PAPER

A Trie-Based Authentication Scheme for Approximate String Queries

Yu WANG[†], **Liangyong YANG**[†], **Jilian ZHANG**[†,††a)], ***and*** **Xuelian DENG**[†††b)], ***Nonmembers***

SUMMARY Cloud computing has become the mainstream computing paradigm nowadays. More and more data owners (DO) choose to outsource their data to a cloud service provider (CSP), who is responsible for data management and query processing on behalf of DO, so as to cut down operational costs for the DO. However, in real-world applications, CSP may be untrusted, hence it is necessary to authenticate the query result returned from the CSP. In this paper, we consider the problem of approximate string query result authentication in the context of database outsourcing. Based on Merkle Hash Tree (MHT) and Trie, we propose an authenticated tree structure named MTrie for authenticating approximate string query results. We design efficient algorithms for query processing and query result authentication. To verify effectiveness of our method, we have conducted extensive experiments on real datasets and the results show that our proposed method can effectively authenticate approximate string query results.

key words: *approximate string query, edit distance, query result authentication, database outsourcing, cloud computing*

1. Introduction

With the advancement of computer technology, the amount of data is growing explosively, which causes difficulties in data storage and management for individuals and enterprises. In this context, cloud computing comes into being. As a new computing model, cloud computing not only provides users with powerful computing power, but also facilitates users to access the resource sharing pool on demand [1]. Therefore, more and more enterprises outsource their massive data to CSP, through which they can enjoy fast and efficient data management services while reducing enterprise operational costs.

In this paper, we consider the problem of approximate string query in the context of database outsourcing, where there are three parties, i.e., data owner (DO), cloud service provider (CSP), and the user. Specifically, DO outsources her string database to CSP, and CSP provides data management functionalities, such as storage, security, query processing and so on. When the user sends a query request to CSP, then CSP will process the query request in time and return the query results to the user.

Although cloud computing has many advantages, it is prone to some data security risks when the data of DO is given to CSP for storage and processing. First, CSP may be untrusted, e.g., deliberately tamper with query results, or only return partial query results to the user in order to save computing power. Therefore, the results returned by CSP must be authenticated, i.e., to verify whether the results are correct. Normally, query result authentication involves two aspects, correctness and completeness. Correctness refers to the fact that the returned result actually exists in the database and has not been tampered with; Completeness means that all the data in database that matches query conditions is correctly returned to the user.

Since textual data is ubiquitous in real applications, many information systems support approximate string query processing. For example, in a biological database, all the approximate protein sequences that meet user's query conditions are retrieved to identify biological clusters. In information retrieval, the desired answer (e.g., "hamburger") can still be returned when user search for a string (e.g., "humburger") that is misspelled. The approximate string query function improves the usability and user experience of the system. At the same time, approximate string query is a basic problem in many application fields, such as data integration, computational linguistics, and bioinformatics.

Nowadays, there are many research work on approximate string query processing, but few of them focus on the problem of authenticating approximate string query results in the context of database outsourcing in cloud computing [2]. Meanwhile, the existing verification data structures based on B-tree, R-tree and other structures cannot solve the problem of approximate string query result authentication. Therefore, in this paper, we focus on the problem of authenticating approximate string query results in the context of database outsourcing. The main contributions of this paper include:

- Based on MHT and Trie, we design an authentication tree structure named MTrie. Compared with other data structures, MTrie is more suitable for efficient approximate string query processing and result authentication.
- We design an efficient approximate string query authentication scheme, in which the verification object (VO) contains the nodes of MTrie tree and the corresponding special tokens.
- We prove that it is possible to identify deceptive behaviors of CSP through VO during authentication, for

Manuscript received September 8, 2023.
Manuscript revised November 30, 2023.
Manuscript publicized December 20, 2023.

[†]The authors are with College of Cyber Security, Jinan University, Guangzhou 510632 China.
[††]The author is with the Engineering Research Center of Trustworthy AI, Ministry of Education, Jinan University, Guangzhou China.
[†††]The author is with Department of Medical Informatics, Guangxi University of Chinese Medicine, Nanning 530200 China.
a) E-mail: zhangjilian@jnu.edu.cn (Corresponding author)
b) E-mail: 173213455@qq.com (Corresponding author)

DOI: 10.1587/transinf.2023EDP7185

example, tampering with query result record data, not returning all eligible query result records.
- We perform extensive experiments on two real datasets. The results show that compared with the existing schemes, our scheme improves the performance in terms of VO construction time, VO size and user verification time.

The main structure of this paper is as follows: in Sect. 2 we briefly review the existing work on query result authentication for outsourced databases. Then in Sect. 3 we introduce some related preliminaries. We then propose the authentication scheme for approximate string query results under the data outsourcing model in Sect. 4. The experiment results are presented in Sect. 5, and finally we conclude the paper in Sect. 6.

2. Related Work

In the past decade, much research work has been emerged on the field of database outsourcing. There are many existing solutions to solve related problems such as completeness guarantee of query results [3]. In this paper, we propose an authentication scheme based on Merkle Hash Tree (MHT), which is a widely used binary tree authenticated structure, as shown in Fig. 1. In general, MHT can be used to protect the integrity and correctness of a collection of data records.

Specifically, an MHT is constructed in a bottom-up fashion. First, all records in a database are sorted on a pre-specified attribute. Each leaf node of MHT corresponds to a data record d_i in the database and stores the hash value of the record $H_i = H(d_i)$, where $H(.)$ is a one-way collision-resistant hash function. Each internal node of the MHT stores the hash values of its left and right child nodes, i.e., concatenates the two hash values and then computers the hash value of the concatenation, e.g., $H_{12} = H(h_1|h_2)$, where symbol "|" stands for string concatenation operation. After the MHT is constructed, DO signs the hash value of the root node using the private key to prevent the data from being tampered with.

Based on MHT and other tree structures, researchers have proposed a series of query result authentication schemes for outsourced databases. Pang et al. [4] proposed a verification structure VBT (Verifiable B-tree) based on B-Tree, which can only guarantee correctness of the one-dimensional range query results but cannot guarantee completeness of the query result. Li et al. [5] proposed MBT (Merkle B-tree) by combining B-tree and MHT. MBT is a disk-based variant of MHT. Compared with MHT, MBT reduces the search time and the number of disk accesses.

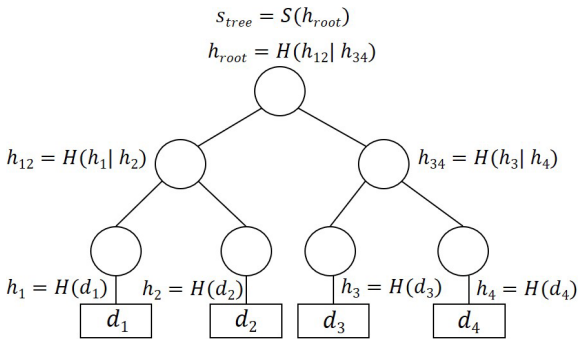


Fig. 1 The Merkle hash tree

Cheng et al. proposed multi-dimensional verification data structure VKD-tree (Verifiable KD-tree) and VR-tree (Verifiable R-tree) [6]. The idea of signature chain is applied to KD-tree and R-tree respectively, aiming to deal with the problem of query result verification for multi-dimensional data. Yang et al. [7] proposed a verification scheme that supports multidimensional top-k query. In the problem of query validation in outsourced spatial databases, Yang et al. [8] proposed MR-Tree (Merkle R-tree) and MR*-tree (Merkle R*-tree). The verification information is combined with the spatial index structure R-tree and R*-tree respectively, and only the hash value of the root node is needed to sign. This scheme is better than VR-tree in terms of the construction time of authenticated data structure (ADS), query result verification time, and smaller VO size.

In [9]–[11], Location-based spatial query verification schemes have been proposed, which effectively solve the problem of result verification for nearest neighbor, k-nearest neighbor and skyline query, respectively. In [12]–[14], the authors proposed a verification scheme to solve the problem of verifying the results of spatial multi-user queries. Multiple SQL query verification schemes are described in [3]. In [15], [16], they cannot only ensure the correctness and completeness of query results, but also support the verification of query results on dynamic data sets. In [17]–[19], the authors solve the problem of privacy protection of outsourced data, where DO uploads encrypted data to server and server can still correctly perform query processing and return query results even if the original data content is not available. Yang et al. designed an authenticated index structure based on MHT and q-gram inverted index, which can verify approximate string query results. However, the size of VO generated based on the structure is large, because the dictionary used in the structure is voluminous [22].

3. Preliminaries

3.1 Trie

Trie is a tree-like data structure for efficient string matching. It is often used in text search, string matching, word occurrence counting and so on. Given a Trie, each path from the root to a leaf represents a string in the dataset and every branch on the path corresponds to a different character in the string. In real applications, many strings have the same prefix, and the characteristic of a Trie is that strings with a same prefix have a common parent path. Therefore, by using Trie we can reduce the number of comparisons of the same prefix, so as to reduce the query overhead and improve search efficiency.

3.2 Edit Distance

In existing research work, various measures have been proposed to measure the similarity between two strings [20]. In this paper, we use edit distance to measure the similarity between two strings. Specifically, edit distance is defined as the minimum number of basic operations (including character insertion, deletion, and substitution) required to convert one string to another. Given two strings s_1 and s_2, the edit distance between them is defined as $ed(s_1, s_2)$.

3.3 Active Node

Given a Trie, a string s, and a Trie node n, if $ed(s,n) \leq d$ is satisfied, then node n is called the active node of string s. In real applications, to judge whether two strings are similar, it is not necessary to calculate the edit distance between two complete strings, instead we use the dynamic programming algorithm to measure their similarity. Specifically, during the calculation process, if the edit distance between two strings is found to be greater than a pre-specified threshold, we can then terminate the calculation early, thus saving the computational cost. Similarly, if n is not the active node of any prefix of s, then all strings below n are not similar to s with respect to a given edit distance threshold [21].

4. Our Proposed Scheme

In this section, we introduce the system model and our method for approximate string search result authentication.

4.1 System Model

The system model in this paper consists of three parties: data owner (DO), cloud service provider (CSP), and the user, as shown in Fig. 2. DO uses a private key to sign the hash of the root of the authenticated data structure (ADS) and sends the ADS and signature to CSP, who is responsible for user query processing on behalf of DO.

When processing user queries, CSP constructs VO of query result R with the help of ADS, and finally returns VO to the user. The user then verifies the correctness and completeness of the query result R based on DO's public key and VO.

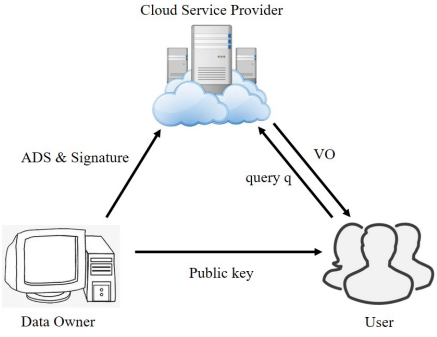


Fig. 2 The system model

4.2 Authenticated Data Structure MTrie

In this section, we propose a new authenticated data structure named MTrie that is based on MHT and Trie. Following the similar idea that MHT uses hierarchical hashing method to realize query result authentication, we embed corresponding digest (i.e., hash value) of each node in the Trie. Specifically, the digest of each leaf node is the hash value of a special symbol "⊥", while the digest of each internal node is based on the hash value of its child nodes and the character corresponding to each branch. For example, given a string set $S = \{in, inn, int, tea, ten, to\}$, Fig. 3 shows an MTrie with respect to S, where the hash values of node 3, 4, and 2 are $h_3 = h(\perp)$, $h_4 = h(\perp)$, and $h_2 = h(h(n)|h_3|h(t)|h_4)$, respectively. Here, $h(.)$ is a public hash function, such as SHA-1.

DO calculates the hash value of each node of MTrie, and finally uses the private key sk to sign the hash value of the root h_{root} of MTrie, in order to get the root signature S_{root} as shown below:

$$S_{root} = sig_{sk}(h_{root}) \tag{1}$$

where $sig(.)$ is a digital signature algorithm, such as RSA or ECDSA.

4.3 VO Construction

In order to process the approximate string query q, CSP uses Algorithm 1 to calculate the query results and construct the VO corresponding to the query results. Specifically, there are four types of information in VO:

(1) General active node, which puts the character data stored in a node into VO (Line 10 of Algorithm 1).
(2) Terminal node of the string and meet the query condition, use "*" to mark the character data of the node (Line 8).
(3) Hash values and character data of the nodes that need to be pruned (Line 14).

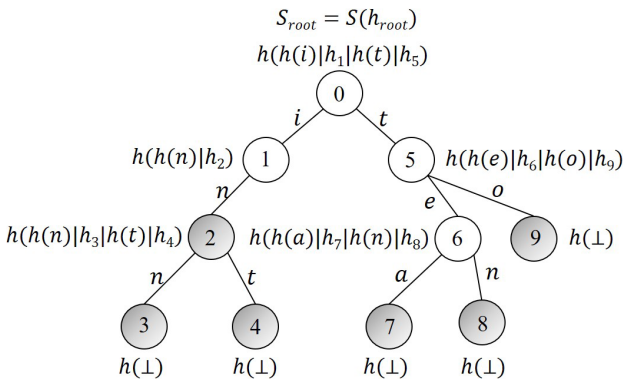


Fig. 3 The structure of an MTrie

Algorithm 1 VO Construction

Input: The root node of MTrie *root*, query q, threshold d
Output: Verification object *VO*

```
1:  if root is leaf then
2:      Append root.data to VO
3:  else
4:      Append "[" to VO
5:      for each e in root.children do
6:          if e is an active node then
7:              if e is terminal node & ed(e,q) ≤ d then
8:                  Append *e.data to VO
9:              else
10:                 Append e.data to VO
11:                 VOConstruction(e, q, d)
12:             end if
13:         else
14:             Append (e.data, e.hash) to VO
15:         end if
16:     end for
17:     Append "]" to VO
18: end if
19: return VO
```

Algorithm 2 Compute Root Hash

Input: Verification object *VO*
Output: h'_{root}

```
1:  str = ∅; res = ∅;
2:  While VO still has entries do
3:      Get next entry ev from VO
4:      if ev is "]" then
5:          return h(str)
6:      else if ev is "[" then
7:          res = rootHash(VO)
8:          Concatenate str with res
9:      else
10:         if ev is a pair of (ev.data, ev.hash)' then
11:             Concatenate str with h(ev.data)|ev.hash
12:         else if next entry ev is not "[" then
13:             Concatenate str with h(ev.data)|h(⊥)
14:         else
15:             Concatenate str with h(ev.data)
16:         end if
17:     end if
18: end while
```

(4) Special symbols " [" and "]" that embrace nodes belonging to a same subtree (Lines 4 and 17).

Let us take Fig. 3 as an example. Assuming that the query string q = "*inf*" and edit distance threshold $d = 1$, we use Algorithm 1 to obtain the final VO as follows:

$$VO = [i[*n[*n, *t]], t[(e, h_6), (o, h_9)]] \quad (2)$$

Finally, CSP returns the VO and the signature of root node S_{root} to the user. It is worth noting that the query result set {"in", "inn", "int"} that meet the edit distance threshold condition are already included in the VO.

4.4 Authentication Phase

In order to verify correctness of the query results, user employs Algorithm 2 to traverse the VO once, and calculates the hash value h'_{root} of the root node recursively in a bottom-up fashion. Then we use the public key pk disclosed by DO to verify the signature S_{root} returned by CSP, so as to check whether the result returned from CSP is the same as the h'_{root} reconstructed by the user.

$$h'_{root} = h(h(i)|h(h(n)|h(h(n)|h(\perp)| h(t)|h(\perp)))|h(t)|h(h(e)|h_6|h(o)|h_9)) \quad (3)$$

$$h'_{root} \stackrel{?}{=} sig_{pk}(S_{root}) \quad (4)$$

Figure 4 shows the process of calculating h'_{root} by Algorithm 2. Specifically, during traversing the *VO*, the user extracts the string with "*" and adds it to the result set. At the same time, the user verifies that the strings represented by the remaining leaf nodes that are not in the result set do not satisfy the edit distance threshold constraint.

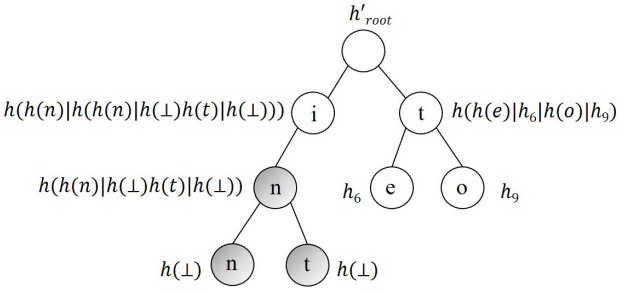


Fig. 4 The MTrie reconstructed by the user

To prove the correctness of our proposed scheme, we assume that a data record p in query result set R is forged or modified into a record p'. Since the hash function is one-way collision-resistant and the calculation of the hash value h_{root} of the root node of MTrie must use the original data record p. If the user calculates the root hash of MTrie by using the query result set R that contains record p', then the resulting root hash h'_{root} does not match $sig_{pk}(s_{root})$. Therefore, the method proposed in this paper can ensure correctness of the result set R.

Next, we prove the completeness of our proposed authentication method. We assume that the data record p is a string that meets the edit distance constraint. To enable the user to calculate the correct root hash h_{root}, there are two cases: (1) data record p exists in VO, and (2) data record p exists in the hash of the pruned node, that is, p is not returned to the user by CSP. For the first case, when traversing VO, the user extracts the data record p that meets the edit distance threshold constraint and puts it into the result set R, so that the correct root hash h_{root} can be calculated. On the other hand, for the second case the user will find that there

are strings that match the edit distance threshold constraint in the pruned node during verification, meaning that there is a potential violation of completeness of the query result. Therefore, all the similar strings in S that match the edit distance threshold constraint will be returned to the user.

5. Experiments

In this section, we conduct experiments on real datasets to verify effectiveness of our proposed authentication scheme. We measure the performance of our scheme in terms of VO construction time, VO size and user verification time. We compare our method named MTrie with the GS2-opt method in [22], which is the most relevant work to ours.

5.1 Experiment Setup

Datasets. We employ two real-world datasets: (1) DBLP Authors (https://dblp.uni-trier.de/xml/) downloaded from the DBLP publication dataset that contains author names; (2) LastName (www.census.gov/topics/population/genealogy/data/1990_census/1990_census_namefiles.html) is a dataset of frequently occurring surnames provided by the U.S. Census Bureau. Table 1 summarizes the detailed statistics of the two datasets, where N denotes the number of strings, *AvgLen* the average length of the strings, and $|\sum|$ the number of different characters.

Experiment environment. The experiments were conducted on a PC running 64-bit Ubuntu operating system with Intel(R) Core(TM) i9-9960X CPU@3.10 GHz. The hash function we adopted in our experiments is SHA1, and the generated hash value is 20 bytes long. The digital signature algorithm we used is the 128 bit RSA. We randomly generate 50 strings as query strings, and each of the reported result is the average of 10 trails.

5.2 Experiment Results

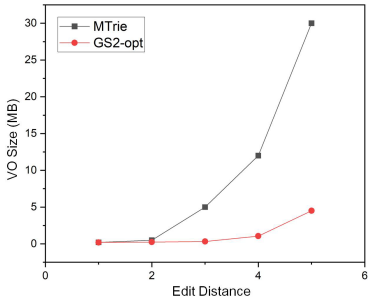
In the query verification scheme, the time to construct VO, the size of VO and the verification time of the user have a certain change relationship with the edit distance. The following is the comparison of the experimental results between this paper and that in [22].

We first investigate the impact of edit distance on VO size. The results are given in Fig. 5, which reveals the relationship between the size of VO and the edit distance d on the two datasets. When the edit distance is less than 4, the size of VO is small and it grows slowly. As the edit distance continues to increase, the number of strings that match the edit distance threshold increases sharply, so the size of VO also increases rapidly.

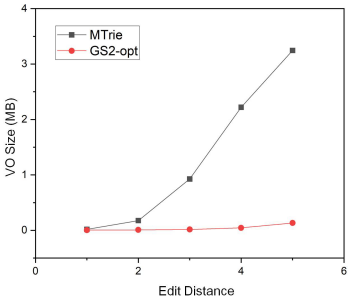
Table 1 Statistics of the two datasets used

Dataset	*N*	*AvgLen*	$\|\Sigma\|$
DBLP Authors	7,552,572	11.04	37
LastName	88,799	6.83	26

Next, we investigate the impact of edit distance on VO construction time, and the experiment results are presented in Fig. 6. From the figure we can see that the VO construction

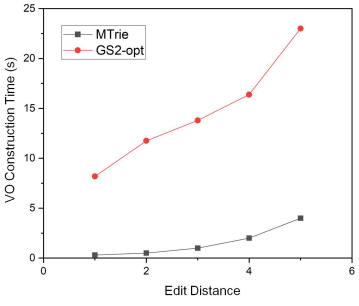


(a) DBLP dataset

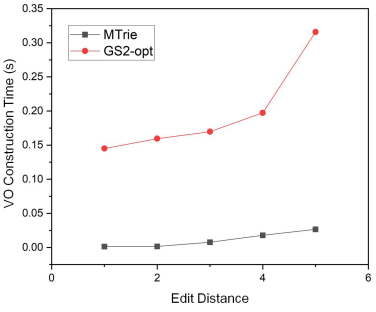


(b) LastName dataset

Fig. 5 Edit distance versus VO size

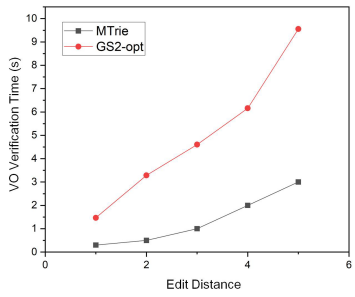


(a) DBLP dataset

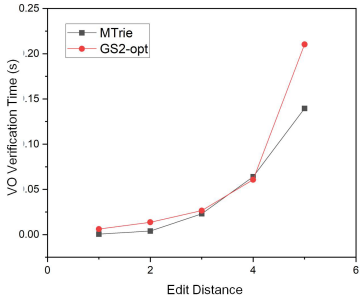


(b) LastName dataset

Fig. 6 Edit distance versus VO construction time



(a) DBLP dataset



(b) LastName dataset

Fig. 7 Edit distance versus VO verification time

time of MTrie is much less than that of [22]. When the edit distance is less than 4, the VO construction time is less than one second. With the increase of edit distance, however, the VO construction time shows a rapid increase trend.

Finally, we look at the impact of edit distance on VO verification time at the user side, and Fig. 7 gives the experiment results. The user verification time required by our proposed scheme is also less than the verification time reported in [22]. There are two parts in the time cost of VO verification at the user side, that is: (1) the time for verifying the correctness of query results. Specifically, the user recalculates the hash value h'_{root} of the root node of MTrie by using VO, employs the public key pk to decrypt the signature S_{root} to get $sig_{pk}(S_{root})$, and then determines whether h'_{root} is equal to $sig_{pk}(S_{root})$; (2) the time for verifying the completeness of query results, i.e., whether the edit distance between the strings composed of all termination nodes in VO (excluding those in the result set) and the query string is greater than the distance threshold d.

The above experiment results show that our MTrie proposed in this paper performs better than GS2-opt when edit distance is small. However, when a larger edit distance threshold is specified, the performance of the proposed scheme will deteriorate, that is, the size of VO and the user verification time will increase. It is worth noting that in real-world applications, using a larger edit distance threshold for approximate string queries often greatly affects the accuracy and usability of query results. Therefore, the authentication scheme proposed in this paper is more suitable and meaningful for most practical application scenarios with a smaller edit distance.

6. Conclusion

In this paper, we focused on the problem of authenticating the approximate string query results in the context of data outsourcing. First, we proposed an effective authenticated data structure called MTrie. Then, based on MTrie, we designed a VO construction algorithm and a query result authentication algorithm to ensure the correctness and completeness of the query results, respectively. Finally, we have conducted extensive experiments on real datasets to verify that our proposed scheme is superior to the existing state-of-the-art method for approximate string query result authentication in the context of data outsourcing.

Acknowledgments

The work of this paper is supported by NSFC (Grant No. 62020106013 and 61972177), Science Plan of Yunfu city (Grant No. 2022010212), and Teaching Quality and Reform Project (Grant No. JG2023057), Jinan University. Xuelian Deng is supported by Guangxi Higher Education Undergraduate Teaching Reform Project (Grant No. 2023JGB234) and Project of Guangxi University of Chinese Medicine (Grant No. 2022MS023 and 2022B061).

References

[1] P. Mell and T. Grance, "The nist definition of cloud computing," 2011.
[2] B. Dong and W. Wang, "Arm: Authenticated approximate record matching for outsourced databases," 2016 IEEE 17th International Conference on Information Reuse and Integration (IRI), IEEE, pp.591–600, 2016.
[3] B. Zhang, B. Dong, and W.H. Wang, "Integrity authentication for SQL query evaluation on outsourced databases: A survey," IEEE Trans. Knowl. Data Eng., vol.33, no.4, pp.1601–1618, 2019.
[4] H.H. Pang and K.-L. Tan, "Authenticating query results in edge computing," Proc. 20th International Conference on Data Engineering, IEEE, pp.560–571, 2004.
[5] F. Li, M. Hadjieleftheriou, G. Kollios, and L. Reyzin, "Dynamic authenticated index structures for outsourced databases," Proc. 2006 ACM SIGMOD international conference on Management of data, pp.121–132, 2006.
[6] W. Cheng, H.H. Pang, and K.-L. Tan, "Authenticating multi-dimensional query results in data publishing," Data and Applications Security XX: 20th Annual IFIP WG 11.3 Working Conference on Data and Applications Security, Sophia Antipolis, France, July 31-Aug. 2, 2006. Proceedings 20. Springer, pp.60–73, 2006.
[7] S. Yang, S. Tang, and X. Zhang, "Privacy-preserving k nearest neighbor query with authentication on road networks," Journal of Parallel and Distributed Computing, vol.134, pp.25–36, 2019.
[8] Y. Yang, S. Papadopoulos, D. Papadias, and G. Kollios, "Spatial outsourcing for location-based services," 2008 IEEE 24th International Conference on Data Engineering, IEEE, pp.1082–1091, 2008.
[9] S. Nutanong, R. Zhang, E. Tanin, and L. Kulik, "The v*-diagram: a query-dependent approach to moving KNN queries," Proc. VLDB Endowment, vol.1, no.1, pp.1095–1106, 2008.
[10] L. Hu, W.-S. Ku, S. Bakiras, and C. Shahabi, "Spatial query integrity with voronoi neighbors," IEEE Trans. Knowl. Data Eng., vol.25, no.4, pp.863–876, 2011.
[11] X. Zhu, J. Wu, W. Chang, G. Wang, and Q. Liu, "Authentication of

skyline query over road networks," Security, Privacy, and Anonymity in Computation, Communication, and Storage: 11th International Conference and Satellite Workshops, SpaCCS 2018, Melbourne, NSW, Australia, Dec. 11-13, 2018, Proceedings 11. Springer, pp.72–83, 2018.

[12] Y. Wang, S. Gao, J. Zhang, X. Nie, X. Duan, and J. Chen, "Authenticating multiple user-defined spatial queries," 2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC), vol.1, IEEE, pp.471–480, 2016.

[13] X. Duan, Y. Wang, J. Chen, and J. Zhang, "Authenticating preference-oriented multiple users spatial queries," 2017 IEEE 41st annual computer software and applications conference (COMPSAC), vol.1, IEEE, pp.602–607, 2017.

[14] Y. Wang, X. Duan, X. Yang, Y. Zhang, and X. Zhang, "Processing multiple-user location-based keyword queries," IEICE Trans. Inf. & Syst., vol.101, no.6, pp.1552–1561, 2018.

[15] H. Zhu, Q. Wei, X. Yang, R. Lu, and H. Li, "Efficient and privacy-preserving online fingerprint authentication scheme over outsourced data," IEEE Trans. Cloud Comput., vol.9, no.2, pp.576–586, 2018.

[16] M. Rady, T. Abdelkader, and R. Ismail, "Integrity and confidentiality in cloud outsourced data," Ain Shams Engineering Journal, vol.10, no.2, pp.275–285, 2019.

[17] W. Song, B. Wang, Q. Wang, Z. Peng, and W. Lou, "Tell me the truth: Practically public authentication for outsourced databases with multi-user modification," Information sciences, vol.387, pp.221–237, 2017.

[18] T. Xiang, X. Li, F. Chen, Y. Yang, and S. Zhang, "Achieving verifiable, dynamic and efficient auditing for outsourced database in cloud," Journal of Parallel and Distributed Computing, vol.112, pp.97–107, 2018.

[19] J. Wang, X. Chen, J. Li, J. Zhao, and J. Shen, "Towards achieving flexible and verifiable search for outsourced database in cloud computing," Future Generation Computer Systems, vol.67, pp.266–275, 2017.

[20] M. Yu, G. Li, D. Deng, and J. Feng, "String similarity search and join: a survey," Frontiers of Computer Science, vol.10, pp.399–417, 2016.

[21] J. Wang, J. Feng, and G. Li, "Trie-join: Efficient trie-based string similarity joins with edit-distance constraints," Proc. VLDB Endowment, vol.3, no.1-2, pp.1219–1230, 2010.

[22] L. Yang, H. Ye, X. Liu, Y. Mao, and J. Zhang, "Authenticating q-gram-based similarity search results for outsourced string databases," Mathematics, vol.11, no.9, 2023. [Online]. Available: https://www.mdpi.com/2227-7390/11/9/2128

Yu Wang received the B.S. degree from Jiangsu University of Technology, and M.S. degree from Jinan University, respectively. Her research interests include big data management, cloud computing, and information security.

Liangyong Yang received the B.S. degree from Dongguan University of Technology, and M.S. degree from Jinan University, Guangzhou China in 2023. His research interests include cloud computing and information security.

Jilian Zhang received the Ph.D. degree in Information Systems from Singapore Management University in 2014. He has published more than 50 refereed papers in refereed conferences and journals, including IEEE TDSC, IEEE TNNLS, ACM SIGMOD, VLDB, WWW, IJCAI, etc. Currently he is an associate professor of Jinan University, Guangzhou China.

Xuelian Deng received the B.S. and M.S. degrees in Computer Science from Central China Normal University and Guangxi Normal University, respectively. Her research interests include information systems, data management and machine learning. She has published more than 10 papers in refereed international journals. Currently, she is an associate professor of Guangxi University of Chinese Medicine, China.