IMPROVING EFFICIENCY IN USER RELATED SECURED QUERY PROCESSING ON METRIC DATA ASSETS

K.MohanaSundaram[1], Dr. R.Sasikumar[2]
[1]ME., Assistant Professor, CSE.,R.M.D Engineering College,
[2]Ph.D,Professor ,, CSE.,R.M.D Engineering College

Abstract
This paper considers a cloud computing setting in which similarity querying of metric data is outsourced to a service provider. The data is to be revealed only to trusted users, not to the service provider or anyone else. Users query the server for the most similar data objects to a query example. Outsourcing offers the data owner scalability and a low-initial investment. The need for privacy may be due to the data being sensitive (e.g., in medicine), valuable (e.g., in astronomy), or otherwise confidential. Given this setting, the paper presents techniques that transform the data prior to supplying it to the service provider for similarity queries on the transformed data. Our techniques provide interesting trade-offs between query cost and accuracy. They are then further extended to offer an intuitive privacy guarantee. Empirical studies with real data demonstrate that the techniques are capable of offering privacy while enabling efficient and accurate processing of similarity queries.

1. INTRODUCTION

Advances in digital measurement and engineering technologies enable the capture of massive amounts of data in fields such as astronomy, medicine, and seismology. The effort for data collection and processing, as well as its potential utility for research or business, creates value for the data owner. He wishes to store them and allow access by himself, colleagues, and other (trusted) scientists or customers. This can be supported by outsourced servers that offer low storage costs for large databases. For instance, outsourcing based on cloud computing is becoming increasingly attractive, as it promises pay-as-yougo, low storage costs as well as easy data access. However, care needs to be taken to safeguard data that is valuable or sensitive against unauthorized access. In this context, we call any item in a data collection an object, individuals with authorized access query users, and the entity offering the storage service the service provider. We illustrate the sensitivity issues with several scenarios. First, consider space programs such as the NASA Apollo program on the Earth's Moon1 or the ESA Mars Express2 that collect scientifically valuable and rare data. The NASA data is known to be private before it is released to the public. For example, time series data is collected from sensors to study the atmosphere's density. Such data is usually analyzed by the scientists involved in setting up the instruments, prior to being made available to the general community.

2. RELATED WORK

High Performance Metric Trees Minimizing Overlap between Nodes [] In this paper we present the Slim-tree, a dynamic tree for organizing metric datasets in pages of fixed size. The Slim-tree uses the "fat-factor" which provides a simple way to quantify the degree of overlap between the nodes in a metric tree. It is well-known that the degree of overlap directly affects the query performance of index structures. There are many suggestions to reduce overlap in multidimensional index structures, but the Slim-tree is the first metric structure explicitly designed to reduce the degree of overlap. Moreover, we present new algorithms for inserting objects and splitting nodes. The new insertion algorithm leads to a tree with high storage utilization and improved query performance, whereas the new split algorithm runs considerably faster than previous ones, generally without sacrificing search performance. Results obtained from experiments with real-world data sets show that the new algorithms of the Slim- tree consistently lead to performance improvements. After performing the Slim-down algorithm, we observed improvements up to a factor of 35% for range queries.Outsourcing Search Services on Private Spatial Data[] Social networking and content sharing service providers, e.g., Facebook and Google Maps, enable their users to upload and share a variety of user-generated content, including location data such as points of interest. Users wish to share location data through an (untrusted) service provider such that trusted friends can perform spatial queries on the data. We solve the problem by transforming the location data before uploading them.. The data owner selects transformation keys and shares them with the trusted friends. Without the keys, it is infeasible for an attacker to reconstruct the exact original data points from the transformed points. For example, given an image database, one may want to retrieve all images that are similar to a given query image. Distance based index structures are proposed for applications where the distance computations between objects of the data domain are expensive (such as high dimensional data), and the distance function used is metric. In this paper, we consider using distance-based index structures for similarity queries on large metric spaces. We elaborate on the approach of using reference points (vantage points) to partition the data space into spherical shell-like regions in a hierarchical manner. We introduce the multi-vantage point tree structure (mvp-tree) that uses more than one vantage points to partition the space into spherical cuts at each level. Empirical studies show that mvp-tree outperforms the vp-tree by 20% to 80% for varying query ranges and different distance distributions. Next, we generalize the idea of using multiple vantage points, and discuss the results of experiments we have done to see how varying the number of vantage points used in a node affects search performance, and how much performance gain is obtained by making use of pre-computed distances. The results show that, after all, it may be best to use a large number of vantage points in an internal node to end up with a single directory node, and keep as many of the pre-computed distances as possible to provide more efficient filtering during search operations

3. PROPOSED WORK

In Proposed System, an application scenario for WSN, the multi-user scenario, is defined. In the multi-user scenario, there are a large number of users who just focus on their interests but know little about the deployment of the network. They initiate their event based queries in the network in order to obtain their intended information of interests, for example, whether the events of interests have happened, and where they are in the network. Our techniques provide interesting trade-offs between query cost and accuracy.In this scenario, time series can be represented as vectors of values in chronological order (see Fig. 1a). At query time, a user specifies an example time series q and wishes to obtain those time series most similar to q; the system then retrieves the time series p in the database with the minimum distance to q.

System Architecture:

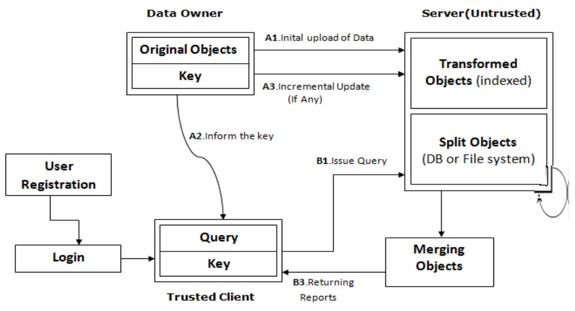


Figure1.System Architecture

It consists of three entities: a data owner, a trusted query user, and an untrusted server. On the one hand, the data owner wishes to upload his data to the server so that users are able to execute queries on those data. On the other hand, the data owner trusts only the users and nobody else (including the server). The data owner has a set P of (original) objects (e.g., actual time series, graphs, strings), and a key to be used for transformation. First, the data owner applies a transformation function (with a key) to convert P into a set P0 of transformed objects, and uploads the set P0 to the server (see step A1 in the figure). The server builds an index structure on the set P0 in order to facilitate efficient search.In addition, the data owner applies a standard Splitting method (e.g. AES) on the set of original objects; the resulting split objects (with their IDs) are uploaded to the server and stored in a relational table (or in the file system

Next, the data owner informs every user of the transformation key (see step A2). In the future, the data owner is allowed to perform incremental insertion/deletion of objects. (See step A3).At query time, a trusted user applies the transformation function (with a key) to the query and then sends the transformed query to the server (see step B1). Then, the server processes the query (see step B2), and reports the results back to the user (see step B3).Eventually, the user merges the retrieved results back into the actual results. Observe that these results contain only the IDs of the actual objects.

1. Cloud Architecture Construction Module

In this module, we create a cloud architecture module with the entities of Data Owner, trusted client, and cloud server. In this module, the data owner uploads their data in the cloud server, where the data should be stored securely. . The data is to be revealed only to trusted users, not to the service provider or anyone else. For this issue, we encrypt the data using encryption algorithm, so the data is not revealed to the service provider or intermediate persons or hacker.

2. Encrypted Hierarchical Index Search (EHI) Module

This section presents a client algorithm, called encrypted hierarchical index, for performing NN search on an encrypted hierarchical index stored at the server. This method offers perfect data privacy for the data owner, but it incurs multiple communication round trips during query processing.. An algorithm for communication between the client and the server needs to be developed in order to answer the NN query correctly.

3. Metric Preserving Transformation (MPT) Module

In this module, we develop a method, called metric preserving transformation, for evaluating the NN query. Unlike the EHI method, MPT incurs only 2 rounds of communication during the query phase. The basic idea behind MPT is to pick a small subset of the data set P as the set of anchor objects and then assign each object of P to its nearest anchor. For each object p, we compute its distance distðai; pÞ from its anchor ai and then apply an order-preserving encryption function OPE on the distance value.

4. Flexible Distance-based Hashing (FDH) Module

In this section, we propose a hashing-based technique, called flexible distance-based hashing, for processing the NN query. The main advantage of this technique is that the server always returns a constant-sized candidate set (in one communication round. Even though FDH is not guaranteed to return the exact result, the final result is very close to the actual NN in practice. In addition, our FDH method employs a novel technique for conceptually linking similar hash buckets, in order to maximize the utility of the transformed data for answering queries

5. Data Integrity verification Module

One of the important concerns that need to be addressed is to assure the customer of the integrity i.e. correctness of his data in the cloud. As the data is physically not accessible to the user the cloud should provide a way for the user to check if the integrity of his data is maintained or is compromised. This proof can be agreed upon by both the cloud and the customer and can be incorporated in the Service level agreement (SLA). It is important to note that our proof of data integrity protocol just checks the integrity of data i.e. if the data has been illegally modified or deleted

6. IMPLEMENTATION

In this section, we present the transformation methods, and the corresponding query processing techniques. We propose three transformation methods (EHI, MPT, FDH). They capture varioustrade-offs among data privacy and query cost and accuracy.We present extensions of MPT and FDH inorder to comply with the _-gap guarantee

Encrypted Hierarchical Index Search (EHI)

In the literature, algorithms have been developed for processing range queries on encrypted Bþ-tree and encrypted R-tree; however, no solutions were proposed for the NN query on those encrypted indexes.

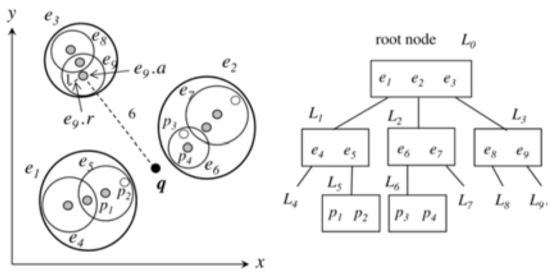


Figure2. Example of query processing in EHI.

Given a query object q and an index entry e, their minimum distance and maximum distance are defined as mindist(q,e)= max{0,dist(q,e.a)-e.r and max dist(q,e)=dist(q,e.a)=e.rthe anchor object of each entry is shown as a gray dot. For the index entry e9, we can computemindist (q,e9)=6-1=5 and maxdist(q,e9)=6+1=7

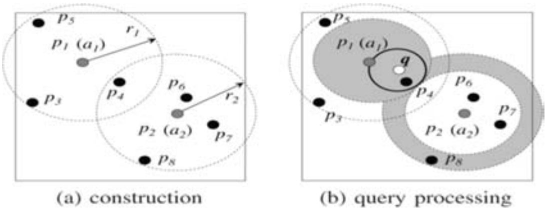
TABLE 1 Requested Nodes in EHI for Communication Rounds

Round	Requested Nodes		
	$\lambda = 1$ (best-first)	$\lambda = 2$	$\lambda = \infty$ (breadth-first)
1	root node	root node	root node
2	e_2	e_2, e_1	e_2, e_1
3	e_6	e_6, e_5	e_6, e_5, e_7, e_4
4	e_1	—	—
5	e_5	—	—

An algorithmfor communication between the client and the server needstobe developed in order to answer the NN query correctly.The total response time of the algorithm consists of theround trip latency and the data transfer time.

1. Metric Preserving Transformation (MPT)

In this section, we develop a method, called metric preserving transformation, for evaluating the NN query. Unlike the EHI method, MPT incurs only 2 rounds of communication during the query phase.The basic idea behind MPT is to pick a small subset of the data set P as the set of anchor objects [11] and then assign each object of P to its nearest anchor. For each object p, we compute its distance dist(ai,p); from its anchor ai and then apply an order preserving encryption function OPE on the distance value.These order-preserving encrypted distances will be stored in the server and utilized for processing NN queries.6A function OPE : IR -> IR is said to be order preserving if it guarantees that OPE(x)> OPE(x') for any values x; x0 thatsatisfy x > x0.As a remark, the transformed data set (at the server side)can be indexed by an R_-tree using a composite key on the anchor ID and the anchor distance.Incremental Data UpdateThe data owner inserts or deletes an object p, for thecorresponding bucket having the anchor object of p.If the data distribution in the data set changes gradually over time, the objects in a par(with a particular object ID We suggest that the data owner precomputes a bitmap (for each object) such that its length equals, titan may be far away from their anchor at some point.



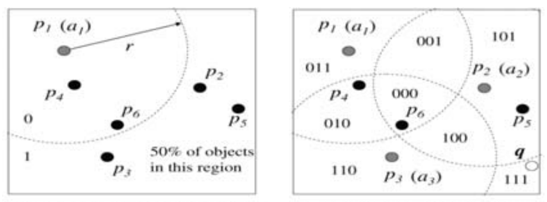
ID	Anchor ID	Anchor distance	Encrypted object
1	1	$\mathcal{OPE}(0.00) = 0.00$	$\mathcal{ENC}(p_1)$ = E249C5F3
2	2	$\mathcal{OPE}(0.00) = 0.00$	$\mathcal{ENC}(p_2)$ = 27E90563
3	1	$\mathcal{OPE}(0.35) = 1.00$	$\mathcal{ENC}(p_3)$ = A6231D25
4	1	$\mathcal{OPE}(0.20) = 0.60$	$\mathcal{ENC}(p_4)$ = 90B23F52
5	1	$\mathcal{OPE}(0.32) = 0.80$	$\mathcal{ENC}(p_5)$ = A796437C
6	2	$\mathcal{OPE}(0.10) = 0.20$	$\mathcal{ENC}(p_6)$ = 627D3935
7	2	$\mathcal{OPE}(0.18) = 0.40$	$\mathcal{ENC}(p_7)$ = 35B690D3
8	2	$\mathcal{OPE}(0.34) = 0.90$	$\mathcal{ENC}(p_8)$ = 2F076723

(c) transformed dataset stored at the server

Figure 3.Example of MPT.

2. Flexible Distance-Based Hashing (FDH)

In the literature, the Hilbert curve transformation has been employed for approximate NN search in 2D space However, its performance degrades rapidly for high dimensional space and it is inapplicable to the metric space. In contrast, our FDH method can be applied for any black box distance function dist().A typical value of A is in the range of tens to hundreds. This parameter provides a trade-off between the cost of transformation and the query accuracy, e.g., a higher value of A leads to better query accuracy, but also high-transformation cost. Let P be the original data set of objects. In the data transformation phase, we choose Arandom objects from the set P as anchor objects. BM(p)[i]={0 if dist(a,pi)<ri1 otherwise.The data owner efficiently computes the bitmap of an object and then requests the server to insert or delete that object.



(a) finding the value of r (b) data transformation

ID	Bitmap	Encrypted object
1	011	$\mathcal{ENC}(p_1)$ = A23692F0
2	101	$\mathcal{ENC}(p_2)$ = 967990B2
3	110	$\mathcal{ENC}(p_3)$ = 6723A623
4	010	$\mathcal{ENC}(p_4)$ = 835223B6
5	101	$\mathcal{ENC}(p_5)$ = D257623E
6	000	$\mathcal{ENC}(p_6)$ = 3023C790

(c) transformed dataset stored at the server

Figure4.Example of FDH.

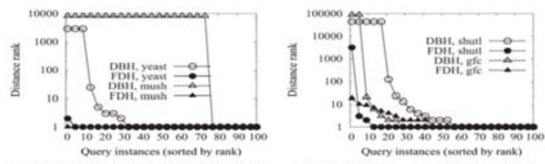
TABLE 2
Summary of Real Data Sets

Name	Attributes	Number of tuples	Distance
YEAST	17, numeric	2,882	L_1 norm
MUSH	22, categorical	8,124	Jaccard dist.
SHUTL	9, numeric	43,500	L_1 norm
GFC	16, numeric	86,984	L_1 norm

TABLE 3
Construction Time and Server Time onReal Data Sets, at Default Setting

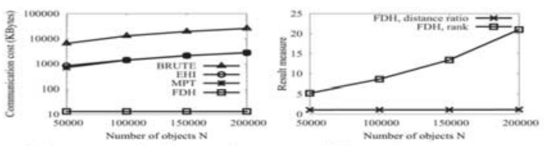
Dataset	Construction time (s)			Server CPU time (s)		
	EHI	MPT	FDH	EHI	MPT	FDH
YEAST	0.016	0.094	0.313	0.001	0.001	0.049
MUSH	0.234	0.531	1.344	0.006	0.002	0.083
SHUTL	2.438	1.187	4.672	0.010	0.006	0.097
GFC	12.141	3.063	10.078	0.007	0.005	0.141

Before evaluating the effects of the parameters, we give an overview of the general performance of the studied algorithms.shows the query communication cost of the solutions on all the data sets. The general tendency is the same for all data sets. The straightforward approaches BRUTE and ANONY have very high-communication cost when compared to the others.



(a) YEAST and MUSH data (b) SHUTL and GFC data

Figure 5. Rank of result NN for individual queries on real data sets.



(a) communication cost (b) result measure

Figure 6.Effect of data size N, on synthetic data SYN.

1. CONCLUSION

We propose similarity search techniques for sensitive metric data, e.g., bioinformatics data, that enable outsourcing of such search. Existing solutions either offer query efficiency at no privacy, or they offer complete data privacy while sacrificing query efficiency. We introduce approaches that shift search functionality to the server. The proposed FlexibleDistance-based Hashing methods finishes in just a single round of communication, Both MPT and FDH are extended to satisfy the _-gap privacy guarantee. We demonstrate their efficiency and privacy on synthetic andreal-world data.

REFERENCES

1. M.L. Yiu, G. Ghinita, C.S. Jensen, and P. Kalnis, "Outsourcing Search Services on Private Spatial Data," Proc. IEEE 25th Int'l Conf. Data Eng. (ICDE).
2. Veronica Gil-Costa and Mauricio Marin"Load Balancing Query Processing in Metric-Space Similarity Search."
3. R. Agrawal and R. Srikant, "Privacy-Preserving Data Mining," Proc. ACM SIGMOD Int'l Conf. Management of Data, pp. 439- 450, 2000.
4. M.L. Yiu, G. Ghinita, C.S. Jensen, and P. Kalnis, "Outsourcing Search Services on Private Spatial Data," Proc. IEEE 25th Int'l Conf. Data Eng. (ICDE).
5. R. Weber, H.-J. Schek, and S. Blott. A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces. In VLDB, 1998
6. R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. Order-Preserving Encryption for Numeric Data. In SIGMOD, 2004.
7. P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias. Preventing Location-Based Identity Inference in Anonymous Spatial Queries. IEEE TKDE, 19(12):1719–1733, 2007.