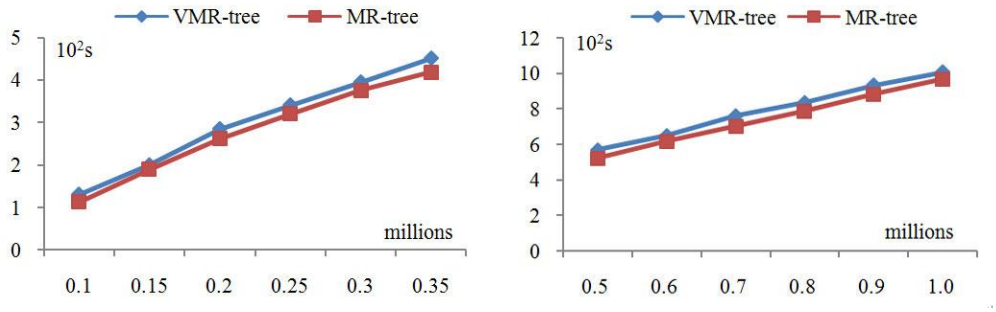
Figure 6. Range Query Region

4. Experimental Evaluation

In order to evaluate the performance of VMR-tree, we performed some experiments. This experiment is deployed on three PC with the Microsoft Windows operating system, one simulates the DO, one as the SP, and one as the client. Computer configurations are: 2.5GHz CPU, 4GB RAM. Our experiments are performed on two datasets:(i) PRS dataset that stores 374,921 locations of parking regulation signs in New York City and can be obtained from New York City Department of Transportation [18], and (ii)CSC dataset that includes 1 million locations with the combination of some small datasets and a custom dataset [19]. We extract the datasets randomly by a suitable sampling rate to get different cardinalities, and respectively compare the time spent on the construction of VMR-tree and MR-tree as well as the storage size of them. And according to change the value of k, we compare the efficiency of the kNN query authentication.

In order to compare the construction time of VMR-tree and MR-tree on different size data sets, we randomly sample data from PRS and CSC two datasets, and finally get 6 datasets with different sizes. Then we calculate their construction time respectively. The VMR-tree construction time includes the R*-tree build, Voronoi build and the calculation of node digest. The construction time of MR-tree includes R*-tree build and node digest calculation. The construction time does not include the encryption time of the digest information of the root node. The experimental results are shown in Figure 7.

Copyright ⓒ 2016 SERSC

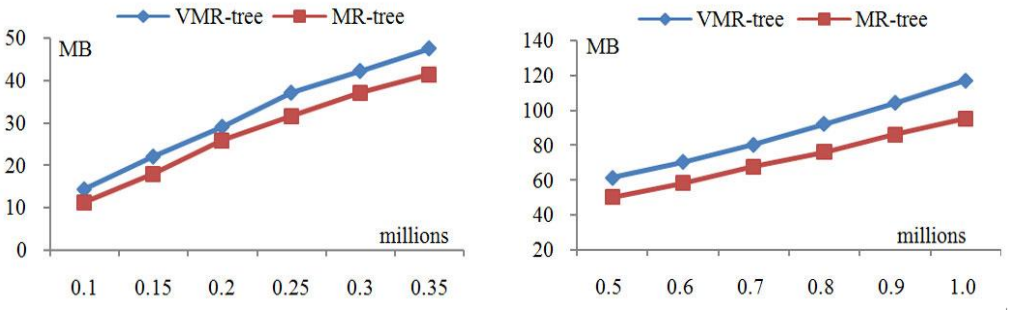


(a) PRS (b) CSC

Figure 7. Construction Time vs. Data Cardinality

It can be seen from Figure 7 that the overall construction time of the VMR-tree is longer than MR-tree, and this is due to the need of the Voronoi diagram construction in the VMR-tree. But in general, the construction times are not much difference between the two structures.

The next experiment is to compare the storage size of VMR-tree and MR-tree. As the comparison method of the construction time, we use the datasets that are randomly sampled from PRS and CSC, and calculate their storage sizes respectively. The result is shown in Figure 8.



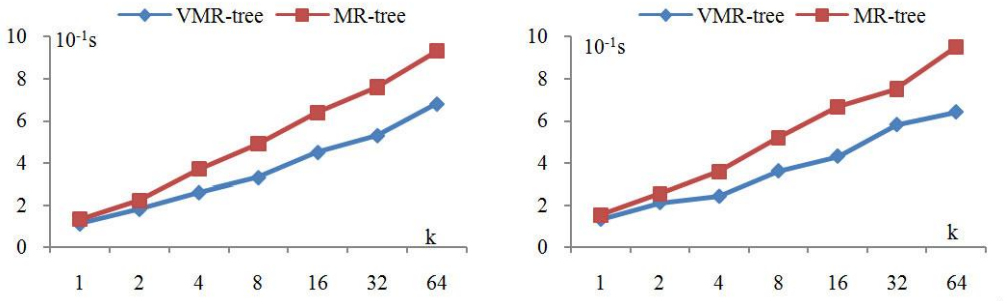
(a) PRS (b) CSC

Figure 8. Storage Size vs. Data Cardinality

As can be seen from Figure 8, VMR-tree structure size is larger than MR-tree. This is because the VN and VP of the index objects are stored in the leaf nodes, so that each leaf node increases a part of the storage overhead. By the property of the Voronoi diagram, we can know that one object's Voronoi neighbor is max to 6, so the VN and VP occupied storage overhead is not very large. In Figure 8, the average storage overhead of each dataset increases by about 25%.

The next sets of experiments are comparing the efficiency of the kNN query verification of VMR-tree and MR-tree. We use the two datasets PRS and CSC, and compare the efficiency through the query and verification time recorded in different k conditions. First a value of k is given, and then the client submits a kNN request. We calculate the time difference between the submitted request and the received message of the client from the SP. For each k value we conduct 10 groups of tests, and finally get the average value of the 10 values. Finally we take the average value as corresponding test results of the k value. Figure 9 shows the final test results.

Copyright ⓒ 2016 SERSC



(a) PRS (b) CSC

Figure 9. Query Authenticated Time vs. K

From the test results of two datasets shown in Figure 9, we can see that regardless of the size of the k value, the average query authentication time of VMR-tree is shorter than that of MR-tree, and the advantage of VMR-tree is more obvious with the increase of k value. This is because when searching for the i-th nearest neighbor of the query object, VMR-tree only need to search in the VN of the first i-1 nearest neighbors, and doesn't need to expand the node and sort the queue continually as MR-tree.

5. Conclusions

In this paper, we construct a new query authenticated index by introducing the Voronoi neighbor relationship of data objects to MR-tree structure, and describe the structure and the query verification process of the VMR-tree. A set of experiments are made to compare the construction time, the storage size and the efficiency of kNN query authentication between MR-tree and VMR-tree. Though the VMR-tree need more construction time and storage space, it increases the efficiency of kNN query authentication obviously, and the increasing efficiency is more obvious with the increase of k value. As future work, we plan to study the methods for other query authentications and study the update of VMR-tree structure.

Acknowledgements

I would like to thank my partners for giving me valuable advices and encouragements. Thanks for all the support and guidance from my mentor Pro. Li, and finally, thanks to all the reviewers for the valuable comments.

References

[1] M. Etemad and A. Küpçü, "Database Outsourcing with Hierarchical Authenticated Data Structures", Icisc, **(2013)**, pp. 381-399.
[2] S. Ma, B. Yang and K. Li, "A Privacy-Preserving Join on Outsourced Database[C]// Information Security, International Conference, ISC 2011, Xi'an, China, October 26-29, 2011. Proceedings. 2011:278-292.
[3] G. Zhi, F. Zeng and H. Liu, "Realization of the communication between devices of BACnet/Ip and real-time database", Review of Computer Engineering Studies, vol. 1, **(2014)**, pp. 17-22.
[4] Y. Yang, S. Papadopoulos and D. Papadias, "Authenticated indexing for outsourced spatial databases", Vldb Journal, vol. 18, no. 3, **(2009)**, pp. 631-648.
[5] L. Balasubramanian and M. Sugumaran, "A State-of-Art in R-Tree Variants for Spatial Indexing", International Journal of Computer Applications, vol. 42, **(2012)**, pp. 35-41.
[6] G. Proietti and C. Faloutsos, "I/O Complexity for Range Queries on Region Data Stored Using an R-tree", **(1999)**, pp. 628-635.
[7] H. Hacigumus, B. Iyer and S. Mehrotra, "Providing Database as a Service", Proceedings of the 18th International Conference on Data Engineering. IEEE Computer Society, **(2002)**, pp. 29 - 38.
[8] W. S. Ku, L. Hu and C. Shahabi, "A query integrity assurance scheme for accessing outsourced spatial databases", Geoinformatica, vol. 17, no. 1, **(2013)**, pp. 97-124.

Copyright ⓒ 2016 SERSC

[9] W. Cheng, H. H. Pang and K. L. Tan, "Authenticating Multi-dimensional Query Results in Data Publishing"m Dbsec, **(2006)**, pp. 60-73.
[10] L. Hu, W. S. Ku and S. Bakiras, "Verifying spatial queries using Voronoi neighbors", Sigspatial International Conference on Advances in Geographic Information Systems. ACM, **(2010)**, pp. 350-359.
[11] F. Li, M. Hadjieftheriou and G. Kollios, "Authenticated Index Structures for Outsourced Databases" Handbook of Database Security. Springer US, **(2008)**, pp. 115-136.
[12] B. N. Kriegel, H. Peter and S. Ralf, "The R*-tree: an efficient and robust access method for points and rectangles", ACM SIGMOD Record, vol. 19, no. 2, **(1990)**, pp. 322-331.
[13] M. Erwig, "The graph Voronoi diagram with applications", Networks, vol. 36, no. 3, **(2000)**, pp. 156-163.
[14] R. D. Juday, "Optimal realizable filters and the minimum Euclidean distance principle", Applied Optics, vol. 32, no. 26, **(1993)**, pp. 5100-11.
[15] A. Okabe, B. Boots and K. Sugihara, "Spatial Tessellations: Concepts and Applications of Voronoi Diagrams, Second Edition", **(2008)**.
[16] N. Roussopoulos, S. Kelley and F. Vincent, "Nearest Neighbor Queries", Nearest Neighbor Search. Springer US, **(2005)**, pp. 71-79.
[17] G. R. Hjaltason, "Distance browsing in spatial databases", ACM Transactions on Database Systems, vol. 24, no. 2, **(1999)**, pp. 265-318.
[18] http://a841-dotvweb01.nyc.gov/ParkingRegs/ViewController/LocationValidation.aspx.
[19] Z. Jing, L. Yongqin and L. Zhenyu, "Analysis of forest fire surveillance & Prewarning application system based on Power grid GIS", Review of Computer Engineering Studies, vol. 1, **(2014)**, pp. 23-28.

Authors

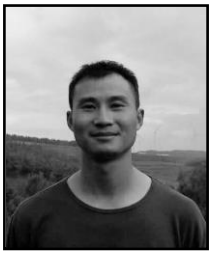
Xiaofu Wei, He received the M.S. degree in Computer application technology from Logistical Engineering University, China, in 2014, and studying for a Ph. D. degree in Science of Military Logistics. He is interested in Geographic Information System and database technology.

Shenglin Li, He received M.S. degree in Mathematics from Southwest University, China, in 1986, and Ph.D. degree in Logistical Engineering University of P.L.A China, in 2008. He is interested in information management engineering and computer science.

Zuofei Tan, He received M.S. degree in detection technology and automatic equipment from Logistical Engineering University, China, in 2014, and studying for a Ph. D. degree in Science of Military Logistics. He is interested in wireless sensor networks and computer science.

Kaiwen Luo, He received M.S. degree in detection technology and automatic equipment from Logistical Engineering University, China, in 2012, and studying for a Ph. D. degree in Science of Military Logistics. He is interested in Remote Monitoring & Fault Diagnosis and Computer Science.

 Copyright ⓒ 2016 SERSC



San Zhang, He received M.S. degree in Science of Military Logistics from Logistical Engineering University, China, in 2014, and is engaged in the study of Military Science in Logistical Engineering University. He is interested in Military Logistics and Computer Science.

Copyright ⓒ 2016 SERSC

Copyright ⓒ 2016 SERSC