This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2021.3101120, IEEE Transactions on Dependable and Secure Computing

Towards Practical and Privacy-Preserving Multi-dimensional Range Query over Cloud

Yandong Zheng, Rongxing Lu, *Fellow, IEEE,* Yunguo Guan, Jun Shao, and Hui Zhu, *Senior Member, IEEE*

Abstract—It is undeniable that Internet of Things (IoT) in big data era can provide us with huge volumes of multi-dimensional data, transforming our society into a much more intelligent one. In order to fit for the multi-dimensional data processing in big data era, multi-dimensional range queries, especially over cloud platform, have received considerable attention in recent years. However, as the cloud server is not fully trustable, designing multi-dimensional range queries over encrypted data becomes a research trend, and many solutions have been proposed in the literature. Nevertheless, most existing solutions suffer from the leakage of the single-dimensional privacy, and such leakage would severely put the data at risk. Although a few existing works have addressed the problem of single-dimensional privacy, they are impractical in some real scenarios due to the issues of inefficiency, inaccuracy, and two-cloud-server requirement. Aiming at solving these issues, in this paper, we propose a practical and privacy-preserving multi-dimensional range query (PRQ) scheme. Specifically, in our proposed PRQ scheme, we first index the multi-dimensional dataset with an R-tree and reduce R-tree based range queries to the problem of point intersection and range intersection. Then, by employing the lightweight matrix encryption technique, we design two novel algorithms for PRQ, i.e., multi-dimensional point intersection predicate encryption (PIPE) and multi-dimensional range intersection predicate encryption (RIPE), which can preserve the privacy of the proposed point intersection algorithm and range intersection algorithm, and further preserve the single-dimensional privacy of the proposed PRQ scheme. Detailed security analysis shows that our proposed PRQ scheme is indeed privacy-preserving. In addition, extensive simulations are conducted, and the results also demonstrate its efficiency.

Index Terms—Multi-dimensional range query, cloud computing, encrypted data, single-dimensional privacy, R-tree

1 Introduction

The advances of Internet of Things [1] and information communcation techniques [2] have recently promoted the surge of big data and the digitization of our lives as well. Motivated by the wealth of intelligence in the big data, data owners desire to mine the data and offer a series of query services to some potential users. Meanwhile, to enjoy the accessibility and scalability of the cloud computing, data owners are willing to outsource their local data and the corresponding query services to a powerful cloud. However, since the data may contain some sensitive information and the cloud server is not fully trusted, data owners demand to encrypt the data for protecting the data privacy [3], [4]. Although the data encryption techniques can successfully address the problem of the privacy, they also introduce a great obstacle into the process of query services execution. Thus, various privacy-preserving query schemes over encrypted data were proposed in the literature. Among them, the multi-dimensional range query over encrypted data [5]–[15] is one of the most popular query services.

Before introducing these schemes, we first give an example over a plaintext dataset to illustrate the multi-dimensional range query and its privacy issues. Suppose that a data owner has a census dataset with ten data records, as shown in TABLE 1. Each data record has three attributes, i.e., gender, age, and degree. For the gender attribute, we use 0 to denote "Female" and 1 to denote "Male". For the degree attribute, we use 1 to denote "Bachelor", 2 to denote "Master", and 3 to denote "PhD". Meanwhile, the data owner outsources the dataset to the cloud. Then, a query user can launch multi-dimensional range query requests to the cloud. For example, a query "$\mathcal{Q}_1 = [0, 0] \wedge [25, 27] \wedge [1, 3]$" aims to search the data records satisfying whose gender is 0 (i.e., "Female"), and age falls in $[25, 27]$ and degree falls in $[1, 3]$ (i.e., {"Bachelor", "Master", "PhD"}). Upon receiving this query, the cloud server searches on the dataset and returns the qualified record, i.e., $ID_2 = \{$"Female", 26, "Master"$\}$, to the query user. To achieve multi-dimensional range query over encrypted data, many schemes were proposed [5]–[15] but they suffer from either the single-dimensional privacy issue or practicality issues.

TABLE 1
An Illustrative Census Dataset

Record identity	Gender	Age	Degree
ID_1	0 (Female)	23	1 (Bachelor)
ID_2	0 (Female)	26	2 (Master)
ID_3	1 (Male)	29	3 (PhD)
ID_4	1 (Male)	27	1 (Bachelor)
ID_5	1 (Male)	24	1 (Bachelor)
ID_6	0 (Female)	29	3 (PhD)
ID_7	0 (Female)	23	1 (Bachelor)
ID_8	1 (Male)	22	2 (Master)
ID_9	1 (Male)	24	1 (Bachelor)
ID_{10}	1 (Male)	28	2 (Master)

- **Single-dimensional privacy issue:** The single-

- *Y. Zheng, R. Lu and Y. Guan are with the Faculty of Computer Science, University of New Brunswick, Fredericton, NB E3B 5A3, Canada (e-mail: yzheng8@unb.ca, rlu1@unb.ca, yguan4@unb.ca).*
- *J. Shao is with School of Computer and Information Engineering, Zhejiang Gongshang University, Hangzhou, 310018, China (e-mail: chn.junshao@gmail.com).*
- *H. Zhu is with the State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an, 710071, China (e-mail: zhuhui@xidian.edu.cn).*

dimensional privacy refers to the information on which records satisfy each single dimension of the query range, e.g., all records $\{ID_1, ID_2, \cdots, ID_{10}\}$ satisfy the query $\mathcal{Q}_1$ on the degree dimension. The leakage of single-dimensional privacy discloses more private information to the cloud server than that the cloud server is allowed to know, which may have a disastrous consequence on the privacy of query requests and dataset. On the one hand, it may leak the privacy of query requests. For example, if the cloud server knows that all records $\{ID_1, ID_2, \cdots, ID_{10}\}$ satisfy $\mathcal{Q}_1$ on the degree dimension, it can deduce that the query range of $\mathcal{Q}_1$ on the degree dimension is $[1, 3]$. On the other hand, it may leak the privacy of the dataset. For example, when a cloud server has processed three query requests, including

$$\begin{cases} \mathcal{Q}_1 = [0, 0] \wedge [25, 27] \wedge [1, 3] \\ \mathcal{Q}_2 = [0, 0] \wedge [23, 27] \wedge [2, 3] \\ \mathcal{Q}_3 = [0, 1] \wedge [28, 30] \wedge [3, 3], \end{cases}$$

from the single-dimensional privacy of $\mathcal{Q}_1$, $\mathcal{Q}_2$ and $\mathcal{Q}_3$ on the degree dimension, it can deduce that

(1) all records $\{ID_1, ID_2, \cdots, ID_{10}\}$ satisfy the query request $\mathcal{Q}_1$ on the degree dimension;
(2) $\{ID_2, ID_3, ID_6, ID_8, ID_{10}\}$ satisfy the query request $\mathcal{Q}_2$ on the degree dimension;
(3) $\{ID_3, ID_6\}$ satisfy the query request $\mathcal{Q}_3$ on the degree dimension.

From these information, the cloud server can easily deduce that $\{ID_3, ID_6\} \subseteq \{ID_2, ID_3, ID_6, ID_8, ID_{10}\} \subseteq \{ID_1, ID_2, \cdots, ID_{10}\}$. Since the domain of degree has three values, from the containment relationship and population ratio of degree, the cloud server can deduce that $\{ID_3, ID_6\}$ have PhD degree, $\{ID_2, ID_8, ID_{10}\}$ have Master degree, and $\{ID_1, ID_4, ID_5, ID_7, ID_9\}$ have Bachelor degree. In this case, the degree information is completely leaked. Similarly, the cloud server can easily break the privacy of other attributes based on the single-dimensional privacy when the corresponding domain is not very large, e.g., gender attribute. Therefore, the leakage of single-dimensional privacy may have a disastrous consequence on the privacy of query requests and dataset.

However, most existing multi-dimensional range query schemes [5], [6], [12]–[15] suffer from the leakage of single-dimensional privacy. Specifically, the order-preserving encryption (OPE) schemes [5], [6] leak the single-dimensional privacy because the OPE leaks the order of the data. Schemes [12]–[15] leak the single-dimensional privacy because they decompose the multi-dimensional range queries into multiple single-dimensional range queries.

• **Practicality issues:** Although some existing schemes can address the single-dimensional privacy, they suffer from practicality issues, including query privacy leakage, inefficiency, inaccuracy, and two-cloud-server requirement. Specifically, the bucketization based range query schemes [7], [8] can only return query results with false-positive records and impose most of range queries' computational cost on query users who are usually considered to be resource-constrained. Public key cryptography based range query schemes [9]–[11] are computationally expensive. Meanwhile, the schemes [9], [11] cannot preserve query privacy. The recently proposed scheme TRQED$^+$ [16] achieves single-dimensional privacy by deploying two non-colluding cloud servers. Compared with a single-server model, the two-serve model is less practical because it has two limitations, i.e., (i) the non-colluding assumption is a little strong in some scenarios; and (ii) there is additional communication cost required between the two servers. Therefore, it is still highly challenging to achieve practical multi-dimensional range queries with single-dimensional privacy.

Aiming at the above challenge, in this paper, we propose a single-dimensional privacy-preserving and practical multi-dimensional range query (PRQ) scheme under the single-server setting. In our proposed PRQ scheme, same as the work in [16], we first index the multi-dimensional dataset with an R-tree structure and reduce R-tree based range queries to the problem of multi-dimensional point intersection and multi-dimensional range intersection. The point intersection is to determine whether a multi-dimensional data point is in the query range or not. The range intersection is to determine whether two multi-dimensional ranges intersect or not. Second, we leverage a coding technique to design a multi-dimensional point intersection algorithm and a multi-dimensional range intersection algorithm. Then, we design a multi-dimensional point intersection predicate encryption (PIPE) scheme and a multi-dimensional range intersection predicate encryption (RIPE) scheme by applying the lightweight matrix encryption technique to preserve the privacy of the proposed point intersection algorithm and range intersection algorithm. The matrix encryption is lightweight because (i) it encrypts vectors and matrices into vectors and matrices; and (ii) the computational cost over the plaintext vectors and matrices is the same as that over the encrypted vectors and matrices. Based on PIPE and RIPE, we propose our PRQ scheme. Specifically, our contributions are three folds as follows.

• First, we leverage the coding technique to design a multi-dimensional point intersection algorithm and a multi-dimensional range intersection algorithm. Concretely, we first figure out why preserving the single-dimensional privacy for the point intersection and range intersection is challenging. We summarize it as one reason, i.e., the basic operations of the point intersection and range intersection are data comparison, and it is hard to incorporate multiple data comparisons into one data comparison. In our algorithms, we employ the coding technique to transform data comparison into the problem of equality test and incorporate multiple equality tests into one equality test.

Specifically, suppose that $\mathbf{x}$ is a multi-dimensional record and $\mathcal{Q}$ is a multi-dimensional query range. The intuition of the coding technique is to respectively represent $\mathbf{x}$ and $\mathcal{Q}$ to vectors $\{\mathbf{X}_i, \mathbf{X'}_i\}_{i=1}^d$ and matrices $\{\mathbf{Q}_i\}_{i=1}^d$ (as discussed in Subsection 4). Then, we can determine whether $\mathbf{x} \stackrel{?}{\in} \mathcal{Q}$ by testing $\sum_{i=1}^d t_i(\mathbf{X}_i\mathbf{Q}_i\mathbf{X'}_i^T - 1) \stackrel{?}{=} 0$, where $\{t_i\}_{i=1}^d$ are random real numbers. That is, the multiple comparisons in range queries are transformed to (i) perform some matrix multiplication operations over the coded vectors or matrices; and (ii) test whether the computed result is equal to 0 or not. In this case, our scheme only leaks whether $\mathbf{x} \stackrel{?}{\in} \mathcal{Q}$, so it can preserve the single-dimensional privacy.

Example 1. Let $\mathbf{x} = (16, 163)$ be a data record and $\mathcal{Q} = [20, 30] \cap [155, 165]$ is a query range. If we intend to

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2021.3101120, IEEE Transactions on Dependable and Secure Computing

determine whether $\mathbf{x} \stackrel{?}{\in} \mathcal{Q}$, we have to do comparisons

$$\begin{cases} 16 \stackrel{?}{\in} [20, 30] \Leftrightarrow 16 \stackrel{?}{\geq} 20 \text{ and } 16 \stackrel{?}{\leq} 30 \\ 163 \stackrel{?}{\in} [155, 165] \Leftrightarrow 163 \stackrel{?}{\geq} 155 \text{ and } 163 \stackrel{?}{\leq} 165. \end{cases}$$

If we respectively do the comparisons $16 \stackrel{?}{\geq} 20$, $16 \stackrel{?}{\leq} 30$, $163 \stackrel{?}{\geq} 155$, and $163 \stackrel{?}{\leq} 165$, it will leak the information of $16 \notin [20, 30]$ and $163 \in [155, 165]$, i.e., the single-dimensional privacy. For a better privacy, we expect that the multi-dimensional range queries only leak the information of $\mathbf{x} \stackrel{?}{\in} \mathcal{Q}$ and preserve the privacy of the information on which dimensions satisfy the query range and which dimensions do not satisfy the query range. To avoid the leakage of the single-dimensional privacy, we use the coding technique to respectively represent $\mathbf{x}$ and $\mathcal{Q}$ to vectors $\{\mathbf{X}_i, \mathbf{X'}_i\}_{i=1}^2$ and matrices $\{\mathbf{Q}_i\}_{i=1}^2$. Then, we can determine whether $\mathbf{x} \stackrel{?}{\in} \mathcal{Q}$ by testing $\sum_{i=1}^2 t_i(\mathbf{X}_i\mathbf{Q}_i\mathbf{X'}_i^T - 1) \stackrel{?}{=} 0$. In this example, our scheme only leaks that $(16, 163) \notin [20, 30] \cap [155, 165]$ and has no idea on which dimensions do not satisfy the query range.

• Second, we design our PIPE scheme and RIPE scheme by applying the lightweight matrix encryption to the above point intersection algorithm and range intersection algorithm. Both PIPE scheme and RIPE scheme can preserve single-dimensional privacy. It is worth noting that besides multi-dimensional range queries, PIPE scheme and RIPE scheme can also be applied to other scenarios with the single-dimensional privacy issue, e.g., conjunctive queries. Meanwhile, based on the PIPE and RIPE schemes, we propose our PRQ scheme.

Since the key idea of our scheme is to transform multiple data comparisons into one equality test, our scheme is also applicable to other multi-dimensional data structures that require multiple data comparisons with single-dimensional privacy issue, e.g., k-d-tree, the variants of R-tree, and BSP tree. It is worth noting that the searching process of these data structures usually involves both single data comparison and multiple data comparisons. The former can be directly performed over the corresponding values. The latter can be achieved by our proposed scheme to protect the single-dimensional privacy.

• Third, we formally prove that PIPE scheme and RIPE scheme are selectively secure and show that our PRQ scheme is privacy-preserving. In addition, we conduct experiments to evaluate the performance of our PRQ scheme, and the results show that our PRQ scheme is more efficient than existing solutions. To the best of our knowledge, our PRQ scheme is the most practical multi-dimensional range query scheme with single-dimensional privacy.

The remainder of this paper is organized as follows. In Section 2, we introduce our system model, security model, and design goal. Then, we describe some preliminaries in Section 3. In Section 4, we present two intersection algorithms. In Section 5, we present the PIPE scheme and RIPE scheme. In Section 6, we propose our PRQ scheme, followed by security analysis and performance evaluation in Section 7 and Section 8, respectively. In Section 9, we present some related work. Finally, we draw our conclusion in Section 10.

2 Models and Design Goal

In this section, we formalize our system model, security model, and identify our design goal.

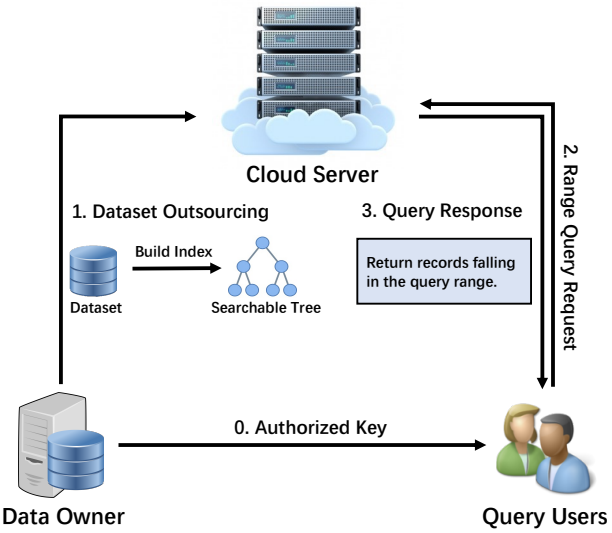


Fig. 1. System model under consideration

2.1 System Model

In our system model, we consider a typical cloud-assisted range query model, which involves three types of entities, i.e., a data owner, a cloud server, and multiple query users $\mathcal{U} = \{U_1, U_2, \cdots\}$ as shown in Fig. 1.

• Data Owner: The data owner has a dataset containing a collection of multi-dimensional data records. To offer a range query service to query users, the data owner builds a tree-based searchable index for the dataset. Then, the data owner encrypts both the dataset and index, and outsources them to the cloud server together.

• Cloud Server: The cloud server has powerful computing capability and abundant storage space. It is a bond between the data owner and query users, as it not only stores the encrypted dataset for the data owner but also offers the range query service to query users. Specifically, on receiving a range query request, the cloud server will search on the encrypted dataset and find out the data records satisfying the query request. Finally, the cloud server will return these records to the query user as the query result.

• Query Users $\mathcal{U} = \{U_1, U_2, \cdots\}$: There are multiple query users $\mathcal{U} = \{U_1, U_2, \cdots\}$ in the system. When these query users register in the system, the data owner will authorize them with an authorized key as shown in Fig. 1. After the authorization, the query users can enjoy the range query service from the cloud.

2.2 Security Model

In our security model, the data owner is assumed to be *trusted* because it provides the dataset and initializes the entire system. For the query users, they are assumed to be *honest*. That is, they will follow the scheme to launch range query requests. For the cloud server, it is assumed to be *honest-but-curious*. It will honestly follow the scheme to store

the outsourced data received from the data owner and offer the range query service to query users. However, it may be curious about some private information, including

(1) Single-dimensional privacy: the query result for each dimension of the query range;
(2) Query privacy: the plaintext of query requests and the corresponding query results;
(3) Dataset privacy: the plaintext of records in the dataset.

Besides, we assume that there is no collusion between the cloud server and query users. The non-collusive assumption is reasonable because the penalty of collusion is high, including being prosecuted. Note that there may be other active attacks in the system, such as DoS attack and data pollution attack. Since this work focuses on privacy preservation, those attacks are beyond the scope of this paper and will be discussed in our future work.

2.3 Design Goal

In this work, our goal is to design an efficient and privacy-preserving multi-dimensional range query scheme, and the following objectives should be satisfied.

• *Privacy preservation:* In our proposed scheme, the basic requirement is privacy preservation, i.e., the content of the outsourced dataset, query requests, as well as query results should be kept secret from the cloud server. Meanwhile, the single-dimensional privacy of the query results should be preserved.

• *Efficiency:* For achieving the above privacy preservation requirement, the range queries in our scheme will inevitably incur an additional computational cost. Thus, we also aim to minimize the computational cost of range queries and improve query efficiency as much as possible.

3 Preliminaries

In this section, we first review the R-tree structure and the R-tree based range query algorithm [17], which serve as the building blocks of our scheme. After that, we introduce a useful data comparison algorithm [18].

3.1 R-tree

R-tree (described in [17]) is a tree data structure and can be used for organizing multi-dimensional data records. The main idea of R-tree construction is to recursively cluster nearby data records and use a minimum bounding rectangle (MBR) to represent them in the higher level of the tree. In the R-tree, there are two types of nodes, i.e., leaf nodes and internal nodes. Each leaf node contains a multi-dimensional data record, and each internal node has an MBR and a pointer that points to its child nodes.

R-tree can be used as an efficient index for range queries over multi-dimensional data [17], and it is rated as the most prominent spatial index [19]. Suppose that $\mathcal{X}$ is a multi-dimensional dataset. We can represent $\mathcal{X}$ to an R-tree T. Without loss of generality, we assume that each leaf node of T corresponds to a data point $\mathbf{x}$ and each internal node is in the form of $(\mathcal{B}, \mathtt{p} = \{p_1, p_2, \cdots\})$, where $\mathcal{B}$ is an MBR and p is a set of pointers that point to its child nodes. Given a range query $\mathcal{Q}$, the searcher can efficiently search on T to find out the data records falling in $\mathcal{Q}$, i.e., $\{\mathbf{x}|\mathbf{x} \in \mathcal{Q}\}$. As shown in Algorithm 1, the search process contains two phases, i.e., filtration and verification as follows.

Algorithm 1 RangeQuery(Tree T, Query range $\mathcal{Q}$)

```
   // Filtration
1: C = ∅; // Initialize the candidate result
2: C = Filtration(T.root, Q);
   // Verification
3: R = ∅; // Initialize the final query result
4: for each x ∈ C do
5:     if x ∈ Q then
6:         R = R ∪ {x};
7: return R;
```

Algorithm 2 Filtration(Node $node$, Query range $\mathcal{Q}$)

```
1: if node is a leaf node then
2:     Add node's data record into C;
3: else
4:     if B ∩ Q ≠ ∅ then
5:         for each p_i ∈ p do
6:             Filtration(node.p_i, Q)
7: return C;
```

• *Filtration phase*: In the filtration phase, the searcher recursively searches on T to find a candidate set $\mathcal{C}$, which contains all data records that possibly fall in $\mathcal{Q}$ as shown in Algorithm 2. Specifically, the searcher starts to search T from the root node. When the searched node is an internal node $(\mathcal{B}, \mathtt{p} = \{p_1, p_2, \cdots\})$, the searcher checks whether $\mathcal{B}$ intersects with $\mathcal{Q}$. If yes, the searcher needs to search each $node.p_i$ for $p_i \in \mathtt{p}$. When the searched node is a leaf node with a data record $\mathbf{x}$, the searcher directly adds $\mathbf{x}$ into the candidate set $\mathcal{C}$.

• *Verification phase*: In the verification phase, the searcher verifies whether each candidate record $\mathbf{x} \in \mathcal{C}$ is in $\mathcal{Q}$ or not. If yes, the searcher adds $\mathbf{x}$ into the query result $\mathcal{R}$.

In summary, the range query algorithm involves two operations:

(1) Point intersection: determine whether $\mathbf{x} \stackrel{?}{\in} \mathcal{Q}$.
(2) Range intersection: determine whether $\mathcal{B} \cap \mathcal{Q} \stackrel{?}{=} \emptyset$.

3.2 Data Comparison Algorithm

In this subsection, we review a data comparison algorithm that was proposed by Boneh et al in [18]. Its idea is to use the coding technique to transform the comparison of two values into an equality test. Let x and q be two integers that need to be compared, and they are in the range of $[0, N^2-1]$. Then, we can compare them as follows.

Step 1: Code all values in $[0, N^2 - 1]$ to be an $N \times N$ matrix as shown in Fig. 2. In this matrix, each value has a row coordinate and a column coordinate. For example, the value $iN + j$ has the row coordinate $(i + 1)$ and column coordinate $(j + 1)$. Based on them, x and q can be represented to two-dimensional coordinates (i_x, j_x) and (i_q, j_q), where i_x, i_q are the row coordinate of x and q, j_x, j_q are the column coordinate of x and q. That is,

$$\begin{cases} i_x = \lfloor \frac{x}{N} \rfloor + 1;\ j_x = x \bmod N + 1; \\ i_q = \lfloor \frac{q}{N} \rfloor + 1;\ j_q = q \bmod N + 1. \end{cases} \quad (1)$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2021.3101120, IEEE Transactions on Dependable and Secure Computing

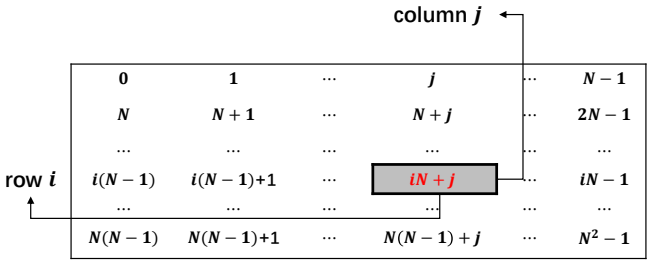


Fig. 2. Representation of the range $[0, N^2-1]$

Step 2: Based on (i_x, j_x), construct three N-dimensional vectors for x as

$$\widetilde{\mathbf{x}} = \begin{bmatrix} \mathbf{O}^{i_x} \\ \mathbf{1}^{N-i_x} \end{bmatrix}, \bar{\mathbf{x}} = \mathbf{e}_{i_x}, \hat{\mathbf{x}} = \begin{bmatrix} \mathbf{O}^{j_x-1} \\ \mathbf{1}^{N-j_x+1} \end{bmatrix}, \tag{2}$$

where (i) $\mathbf{O}^{i_x}$ denotes an i_x-dimensional zero vector; (ii) $\mathbf{1}^{N-i_x}$ denotes an $(N-i_x)$-dimensional vector and all elements are 1; and (iii) $\mathbf{e}_{i_x}$ denotes an N-dimensional identity vector and the i_x-th element is 1.

Observation 1. We can observe that

$$\begin{cases} x \le q \Leftrightarrow \widetilde{\mathbf{x}}[i_q] + \bar{\mathbf{x}}[i_q] * \hat{\mathbf{x}}[j_q] = 1 \\ x > q \Leftrightarrow \widetilde{\mathbf{x}}[i_q] + \bar{\mathbf{x}}[i_q] * \hat{\mathbf{x}}[j_q] = 0. \end{cases} \tag{3}$$

Furthermore, let $\mathbf{X} = [\widetilde{\mathbf{x}}^T\ \bar{\mathbf{x}}^T]$ and $\mathbf{X}' = [1\ \hat{\mathbf{x}}^T]$. Let $\mathbf{Q}$ be a $2N \times (N+1)$ matrix satisfying

$$\mathbf{Q}[i_q, 1] = \mathbf{Q}[N + i_q, j_q + 1] = 1, \tag{4}$$

and other elements in $\mathbf{Q}$ are 0. We have

$$\widetilde{\mathbf{x}}[i_q] + \bar{\mathbf{x}}[i_q] * \hat{\mathbf{x}}[j_q] = \begin{bmatrix} \widetilde{\mathbf{x}}^T & \bar{\mathbf{x}}^T \end{bmatrix} \mathbf{Q} \begin{bmatrix} 1 \\ \hat{\mathbf{x}} \end{bmatrix} = \mathbf{X}\mathbf{Q}\mathbf{X}'^T.$$

Then, we can deduce that

$$\begin{cases} x \le q \Leftrightarrow \widetilde{\mathbf{x}}[i_q] + \bar{\mathbf{x}}[i_q] * \hat{\mathbf{x}}[j_q] = 1 \Leftrightarrow \mathbf{X}\mathbf{Q}\mathbf{X}'^T = 1 \\ x > q \Leftrightarrow \widetilde{\mathbf{x}}[i_q] + \bar{\mathbf{x}}[i_q] * \hat{\mathbf{x}}[j_q] = 0 \Leftrightarrow \mathbf{X}\mathbf{Q}\mathbf{X}'^T = 0. \end{cases}$$

From Observation 1, we can see that the data comparison between x and q can be transformed to the equality test of $\mathbf{X}\mathbf{Q}\mathbf{X}'^T = 1$ or $\mathbf{X}\mathbf{Q}\mathbf{X}'^T = 0$.

4 Intersection Algorithms

Since the basic operations of the R-tree based range query algorithm are multi-dimensional point intersection and multi-dimensional range intersection, in this section, we propose a multi-dimensional point intersection algorithm and a multi-dimensional range intersection algorithm by using the data comparison algorithm in Subsection 3.2.

4.1 Point Intersection Algorithm

In this subsection, we introduce a multi-dimensional point intersection algorithm. For a clear description, we first introduce a one-dimensional point intersection algorithm.

One-dimensional point intersection algorithm: The idea of one-dimensional point intersection algorithm is to transform the problem of the point intersection into the problem of equality test by using the coding technique. Suppose that x is a data and $Q = (q_l, q_u]$ is a one-dimensional range, where $x, q_l, q_u \in [0, N^2-1]$. Although our scheme only considers the query ranges in the form of $Q = (q_l, q_u]$, it can be adapted to process other forms of query ranges like $[q_l, q_u]$. Specifically, since the query ranges in our scheme are integers, the query range $[q_l, q_u]$ is equivalent to the query range $(q_l - 1, q_u]$. With x and Q, the one-dimensional point intersection algorithm can determine whether $x \in Q$ or not by the following steps.

Step 1: According to the data comparison algorithm in Subsection 3.2, we represent x to three vectors $\{\widetilde{\mathbf{x}}, \bar{\mathbf{x}}, \hat{\mathbf{x}}\}$ and further use them to construct $\mathbf{X} = [\widetilde{\mathbf{x}}^T\ \bar{\mathbf{x}}^T]$ and $\mathbf{X}' = [1\ \hat{\mathbf{x}}^T]$.

Step 2: We respectively represent q_l and q_u in Q to two coordinates (i_l, j_l) and (i_u, j_u). We have

$$\begin{aligned} x \in (q_l, q_u] &\Leftrightarrow x > q_l \text{ and } x \le q_u \\ &\Leftrightarrow \begin{cases} \widetilde{\mathbf{x}}[i_l] + \bar{\mathbf{x}}[i_l] * \hat{\mathbf{x}}[j_l] = 0 \\ \widetilde{\mathbf{x}}[i_u] + \bar{\mathbf{x}}[i_u] * \hat{\mathbf{x}}[j_u] = 1 \end{cases} \\ &\Leftrightarrow \widetilde{\mathbf{x}}[i_l] + \bar{\mathbf{x}}[i_l] * \hat{\mathbf{x}}[j_l] + \widetilde{\mathbf{x}}[i_u] + \bar{\mathbf{x}}[i_u] * \hat{\mathbf{x}}[j_u] = 1. \end{aligned} \tag{5}$$

Similarly, we have

$$\begin{aligned} & x \notin (q_l, q_u] \\ \Leftrightarrow & \{x \le q_l \text{ and } x < q_u\} \text{ or } \{x > q_u \text{ and } x > q_l\} \\ \Leftrightarrow & \begin{cases} \widetilde{\mathbf{x}}[i_l] + \bar{\mathbf{x}}[i_l] * \hat{\mathbf{x}}[j_l] = 1 \\ \widetilde{\mathbf{x}}[i_u] + \bar{\mathbf{x}}[i_u] * \hat{\mathbf{x}}[j_u] = 1 \end{cases} \text{ or } \begin{cases} \widetilde{\mathbf{x}}[i_l] + \bar{\mathbf{x}}[i_l] * \hat{\mathbf{x}}[j_l] = 0 \\ \widetilde{\mathbf{x}}[i_u] + \bar{\mathbf{x}}[i_u] * \hat{\mathbf{x}}[j_u] = 0 \end{cases} \\ \Leftrightarrow & \begin{cases} \widetilde{\mathbf{x}}[i_l] + \bar{\mathbf{x}}[i_l] * \hat{\mathbf{x}}[j_l] + \widetilde{\mathbf{x}}[i_u] + \bar{\mathbf{x}}[i_u] * \hat{\mathbf{x}}[j_u] = 2 \\ \text{or} \\ \widetilde{\mathbf{x}}[i_l] + \bar{\mathbf{x}}[i_l] * \hat{\mathbf{x}}[j_l] + \widetilde{\mathbf{x}}[i_u] + \bar{\mathbf{x}}[i_u] * \hat{\mathbf{x}}[j_u] = 0. \end{cases} \end{aligned} \tag{6}$$

Based on (i_l, j_l) and (i_u, j_u), we construct a $2N \times (N+1)$ matrix $\mathbf{Q}$ such that

$$\mathbf{Q}[i_l, 1] = \mathbf{Q}[N + i_l, j_l + 1] = \mathbf{Q}[i_u, 1] = \mathbf{Q}[N + i_u, j_u + 1] = 1, \tag{7}$$

and other elements in $\mathbf{Q}$ are 0. Especially, if $i_l = i_u$, we set $\mathbf{Q}[i_l, 1] = \mathbf{Q}[i_u, 1] = 2$. Similarly, if $i_l = i_u$ and $j_l = j_u$, we set $\mathbf{Q}[N + i_l, j_l + 1] = \mathbf{Q}[N + i_u, j_u + 1] = 2$. Then, Eq. (5) is equivalent to $\mathbf{X}\mathbf{Q}\mathbf{X}'^T = 1$, and Eq. (6) is equivalent to $\mathbf{X}\mathbf{Q}\mathbf{X}'^T = 0$ or $\mathbf{X}\mathbf{Q}\mathbf{X}'^T = 2$. We can further deduce that

$$\begin{cases} x \in (q_l, q_u] \Leftrightarrow \mathbf{X}\mathbf{Q}\mathbf{X}'^T = 1 \\ x \notin (q_l, q_u] \Leftrightarrow \mathbf{X}\mathbf{Q}\mathbf{X}'^T = 0 \text{ or } \mathbf{X}\mathbf{Q}\mathbf{X}'^T = 2. \end{cases} \tag{8}$$

Thus, one-dimensional point intersection "$x \stackrel{?}{\in} Q$" is transformed into the equality test, i.e., "$\mathbf{X}\mathbf{Q}\mathbf{X}'^T \stackrel{?}{=} 1$". Next, we extend one-dimensional point intersection algorithm to multi-dimensional point intersection algorithm.

Point intersection for multi-dimensional data: The key idea of multi-dimensional point intersection algorithm is to

(i) decompose multi-dimensional point intersection into multiple one-dimensional point intersection;
(ii) transform the problem of one-dimensional point intersection determination into an equality test;
(iii) incorporate all one-dimensional equality tests into one equality test.

Suppose that $\mathbf{x} = (x_1, x_2, \cdots, x_d)$ is a d-dimensional data point and $\mathcal{Q} = (Q_1, Q_2, \cdots, Q_d)$ is a d-dimensional range, where $Q_i = (q_{l,i}, q_{u,i}]$ and $x_i, q_{l,i}, q_{u,i} \in [0, N^2-1]$ for $i = 1, 2, \cdots, d$. Then, the multi-dimensional point intersection algorithm determines whether $\mathbf{x} \in \mathcal{Q}$ as follows.

Step 1: According to the point intersection algorithm for one-dimensional data, we represent each $x_i \in \mathbf{x}$ to three

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2021.3101120, IEEE Transactions on Dependable and Secure Computing

vectors as $\{\widetilde{\mathbf{x}}_i, \bar{\mathbf{x}}_i, \hat{\mathbf{x}}_i\}$ and further use them to construct $\mathbf{X}_i = [\widetilde{\mathbf{x}}_i^T \ \bar{\mathbf{x}}_i^T]$ and $\mathbf{X}'_i = [1 \ \hat{\mathbf{x}}_i^T]$ for $i = 1, 2, \cdots, d$.

Step 2: We respectively represent each $Q_i = (q_{l,i}, q_{u,i}]$ to a $2N \times (N+1)$ matrix $\mathbf{Q}_i$ as Eq. (7) for $i = 1, 2, \cdots, d$. Based on Eq. (8), we have

$$\mathbf{x} \in \mathcal{Q} \Leftrightarrow \begin{cases} x_1 \in Q_1 \Leftrightarrow \mathbf{X}_1\mathbf{Q}_1\mathbf{X}'^T_1 = 1 \\ x_2 \in Q_2 \Leftrightarrow \mathbf{X}_2\mathbf{Q}_2\mathbf{X}'^T_2 = 1 \\ \cdots\cdots\cdots \\ x_d \in Q_d \Leftrightarrow \mathbf{X}_d\mathbf{Q}_d\mathbf{X}'^T_d = 1. \end{cases}$$

In this case, we have transformed the problem of multi-dimensional point intersection into the above multiple equality tests. In the following, we consider how to incorporate multiple equality tests into one equality test. If each equality $\mathbf{X}_i\mathbf{Q}_i\mathbf{X}'^T_i = 1$ holds for $i = 1, 2, \cdots, d$, for any d random non-zero real numbers $\{t_1, t_2, \cdots, t_d\}$, we have $\sum_{i=1}^{d} t_i(\mathbf{X}_i\mathbf{Q}_i\mathbf{X}'^T_i - 1) = 0$.

Note that the probability that $\sum_{i=1}^{d} t_i(\mathbf{X}_i\mathbf{Q}_i\mathbf{X}'^T_i - 1) = 0$ but there exists i such that $\mathbf{X}_i\mathbf{Q}_i\mathbf{X}'^T_i \neq 1$ is negligible. That is, if there exists i such that $\mathbf{X}_i\mathbf{Q}_i\mathbf{X}'^T_i \neq 1$, we have $\sum_{i=1}^{d} t_i(\mathbf{X}_i\mathbf{Q}_i\mathbf{X}'^T_i - 1) = 0 \Leftrightarrow t_i = \frac{\sum_{1\leq j\leq d, j\neq i} t_j(\mathbf{X}_j\mathbf{Q}_j\mathbf{X}'^T_j - 1)}{\mathbf{X}_i\mathbf{Q}_i\mathbf{X}'^T_i - 1}$. Since t_i is randomly chosen from the real domain and the real domain has an unlimited number of real numbers, the probability that t_i is exactly chosen to be $t_i = \frac{\sum_{1\leq j\leq d, j\neq i} t_j(\mathbf{X}_j\mathbf{Q}_j\mathbf{X}'^T_j - 1)}{\mathbf{X}_i\mathbf{Q}_i\mathbf{X}'^T_i - 1}$ is theoretically to be 0. In experiments, we usually use 64-bit double data type to represent t_i. Since the total number of double numbers are 2^{64}, the probability that t_i is exactly chosen to be $t_i = \frac{\sum_{1\leq j\leq d, j\neq i} t_j(\mathbf{X}_j\mathbf{Q}_j\mathbf{X}'^T_j - 1)}{\mathbf{X}_i\mathbf{Q}_i\mathbf{X}'^T_i - 1}$ is about $\frac{1}{2^{64}}$ that is negligible. Thus, we can deduce that

$$\mathbf{x} \in \mathcal{Q} \Leftrightarrow \sum_{i=1}^{d} t_i(\mathbf{X}_i\mathbf{Q}_i\mathbf{X}'^T_i - 1) = 0, \tag{9}$$

and can achieve the multi-dimensional equality test by a single equality test.

4.2 Range Intersection Algorithm

In this subsection, based on the data comparison algorithm in Subsection 3.2, we introduce a multi-dimensional range intersection algorithm. For a clear description, we first introduce a one-dimensional range intersection algorithm.

• **One-dimensional range intersection algorithm:** Same as the one-dimensional point intersection algorithm, the key idea of the one-dimensional range intersection algorithm is to transform the problem of the range intersection into the problem of equality test by using coding technique. Suppose that $B = [b_l, b_u]$ and $Q = (q_l, q_u]$ are two ranges, where $b_l, b_u, q_l, q_u \in [0, N^2 - 1]$. Then, the range intersection algorithm is able to determine whether $B \cap Q \stackrel{?}{=} \emptyset$. When $B \cap Q \neq \emptyset$, we have $b_l \leq q_u$ and $b_u > q_l$. According to the data comparison algorithm, we represent b_l to three vectors $\{\widetilde{\mathbf{b}}_l, \bar{\mathbf{b}}_l, \hat{\mathbf{b}}_l\}$ and further use them to construct $\mathbf{B}_l = [\widetilde{\mathbf{b}}_l^T \ \bar{\mathbf{b}}_l^T]$ and $\mathbf{B}'_l = [1 \ \hat{\mathbf{b}}_l^T]$. Similarly, we represent b_u to three vectors as $\{\widetilde{\mathbf{b}}_u, \bar{\mathbf{b}}_u, \hat{\mathbf{b}}_u\}$ and use them to construct $\mathbf{B}_u = [\widetilde{\mathbf{b}}_u^T \ \bar{\mathbf{b}}_u^T]$ and $\mathbf{B}'_u = [1 \ \hat{\mathbf{b}}_u^T]$. Meanwhile, we can represent q_l and q_u to two $2N \times (N+1)$ matrices $\mathbf{Q}_l$ and $\mathbf{Q}_u$ as Eq. (4). In this case, we have

$$B \cap Q \neq \emptyset \Leftrightarrow b_l \leq q_u \text{ and } b_u > q_l \Leftrightarrow \begin{cases} \mathbf{B}_l\mathbf{Q}_u\mathbf{B}'^T_l = 1 \\ \mathbf{B}_u\mathbf{Q}_l\mathbf{B}'^T_u = 0. \end{cases} \tag{10}$$

Then, the one-dimensional range intersection $B \cap Q \stackrel{?}{=} \emptyset$ is transformed into the equality test $\mathbf{B}_l\mathbf{Q}_u\mathbf{B}'^T_l \stackrel{?}{=} 1$ and $\mathbf{B}_u\mathbf{Q}_l\mathbf{B}'^T_u \stackrel{?}{=} 0$.

• **Multi-dimensional range intersection algorithm:** Same as the multi-dimensional point intersection algorithm, the key idea of the multi-dimensional range intersection algorithm is to

(i) decompose multi-dimensional range intersection into multiple one-dimensional range intersection;
(ii) transform the problem of one-dimensional range intersection determination into an equality test;
(iii) incorporate all equality tests into one equality test.

Suppose that $\mathcal{B} = (B_1, B_2, \cdots, B_d)$ and $\mathcal{Q} = (Q_1, Q_2, \cdots, Q_d)$ are two ranges, where $B_i = [b_{l,i}, b_{u,i}]$, $Q_i = (q_{l,i}, q_{u,i}]$, and $b_{l,i}, b_{u,i}, q_{l,i}, q_{u,i} \in [0, N^2 - 1]$ for $i = 1, 2, \cdots, d$. Then, the range intersection algorithm is able to determine whether $\mathcal{B} \cap \mathcal{Q} \stackrel{?}{=} \emptyset$. When $\mathcal{B} \cap \mathcal{Q} \neq \emptyset$, $B_i \cap Q_i \neq \emptyset$ for $i = 1, 2, \cdots, d$. Then, we can determine whether $\mathcal{B} \cap \mathcal{Q} \stackrel{?}{=} \emptyset$ as follows.

According to the range intersection algorithm for one-dimensional data, we represent each B_i to four matrices $\{\mathbf{B}_{l,i}, \mathbf{B}'_{l,i}, \mathbf{B}_{u,i}, \mathbf{B}'_{u,i}\}$ and represent each Q_i to two matrices $\{\mathbf{Q}_{l,i}, \mathbf{Q}_{u,i}\}$ as Eq. (4). In this case, we can deduce that

$$\begin{aligned} \mathcal{B} \cap \mathcal{Q} \neq \emptyset &\Leftrightarrow B_i \cap Q_i \neq \emptyset \text{ for } i = 1, 2, \cdots, d \\ &\Leftrightarrow \begin{cases} \mathbf{B}_{l,i}\mathbf{Q}_{u,i}{\mathbf{B}'_{l,i}}^T = 1 \\ \mathbf{B}_{u,i}\mathbf{Q}_{l,i}{\mathbf{B}'_{u,i}}^T = 0 \end{cases} \text{for } i = 1, 2, \cdots, d. \end{aligned}$$

In this case, we have transformed the problem of multi-dimensional range intersection into the above multiple equality tests. Then, we incorporate multiple equality tests into one equality test. Specifically, if the equalities $\mathbf{B}_{l,i}\mathbf{Q}_{u,i}{\mathbf{B}'_{l,i}}^T = 1$ and $\mathbf{B}_{u,i}\mathbf{Q}_{l,i}{\mathbf{B}'_{u,i}}^T = 0$ hold for $i = 1, 2, \cdots, d$, for any random non-zero real numbers $\{t_{l,i}, t_{u,i}\}_{i=1}^{d}$, we have

$$\sum_{i=1}^{d} t_{l,i}(\mathbf{B}_{l,i}\mathbf{Q}_{u,i}{\mathbf{B}'_{l,i}}^T - 1) + \sum_{i=1}^{d} t_{u,i}\mathbf{B}_{u,i}\mathbf{Q}_{l,i}{\mathbf{B}'_{u,i}}^T = 0.$$

Similar to the multi-dimensional point intersection algorithm, the probability that $\sum_{i=1}^{d} t_{l,i}(\mathbf{B}_{l,i}\mathbf{Q}_{u,i}{\mathbf{B}'_{l,i}}^T - 1) + \sum_{i=1}^{d} t_{u,i}\mathbf{B}_{u,i}\mathbf{Q}_{l,i}{\mathbf{B}'_{u,i}}^T = 0$ but there exists i such that $\mathbf{B}_{l,i}\mathbf{Q}_{u,i}{\mathbf{B}'_{l,i}}^T \neq 1$ or $\mathbf{B}_{u,i}\mathbf{Q}_{l,i}{\mathbf{B}'_{u,i}}^T \neq 0$ is negligible. Thus, we can deduce that

$$\begin{aligned} &\mathcal{B} \cap \mathcal{Q} \neq \emptyset \\ \Leftrightarrow &\sum_{i=1}^{d} t_{l,i}(\mathbf{B}_{l,i}\mathbf{Q}_{u,i}{\mathbf{B}'_{l,i}}^T - 1) + \sum_{i=1}^{d} t_{u,i}\mathbf{B}_{u,i}\mathbf{Q}_{l,i}{\mathbf{B}'_{u,i}}^T = 0. \end{aligned} \tag{11}$$

5 Intersection Predicate Encryption

In this section, base on the multi-dimensional point intersection algorithm and multi-dimensional range intersection algorithm, we propose a point intersection predicate encryption scheme, i.e., PIPE scheme, and a range intersection predicate encryption scheme, i.e., RIPE scheme, which can perform multi-dimensional point intersection and range intersection over encrypted data.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2021.3101120, IEEE Transactions on Dependable and Secure Computing

5.1 Point Intersection Predicate Encryption

In this subsection, we introduce the PIPE scheme, which can determine $\mathbf{x} \stackrel{?}{\in} \mathcal{Q}$ based on the ciphertexts of $\mathbf{x}$ and $\mathcal{Q}$.

For a clear description, we first introduce the idea about how to encrypt $\mathbf{x}$ and $\mathcal{Q}$. According to Eq. (9), we have $\mathbf{x} \in \mathcal{Q} \Leftrightarrow \sum_{i=1}^{d} t_i(\mathbf{X}_i\mathbf{Q}_i\mathbf{X'}_i^T - 1) = 0$, where $\mathbf{X}_i$ is a $1 \times 2N$ vector, $\mathbf{Q}_i$ is a $2N \times (N+1)$ matrix, and $\mathbf{X}'_i$ is an $(N+1) \times 1$ vector. Then, we can determine $\mathbf{x} \stackrel{?}{\in} \mathcal{Q}$ by determining $\sum_{i=1}^{d} t_i(\mathbf{X}_i\mathbf{Q}_i\mathbf{X'}_i^T - 1) \stackrel{?}{=} 0$. In this case, we only need to encrypt $\mathbf{X}_i$, $\mathbf{Q}_i$, and $\mathbf{X'}_i^T$. We desire to use the produced ciphertexts to determine $\sum_{i=1}^{d} t_i(\mathbf{X}_i\mathbf{Q}_i\mathbf{X'}_i^T - 1) \stackrel{?}{=} 0$ without disclosing the private information of the plaintexts. Since $\mathbf{X}_i$, $\mathbf{Q}_i$, and $\mathbf{X'}_i^T$ are either vector or matrix, we employ the lightweight matrix encryption to encrypt $\mathbf{X}_i, \mathbf{Q}_i, \mathbf{X'}_i^T$. However, it is not secure to directly encrypt $\mathbf{X}_i, \mathbf{Q}_i$, and $\mathbf{X'}_i^T$ [20]. Thus, we first introduce some random numbers into them, and then apply the matrix encryption to encrypt them. The details are as follows.

Step 1: We first choose d random numbers $\{s_i\}_{i=1}^{d}$ satisfying $\sum_{i=1}^{d} s_i = 0$. Then, we choose random matrices $\mathbf{R}_{\mathbf{x},i} \in \mathbb{R}^{1\times 2}$, $\mathbf{R}'_{\mathbf{x},i} \in \mathbb{R}^{2\times 1}$ and $\mathbf{R}_{\mathcal{Q},i} \in \mathbb{R}^{2\times 2}$ such that $\mathbf{R}_{\mathbf{x},i}\mathbf{R}_{\mathcal{Q},i}\mathbf{R}'_{\mathbf{x},i} = 0$ for $i = 1, 2, \cdots, d$. In this case, we have

$$\begin{aligned}
&\sum_{i=1}^{d} t_i(\mathbf{X}_i\mathbf{Q}_i\mathbf{X'}_i^T - 1) = 0 \\
\Leftrightarrow &\sum_{i=1}^{d} (t_i(\mathbf{X}_i\mathbf{Q}_i\mathbf{X'}_i^T - 1) + \mathbf{R}_{\mathbf{x},i}\mathbf{R}_{\mathcal{Q},i}\mathbf{R}'_{\mathbf{x},i}) = 0 \\
\Leftrightarrow &\sum_{i=1}^{d} (t_i(\mathbf{X}_i\mathbf{Q}_i\mathbf{X'}_i^T - 1) + \mathbf{R}_{\mathbf{x},i}\mathbf{R}_{\mathcal{Q},i}\mathbf{R}'_{\mathbf{x},i} + s_i) = 0.
\end{aligned}$$

Let

$$\begin{cases}
\widetilde{\mathbf{X}}_i = \begin{bmatrix} t_i\mathbf{X}_i & \mathbf{R}_{\mathbf{x},i} & t_i & 1 \end{bmatrix}; \\
\widetilde{\mathbf{Q}}_i = \begin{bmatrix} \mathbf{Q}_i & \mathbf{O} & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \mathbf{R}_{\mathcal{Q},i} & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \mathbf{O} & -1 & 0 \\ \mathbf{O} & \mathbf{O} & 0 & s_i \end{bmatrix}; \widetilde{\mathbf{X}}'_i = \begin{bmatrix} \mathbf{X'}_i^T \\ \mathbf{R}'_{\mathbf{x},i} \\ 1 \\ 1 \end{bmatrix}.
\end{cases}$$

We have $\mathbf{x} \in \mathcal{Q} \Leftrightarrow \sum_{i=1}^{d} \widetilde{\mathbf{X}}_i\widetilde{\mathbf{Q}}_i\widetilde{\mathbf{X}}'_i = 0$. Note that the random numbers $\{s_i\}_{i=1}^{d}$ can be removed *iff* the data of all dimensions are processed together. With this property, $\{s_i\}_{i=1}^{d}$ can preserve the single-dimensional privacy.

Step 2: We employ the matrix encryption to preserve the privacy of the computation $\sum_{i=1}^{d} \widetilde{\mathbf{X}}_i\widetilde{\mathbf{Q}}_i\widetilde{\mathbf{X}}'_i$. Let $\widetilde{\mathbf{M}} \in \mathbb{R}^{(2N+4)\times(2N+4)}$ and $\widetilde{\mathbf{M}}' \in \mathbb{R}^{(N+5)\times(N+5)}$ denote two invertible matrices. We respectively encrypt $\{\widetilde{\mathbf{X}}_i, \widetilde{\mathbf{X}}'_i\}_{i=1}^{d}$ and $\{\widetilde{\mathbf{Q}}_i\}_{i=1}^{d}$ as the following equations

$$\begin{cases}
\mathtt{CP}_{i,1} = r_1\widetilde{\mathbf{X}}_i\widetilde{\mathbf{M}}; \ \mathtt{CP}_{i,2} = r_2\widetilde{\mathbf{M}}'\widetilde{\mathbf{X}}'_i \\
\mathtt{PT}_i = r_q\widetilde{\mathbf{M}}^{-1}\widetilde{\mathbf{Q}}_i\widetilde{\mathbf{M}}'^{-1},
\end{cases} \tag{12}$$

where r_1, r_2, r_q are non-zero random real numbers. Meanwhile, we have

$$\begin{aligned}
\sum_{i=1}^{d} \mathtt{CP}_{i,1}\mathtt{PT}_i\mathtt{CP}_{i,2} &= \sum_{i=1}^{d} (r_1 * r_2 * r_q * \widetilde{\mathbf{X}}_i\widetilde{\mathbf{Q}}_i\widetilde{\mathbf{X}}'_i) \\
&= r_1 * r_2 * r_q * (\sum_{i=1}^{d} \widetilde{\mathbf{X}}_i\widetilde{\mathbf{Q}}_i\widetilde{\mathbf{X}}'_i)
\end{aligned}$$

Since r_1, r_2, r_q are non-zero numbers, we can deduce that $\sum_{i=1}^{d} \mathtt{CP}_{i,1}\mathtt{PT}_i\mathtt{CP}_{i,2} = 0 \Leftrightarrow \sum_{i=1}^{d} \widetilde{\mathbf{X}}_i\widetilde{\mathbf{Q}}_i\widetilde{\mathbf{X}}'_i = 0$, and have $\mathbf{x} \in \mathcal{Q} \Leftrightarrow \sum_{i=1}^{d} \widetilde{\mathbf{X}}_i\widetilde{\mathbf{Q}}_i\widetilde{\mathbf{X}}'_i = 0 \Leftrightarrow \sum_{i=1}^{d} \mathtt{CP}_{i,1}\mathtt{PT}_i\mathtt{CP}_{i,2} = 0$.

Based on this idea, the PIPE scheme $\Pi_{\mathsf{PIPE}} = (\mathsf{PipeKeyGen}, \mathsf{PipeEnc}, \mathsf{PipeTokenGen}, \mathsf{PipeQuery})$ can be defined as follows.

- $\mathsf{PipeKeyGen}(N)$: Given N, the key generation algorithm outputs the secret key $sk_P = \{\widetilde{\mathbf{M}}, \widetilde{\mathbf{M}}'\}$.
- $\mathsf{PipeEnc}(\mathbf{x}, sk_P)$: On input $\mathbf{x}$ and sk_P, the encryptor first constructs the matrices $\{\widetilde{\mathbf{X}}_i, \widetilde{\mathbf{X}}'_i\}_{i=1}^{d}$ based on $\mathbf{x}$. Then, the encryptor encrypts $\{\widetilde{\mathbf{X}}_i, \widetilde{\mathbf{X}}'_i\}_{i=1}^{d}$ into the ciphertexts $\{\mathtt{CP}_{i,1}, \mathtt{CP}_{i,2}\}_{i=1}^{d}$ as Eq. (12).
- $\mathsf{PipeTokenGen}(\mathcal{Q}, sk_P)$: On input $\mathcal{Q}$ and sk_P, the token generator first constructs matrices $\{\widetilde{\mathbf{Q}}_i\}_{i=1}^{d}$ based on $\mathcal{Q}$, and outputs the query tokens $\{\mathtt{PT}_i\}_{i=1}^{d}$ as Eq. (12).
- $\mathsf{PipeQuery}(\{\mathtt{CP}_{i,1}, \mathtt{CP}_{i,2}\}_{i=1}^{d}, \{\mathtt{PT}_i\}_{i=1}^{d})$: On input the ciphertexts $\{\mathtt{CP}_{i,1}, \mathtt{CP}_{i,2}\}_{i=1}^{d}$ and the query token $\{\mathtt{PT}_i\}_{i=1}^{d}$, the query algorithm determines whether $\mathbf{x} \stackrel{?}{\in} \mathcal{Q}$ by determining $\sum_{i=1}^{d} \mathtt{CP}_{i,1}\mathtt{PT}_i\mathtt{CP}_{i,2} \stackrel{?}{=} 0$. If $\sum_{i=1}^{d} \mathtt{CP}_{i,1}\mathtt{PT}_i\mathtt{CP}_{i,2} = 0$, the query algorithm returns 1 to denote $\mathbf{x} \in \mathcal{Q}$ and returns 0 to denote $\mathbf{x} \notin \mathcal{Q}$ otherwise.

Remark. In the PIPE scheme, to involve more random numbers into the ciphertexts and tokens, we respectively add three random matrices $\mathbf{R}_{\mathbf{x},i} \in \mathbb{R}^{1\times 2}$, $\mathbf{R}'_{\mathbf{x},i} \in \mathbb{R}^{2\times 1}$ and $\mathbf{R}_{\mathcal{Q},i} \in \mathbb{R}^{2\times 2}$ into the ciphertexts $\mathtt{CP}_{i,1}$, $\mathtt{CP}_{i,2}$, and the token $\mathtt{PT}_i$, where $\mathbf{R}_{\mathbf{x},i}\mathbf{R}_{\mathcal{Q},i}\mathbf{R}'_{\mathbf{x},i} = 0$ for $i = 1, 2, \cdots, d$. Since the ciphertexts and tokens are separately generated by the data owner and query users in our scheme, the random matrices $\{\mathbf{R}_{\mathbf{x},i}, \mathbf{R}'_{\mathbf{x},i}\}$ and $\mathbf{R}_{\mathcal{Q},i}$ should be separately chosen by the data owner and query users. In the following, we show a method to separately choose them satisfying $\mathbf{R}_{\mathbf{x},i}\mathbf{R}_{\mathcal{Q},i}\mathbf{R}'_{\mathbf{x},i} = 0$.

- $\{\mathbf{R}_{\mathbf{x},i}, \mathbf{R}'_{\mathbf{x},i}\}$: The data owner chooses two random matrices $\{\mathbf{R}_{\mathbf{x},i}, \mathbf{R}'_{\mathbf{x},i}\}$ as

$$\mathbf{R}_{\mathbf{x},i} = [u_{\mathbf{x},i}, u_{\mathbf{x},i}]; \ \mathbf{R}'_{\mathbf{x},i} = \begin{bmatrix} u'_{\mathbf{x},i} \\ u'_{\mathbf{x},i}, \end{bmatrix}$$

where $u_{\mathbf{x},i} \in \mathbb{R}$ and $u'_{\mathbf{x},i} \in \mathbb{R}$ are two random numbers.

- $\mathbf{R}_{\mathcal{Q},i}$: Query users choose a random matrix $\mathbf{R}_{\mathcal{Q},i}$ as

$$\mathbf{R}_{\mathcal{Q},i} = \begin{bmatrix} r_{\mathcal{Q},i,1} & r_{\mathcal{Q},i,2} \\ r_{\mathcal{Q},i,3} & -(r_{\mathcal{Q},i,1} + r_{\mathcal{Q},i,2} + r_{\mathcal{Q},i,3}) \end{bmatrix}$$

where $\{r_{\mathcal{Q},i,1}, r_{\mathcal{Q},i,2}, r_{\mathcal{Q},i,3}\} \in \mathbb{R}$ are random numbers.

Correctness. It is easy to verify that these generated random matrices satisfy that $\mathbf{R}_{\mathbf{x},i}\mathbf{R}_{\mathcal{Q},i}\mathbf{R}'_{\mathbf{x},i} = 0$. Specifically, we have

$$\begin{aligned}
&\mathbf{R}_{\mathbf{x},i}\mathbf{R}_{\mathcal{Q},i}\mathbf{R}'_{\mathbf{x},i} \\
&= [u_{\mathbf{x},i}, u_{\mathbf{x},i}] \begin{bmatrix} r_{\mathcal{Q},i,1} & r_{\mathcal{Q},i,2} \\ r_{\mathcal{Q},i,3} & -(r_{\mathcal{Q},i,1} + r_{\mathcal{Q},i,2} + r_{\mathcal{Q},i,3}) \end{bmatrix} \begin{bmatrix} u'_{\mathbf{x},i} \\ u'_{\mathbf{x},i} \end{bmatrix} \\
&= \sum_{k=1}^{2}\sum_{j=1}^{2} (\mathbf{R}_{\mathbf{x},i})_{1,k} * (\mathbf{R}_{\mathcal{Q},i})_{k,j} * (\mathbf{R}'_{\mathbf{x},i})_{j,1} \\
&= \sum_{k=1}^{2}\sum_{j=1}^{2} u_{\mathbf{x},i} * (\mathbf{R}_{\mathcal{Q},i})_{k,j} * u'_{\mathbf{x},i} \\
&= u_{\mathbf{x},i} * u'_{\mathbf{x},i} * (\sum_{k=1}^{2}\sum_{j=1}^{2} (\mathbf{R}_{\mathcal{Q},i})_{k,j}) = u_{\mathbf{x},i} * u'_{\mathbf{x},i} * 0 = 0,
\end{aligned}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2021.3101120, IEEE Transactions on Dependable and Secure Computing

Thus, $\mathbf{R}_{\mathbf{x},i}$, $\mathbf{R}'_{\mathbf{x},i}$, and $\mathbf{R}_{\mathcal{Q},i}$ satisfy $\mathbf{R}_{\mathbf{x},i}\mathbf{R}_{\mathcal{Q},i}\mathbf{R}'_{\mathbf{x},i} = 0$. In this case, the data owner and query users can use the above method to generate random matrices.

5.2 Range Intersection Predicate Encryption

In this subsection, we propose a multi-dimensional range intersection predicate encryption (RIPE) scheme, which can determine $\mathcal{B} \cap \mathcal{Q} \stackrel{?}{=} \emptyset$ based on the ciphertexts of $\mathcal{B}$ and $\mathcal{Q}$.

For a clear description, we first introduce the idea on how to encrypt $\mathcal{B}$ and $\mathcal{Q}$. According to Eq. (11), we have $\mathcal{B} \cap \mathcal{Q} \neq \emptyset \Leftrightarrow \sum_{i=1}^{d} t_{l,i}(\mathbf{B}_{l,i}\mathbf{Q}_{u,i}{\mathbf{B}'_{l,i}}^T - 1) + \sum_{i=1}^{d} t_{u,i}\mathbf{B}_{u,i}\mathbf{Q}_{l,i}{\mathbf{B}'_{u,i}}^T = 0$. Then, we can determine $\mathcal{B} \cap \mathcal{Q} \stackrel{?}{=} \emptyset$ by determining $\sum_{i=1}^{d} t_{l,i}(\mathbf{B}_{l,i}\mathbf{Q}_{u,i}{\mathbf{B}'_{l,i}}^T - 1) + \sum_{i=1}^{d} t_{u,i}\mathbf{B}_{u,i}\mathbf{Q}_{l,i}{\mathbf{B}'_{u,i}}^T \stackrel{?}{=} 0$. In this case, we only need to encrypt $\mathbf{B}_{l,i}, \mathbf{Q}_{l,i}, \mathbf{B}'_{l,i}, \mathbf{B}_{u,i}, \mathbf{Q}_{u,i}$, and $\mathbf{B}'_{u,i}$. We desire to use the produced ciphertexts to determine $\sum_{i=1}^{d} t_{l,i}(\mathbf{B}_{l,i}\mathbf{Q}_{u,i}{\mathbf{B}'_{l,i}}^T - 1) + \sum_{i=1}^{d} t_{u,i}\mathbf{B}_{u,i}\mathbf{Q}_{l,i}{\mathbf{B}'_{u,i}}^T \stackrel{?}{=} 0$ without disclosing the private information of the plaintexts. Similar to the PIPE scheme, we employ the lightweight matrix encryption to encrypt $\mathbf{B}_{l,i}, \mathbf{Q}_{l,i}, \mathbf{B}'_{l,i}, \mathbf{B}_{u,i}, \mathbf{Q}_{u,i}, \mathbf{B}'_{u,i}$. Specifically, we first introduce some random numbers into them, and then apply the matrix encryption to encrypt them. The details are as follows.

Step 1: We choose random numbers $\{s_{l,i}, s_{u,i}\}_{i=1}^{d}$ satisfying $\sum_{i=1}^{d}(s_{l,i} + s_{u,i}) = 0$. Then, we choose random matrices $\{\mathbf{R}_{\mathcal{B},l,i}, \mathbf{R}_{\mathcal{B},u,i}\} \in \mathbb{R}^{1\times 2}$, $\{\mathbf{R}_{\mathcal{Q},l,i}, \mathbf{R}_{\mathcal{Q},u,i}\} \in \mathbb{R}^{2\times 2}$ and $\{\mathbf{R}'_{\mathcal{B},l,i}, \mathbf{R}'_{\mathcal{B},u,i}\} \in \mathbb{R}^{2\times 1}$ such that $\mathbf{R}_{\mathcal{B},l,i}\mathbf{R}_{\mathcal{Q},u,i}\mathbf{R}'_{\mathcal{B},l,i} = 0$ and $\mathbf{R}_{\mathcal{B},u,i}\mathbf{R}_{\mathcal{Q},l,i}\mathbf{R}'_{\mathcal{B},u,i} = 0$ for $1 \le i \le d$. Then, we have

$$\begin{aligned}
&\sum_{i=1}^{d} t_{l,i}(\mathbf{B}_{l,i}\mathbf{Q}_{u,i}{\mathbf{B}'_{l,i}}^T - 1) + \sum_{i=1}^{d} t_{u,i}\mathbf{B}_{u,i}\mathbf{Q}_{l,i}{\mathbf{B}'_{u,i}}^T = 0\\
\Leftrightarrow &\sum_{i=1}^{d}(t_{l,i}(\mathbf{B}_{l,i}\mathbf{Q}_{u,i}{\mathbf{B}'_{l,i}}^T - 1) + \mathbf{R}_{\mathcal{B},l,i}\mathbf{R}_{\mathcal{Q},u,i}\mathbf{R}'_{\mathcal{B},l,i} + s_{u,i})\\
&+\sum_{i=1}^{d}(t_{u,i}\mathbf{B}_{u,i}\mathbf{Q}_{l,i}{\mathbf{B}'_{u,i}}^T + \mathbf{R}_{\mathcal{B},u,i}\mathbf{R}_{\mathcal{Q},l,i}\mathbf{R}'_{\mathcal{B},u,i} + s_{l,i}) = 0.
\end{aligned}$$

If we let

$$\begin{cases}
\bar{\mathbf{B}}_{l,i} = \begin{bmatrix} t_{l,i}\mathbf{B}_{l,i} & \mathbf{R}_{\mathbf{B},l,i} & -t_{l,i} & 1 \end{bmatrix};\\
\bar{\mathbf{Q}}_{u,i} = \begin{bmatrix} \mathbf{Q}_{u,i} & \mathbf{O} & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \mathbf{R}_{\mathcal{Q},u,i} & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \mathbf{O} & 1 & 0 \\ \mathbf{O} & \mathbf{O} & 0 & s_{u,i} \end{bmatrix}; \bar{\mathbf{B}}'_{l,i} = \begin{bmatrix} {\mathbf{B}'_{l,i}}^T \\ \mathbf{R}'_{\mathcal{B},l,i} \\ 1 \\ 1 \end{bmatrix};\\
\bar{\mathbf{B}}_{u,i} = \begin{bmatrix} t_{u,i}\mathbf{B}_{u,i} & \mathbf{R}_{\mathbf{B},u,i} & 0 & 1 \end{bmatrix};\\
\bar{\mathbf{Q}}_{l,i} = \begin{bmatrix} \mathbf{Q}_{l,i} & \mathbf{O} & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \mathbf{R}_{\mathcal{Q},l,i} & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \mathbf{O} & 0 & 0 \\ \mathbf{O} & \mathbf{O} & 0 & s_{l,i} \end{bmatrix}; \bar{\mathbf{B}}'_{u,i} = \begin{bmatrix} {\mathbf{B}'_{u,i}}^T \\ \mathbf{R}'_{\mathcal{B},u,i} \\ 0 \\ 1 \end{bmatrix};
\end{cases}$$

we have $\mathcal{B} \cap \mathcal{Q} \neq \emptyset \Leftrightarrow \sum_{i=1}^{d}(\bar{\mathbf{B}}_{l,i}\bar{\mathbf{Q}}_{u,i}\bar{\mathbf{B}}'_{l,i} + \bar{\mathbf{B}}_{u,i}\bar{\mathbf{Q}}_{l,i}\bar{\mathbf{B}}'_{u,i}) = 0$. Similar to PIPE scheme, the random numbers $\{s_{l,i}, s_{u,i}\}_{i=1}^{d}$ are used for preserving the single-dimensional privacy.

Step 2: We employ the matrix encryption to preserve the privacy of the computation $\sum_{i=1}^{d}(\bar{\mathbf{B}}_{l,i}\bar{\mathbf{Q}}_{u,i}\bar{\mathbf{B}}'_{l,i} + \bar{\mathbf{B}}_{u,i}\bar{\mathbf{Q}}_{l,i}\bar{\mathbf{B}}'_{u,i})$. Let $\bar{\mathbf{M}} \in \mathbb{R}^{(2N+4)\times(2N+4)}$ and $\bar{\mathbf{M}}' \in \mathbb{R}^{(N+5)\times(N+5)}$ denote two invertible matrices. We respectively encrypt $\{\bar{\mathbf{B}}_{l,i}, \bar{\mathbf{B}}'_{l,i}, \bar{\mathbf{B}}_{u,i}, \bar{\mathbf{B}}'_{u,i}\}$ and $\{\bar{\mathbf{Q}}_{u,i}, \bar{\mathbf{Q}}_{l,i}\}$ as the following equations

$$\begin{cases}
\mathtt{CB}_{l,i,1} = r'_1\bar{\mathbf{B}}_{l,i}\bar{\mathbf{M}}; & \mathtt{CB}_{u,i,1} = r'_1\bar{\mathbf{B}}_{u,i}\bar{\mathbf{M}};\\
\mathtt{CB}_{l,i,2} = r'_2\bar{\mathbf{M}}'^{-1}\bar{\mathbf{B}}'_{l,i}; & \mathtt{CB}_{u,i,2} = r'_2\bar{\mathbf{M}}'^{-1}\bar{\mathbf{B}}'_{u,i};\\
\mathtt{RT}_{u,i} = r'_q\bar{\mathbf{M}}^{-1}\bar{\mathbf{Q}}_{u,i}\bar{\mathbf{M}}'; & \mathtt{RT}_{l,i} = r'_q\bar{\mathbf{M}}^{-1}\bar{\mathbf{Q}}_{l,i}\bar{\mathbf{M}}',
\end{cases} \tag{13}$$

where r'_1, r'_2, r'_q are non-zero random real numbers. Meanwhile, we have

$$\begin{aligned}
&\sum_{i=1}^{d}(\mathtt{CB}_{l,i,1}\mathtt{RT}_{u,i}\mathtt{CB}_{l,i,2} + \mathtt{CB}_{u,i,1}\mathtt{RT}_{l,i}\mathtt{CB}_{u,i,2})\\
=&\sum_{i=1}^{d} r'_1 * r'_2 * r'_q * (\bar{\mathbf{B}}_{l,i}\bar{\mathbf{Q}}_{u,i}\bar{\mathbf{B}}'_{l,i} + \bar{\mathbf{B}}_{u,i}\bar{\mathbf{Q}}_{l,i}\bar{\mathbf{B}}'_{u,i})\\
=&\, r'_1 * r'_2 * r'_q * \sum_{i=1}^{d}(\bar{\mathbf{B}}_{l,i}\bar{\mathbf{Q}}_{u,i}\bar{\mathbf{B}}'_{l,i} + \bar{\mathbf{B}}_{u,i}\bar{\mathbf{Q}}_{l,i}\bar{\mathbf{B}}'_{u,i}).
\end{aligned}$$

Since r'_1, r'_2, r'_q are non-zero numbers, we can deduce that

$$\begin{aligned}
\mathcal{B} \cap \mathcal{Q} \neq \emptyset &\Leftrightarrow \sum_{i=1}^{d}(\bar{\mathbf{B}}_{l,i}\bar{\mathbf{Q}}_{u,i}\bar{\mathbf{B}}'_{l,i} + \bar{\mathbf{B}}_{u,i}\bar{\mathbf{Q}}_{l,i}\bar{\mathbf{B}}'_{u,i}) = 0\\
&\Leftrightarrow \sum_{i=1}^{d}(\mathtt{CB}_{l,i,1}\mathtt{RT}_{u,i}\mathtt{CB}_{l,i,2} + \mathtt{CB}_{u,i,1}\mathtt{RT}_{l,i}\mathtt{CB}_{u,i,2}) = 0.
\end{aligned}$$

Based on this idea, the RIPE scheme $\Pi_{\mathsf{RIPE}} = (\mathsf{RipeKeyGen}, \mathsf{RipeEnc}, \mathsf{RipeTokenGen}, \mathsf{RipeQuery})$ can be defined as follows.

- $\mathsf{RipeKeyGen}(N)$: Given N, the key generation algorithm outputs the secret key $sk_R = \{\bar{\mathbf{M}}, \bar{\mathbf{M}}'\}$.
- $\mathsf{RipeEnc}(\mathcal{B}, sk_R)$: On input $\mathcal{B}$ and sk_R, the encryptor constructs the matrices $\{\bar{\mathbf{B}}_{l,i}, \bar{\mathbf{B}}'_{l,i}, \bar{\mathbf{B}}_{u,i}, \bar{\mathbf{B}}'_{u,i}\}_{i=1}^{d}$ based on $\mathcal{B}$. Then, it encrypts $\{\bar{\mathbf{B}}_{l,i}, \bar{\mathbf{B}}'_{l,i}, \bar{\mathbf{B}}_{u,i}, \bar{\mathbf{B}}'_{u,i}\}_{i=1}^{d}$ into the ciphertext $\mathtt{C} = \{\mathtt{CB}_{l,i,1}, \mathtt{CB}_{l,i,2}, \mathtt{CB}_{u,i,1}, \mathtt{CB}_{u,i,2}\}_{i=1}^{d}$ as Eq. (13).
- $\mathsf{RipeTokenGen}(\mathcal{Q}, sk_R)$: On input $\mathcal{Q}$ and sk_R, the token generator first constructs matrices $\{\bar{\mathbf{Q}}_{l,i}, \bar{\mathbf{Q}}_{u,i}\}_{i=1}^{d}$ based on $\mathcal{Q}$, and then encrypts them into the query token $\mathtt{TK} = \{\mathtt{RT}_{l,i}, \mathtt{RT}_{u,i}\}_{i=1}^{d}$ as Eq. (13).
- $\mathsf{RipeQuery}(\mathtt{C}, \mathtt{TK})$: On input the ciphertext $\mathtt{C} = \{\mathtt{CB}_{l,i,1}, \mathtt{CB}_{l,i,2}, \mathtt{CB}_{u,i,1}, \mathtt{CB}_{u,i,2}\}_{i=1}^{d}$ and the query token $\mathtt{TK} = \{\mathtt{RT}_{l,i}, \mathtt{RT}_{u,i}\}_{i=1}^{d}$, the query algorithm can determine whether $\mathcal{B} \cap \mathcal{Q} \stackrel{?}{=} \emptyset$ by determining $\sum_{i=1}^{d}(\mathtt{CB}_{l,i,1}\mathtt{RT}_{u,i}\mathtt{CB}_{l,i,2} + \mathtt{CB}_{u,i,1}\mathtt{RT}_{l,i}\mathtt{CB}_{u,i,2}) \stackrel{?}{=} 0$. If $\sum_{i=1}^{d}(\mathtt{CB}_{l,i,1}\mathtt{RT}_{u,i}\mathtt{CB}_{l,i,2} + \mathtt{CB}_{u,i,1}\mathtt{RT}_{l,i}\mathtt{CB}_{u,i,2}) = 0$, the query algorithm returns 1 to denote $\mathcal{B} \cap \mathcal{Q} \neq \emptyset$. Otherwise, the algorithm returns 0 to denote $\mathcal{B} \cap \mathcal{Q} = \emptyset$.

6 Our Privacy-Preserving Range Query Scheme

Based on the PIPE and RIPE schemes, we present a privacy-preserving range query scheme, i.e., PRQ scheme. The PRQ scheme $\Pi_{\mathsf{PRQ}} = (\mathsf{PrqKeyGen}, \mathsf{PrqEnc}, \mathsf{PrqTokenGen}, \mathsf{PrqQuery})$ can be defined as follows.

- $\mathsf{PrqKeyGen}(N, \kappa)$: On input N and the security parameter κ, the data owner generates three secret keys as

$$\begin{aligned}
sk_P &= \{\widetilde{\mathbf{M}}, \widetilde{\mathbf{M}}'\} \leftarrow \mathsf{PipeKeyGen}(N)\\
sk_R &= \{\bar{\mathbf{M}}, \bar{\mathbf{M}}'\} \leftarrow \mathsf{RipeKeyGen}(N)\\
K &\in \{0,1\}^{\kappa} \text{ for AES algorithm},
\end{aligned}$$

where sk_P, K are used for encrypting data points, and sk_R is used for encrypting the MBRs. For each query user, the data owner authorizes them with $\{sk_P, sk_R, K\}$.

• $\mathsf{PrqEnc}(\mathcal{X}, sk_P, sk_R, K)$: On input a multi-dimensional dataset $\mathcal{X}$ and the secret keys $\{sk_P, sk_R, K\}$, the data owner encrypts $\mathcal{X}$ as follows.

Step 1: The data owner represents $\mathcal{X}$ to an R-tree T. For preserving the privacy of the dataset, the data owner randomly permutes the child nodes of each internal node.

Step 2: Each internal node of T corresponds to an MBR $\mathcal{B}$ and it is encrypted as $\{\mathtt{CB}_{l,i,1}, \mathtt{CB}_{l,i,2}, \mathtt{CB}_{u,i,1}, \mathtt{CB}_{u,i,2}\}_{i=1}^{d} \leftarrow \mathsf{RipeEnc}(\mathcal{B}, sk_R)$. Each leaf node of T corresponds to a data point $\mathbf{x}$ and is encrypted as $\{\mathtt{CP}_{i,1}, \mathtt{CP}_{i,2}\}_{i=1}^{d} \leftarrow \mathsf{PipeEnc}(\mathbf{x}, sk_P)$ and $\mathsf{AES}_K(\mathbf{x})$.

Finally, the data owner outsources the encrypted R-tree, denoted by $E(\mathrm{T})$, to the cloud server via a secure channel.

• $\mathsf{PrqTokenGen}(\mathcal{Q}, sk_P, sk_R)$: On input the query range $\mathcal{Q}$ and $\{sk_P, sk_R\}$, the query user respectively generates a point intersection query token and a range intersection query token as

$$\{\mathtt{PT}_i\}_{i=1}^{d} \leftarrow \mathsf{PipeTokenGen}(\mathcal{Q}, sk_P)$$
$$\{\mathtt{RT}_{l,i}, \mathtt{RT}_{u,i}\}_{i=1}^{d} \leftarrow \mathsf{RipeTokenGen}(\mathcal{Q}, sk_R).$$

Then, the query user sends the tokens $\{\mathtt{PT}_i, \mathtt{RT}_{l,i}, \mathtt{RT}_{u,i}\}_{i=1}^{d}$ to the cloud via a secure channel.

• $\mathsf{PrqQuery}(E(\mathrm{T}), \{\mathtt{PT}_i, \mathtt{RT}_{l,i}, \mathtt{RT}_{u,i}\}_{i=1}^{d})$: On input $\{\mathtt{PT}_i, \mathtt{RT}_{l,i}, \mathtt{RT}_{u,i}\}_{i=1}^{d}$, the cloud server searches on $E(\mathrm{T})$ to find out the query result. The searching algorithm over an encrypted R-tree is similar to that over a plaintext R-tree in Algorithm 1. Differently, the conditions $\mathbf{x} \in \mathcal{Q}$ and $\mathcal{B} \cap \mathcal{Q} \neq \emptyset$ are respectively replaced with $\sum_{i=1}^{d} \mathtt{CP}_{i,1}\mathtt{PT}_i\mathtt{CP}_{i,2} = 0$ and $\sum_{i=1}^{d}(\mathtt{CB}_{l,i,1}\mathtt{RT}_{u,i}\mathtt{CB}_{l,i,2} + \mathtt{CB}_{u,i,1}\mathtt{RT}_{l,i}\mathtt{CB}_{u,i,2}) = 0$. Specifically, the searching process contains a filtration phase and a verification phase.

∗ *Filtration phase*: In the filtration phase, the cloud server recursively searches on $E(\mathrm{T})$ to find a candidate set $\mathcal{C}$, which contains all encrypted data records that possibly fall in $\mathcal{Q}$. The cloud server starts the search from the root node. When the searched node is an internal node $(\{\mathtt{CB}_{l,i,1}, \mathtt{CB}_{l,i,2}, \mathtt{CB}_{u,i,1}, \mathtt{CB}_{u,i,2}\}_{i=1}^{d}, \mathtt{p})$, the searcher checks whether $\mathcal{B}$ intersects with $\mathcal{Q}$ by checking $\sum_{i=1}^{d}(\mathtt{CB}_{l,i,1}\mathtt{RT}_{u,i}\mathtt{CB}_{l,i,2} + \mathtt{CB}_{u,i,1}\mathtt{RT}_{l,i}\mathtt{CB}_{u,i,2}) \stackrel{?}{=} 0$. If $\sum_{i=1}^{d}(\mathtt{CB}_{l,i,1}\mathtt{RT}_{u,i}\mathtt{CB}_{l,i,2} + \mathtt{CB}_{u,i,1}\mathtt{RT}_{l,i}\mathtt{CB}_{u,i,2}) = 0$, the searcher needs to search each $node.p_i$ for $p_i \in \mathtt{p}$. When the searched node is a leaf node $(\{\mathtt{CP}_{i,1}, \mathtt{CP}_{i,2}\}_{i=1}^{d}, \mathsf{AES}_K(\mathbf{x}))$. The searcher adds $(\{\mathtt{CP}_{i,1}, \mathtt{CP}_{i,2}\}_{i=1}^{d}, \mathsf{AES}_K(\mathbf{x}))$ into the candidate set $\mathcal{C}$.

∗ *Verification phase*: In the verification phase, the cloud server verifies whether each candidate encrypted data $(\{\mathtt{CP}_{i,1}, \mathtt{CP}_{i,2}\}_{i=1}^{d}, \mathsf{AES}_K(\mathbf{x}))$ is in $\mathcal{Q}$ or not. If $\sum_{i=1}^{d} \mathtt{CP}_{i,1}\mathtt{PT}_i\mathtt{CP}_{i,2} = 0$, the cloud server adds $\mathsf{AES}_K(\mathbf{x})$ into the query result $\mathcal{R}$.

Then, the cloud server returns the query result $\mathcal{R}$ to the query user via a secure channel. On receiving $\mathcal{R}$, the query user decrypts each $\mathsf{AES}_K(\mathbf{x}) \in \mathcal{R}$ with K and obtains the data point $\mathbf{x}$.

7 Security Analysis

In this section, we analyze the security of our PRQ scheme. Since PIPE scheme and RIPE scheme are important building blocks of our PRQ scheme. Thus, we first prove the security of PIPE scheme and RIPE scheme, and then analyze the security of our PRQ scheme.

7.1 Security of PIPE Scheme

PIPE scheme is essentially a functional encryption scheme that is usually proved under two kinds of security models, i.e., selective security and adaptive security. Selective security guarantees the privacy of messages that are fixed before the adversary intersects with the system. While adaptive security guarantees that the privacy of messages that are adaptively chosen at any time (can be before or after the adversary intersects with the system). Due to the hardness of achieving adaptive security, the overwhelming majority of functional schemes prove their security under the selective security model [21]. Same as the existing schemes in [22], [23], we prove that PIPE is selectively secure in the real/ideal world model. Before defining the real and ideal worlds, we first define the leakage of the PIPE scheme. Suppose that $\{\mathtt{CP}_{i,1}, \mathtt{CP}_{i,2}\}_{i=1}^{d}$ and $\{\mathtt{PT}_i\}_{i=1}^{d}$ are respectively the ciphertext of $\mathbf{x}$ and the point intersection token of $\mathcal{Q}$. The leakage of the PIPE scheme is $\mathcal{L}_P(\mathbf{x}, \mathcal{Q}) = \mathsf{PipeQuery}(\{\mathtt{CP}_{i,1}, \mathtt{CP}_{i,2}\}_{i=1}^{d}, \{\mathtt{PT}_i\}_{i=1}^{d})$. Then, we formally define the real and ideal worlds.

Real world: In the real world, there are two participants including a probabilistic polynomial time (PPT) adversary $\mathcal{A}$ and a challenger. They interact with each other as follows.

• ***Key generation:*** $\mathcal{A}$ chooses a data point $\mathbf{x}$ and sends it to the challenger. Given N, the challenger calls $\mathsf{PipeKeyGen}(N)$ to generate the secret key $sk_P = \{\widetilde{\mathbf{M}}, \widetilde{\mathbf{M}}'\}$. Then, the challenger encrypts $\mathbf{x}$ as $\{\mathtt{CP}_{i,1}, \mathtt{CP}_{i,2}\}_{i=1}^{d} \leftarrow \mathsf{PipeEnc}(\mathbf{x}, sk_P)$.

• ***Token generation phase 1:*** $\mathcal{A}$ chooses p_1 query ranges $\{\mathcal{Q}_\beta\}_{\beta=1}^{p_1}$ and sends them the challenger, where p_1 is a polynomial number. The challenger generates a point intersection token for $\mathcal{Q}_\beta$ as $\{\mathtt{PT}_i^\beta\}_{i=1}^{d} \leftarrow \mathsf{PipeTokenGen}(\mathcal{Q}_\beta, sk_P)$ for $\beta = 1, 2, \cdots, p_1$. Finally, the challenger returns $\{\{\mathtt{PT}_i^\beta\}_{i=1}^{d}\}_{\beta=1}^{p_1}$ to $\mathcal{A}$.

• ***Challenge phase:*** The challenger returns the ciphertext $\{\mathtt{CP}_{i,1}, \mathtt{CP}_{i,2}\}_{i=1}^{d}$ to $\mathcal{A}$.

• ***Token generation phase 2:*** Same as the token generation phase 1, $\mathcal{A}$ chooses $p_2 - p_1$ query ranges $\{\mathcal{Q}_\beta\}_{\beta=p_1+1}^{p_2}$ and gets the query tokens $\{\{\mathtt{PT}_i^\beta\}_{i=1}^{d}\}_{\beta=p_1+1}^{p_2}$ from the challenger, where p_2 is a polynomial number.

Ideal world: The ideal experiment involves a PPT adversary $\mathcal{A}$ and a simulator with the leakage $\mathcal{L}_P$. They interact with each other as follows.

• ***Key generation:*** $\mathcal{A}$ chooses a data point $\mathbf{x}$ and sends it to the simulator. Then, the simulator randomly chooses $2d$ vectors $\{\mathtt{CP}'_{i,1}, \mathtt{CP}'_{i,2}\}_{i=1}^{d}$ as the ciphertext of $\mathbf{x}$.

• ***Token generation phase 1:*** $\mathcal{A}$ chooses p_1 query ranges $\{\mathcal{Q}_\beta\}_{\beta=1}^{p_1}$ and sends them the simulator, where p_1 is a polynomial number. The simulator randomly generates point intersection tokens for $\{\mathcal{Q}_\beta\}_{\beta=1}^{p_1}$. For each $\mathcal{Q}_\beta$, the simulator uses $\mathcal{L}_P$ to generate a random number r_β such that

$$\begin{cases} r_\beta = 0 & \text{If } \mathcal{L}_P(\mathbf{x}, \mathcal{Q}_\beta) = 0; \\ r_\beta \neq 0 & \text{If } \mathcal{L}_P(\mathbf{x}, \mathcal{Q}_\beta) \neq 0. \end{cases}$$

Then, the simulator generates d matrices $\{\mathtt{PT}'^{\beta}_i\}^d_{i=1}$ such that

$$\sum_{i=1}^{d}(\mathtt{CP}'_{i,1}\mathtt{PT}'^{\beta}_i\mathtt{CP}'_{i,2}) = r_\beta.$$

Since r_β is a random number, $\{\mathtt{PT}'^{\beta}_i\}^d_{i=1}$ are random matrices. Then, the simulator returns $\{\{\mathtt{PT}'^{\beta}_i\}^d_{i=1}\}^{p_1}_{\beta=1}$ to $\mathcal{A}$.

• ***Challenge phase:*** The simulator returns the ciphertext $\{\mathtt{CP}'_{i,1}, \mathtt{CP}'_{i,2}\}^d_{i=1}$ to $\mathcal{A}$.

• ***Token generation phase 2:*** Same as the token generation phase 1, $\mathcal{A}$ chooses $p_2 - p_1$ query ranges $\{\mathcal{Q}_\beta\}^{p_2}_{\beta=p_1+1}$ and gets the query tokens $\{\{\mathtt{PT}'^{\beta}_i\}^d_{i=1}\}^{p_2}_{\beta=p_1+1}$ from the simulator, where p_2 is a polynomial number.

In the real experiment, the view of $\mathcal{A}$ is $\mathrm{View}_{\mathcal{A},\mathrm{Real}} = \{\{\mathtt{CP}_{i,1}, \mathtt{CP}_{i,2}\}^d_{i=1}, \{\{\mathtt{PT}^{\beta}_i\}^d_{i=1}\}^{p_2}_{\beta=1}\}$. In the ideal experiment, the view of $\mathcal{A}$ is $\mathrm{View}_{\mathcal{A},\mathrm{Ideal}} = \{\{\mathtt{CP}'_{i,1}, \mathtt{CP}'_{i,2}\}^d_{i=1}, \{\{\mathtt{PT}'^{\beta}_i\}^d_{i=1}\}^{p_2}_{\beta=1}\}$. Based on $\mathrm{View}_{\mathcal{A},\mathrm{Real}}$ and $\mathrm{View}_{\mathcal{A},\mathrm{Ideal}}$, we can formally define the selective security of the PIPE scheme.

Definition 1 (Security of PIPE Scheme). The PIPE scheme is selectively secure with the leakage $\mathcal{L}_P$ *iff* for any PPT $\mathcal{A}$ issuing a polynomial number of query token generations, there exists a simulator such that the advantage that $\mathcal{A}$ can distinguish the views of real and ideal worlds is negligible.

Theorem 1. The PIPE scheme is selectively secure with the leakage $\mathcal{L}_P$.

Proof: We prove the PIPE scheme is selectively secure by proving that $\mathrm{View}_{\mathcal{A},\mathrm{Real}}$ is indistinguishable from $\mathrm{View}_{\mathcal{A},\mathrm{Ideal}}$. Since the ciphertext and tokens in $\mathrm{View}_{\mathcal{A},\mathrm{Ideal}}$ are randomly generated, distinguishing $\mathrm{View}_{\mathcal{A},\mathrm{Real}}$ from $\mathrm{View}_{\mathcal{A},\mathrm{Ideal}}$ is equivalent to distinguish $\mathrm{View}_{\mathcal{A},\mathrm{Real}}$ from random numbers. In the following, we show that $\mathcal{A}$ cannot distinguish $\mathrm{View}_{\mathcal{A},\mathrm{Real}} = \{\{\mathtt{CP}_{i,1}, \mathtt{CP}_{i,2}\}^d_{i=1}, \{\{\mathtt{PT}^{\beta}_i\}^d_{i=1}\}^{p_2}_{\beta=1}\}$ from random numbers.

• $\mathcal{A}$ cannot distinguish $\{\mathtt{CP}_{i,1}, \mathtt{CP}_{i,2}\}^d_{i=1}$ from random numbers. Since the ciphertexts of $\mathtt{CP}_{i,1}$ and $\mathtt{CP}_{i,2}$ respectively involve random numbers $\{t_i, \mathbf{R}_{\mathbf{x},i}, r_1\}^d_{i=1}$ and $\{\mathbf{R}'_{\mathbf{x},i}, r_2\}^d_{i=1}$, which make $\{\mathtt{CP}_{i,1}, \mathtt{CP}_{i,2}\}^d_{i=1}$ look like random numbers.

• $\mathcal{A}$ cannot distinguish $\{\{\mathtt{PT}^{\beta}_i\}^d_{i=1}\}^{p_2}_{\beta=1}$ from random numbers. This is because $\{\mathtt{PT}^{\beta}_i\}^d_{i=1}$ contains random numbers $\{\mathbf{R}_{\mathcal{Q}_\beta,i}, r_{q_\beta}, s_i\}^d_{i=1}$, which make $\{\{\mathtt{PT}^{\beta}_i\}^d_{i=1}\}^{p_2}_{\beta=1}$ look like random numbers.

Thus, $\mathcal{A}$ cannot distinguish the views of real and ideal worlds, and the PIPE scheme is selectively secure. □

7.2 Security of RIPE Scheme

Theorem 2. The RIPE scheme is also selectively secure.

Proof: The proof of RIPE's selective security is the same as that of the PIPE's selective security, and the details are omitted here. □

7.3 Security of PRQ Scheme

In this subsection, we analyze the security of the PRQ scheme. We show that the dataset, range query requests as well as query results are privacy-preserving and can be kept from the cloud server. Meanwhile, we show that the single-dimensional privacy of the query results can be preserved.

• *The dataset is privacy-preserving.* In the PRQ scheme, the cloud server is assumed to be *honest-but-curious*, so it may be curious about the plaintext of the dataset. However, when running the PRQ scheme, the cloud server can only access the encrypted R-tree $E(\mathrm{T})$. On the one hand, the internal nodes and leaf nodes in $E(\mathrm{T})$ have been respectively encrypted by the PIPE scheme, RIPE scheme and AES algorithm. Since the PIPE scheme, RIPE scheme, and AES algorithm are secure, the cloud server cannot recover the underlying points of $E(\mathrm{T})$'s leaf nodes and the underlying MBRs of $E(\mathrm{T})$'s internal nodes. On the other hand, since the child nodes of each internal node in $E(\mathrm{T})$ have been permutated, the cloud server cannot obtain the information about the plaintext dataset from the structure of $E(\mathrm{T})$. Thus, the cloud server cannot obtain the plaintext dataset from the encrypted R-tree $E(\mathrm{T})$. In addition, the cloud server may attempt to infer the plaintext dataset while performing range queries over $E(\mathrm{T})$. However, when the cloud server performs queries over the internal nodes and leaf nodes, the security of the PIPE scheme and RIPE scheme guarantees that the cloud server can only obtain (i) whether the data points in leaf nodes are in the query range; and (ii) whether the ranges in internal nodes intersect with the query range. Thus, the cloud server cannot obtain the plaintext dataset during the process of query processing. Therefore, the dataset is privacy-preserving.

• *The query requests are privacy-preserving.* The query requests should be kept from the cloud server. On the one hand, the query range $\mathcal{Q}$ has been encrypted into two tokens $\{\mathtt{PT}_i\}^d_{i=1}$ and $\{\mathtt{RT}_{l,i}, \mathtt{RT}_{u,i}\}^d_{i=1}$, and the security of the PIPE scheme and RIPE scheme can guarantee that the cloud server cannot infer the plaintext query range from the tokens $\{\mathtt{PT}_i\}^d_{i=1}$ and $\{\mathtt{RT}_{l,i}, \mathtt{RT}_{u,i}\}^d_{i=1}$. On the other hand, the tokens $\{\mathtt{PT}_i\}^d_{i=1}$ and $\{\mathtt{RT}_{l,i}, \mathtt{RT}_{u,i}\}^d_{i=1}$ involve random numbers $\{r_q, r'_q\}$ and random matrices $\{\mathbf{R}_{\mathcal{Q},i}, \mathbf{R}_{\mathcal{Q},l,i}, \mathbf{R}_{\mathcal{Q},u,i}\}$. These random numbers and random matrices guarantee that the pattern of the query ranges is hidden, i.e., which queries refer to the same query range. In other words, one query range $\mathcal{Q}$ will have two different pairs of point/range tokens when it is encrypted twice. Therefore, the query requests are privacy-preserving.

• *The query results are privacy-preserving.* The query results should be kept from the cloud server. The query results are ciphertexts of data points $\mathsf{AES}_K(\mathbf{x})$ that are encrypted by the AES algorithm. Since the security of AES algorithm can guarantee that the cloud server cannot infer the plaintext data points from the corresponding ciphertexts, the query results are also privacy-preserving.

• *The single-dimensional privacy of the PRQ scheme can be preserved.* The basic operations of query processing in the PRQ scheme are point intersection query and range intersection query. To preserve the single-dimensional privacy, we respectively choose random numbers $\{s_i\}^d_{i=1}$ and $\{s_{l,i}, s_{u,i}\}^d_{i=1}$ for query range $\tilde{\mathbf{Q}}_i$ and $\{\tilde{\mathbf{Q}}_{u,i}, \tilde{\mathbf{Q}}_{l,i}\}$. In this

case, only the cloud server processes the queries using all dimensions, can it remove the random numbers $\{s_i\}_{i=1}^{d}$ and $\{s_{l,i}, s_{u,i}\}_{i=1}^{d}$ and get the correct results. In this case, the cloud server has no idea on the query result for each dimension of the query range. Thus, the single-dimensional privacy of the PRQ scheme can be preserved.

8 Performance Evaluation

In this section, we evaluate the performance of our PRQ scheme and compare it with other privacy-preserving multi-dimensional range query schemes.

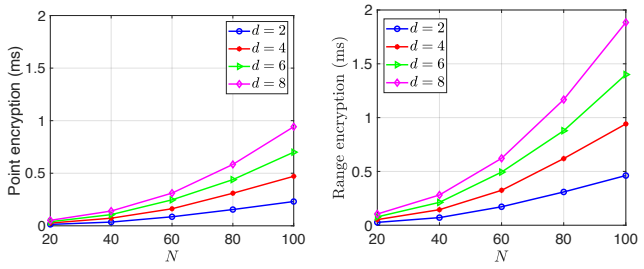
8.1 Performance of PRQ Scheme

We evaluate the computational cost of our PRQ scheme from the perspective of dataset encryption, query token generation, range query processing, and query result recovery.

Experiment Setting: We implement our PRQ scheme and the compared schemes in Java and conduct experiments on a machine with an Intel(R) Core(TM) i7-3770 CPU @3.40GHz, 16GB RAM and Windows 10 operating system. The dataset we evaluate is a part of US Census Data (1990) [24]. For the R-tree, the number of child nodes for each node is from 2 to 8. For the AES algorithm, we set the length of the encryption key K to 256 bits. Meanwhile, each experiment is conducted 10000 times, and the average result is reported.

8.1.1 Dataset Encryption

The dataset encryption algorithm is to build and encrypt an R-tree for the dataset $\mathcal{X}$, and the major computational cost is from the dataset encryption. In the dataset encryption, there are two types of basic encryption operations, i.e., point encryption and range encryption. Encrypting one point takes $O(d * N^2)$ computational cost and encrypting one range also takes $O(d * N^2)$, where d is the number of dimensions of $\mathcal{X}$ and N is the square root of the upper bound of $\mathcal{X}$'s values. Besides N and d, the computational cost of dataset encryption is affected by the size of the dataset, i.e., $|\mathcal{X}|$. Next, we first show how the computational cost of the point encryption and range encryption varies with N and d. Then, we evaluate how the computational cost of dataset encryption varying with $|\mathcal{X}|$, N and d.

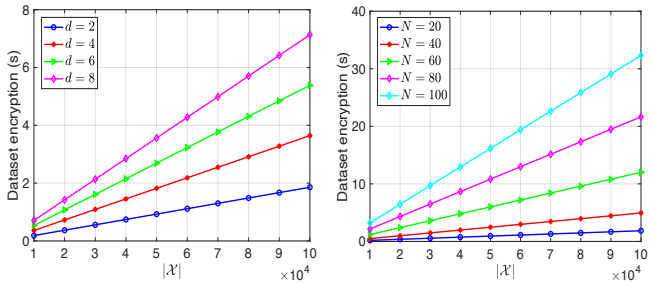


(a) Point encryption (b) Range encryption

Fig. 3. Computational cost of point encryption and range encryption

• *Point encryption and range encryption:* In Fig. 3(a), we plot the computational cost of the point encryption varying with N and d. In this experiment, we set $N = \{20, 40, 60, 80, 100\}$ and $d \in \{2, 4, 6, 8\}$. From this figure, we can see that the computational cost of the point encryption linearly increases with d and quadratically increases with N. In Fig. 3(b), we plot the computational cost of the range encryption varying with N and d. In this experiment, we also set $N = \{20, 40, 60, 80\}$ and $d \in \{2, 4, 6, 8\}$. We can see that the computational cost of the range encryption linearly increases with d and quadratically increases with N.

• *Dataset encryption:* In Fig. 4(a), we plot the computational cost of dataset encryption varying with $|\mathcal{X}|$ and d. In this experiment, we set $|\mathcal{X}| = \{10\text{K}, 20\text{K}, \cdots, 100\text{K}\}$, $N = 20$, and $d = \{2, 4, 6, 8\}$. This figure shows that the computational cost of dataset encryption linearly increases with $|\mathcal{X}|$ and linearly increases with d. In Fig. 4(b), we plot the computational cost of dataset encryption varying with $|\mathcal{X}|$ and N. In this experiment, we set $|\mathcal{X}| = \{10\text{K}, 20\text{K}, \cdots, 100\text{K}\}$, $N = \{20, 40, 60, 80, 100\}$, and $d = 2$. This figure shows that the computational cost of dataset encryption linearly increases with $|\mathcal{X}|$ and quadratically increases with N.



(a) Cost with $|\mathcal{X}|$ and d (b) Cost with $|\mathcal{X}|$ and N

Fig. 4. Computational cost of dataset encryption

8.1.2 Query Token Generation

The query token generation algorithm generates a point token and a range token for the query range, and the corresponding computational cost is $O(d * N^3)$. In Fig. 5, we plot the computational cost of the token generation varies with d and N. In this experiment, we set $d \in \{2, 4, 6, 8\}$ and $N \in \{20, 40, 60, 80, 100\}$. From this figure, we can see that the computational cost linearly grows with d. When fixing d, the computational cost cubically grows with N.

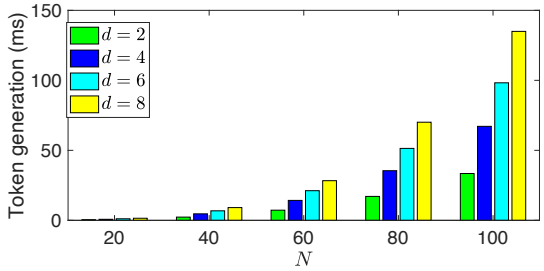


Fig. 5. Computational cost of token generation versus d and N

8.1.3 Range Query Processing

The range query processing algorithm is to search for the query result over the encrypted dataset $E(\mathcal{X})$. The basic operations of the range query include (i) point intersection query, i.e., determine whether $\mathbf{x} \stackrel{?}{\in} \mathcal{Q}$; and (ii) range

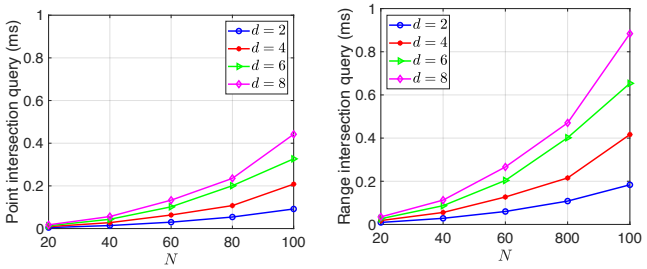
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2021.3101120, IEEE Transactions on Dependable and Secure Computing

TABLE 2
Comparison between our PRQ scheme and existing schemes

Scheme	Asymptotic search complexity	Basic operation	Query privacy	Single-dimensional privacy	Number of cloud servers
Boneh et al's scheme [9]	$O(\|\mathcal{X}\| * d)$	Bilinear pairing	×	√	Single
Shi et al. scheme [10]	$O(\|\mathcal{X}\| * (\log N^2)^d)$	Bilinear pairing	√	√	Single
Maple [11]	$O(\log \|\mathcal{X}\| * d)$	Bilinear pairing	×	√	Single
LSED+ [12]	$O(\log \|\mathcal{X}\| * (\log N^2)^2 * d)$	Bilinear pairing	√	×	Single
Wang et al. scheme [13]	$O(\log \|\mathcal{X}\| * d)$	Multiplication over $\mathbb{R}$	√	×	Single
Mei et al. scheme [15]	$O(\log \|\mathcal{X}\| * d)$	Multiplication over $\mathbb{R}$	√	×	Single
TRQED [16]	$O(\log \|\mathcal{X}\| * d)$	Multiplication over $\mathbb{R}$	√	×	Single
TRQED$^+$ [16]	$O(\log \|\mathcal{X}\| * d)$	Multiplication over $\mathbb{R}$	√	√	Two
Our PRQ	$O(\log \|\mathcal{X}\| * d * N^2)$	Multiplication over $\mathbb{R}$	√	√	Single

intersection query, i.e., determine whether $\mathcal{B} \cap \mathcal{Q} \stackrel{?}{=} \emptyset$. The computational cost of the point intersection query and range intersection query is $O(d * N^2)$. In addition, besides N and d, the computational cost of range query over the encrypted dataset is related to the size of the dataset, i.e., $|\mathcal{X}|$. Specifically, the computational complexity of range query is $O(d * N^2 * \log |\mathcal{X}|)$. In the following, we show how the computational cost of point and range intersection query varies with d and N. Then, we show how the computational cost of range query over dataset varies with $|\mathcal{X}|$, N and d.

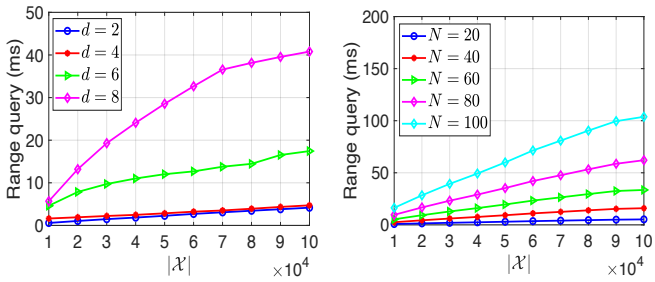
• *Point and range intersection query:* In Fig. 6(a), we plot the computational cost of the point intersection query varies with d and N. In this experiment, we set $d = \{2, 4, 6, 8\}$ and $N \in \{20, 40, 60, 80, 100\}$. From this figure, we can see that the computational cost of the point query linearly grows with d and quadratically grows with N. In Fig. 6(b), we plot the computational cost of the range intersection query varies with d and N. In this experiment, we set $d = \{2, 4, 6, 8\}$ and $N \in \{20, 40, 60, 80, 100\}$. This figure also shows that the computational cost of the range query also linearly grows with d and quadratically grows with N.



(a) Point intersection query (b) Range intersection query

Fig. 6. Computational cost of point and range intersection query

• *Range query over the dataset:* In Fig. 7(a), we plot the computational cost of range query processing varies with $|\mathcal{X}|$ and d. In this experiment, we set $|\mathcal{X}| = \{10\text{K}, 20\text{K}, \cdots, 100\text{K}\}$, $d = \{2, 4, 6, 8\}$ and $N = 20$. For each range query, there are 10 to 20 data records in the dataset that satisfy the query condition. From this figure, we can see that the computational cost of range queries logarithmically increases with $|\mathcal{X}|$ and also increases with d. In Fig. 7(b), we plot the computational cost of range query processing varies with $|\mathcal{X}|$ and N. In this experiment, we set $|\mathcal{X}| = \{10\text{K}, 20\text{K}, \cdots, 100\text{K}\}$, $d = 2$ and $N = \{20, 40, 60, 80, 100\}$. The query result of each query also contains 10 to 20 data records. This figure shows that the computational cost of range queries logarithmically increases with $|\mathcal{X}|$ and quadratically increases with N.



(a) Cost with $|\mathcal{X}|$ and d (b) Cost with $|\mathcal{X}|$ and N

Fig. 7. Computational cost of range query over the encrypted dataset

8.1.4 Query Result Recovery

In the PRQ scheme, the query results are encrypted by the AES algorithm. Thus, the computational cost of the query result recovery depends on the decryption algorithm. For a d-dimensional point, the decryption cost is about 3.96 μs, where $d \in \{2, 4, 6, 8\}$. Thus, the query result recovery is pretty efficient.

8.2 Performance Comparison

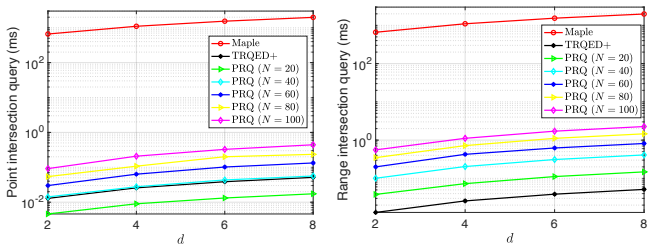
We compare our PRQ scheme with existing privacy-preserving multi-dimensional range query schemes. Since the security of the OPE-based schemes [5], [6] is weak. The bucketization based range query schemes [7], [8] impose most of the computational cost on query users, and the corresponding query results contain false positive data. Thus, we do not take these schemes into comparison. As shown in TABLE 2, we compare our scheme with other schemes from five aspects, i.e., (i) asymptotic search complexity; (ii) basic operation when calculating the asymptotic search complexity; (iii) query privacy; (iv) single-dimensional privacy; and (v) the number of deployed cloud servers. TABLE 2 shows that Boneh et al.'s scheme in [9], Shi et al.'s scheme in [10], Maple scheme in [11] and LSED+ scheme in [12] are inefficient in range queries because they are designed based on the public key cryptography and their basic operations are bilinear pairing. Especially, the computational cost of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2021.3101120, IEEE Transactions on Dependable and Secure Computing

Boneh et al.'s scheme in [9] and Shi et al.'s scheme in [10] is linear to the size of the dataset. Meanwhile, Boneh et al.'s scheme in [9] and Maple scheme in [11] cannot preserve the query privacy. Although the computational cost of Wang et al.'s scheme in [13], Mei et al.'s scheme in [15], and TRQED scheme in [16] logarithmically increases with the size of the dataset, they fail to preserve the single-dimensional privacy. Thus, TRQED$^+$ scheme in [16] and our PRQ scheme are the only two schemes with logarithmic search efficiency, query privacy and single-dimensional privacy simultaneously. However, the TRQED$^+$ scheme is built in the two-server setting, which is impractical in some real scenarios. Thus, our PRQ scheme is the only privacy-preserving range query scheme with efficient query processing, query privacy, single-dimensional privacy in the single-server setting.

To validate the query efficiency of our PRQ scheme, we compare its range query efficiency with that of existing schemes. Among existing schemes, Maple scheme [11] and TRQED$^+$ scheme can be regarded as two representatives, where Maple scheme [11] was designed based on the public key cryptography and TRQED$^+$ scheme [16] was designed based on symmetric matrix encryption scheme. Meanwhile, TRQED$^+$ scheme [16] is the most state-of-the-art scheme.

Maple scheme [11] was designed based on the bilinear pairing. In our experiment, we implement its bilinear pairing operations by JPBC Pairing-Based Cryptography Library and set the security parameter to 512 bits, i.e., $\kappa = 512$. TRQED$^+$ scheme [16] was designed using matrix encryption and permutation technique. Although TRQED$^+$ scheme [16] is able to preserve the privacy of access pattern, such function must be supported by a homomorphic encryption based flag label, which will incur additional cost. For a fair comparison, we compare our PRQ scheme with the TRQED$^+$ scheme without flag labels. Since Maple scheme [11] and TRQED$^+$ scheme [16] also employ R-tree to represent datasets and reduce the problem of multi-dimensional range queries into that of point intersection and range intersection, we experimentally compare our PRQ scheme with Maple scheme and TRQED$^+$ from the aspects of point intersection query, range intersection query, and range query processing.



(a) Point intersection query (b) Range intersection query

Fig. 8. Comparison of point and range intersection queries

• *Point and range intersection queries comparison:* In Fig. 8(a) and Fig. 8(b), we respectively plot the computational cost of processing one point intersection query and one range intersection query varying with d among the Maple scheme [11], TRQED$^+$ scheme and our PRQ scheme. In both experiments, the parameter d ranges from 2 to 8 and the computational cost of our PRQ scheme is evaluated under different N values, i.e., $N = \{20, 40, 60, 80, 100\}$. These two figures show that the point intersection and range intersection queries of our PRQ scheme and the TRQED$^+$ scheme are much more efficient than the public key cryptography based Maple scheme. Although the query efficiency of our scheme is a little lower than that of the TRQED$^+$ scheme, it is also pretty efficient. For example, when $N = 100$ and $d = 8$, processing one point intersection query and one range intersection query only takes about 0.442 ms and 2.264 ms, respectively.

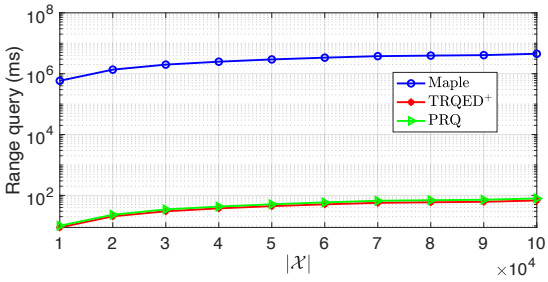


Fig. 9. Computational cost of range query processing versus $|\mathcal{X}|$

• *Range queries comparison:* In Fig. 9, we plot the computational cost of range query processing among the Maple scheme [11], TRQED$^+$ scheme, and our PRQ scheme. In this experiment, we set the parameters as $d = 8$, $|\mathcal{X}| = \{10\text{K}, 20\text{K}, 30\text{K}, 40\text{K}, 50\text{K}\}$, and $N = 20$. For each range query, there are 10 to 20 data records in the dataset that satisfies the query condition. This figure shows that the range query efficiency of our PRQ scheme and the TRQED$^+$ scheme is much more efficient than the public key cryptography based Maple scheme. Although the query efficiency of our scheme is a little lower than that of TRQED$^+$ scheme, it is also efficient, e.g., when $|\mathcal{X}| = 100\text{K}$, processing one range query only takes about 70 ms.

In addition, our scheme deals with range queries in a single server without any communication cost while TRQED$^+$ scheme does range queries by two servers, which requires communication between them. As shown in TABLE 3, we experimentally quantify the communication cost of each range query in TRQED$^+$ scheme. In this experiment, we set $d = 8$, $|\mathcal{X}| = \{60\text{K}, 80\text{K}, 100\text{K}\}$. For each range query, there are 10 to 20 data records in the dataset that satisfies the query condition. From TABLE 3, we can see that the communication cost of TRQED$^+$ scheme increases with the size of the dataset $|\mathcal{X}|$. When $|\mathcal{X}| = 100\text{K}$, the communication cost is about 1.84 MB.

TABLE 3
Communication cost comparison of each range query

	$\|\mathcal{X}\| = 60\text{K}$	$\|\mathcal{X}\| = 80\text{K}$	$\|\mathcal{X}\| = 100\text{K}$
Our Scheme	0 MB	0 MB	0 MB
TRQED$^+$ scheme	1.37 MB	1.60 MB	1.84 MB

9 Related Work

Privacy-preserving multi-dimensional range queries over encrypted data have been extensively studied in the literature and various schemes were proposed.

The order-preserving encryption (OPE) works [5], [6] can intrinsically support the range queries because the order of the plaintext data can be preserved in the ciphertexts. However, as shown in [25], the OPE technique is not secure under the ordered chosen plaintext attack. Thus, the security of the OPE-based range query schemes is weak because they will leak the order information of the plaintext data.

To prevent from leaking the order information, the bucketization based range query schemes were proposed in [7], [8]. In such schemes, the plaintext space is divided into buckets such that the order information of data in the same bucket is preserved. Specifically, Lee [7] proposed a secure encryption scheme with ordered bucketization to support range queries but it can only support range queries for single-dimensional data. Hore et al. [8] introduced a bucketization based range query scheme, and they designed an optimization algorithm to adjust the trade-off between the computational cost of query processing and information leakage. However, both schemes in [7], [8] have two limitations. First, most of the computational costs in range queries are imposed on query users who are usually considered to be resource-constrained. Second, the query results contain some false positive records, so the query users have to take additional computational cost to filter out the correct results.

Some privacy-preserving multi-dimensional range query schemes were proposed based on the public key cryptography [9]–[11]. Specifically, Boneh et al. [9] presented a privacy-preserving multi-dimensional range query scheme based on the public key cryptography based hidden vector encryption (HVE). Shi et al. [10] decomposed multi-dimensional range queries to several single-dimensional range queries. Then, they employed the interval tree as index and bilinear pairing based encryption algorithm to design a privacy-preserving multi-dimensional range query scheme. Wang et al. [11] proposed a privacy-preserving multi-dimensional range query scheme based on the public key cryptography based hidden vector encryption technique. However, since the public key cryptography involves high computational cost, the above schemes are inefficient.

To improve the query efficiency, some efficient multi-dimensional range query schemes were proposed [12]–[15]. Lu et al. [12] deployed B+-tree structure and the symmetric key encryption scheme to design a range query scheme for the single-dimensional data. Since multi-dimensional range queries can be decomposed into multiple single range queries, the proposed scheme in [12] can be extended to support multi-dimensional range queries. The schemes presented in [13], [14] also decomposed the multi-dimensional range queries into multiple single-dimensional range queries. Further, they were proposed by using the R-tree as the index structure and a kind of matrix encryption technique (i.e., asymmetric privacy-preserving encryption scheme or its variant) as the encryption algorithm. However, range query decomposing results in the leakage of the single-dimensional privacy, i.e., the records satisfying each single dimension of the range query will be leaked. Although the scheme presented in [14] attempted to solve the single-dimensional privacy leakage by padding additional dimensions, it still cannot fully protect the single-dimensional privacy especially after the sever processes a large number of range queries. To fully preserve the single-dimensional privacy, Yang et al. [16] proposed a new range query scheme based on [14]. The proposed scheme can not only preserve the single-dimensional privacy but also preserve the access pattern on the R-tree. However, the proposed scheme was constructed under the non-colluding two-server model. Compared with a single-server model, the two-serve model is less practical due to its limitations in the non-colluding assumption and additional communication cost required between the two servers. Mei et al. [15] proposed a flexible multi-dimensional range query scheme based on an interval tree structure, which still suffers from the problem of single-dimensional privacy leakage. Meanwhile, this scheme is impractical because the ranges that a query user can query must be predefined.

10 Conclusion

In this paper, we have proposed a practical and privacy-preserving multi-dimensional range query scheme over encrypted data. In our scheme, we indexed the multi-dimensional dataset to an R-tree and the R-tree based multi-dimensional range query is reduced to the operations of point intersection and range intersection. Then, we delicately designed a PIPE scheme and a RIPE scheme such that we can privately determine whether a point is in the query range or whether two ranges have an intersection through their ciphertexts. Furthermore, we proposed our PRQ scheme based on the PIPE scheme and RIPE scheme. Meanwhile, we analyzed the security of our PRQ scheme and conducted experiments to validate its efficiency. In our future work, we plan to design more efficient multi-dimensional range query scheme over encrypted data.

Acknowledgements

This research was supported in part by NSERC Discovery Grants (04009), Natural Science Foundation of Shaanxi Province (2019ZDLGY12-02), ZJNSF (LZ18F020003), and NSFC (U1709217, 61972304).

References

[1] R. Lu, "A new communication-efficient privacy-preserving range query scheme in fog-enhanced iot," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2497–2505, 2019.

[2] A. Guidotti, S. Cioni, G. Colavolpe, and M. C. et al., "Architectures, standardisation, and procedures for 5g satellite communications: A survey," *Computer Networks*, p. 107588, 2020.

[3] Y. Zheng, R. Lu, Y. Guan, J. Shao, and H. Zhu, "Achieving efficient and privacy-preserving exact set similarity search over encrypted data," *IEEE Transactions on Dependable and Secure Computing*, 2020.

[4] Y. Guan, R. Lu, Y. Zheng, J. Shao, and G. Wei, "Toward oblivious location-based k-nearest neighbor query in smart cities," *IEEE Internet of Things Journal*, pp. 1–1, 2021.

[5] A. Boldyreva, N. Chenette, Y. Lee, and A. O'Neill, "Order-preserving symmetric encryption," in *EUROCRYPT 2009*, ser. Lecture Notes in Computer Science, vol. 5479, 2009, pp. 224–241.

[6] A. Boldyreva, N. Chenette, and A. O'Neill, "Order-preserving encryption revisited: Improved security analysis and alternative solutions," in *CRYPTO 2011*, ser. Lecture Notes in Computer Science, vol. 6841, 2011, pp. 578–595.

[7] Y. Lee, "Secure ordered bucketization," *IEEE Trans. Dependable Secur. Comput.*, vol. 11, no. 3, pp. 292–303, 2014.

[8] B. Hore, S. Mehrotra, M. Canim, and M. Kantarcioglu, "Secure multidimensional range queries over outsourced data," *VLDB J.*, vol. 21, no. 3, pp. 333–358, 2012.

[9] D. Boneh and B. Waters, "Conjunctive, subset, and range queries on encrypted data," in *Theory of Cryptography, 4th Theory of Cryptography Conference, TCC 2007*, ser. Lecture Notes in Computer Science, vol. 4392, 2007, pp. 535–554.

[10] E. Shi, J. Bethencourt, T. H. Chan, D. X. Song, and A. Perrig, "Multi-dimensional range query over encrypted data," in *(S&P 2007)*. IEEE Computer Society, 2007, pp. 350–364.

[11] B. Wang, Y. Hou, M. Li, H. Wang, and H. Li, "Maple: scalable multi-dimensional range search over encrypted cloud data with tree-based index," in *Proceedings of the 9th ACM symposium on Information, computer and communications security*, 2014, pp. 111–122.

[12] Y. Lu, "Privacy-preserving logarithmic-time search on encrypted data in cloud," in *NDSS 2012*. The Internet Society, 2012.

[13] P. Wang and C. V. Ravishankar, "Secure and efficient range queries on outsourced databases using rp-trees," in *ICDE 2013*, 2013, pp. 314–325.

[14] W. Yang, Y. Xu, Y. Nie, Y. Shen, and L. Huang, "TRQED: secure and fast tree-based private range queries over encrypted cloud," in *DASFAA 2018*, ser. Lecture Notes in Computer Science, vol. 10828, 2018, pp. 130–146.

[15] Z. Mei, H. Zhu, Z. Cui, Z. Wu, G. Peng, B. Wu, and C. Zhang, "Executing multi-dimensional range query efficiently and flexibly over outsourced ciphertexts in the cloud," *Inf. Sci.*, vol. 432, pp. 79–96, 2018.

[16] W. Yang, Y. Geng, L. Li, X. Xie, and L. Huang, "Achieving secure and dynamic range queries over encrypted cloud data," *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[17] A. Guttman, "R-trees: A dynamic index structure for spatial searching," in *SIGMOD 1984*, B. Yormark, Ed. ACM Press, 1984, pp. 47–57.

[18] D. Boneh, A. Sahai, and B. Waters, "Fully collusion resistant traitor tracing with short ciphertexts and private keys," in *EUROCRYPT 2006*, ser. Lecture Notes in Computer Science, vol. 4004, 2006, pp. 573–592.

[19] S. Sprenger, P. Schäfer, and U. Leser, "Bb-tree: A practical and efficient main-memory index structure for multidimensional workloads," in *EDBT 2019*, 2019, pp. 169–180.

[20] W. K. Wong, D. W. Cheung, B. Kao, and N. Mamoulis, "Secure knn computation on encrypted databases," in *SIGMOD*, 2009, pp. 139–152.

[21] P. Ananth, Z. Brakerski, G. Segev, and V. Vaikuntanathan, "From selective to adaptive security in functional encryption," in *CRYPTO 2015*, ser. Lecture Notes in Computer Science, vol. 9216, 2015, pp. 657–677.

[22] Y. Zheng, R. Lu, Y. Guan, J. Shao, and H. Zhu, "Achieving efficient and privacy-preserving set containment search over encrypted data," *IEEE Transactions on Services Computing*, pp. 1–1, 2021.

[23] ——, "Efficient and privacy-preserving similarity range query over encrypted time series data," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–1, 2021.

[24] "UCI Machine Learning Repository: US Census Data (1990) Data Set," Available: https://archive.ics.uci.edu/ml/datasets/US+Census+Data+%281990%29.

[25] L. Xiao and I. Yen, "A note for the ideal order-preserving encryption object and generalized order-preserving encryption," *IACR Cryptol. ePrint Arch.*, vol. 2012, p. 350, 2012.

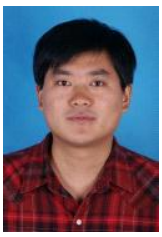
Yandong Zheng received her M.S. degree from the Department of Computer Science, Beihang University, China, in 2017 and she is currently pursuing her Ph.D. degree in the Faculty of Computer Science, University of New Brunswick, Canada. Her research interest includes cloud computing security, big data privacy and applied privacy.

Rongxing Lu (S'09-M'11-SM'15-F'21) is an associate professor at the Faculty of Computer Science (FCS), University of New Brunswick (UNB), Canada. Before that, he worked as an assistant professor at the School of Electrical and Electronic Engineering, Nanyang Technological University (NTU), Singapore from April 2013 to August 2016. Rongxing Lu worked as a Postdoctoral Fellow at the University of Waterloo from May 2012 to April 2013. He was awarded the most prestigious "Governor General's Gold Medal", when he received his PhD degree from the Department of Electrical & Computer Engineering, University of Waterloo, Canada, in 2012; and won the 8th IEEE Communications Society (ComSoc) Asia Pacific (AP) Outstanding Young Researcher Award, in 2013. Also, Dr. Lu received his first PhD degree at Shanghai Jiao Tong University, China, in 2006. Dr. Lu is an IEEE Fellow. His research interests include applied cryptography, privacy enhancing technologies, and IoT-Big Data security and privacy. He has published extensively in his areas of expertise (with H-index 77 from Google Scholar as of July 2021), and was the recipient of 9 best (student) paper awards from some reputable journals and conferences. Currently, Dr. Lu serves as the Vice-Chair (Conferences) of IEEE ComSoc CIS-TC (Communications and Information Security Technical Committee). Dr. Lu is the Winner of 2016-17 Excellence in Teaching Award, FCS, UNB.

Yunguo Guan is a PhD student of the Faculty of Computer Science, University of New Brunswick, Canada. His research interests include applied cryptography and game theory.

Jun Shao received the Ph.D. degree from the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China, in 2008.

He was a Post-Doctoral Fellow with the School of Information Sciences and Technology, Pennsylvania State University, Pennsylvania, PA, USA, from 2008 to 2010. He is currently a Professor with the School of Computer and Information Engineering, Zhejiang Gongshang University, Hangzhou, China. His current research interests include network security and applied cryptography.



Hui Zhu (M'13-SM'19) received the B.Sc. degree from Xidian University, Xian, China, in 2003, the M.Sc. degree from Wuhan University, Wuhan, China, in 2005, and the Ph.D. degree from Xidian University, in 2009.

He was a Research Fellow with the School of Electrical and Electronics Engineering, Nanyang Technological University, Singapore, in 2013. Since 2016, he has been a Professor with the School of Cyber Engineering, Xidian University. His current research interests include applied cryptography, data security, and privacy.