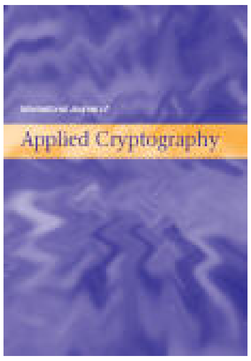


International Journal of Applied Cryptography

ISSN online: 1753-0571 - ISSN print: 1753-0563
https://www.inderscience.com/ijact

WeightedSim: privacy-preserving weighted similarity query over encrypted healthcare data

Guojun Tang, Rongxing Lu, Mohammad Mamun

DOI: 10.1504/IJACT.2023.10063607

Article History:

Received:	14 January 2023
Last revised:	04 April 2023
Accepted:	14 May 2023
Published online:	03 May 2024

WeightedSim: privacy-preserving weighted similarity query over encrypted healthcare data

Guojun Tang and Rongxing Lu*

Faculty of Computer Science,
University of New Brunswick,
Fredericton, NB, Canada
Email: guojun.tang@unb.ca
Email: rlu1@unb.ca
*Corresponding author

Mohammad Mamun

National Research Council Canada (NRC-CNRC),
Government of Canada,
Fredericton, NB, Canada
Email: mohammad.mamun@nrc-cnrc.gc.ca

Abstract: The development of smart applications in e-healthcare has aroused the exponential growth of healthcare data. Therefore, the data owner tends to outsource them to powerful cloud servers, which can provide query services. However, for privacy concerns, the data owner may outsource the encrypted data instead of plaintexts. Moreover, when using the data query service, query users may search the data based on their preferences. To address the aforementioned issues, in this paper, we propose WeightedSim, an efficient and privacy-preserving weighted similarity range query scheme for outsourced healthcare data. Specifically, we first develop an encrypted R-tree index by utilising the symmetric homomorphic encryption (SHE) technique and then employ it to perform a weighted similarity range query under the two cloud servers model. We analyse the security of our scheme to be selectively secure when the SHE is semantically secure against CPA and also conduct extensive experiments to validate the scheme's efficacy.

Keywords: privacy-preserving data query; R-tree; symmetric homomorphic encryption; SHE.

Reference to this paper should be made as follows: Tang, G., Lu, R. and Mamun, M. (2023) 'WeightedSim: privacy-preserving weighted similarity query over encrypted healthcare data', *Int. J. Applied Cryptography*, Vol. 4, Nos. 3/4, pp.143–155.

Biographical notes: Guojun Tang received his BS from th7e South China Normal University, Guangzhou, China, in 2015 and MS from the University of New Brunswick, Fredericton, Canada, in 2022. His current research interests include data privacy, applied cryptography, and federated learning.

Rongxing Lu received his PhD from the Department of Electrical and Computer Engineering, University of Waterloo, Canada, in 2012. He is currently an Associate Professor with the Faculty of Computer Science (FCS), University of New Brunswick (UNB), Canada. Before that, he worked as an Assistant Professor with the School of Electrical and Electronic Engineering, Nanyang Technological University (NTU), Singapore, from April 2013 to August 2016. He was a Post-Doctoral Fellow with the University of Waterloo from May 2012 to April 2013. His research interests include applied cryptography, privacy-enhancing technologies, and the IoT-big data security and privacy.

Mohammad Mamun is a research officer with the cybersecurity team at DTRC. He is also serving as an Adjunct Professor at the UNB and Uwindsor. He also earned his MS in Information and Communication System Security from the Royal Institute of Technology (KTH) in Sweden, and he also obtained his Doctorate from the Japan Advanced Institute of Technology (JAIST) in 2014 with the Outstanding Performance Award. Prior to joining NRC, Mamun held several R&D positions, including as a research associate at the CIC-UNB and analytics manager at The Learning Bar Inc. In 2022, he received a NRC Rising Star Award and was promoted to IEEE senior member. He is currently leading several NRC G&C projects and an author of more than 45 cybersecurity-related articles, including a patent on cybersecurity risk assessment. His research interests are human-centric cybersecurity, entity behavioural fingerprinting, insider threat detection, applied cryptography, applied machine learning, network security, IoT security and privacy.

1 Introduction

The state-of-the-art technologies from IoT (Habibzadeh et al., 2020) and artificial intelligence (Ahmed et al., 2020) have led to an evolution of healthcare applications. Benefitting from those innovations, many healthcare scenarios have dramatic improvements, such as online medical primary diagnosis, electronic healthcare records, health monitoring and precision medicine (Habibzadeh et al., 2020; Ahmed et al., 2020; Hua et al., 2019). As a result, there has been a significant increase in healthcare data and the data owners of healthcare data, usually healthcare centres (HCs), offer the data query service for doctors for further various applications. Similarity query, which retrieves similar data with specific metrics, is one of the most popular data query service options. Euclidean distance is also a very common data measurement for multi-dimensional data. The Euclidean distance-based similarity query has been widely used in a variety of healthcare applications. For example, Kumar (2020) used deep learning and Euclidean distance-based similarity query to solve the duplicated questions within the medical community question-answering sites. In the majority of instances, however, the HC lacks the computing resources and storage capacity to accommodate such a high volume of query requests. To address the computing capabilities shortcoming, the HC typically outsources their healthcare data to third-party cloud servers, which contain robust computing power and may provide a more feasible and reliable data query service. Due to the sensitivity of healthcare data privacy, it raises a new issue about how to prevent private information leakage from cloud servers, which may not be fully trusted. Therefore, the privacy-preserving similarity range query over the healthcare data has been a notable topic.

Encrypting the outsourced data and executing a similarity range query over the ciphertexts is a currently viable solution for privacy issues. In the literature, many privacy-preserving similarity range schemes have been proposed for various data metrics (data models), such as key-value model (Lin et al., 2021), spatial data (Song et al., 2022), string edit distance (Jin et al., 2021), time-series model (Zheng et al., 2021a) and Euclidean distance (Zheng et al., 2021b). Zheng et al. (2021b) designed a privacy-preserving healthcare data query scheme based on Euclidean distance. However, this scheme did not account for the preference on certain dimensions. In our scheme, we consider the similarity range query with the Weighted Euclidean distance (Zheng et al., 2022) (see the definition in Subsection 3.1), which enables the query user to specify weighted values for each data dimension, hence making the similarity range query more specific. We may provide a privacy-preserving data query service for further real-world applications such as Nasser et al. (2019). Additionally, to achieve a more effective and secure query scheme, some schemes leverage the two cloud servers model instead of a single cloud server. For example, Zheng et al. (2021a) proposed a similarity range query scheme over the time series data based on a two semi-trusted cloud servers model. Zhang et al. (2021) employed the symmetric homomorphic encryption (SHE) (Mahdikhani et al., 2020) to implement the privacy-preserving dynamic skyline query scheme for the online medical diagnosis system, which was also based on the two cloud servers model. However, homomorphic encryption and its homomorphic properties are limited to integers only. Therefore, we should still ensure that the data type of vectors is fixed-point decimals, which can be converted into integers for homomorphic encryption by multiplying by the number of multiples of ten.

In this paper, we propose WeightedSim, an efficient and privacy-preserving weighted similarity range query scheme for the healthcare data outsourcing based on SHE (Mahdikhani et al., 2020) and R-tree under a two cloud servers model. In our scheme, the healthcare data are represented as multi-dimensional vectors. Specifically, our main contributions are shown as follows.

- First, we propose an efficient and privacy-preserving weighted similarity range query over healthcare data using encrypted R-tree building blocks. To achieve query efficiency, the R-tree index filters data points that are unreachable by the query vector, thereby reducing the amount of data searched. As for the privacy-preservation, we encrypt the data using SHE technique (Mahdikhani et al., 2020).
- Secondly, based on the SHE homomorphic properties, we present a set of privacy-preserving protocols that may figure out plaintexts' relations without decryption. With those protocols, the cloud servers can launch the data query over the encrypted R-tree without the original plaintext data.
- Finally, we analyse the security of our scheme and conduct a simulation to evaluate its performance. The result indicates that our scheme is privacy-preserving and is selectively secure (Wang et al., 2014) when the SHE is IND-CPA secure. Also, it shows that our scheme is efficient on the multi-dimensional data search.

The remainder of this paper is organised as follows. In Section 2, we formalise our system model and overview of our scheme. Then, we describe the necessary preliminaries in Section 3. After that, we present the details of our proposed scheme in Section 4, followed by the security analysis, experiments in Sections 5 and 6, respectively. In Section 7, we discuss some related works. Finally, we draw our conclusion about our scheme in Section 8.

2 Models and design goals

In this section, we formalise our system model, security model, and identify our design goals.

2.1 System model

In the system model, we propose a weighted similarity range query scheme among the HC, a cloud with two servers $\{S_1, S_2\}$, and multiple query doctors (query users) $U = \{U_1, U_2, ...\}$. The overview of our scheme is shown in Figure 1.

- *HC:* The HC is treated as the data owner of a large amount of healthcare data. HC will offer a weighted similarity range query of database X to the query user. Due to the constrained computing and storage resources, HC will outsource the database to the powerful cloud server. For the privacy concern, the outsourced data should be encrypted, denoted by $E(X)$, in the cloud server.
- *Cloud server:* The cloud server model consists of two cloud servers $\{S_1, S_2\}$. Both of them can provide powerful computing power and abundant storage. S_1 is responsible for the storage of the encrypted database $E(X)$ while S_2 is to assist the data search. S_1 and S_2 will cooperate to offer the weighted similarity range query service to query users. When the query user obtains the authorised key from the HC, it can construct a query request (q, w, τ) and encrypt it into the corresponding query token. Once the cloud server receives the query token from the query user, it will search the encrypted database $E(X)$ for the data satisfying $d_w(x_i, q) = \sqrt{\sum_{j=1}^{k} w_j(x_{i,j} - q_j)^2} \leq \tau$ and returns the ciphertext of those data to the query user.
- *Query user:* Query users $U = \{U_1, U_2, ...\}$, once authorised by the HC, will obtain the authorised key from it. When the query user applies the weighted similarity range query on the encrypted data, it will first generate a query token with the authorised key and send the query token to the cloud server. After receiving the encrypted data from the cloud server, the query user may recover the original healthcare data.

2.2 Security model

In our security model, the HC works as the data owner of healthcare data and is considered as fully trusted. Query users, once authorised by the HC, are also regarded as honest entities who faithfully follow our proposed design to generate query tokens and launch weighted similarity range query requests to the cloud servers. However, the cloud servers $\{S_1, S_2\}$ are considered to be honest-but-curious. It means that they will follow the proposed design and provide a weighted similarity range query on the encrypted database $E(X)$ honestly, but at the same time, cloud servers are curious about the ciphertext and try to obtain the private information of the data. In addition, we assume that there is no collusion between S_1 and S_2. Although no collusion is a very strong security assumption, it is still reasonable in reality by using two different cloud service providers. Cloud servers from different providers can hardly build up mutual trust, so they will encounter much higher risks if they collude with each other. Since our scheme focuses on privacy preservation, we will not discuss attacks out of this scope, e.g., denial of service (DoS) attacks.

2.3 Design goals

In this work, we aim to design an efficient and privacy-preserving weighted similarity range query scheme and have the following two objectives.

- *Privacy preservation:* Plaintexts of the healthcare data and query requests should be secret to the cloud server under the honest-but-curious model.
- *Efficiency:* We also intend to cut down the computational cost of weighted similarity range query among the query user and cloud server and should improve the query efficiency as much as possible.

3 Preliminaries

In this section, we will introduce the basic definition of weighted similarity range query, R-tree and the SHE technique.

3.1 Weighted similarity range query

Let $X = \{x_i = (x_{i,1}, x_{i,2}, ..., x_{i,k}) | i = 1, 2, ..., n\}$ be the healthcare database with which there are n data records totally in X and k dimensions in each data record. Let (q, w, τ) be the query request, where $\mathbf{q} = (\mathbf{q_1}, \mathbf{q_2}, ..., \mathbf{q_k})$ is a k-dimensional query vector, $\mathbf{w} = (\mathbf{w_1}, \mathbf{w_2}, ..., \mathbf{w_k})$ is a k-dimensional weighted vector satisfying $\sum_{j=1}^{k} w_j = 1$, and τ is the distance threshold. With a given query request (q, w, τ) and a data record x_i, we define the weighted Euclidean distance as $d_w(x_i, q) = \sqrt{\sum_{j=1}^{k} w_j(x_{i,j} - q_j)^2}$ and the weighted similarity range query is to figure out all the data records in X which satisfy $d_w(x_i, q) \leq \tau$. Because it is hard to implement the square root computation in homomorphic encryption and $d_w(x_i, q)$ and τ are non-negative values, we will only consider $d_w(x_i, q)^2 \leq \tau^2$ in later sections. Homomorphic properties are only applicable to integers, thus we must transform real numbers to integers by multiplying them by the number of multiples of ten.

3.2 R-tree

R-tree (Guttman, 1984) is a spatial data structure that is friendly to multi-dimensional data and can improve the efficiency of the range query. The main idea behind the R-tree's construction is to group nearby objects at the same level space with a minimum bounding rectangle in the tree's upper level. An example is presented in Figure 3.

Figure 1 Overview of the proposed WeightedSim system

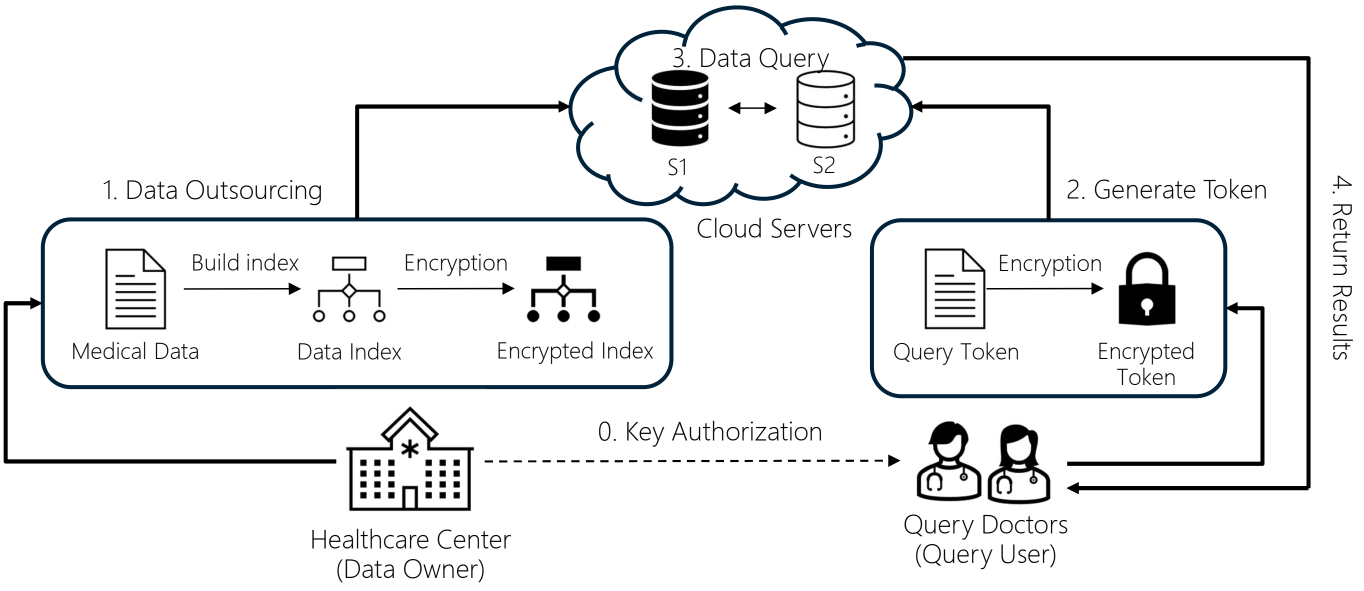


Figure 2 Range query on R-tree (see online version for colours)

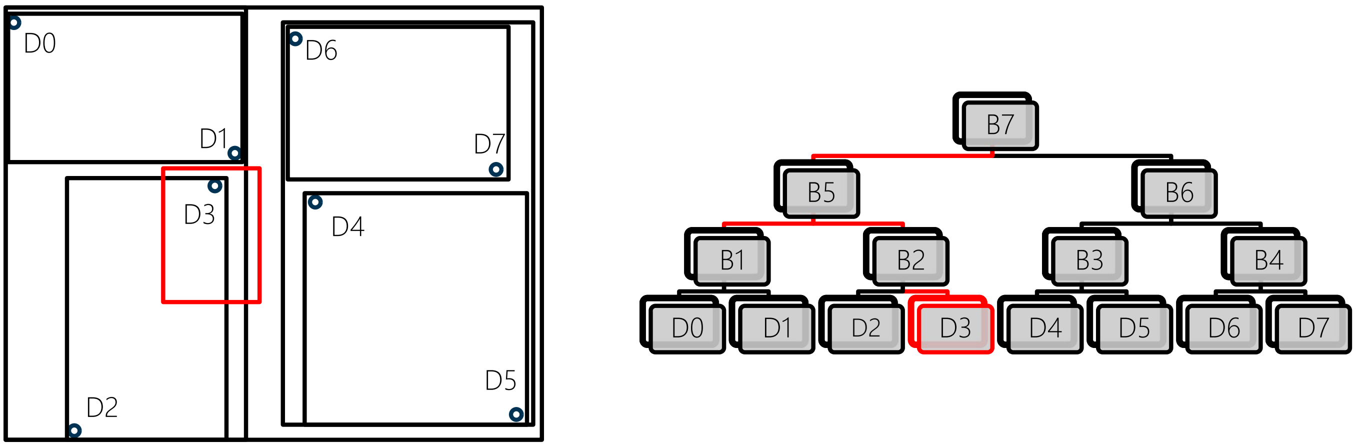
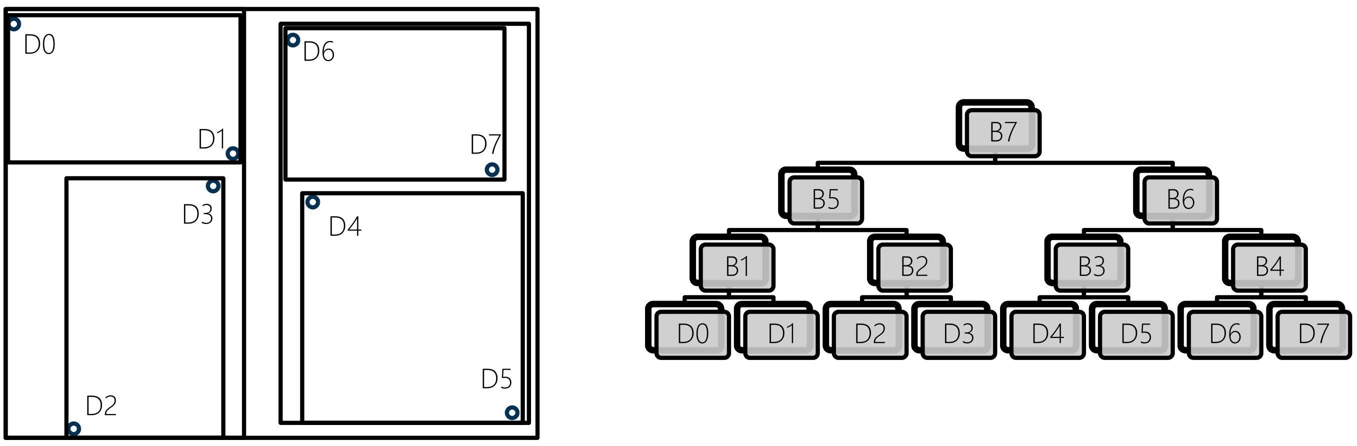


Figure 3 Construction of R-tree



The R-tree is defined as $T \leftarrow \{B_1, B_2, ..., B_m, D_1, D_2, ..., D_n\}$. $D = \{D_1, D_2, ..., D_n\}$ is the data record stored in the leaf-node of R-tree, where n is the number of data and m is the number of internal nodes. $D_i = (d_{i,1}, d_{i,2}, ..., d_{i,k})$ is a k-dimensional vector. $B = \{B_1, B_2, ..., B_m\}$ is the set of the internal node in R-tree with k elements, where $B_j = \{(l_{j,1}, u_{j,1}), (l_{j,2}, u_{j,2}), ..., (l_{j,k}, u_{j,k})\}$ is the k-dimensional rectangle with which $(l_{j,\alpha}, u_{j,\alpha})$ is the lower bound and upper bound of the rectangle in the α^{th} dimension. B_j is the parent or ancestor node of D_j as long as $l_{j,\alpha} \leq$

$d_{i,\alpha} \leq u_{j,\alpha}$ for $1 \leq \alpha \leq k$. In other words, a rectangle and a data point satisfy $l_{0,k} > d_{0,k} || u_{0,k} < d_{0,k}$ indicating that this data point is not involved within this rectangle.

The range query algorithm is shown in Figure 2. To apply a query to the R-tree, the query user first generates a query rectangle to show the range of the query (Wang et al., 2016). And then the query rectangle will check whether it overlaps with the object in the R-tree. The query result will return all the data involved within this query rectangle.

3.3 *The SHE technique*

3.3.1 *The description of SHE*

SHE can provide the efficient homomorphic addition and multiplication for data encryption (Mahdikhani et al., 2020). The SHE technique primarily consists of three algorithms, including SHEKeyGen, SHEEnc and SHEDec. The definition of these algorithms is as follows.

- *SHEKeyGen(k_0, k_1, k_2):* Given security parameters k_0, k_1, k_2 satisfying $k_0 > k_2 \gg k_1$, the key generation algorithm sets up a number $N = pq$ with two large prime numbers p,q where $|p| = |q| = k_0$ and a random number L where $|L| = k_2$. And then the algorithm outputs the public parameter $pb \leftarrow (k_0, k_1, k_2, N)$, the secret key $sk \leftarrow (p, L)$ and the message space $M \leftarrow \{m | m \in [-2^{k_1-1}, 2^{k_1-1})\}$.
- *SHEEnc (sk, m):* Given the secret key sk and a message $m \in M$, the encryption algorithm outputs the ciphertext of m as $E(m) = (rL + m)(1 + r'p)\ mod\ N$, where r and r' are two random numbers satisfying $r \in \{0,1\}^{k_2}$ and $r' \in \{0,1\}^{k_0}$.
- *SHEDec (sk, E(m)):* Given the secret key sk and the ciphertext $E(m)$, the decryption algorithm recovers the message $m' = (E(m)\ mod\ p)\ mod\ L$. If $m' < \frac{L}{2}$, it indicates $m \geq 0$ and $m = m'$. Otherwise, $m < 0$ and $m = m' - L$.

The SHE technique has the homomorphic properties as follows.

- *Homomorphic addition-I:* Two ciphertexts $E(m_1)$ and $E(m_2)$ satisfy $E(m_1) + E(m_2)\ mod\ N \rightarrow E(m_1 + m_2)$.
- *Homomorphic multiplication-I:* Two ciphertexts $E(m_1)$ and $E(m_2)$ satisfy $E(m_1) * E(m_2)\ mod\ N \rightarrow E(m_1 * m_2)$.
- *Homomorphic addition-II:* A ciphertext $E(m_1)$ and a plaintext message m_2 satisfy $E(m_1) + m_2\ mod\ N \rightarrow E(m_1 + m_2)$.
- *Homomorphic multiplication-II:* A ciphertext $E(m_1)$ and a plaintext message $m_2 > 0$ satisfy $E(m_1) * m_2\ mod\ N \rightarrow E(m_1 * m_2)$.

Note that, as SHE is a leveled homomorphic encryption, which can only provide a limited round of calculation in homomorphic multiplication-I (Zheng et al., 2021a). To avoid incorrect decryption results, we make sure the depth of homomorphic multiplication-I should be within $\theta = \lfloor \frac{k_0}{2k_2} - 1 \rfloor$.

3.3.2 *SHE public key setting*

According to the homomorphic properties, we can encrypt the message with the public key $pk \leftarrow \{E(0)_1, E(0)_2\}$, where $E(0)_1$ and $E(0)_2$ are two ciphertexts of 0 with different random numbers. When we apply the public key encryption on the message, we have $E(m) = (m + r_1E(0)_1 + r_2E(0)_2)\ mod\ N$, where r_1 and r_2 are two random numbers satisfying $r_1, r_2 \in \{0,1\}^{k_2}$. In later sections, we denote the public key encryption as $SHEEnc(pk, m)$.

3.4 *Privacy-preserving protocol*

Based on the SHE-based privacy-preserving protocol in Zhang et al. (2021), to implement the secure two-party computation, we design three atomic privacy-preserving protocols: secure less than or equal (SLESSE), directly less than (DLESS) and directly within (DWITHIN). Generally, S_1 keeps the homomorphic encryption public parameter and the encrypted data while S_2 holds the homomorphic encryption secret key. Via these protocols, S_1 and S_2 may figure out the comparison among those data without leaking the plaintext information to both parties.

1 *SLESSE:* Cloud server S_1, S_2 work together to calculate the result of whether $m_1 \leq m_2$ with $E(m_1)$ and $E(m_2)$ while the original information of m_1 and m_2 will be in secret. In this protocol, S_1 holds the public parameter *pb* and ciphertexts $\{E(m_1)$, $E(m_2)$, $E(-1)\}$ while S_2 holds the public parameter *pb* and the secret key *sk*. In the later section, we denote it as $SLESSE(m_1, m_2)$. The detail of this protocol is as follows.

   - *Step 1:* S_1 calculates the ciphertext $E(x') = E(m_1 - m_2) = E(m_1) + E(-1) * E(m_2)$ and generates a random number $s \in \{-1, 1\}$ by coin flipping. According to properties of homomorphic addition-II and homomorphic multiplication-II, then S_1 figures out $E(x) = E(s * r_1 * x' - s * r_2)$ where $r_1, r_2 \in \{0,1\}^{k_1}$ are two random numbers satisfying $r_1 > r_2 > 0$. Notably, the plaintext message in homomorphic multiplication-II should not be negative. When $s = -1$, we need to multiply $E(x')$ by $E(-1)$.

$$E(x) = \begin{cases} s * r_1 * E(x') - s * r_2 & s = 1 \\ E(-1) * r_1 * E(x') - s * r_2 & s = -1 \end{cases}$$

   S_1 sends the $E(x)$ to S_2.

- *Step 2:* After receiving the $E(x)$, S_2 utilises the secret key sk to recover x. S_2 returns a encrypted value $E(b')$ to S_1. If $x \leq 0$, $E(b') = E(1)$ otherwise $E(b') = E(0)$.
- *Step 3:* S_1 receives $E(b')$ from S_2 and outputs the result $E(b)$. If $s = 1$, $E(b) = E(b')$. If $s = -1$ and $E(b) = E(1) + E(s) * E(b') = E(1 - b')$. $E(b)$ outputs $E(b) = E(1)$ if $m_1 \leq m_2$, otherwise $E(b) = E(0)$.

$$E(b) = \begin{cases} E(b') & s = 1 \\ E(1) + E(s) * E(b') & s = -1 \\ = E(1 - b') & \end{cases}$$

- *Correctness:* The protocol is correct iff when $m_1 \leq m_2$ it outputs $E(b) = E(1)$ otherwise $E(b) = E(0)$. When $s = 1$, if $m_1 \leq m_2$, because of $r_1 > r_2 > 0$ and $m_1 - m_2 \leq 0$, we may figure out that $x = s * r_1 * (m_1 - m_2) - s * r_2 \leq 0$ and the S_2 will return $E(b') = E(1)$ to S_1. Due to $s = 1$, the protocol will return $E(b) = E(b') = E(1)$. When $s = -1$, If $m_1 \leq m_2$, we have $x \geq 0$ and the S_2 will return $E(b') = 0$ to S_1. And the S_1 will compute the result $E(b) = E(1 - b') = E(1)$. We can use the similar method to prove that the protocol will return $E(b) = E(0)$ if $m_1 > m_2$. Therefore, the Secure Less Equal Than protocol is correct.

2 *DLESS:* Cloud server S_1, S_2 work together to calculate the result of whether $m_1 < m_2$ with $E(m_1)$ and $E(m_2)$. However, S_1 gets the result directly while it still keeps secret to S_2. The other private information such as plaintexts should still be in secret. In this protocol, S_1 holds the public parameter pb and ciphertexts $\{E(m_1), E(m_2), E(-1)\}$ while S_2 holds the public parameter pb and the secret key sk. In the later section, we denote it as $DLESS(m_1, m_2)$. The detail of this protocol is as follows.

- *Step 1:* S_1 calculates the ciphertext $E(x') = E(m_1 - m_2) = E(m_1) + E(-1) * E(m_2)$ and, according to properties of homomorphic addition-II and homomorphic multiplication-II, then figures out $E(x) = E(s * r_1 * x' + s * r_2) = s * r_1 * E(x') + s * r_2$ where $r_1, r_2 \in \{0,1\}^{k_1}$ are two random numbers satisfying $r_1 > r_2 > 0$ and $s \in \{-1, 1\}$ is a random number generated by coin flipping by S_1. S_1 sends $E(x)$ to S_2.
- *Step 2:* After receiving the $E(x)$, S_2 utilises the secret key sk to recover x. S_2 returns the result b' directly to S_1. If $x < 0$, $b' = true$ otherwise $b' = false$.
- *Step 3:* S_1 receives b' from S_2 and outputs the result b. If $s = 1$, $b = b'$. If $s = -1$ and $b = \neg b'$. The protocol outputs $b = true$ if $m_1 < m_2$, otherwise $b = false$.

$$E(b) = \begin{cases} b' & s = 1 \\ \neg b' & s = -1 \end{cases}$$

- *Correctness:* We can prove the correctness of this protocol in a similar method with SLESSE.

3 *DWITHIN:* Given three ciphertexts $E(m_1)$, $E(m_2)$, $E(m_3)$ and assumed that $m_1 < m_3$, this protocol is to check whether $m_1 \leq m_2 \leq m_3$. In this protocol, S_1 holds the public parameter pb and ciphertexts $\{E(m_1), E(m_2), E(m_3), E(-1)\}$ while S_2 holds the public parameter pb and the secret key sk. Similar to the DLESS, S_1 will figure out the result directly while the result is still in secret in S_2. In the later section, we denote it as $DWITHIN(m_1, m_2, m_3)$. The detail instruction is as follow.

- *Step 1:* S_1 calculates the ciphertext $E(x') = E((m_1 - m_2) * (m_3 - m_2)) = (E(m_1) + E(-1) * E(m_2)) * (E(m_3) + E(-1) * E(m_2))$. Similar to $SLESSE$, S_1 flips a coin $s \in \{-1, 1\}$ and chooses two random numbers $r_1, r_2 \in \{0,1\}^{k_1}$ which satisfy $r_1 > r_2 > 0$ to compute $E(x) = E(s * r_1 * x' + s * r_2) = s * r_1 * E(x') - s * r_2$ and then sends $E(x)$ to S_2.
- *Step 2:* When S_2 receives the $E(x)$, it recovers x with the secret key sk. If $x \leq 0$, S_2 returns a plaintext flag $b' = true$ to S_1 otherwise returns $b' = false$.
- *Step 3:* If $s = 1$, S_1 computes the result bit b as $b = b'$. If $s = -1$, S_1 returns the flipped flag $b = \neg b'$ as result. If m_1,m_2 and m_3 satisfy $m_1 \leq m_2 \leq m_3$, the protocol output $b = true$, otherwise $b = false$.

$$b = \begin{cases} b' & s = 1 \\ \neg b' & s = -1 \end{cases}$$

- *Correctness:* If $m_1 \leq m_2 \leq m_3$ and $m_1 < m_3$, we may deduce the condition to $(m_1 - m_2) * (m_3 - m_2) <= 0$. And then we can use the similar method to prove the DWITHIN.

4 Our proposed scheme

4.1 Weighted similarity range query over plaintexts

We first build up the R-tree index over the database. To reduce the calculation of the weighted Euclidean distance, we apply the range query on R-tree to filter the data points which are impossible to include within the weighted Euclidean distance with the query point. Given a query request (q, w, τ), for a data point x_i, if it exists a dimension j that satisfies $\sqrt{w_j(x_{i,j} - q_j)^2} > \tau$, we can consider this data point is impossible to be involved within the weighted similarity range query and preclude it from the result set. Therefore, we setup a query rectangle $B_0 = \{(q_j - \frac{\tau}{\sqrt{w_j}}, q_j + \frac{\tau}{\sqrt{w_j}})|j = 1, ..., k\}$. Because homomorphic properties are only applicable to integers, we must multiply real numbers in the data and the query rectangle by the number of multiples of ten to convert them to integers. Once we apply this query rectangle to the R-tree, we can acquire those data points which may probably satisfy the weighted similarity range query and add them to a candidate set. Finally, we check whether the weighted euclidean distance between each candidate set element and the query point is less than or equal to the τ.

Algorithm 1 secureRangeQuery($E(B_0)$, $E(T)$)

```
Input: Encrypted query rectangle $E(B_0)$, Encrypted R-tree node $E(T)$
Output: Candidate set $C$
  for each $E(t) \in T.Children$ do
    if $E(t).isLeadNode() == false$ then
      if $secureIntersect(E(B_0), E(t))$ then
        secureRangeQuery($E(B_0)$, $E(t)$)
      end if
    else
      if $secureInside(E(B_0), E(t))$ then
        $E(x_i) \leftarrow E(t).getData()$
        $R \leftarrow R \cup \{E(x_i)\}$
      end if
    end if
  end for
```

Algorithm 2 secureInside($E(B_0)$, $E(D_0)$)

```
Input: Encrypted rectangle $E(B_0)$, encrypted data point $E(D_0)$
Output: Return true if $D_0 \in B_0$; Otherwise return false.
  for $i \leftarrow 1$ to $k$ do
    if $\neg(DWITHIN(l_{0,k}, d_{0,k}, u_{0,k}))$ then
      return false
    end if
  end for
  return true
```

Algorithm 3 secureIntersect($E(B_0)$, $E(B_1)$)

```
Input: Encrypted rectangle $E(B_0)$, $E(B_1)$
Output: Return true if $B_0$ intersects with $B_1$; Otherwise return false.
  for $i \leftarrow 1$ to $k$ do
    if $(DLESS(u_{1,k}, l_{0,k}) || DLESS(u_{0,k}, l_{1,k}))$ then
      return false
    end if
  end for
  return true
```

4.2 Description of our scheme

In this section, we will introduce the detailed description of our privacy-preserving weighted similarity range query on healthcare data, including the system initialisation, healthcare data outsourcing and privacy-preserving weighted similarity range query.

4.2.1 System initialisation

The system initialisation starts at the HC. In this step, the HC generates and distributes security keys to the corresponding entity. The HC first selects the proper security parameters k_0, k_1, k_2 before invoking the $SHEKeyGen(k_0, k_1, k_2)$ to generate the public parameter and secret key $\{pb, sk\}$ for the encryption. After that, HC generates the public key $pk \leftarrow \{E(0)_1, E(0)_2\}$ at Subsection 3.3.2. HC delivers pb to S_1 and sk to S_2. In our model, S_1 stores and handles encrypted data. Without the sk, S_1 is unable to access plaintexts. While S_2 possesses the encryption secret key, without the encrypted data, it also cannot access the plaintext of the original data. If both parties adhere to our security assumptions, they cannot access the plaintext data. And then HC generates ciphertext $\{E(-1)\}$, which are leveraged for the filtration stage and refinement stage in the weighted range query, and sends the ciphertext to S_1. HC delegates the query users $U = \{U_1, U_2, ...\}$ in the system and sends pk to them. When the new query user registers on the system and is authorised by HC, it will also receive the key from HC. Using pk, the query user can encrypt the query data before sending it to the cloud server. Because query users may have a lower security level, we only delegate the public key instead of the secret key to them.

4.2.2 Healthcare data outsourcing

Given the healthcare database $X = \{x_i = (x_{i,1}, x_{i,2}, ..., x_{i,k})|i = 1, 2, ..., n\}$, the HC will construct the encrypted R-tree over X and then outsource it to S_1. The detail of data outsourcing is as follows.

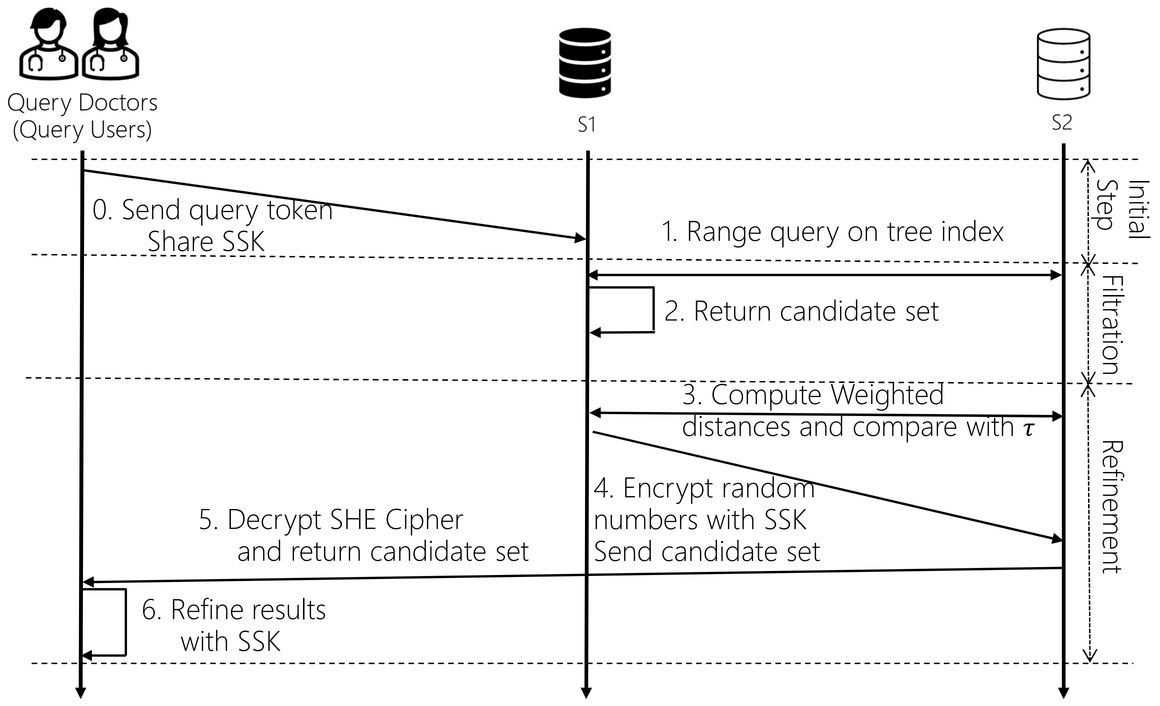
Step 1 HC builds up a k-dimensional R-tree $T \leftarrow \{B_1, B_2, ..., B_m, D_1, D_2, ..., D_n\}$ on the dataset X. As stated in Subsection 3.1, if the numbers in the database are not integers, they must first be pre-processed and converted into integers.

Step 2 HC encrypts the the tree into $E(T) \leftarrow \{E(B_1), ..., E(B_m), E(D_1), ..., E(D_n)\}$, where $B_j = \{(E(l_{j,1}), E(u_{j,1})), ..., (E(l_{j,k}), E(u_{j,k}))\}$ and $E(D_i) = \{E(x_i)\}$ with $SHEEnc(sk, m)$. After that, the HC outsources the $E(T)$ on S_1.

4.2.3 Weighted similarity range query over ciphertexts

After receiving the encrypted tree $E(T)$, S_1 will cooperate with S_2 to offer the weighted similarity range query for query users. The query user may launch the similarity query by submitting the query token. Specifically, it consists of three steps: an initial step for query initialisation, a filtration step for excluding data points inaccessible to the query point, and a refinement step for calculating the weighted Euclidean distance between the data and query point and returning the result to the user. The overview of the range query is shown in Figure 4.

Figure 4 Overview of range query in WeightedSim


Query Doctors (Query Users)
S1
S2
0. Send query token Share SSK
1. Range query on tree index
2. Return candidate set
3. Compute Weighted distances and compare with τ
4. Encrypt random numbers with SSK Send candidate set
5. Decrypt SHE Cipher and return candidate set
6. Refine results with SSK
Initial Step
Filtration
Refinement

- *Initial step:* The query user constructs a query request (q, w, τ) and generates the refinement token ($E(q)$, $E(w)$, $E(\tau)$) with $SHEEnc(pk, m)$. And then the query user constructs the query rectangle $B_0 = \{(q_j - \frac{\tau}{\sqrt{w_j}}, q_j + \frac{\tau}{\sqrt{w_j}})|j = 1, ..., k\}$ as the instructions in Subsection 4.1 and encrypted it into the filtration token. As stated in Subsection 3.1, we should convert the real numbers in the query rectangle into integers before generating the filtration token. The query user then chooses a session key ssk and a symmetric encryption to recover the query result. In our scheme, we select AES as the encryption algorithm, denoted as $AES_{ssk}(.)$. Finally, the user sends the session key ssk and query token $\{E(q), E(w), E(\tau), E(B_0)\}$ to S_1.
- *Filtration step:* We will use R-tree range query to preclude the inaccessible data and filter a candidate set. Because the internal nodes and leaf nodes data are encrypted by SHE, we implement the range query using the privacy protocols in Subsection 3.4. We first design the top level function as Algorithm 1. To check whether the encrypted data point is inside the encrypted rectangle, because the statement $l_{0,k} > d_{0,k}||u_{0,k} < d_{0,k}$ is equivalent to $\neg(l_{0,k} \leq d_{0,k} \leq u_{0,k})$, we may design the $secureInside(E(B_0), E(D_0))$ as Algorithm 2 by implementing the conditional statement with DWITHIN protocol in Subsection 3.4. Correspondingly, we can implement $secureIntersect(E(B_1), E(B_0))$ as Algorithm 3 to check whether two encrypted rectangles intersect with DLESS. Eventually, The algorithm outputs a candidate set $C = \{E(x_i)|x_i$ may probably satisfy $d_w(x_i, q) \leq \tau\}$.
- *Refinement step:* For $x_i \in C$, S_1 computes the weighted Euclidean distance between x_i and q according to the SHE homomorphic properties. Because it is difficult to apply square root in the homomorphic encryption, S_1 calculates $E(d_w(x_i, q)^2) = E(\sum_{j=1}^{k} w_j(x_{i,j} - q_j)^2)$ in the practical. After that, S_1 and S_2 cooperate to launch the SLESSE in Subsection 3.4 to compare $E(d_w(x_i, q)^2)$ and $E(\tau^2)$. If $d_w(x_i, q)^2 \leq \tau^2$, in other words, x_i satisfies the similarity query, the protocol returns a encrypted result bit $E(b_i) = E(1)$, otherwise $E(b_i) = E(0)$. Next, S_1 generates a k-dimensional random vectors $\Gamma_i = \{\gamma_{i,1}, ..., \gamma_{i,k}\}$ and random value $\gamma_{i,0}$ for each candidate x_i and calculates $E(x_i + \Gamma_i) = E(x_i) + \Gamma_i$ and $E(b_i + \gamma_{i,0}) = E(b_i) + \gamma_{i,0}$ to obfuscate plaintexts. Those random numbers belong to the SHE message space $\{m|m \in [-2^{k_1-1}, 2^{k_1-1})\}$. S_1 encrypts each random vector and random value with symmetric encryption, represented as $AES_{ssk}(\Gamma_i)||AES_{ssk}(\gamma_{i,0})$. Finally, S_1 construct a result set $R = \{(E(x_i + \Gamma_i), E(b_i + \gamma_{i,0}), AES_{ssk}(\Gamma_i)||AES_{ssk}(\gamma_{i,0}))|x_i \in C\}$ and sends to S_2.

Once S_2 receives the result set, it recovers the SHE plaintexts by the private key sk and transfers the new result set $\{x_i + \Gamma_i, b_i + \gamma_{i,0}, AES_{ssk}(\Gamma_i)||AES_{ssk}(\gamma_{i,0})\}$ to the query user.

Because the query user also owns the session key ssk, it can remove random values easily. Firstly, it decrypts the

$AES_{ssk}(\gamma_{i,0})$ to obtain b_i by removing the random value $\gamma_{i,0}$. If $b_i = 1$, it means that x_i is one of the results of the range query, and then the query user continues to decrypt $AES_{ssk}(\Gamma_i)$ and deduces the random vector Γ_i to obtain the original data x_i.

5 Security analysis

In this section, firstly we will present the security analysis of the SHE-based privacy-preserving protocol. And then we will analyse the security of our scheme.

5.1 Security of privacy-preserving protocol

With the given ciphertexts $\{E(m_1), E(m_2)\}$, the SLESSE protocol outputs the result of whether $m_1 < m_2$ while S_1 and S_2 can not obtain the information of plaintexts. In this protocol, S_1 holds the encrypted inputs [ciphertexts $\{E(m_1), E(m_2)\}$] and the encrypted output [$E(b)$]. Those messages are encrypted by SHE, which has been proven to be semantically secure against CPA (Zheng et al., 2021a). Therefore, without the secret key sk, S_1 can not obtain any information of those messages from the ciphertexts. On the other side, S_2 can recover the plaintext $x = s * r_1 * (m_1 - m_2) - s * r_2$ with the sk. However, the random numbers r_1 and r_2 prevent the S_2 from obtaining the original information of $(m_1 - m_2)$. And S_2 can not figure out whehter $(m_1 - m_2)$ is greater than, less than or equal to 0 without the coin flipping number s.

We can use a similar way to analyse DLESS and DWITHIN protocols. However, S_1 directly knows the result of the protocol while S_2 can not figure it out due to the coin-flipping.

5.2 Security of weighted similarity range query

We will prove that our scheme is selectively secure by the real/ideal world model. Firstly, we will define the leakage function from the perspectives of S_1 and S_2. After that, we will present the definitions of the real-world model and ideal world model and prove that the probability of the adversaries in our security definition distinguishing the view of the real/ideal world model is negligible.

1 *Leakage function:* The leakage function $L(\cdot)$ denotes the leak information to an entity. Here, we define the leak information to S_1 and S_2.

- $L(S_1)$: The information leakage to S_1 involves the public parameter pb, the encrypted R-tree $E(T)$, the encrypted query token ($E(q)$, $E(w)$, $E(\tau)$, $E(B_0)$), ciphertext $\{E(-1)\}$. Also, because this is a tree-based search scheme, it is inevitable to leak the access pattern and search pattern (Wang et al., 2014). Due to the R-tree structure and DWITHIN and DLESS protocols, the relations among some plaintexts will also be revealed.
- $L(S_2)$: The information leakage to S_2 involves the public parameter pb, the secret key sk, and the number of data records in the candidate set.

2 *Real world model:* In the real world model, there are two adversaries A_1 and A_2 with probabilistic polynomial time (PPT), and we assume that they will not collude. Their interactions are as follows.

- *Initialisation phase:* A_1 sends a dataset $X = \{x_i = (x_{i,1}, x_{i,2}, ..., x_{i,k}) | i = 1, 2, ..., n\}$ to the challenger. And then, the challenger follows the instructions of Subsection 4.2.1 to generate the SHE key $\{pb, sk\}$. The challenger sends pb to A_1 and sk to A_2. As Subsection 4.2.2, the challenger builds up the R-tree T according to X and encrypts it into $E(T)$ with SHE key.
- *Token phase 1:* A_1 submits c_0 weighted similarity range queries, denoted as $\{q_i, w_i, \tau_i | 1 \le i \le c_0\}$, where c_0 is a polynomial number, and sends to the challenger. The challenger applies the token generation algorithm in Subsection 4.2.3 and generates the query tokens $\{E(q_i)$, $E(w_i)$, $E(\tau_i)$, $E(B_i) | 1 \le i \le c_0\}$. After that, the challenger returns these query tokens to A_1.
- *Challenge phase:* The challenger returns $E(T)$ to A_1.
- *Token phase 2:* A_1 submits $c_1 - c_0$ weighted similarity range queries, denoted as $\{q_i, w_i, \tau_i | c_0 + 1 \le i \le c_1\}$, where c_1 is also a polynomial number, and gets the query tokens $\{E(q_i)$, $E(w_i)$, $E(\tau_i)$, $E(B_i) | c_0 + 1 \le i \le c_1\}$ from challenger.
- *Query processing:* With these query tokens $\{E(q_i)$, $E(w_i)$, $E(\tau_i)$, $E(B_i) | 1 \le i \le c_1\}$, A_1 and A_2 cooperate to apply the weighted similarity range query on $E(T)$ to obtain the query result.

We denote the view of A_1 in the real world as $realView(A_1)$ which involves the query tokens, SHE ciphertexts from $E(T)$ and SLESSE protocol, the access pattern and search pattern of $E(T)$, the plaintexts relation from DWITHIN and DLESS protocols, and the R-tree. Similarly, we denote the view of A_2 in the real world as $realView(A_2)$ which includes the AES ciphertexts in the result set, plaintexts in the result set and some plaintexts from protocols.

3 *Ideal world model:* In the ideal world model, there are two PPT adversaries A_1 and A_2 (assume that these two adversaries will not collude with each other) and a simulator based on $L(S_1)$ and $L(S_2)$. They interact as follows.

- *Initialisation phase:* A_1 sends a dataset $X = \{x_i = (x_{i,1}, x_{i,2}, ..., x_{i,k})|i = 1, 2, ..., n\}$ to the simulator. And then, the simulator delivers $pb \in L(S_1)$ to A_1 and $sk \in L(S_2)$ to A_2. According to the $E(T) \in L(S_1)$, the simulator builds up a encrypted R-tree $E'(T)$ which is isomorphic to $E(T)$. The main idea of constructing $E'(T)$ is to apply SHE self-binding (Zheng et al., 2021a). Firstly, the simulator leverages $sk \in L(S_2)$ to compute two different ciphertexts of 0 $E_1(0)$ and $E_2(0)$. After that, for each ciphertext $E(x)$ in each node, the simulator calculates $E'(x) = E(x) + r_1 * E_1(0) + r_2 * E_2(0)$, where r_1 and r_2 are random numbers.
- *Token phase 1:* A_1 submits c_0 weighted similarity range queries, denoted as $\{q_i, w_i, \tau_i|1 \le i \le c_0\}$ and sends to the simulator. The simulator generates the query tokens $\{E(q_i), E(w_i), E(\tau_i), E(B_i)|1 \le i \le c_0\}$ with the SHE key in $L(S_1)$ and converts the query token into $\{E'(q_i), E'(w_i), E'(\tau_i), E'(B_i)|1 \le i \le c_0\}$ by self-binding. After that, the simulator returns these query tokens to A_1.
- *Challenge phase:* The simulator returns $E'(T)$ to A_1.
- *Token phase 2:* A_1 submits $c_1 - c_0$ weighted similarity range queries and gets the query tokens $\{E'(q_i), E'(w_i), E'(\tau_i), E'(B_i)|c_0 + 1 \le i \le c_1\}$ from simulator.
- *Query processing:* With these query tokens $\{E'(q_i), E'(w_i), E'(\tau_i), E'(B_i)|1 \le i \le c_1\}$, A_1 and A_2 cooperate to apply the weighted similarity range query on $E'(T)$ to obtain the query result.

We denote the view of A_1 in the ideal world as $idealView(A_1)$ which involves the query tokens, SHE ciphertexts from $E'(T)$ and SLESSE protocol, the access pattern and search pattern of $E'(T)$, the plaintexts relation from DWITHIN and DLESS protocols, and the R-tree. In the similar way, we denote the view of A_2 in the ideal world as $idealView(A_2)$ which includes the AES ciphertexts in the candidate set, some plaintexts received from protocols and candidate set.

Definition 1 (selevtive security of our scheme): Our scheme is selectively secure with $L(S_1)$ and $L(S_2)$ iff, for any two PPT adversaries A_1 and A_2, there exists a simulator that the probability of A_1 and A_2 to distinguish the real world model and ideal world model is negligible.

Theorem 1: Our scheme is selectively secure with the leakage $L(S_1)$ and $L(S_2)$ iff the SHE technique is semantically secure against CPA.

Proof: According to Definition 1, to prove the selective security of our scheme, we will show that A_1 and A_2 can not distinguish between the view of the real world and the ideal world respectively, as follows.

- *View of A_1:* Both $idealView(A_1)$ and $realView(A_1)$ include the query tokens, SHE-encrypted R-tree, the access pattern and search pattern of R-tree. As for the $E(T)$ and $E'(T)$, because $E(T)$ is isomorphic to $E'(T)$, A_1 can not distinguish these two views from the tree structures. The internal nodes and leaf nodes in $E'(T)$ are generated by the self-binding from $E(T)$, also these nodes are encrypted by SHE which has been proven to IND-CPA (Zheng et al., 2021a), so A_1 can not distinguish the nodes in $E(T)$ and $E'(T)$. Correspondingly, A_1 can not distinguish the ciphertexts of SLESSE protocol and query token from $idealView(A_1)$ and $realView(A_1)$ because they also are encrypted by SHE. Because of the self-binding in $E'(T)$, the ciphertexts still hold up the same plaintext relation. As a consequence, the same query token will have the same DLESS and DWITHIN results as well as the search path on the tree structure. Therefore, A_1 can not distinguish the access pattern and search pattern of $E(T)$ and $E'(T)$. Therefore, A_1 can not distinguishing $idealView(A_1)$ and $realView(A_1)$.
- *View of A_2:* Both $idealView(A_2)$ and $realView(A_2)$ include the AES ciphertexts in the candidate set, some plaintexts received from privacy-preserving protocols and candidate set. Because the AES ciphertexts are encrypted from random numbers, A_2 can not distinguish the ciphertexts from $idealView(A_2)$ and $realView(A_2)$. As for the plaintexts from protocols and candidate set, they are all garbled by random numbers, so A_2 can not distinguish the plaintexts from the real world and ideal world.

In conclusion, A_1 and A_2 can not distinguish between the view of the real world and the ideal world.

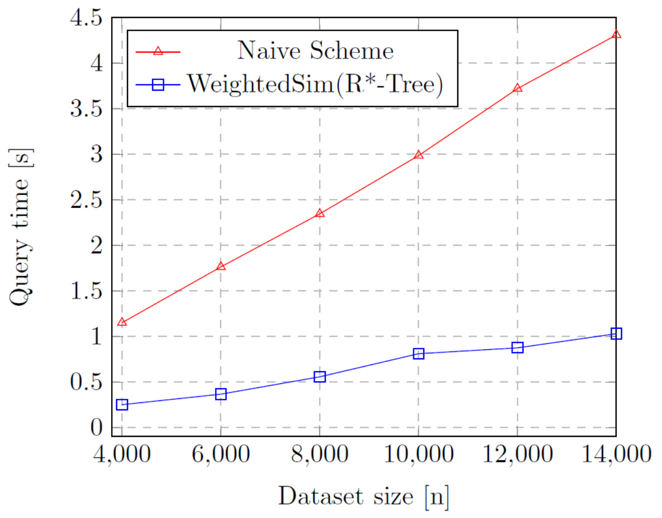
6 Experiments

In this section, we conduct a simulation of WeightedSim to evaluate the performance of the weighted similarity range query. Specifically, to verify the efficiency of the filtration step of our scheme, we compare WeightedSim with the naive scheme that does not apply filtration in the query. We implement the WeightedSim and naive scheme with Java (JDK 15 as execution environment) and execute on a machine with 16 GB memory, 2.90 GHz AMD Ryzen 7 4800H CPU and Win 11 OS. In the simulation, we evaluate the performance on the EEG (2013) dataset, which contains 14,980 rows of data with 15 attributes. As for the R-tree, we leverage R*-tree (Beckmann et al., 1990) as the data index construction strategy and implement the scheme based on an open-source R-tree repository (https://github.com/davidmoten/rtree-multi).

Also, the security parameters of SHE are set as k_0 = 1,200, k_1 = 30, k_2 = 80. We can figure out the maximum multiplicative depth θ = 6 and all the SHE privacy-preserving protocols are involved within the maximum depth. The summary of the experimental environment is shown in Table 1. The computational cost is correlated with three factors: the dataset size n, the data dimension k and the query threshold τ. In the following part, we conduct the experiments around these three factors.

- *Computational cost versus n:* In our scheme, the similarity query time increases correspondingly by the size of the dataset. In this experiment, we setup the parameters as n ranging from 4,000 to 14,000, τ = 6 and k = 5. Figure 5 indicates the computational cost of similarity query between our scheme and the naive scheme varying with the dataset size n. From the diagram, we can learn that after applying the filtration strategy, our scheme can considerably reduce the computation cost compared with the naive scheme without filtration. Besides, the increasing rate of our scheme is smaller than the naive scheme.

Figure 5 Computational cost varying from dataset size (see online version for colours)



- *Computational cost versus k:* In our scheme, the performance of the similarity query is also affected by the data dimension. In this experiment, we setup the parameters as n = 4,000, τ = 6 and k ranging from 4 to 14 and the experiment result is as shown in Figure 6. From the diagram, we can know that the query time will also increase with the increment of the data dimension in both schemes and our scheme is much more efficient than the naive scheme. Also, the increase of data dimension only has a slight impact on the performance in the WeightedSim compared with the naive scheme.
- *Computational cost versus τ:* In our scheme, the query threshold also impacts the query efficiency. With the increase of τ, the size of the candidate set in the filtration step will become larger so that the query time will also increase correspondingly. In the experiment, we setup the parameters as n = 4,000, τ ranging from 2 to 8 and k = 5 and plot the computational cost varying with τ in Figure 7. The chart shows that the query time of our scheme increases by the increase of τ while the computational cost of the naive scheme stays stable.

Figure 6 Computational cost varying from data dimension (see online version for colours)

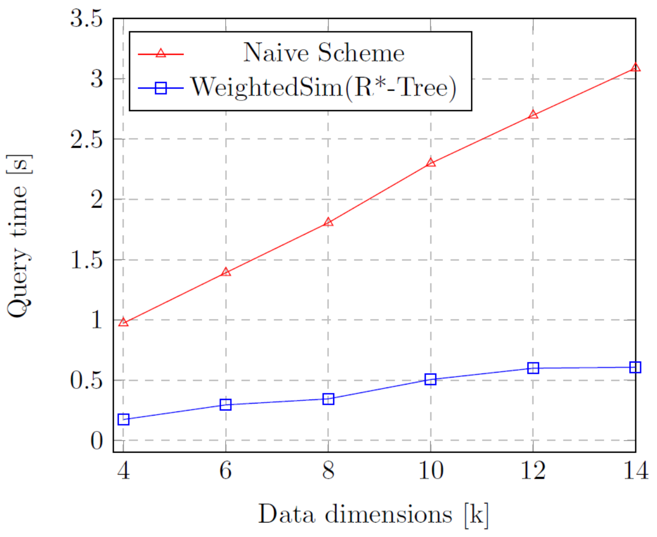
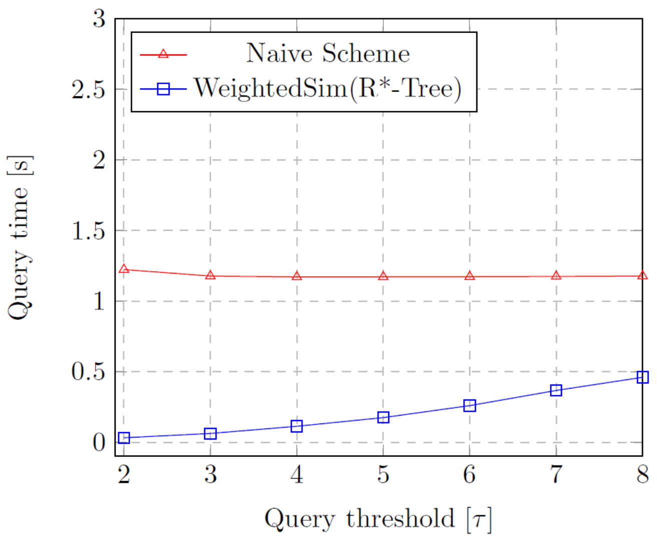


Figure 7 Computational cost varying from query threshold (see online version for colours)



7 Related works

As we stated, there have appeared some research works on privacy-preserving data query (Hua et al., 2019; Zhang et al., 2021; Lin et al., 2021; Jin et al., 2021; Song et al., 2022; Zheng et al., 2021a, 2021b, 2022) which may be applicable in the healthcare scenario to protect the data privacy.

Wang et al. (2016) employed R-tree and OPE techniques to design a privacy-preserving nearest neighbour search that achieved IND-OCPA (Boldyreva et al., 2009). However, the encryption revealed the order of ciphertexts and reached a relatively weak secure level. Zhang et al. (2021) based on SHE technique building blocks, proposed a privacy-preserving dynamic skyline query scheme for the secure online medical diagnosis system. In their scheme, they also designed a series of SHE privacy-preserving protocols under the two cloud servers model. Similarly, based on the SHE technique and two cloud servers model, Zheng et al. (2021a) presented a privacy-preserving similarity range query scheme over the encrypted time series data with the k-d tree. To avoid exceeding the multiplicative depth of homomorphic multiplication in SHE, they designed a bootstrap protocol to refresh the ciphertexts. However, the bootstrap protocol will cause additional computation and reduce the efficiency of the data query.

Table 1 Experimental environment

Parameter	*Setting*
Evaluating CPU	2.90 GHz AMD Ryzen 7 4800H
Evaluating RAM	16 GB
Programming language	Java
Execution environment	JDK 15
Operating system	Win 11
SHE security parameters	k_0 = 1,200, k_1 = 30, k_2 = 80
Dataset	EEG (2013) eye state dataset

Secure two-party computation was originated by Yao (1982). Although the original design of Yao was inefficient, this area has been discussed profoundly and there has been significant progress in secure two-party computation. Toft (2011) presented secure comparison schemes for two non-colluding parties. Shelat and Shen (2013) implemented Yao's (1982) protocol with weaker security assumptions while improving the performance. This technique has also been applied in real-world applications. In our scheme, we proposed a set of secure two-party computation protocols to compare the data based on homomorphic encryption under the honest-but-curious and no collusion security assumptions.

Zheng et al. (2021b) designed a Euclidean distance-based privacy-preserving similarity range query for healthcare data. In their scheme, these healthcare data were encrypted with modified asymmetric-scalar-product encryption (MASPE) and a quadsector tree was created as the index. They applied the filtration strategy to prune the unnecessary tree searching paths and the refinement strategy to validate whether the data satisfied the similarity range query. However, this scheme did not consider the preference of some specific data dimensions when querying. Later, Zheng et al. (2022) firstly introduced the weighted similarity range query on the healthcare data. In their scheme, they defined the negative infinity norm (NIND) distance as the lower bound of the weighted Euclidean distance. When building an index, they selected reference points from the data and applied the triangle inequality of NIND to build up a sorted list as the index in which these data were also protected by MASPE. However, NIND does not satisfy the triangle inequality and the query results are incorrect.

8 Conclusions

In this paper, we proposed an efficient and privacy-preserving weighted similarity range query scheme (WeightedSim) for the healthcare data outsourcing scenario under the two cloud servers model. Specifically, we design an encrypted R-tree as the encrypted data index for the healthcare data using the SHE technique, followed by the corresponding SHE privacy-preserving protocols on the two cloud servers model for the range query on the encrypted R-tree. Using the provided encrypted R-tree, we setup the filtration and refinement strategies to apply the weighted similarity range query over the ciphertexts. Finally, the security analysis shows that our scheme is selectively secure as long as SHE is IND-CPA and the experimental results demonstrate that our scheme is efficient. There are still some limitations in our scheme. For example, the result in Figure 7 shows the data query time strongly related to the query threshold τ. As τ expands, the improvement of our strategies will become less effective. In addition, we did not consider the computational cost between two cloud servers during our experiment. In the future, we will adjust our scheme to be more expected and robust by introducing new methods that adapt our data query scheme to arbitrary query thresholds. Additionally, communication costs will be considered into the future works, which will be based on the two cloud servers model.

References

Ahmed, Z., Mohamed, K., Zeeshan, S. and Dong, X. (2020) 'Artificial intelligence with multi-functional machine learning platform development for better healthcare and precision medicine', *Database*, March, baaa010.

Beckmann, N., Kriegel, H-P., Schneider, R. and Seeger, B. (1990) 'The R*-tree: an efficient and robust access method for points and rectangles', *SIGMOD Rec.*, May, Vol. 19, No. 2, pp.322–331.

Boldyreva, A., Chenette, N., Lee, Y. and O'Neill, A. (2009) 'Order-preserving symmetric encryption', in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Springer, pp.224–241.

EEG (2013) *Eye State Dataset* [online] https://archive.ics.uci.edu/ml/datasets/EEG+Eye+State.

Guttman, A. (1984) 'R-trees: a dynamic index structure for spatial searching', in *Proceedings of the 1984 ACM SIGMOD International Conference on Management of Data*, pp.47–57.

Habibzadeh, H., Dinesh, K., Shishvan, O.R., Boggio-Dandry, A., Sharma, G. and Soyata, T. (2020) 'A survey of healthcare internet of things (HIoT): a clinical perspective', *IEEE Internet of Things Journal*, Vol. 7, No. 1, pp.53–71.

Hua, J., Zhu, H., Wang, F., Liu, X., Lu, R., Li, H. and Zhang, Y. (2019) 'Cinema: efficient and privacy-preserving online medical primary diagnosis with skyline query', *IEEE Internet of Things Journal*, Vol. 6, No. 2, pp.1450–1461.

Jin, J., Zheng, Y. and Xiong, P. (2021) 'EPSIM-GS: efficient and privacy-preserving similarity range query over genomic sequences', in *2021 18th International Conference on Privacy, Security and Trust (PST)*, pp.1–10.

Kumar, A. (2020) 'Using cognition to resolve duplicacy issues in socially connected healthcare for smart cities', *Computer Communications*, Vol. 152, pp.272–281.

Lin, W., Cui, H., Li, B. and Wang, C. (2021) 'Privacy-preserving similarity search with efficient updates in distributed key-value stores', *IEEE Transactions on Parallel and Distributed Systems*, Vol. 32, No. 5, pp.1072–1084.

Mahdikhani, H., Lu, R., Zheng, Y., Shao, J. and Ghorbani, A.A. (2020) 'Achieving o($\log^3 n$) communication-efficient privacy-preserving range query in fog-based IoT', *IEEE Internet of Things Journal*, Vol. 7, No. 6, pp.5220–5232.

Nasser, A.A., Alkhulaidi, A.A., Ali, M.N., Hankal, M. and Al-Olofe, M. (2019) 'A weighted Euclidean distance-statistical variance procedure based approach for improving the healthcare decision making system in Yemen', *Indian Journal of Science and Technology*, Vol. 12, No. 3, pp.1–15.

Shelat, A. and Shen, C-H. (2013) 'Fast two-party secure computation with minimal assumptions', in *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, pp.523–534.

Song, F., Qin, Z., Xue, L., Zhang, J., Lin, X. and Shen, X. (2022) 'Privacy-preserving keyword similarity search over encrypted spatial data in cloud computing', *IEEE Internet of Things Journal*, Vol. 9, No. 8, pp.6184–6198.

Toft, T. (2011) 'Sub-linear, secure comparison with two non-colluding parties', in *International Workshop on Public Key Cryptography*, Springer, pp.174–191.

Wang, B., Hou, Y., Li, M., Wang, H. and Li, H. (2014) 'Maple: scalable multi-dimensional range search over encrypted cloud data with tree-based index', in *Proceedings of the 9th ACM Symposium on Information, Computer and Communications Security*, pages 111–122.

Wang, B., Hou, Y. and Li, M. (2016) 'Practical and secure nearest neighbor search on encrypted large-scale data', in *IEEE INFOCOM 2016 – The 35th Annual IEEE International Conference on Computer Communications*, IEEE, pp.1–9.

Yao, A.C. (1982) 'Protocols for secure computations', in *23rd Annual Symposium on Foundations of Computer Science (SFCS 1982)*, IEEE, pp.160–164.

Zhang, S., Ray, S., Lu, R., Zheng, Y., Guan, Y. and Shao, J. (2021) 'Achieving efficient and privacy-preserving dynamic skyline query in online medical diagnosis', *IEEE Internet of Things Journal*, 15 June, Vol. 9, No. 12, p.1.

Zheng, Y., Lu, R., Guan, Y., Shao, J. and Zhu, H. (2021a) 'Efficient and privacy-preserving similarity range query over encrypted time series data', *IEEE Transactions on Dependable and Secure Computing*, 1 July–August, Vol. 19, No. 4, p.1.

Zheng, Y., Lu, R., Guan, Y., Shao, J. and Zhu, H. (2021b) 'Efficient privacy-preserving similarity range query with quadsector tree in ehealthcare', *IEEE Transactions on Services Computing*, 1 September-October, Vol. 15, No. 5, p.1.

Zheng, Y., Lu, R. and Zhang, S. (2022) 'Achieving privacy-preserving weighted similarity range query over outsourced ehealthcare data', *IEEE ICC'22*, Seoul, South Korea.

Websites

https://github.com/davidmoten/rtree-multi.