PPSKSQ: Towards Efficient and Privacy-Preserving Spatial Keyword Similarity Query in Cloud

Changrui Wang, Lei Wu*, Lijuan Xu, Haojie Yuan, Hao Wang, *Member, IEEE*, Wenying Zhang and Weizhi Meng, *Senior Member, IEEE*

***Abstract*—The growth of cloud computing has led to the widespread use of location-based services, such as spatial keyword queries, which return spatial data points within a given range that have the highest similarity in keyword sets to the user's. As the volume of spatial data increases, providers commonly outsource data to powerful cloud servers. Because cloud servers are untrustworthy, privacy-preserving keyword query schemes have been proposed. However, existing schemes consider only location queries or exact keyword matching. To address these issues, we propose the Privacy-Preserving Spatial Keyword Similarity Query Scheme (PPSKSQ), designed to search for spatial data points with the highest similarity while protecting the privacy of outsourced data, query requests, and results. First, we design two sub-protocols based on improved symmetric homomorphic encryption (iSHE): iSHE-SC for secure size comparison and iSHE-SIP for secure inner product computation. Then, we encode range information and integrate it with a quadtree to construct a novel index structure. Additionally, we use the Jaccard to measure similarity in conjunction with the iSHE-SC protocol, transforming similarity comparison into a matrix trace operation. Finally, rigorous security analysis and extensive simulation experiments confirm the flexibility, efficiency, and scalability of our scheme.**

***Index Terms*—Cloud computing, rectangular range query, jaccard similarity query, privacy preservation, symmetric homomorphic encryption.**

I. Introduction

In the era of ubiquitous big data, the widespread adoption of cloud computing has revolutionized everyday life by offering significant convenience and enabling efficient computation. This has also driven research into spatial keyword querying, a versatile and flexible method for matching keywords within a specified range. Early studies in this field focused on location-based services (LBSs) [1], [2], which later expanded into various applications such as mobile crowdsensing [3], online medical diagnosis [4], VANETs [5], intelligent transportation [6], social networking [7], and trajectory services [8]. In LBSs research, the primary focus is on recommendation systems, such as taxi services, where users look for Points of Interest (POIs) that meet their particular needs. This has led to growing attention to spatial keyword querying as an emerging research area.

Existing spatial keyword query methods primarily focus on matching all user keywords, which limits the general applicability of the service to some extent. Previous studies have proposed several solutions to spatial keyword queries, such as [9]–[11]. However, these approaches predominantly emphasize exact keyword matching. For example, in these schemes, if the set of keywords is $\{$*swimming, music, supermarket, hotel*$\}$ and the user query set is $\{$*swimming, supermarket, stadium*$\}$, the system only returns results when all the query keywords exactly match the existing set. In practice, user queries are often dynamic, and users typically prefer to receive results when only some of the keywords match. Therefore, enabling keyword similarity queries has become a challenging problem that requires a solution.

Keyword similarity queries aim to enable approximate matching of keyword information. However, as user privacy concerns are increasingly prominent, achieving spatial location and similarity queries in a ciphertext environment represents a major challenge in the field. Several privacy-preserving schemes have been proposed to address this problem, such as [12]–[18]. Nonetheless, these schemes have certain limitations: 1) [12], [14] only support single-location queries, which lack flexibility and are challenging to adapt to diverse real-world application scenarios; 2) [15], [17], [18] although combining spatial and textual information, only support Boolean queries, which require precise keyword matching and cannot satisfy the demand for approximate matching; 3) [16] despite considering the similarity between keywords, uses Euclidean distance as a similarity metric, which is more suitable for fixed-length keyword comparisons. Therefore, to address the above challenges, we propose a solution for privacy-preserving spatial keyword similarity queries among dynamic keyword sets, aiming to provide users with a more flexible and efficient approach.

Since range filtering and similarity comparison in a ciphertext environment is computationally intensive, we use a dual-cloud environment to implement range query and Jaccard similarity comparison to achieve privacy preservation. Recently, many privacy-preserving dual-cloud schemes [19]–[23] have been proposed. As our scheme performs range queries and Jaccard similarity metrics in a ciphertext environment, dual-cloud allows our scheme to support more computations and

This work was supported in part by the National Natural Science Foundation of China under Grant 62472266, Grant 62472265, Grant 62302280, and Grant 62172258, in part by the Key Laboratory of Computing Power Network and Information Security, Ministry of Education, under Grant 2023ZD021.

Changrui Wang, Lei Wu, Haojie Yuan, Hao Wang and Wenying Zhang are with the School of Information Science and Engineering, Shandong Normal University, Jinan 250358, China (E-mail: wangchangrui0705@163.com; wulei@sdnu.edu.cn; yuanhaojie7218@163.com; wanghao@sdnu.edu.cn; zhangwenying@sdnu.edu.cn).

Lijuan Xu, is with Key Laboratory of Computing Power Network and Information Security, Ministry of Education, Qilu University of Technology (Shandong Academy of Sciences) (e-mail: xulj@sdas.org).

Weizhi Meng is a Full Professor in the School of Computing and Communications, Lancaster University, United Kingdom(e-mail: weizhi.meng@ieee.org).

* Corresponding author.

greatly improves the efficiency of queries and comparisons. How to efficiently implement range queries and Jaccard similarity metrics in a dual-cloud system will surely be a major challenge in our work. In addition, exact keyword queries only need to determine whether the keywords are equal or not, whereas similarity queries based on Jaccard metrics necessitate inter-set comparisons and fractional computations. This computational complexity represents a significant challenge in an outsourced ciphertext cloud environment. Currently, there is no generalized solution for similarity queries, which further compounds the challenge of developing a highly scalable scheme. Meanwhile, similarity computation inevitably incurs efficiency and communication overhead. Therefore, designing a solution that balances flexibility, high performance, and privacy preservation remains our key challenge.

In this paper, we propose a Privacy-Preserving Spatial Keyword Similarity Query scheme (PPSKSQ), which aims to identify spatial location points that conform to the location range and exhibit the highest similarity. The main idea is to transform the similarity into a matrix trace and thus operate on it using a lightweight matrix that we have designed. We found that the operation on the matrix trace can satisfy the two encryption protocols we designed, and finally obtain the encryption flag so that the similarity can be judged. Specifically, our contributions are fourfold:

- First, we design two privacy-preserving protocols based on lightweight improved symmetric homomorphic encryption (iSHE): iSHE-SC and iSHE-SIP. When filtering sets of keywords for the highest similarity, the iSHE-SC protocol can securely compare the size of similarity values among keyword sets. During range queries, both location and range data are encoded into ciphertext vectors, allowing the iSHE-SIP protocol, which securely computes inner products, to determine intersections between range-to-range and point-to-range queries.
- Second, we secure the encoded keyword sets using lightweight matrix encryption. Furthermore, by leveraging the properties of matrix encryption, we transform the keyword information into the trace of a matrix. This approach enables operation on the matrix trace using the iSHE-SC protocol to perform Jaccard similarity comparisons, significantly reducing computational costs.
- Third, we leverage quadtree properties to construct a tree-based index that encodes spatial location points and their segmented rectangular ranges and stores them within the index. Consequently, the iSHE-SIP protocol is employed to efficiently filter the entire two-dimensional space during range queries. The PPSKSQ scheme enables the selection of appropriate quadtree division hierarchies for various densities of location points on a map within the same area size, to achieve optimal efficiency in the outsourcing and querying phases.
- Finally, we prove the adaptive security of our scheme through rigorous theoretical analysis. By conducting a large number of comparative experiments, our scheme significantly outperforms existing keyword similarity query schemes in the outsourcing, token generation, and query phases. Additionally, we evaluate the feasibility and scalability of the PPSKSQ scheme for different densities of datasets, and the flexibility of the quadtree hierarchy selection in terms of its impact on efficiency.

The remainder of this paper is organized as follows: Related works are introduced in Section II. In Section III, the system model, threat model, and design goals are presented. Section IV revisits the preliminaries used in this study. Section V focuses on demonstrating the PPSKSQ scheme. Subsequently, Section VI carries out the security analysis of the scheme, followed by a detailed experimental evaluation in Section VII. Finally, conclusions are discussed in Section VIII.

II. RELATED WORK

With the rapid advancement of LBSs, various types of spatial queries, including spatial keyword queries and top-k spatial keyword queries, have been introduced, accompanied by numerous privacy-preserving schemes. In this subsection, we review the related privacy-preserving schemes for each of these categories.

Spatial queries. Privacy-preserving schemes for spatial queries have been suggested in the literature [24]–[27]. These schemes utilized Secure k-Nearest Neighbors and Shen-Shi-Waters (SSW) encryption methods in conjunction with a tree-based structure to efficiently retrieve spatial data. However, these schemes operated within a symmetric environment, requiring all participating parties to possess secret keys. Recently, Tong et al. [28] introduced a spatial Boolean range query scheme, termed PBRQ, which combined Bloom filters and Katz-Sahai-Waters (KSW) encryption. This scheme achieved linear search efficiency in querying through the use of Gray code-encoded constructions. Miao et al. [14] designed a novel Bloom filter structure and then proposed an efficient privacy-preserving range query scheme, denoted as PSRQ. The scheme could provide security against Chosen-Plaintext Attacks, but its query ranges were limited to the Geohash algorithm, which lacked diversity.

Spatial keyword queries. Spatial keyword queries, a key element of LBSs, have attracted considerable attention in recent years. Yang et al. [9] proposed a keyword search scheme for polygon ranges that employs polynomial fitting technology for range queries and matrix multiplication for keyword queries. Cui et al. [10] proposed a boolean keyword query scheme under known background attacks by mapping spatial keyword information into Bloom filters. Tu et al. [11] shifted the research focus to multi-keyword queries and, for ranges and keywords, designed intersection and subset predicate encryption schemes, known as PMRK. However, none of these schemes considered the similarity between keywords. Recently, numerous schemes have been proposed for keyword similarity studies [13], [16], [29], [30]. In [13] and [16], Euclidean distance and Term Frequency-Inverse Document Frequency(TF-IDF) were used to measure similarity, respectively, which are less suitable than Jaccard for determining similarity between sets of keywords. In [29], Bloom filters were used to transform similarity measures into comparisons of inequalities, necessitating extensive preparatory work from

users, thereby increasing their overhead. In [30], the scheme encoded location and keyword data together into a vector but still required a bilinear pairing operation during the final validation, significantly increasing the computational burden. In contrast to the previously mentioned schemes, our approach supports both keyword matching and similarity measurement among keyword sets.

Top-k spatial keyword queries. In spatial keyword similarity queries, there is a need to retrieve the top-k keywords, thus, top-k keyword queries have emerged as a new research problem. Li et al. [31] proposed a scheme, denoted as PSKF, to facilitate top-k spatial keyword queries, leveraging fog computing. The scheme achieved text pruning by enhancing the R-tree structure. Xu et al. [32] proposed a method for retrieving the top-k diagnostic documents applicable in healthcare settings. Specifically, they combined the secure kNN algorithm as well as Paillier homomorphic encryption to conduct comparisons of Euclidean distances among diagnostic documents, ultimately obtaining the top-k diagnostic documents with the highest similarity. In [33], Liu et al. employed the homomorphic order-preserving encryption algorithm (FHOPE) to encrypt index and query vectors for conducting top-k multi-keyword queries. In contrast to the aforementioned scheme, our approach facilitates efficient tree pruning while also providing top-k keyword information retrieval.

III. MODELS AND DESIGN GOALS

In this subsection, we discuss the system model, threat model, and design goals of our PPSKSQ scheme.

A. System Model

Our system model, depicted in Fig. 1, utilizes this model to provide privacy-preserving spatial range and similarity queries within an outsourced cloud environment. It consists of three principal entities: data provider (DP), query users ($QU = u_1, u_2, ..., u_n$), and two cloud servers (C_1,C_2).

• Data provider (DP). The data providers are responsible for initializing the entire system and have a large amount of location and keyword data, denoted as $\mathcal{D} = \{p_i = (x_i, y_i), W_i\}$, where p_i is the location information of each data point, and W_i is the n-dimensional keyword set information. DP will outsource these sensitive data to a computationally powerful cloud server and provide registered users with spatial range and keyword similarity query (SKSQ) services. To protect privacy and improve query efficiency, DP will construct a quadtree-based structure, store spatial data in the quadtree, and encrypt the tree. In addition, DP will encrypt the dataset $\mathcal{D}$. The ciphertext tree and ciphertext dataset are finally uploaded to the cloud server.

• Two cloud servers (C_1,C_2). Our system is deployed under a two-cloud server model, and each cloud server has a lot of storage and computing power. Without compromising the privacy of DP and QU, C_1 and C_2 will collaborate to store and perform spatial range and keyword similarity queries.

• Query users ($QU = \{u_1, u_2, ..., u_n\}$). A registered query user u_i wants to search for query results within a rectangular range that satisfies the keyword similarity threshold. The u_i will launch a request to the cloud servers via SKSQ, and the cloud servers will return the corresponding query results to the query user based on the user's requests.

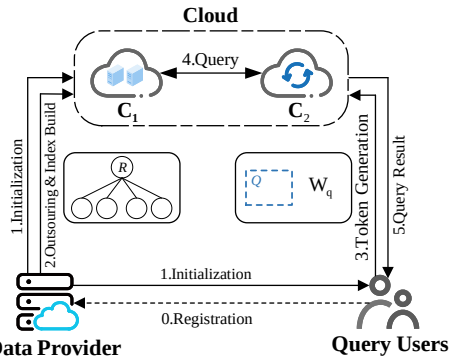


Fig. 1. PPSKSQ system model.

B. Threat Model

The data provider (DP) is considered a honest party within the system [21]. The cloud servers C_1 and C_2 are considered semi-honest entities; they store DP's dataset $\mathcal{D}$ and provide SKSQ services to query users (u_i) reliably. However, C_1 and C_2 may attempt to infer details about the encrypted dataset $\mathcal{D}$, the encrypted tree structure, query requests from u_i, and the final results. It is important to note that we assume no collusion between C_1 and C_2 [22], a presumption grounded in the strict regulatory environment of the real world, where any improper conduct would jeopardize their reputation. Regarding u_i, we consider that he/she is also an honest participant and will not collude with other entities because they want to go through the SKSQ service to find their desired query results. In summary, our system supports secure querying while maintaining privacy-preserving properties and satisfies the security requirements of the adaptively chosen plaintext attack (IND-CPA) model. Specifically, it can defend against data leakage attacks, query pattern analysis attacks, result analysis attacks, and known plaintext attacks (KPA) because our scheme limits the exposure of plaintext-ciphertext pairs to a finite set, ensuring that attackers cannot infer other ciphertext structures. We do not provide an extensive discussion on other attack methods, such as channel attacks, as these issues will be addressed as part of our future work.

C. Design Goals

Our PPSKSQ scheme aims to achieve efficient and privacy-preserving spatial range and keyword similarity queries in an outsourcing environment and thus will satisfy the following design goals.

• Data privacy. The PPSKSQ scheme must meet privacy preservation requirements, ensuring the protection of sensitive data and results within an outsourced cloud environment. First, protecting the datasets, index information, and quadtree structures uploaded by data providers from access by cloud servers is essential. Second, query requests and tokens from querying users must not be leaked to the cloud servers. Finally,

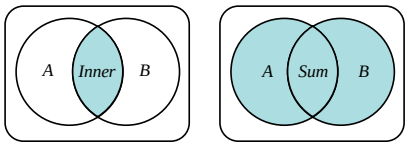


Fig. 2. Graphical representation of Jaccard's definition of similarity. The figure on the left illustrates the intersection of sets, while the figure on the right depicts their union.

the query results returned to users must also be protected from being disclosed to the cloud servers.

• Unlinkability. Ensure that the cloud servers cannot infer the tokens uploaded by a few random querying users, i.e., the cloud servers cannot infer the relationship between the ciphertext and plaintext from the query requests.

• Query efficiency. The PPSKSQ scheme avoids highly consumptive query operations and implements range queries and keyword similarity queries efficiently and cost-effectively.

IV. PRELIMINARIES

In this section, we introduce the concept of Jaccard similarity, which leads to the definition of spatial range and keyword similarity queries, and finally to the iSHE encryption scheme.

A. Jaccard similarity

In the field of similarity measurement, numerous methods have been utilized, such as Jaccard similarity, Euclidean distance, TF-IDF, and others. Euclidean distance is primarily suitable for comparing numerical data and lacks semantic explanatory power for the similarity measure of keyword sets; TF-IDF, while effective for exact textual data matching, is limited in addressing fuzzy matching between sets. Given that keywords are represented as sets in our scheme, employing Jaccard similarity is the optimal choice for measuring similarity. Furthermore, Jaccard similarity more accurately reflects the similarity between keyword sets by calculating the ratio of intersection to union, rather than the number or orientation of elements. In large-scale spatial keyword querying, since our PPSKSQ scheme is deployed in a dual-cloud environment, this allows us to maximize the advantages of this system to reduce the computational complexity of Jaccard, and thus achieve similarity querying more efficiently. Typically, for two sets $A = (a_1, a_2, ..., a_n)$ and $B = (b_1, b_2, ..., b_n)$, Jaccard similarity is calculated as $J(A,B) = \frac{|A \cap B|}{|A \cup B|}$. Since our scheme converts the set of keywords into binary form, we can thus transform the definition of Jaccard similarity as follows Eq. (1), and it is graphically represented as shown in Fig. 2.

$$J(A,B) = \frac{\sum_{i=1}^{n}(a_i \cdot b_i)}{\sum_{i=1}^{n}(a_i + b_i)} = \frac{Inner}{Sum}, \tag{1}$$

B. Spatial range and keyword similarity queries

A dataset in two-dimensional space comprises numerous location points, each encoding keyword information, represented as $\mathcal{D} = \{p_i = (x_i, y_i), W_i\}$. Given a query request $\mathcal{Q} = \{R_q = (R_q^{(x)}, R_q^{(y)}), W_q\}$, where $R_q^{(x)} = (x_q^l, x_q^r)$ and $R_q^{(y)} = (y_q^l, y_q^r)$ represent the x- and y-directions of the rectangular range, and denotes the d-dimensional query keywords, with $(d < n)$. For the query request, our primary objective is to identify data points whose locations fall within the specified rectangular range and exhibit the highest keyword similarity.

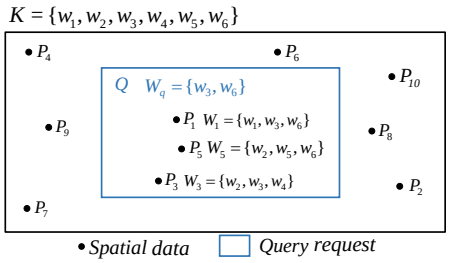


Fig. 3. Example of SKSQ. The keyword dictionary consists of six keywords, with ten location points and query regions (represented as blue rectangles) within the rectangular area. The keyword for the query request is W_q.

• *Spatial range queries.* To ensure that the data points are within the rectangular range of the query, i.e., both $(x_i - x_q^l)(x_q^r - x_i) > 0$ and $(y_i - y_q^l)(y_q^r - y_i) > 0$ are satisfied.

• *Spatial keyword similarity queries.* Given a set of keywords $W_q = (q_1, q_2, ..., q_d)$ for a query, there are the following steps to filter the data points with the highest similarity. First, map the set of keywords to a binary vector, where each element in the vector is set to 1 if the corresponding keyword is in the keyword dictionary and 0 otherwise. Second, compare the magnitude of similarity between the data points and the keyword binary vector of the query request, i.e., $J(W_i, W_q) - J(W_j, W_q)$, by defining this comparison equation as Φ. Finally, sort the similarity of all the data points and select the one with the highest similarity as the query result, i.e., $Top(\Phi)$.

Formal definitions of spatial range and keyword similarity queries are presented below.

Definition 1 (Spatial range and keyword similarity queries)

Given a query request $\mathcal{Q}$, the spatial range and keyword similarity query (SKSQ) filters the data points to the query user that fulfills the following conditions:

1) Spatial range requirement, i.e., $((x_i - x_q^l)(x_q^r - x_i) > 0) \wedge ((y_i - y_q^l)(y_q^r - y_i) > 0)$.
2) Spatial keyword similarity requirement, i.e., $Top(\Phi)$.

Fig. 3 presents a simple example of SKSQ, where $P_i (1 \leq i \leq 10)$ represents spatial data points, each possessing a set of keywords. Firstly, we define the keyword dictionary set as $K = \{w_1, w_2, w_3, w_4, w_5, w_6\}$. Secondly, the keyword sets for data points P_1, P_3, and P_5 are denoted as $W_1 = \{w_1, w_3, w_6\}$, $W_3 = \{w_2, w_3, w_4\}$, and $W_5 = \{w_2, w_5, w_6\}$, respectively. Given a query request $\mathcal{Q} = \{R_q, W_q\}$, where $W_q = \{w_3, w_6\}$ denotes the query keyword set. When the querying user launches SKSQ, it will first perform the filtering of spatial range requirement to obtain $\{P_1, P_3, P_5\}$, followed by the filtering of spatial keyword similarity requirement

by calculating $J_1(W_1, W_q) = \frac{2}{5}, J_2(W_3, W_q) = \frac{1}{5}$, and $J_3(W_5, W_q) = \frac{1}{5}$, respectively, and sorting the keyword similarity, i.e., $Top(J_1, J_2, J_3) = J_1$, and finally return the result of SKSQ, $\{P_1\}$, to the user.

C. iSHE encryption scheme

The iSHE encryption scheme [34] is a symmetric encryption scheme with homomorphic properties and is shown to be semantically secure with CPA and also resistant to AGCD attacks. Specifically, the iSHE encryption scheme comprising three algorithms is as follows.

• $KenGen(k_0, k_1, k_2, k_r) \rightarrow (pp, sk)$: The key generation algorithm takes the security parameters (k_0, k_1, k_2, k_r) as input and selects random large prime numbers p and q of size k_0, i.e., $|p| = |q| = k_0$, and further computes $\mathcal{N} = pq$. Next, the algorithm selects a random number s. Then, the algorithm selects another random number $\mathcal{L}$ of size k_2, i.e., $|q| = k_2$, and sets the secret key to $sk = (s, p, \mathcal{L})$. In addition, the message space size is $M = \left[-2^{k_1-1}, 2^{k_1-1}\right)$, and set the public parameters as $pp = (k_0, k_r, M, \mathcal{N})$, where the security parameters satisfy $k_1 < k_2 = k_r < k_0$.

• $Enc(m, sk) \rightarrow E(m)$: Given a secret key sk and a message $m \in M$, the encryption algorithm chooses two random numbers r and r' such that $r \in \{0,1\}^{k_r}, r' \in \{0,1\}^{k_0}$, and sets an exponent value d of sk. Therefore, the message m is encrypted $(E(m), d) = Enc(m, sk) = s \cdot (r\mathcal{L} + m)(1 + r'p) \bmod \mathcal{N}$ in the following way, where d is 1.

• $Dec(E(m), sk, d) \rightarrow m$: The decryption algorithm takes as input the ciphertext $E(m)$, the secret key sk, and its counterpart d, and obtains the plaintext of the message as follows: $m' = (((s^d)^{-1}) \cdot E(m) \bmod p) \bmod \mathcal{L}$, where $m = \begin{cases} m' & \text{if } m' < \mathcal{L}/2 \\ m' - L & \text{if } m' > \mathcal{L}/2 \end{cases}$ and $(s^d)^{-1} \cdot s^d \equiv 1 \bmod \mathcal{N}$.

iSHE encryption scheme has homomorphic addition and homomorphic multiplication properties in the ciphertext state. Homomorphic addition: $Dec(E(m_1) + E(m_2), sk, d) \rightarrow E(m_1 + m_2)$, where $d = max(d_1, d_2)$; $Dec(E(m_1) + m_2, sk, d) \rightarrow E(m_1 + m_2)$, where $d = d_1$. Homomorphic multiplication: $Dec(E(m_1) \cdot E(m_2), sk, d) \rightarrow E(m_1 \cdot m_2)$, where $d = d_1 + d_2$; $Dec(E(m_1) \cdot m_2, sk, d) \rightarrow E(m_1 \cdot m_2)$, where $d = d_1$.

Based on the above homomorphic properties, the iSHE symmetric encryption scheme likewise has a public key way to encrypt messages. The public key is set to $pk = \{E(0), E(0)'), E(1)\}$, where $E(0)$ and $E(0)'$ are two different cryptography representations of 0. Given the public key pk, it is possible to encrypt a message $m \in M$ in the following way and by utilizing homomorphic properties,

$$E(m) = (m \cdot E(1) + r_1 \cdot E(0) + r_2 \cdot E(0)') \bmod \mathcal{N}, \quad (2)$$

where the random numbers $r_1, r_2 \in \{0,1\}^{k_r}$. The public key version encryption scheme also satisfies the semantic security of IND-CPA [34].

TABLE I
NOTATION DEFINITION

Notation	Definition
h	Height of the quadtree
$\hat{d}$	Number of spatial points within the rectangle
d'	Length of vector
k	Number of range vectors in PBRQ-T+
η	Degree of polynomial fit in TSKS
T_M, T_V	Cost of matrix/vector multiplication
T_E	Cost of modular exponentiation in PBRQ-T+
T_P	Cost of bilinear pairing in PBRQ-T+
T_{TFIDF}	Cost of TFIDF calculation
iS_{pk}, iS_{sk}	Cost of iSHE public/secret key encryption
iS_a	Cost of iSHE homomorphic addition
iS_m	Cost of iSHE homomorphic multiplication
DS	Cost of iSHE decryption
$\|Z\|$	Size of the ciphertext in TSKS
$\|V_n\|$	Size of an n-dimensional vector
$\|M_{n*n}\|$	Size of $n*n$ dimensional matrix
$\|D\|, \|C\|$	Size of the dataset and ciphertext
$\|G\|, \|G_T\|$	Element lengths of cyclic groups in PSRQ-T+

V. OUR PROPOSED SCHEME

In this section, we first design a new secure comparison protocol based on some basic protocols of iSHE. Second, for spatial range queries, we propose a secure computational inner product protocol, iSHE-SIP, based on the random mask protocol in [35]. Then, we illustrate how to utilize the Jaccard similarity metric to measure keyword similarity queries in our scheme. Finally, with the above building blocks, we propose the privacy-preserving spatial keyword similarity query scheme, i.e., PPSKSQ. For the sake of clarity of the subsequent descriptions in the article, the notations used are shown in Table I.

A. Construction of Basic Protocols and Query Patterns

1) **iSHE-SC protocol.** Unlike previous secure comparison protocols [36], [37], our iSHE-SC protocol enables us to determine whether two ciphertexts are greater than, less than, or equal to each other in a single round of comparison. As we utilize an enhanced version of the symmetric homomorphic encryption scheme iSHE, our secure comparison protocol is extremely robust in terms of security. The secure comparison protocol is deployed on two cloud servers C_1 and C_2 to securely determine the size relationship between two given ciphertexts $E(m_1)$ and $E(m_2)$ without leaking any information about plaintexts m_1 or m_2 to C_1 and C_2. Where C_1 holds the ciphertexts $E(m_1)$ and $E(m_2)$, and C_2 holds the secret key sk. The protocol outputs $E(1)$ if $m_1 > m_2$, $E(-1)$ if $m_1 < m_2$, and $E(0)$ otherwise if $m_1 = m_2$.

• *Step-1*: C_1 chooses a number s whose values $E(1)$ and $E(-1)$ are calculated by Eq. (2). Then, C_1 randomly chooses two random numbers $r_1, r_2 \in M$ such that they satisfy $r_1 \gg r_2 > 0$. Next, C_1 computes $E(z) = s \cdot E(r_1) \cdot (E(m_1) + E(m_2) \cdot E(-1)) + s \cdot E(r_2) \cdot E((m_1 - m_2)^2) \cdot E(-1) = E(s \cdot r_1 \cdot (m_1 - m_2) - s \cdot r_2 \cdot (m_1 - m_2)^2)$. Afterwards, C_1 sends $E(z)$ to C_2.

• *Step-2*: Upon receiving $E(z)$, C_2 uses sk to get z and determine the size of z. If $z > 0$, C_2 sets $\theta = 1$ and encrypts

it as $E(\theta)$; if $z < 0$, C_2 lets $\theta = -1$; otherwise $\theta = 0$. Then, C_2 returns $E(\theta)$ to C_1.

• *Step-3*: After receiving $E(\theta)$, C_1 computes $E(\mu)$ as

$$E(\mu) = \begin{cases} E(\theta) & \text{if } s = E(1) \\ E(\theta) \cdot E(-1) & \text{if } s = E(-1) \end{cases}.$$

Finally, $E(\mu)$ is the output of the entire protocol.

Correctness. If $s = 1$, then $E(z) = E(r_1 \cdot (m_1 - m_2) - r_2 \cdot (m_1 - m_2)^2)$. Since $r_1 \gg r_2 > 0$, if $m_1 > m_2$, then $z > 0$. Hence, $E(\mu) = E(\theta) = E(1)$. If $m_1 < m_2$, then $z < 0$. Then, $E(\mu) = E(\theta) = E(-1)$. If $m_1 = m_2$, then $z = 0$. In this case, $E(\mu) = E(\theta) = E(0)$. Similarly, when $s = -1$, we can equally prove that the protocol outputs $E(\mu) = E(1)$ if $m_1 > m_2$, $E(\mu) = E(-1)$ if $m_1 < m_2$, and $E(\mu) = E(0)$ if $m_1 = m_2$. In summary, the secure comparison protocol is correct.

2) **Spatial range queries.** To determine the inequality in Definition 1, our main idea is to determine the sign of two inequalities by their inner product. First, the spatial point (x_i, y_i) is encoded into vectors $\overrightarrow{P_i^{(x)}} = (x_i, x_i, -1, x_i^2)$ and $\overrightarrow{P_i^{(y)}} = (y_i, y_i, -1, y_i^2)$ along the x-direction and y-direction, respectively. Second, we encode the rectangular range R_q also into $\overrightarrow{P_{R_q}^{(x)}} = (x_q^l, x_q^r, x_q^l \cdot x_q^r, -1)$ and $\overrightarrow{P_{R_q}^{(y)}} = (y_q^l, y_q^r, y_q^l \cdot y_q^r, -1)$ along the x-direction and y-direction, respectively. Finally, we determine whether $(\overrightarrow{P_i^{(x)}}, \overrightarrow{P_{R_q}^{(x)}})$ and $(\overrightarrow{P_i^{(y)}}, \overrightarrow{P_{R_q}^{(y)}})$ are greater than 0 by using the inner product to determine if the spatial points fall within the rectangle. In this way, we can use a single inner product in a quadtree recursion to determine whether a point lies within a rectangle and whether the quadtrees intersect. To ensure the secure computation of the inner product of two vectors, we designed a dedicated protocol, iSHE-SIP, which is built on the iSHE framework.

3) **iSHE-SIP protocol.** Unlike previous methods for securely computing inner products, such as the Secure k-Nearest Neighbor algorithm described in [30], which requires additional matrices to maintain privacy, our iSHE-SIP protocol offers a secure computational inner product method utilizing an eliminable random number design. This approach demonstrates significantly higher efficiency, particularly for dense computational vector dimensions. Given two vectors $\overrightarrow{X} = (x_1, x_2, ..., x_n)$ and $\overrightarrow{Y} = (y_1, y_2, ..., y_n)$, the input of the protocol is two vectors $E(\overrightarrow{X})$ and $E(\overrightarrow{Y})$ encrypted by iSHE, and if the inner product of $E(\overrightarrow{X})$ and $E(\overrightarrow{Y})$ is greater than 0, then the output is $E(1)$, otherwise $E(0)$. C_1 holds $E(\overrightarrow{X})$, C_2 holds $E(\overrightarrow{Y})$, and the flow of the protocol is as follows.

• *Step-1*: C_1 chooses an n-dimensional random vector $\overrightarrow{R} = (r_1, r_2, ..., r_n), r_i \in M$ and computes $\overrightarrow{Z} = E(\overrightarrow{X}) - \overrightarrow{R} = E(x_1 - r_1, x_2 - r_2, ..., x_n - r_n)$. In addition, C_1 chooses a random number r at random and computes $\overrightarrow{W} = r \cdot \overrightarrow{Z} = (w_1, w_2, ..., w_n)$. C_1 then sends $\{\overrightarrow{R}, \overrightarrow{W}\}$ to C_2.

• *Step-2*: Once $\{\overrightarrow{R}, \overrightarrow{W}\}$ is received, C_2 computes

$$\begin{cases} u = \overrightarrow{R} \cdot E(\overrightarrow{Y}) = E(r_1 \cdot y_1 + r_2 \cdot y_2 + ... + r_n \cdot y_n) \\ v = \overrightarrow{W} \cdot E(\overrightarrow{Y}) = E(w_1 \cdot y_1 + w_2 \cdot y_2 + ... + w_n \cdot y_n) \end{cases}.$$

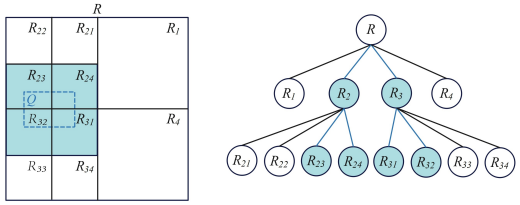


Fig. 4. Examples of quadtree construction and search. Example of quadtree construction and search, where the left figure shows the two-dimensional space divided and the right figure shows the corresponding quadtree structure with a depth of 3.

Next, C_2 returns $\{u, v\}$ to C_1.

• *Step-3*: After receiving $\{u, v\}$, C_1 computes $E(\theta) = u + r^{-1} \cdot v$ and sends $E(\theta)$ to C_2.

• *Step-4*: After accepting $E(\theta)$, C_2 recovers θ with the secret key and judges its size; if $\theta > 0$, it sets $E(\mu) = E(1)$, otherwise $E(\mu) = E(0)$. Next, C_2 sends $E(\mu)$ to C_1.

Finally, $E(\mu)$ is the output of the whole protocol.

Correctness. Due to $\overrightarrow{W} = r \cdot \overrightarrow{Z} = (E(r \cdot (x_1 - r_1)), E(r \cdot (x_2 - r_2)), ..., E(r \cdot (x_n - r_n)))$, it follows that $u + r^{-1} \cdot v = E(r_1 \cdot y_1 + (x_1 - r_1)y_1 + r_2 \cdot y_2 + (x_2 - r_2)y_2 + ... + r_n \cdot y_n + (x_n - r_n)y_n) = E(x_1 \cdot y_1 + x_2 \cdot y_2 + ... + x_n \cdot y_n) = E(\overrightarrow{X} \cdot \overrightarrow{Y})$. Moreover, since $u + r^{-1} \cdot v = \overrightarrow{R} \cdot E(\overrightarrow{Y}) + r^{-1} \cdot \overrightarrow{W} \cdot E(\overrightarrow{Y}) = \overrightarrow{R} \cdot E(\overrightarrow{Y}) + r^{-1} \cdot r \cdot \overrightarrow{Z} \cdot E(\overrightarrow{Y}) = (\overrightarrow{R} + \overrightarrow{Z}) \cdot E(\overrightarrow{Y}) = E(\overrightarrow{X}) \cdot E(\overrightarrow{Y})$, the iSHE-SIP protocol is correct.

However, as spatial datasets expand, the complexity of queries also increases. To efficiently handle large-scale spatial datasets, we utilize quadtree properties [38]–[40] for two-dimensional spatial data retrieval. The quadtree-based construction comprises three main types of nodes: root node, internal nodes, and leaf nodes. In Fig. 4, we have given a schematic diagram of quadtree construction and search.

• Root node. The root node, representing the largest rectangle that encompasses the entire two-dimensional space, is generally disregarded as it inherently contains all information within this space.

• Internal nodes. Each root node comprises four internal nodes that divide the two-dimensional space into four rectangular subspaces, each corresponding to a subtree. Additionally, the internal nodes represent the information of the rectangular subspaces, which we encode as vectors stored in them, i.e., $R_i = (\overrightarrow{P_{R_i}^{(x)}}, \overrightarrow{P_{R_i}^{(y)}})$. Consequently, each internal node houses $\{R_i, n_i.child\}$, with $n_i.child$ serving as a pointer to the child node.

• Leaf nodes. A leaf node is the smallest rectangular subspace formed after constant recursion, similar to an internal node. Furthermore, leaf nodes retain information pertaining to the spatial data encompassed by these rectangular subspaces.

Using the quadtree construction, we traverse the entire tree to determine if the query ranges intersect with its nodes. Given a rectangle query range $R_q = (R_q^{(x)}, R_q^{(y)})$ and information about the non-leaf nodes of a quadtree $(R_i^{(x)}, R_i^{(y)})$, analogous to Definition 1, we transform the intersection of rectangles into a judgment of the following two lower equations: $(x_q^r - x_i^l)(x_i^r - x_q^l) > 0$ and $(y_q^r - y_i^l)(y_i^r - y_q^l) > 0$. Therefore, for the

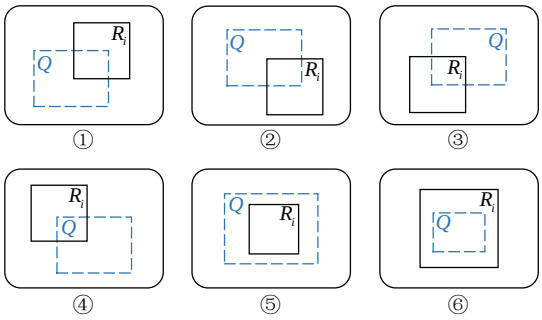


Fig. 5. Rectangular intersection case. Rectangular intersection case, where the blue dashed box represents the query range, and the black solid box represents the node range of the corresponding quadtree.

non-leaf node information R_i of the quadtree, the x-direction and y-direction are encoded as vectors $\overrightarrow{P_{R_i}^{(x)}} = (x_i^l, x_i^r, -1, x_i^l \cdot x_i^r)$ and $\overrightarrow{P_{R_i}^{(y)}} = (y_i^l, y_i^r, -1, y_i^l \cdot y_i^r)$, respectively. This approach has the advantage of merging all the original range information into one vector and filtering out the eligible spatial points with a single inner product calculation. The query range R_q is also encoded into vectors $\overrightarrow{P_{R_q}^{(x)}} = (x_q^l, x_q^r, x_q^l \cdot x_q^r, -1)$ and $\overrightarrow{P_{R_q}^{(y)}} = (y_q^l, y_q^r, y_q^l \cdot y_q^r, -1)$ along the x-direction and y-direction, respectively. Finally, we can tell whether the query range intersects an internal node of the quadtree by computing whether the inner products of $(\overrightarrow{P_{R_i}^{(x)}}, \overrightarrow{P_{R_q}^{(x)}})$ and $(\overrightarrow{P_{R_i}^{(y)}}, \overrightarrow{P_{R_q}^{(y)}})$ are both greater than 0. In Fig. 5, several cases of intersecting or not intersecting two rectangles are given.

4) **Spatial keyword similarity queries.** In PPSKSQ, after executing the spatial range queries, the next step is to compare the magnitude of similarity. The comparison of similarities is conducted by calculating their differences and sorting the results, subsequently returning the data record with the highest similarity to the querying user. Given a keyword set $W_q = (q_1, q_2, ..., q_d)$ for a query, and data points P_i and P_j that meet the spatial range query, their keyword sets are denoted as $W_i = (a_1, a_2, ..., a_n)$ and $W_j = (b_1, b_2, ..., b_n)$, respectively. From Jaccard's definition of similarity in Section IV, we need to compute $\Phi = J(W_i, W_q) - J(W_j, W_q) = \frac{Inner_1}{Sum_1} - \frac{Inner_2}{Sum_2} = \frac{Inner_1 \cdot Sum_2 - Inner_2 \cdot Sum_1}{Sum_1 \cdot Sum_2}$ and determine the size of the equation. By observation, we know that the denominator is constantly greater than zero, so we only need to go to the numerator to determine its size. Setting the numerator to be ω, once we know the size of ω, we can determine the size of $J(W_i, W_q) - J(W_j, W_q)$. To determine the size of ω, we expand the equation, $\omega = \sum_{i=1}^{n}(a_i \cdot q_i)\sum_{j=1}^{n}(b_i + q_i) - \sum_{i=1}^{n}(b_i \cdot q_i)\sum_{j=1}^{n}(a_i + q_i) = \sum_{i=1}^{n} a_i q_i \cdot (\sum_{j=1}^{n} b_i + \sum_{j=1}^{n} q_i) - \sum_{i=1}^{n} b_i q_i \cdot (\sum_{j=1}^{n} a_i + \sum_{j=1}^{n} q_i)$.

We set $Sum_{W_i} = \sum_{i=1}^{n} a_i, Sum_{W_j} = \sum_{i=1}^{n} b_i, Sum_{W_q} = \sum_{i=1}^{n} q_i$. We can get

$$\begin{aligned} \omega = &(Sum_{W_j} + Sum_{W_q}) \cdot \sum_{i=1}^{n} a_i q_i \\ &- (Sum_{W_i} + Sum_{W_q}) \cdot \sum_{i=1}^{n} b_i q_i. \end{aligned} \tag{3}$$

Observing ω, it can be observed that the binary addition operation can be implemented locally efficiently, whereas computing the multiplication locally incurs severe computational overhead. To solve the latter computation, we utilize the matrix multiplication technique to convert the operation of product sum into the operation of matrix trace [13], which will be elaborated in subsection V-B. Finally, we can obtain $Top(\Phi)$, i.e., return the data record with the highest similarity.

B. Detailed construction of PPSKSQ

In this subsection, based on all the above constructions, we propose our scheme, i.e., PPSKSQ, whose main idea is to first filter out a portion of the data records that meet the requirements based on the spatial range requirement, and then go on to derive the data points that meet the final requirements based on the spatial similarity requirement. To facilitate the introduction, we divide it into five phases: 1) Registration and Initialization; 2) Outsourcing and Index Build; 3) Token Generation; 4) Query; and 5) Result Recovery.

Registration and Initialization. The querying user first registers with the DP and only the registered u_i can participate in the service and the whole system is initialized by the DP. The initialization steps are as follows.

- *Step-1*: Given the security parameters (k_0, k_1, k_2, k_r), DP calls $iSHE.KeyGen()$ to generate the public parameter $pp = (k_0, k_r, M, \mathcal{N})$ and the secret key $sk = (s, p, \mathcal{L})$, and computes $\{E(0), E(0)'E(1)\}$ as a way to set the public key to $pk = \{pp, E(0), E(0)', E(1)\}$.
- *Step-2*: Inputting the security parameter λ, the DP randomly outputs two n-dimensional invertible matrices $\{M_1, M_2\}$, a permutation function π, and computes its inverse matrix $\{M_1^{-1}, M_2^{-1}\}$.
- *Step-3*: Finally, the DP publishes pk and sends $\{M_1^{-1}, M_2^{-1}, \pi\}$ to the registered u_i as well as the secret key sk to C_2.

Outsourcing and Index Build. In this phase, to efficiently filter the spatial location data, DP constructs a quadtree in two-dimensional space T. Additionally, DP maps the set of keywords to a binary vector and encrypts the vector. Finally, the encrypted T and spatial data are uploaded to C_1 as follows.

- *Step-1*: Given a spatial dataset $\mathcal{D} = \{p_i = (x_i, y_i), W_i\}$ based on the keyword dictionary, DP maps the keyword set information of each p_i into binary vectors. Next, DP encodes each spatial point $p_i = (x_i, y_i)$ into vectors $\overrightarrow{P_i^{(x)}} = (x_i, x_i, -1, x_i^2)$ and $\overrightarrow{P_i^{(y)}} = (y_i, y_i, -1, y_i^2)$, respectively, and randomly chooses a positive random number α to compute $\{\overrightarrow{P_i^{(x)'}} = \alpha\overrightarrow{P_i^{(x)}}, \overrightarrow{P_i^{(y)'}} = \alpha\overrightarrow{P_i^{(y)}}\}$. Subsequently, the location information p_i, the set of keywords W_i, and the vectors $\{\overrightarrow{P_i^{(x)'}}, \overrightarrow{P_i^{(y)'}}\}$ are encrypted as $E(\mathcal{D}) = \{E(p_i) = (E(x_i), E(y_i)), E(W_i), E(\overrightarrow{P_i^{(x)'}}), E(\overrightarrow{P_i^{(y)'}})\}$ using the secret key sk. In addition to this, DP computes and encrypts the sum of the keyword sets for each p_i, denoted as $E(Sum_{W_i})$.
- *Step-2*: The internal nodes of T are $\{R_i^{(x)} = (x_i^l, x_i^r), R_i^{(y)} = (y_i^l, y_i^r)\}$, which DP encodes them into vectors $\overrightarrow{P_{R_i}^{(x)}} = (x_i^l, x_i^r, -1, x_i^l \cdot x_i^r)$ and $\overrightarrow{P_{R_i}^{(y)}} = (y_i^l, y_i^r, -1, y_i^l \cdot y_i^r)$, respectively, and computes $\{\overrightarrow{P_{R_i}^{(x)'}} = \alpha\overrightarrow{P_{R_i}^{(x)}}, \overrightarrow{P_{R_i}^{(y)'}} = \alpha\overrightarrow{P_{R_i}^{(y)}}\}$.

In addition, each bit of the vector $\{\overrightarrow{P_{R_i}^{(x)'}}, \overrightarrow{P_{R_i}^{(y)'}}\}$ is encrypted with the secret key sk, and the encrypted vector is denoted as $\{E(\overrightarrow{P_{R_i}^{(x)'}}), E(\overrightarrow{P_{R_i}^{(y)'}})\}$ The leaf nodes of T store the information of the spatial dataset p_i, which we have stored encrypted in Step-1. To enhance privacy preserving, DP randomly shuffles the order of the child nodes belonging to the same parent node to obtain the encrypted tree $E(T)$.

• *Step-3*: For each spatial point mapping followed by an n-dimensional keyword set $W_i = (w_{i,1}, w_{i,2}, ..., w_{i,n}), i \in \mathcal{D}$. To protect privacy, DP first utilizes a random number α to compute αW_i and applies a permutation function π to the keyword set, denoted as $\overline{W_i} = \pi(\alpha W_i)$. Subsequently, DP constructs a diagonal matrix U_{W_i} with $\overline{W_i}$ as its diagonal elements and randomly constructs an upper triangular matrix V_{W_i} with all diagonal elements equal to 1, where the non-zero elements are randomly selected random numbers. Then, DP encrypts the keyword set W_i as $C(W_i) = M_1 V_{W_i} U_{W_i} M_2$.

Finally, DP outsources to C_1

$$\{E(\mathcal{D}), E(Sum_{W_i}), E(T), C(W_i)\}.$$

Token Generation. In this phase, the registered query users can launch SKSQ and generate token based on their query requests $\mathcal{Q} = \{R_q = (R_q^{(x)}, R_q^{(y)}), W_q\}$. The token generation process is as follows.

• *Step-1*: u_i maps the set of query keywords W_q to binary vectors based on the keyword dictionary, computes the sum Sum_{W_q} of W_q locally, and encrypts its sum with the public key pk as in Eq. (2), denoted as $E(Sum_{W_q})$.

• *Step-2*: For a rectangular query range $R_q = (R_q^{(x)}, R_q^{(y)})$, u_i encodes it into vectors $\overrightarrow{P_{R_q}^{(x)}} = (x_q^l, x_q^r, x_q^l \cdot x_q^r, -1)$ and $\overrightarrow{P_{R_q}^{(y)}} = (y_q^l, y_q^r, y_q^l \cdot y_q^r, -1)$, respectively, and randomly chooses a positive random number β to compute $\{\overrightarrow{P_{R_q}^{(x)'}} = \beta\overrightarrow{P_{R_q}^{(x)}}, \overrightarrow{P_{R_q}^{(y)'}} = \beta\overrightarrow{P_{R_q}^{(y)}}\}$, and subsequently, encrypts each bit of it as $\{E(\overrightarrow{P_{R_q}^{(x)'}}), E(\overrightarrow{P_{R_q}^{(y)'}})\}$ using pk.

• *Step-3*: For the query keyword request W_q, similar to a data provider, u_i first utilizes a random number β to compute βW_q and then applies the permutation function π on the set of query keywords, denoted as $\overline{W_q} = \pi(\beta W_q)$. Subsequently, u_i constructs a diagonal matrix U_{W_q} with diagonal elements $\overline{W_q}$, and a random upper triangular matrix V_{W_q} whose main diagonal elements are all 1, where the non-zero elements are randomly selected random numbers. Then, u_i encrypts the keyword set W_q as $C(W_q) = M_2^{-1} U_{W_q} V_{W_q} M_1^{-1}$ using $\{M_1^{-1}, M_2^{-1}\}$.

Finally, u_i generates query token $\underline{Tk_1} = \{E(Sum_{W_q}), C(W_q)\}$ and $Tk_2 = \{E(\overrightarrow{P_{R_q}^{(x)'}}), E(\overrightarrow{P_{R_q}^{(y)'}})\}$ and sends Tk_1 to C_1 and Tk_2 to C_2.

Query. In this phase, the two cloud servers collaborate to complete the query operation, they first perform the spatial range queries and then the keyword similarity queries. Since only the spatial data points that satisfy the query rectangular range continue to be performed for the keyword similarity queries, we elaborate the query in two phases, i.e., spatial range queries and keyword similarity queries.

1) Spatial range queries. In this subphase, C_1 holds $E(T)$ and C_2 holds Tk_2. Firstly, the filtering starts from the root node of $E(T)$, and when filtering to the internal nodes, C_1 possesses the encoded vector of each encrypted subrectangle $\{R_i^{(x)}, R_i^{(y)}\}$, i.e., $\{E(\overrightarrow{P_{R_i}^{(x)'}}), E(\overrightarrow{P_{R_i}^{(y)'}})\}$. At this time, C_2 owns the encoded vector $\{E(\overrightarrow{P_{R_q}^{(x)'}}), E(\overrightarrow{P_{R_q}^{(y)'}})\}$ of the encrypted query range $\{R_q^{(x)}, R_q^{(y)}\}$, and takes $E(\overrightarrow{P_{R_i}^{(x)'}})$ and $E(\overrightarrow{P_{R_q}^{(x)'}})$, as well as $E(\overrightarrow{P_{R_i}^{(y)'}})$ and $E(\overrightarrow{P_{R_q}^{(y)'}})$ as inputs to the iSHE-SIP protocol, respectively, and determines whether the two rectangles intersect or not by securely computing the inner product. Moreover, set the output of the former to be $E(\mu_1)$ and the latter to be $E(\mu_2)$. If and only if C_2 computes $E(\mu_1)E(\mu_2) > 0$, we then continue recursively to the child node of the current node, that is, $ni.child$; otherwise, the node will be pruned. This process continues until we filter down to the leaf nodes, at which point C_1 possesses the encrypted spatial data point p_i's encoding vectors $\{E(\overrightarrow{P_i^{(x)'}}), E(\overrightarrow{P_i^{(y)'}})\}$. Then, the iSHE-SIP protocol is executed once again, this time with inputs $E(\overrightarrow{P_i^{(x)'}})$ and $E(\overrightarrow{P_{R_q}^{(x)'}})$ as well as $E(\overrightarrow{P_i^{(y)'}})$ and $E(\overrightarrow{P_{R_q}^{(y)'}})$, to determine whether the spatial data point p_i falls within the query rectangle range. Similarly, we set the output of the former as $E(\mu_1)$ and the latter as $E(\mu_2)$, and C_2 computes and determines whether the product of $E(\mu_1)$ and $E(\mu_2)$ is greater than 0. If C_2 determines the result to be greater than 0, then C_1 will set a query result candidate set $R' = \{E(p_{\hat{d}}), 0 \le \hat{d} \le i\}$, where $E(p_{\hat{d}})$ only includes spatial data points that satisfy the rectangle range criteria.

1) Keyword similarity queries. After executing the spatial range queries phase, in this subphase, the cloud server has to filter the spatial data points with the highest similarity to be returned to u_i as query results. The specific steps are as follows.

• *Step-1*: C_1 first extracts the encrypted keyword sets $C(W_{\hat{d}}) = M_1 V_{W_{\hat{d}}} U_{W_{\hat{d}}} M_2$ in the candidate set R', where $\{V_{W_{\hat{d}}}, U_{W_{\hat{d}}}\}$ is similar to the Outsourcing and Index Build phase. Next, C_1 uses Tk_1 to separately calculate each spatial data point $C(W_{\hat{d}})C(W_q), 0 \le \hat{d} \le i$, that falls within the rectangular range. Moreover, C_1 computes the matrix trace of each $C(W_{\hat{d}})C(W_q)$ separately, that is, $tr_{\hat{d}}(C(W_{\hat{d}})C(W_q))$, and encrypts the trace as $E(tr_{\hat{d}})$ with the public key pk.

• *Step-2*:C_1 has $\{E(Sum_{W_{\hat{d}}}), E(Sum_{W_q}), E(tr_{\hat{d}})\}$, and then it sorts $\hat{d}$ spatial data points in the candidate set R'. C_1 randomly selects two data points from the candidate set R' and calculates them as follows

$$\begin{cases} E(Sim_{\hat{d}_1}) = (E(Sum_{W_{\hat{d}_1}}) + E(Sum_{W_q}))E(tr_{\hat{d}_1}) \\ E(Sim_{\hat{d}_2}) = (E(Sum_{W_{\hat{d}_2}}) + E(Sum_{W_q}))E(tr_{\hat{d}_2}) \end{cases}.$$

As illustrated in Eq. (3)$\{\hat{d}_1, \hat{d}_2\}$ represents two data points chosen at random by C_1, while $\{E(Sum_{\hat{d}_1}), E(Sum_{\hat{d}_2})\}$ serves as an indicator of the similarity between the keywords associated with these two data points. C_1 takes $\{E(Sum_{\hat{d}_1}), E(Sum_{\hat{d}_2})\}$ as the input to the iSHE-SC protocol, and C_2 holds the secret key sk, and the two work in concert to execute the protocol. If the protocol outputs $E(\mu) =$

$E(1)$, then C_1 can know that $E(Sum_{\hat{d}_1}) > E(Sum_{\hat{d}_2})$; if the protocol outputs $E(\mu) = E(-1)$, then C_1 can know that $E(Sum_{\hat{d}_1}) < E(Sum_{\hat{d}_2})$; otherwise, if the output $E(\mu) = E(0)$, then C_1 can know that $E(Sum_{\hat{d}_1}) = E(Sum_{\hat{d}_2})$.

• *Step-3*: C_1 goes through all the data points in the candidate set R' as in Step 2 and sorts the similarity of all the data points. Subsequently, C_1 gets a data point with the highest similarity to the query keyword set, i.e., $Top(R')$. Finally, C_1 sets the encrypted query result set $E(R) = \{E(p_{\hat{d}}) = (E(x_{\hat{d}}), E(y_{\hat{d}})), E(W_{\hat{d}})\}$.

Correctness. Similar to the proof in [13], in linear algebra, there is the *lemma*: the trace of a square matrix A multiplied by an invertible matrix M and its inverse M^{-1} is equal, formally stated as $tr(A) = tr(MAM^{-1})$. In the process of our keyword similarity queries, the encrypted keyword set of the candidate set R' is $C(W_{\hat{d}}) = M_1 V_{W_{\hat{d}}} U_{W_{\hat{d}}} M_2$, and the encrypted keyword set of the query request is $C(W_q) = M_2^{-1} U_{W_q} V_{W_q} M_1^{-1}$. By the lemma, we know that $C(W_{\hat{d}})C(W_q) = M_1 V_{W_{\hat{d}}} U_{W_{\hat{d}}} U_{W_q} V_{W_q} M_1^{-1}$, as well as $tr(C(W_{\hat{d}})C(W_q)) = tr(V_{W_{\hat{d}}} U_{W_{\hat{d}}} U_{W_q} V_{W_q})$. Therefore, the keyword similarity queries are correct.

Result Recovery. In the recovery phase, C_1 and C_2 work together to complete the return of query results to u_i, and u_i has the ability and ease to recover the results. The result recovery process is as follows.

• *Step-1*: C_1 randomly selects a series of random numbers $r_{\hat{d}}$ for the location information $p_{\hat{d}}$ of each data point in the result set $E(R)$ and generates a corresponding sequence of random numbers $r_{W_{\hat{d}}} = \{r_{W_{\hat{d}_1}}, ..., r_{W_{\hat{d}_n}}\}$ for each bit of the keyword sets information $W_{\hat{d}}$, where n indicates the dimension of the keyword vector, and $\{r_{\hat{d}}, r_{W_{\hat{d}}}\}$ are elements of set M. Then, C_1 computes

$$\begin{cases} E(X_{\hat{d}}) = E(x_{\hat{d}}) + r_{\hat{d}} = E(x_{\hat{d}} + r_{\hat{d}}) \\ E(Y_{\hat{d}}) = E(y_{\hat{d}}) + r_{\hat{d}} = E(y_{\hat{d}} + r_{\hat{d}}) \\ E(K_{\hat{d}}) = E(W_{\hat{d}}) + r_{W_{\hat{d}}} = E(W_{\hat{d}} + r_{W_{\hat{d}}}) \end{cases}.$$

Next, C_1 sends $\{E(X_{\hat{d}}), E(Y_{\hat{d}}), E(K_{\hat{d}})\}$ to C_2 and random numbers $\{r_{\hat{d}}, r_{W_{\hat{d}}}\}$ to u_i.

• *Step-2*: Once $\{E(X_{\hat{d}}), E(Y_{\hat{d}}), E(K_{\hat{d}})\}$ are received, C_2 decrypts them separately with the secret key sk to obtain $\{x_{\hat{d}} + r_{\hat{d}}, y_{\hat{d}} + r_{\hat{d}}, W_{\hat{d}} + r_{W_{\hat{d}}}\}$. Subsequently, C_2 sends $\{x_{\hat{d}} + r_{\hat{d}}, y_{\hat{d}} + r_{\hat{d}}, W_{\hat{d}} + r_{W_{\hat{d}}}\}$ to u_i.

• *Step-3*: After receiving the encrypted random numbers $\{r_{\hat{d}}, r_{W_{\hat{d}}}\}$ and $\{x_{\hat{d}} + r_{\hat{d}}, y_{\hat{d}} + r_{\hat{d}}, W_{\hat{d}} + r_{W_{\hat{d}}}\}$, u_i can easily eliminate the random numbers to get the query result $R = \{p_{\hat{d}} = (x_{\hat{d}}, y_{\hat{d}}), W_{\hat{d}}\}$.

VI. SECURITY ANALYSIS

We first prove the security of the two privacy-preserving protocols, iSHE-SC and iSHE-SIP, and then justify the overall security of the scheme.

For security analysis, we employ ideal-real simulation [41]. Let Π denote the protocol. For a semi-honest adversary, if it cannot distinguish between the real-world and ideal-world views in probabilistic polynomial time, then Π is considered secure. Our focus is on adaptive adversaries, which can query based on information obtained from previous token and indexing phases. Specifically, to satisfy the security requirements of the adaptive IND-CPA model, we must demonstrate that in the ideal world, there exist two simulators, $\{\mathcal{S}im_1, \mathcal{S}im_2\}$, that simulate an ideal environment using an ideal function f. Meanwhile, in the real world, there exist two semi-honest adversaries $\{\mathcal{A}_1, \mathcal{A}_2\}$, whose purpose is to destroy the cloud server. By comparing the views of the ideal and real worlds, if the adversaries cannot distinguish between the two in probabilistic polynomial time, it can be concluded that the scheme is resilient to attacks from adaptive adversaries.

A. Security of the iSHE-SC and iSHE-SIP protocol

The iSHE-SC protocol is used to securely determine the size relationship between two ciphertexts $E(m_1)$ and $E(m_2)$ in the case of semi-honest adversaries $\{\mathcal{A}_1, \mathcal{A}_2\}$. In the real world, the view of $\mathcal{A}_1$ is the two encrypted values $\{E(m_1), E(m_2)\}$ and the final output of the protocol, $E(\mu)$, denoted as $View^{\Pi}_{real,\mathcal{A}_1}$, while the view of $\mathcal{A}_2$ is the $z = s \cdot r_1 \cdot (m_1 - m_2) - s \cdot r_2 \cdot (m_1 - m_2)^2$ received from C_1, denoted as $View^{\Pi}_{real,\mathcal{A}_2}$. Whereas, in the ideal world, we define the leakage of $\mathcal{S}im_1$ to be $\mathcal{L}_1 = \{pk, E(m_1), E(m_2), E(\mu)\}$ and the leakage of $\mathcal{S}im_2$ to be $\mathcal{L}_2 = \{sk\}$, and thus $\mathcal{A}_1$'s view is the $\{E(m_1'), E(m_2'), E(\mu')\}$ randomly generated by $\mathcal{S}im_1$ and encrypted with iSHE, denoted $View^{f}_{ideal,\mathcal{S}im_1,\mathcal{L}_1}$ and the view of $\mathcal{A}_2$ is the z' randomly generated by $\mathcal{S}im_2$, denoted as $View^{f}_{ideal,\mathcal{S}im_2,\mathcal{L}_2}$. For $\mathcal{A}_1$, all values are encrypted by iSHE, and due to the semantic security of iSHE [34], $\mathcal{A}_1$ cannot distinguish $View^{\Pi}_{real,\mathcal{A}_1}$ from $View^{f}_{ideal,\mathcal{S}im_1,\mathcal{L}_1}$. For $\mathcal{A}_2$, due to the introduction of the random numbers $\{r_1, r_2\}$ in the real world, which makes all subsequent values with randomness, and thus $\mathcal{A}_2$ cannot distinguish $View^{\Pi}_{real,\mathcal{A}_2}$ from $View^{f}_{ideal,\mathcal{S}im_2,\mathcal{L}_2}$ either. In summary, the iSHE- SC protocol is resistant to semi-honest adversaries.

The iSHE-SIP protocol securely computes the inner product of two encrypted vectors under a two-cloud environment, where the ideal function f can output the product of the two vectors under the simulation by $\mathcal{S}im_1$. In the real world, $\mathcal{A}_1$'s view consists of the encrypted vector $E(\overrightarrow{X})$, the pair $\{u, v\}$ received from C_2, and the randomly selected $\{\overrightarrow{R}, r\}$ by C_1, denoted as $View^{\Pi}_{real,\mathcal{A}_1}$; $\mathcal{A}_2$'s view includes the $\{\overrightarrow{R}, \overrightarrow{W}\}$ sent by C_1 and the encrypted vector $E(\overrightarrow{Y})$, denoted as $View^{\Pi}_{real,\mathcal{A}_2}$. In the ideal world, $\mathcal{S}im_2$ can randomly choose $\{\overrightarrow{R'}, \overrightarrow{W'}\}$ and output $\{\overrightarrow{R'}, \overrightarrow{W'}, E(\overrightarrow{Y})\}$, so $\mathcal{A}_2$'s view is denoted as $View^{f}_{ideal,\mathcal{S}im_2}$; in addition, $\mathcal{S}im_1$ also randomly selects u' and uses f to compute v', outputting $v' = r(E(\overrightarrow{X})E(\overrightarrow{Y}) - u')$, so $\mathcal{A}_1$'s view is denoted as $View^{f}_{ideal,\mathcal{S}im_1}$. Finally, similar to what is proved in the iSHE-SC protocol, since $\mathcal{A}_1$, $\mathcal{A}_2$ cannot distinguish between $\{\overrightarrow{R}, \overrightarrow{R'}\}, \{\overrightarrow{W}, \overrightarrow{W'}\}$, $\{u, u'\}$, and $\{v, v'\}$, it is also impossible to distinguish between $View^{\Pi}_{real,\mathcal{A}_1}$ and $View^{f}_{ideal,\mathcal{S}im_1}$ as well as $View^{\Pi}_{real,\mathcal{A}_2}$ and $View^{f}_{ideal,\mathcal{S}im_2}$.

B. Security of the PPSKSQ scheme

Our PPSKSQ scheme retrieves spatial data points that are within a rectangle and satisfy the Jaccard similarity requirement, similar to the proofs of the iSHE-SC and iSHE-SIP protocols, again utilizing the ideal-real security model to

analyze the scheme as adaptively secure. For the PPSKSQ scheme, all leakage occurs in C_1 and C_2. Therefore, we define the leakage function of the scheme as follows.

• $\mathcal{L}(C_1)$: In the *Registration and Initialization* phase, it learns the public key pk. In the *Outsourcing and Index Build* phase, it receives from the DP the encrypted spatial dataset $E(\mathcal{D})$, the encrypted sum of the set of keywords in the dataset $E(Sum_{W_i})$, the encrypted tree $E(T)$, and the encrypted keyword set $C(W_i)$. In the *Token Generation* phase, C_1 receives the query token Tk_1 from u_i. In the *Result Recovery* phase, C_1 leaks the result set $E(R)$. Thus, C_1 can know the number of dataset sizes, the access pattern of $E(T)$, and the query pattern for query range and keyword similarity.

• $\mathcal{L}(C_2)$: In the *Registration and Initialization* phase, it can learn the secret key sk. In the *Token Generation* phase, C_2 receives the query token Tk_2 from u_i.

Next, we analyze the flow of the PPSKSQ scheme in the ideal-real world based on the leakage functions $\mathcal{L}(C_1)$ and $\mathcal{L}(C_2)$.

Ideal world. In the ideal world, there exist two simulators, $\{\mathcal{S}im_1, \mathcal{S}im_2\}$, each possessing $\mathcal{L}(C_1)$ and $\mathcal{L}(C_2)$ respectively. Additionally, there are two semi-honest adversaries, $\{\mathcal{A}_1, \mathcal{A}_2\}$. In this scenario, the construction of the simulators and adversaries is as follows.

• *Registration and Initialization.* In this phase, the simulator calls $iSHE.KeyGen()$ to output the public key pk, set the secret key sk, and send sk to $\mathcal{A}_2$.

• *Outsourcing and Index Build.* In this phase, based on the leakage in $\mathcal{L}(C_1)$, $\mathcal{S}im_1$ constructs a simulated version of the encrypted tree structure, denoted as $E(T)^{\mathcal{S}im}$. Specifically, since the internal nodes of the tree are sub-rectangles of the quadtree dividing the two-dimensional space, it includes the x and y directions of the rectangles and is encoded and stored in the tree by the DP in advance in the PPSKSQ scheme. Therefore, $\mathcal{S}im_1$, based on the representation of sub-rectangles, randomly selects four numbers $\{R_i^{(x),\mathcal{S}im} = (x_i^{l,\mathcal{S}im}, x_i^{r,\mathcal{S}im}), R_i^{(y),\mathcal{S}im} = (y_i^{l,\mathcal{S}im}, y_i^{r,\mathcal{S}im})\}$ to represent each sub-rectangle for every internal node. As for the leaf nodes, which store spatial data information, $\mathcal{S}im_1$ randomly selects two numbers to replace the information of each location point, similar to the method for internal nodes. Additionally, the sum of the keyword set is handled similarly. Moreover, for an n-dimensional keyword set W_i, $\mathcal{S}im_1$ randomly selects an n-dimensional vector to replace it, denoted as $W_i^{\mathcal{S}im}$. Following this, $\mathcal{S}im_1$ constructs a diagonal matrix $U_{W_i}^{\mathcal{S}im}$ based on $W_i^{\mathcal{S}im}$, and also randomly constructs an upper triangular matrix $V_{W_i}^{\mathcal{S}im}$.

It is worth noting that for all the data encrypted with iSHE, $\mathcal{S}im_1$ utilizes pk and chooses random numbers $r_1, r_2 \in \{0,1\}^{k_2}$ to compute its ciphertext by using Eq. (2). Finally, $\mathcal{S}im_1$ sends all the results simulated at this phase to $\mathcal{A}_1$.

• *Token Generation.* At this phase, $\mathcal{S}im_1$ and $\mathcal{S}im_2$ construct simulated versions of $Tk_1^{\mathcal{S}im}$ and $Tk_2^{\mathcal{S}im}$, respectively, based on $\mathcal{L}(C_1)$ and $\mathcal{L}(C_2)$. For $E(Sum_{W_q})$ in Tk_1, $\mathcal{S}im_1$ also chooses a random number to replace it, denoted as $E(Sum_{W_q})^{\mathcal{S}im}$. Similarly, for Tk_2, $\mathcal{S}im_2$ chooses two 4-dimensional vectors instead of Tk_2, respectively. In addition, for the keyword request of the query, similar to the *Outsourcing and Index Build* phases, $\mathcal{S}im_1$ goes ahead and constructs a diagonal matrix $U_{W_q}^{\mathcal{S}im}$ and an upper triangular matrix $V_{W_q}^{\mathcal{S}im}$. Finally, $\mathcal{S}im_1$ and $\mathcal{S}im_2$ send the simulated information to the corresponding $\mathcal{A}_1$ and $\mathcal{A}_2$, respectively.

Moreover, if it is assumed that there is collusion between C_1 and C_2 at this point, a query token for u_i is not obtained. Even though C_2 holds the secret key sk and can recover $E(Sum_{W_q})$ and Tk_2 uploaded by u_i, u_i randomly chooses a positive random number β to obfuscate the original vectors before encrypting $\{E(\overrightarrow{P_{R_q}^{(x)'}})\}$ and $\{E(\overrightarrow{P_{R_q}^{(y)'}})\}$. Consequently, cloud servers cannot determine the user's query range vectors. Regarding $E(Sum_{W_q})$, although it is not obscured with a random number before encryption, the cloud servers are unaware of the mapping relationship of the set of keywords. During the query process, when calculating the similarity between the keywords, the trace of the point $E(tr_{\hat{d}_i})$ contains the random number α chosen by the DP. Thus, even if the cloud servers can decrypt the encrypted trace, they cannot deduce the query token uploaded by u_i.

• *Result Recovery.* In the result recovery phase, $\mathcal{S}im_1$ generates a series of random numbers $r_{\hat{d}}^{\mathcal{S}im}$ and $r_{W_{\hat{d}}}^{\mathcal{S}im}$ and uses them to compute the simulation views, denoted as $\{E(X_{\hat{d}})^{\mathcal{S}im}, E(Y_{\hat{d}})^{\mathcal{S}im}, E(K_{\hat{d}})^{\mathcal{S}im}\}$, respectively. Finally, $\mathcal{S}im_1$ sends them to $\mathcal{A}_1$.

Real world. The two semi-honest adversaries $\{\mathcal{A}_1, \mathcal{A}_2\}$ exist in the real world and the real-world view is consistent with the construction process of our PPSKSQ scheme, so we don't expand the discussion in detail here.

Theorem 1: *If the iSHE scheme has CPA security, then our PPSKSQ scheme can achieve adaptive security.*

Proof: The PPSKSQ scheme can be considered adaptively secure if, in the ideal-real world, the simulators $\mathcal{S}im_1$ and $\mathcal{S}im_2$ ensure that the adversaries $\mathcal{A}_1$ and $\mathcal{A}_2$ have a negligible advantage in distinguishing between the ideal and real environments. In the *Outsourcing and Index Build* phase, for each spatial point $\{p_i = (x_i, y_i)\}$ after each encoding, DP uses a positive random number α once to obfuscate the value of the original vector before encrypting $\{\overrightarrow{P_i^{(x)'}}, \overrightarrow{P_i^{(y)'}}\}$. Then, DP utilizes the iSHE encryption scheme to encrypt the obfuscated vectors as $\{E(\overrightarrow{P_i^{(x)'}}), E(\overrightarrow{P_i^{(y)'}})\}$. For the constructed quadtree T, DP does the same as above. In addition, for the n-dimensional keyword set W_i, DP first obfuscates it by utilizing a random number α as well. Then, the permutation function π is applied to the obfuscated keyword set, denoted as $\overline{W_i}$. Next, DP transforms $\overline{W_i}$ into a diagonal matrix U_{W_i} and randomly generates an upper triangular matrix V_{W_i}. Finally, the set of keywords is encrypted using $\{M_1, M_2\}$, denoted as $C(W_i) = M_1 V_{W_i} U_{W_i} M_2$.

It is easy to see that for the spatial data, $\mathcal{A}_1$ cannot observe or guess the random number α, the substitution function π, and the randomly selected upper triangular matrix V_{W_i}. Moreover, the data are encrypted by utilizing iSHE as well as $\{M_1, M_2\}$ are unknown to $\mathcal{A}_1$. Similarly, in the *Token Generation* phase, the upper triangular matrix V_{W_q} and the secret key $\{M_1^{-1}, M_2^{-1}\}$ are not known to $\mathcal{A}_1$. Therefore, due to our construction and the CPA security of the iSHE scheme,

$\mathcal{A}_1$ is unable to distinguish between the ideal and the real world.

For $\mathcal{A}_2$, a query token $Tk_2 = \{E(\overrightarrow{P_{R_i}^{(x)'}}), E(\overrightarrow{P_{R_i}^{(y)'}})\}$ from u_i is received, and in addition, $\mathcal{A}_2$ has to run the protocols iSHE-SC and iSHE-SIP in collaboration with $\mathcal{A}_1$, the proofs of both of which have been discussed in VI-A and VI-B, respectively. The fact that $\mathcal{A}_2$ does not know the random number β chosen by u_i, as well as the semantic security of the security protocols and iSHE, makes it equally impossible for $\mathcal{A}_2$ to distinguish between the ideal and the real world.

VII. PERFORMANCE EVALUATION

In this subsection, we thoroughly analyze the performance of the proposed PPSKSQ scheme and compare it with existing spatial keyword similarity schemes [9], [28], [42]. All experiments are conducted on a machine with an Intel(R) Core(TM) i5-8250U CPU @ 1.80 GHz, 4.00 GB of RAM, and the Windows 11 operating system. The experimental environment is implemented in Java language with JDK11. For the iSHE symmetric homomorphic encryption used in the scheme, we set its security parameters to $k_0 = 4096$, $k_1 = 40$, and $k_2 = k_r = 160$ to achieve a higher security goal. The experimental data is selected from the real dataset Yelp[1] and we use its business data. The dataset contains 21,900 entries, with the maximum size of the keyword sets being 27 and the keyword dictionary consisting of 1,123 terms. A subset of the data was selected and preprocessed in advance. Finally, we assess the scheme's performance by analyzing communication overhead, computational overhead, and experimental results.

A. Communication overhead

The comparison between our PPSKSQ scheme and the schemes [9], [28], [42] is shown in Table II. In our scheme, the main communication overhead arises during the iSHE-SC protocol, iSHE-SIP protocol, *Outsourcing and Index Build*, *Token Generation*, and *Query* phases. In the iSHE-SC and iSHE-SIP protocols, communication between cloud servers are restricted to the exchange of ciphertexts and vectors, which leads to a communication overhead characterized by $2\,|C|$ for the former protocol and $8\,|C| + |V_4|$ for the latter. During the *Outsourcing and Index Build* phase, which actually includes two processes, namely data outsourcing and index building, the stage involves transferring all information of the encrypted dataset and building an encrypted quadtree. The overhead is slightly larger than schemes that do not build a tree, but this is acceptable because the quadtree we construct aims for higher efficiency during the query phase. Hence, the total communication overhead for this phase is $(9+\frac{4(4^{h-1}-1)}{3})\,|C|+|M_{n*n}|$. In the *Token Generation* phase, although two tokens are generated, the overhead only relates to the size of the ciphertext and matrix, thus making the communication overhead for the two tokens $9\,|C| + |M_{n*n}|$. Furthermore, during the Query phase, which includes two processes, namely range queries and keyword similarity queries, the former incurs a communication overhead of $(h-1)(8\,|C| + |V_4|)$, and the latter $2(\hat{d}-1)\,|C|$.

[1]Yelp dataset, 2020. [Online]. Available:: https://www.yelp.com/dataset

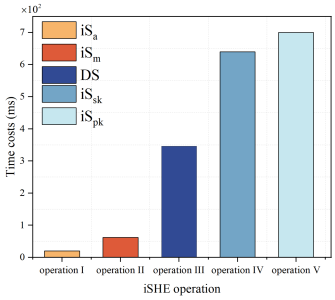


Fig. 6. Time cost of each iSHE operation

B. Computation overhead

The comparison of computational overhead between schemes is shown in Table III. Similar to the analysis in Subsection VII-A, for the computational overhead, we also go to evaluate these parts. In the PPSKSQ scheme, the computational overhead is mainly caused by the iSHE encryption algorithm. Therefore, before the evaluation, the time taken by the most time-consuming operations $(iS_a, iS_m, DS, iS_{sk}, iS_{pk})$ in iSHE is shown in Fig. 6. It can be seen that the efficiency of homomorphic addition and homomorphic multiplication is fast to perform. In addition, public key encryption and secret key encryption are the most time-consuming in iSHE compared to decryption operations.

• **iSHE-SC and iSHE-SIP protocols.** The operations of both protocols are homomorphic multiplication and homomorphic addition constructs, the only difference being that only C_1 in the first step of the iSHE-SC protocol requires cryptographic operations, while the iSHE-SIP protocol does not involve any cryptographic operations. Thus, the two protocols are theoretically very efficient. Their total computational overhead is $2iS_{pk} + DS + 2iS_a + 5iS_m$ for the former and $11iS_a + 13iS_m + DS$ for the latter.

• **Outsourcing and Index Build.** The data outsourcing phase requires encrypting the entire dataset and constructing the encryption tree, thus its overhead is related to the tree height and the size of the dataset. However, for the tree height, we can utilize the homomorphism of iSHE to encrypt the entire quadtree in an extended manner, which we consider the most complicated case. In the index construction phase, there are three matrix multiplication operations, but the matrices selected in our scheme are upper triangular matrices, and thus all the computations are not required at the time of computation. So, the computational overhead of the entire phase is $(9 + \frac{8(4^{h-1}-1)}{3})iS_{sk} + 3T_{M_{n*n}}$.

• **Token Generation.** Since u_i uploads the encrypted rectangular range, public key encryption is used here. In addition, for the subsequent keyword similarity queries, u_i also needs to be similar to the three matrix multiplication operations in the outsourced indexing phase. Therefore, the total computational overhead of the token generation phase is $9iS_{pk} + 3T_{M_{n*n}}$.

• **Query.** In the range queries, the cloud servers recursively go through the tree nodes to search whether they intersect with the query ranges or not, and at this point, the iSHE-SIP protocol is used. Therefore, in this small stage, the computational overhead is closely related to the iSHE-SIP

TABLE II
COMMUNICATION OVERHEAD

Scheme	Outsourcing and Index Build	Token Generation	Query
TSKS	$\|Z\|$	$2\|Z\|$	$\hat{d}\|V_{\eta+n+3}\|$
MRSF	$\|M_{n*n}\| + \|M_{d'*d'}\|$	$\|M_{d'*d'}\|$	$\|M_{\hat{d}*\hat{d}}\| + \|V_{\hat{d}}\|$
PBRQ-T+	$(1+3d')\|G\|$	$(1+2kd'+4d')\|G\|$	$(3+k)\|G_T\|$
PPSKSQ	$(9+\frac{4(4^{h-1}-1)}{3})\|C\| + M_{n*n}$	$9\|C\| + \|M_{n*n}\|$	$(h-1)(9\|C\| + \|V_4\|) + 2\hat{d}\|C\|$

TABLE III
COMPUTATION OVERHEAD

Scheme	Outsourcing and Index Build	Token Generation	Query
MRSF	$T_{TFIDF} + 2T_{M_{d'*d'}}$	$2T_{M_{d'*d'}}$	$2T_{M_{d'*d'}}$
TSKS	$4T_{M_{(\eta+n+3)*(\eta+n+3)}}$	$4T_{M_{(\eta+n+3)*(\eta+n+3)}}$	$2T_{M_{(\eta+n+3)*(\eta+n+3)}}$
PBRQ-T+	$28T_E + \frac{4(4^{h-1}-1)}{3}$	$(4d'k+4d'+2k)T_E + T_S$	$(\log_2 L)(2d'k+2d')T_P + 2d'T_P$
PPSKSQ	$(9+\frac{8(4^{h-1}-1)}{3})iS_{sk} + 3T_{M_{n*n}}$	$9iS_{pk} + 3T_{M_{n*n}}$	$(h-1)(25iS_m + DS)$ $\hat{d}T_{V_4} + (3\hat{d}-2)iS_{pk} + (\hat{d}-1)(DS+7iS_m)$

TABLE IV
COMPARISON WITH EXISTING SCHEMES

Schemes	SAGTree [10]	PBRQ-T+ [28]	PRSQ-F [30]	MRSF [42]	GRQ+MSSAC [13]	PPTR-F [12]	PSDQ+ [14]	TSKS [9]	PPSKSQ
Keyword Queries	Boolean	Boolean	TF-IDF	TF-IDF	TF-IDF	–	–	–	Jaccard
Security	IND-CPA	IND-SCPA	IND-SCPA	IND-CLS-CPA	IND-SCPA	IND-CPA	IND-CPA	IND-CPA	IND-CPA
Sublinear Efficiency	✓	×	✓	✓	✓	✓	✓	×	✓
Scalability	✓	×	✓	×	✓	×	✓	×	✓

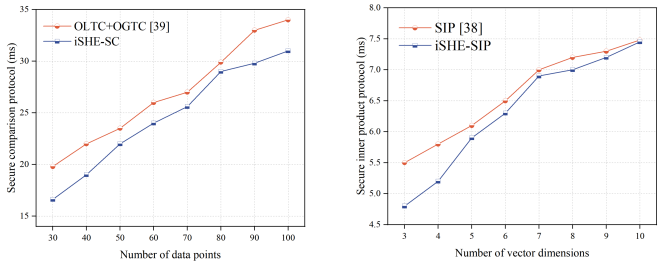
Noteworthy: – indicates no similar.

protocol. In the keyword similarity queries, at this point, the leaf nodes of the tree have been filtered, and to filter out the spatial points with the highest similarity, the cloud servers use the iSHE-SC protocol to compare the similarity size in the candidate set. In summary, the computational overhead of the former is $(h-1)(11iS_a + 14iS_m + DS)$; the total computational overhead required in the latter query phase is $\hat{d}T_{V_4} + (3\hat{d}-2)iS_{pk} + (\hat{d}-1)(DS + 2iS_a + 5iS_m)$.

In summary, the scheme [42] employs the secure kNN algorithm for encrypted data retrieval; the matrix dimension of the scheme [28] is significantly larger than that of our scheme, and the scheme [9] relies on pairwise operations to generate a large amount of gray code for range queries. Since our PPSKSQ scheme leverages efficient iSHE and lightweight matrix multiplication through a tree structure for filtered queries, it can significantly reduce computational overhead. In subsequent experimental evaluations, we will examine the impact of tree height on the scheme's performance to identify the optimal height that minimizes computational overhead.

C. Experimental results

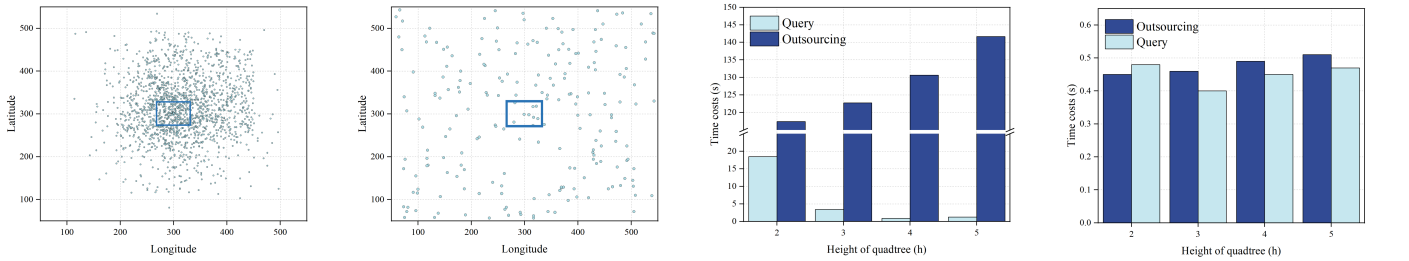
Firstly, we evaluate the performance of both iSHE-SC and iSHE-SIP protocols. Secondly, our PPSKSQ scheme is compared in detail with existing privacy-preserving keyword query schemes at a macro level in Table IV. It is clear that our approach uniquely utilizes the Jaccard index to assess the similarity between sets of keywords. Then, we measure the impact of quadtree height on the partitioning of dense and sparse maps to select the optimal tree height, thus achieving the best query efficiency for our scheme. Finally, we make a detailed comparison with the three schemes [28], [9], and [42].



(a) Vector dimension is 4 (b) Keyword set size is 4

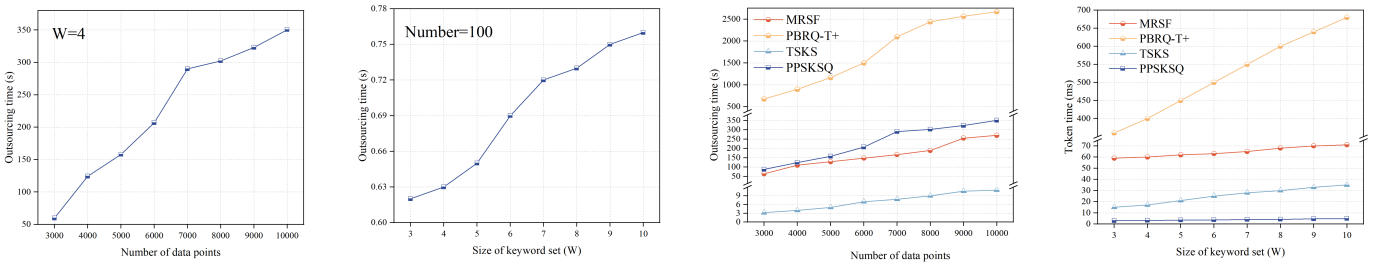
Fig. 7. Time cost of iSHE-SC and iSHE-SIP protocol.

• **iSHE-SC and iSHE-SIP protocols.** Both protocols are implemented on two cloud servers. The iSHE-SC protocol compares the sizes of two numbers in ciphertext, whereas the iSHE-SIP protocol securely computes the inner product of two vectors. Consequently, the performance of the former is influenced by the number of spatial data points, while the dimensionality of the vectors impacts the latter's performance. In Fig. 7, we show a comparison between the iSHE-SC protocol and the secure comparison protocol designed by [36], as well as between the iSHE-SIP protocol and the secure inner product protocol designed by [35]. Fig. 7a indicates that the time cost of the iSHE-SC protocol escalates with the increase in the number of data points, maintaining a constant vector dimension of 4. Fig. 7b demonstrates that the time cost of the iSHE-SIP protocol gradually increases with the vector dimension when the size of the keyword set W is constant



(a) 5018 location points (b) 214 location points (c) Performance of PPSKSQ with Dense datasets (d) Performance of PPSKSQ with sparse datasets

Fig. 8. Effect of quadtree tree height on the distribution of different datasets. (a) Dense datasets. (b) Sparse datasets. (c) Time cost of querying and outsourcing under dense datasets. (d) Time cost of querying and outsourcing under sparse datasets.



(a) Keyword set size is 4 (b) Spatial data points are 100 (c) Varying with data points (d) Varying with keyword set

Fig. 9. Time cost of Outsourcing and Index Build, Token Generation. (a) Impact of the number of data points on the time cost of outsourcing. (b) Impact of the size of keyword set on the time cost of outsourcing. (c) Cost of outsourcing time between schemes. (d) Cost of Token time between schemes.

at 4, showing that when the vector dimension reaches 8, the protocol only requires about 7ms.

• **Height of the quadtree.** To comprehensively demonstrate the flexibility of the PPSKSQ scheme, we examine the impact of tree height h on both dense and sparse datasets. Furthermore, the blue box in the diagram is defined as the query range u_i. Fig. 8a illustrates a scenario where 5,018 location points are concentrated in a dense distribution, whereas Fig. 8b depicts only 214 location points scattered across a sparse distribution. This is also very practical to consider, because sometimes the space area is bustling, and sometimes it is very scattered.

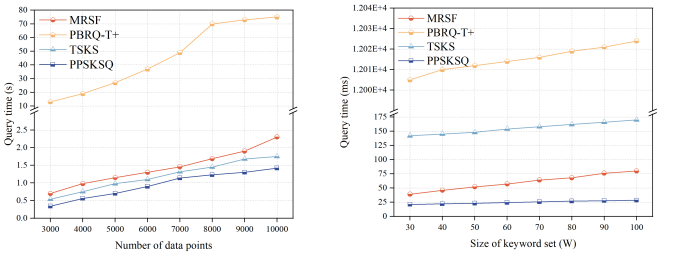
Tree height primarily influences the outsourcing and query phases, and our objective is to optimize h for maximum query efficiency. We test the impact of tree height h at partition levels 2, 3, 4, and 5 on outsourcing and querying, respectively. Fig. 8c illustrates the influence of tree height on the outsourcing and querying phases under a dense dataset. It is observed that the time cost of the outsourcing stage incrementally increases with higher levels of tree division, although the differences are not significant. In the query phase, we can find that when the tree is divided into four levels, the query efficiency is optimal and only needs 0.68s, which is significantly faster than the two and three levels. Fig. 8d examines the balance of tree height in a sparse dataset. The time cost of the outsourcing stage is observed to still increase with tree height, yet remains generally stable. During the querying phase, it is determined that a tree height of 3 yields optimal querying efficiency, requiring only 0.4s.

Therefore, various quadtree heights are selected based on the number of data points and the degree of distribution to optimize query efficiency. In subsequent experiments, we choose the case where the tree height is optimal to evaluate the performance at different stages.

• **Outsourcing and Index Build.** At this phase, it is necessary to construct an encrypted quadtree with all the data points in the two-dimensional space. In Fig. 9, the impact of the number of spatial data points and the size of the keyword set on the PPSKSQ scheme is evaluated. Fig. 9a shows that with an increasing number of spatial data points, the complexity of the PPSKSQ scheme also increases when the keyword set size W is set at 4. This increase in complexity is attributable to the extended time required for constructing the tree as the number of data points grows. Fig. 9b demonstrates that when the number of spatial data points p_i is fixed at 100, the PPSKSQ scheme exhibits a gradual increase in complexity as W expands. Additionally, Fig. 9c shows a comparison of our scheme with other schemes. As p_i increases, it is observed that the time required to construct the tree also escalates, resulting in a higher outsourcing time for our scheme compared to the MRSF and TSKS schemes. This is tolerable because building a tree is a time-consuming operation, but it significantly improves efficiency during the query phase. Furthermore, we can find that our scheme is significantly faster than PBRQ-T+ by several tens of folds.

• **Token Generation.** The token generation phase is performed by u_i, u_i selects a rectangular range based on the query requirements, encodes and encrypts this range, and sets the set

of query keywords, which are encrypted using the secret key $\{M_1^{-1}, M_2^{-1}\}$. In the previous theoretical analysis, it is not difficult to find that the most important factor affecting this phase is the size of the query keyword set by u_i. Therefore, we evaluate the time consumption of the token generation phase by adjusting the size of W in Fig. 9d. Experimental results demonstrate that as the number of keywords in W increases, the PPSKSQ scheme requires only approximately 1.5ms, whereas the MRSF, TSKS, and PBRQ-T+ schemes require approximately 65ms, 25ms, and 0.5s, respectively. This is because, in the token generation phase, u_i only needs to select a rectangular range and set of query keywords and perform their encoding encryption, and these operations were fast in the previous analyses. In summary, the time cost of our scheme during the token generation phase is significantly lower compared to the MRSF, TSKS, and PBRQ-T+ schemes.



(a) Varying with data points (b) Varying with keyword set

Fig. 10. Time cost of Query. (a) Cost of query time between schemes under the impact of spatial data points. (b) Cost of query time between schemes under the impact of keyword sets.

• **Query.** The query phase is conducted collaboratively by two cloud servers, C_1 and C_2, who initially perform a range query to identify spatial points within a specified range. Subsequently, a keyword similarity search and sorting are conducted on these points. Finally, the spatial data points that conform to the range and have the highest similarity are returned to the u_i. Therefore, in this phase, we first measure the time cost of the PPSKSQ scheme in range queries and then compare the performance in similarity queries. Fig. 10 shows the performance of the two steps in the query phase separately. In Fig. 10a, the impact of increasing numbers of spatial data points on various schemes is compared. In this test, it can reflect the performance of our scheme in the range queries, and we assume that the tree height $h = 5$. It is observed that the index tree structure, constructed during the outsourcing indexing phase, allows for efficient searching and filtering despite the initial non-optimal time cost, thereby enabling quick access to leaf nodes. When handling 10,000 spatial points, our scheme requires approximately 1.42s, compared to approximately 2.3s for the MRSF scheme, 1.75s for the TSKS scheme, and 75s for the PBRQ-T+ scheme. Our scheme is significantly superior to PBRQ-T+ and MRSF schemes and is close to TSKS schemes. However, the TSKS scheme does not consider the similarity between keyword sets, thereby the PPSKSQ scheme has strong scalability.

In addition, in Fig. 10b, we measure the comparison of the performance between the schemes as the size of the keyword set increases. In this phase, it can reflect the performance of keyword similarity queries of the PPSKSQ scheme. It can be seen that our scheme is significantly better than all the schemes. Moreover, when the keyword similarity query stage is carried out, it means that there are $\hat{d}$ spatial data points that satisfy the query range. Here, we set the $\hat{d}$ constant to 10, but in fact, we can already achieve good efficiency in the range queries phase, and the corresponding $\hat{d}$ will be less. So we have considered the general case of our scheme, which in reality would be faster in terms of query efficiency. In summary, the PPSKSQ scheme can efficiently filter the spatial data points that match the range and have the highest similarity in the query phase.

VIII. CONCLUSION

In this paper, we have proposed a Privacy-Preserving Keyword Similarity Query scheme (PPSKSQ) that enables range queries and keyword similarity metrics while preserving user privacy. Furthermore, we utilize the Jaccard metric to measure the similarity between keyword sets. Specifically, we first design two protocols based on the iSHE encryption scheme, namely iSHE-SC and iSHE-SIP, which serve as building blocks of our scheme to realize the size comparison of ciphertexts as well as secure inner product computation between two vectors. With the above building blocks, for range queries, we encode the location information into vectors and combine them with the efficient filtering of quadtree and iSHE-SIP protocols, which can quickly achieve range queries. For keyword similarity queries, we combine lightweight matrix encryption and iSHE-SC protocol to sort the matrix traces and thus filter the spatial location points with the highest similarity. Finally, through security analysis and experimental evaluation, we demonstrate the adaptive security, efficiency, flexibility, and scalability of our PPSKSQ scheme. In future work, as users may show greater interest in specific keyword information, we will consider the effect of weights on the set of keywords to improve the flexibility of the scheme even further. Moreover, we will explore protecting access patterns to improve the security of our scheme.

REFERENCES

[1] D. Li, J. Wu, J. Le, X. Liao, and T. Xiang, "A novel privacy-preserving location-based services search scheme in outsourced cloud," *IEEE Transactions on Cloud Computing*, 2021.

[2] Z. Li, J. Ma, Y. Miao, X. Wang, J. Li, and C. Xu, "Enabling efficient privacy-preserving spatio-temporal location-based services for smart cities," *IEEE Internet of Things Journal*, 2023.

[3] Y. Cheng, J. Ma, Z. Liu, Y. Wu, K. Wei, and C. Dong, "A lightweight privacy preservation scheme with efficient reputation management for mobile crowdsensing in vehicular networks," *IEEE Transactions on Dependable and Secure Computing*, 2022.

[4] D. Zhu, H. Zhu, X. Wang, R. Lu, and D. Feng, "Efficient and privacy-preserving similar patients query scheme over outsourced genomic data," *IEEE Transactions on Cloud Computing*, vol. 11, no. 2, pp. 1286–1302, 2021.

[5] M. Li, Y. Chen, S. Zheng, D. Hu, C. Lal, and M. Conti, "Privacy-preserving navigation supporting similar queries in vehicular networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 2, pp. 1133–1148, 2020.

[6] B. Baruah and S. Dhal, "A security and privacy preserved intelligent vehicle navigation system," *IEEE Transactions on Dependable and Secure Computing*, vol. 20, no. 2, pp. 944–959, 2022.

[7] X. Zuo, L. Li, H. Peng, S. Luo, and Y. Yang, "Privacy-preserving subgraph matching scheme with authentication in social networks," *IEEE Transactions on Cloud Computing*, vol. 10, no. 3, pp. 2038–2049, 2020.
[8] L. Wu, C. Qin, Z. Xu, Y. Guan, and R. Lu, "Tcpp: Achieving privacy-preserving trajectory correlation with differential privacy," *IEEE Transactions on Information Forensics and Security*, 2023.
[9] Y. Yang, Y. Miao, Z. Ying, J. Ning, X. Meng, and K.-K. R. Choo, "Privacy-preserving threshold spatial keyword search in cloud-assisted iiot," *IEEE Internet of Things Journal*, vol. 9, no. 18, pp. 16 990–17 001, 2021.
[10] N. Cui, X. Yang, Y. Chen, J. Li, B. Wang, and G. Min, "Secure boolean spatial keyword query with lightweight access control in cloud environments," *IEEE Internet of Things Journal*, vol. 9, no. 12, pp. 9503–9514, 2021.
[11] X. Tu, H. Bao, R. Lu, C. Huang, and H.-N. Dai, "Pmrk: Privacy-preserving multidimensional range query with keyword search over spatial data," *IEEE Internet of Things Journal*, 2023.
[12] C. Zhang, L. Zhu, C. Xu, J. Ni, C. Huang, and X. Shen, "Location privacy-preserving task recommendation with geometric range query in mobile crowdsensing," *IEEE Transactions on Mobile Computing*, vol. 21, no. 12, pp. 4410–4425, 2021.
[13] F. Song, Z. Qin, L. Xue, J. Zhang, X. Lin, and X. Shen, "Privacy-preserving keyword similarity search over encrypted spatial data in cloud computing," *IEEE Internet of Things Journal*, vol. 9, no. 8, pp. 6184–6198, 2021.
[14] Y. Miao, Y. Yang, X. Li, Z. Liu, H. Li, K.-K. R. Choo, and R. H. Deng, "Efficient privacy-preserving spatial range query over outsourced encrypted data," *IEEE Transactions on Information Forensics and Security*, 2023.
[15] Z. Xu, L. Wu, C. Qin, S. Li, S. Zhang, and R. Lu, "Pritaec: Privacy-preserving task assignment based on oblivious transfer and edge computing in vanet," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 4, pp. 4996–5009, 2022.
[16] X. Zhao, L. Gan, and K. Fan, "Privacy-preserving spatial keyword search with lightweight access control in cloud environments," *IEEE Internet of Things Journal*, 2023.
[17] X. Wang, J. Ma, F. Li, X. Liu, Y. Miao, and R. H. Deng, "Enabling efficient spatial keyword queries on encrypted data with strong security guarantees," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 4909–4923, 2021.
[18] X. Wang, J. Ma, X. Liu, R. H. Deng, Y. Miao, D. Zhu, and Z. Ma, "Search me in the dark: Privacy-preserving boolean range query over encrypted spatial data," in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*. IEEE, 2020, pp. 2253–2262.
[19] H. Ren, H. Li, D. Liu, G. Xu, N. Cheng, and X. Shen, "Privacy-preserving efficient verifiable deep packet inspection for cloud-assisted middlebox," *IEEE Transactions on Cloud Computing*, vol. 10, no. 2, pp. 1052–1064, 2020.
[20] F. Sun, J. Yu, and J. Hu, "Privacy-preserving approximate minimum community search on large networks," *IEEE Transactions on Information Forensics and Security*, 2024.
[21] S. Zhang, S. Ray, R. Lu, Y. Zheng, Y. Guan, and J. Shao, "Towards efficient and privacy-preserving user-defined skyline query over single cloud," *IEEE Transactions on Dependable and Secure Computing*, vol. 20, no. 2, pp. 1319–1334, 2022.
[22] Y. Guan, R. Lu, Y. Zheng, S. Zhang, J. Shao, and G. Wei, "Achieving efficient and privacy-preserving (,)-core query over bipartite graphs in cloud," *IEEE Transactions on Dependable and Secure Computing*, 2022.
[23] C. Hu, C. Zhang, D. Lei, T. Wu, X. Liu, and L. Zhu, "Achieving privacy-preserving and verifiable support vector machine training in the cloud," *IEEE Transactions on Information Forensics and Security*, 2023.
[24] Y. Luo, S. Fu, D. Wang, M. Xu, and X. Jia, "Efficient and generalized geometric range search on encrypted spatial data in the cloud," in *2017 IEEE/ACM 25th International Symposium on Quality of Service (IWQoS)*. IEEE, 2017, pp. 1–10.
[25] G. Xu, H. Li, Y. Dai, K. Yang, and X. Lin, "Enabling efficient and geometric range query with access control over encrypted spatial data," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 4, pp. 870–885, 2018.
[26] B. Wang, M. Li, and L. Xiong, "Fastgeo: Efficient geometric range queries on encrypted spatial data," *IEEE transactions on dependable and secure computing*, vol. 16, no. 2, pp. 245–258, 2017.
[27] X. Li, T. Xiang, S. Guo, H. Li, and Y. Mu, "Privacy-preserving reverse nearest neighbor query over encrypted spatial data," *IEEE Transactions on Services Computing*, vol. 15, no. 5, pp. 2954–2968, 2021.
[28] Q. Tong, X. Li, Y. Miao, X. Liu, J. Weng, and R. H. Deng, "Privacy-preserving boolean range query with temporal access control in mobile computing," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 5, pp. 5159–5172, 2022.
[29] S. Zhang, S. Ray, R. Lu, Y. Guan, Y. Zheng, and J. Shao, "Efficient and privacy-preserving spatial keyword similarity query over encrypted data," *IEEE Transactions on Dependable and Secure Computing*, 2022.
[30] Q. Tong, Y. Miao, H. Li, X. Liu, and R. H. Deng, "Privacy-preserving ranked spatial keyword query in mobile cloud-assisted fog computing," *IEEE Transactions on Mobile Computing*, vol. 22, no. 6, pp. 3604–3618, 2021.
[31] X. Li, L. Bai, Y. Miao, S. Ma, J. Ma, X. Liu, and K.-K. R. Choo, "Privacy-preserving top-k k spatial keyword queries in fog-based cloud computing," *IEEE Transactions on Services Computing*, vol. 16, no. 1, pp. 504–514, 2021.
[32] C. Xu, N. Wang, L. Zhu, C. Zhang, K. Sharif, and H. Wu, "Reliable and privacy-preserving top-k disease matching schemes for e-healthcare systems," *IEEE Internet of Things Journal*, vol. 9, no. 7, pp. 5537–5547, 2021.
[33] G. Liu, G. Yang, S. Bai, H. Wang, and Y. Xiang, "Fase: A fast and accurate privacy-preserving multi-keyword top-k retrieval scheme over encrypted cloud data," *IEEE Transactions on Services Computing*, vol. 15, no. 4, pp. 1855–1867, 2020.
[34] S. Zhang, R. Lu, H. Zhu, Y. Zheng, Y. Guan, F. Wang, J. Shao, and H. Li, "Performance enhanced secure spatial keyword similarity query with arbitrary spatial ranges," *IEEE Transactions on Information Forensics and Security*, 2024.
[35] J. Lei, Q. Pei, Y. Wang, W. Sun, and X. Liu, "Privface: fast privacy-preserving face authentication with revocable and reusable biometric credentials," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 5, pp. 3101–3112, 2021.
[36] Y. Zheng, R. Lu, H. Zhu, S. Zhang, Y. Guan, J. Shao, F. Wang, and H. Li, "Setrknn: Efficient and privacy-preserving set reverse knn query in cloud," *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 888–903, 2022.
[37] J. Chen, L. Liu, R. Chen, W. Peng, and X. Huang, "Secrec: A privacy-preserving method for the context-aware recommendation system," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 5, pp. 3168–3182, 2021.
[38] X. Wang, J. Ma, Y. Miao, X. Liu, D. Zhu, and R. H. Deng, "Fast and secure location-based services in smart cities on outsourced data," *IEEE Internet of Things Journal*, vol. 8, no. 24, pp. 17 639–17 654, 2021.
[39] H. Zhang, Z. Guo, S. Zhao, and Q. Wen, "Privacy-preserving linear region search service," *IEEE Transactions on Services Computing*, vol. 14, no. 1, pp. 207–221, 2017.
[40] F. Wang, H. Zhu, G. He, R. Lu, Y. Zheng, and H. Li, "Efficient and privacy-preserving arbitrary polygon range query scheme over dynamic and time-series location data," *IEEE Transactions on Information Forensics and Security*, 2023.
[41] S. Kamara, P. Mohassel, and M. Raykova, "Outsourcing multi-party computation," *Cryptology ePrint Archive*, 2011.
[42] J. Li, J. Ma, Y. Miao, R. Yang, X. Liu, and K.-K. R. Choo, "Practical multi-keyword ranked search with access control over encrypted cloud data," *IEEE Transactions on Cloud Computing*, vol. 10, no. 3, pp. 2005–2019, 2020.

Changrui Wang received a B.S. degree in Computer Science and Technology from the School of Information Science and Engineering at Qilu Normal University in 2021. He is currently pursuing an M.S. degree in Computer Science and Technology at Shandong Normal University. His research interests include privacy preservation and query privacy.

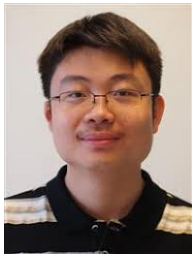
Lei Wu received his Ph.D. degree in applied mathematics from School of Mathematics, Shandong University in 2009. He is currently a professor with the School of Information Science and Engineering, Shandong Normal University, China. His research interests include applied cryptography, privacy preservation, and cloud computing security.

Lijuan Xu received Ph.D. degree from the School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China, in 2023. She is currently an Associate Professor with Shandong Computer Science Center (National Supercomputer Center in Jinan), China. Her main research interests include network security, industrial internet security, and computer forensics.

Haojie Yuan received a B.S. degree in Computer Science and Technology from the School of Information Science and Engineering at Shandong Normal University in 2022. He is currently pursuing an M.S. degree in Computer Science and Technology at Shandong Normal University. His research interests include privacy preservation and differential privacy.

Hao Wang received his Ph.D degree in Computer Science from Shandong University in 2012. He is currently a professor at Shandong Normal University. He has co-authored over 60 research papers in prestigious journals and conferences. His current research interests include applied cryptography, secure multi-party computation, and Blockchain.

Wenying Zhang received her Ph.D.degree in Cryptography in Department of Information Research, Information Engineering University of PLA in June 2004 in Zhengzhou, Henan, China. From July 2004 to September 2006, she was a Postdoctoral Fellow at the Institute of Software, Chinese Academy of Sciences, Beijing, China. She is now a professor and Ph.d. supervisor of Shandong Normal University. Her research interests include cryptography, Boolean function, hash function analysis and password security.



Weizhi Meng (Senior Member, IEEE) received the Ph.D. degree in Computer Science from the City University of Hong Kong. He is a Full Professor in the School of Computing and Communications, Lancaster University, United Kingdom, and an adjunct faculty in the Department of Applied Mathematics and Computer Science, Technical University of Denmark, Denmark. He was a recipient of the Hong Kong Institution of Engineers (HKIE) Outstanding Paper Award for Young Engineers/Researchers in both 2014 and 2017. He also received the IEEE ComSoc Best Young Researcher Award for Europe, Middle East, & Africa Region (EMEA) in 2020 and the IEEE ComSoc Communications & Information Security (CISTC) Early Career Award in 2023. His primary research interests are blockchain technology, cyber security and artificial intelligence in security including intrusion detection, blockchain applications, smartphone security, biometric authentication, and IoT security.