Hindawi
Security and Communication Networks
Volume 2018, Article ID 2373545, 10 pages
https://doi.org/10.1155/2018/2373545

Research Article

Privacy-Preserving Sorting Algorithms Based on Logistic Map for Clouds

Hua Dai,[1,2] **Hui Ren**,[1,3] **Zhiye Chen**,[4] **Geng Yang**,[1,2] **and Xun Yi**[5]

[1]*Nanjing University of Post and Telecommunication, Nanjing 210023, China*
[2]*Jiangsu Security and Intelligent Processing Lab of Big Data, Nanjing 210023, China*
[3]*China Information Consulting and Designing Institute Co., Ltd., Nanjing 210019, China*
[4]*TIZA Information Industry Corporation Inc., Nanjing 210019, China*
[5]*Royal Melbourne Institute of Technology University, Melbourne 3001, Australia*

Correspondence should be addressed to Hua Dai; daihua@njupt.edu.cn

Received 4 June 2018; Accepted 7 August 2018; Published 4 September 2018

Academic Editor: Zhaoqing Pan

Outsourcing data in clouds is adopted by more and more companies and individuals due to the profits from data sharing and parallel, elastic, and on-demand computing. However, it forces data owners to lose control of their own data, which causes privacy-preserving problems on sensitive data. Sorting is a common operation in many areas, such as machine learning, service recommendation, and data query. It is a challenge to implement privacy-preserving sorting over encrypted data without leaking privacy of sensitive data. In this paper, we propose privacy-preserving sorting algorithms which are on the basis of the logistic map. Secure comparable codes are constructed by logistic map functions, which can be utilized to compare the corresponding encrypted data items even without knowing their plaintext values. Data owners firstly encrypt their data and generate the corresponding comparable codes and then outsource them to clouds. Cloud servers are capable of sorting the outsourced encrypted data in accordance with their corresponding comparable codes by the proposed privacy-preserving sorting algorithms. Security analysis and experimental results show that the proposed algorithms can protect data privacy, while providing efficient sorting on encrypted data.

1. Introduction

With the profits from data sharing and parallel, elastic, and on-demand computing, clouds are becoming more and more popular with companies and individuals. Many kinds of services are provided by cloud service providers (CSP), such as Amazon EC2 and Alibaba Cloud. As one of the most important technologies, machine learning is very useful and wildly adopted in many areas, such as prediction [1, 2] and multimedia data processing [3, 4]. And it usually utilizes huge data volume, such as wireless multimedia data and human health data, to build intelligent models and systems for practical applications. Due to the need of large and elastic scale of storage and computing resources, those huge volume data are usually processed in clouds [5–7]. Data owner (DO) outsources their data in the cloud server (CS) for on-demand services which enhance the efficiency of complex computation such as machine learning and save the hardware/software cost.

However, in the cloud environment, DO lose direct control of their own data placed in remote CS, which may cause concerns about their outsourced data being illegally acquired or abused by CSPs, especially for sensitive data, such as national defence data and human health data. Although many CSPs claim that they deployed several safety measures in CS, such as access control, firewalls, or intrusion detection, doubts about the privacy of outsourced data obstruct the promotion and application of cloud computing. How to preserve the security and privacy of DO's outsourced data while CS providing reliable and efficient computing services has become a hot issue [8–10].

Data encryption is a common technique to protect the privacy of outsourced data on clouds, such as sensitive wireless multimedia data and human health data. Sorting is

one of the basic methods in practical applications, such as machine learning, service recommending, and data query. However, applying to sort over encrypt data on clouds is a challenge without leaking private information. The existing privacy-preserving sorting algorithms based on order-preserved encryption (OPE) [11–14] have security problems [15]. In addition, privacy-preserving sorting algorithms based on fully homomorphic encryption (FHE) [16–19] are too slow because of the complexity of FHE. It is significant to research the efficient privacy-preserving sorting algorithms for clouds.

In this paper, we assume that the honest-but-curious threat model [20] is adopted where CS strictly abides by established protocols but has the curiosity to snoop on DO's private data. On the basis of the threat model, we propose privacy-preserving sorting algorithms based on the logistic map. The main contributions of this paper are as follows. Firstly, by introducing the logistic map, we propose a secure comparison model which can be utilized to compare data without knowing their real values. Secondly, we give a data preprocessing algorithm. Data owners preprocess their private data by a symmetric encryption and logistic map. The encrypted data and corresponding comparable codes are generated and then outsourced to clouds, where the former is to protect data privacy, while the latter is to support secure comparisons. Finally, on the basis of secure comparison model, we propose privacy-preserving sorting algorithms for clouds. Also, security analysis and performance experiment are given, where results show that the proposed algorithms can protect data privacy from curious cloud administrators while providing efficient sorting on encrypted data.

The paper is organized as follows. Section 2 describes the related work. Section 3 gives the problem descriptions. Section 4 gives notations and necessary preliminaries. In Section 5, we firstly present secure comparison model on the basis of the logistic map, and then the data preprocessing algorithm and privacy-preserving sorting algorithms are given. Section 6 analyzes the security of our schemes. Section 7 gives evaluations on correctness, correlation coefficient, and performance of our proposed schemes.

2. Related Works

There are two kinds of methods achieving encrypted data sorting on clouds, one is sorting algorithms based on order-preserving encryption (OPE) [11–14], and the other is sorting algorithms based on fully homomorphic encryption [16–19].

Agrawal et al. [11] originally proposed an OPE method which is a deterministic encryption scheme whose encryption function preserves numerical ordering of the plaintext. Due to the unachievable of the indistinguishability against chosen-plaintext attack (IND-CPA) in [11], Boldyreva et al. proposed an efficient OPE scheme [12] which is based on a natural relation between a random order-preserving function and the hypergeometric probability distribution. Jaiman et al. [13] proposed an OPE algorithm by introducing shuffling, impurity insertion, and randomness in order-preserving functions. Liu et al. [14] propose a new OPE model which uses message space expansion and nonlinear space split to hide data distribution and frequency. Any proposed OPE is clearly suitable for application of privacy-preserving sorting in clouds if the data security is ensured. However, OPE is vulnerable to ciphertext-only attack [15], especially when encrypted data are massive. Therefore, those sorting algorithms based on OPE have potential security risks.

Gentry et al. [21] proposed the fully homomorphic encryption (FHE) which is a special encryption algorithm which allows computation (such as addition and multiplication) on the ciphertext. Melchor et al. [16] give an idea about sorting encrypted data by FHE. Chatterjee et al. [17] propose the sorting algorithm over encrypted data on the basis of FHE. They tried to get higher sorting efficiency by reducing costs of reencryption. Afterwards, they applied the algorithm to clouds [18, 19]. The volume of encrypted data generated by FHE is very large, due to the inclusion of big floating-point numbers which take the place of numerous storage space. Thus, the calculation of comparison for sorting based on FHE is very complex and its time efficiency is also very slow. Since the fully homomorphic encryption based sorting requires CS to reencrypt data frequently, it is not suitable for storing and managing big data on clouds.

To support efficiency and privacy in sorting algorithms for cloud environments, we propose logistic map based privacy-preserving sorting algorithms in this paper, the abstract of which has been shown in [22].

3. Problem Description

The model of the privacy-preserving sorting for clouds, proposed in this paper, is similar to recent works [18, 19]. It mainly consists of two entities, data owner (DO) and cloud server (CS). The interactions between DO and CS are introduced as follows: firstly, DO encrypts its sensitive data and generates corresponding codes which are used for privacy-preserving sorting. Then DO outsources the encrypted data and codes to CS. Secondly, CS stores the data uploaded by DO and performs data sorting over the received data. Any proved secure symmetric encryption could be adopted, such as DES and AES. If authorized users want to access DO's sensitive data, they can get the encrypted data from CS and perform decryption to obtain the plaintext data by using the shared key with DO.

In this paper, we assume that CS provides services following the curious-but-honest threat model [20]. CS is assumed to strictly follow the established protocols, but it attempts to snoop on DO's private data. There are two kinds of attacks: (1) CS has already known DO's preprocessing algorithms but does not know its initial parameters, and it tries to use exhaustive attacks against to encrypted data for plaintext information; (2) because of the massive quantity of outsourced encrypted data, statistical attacks are common methods for CS to analyze the distribution of ciphertext and speculate on the relationship between the ciphertext and plaintext to obtain plaintext information.

We focus on privacy-preserving sorting for clouds, and the key issues of this paper are introduced as follows: (1) privacy protection on outsourced data: DO preprocesses its plaintext data to keep it in confidential. Thus, CS cannot obtain DO's plaintext information via the outsourced data; (2)

Table 1: Notation descriptions.

Notations	Descriptions
μ	The bifurcation parameter of the logistic map function.
n	The iteration number of the logistic map function.
t	The constraint factor where $0 < t < x/(2 \cdot \mu^n)$
$L(t/x, n)$	The logistic mapping function.
d_i	A plaintext data item of DO.
k	The key of a symmetric encryption which is only owned by DO.
$Enc(x, k)$	The encryption function where x is the input plaintext data and k is a key.
g_i	A secure data pair after data preprocessing on d_i, $g_i = (e, c)$, where $g_i.e = Enc(d_i, k)$ and $g_i.c = L(t/x, n)$ are the corresponding encrypted data and logistic mapping codes of d_i.

privacy-preserving sorting on encrypted data: if a symmetric encryption algorithm is adopted, the privacy is guaranteed, but the encrypted data is hard for data sorting. Therefore, privacy-preserving sorting algorithms have to support sorting on encrypted data even without knowing the values.

4. Preliminaries and Notations

4.1. Preliminaries. Chaos theory [23, 24] originated in the 1960s, which has been widely adopted in medicine, astrophysics, image encryption, and hydromechanics. The basic characteristic of chaotic motion is extremely sensitive to the initial value. The difference between two chaotic motions with different initial values will become larger and larger over time. Therefore, on the basis of any given initial conditions, the chaotic motion is unpredictable.

Logistic map [25–27] is one of the important and practical chaotic motions and has been widely used in data encryption [28–32]. The equation of the logistic map is shown as

$$L(x, n) = \begin{cases} \mu \cdot L(x, n-1) \cdot (1 - L(x, n-1)) & n > 1 \\ x & n = 1, \end{cases} \quad (1)$$

where $x \in [0, 1]$, $\mu \in [0, 4]$ is called bifurcation parameter, and n is the iteration number. Studies [25–27] show that the sequence generated from (1) is chaotic if $\mu \in (3.5699456, 4]$. The output of such logistic map is extremely sensitive to the initial parameters. Any minor changes of initial parameters will lead to a tremendous difference of outputs. Therefore, the sequences generated by the logistic map are unpredictable.

4.2. Notations. The notations used in this paper are described as shown in Table 1.

5. Privacy-Preserving Sorting Algorithms

5.1. Secure Comparison Model Based on Logistic Map. The chaos characteristic of the logistic map can be used to compare data values secretly under certain conditions. Current work such as [28–32] mainly focuses on data encryption with logistic map algorithms, but there is no literature discussing the secure comparison of encrypted data with the logistic map.

If we use the logistic map for data comparison directly, then we may get wrong comparison results during sorting. But if proper constraint factors are introduced in the logistic map, we will get correct compare results invariably. The main idea of the proposed secure comparison model based on the logistic map is briefly given by three lemmas as follows.

Lemma 1. *For any given data x, where $x \geq 1$, t/x is settled as the initial value for the logistic map function, i.e., (1), where t is the constraint factor, where $0 < t < x/(2 \cdot \mu^n)$ and $\mu \in (3.5699456, 4]$. Then we have*

$$0 < L\left(\frac{t}{x}, n\right) < \frac{1}{2}. \quad (2)$$

Proof. We use mathematical inductions to prove Lemma 1 as follows.

(1) When $n = 1$, according to (1), we have

$$L\left(\frac{t}{x}, 1\right) = \frac{t}{x}. \quad (3)$$

According to the given assumptions $0 < t < x/(2 \cdot \mu)$ and $\mu \in (3.5699456, 4]$, we have $0 < L(t/x, 1) < 1/(2 \cdot \mu)$, and then $0 < L(t/x, 1) < 1/2$ is deduced.

(2) When $n = 2$, according to (1), we have

$$L\left(\frac{t}{x}, 2\right) = \mu \cdot L\left(\frac{t}{x}, 1\right) \cdot \left(1 - L\left(\frac{t}{x}, 1\right)\right). \quad (4)$$

According to the conclusion of (1), we have $1/2 < 1\text{-}L(t/x, 1) < 1$ and $L(t/x, 1) > 0$. In addition, because of $\mu \in (3.5699456, 4]$, then we deduce that

$$0 < L\left(\frac{t}{x}, 2\right) < \mu \cdot L\left(\frac{t}{x}, 1\right). \quad (5)$$

According to the given assumption $0 < t < x/(2 \cdot \mu^2)$ when $n = 2$ and the deduced result $L(t/x, 1) = t/x$ in (3), we deduce (6) from (5), where

$$0 < L\left(\frac{t}{x}, 2\right) < \frac{\mu}{(2 \cdot \mu^2)}. \quad (6)$$

Due to $\mu \in (3.5699456, 4]$, then we have that $0 < L(t/x, 2) < 1/2$ holds.

(3) We assume that Lemma 1 holds when $n = k$, i.e., $0 < L(t/x, k) < 1/2$. According to the assumption $0 < t < x/(2\cdot\mu^n)$ and the deduced result $L(t/x, 1) = t/x$ in (3), we have

$$0 < L\left(\frac{t}{x}, 1\right) < \frac{1}{(2\cdot\mu^k)}. \tag{7}$$

When $n = k + 1$, according to (1), we have

$$\begin{aligned} L\left(\frac{t}{x}, k+1\right) &= \mu\cdot L\left(\frac{t}{x}, k\right)\cdot\left(1 - L\left(\frac{t}{x}, k\right)\right) \\ &= \mu^2\cdot L\left(\frac{t}{x}, k-1\right)\cdot\left(1 - L\left(\frac{t}{x}, k-1\right)\right) \\ &\quad\cdot\left(1 - L\left(\frac{t}{x}, k\right)\right) \\ &= \mu^k\cdot L\left(\frac{t}{x}, 1\right) \\ &\quad\cdot\left(1 - L\left(\frac{t}{x}, 1\right)\right)\cdots\left(1 - L\left(\frac{t}{x}, k-1\right)\right) \\ &\quad\cdot\left(1 - L\left(\frac{t}{x}, k\right)\right). \end{aligned} \tag{8}$$

According to the assumption of (3), i.e., $0 < L(t/x, k) < 1/2$, we have

$$\begin{aligned} &\left(1 - L\left(\frac{t}{x}, 1\right)\right)\cdots\left(1 - L\left(\frac{t}{x}, k-1\right)\right) \\ &\quad\cdot\left(1 - L\left(\frac{t}{x}, k\right)\right) < 1. \end{aligned} \tag{9}$$

On the basis of (7), (8), and (9), we have

$$\begin{aligned} &L\left(\frac{t}{x}, k+1\right) \\ &\quad= \mu^k\cdot L\left(\frac{t}{x}, 1\right) \\ &\qquad\cdot\left(1 - L\left(\frac{t}{x}, 1\right)\right)\cdots\left(1 - L\left(\frac{t}{x}, k-1\right)\right) \\ &\qquad\cdot\left(1 - L\left(\frac{t}{x}, k\right)\right) < \mu^k\cdot L\left(\frac{t}{x}, 1\right) \\ &\quad< \mu^k\cdot\frac{1}{(2\cdot\mu^k)}. \end{aligned} \tag{10}$$

Therefore, we have that $L(t/x, k+1) < 1/2$ holds.

According to the above mathematical induction proofs, we have that Lemma 1 holds. □

Lemma 2. *For any given data x and y, where $1 \le x \le y$, let t/x and t/y as initial values for the logistic map function, i.e., (1), and the nth iteration results are $L(t/x, n)$ and $L(t/y, n)$, respectively, then we have*

$$L\left(\frac{t}{x}, n\right) \ge L\left(\frac{t}{y}, n\right), \tag{11}$$

where $0 < t < x/(2\cdot\mu^n)$, $\mu \in (3.5699456, 4]$, and $x \ge 1$.

Proof. We also use mathematical inductions to prove Lemma 2 as follows.

(1) When $n = 1$, according to (1), we have

$$L\left(\frac{t}{x}, 1\right) = \frac{t}{x} \tag{12}$$

and

$$L\left(\frac{t}{y}, 1\right) = \frac{t}{y}. \tag{13}$$

After applying subtraction between $L(t/x, 1)$ and $L(t/y, 1)$, we have

$$L\left(\frac{t}{x}, 1\right) - L\left(\frac{t}{y}, 1\right) = \frac{t}{x} - \frac{t}{y}. \tag{14}$$

Since $1 \le x \le y$ and $0 < t < x/(2\cdot\mu)$ are given conditions when $n = 1$, we can easily deduce

$$L\left(\frac{t}{x}, 1\right) - L\left(\frac{t}{y}, 1\right) = \frac{t}{x} - \frac{t}{y} \ge 0. \tag{15}$$

Therefore, $L(t/x, 1) \ge L(t/y, 1)$ holds.

(2) When $n = 2$, according to (1), we have

$$L\left(\frac{t}{x}, 2\right) = \mu\cdot\left(\frac{t}{x}\right)\cdot\left(1 - \frac{t}{x}\right) \tag{16}$$

and

$$L\left(\frac{t}{y}, 2\right) = \mu\cdot\left(\frac{t}{y}\right)\cdot\left(1 - \frac{t}{y}\right). \tag{17}$$

After applying subtraction between $L(t/x, 2)$ and $L(t/y, 2)$, we have

$$\begin{aligned} L\left(\frac{t}{x}, 2\right) - L\left(\frac{t}{y}, 2\right) &= \mu\cdot\left(\frac{t}{x}\right)\cdot\left(1 - \frac{t}{x}\right) - \mu\cdot\left(\frac{t}{y}\right) \\ &\quad\cdot\left(1 - \frac{t}{y}\right) \\ &= \mu\cdot\left(\frac{t}{x}\right) - \mu\cdot\left(\frac{t}{x}\right)^2 - \mu \\ &\quad\cdot\left(\frac{t}{y}\right) + \mu\cdot\left(\frac{t}{y}\right)^2 \\ &= \mu\cdot\left(\frac{t}{x} - \frac{t}{y}\right) - \mu\cdot\left(\frac{t}{x} + \frac{t}{y}\right) \\ &\quad\cdot\left(\frac{t}{x} - \frac{t}{y}\right) \\ &= \mu\cdot\left(\frac{t}{x} - \frac{t}{y}\right) \\ &\quad\cdot\left(1 - \left(\frac{t}{x} + \frac{t}{y}\right)\right). \end{aligned} \tag{18}$$

In accordance with the given conditions $1 \le x \le y$, $0 < t < x/(2\cdot\mu^n)$, $n = 2$ and $\mu \in (3.5699456, 4]$, we have

$$0 < \frac{t}{y} \le \frac{t}{x} \le \frac{1}{(2\cdot\mu^2)} < \frac{1}{2}. \tag{19}$$

Then we have

$$\frac{t}{x} - \frac{t}{y} \ge 0 \tag{20}$$

and

$$1 - \left(\frac{t}{x} + \frac{t}{y}\right) > 0. \tag{21}$$

According to (18), (20), and (21), we deduce that

$$L\left(\frac{t}{x}, 2\right) - L\left(\frac{t}{y}, 2\right) = \mu \cdot \left(\frac{t}{x} - \frac{t}{y}\right) \cdot \left(1 - \left(\frac{t}{x} + \frac{t}{y}\right)\right) \ge 0. \tag{22}$$

Therefore, $L(t/x, 2) \ge L(t/y, 2)$ holds.

(3) We assume that Lemma 2 holds when $n = k$; then we have

$$L\left(\frac{t}{x}, k\right) \ge L\left(\frac{t}{y}, k\right). \tag{23}$$

When $n = k + 1$, according to (1), we have

$$L\left(\frac{t}{x}, k+1\right) = \mu \cdot L\left(\frac{t}{x}, k\right) \cdot \left(1 - L\left(\frac{t}{x}, k\right)\right) \tag{24}$$

and

$$L\left(\frac{t}{y}, k+1\right) = \mu \cdot L\left(\frac{t}{y}, k\right) \cdot \left(1 - L\left(\frac{t}{y}, k\right)\right). \tag{25}$$

After applying subtraction between $L(t/x, k+1)$ and $L(t/y, k+1)$, we have

$$L\left(\frac{t}{x}, k+1\right) - L\left(\frac{t}{y}, k+1\right) = \mu \cdot \left(L\left(\frac{t}{x}, k\right) - L\left(\frac{t}{y}, k\right)\right) \cdot \left(1 - L\left(\frac{t}{x}, k\right) - L\left(\frac{t}{y}, k\right)\right). \tag{26}$$

According to Lemma 1, we have $0 < L(t/x, k+1) < 1/2$ and $0 < L(t/y, k+1) < 1/2$; then we deduce

$$0 < 1 - L\left(\frac{t}{x}, k\right) - L\left(\frac{t}{y}, k\right) < 1. \tag{27}$$

According to (7), (23), and (27) and the given condition $\mu \in (3.5699456, 4]$, then we have

$$L\left(\frac{t}{x}, k+1\right) - L\left(\frac{t}{y}, k+1\right) \ge 0. \tag{28}$$

Therefore, $L(t/x, k+1) \ge L(t/y, k+1)$ holds.

According to the above mathematical induction proofs, we have that Lemma 2 holds. □

Definition 3. For a given data item $x_i > 1$, $L(t/x_i, n)$ is denoted as the corresponding *comparable code* of x_i, where $L(*)$ is the logistic map function as (1).

Lemma 4. *For a given data set $X = \{x_1, x_2, \ldots, x_m\}$, where $x_i > 1$ and $i \in \{1, 2, \ldots, m\}$, we can get the corresponding comparable codes set $Y = \{y_1, y_2, \ldots, y_m\}$, where $y_i = L(t/x_i, n)$. Then we have*

$$x_i \le x_j \longleftrightarrow y_i \ge y_j, \tag{29}$$

where $x_i \in X$ and $x_j \in X$.

Proof. To prove Lemma 4, we have to prove the sufficiency and necessity of Lemma 4, respectively, i.e., $x_i \le x_j \longrightarrow y_i \ge y_j$ and $y_i \ge y_j \longrightarrow x_i \le x_j$.

(Sufficiency) According to Lemma 2, we can easily deduce $y_i \ge y_j$ when $x_i \le x_j$, where $y_i = L(t/x_i, n)$ and $y_j = L(t/x_j, n)$. The sufficiency of Lemma 4 is proved.

(Necessity) We prove the necessity of Lemma 4 by contradiction. Assuming that $x_i > x_j$ holds when $y_i \ge y_j$, where $y_i = L(t/x_i, n)$ and $y_j = L(t/x_j, n)$; then we have $L(t/x_i, n) < L(t/x_j, n)$ according to Lemma 2, i.e., $y_i < y_j$. It is obvious that the derivation is inconsistent with the given hypothesis $y_i \ge y_j$. Therefore, if we have $y_i \ge y_j$, where $y_i = L(t/x_i, n)$ and $y_j = L(t/x_j, n)$, then $x_i \le x_j$ holds.

In accordance with the sufficiency and necessity proofs, we have that Lemma 4 holds. □

According to Lemma 4, the computation of comparable is order-preserving reversely with the increasing of input data. For any two real numbers both larger than 1, we can achieve the comparison by comparing their corresponding comparable codes. Obviously, such comparison does not need to know the real values of them. If the given real numbers are less than 1, they are still comparable by using our proposed secure comparison model based on the logistic map. For example, if they are less than -1, the corresponding absolute values will be bigger than 1. And if they are between -1 and 1, by adding the constant number 2, then the result data will be also bigger than 1. Therefore, any two real numbers can be compared. As a result, we have that the proposed secure comparison model based on the logistic map is capable of performing data comparison without knowing their corresponding values. In order to describe conveniently, we focus on the data larger than 1 in the subsequent chapters.

On the basis of secure comparison model, the privacy-preserving sorting mechanism is proposed in the next sections, including the data preprocessing algorithm and privacy-preserving sorting algorithm. The brief flowchart of our proposed work is shown in Figure 1.

5.2. Data Preprocessing Algorithm. DO preprocesses its outsourced data with encryption and logistic map in order to protect private data from CS and support privacy-preserving sorting in CS. We use a symmetric encryption algorithm such as DES and AES to preserve data privacy and the logistic map is utilized to generate comparable codes for secure comparison.

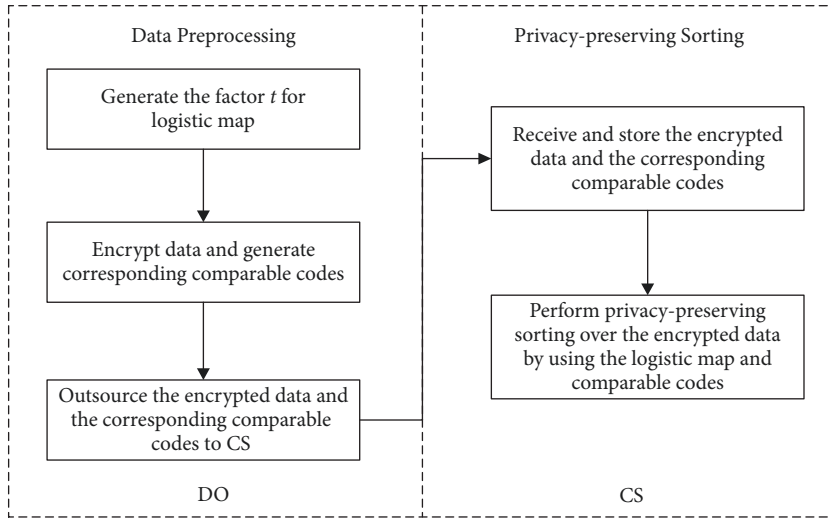

Figure 1: The brief flowchart of privacy-preserving sorting for clouds.

Begin
(1) $t = \text{rand}(0, \min(\{d_1, d_2, \ldots, d_m\})/(2 \cdot \mu^n))$;
(2) **FOR** $d_i \in \{d_1, d_2, \ldots, d_m\}$ **DO**
(3) Create a data pair g_i for d_i;
(4) $g_i.e = Enc(d_i, k)$; //generate encrypted data
(5) $g_i.c = L(t/d_i, n)$; //generate comparable code
(6) **END FOR**
(7) Upload and outsource $\{g_1, g_2, \ldots, g_m\}$ to CS;
END

Algorithm 1: **DP-LM**($\{d_1, d_2, \ldots, d_m\}$).

We assume that the outsourced data of DO are $\{d_1, d_2, \ldots, d_m\}$. After data preprocessing, a data pair $g_i = (e, c)$ will be generated for d_i, where $g_i.e$ and $g_i.c$ are the corresponding encrypted data and comparable code of d_i. In addition, we assume that k is a private key, while μ and n are bifurcation parameter and number of iterations of the logistic map function, respectively. And k, μ, and n are all owned by DO privately. The data preprocessing algorithm based on the logistic map (DP-LM) is shown in Algorithm 1.

In Algorithm 1, $\text{rand}(0, x)$ is to randomly pick a float number between 0 and x, $\min(S)$ is to get the minimum of the set S, $Enc(d_i, k)$ is to encrypt d_i with private key k by a symmetric encryption, and $L(*)$ is a logistic map function as (1). After finishing data preprocess, the generated data pairs will be uploaded and outsourced to CS.

5.3. Privacy-Preserving Sorting Algorithm. Privacy-preserving sorting is performed in CS after receiving the outsourced data from DO. Obviously, traditional sorting algorithms (e.g., merge sort, quicksort, and heap sort) cannot solve the problem of sorting encrypted data items, but by introducing the proposed secure comparison model, the encrypted data will be sorted by using the corresponding comparable codes.

We give the privacy-preserving quick sorting algorithm based on the logistic map (PQS-LM) for sorting over encrypted data in CS. The specific implementation of PQS-LM is shown in Algorithm 2.

Begin
(1) **IF** $start < end$ **THEN**
(2) $i = start$, $j = end + 1$;
(3) **WHILE** TRUE **DO**
(4) **WHILE** $i < end \wedge g_{start}.c < g_i.c$ **DO**
(5) i++;
(6) **END WHILE**
(7) **WHILE** $j > start \wedge g_j.c < g_{start}.c$ **DO**
(8) j--;
(9) **END WHILE**
(10) **IF** $i < j$ **THEN**
(11) $Swap(g_i, g_j)$;
(12) **ELSE**
(13) Finish the current loop and start the next loop;
(14) **END IF**
(15) **END WHILE**
(16) $Swap(g_j, g_{start})$;
(17) PQS-LM($\{g_1, g_2, \ldots, g_m\}$, *start*, j-1);
(18) PQS-LM($\{g_1, g_2, \ldots, g_m\}$, $j + 1$, *end*);
(19) **END IF**
END

Algorithm 2: **PQS-LM**($\{g_1, g_2, \ldots, g_m\}$, *start*, *end*).

In Algorithms 2, $Swap(x, y)$ is to swap the positions of elements x and y. The comparable codes are compared during sorting procedures, and the number of comparisons determines the efficiency of sorting. The complexity of the comparison based on our proposed model is equivalent to the comparison of plaintext. Therefore, the time complexity of PQS-LM is $O(m \cdot \log_2 m)$.

Other classic sorting algorithms, such as merge sorting and heap sorting, can also be improved to be the corresponding privacy-preserving sorting algorithms as Algorithm 2 on the basis of the proposed secure comparison model. We denote the privacy-preserving merge sorting and heap

sorting as PMS-LM and PHS-LM, respectively. Because the implementation ideas are similar to PQS-LM, we omit the details of those algorithms. The analysis and performance evaluations of our proposed privacy-preserving sorting algorithms will be given in the latter sections.

6. Security Analysis

There are two types of data outsourced in CS. One is the encrypted data generated by a symmetric encryption, and the other is the comparable code generated by the logistic map. The former is to protect data privacy, while the latter is to support secure comparisons. For encrypted data, it has an identical security level with the adopted symmetric encryption. For comparable codes generated by the logistic map, we conduct security analysis as follows.

(1) Space of initial parameters: the data preprocessing algorithm in this paper is based on a logistic chaotic system. The corresponding parameters are initialized before preprocessing, including the number of iterations n, the constraint factor t, and the bifurcation parameter μ. We assume that the attacker uses an exhaustive attack against the initial parameters. The precision of t and μ is assumed to be 10^{-p} and 10^{-q}, respectively. The space of initial parameters is $10^{p+q} \bullet n$. For example, if we take $p = 32$, $q = 32$ and randomly pick n from the interval $[1, 1000]$, then the space of initial parameters is 10^{67}. It is computation infeasible to commit successful attacks by using exhaustive search in such a large space.

(2) Sensitivity of initial parameters: since the sequence generated by the logistic map is extremely sensitive to initial parameters, any small modification of them leads to completely different results. For example, we take the same μ and n, where $\mu = 3.95362$ and $n = 3$, while we take two different constraint factors which are very close to each other, such as $t_1 = 1 \times 10^{-8}$ and $t_2 = 2 \times 10^{-8}$. For the real number 11, we will get two completely different comparable codes $L(t_1/11, 3) = 5.62 \times 10^{-8}$ and $L(t_2/11, 3) = 1.12 \times 10^{-7}$.

(3) Antistatistic ability: the logistic map has good cryptographic properties such as sensitivity to initial parameters, driven by white noise, unpredictability, etc. [31]. Even if an attacker obtains some statistic information about the input data and the corresponding comparable codes, he or she still cannot get configurations of initial parameters. Lots of simulation cases show that the data generated by the logistic map with different initial parameters are in equi-distribution [29, 32] which can prevent statistical attacks. In addition, we will give the correlation coefficient evaluation in the next section to analyze the antistatistic ability quantitatively.

As a result, we have that our proposed algorithms can support sorting over encrypted data while preserving data privacy.

7. Experiments

In this section, we give the correctness, correlation coefficient, and performance evaluations of our proposed method. The experimental datasets are generated by a random number generator. The software environment of the experiment is Windows 10 and NetBeans 8, and the hardware environment is Core i5 5200U and 8 GB DDR3 RAM.

7.1. Correctness Evaluation on Secure Comparison Model. We proposed the secure comparison model which is on the basis of the logistic map. It is the foundation of achieving the privacy-preserving sorting algorithms. Theoretical proofs are given to prove the correctness of the secure comparison model in the above sections, such as the proofs in Lemmas 1, 2, and 4. Additionally, we give the correctness evaluation on the proposed model by quantitative experiments.

In this evaluation, almost 100 thousand random numbers are generated as the input, and the corresponding comparable codes are calculated by the logistic map function with the initial parameter configuration as μ=3.67435, $n = 100$, and $t = 0.423124/(2 \times \mu^n)$. The diagram of the input data and the corresponding comparable codes are shown in Figure 2.

Figure 2 shows that the values of comparable codes decrease along with the increasing of the input data values. It indicates that the comparable codes computation is with the order-preserving property which is consistent with the proposed conclusions of our proposed secure comparison model. Therefore, the experimental result has verified the correctness of the security comparison model quantitatively.

7.2. Correlation Coefficient Evaluation on Secure Comparison Model. We use the Pearson correlation coefficient formula [33] to analyze the correlation between the input data and the corresponding comparable codes generated in secure comparison model. The correlation coefficient formula is shown as

$$C = \frac{n \cdot \sum_{x_i \in DS} \left(x_i \cdot L\left(t/x_i, n\right)\right) - \sum_{x_i \in DS} x_i \cdot \sum_{x_i \in DS} L\left(t/x_i, n\right)}{\sqrt{n \cdot \sum_{x_i \in DS} {x_i}^2 - \left(\sum_{x_i \in DS} x_i\right)^2} \cdot \sqrt{n \cdot \sum_{x_i \in DS} L\left(t/x_i, n\right)^2 - \left(\sum_{x_i \in DS} L\left(t/x_i, n\right)\right)^2}}, \tag{30}$$

where x_i and $L(t/x_i, n)$ are the input data and corresponding comparable codes, respectively, C is the correlation coefficient factor, and DS is the evaluated dataset. We calculate correlation coefficients on the basis of five datasets which are generated by a random number generator, and the results are shown in Table 2.

According to the result of Table 2, we can see that the average correlation coefficient decreases with the increasing

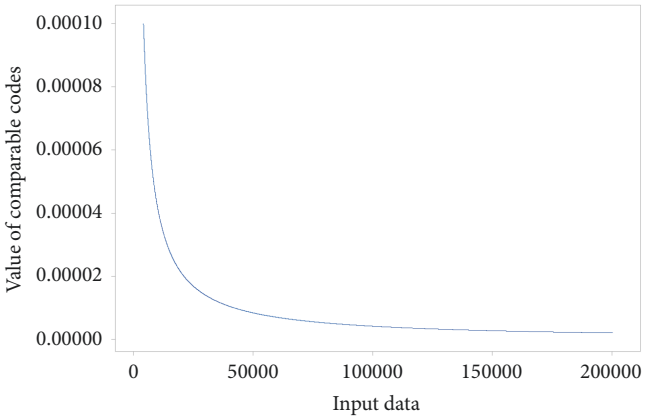

Figure 2: Values of comparable codes versus input data.

Table 2: Correlation coefficients of random datasets.

Dataset ID	Dataset scale	Average correlation coefficient
1	1000	-0.156
2	5000	-0.110
3	9000	-0.087
4	13000	-0.066
5	17000	-0.068

of dataset scales. And the average correlation coefficient is very small which indicates that the correlation between the input data and corresponding comparable codes is negligible. Therefore, our proposed secure comparison model is with the antistatistic ability.

7.3. Performance Evaluation on Algorithms. We implement our proposed logistic map-based data preprocessing and privacy-preserving sorting algorithms. To make comparisons with related works, we adopt the classic Boldyreva's order-preserving symmetric encryption (OPE) [12] to implement privacy-preserving sorting, which is more secure than [11] and more efficient than [13, 14]. We denote the OPE based data preprocessing as DP-OPE and denote the OPE based privacy-preserving quick sorting, merge sorting, and heap sorting algorithms as PQS-OPE, PMS-OPE, and PHS-OPE, respectively. Then we evaluate and compare the time cost performance of those algorithms.

It is noticeable that there are fully homomorphic encryption (FHE) based privacy-preserving sorting schemes proposed in [18, 19]. But they are too slow because of the complexity of FHE. The experiments of them show that thousands of seconds are consumed even sorting only 40 encrypted data items. Thus, we do not choose them to implement performance comparisons.

7.3.1. Evaluation on Time Cost of Data Preprocessing. The time cost of DP-OPE and DP-LM is evaluated on the basis of five given datasets. The experimental result is shown in Table 3.

Table 3: The time cost of data preprocessing (ms).

Dataset scale	DP-OPE	DP-LM
1000	1194.39	4.35
5000	4036.15	14.36
9000	6930.81	20.05
13000	10411.12	35.91
17000	14178.67	58.01

Table 3 shows that the time costs of DP-LM and DP-OPE are both increasing along with the expansion of datasets, but DP-LM is obviously much faster than DP-OPE. The reason is given as follows. DP-OPE needs to execute order-preserving encryption for plaintext by mapping amount of consecutive integers in a domain to integers in a much larger range. Each integer is assigned a pseudorandom value in its subrange. The OPE algorithm recursively bisects the range and samples from the domain at each recursion until it hits the input plaintext value. Thus, the calculation load of OPE is higher relatively which makes DP-OPE much slower than DP-LM.

7.3.2. Evaluation on Time Cost of Data Sorting. We also use the same datasets to evaluate the privacy-preserving sorting algorithms based on the logistic map and OPE. The result is shown in Table 4.

The experimental result in Table 4 shows that the performance of our proposed privacy-preserving sorting algorithms is better than those sorting algorithms based on OPE, respectively. The reason is that the output data of OPE, which is used for privacy-preserving sorting, is more complex than the comparable codes generated by the logistic map.

8. Conclusions

When the clouds provide outsourcing services, the privacy of outsourced data, such as national defence data and human health data, can be protected by common encryption. However, those encrypted data are useless for data sorting which is a common operation in many areas, such as machine learning, service recommending, and data query. It is a challenge to achieve privacy-preserving sorting in clouds. In this paper, we introduce a secure comparison model based on the logistic map and propose privacy-preserving sorting algorithms. The security analysis and experimental result show that the proposed algorithms can protect data privacy while providing efficient sorting on encrypted data.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Disclosure

The abstract of this paper appeared in the 4th International Conference on Cloud Computing and Security (ICCCS 2018), June 8-10, Haikou. This version is the full paper.

Table 4: The time cost of privacy-preserving algorithms (ms).

Dataset scale	Privacy-preserving quick sorting		Privacy-preserving merge sorting		Privacy-preserving heap sorting	
	PQS-LM	PQS-OPE	PMS-LM	PMS-OPE	PHS-LM	PHS-OPE
1000	0.63	0.65	0.76	0.81	0.74	0.75
5000	1.02	1.55	5.45	6.31	2.35	3.31
9000	3.15	4.23	7.23	7.85	3.19	4.12
13000	3.91	4.41	8.03	9.47	3.92	4.65
17000	4.65	5.01	11.75	12.31	5.31	5.45

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was supported by the National Natural Science Foundation of China under Grant nos. 61872197, 61572263, 61672297, 61502251, and 61472193; the Natural Science Foundation of Jiangsu Province under Grant nos. BK20151511, BK20141429, and BK20161516; the Postdoctoral Science Foundation of China under Grant no. 2015M581794; the Natural Science Foundation of Anhui Province under Grant no. 1608085MF127; and the Natural Research Foundation of Nanjing University of Posts and Telecommunications under Grant no. NY217119.

References

[1] P. H. Abreu, M. S. Santos, M. H. Abreu, B. Andrade, and D. C. Silva, "Predicting Breast Cancer Recurrence Using Machine Learning Techniques," *ACM Computing Surveys*, vol. 49, no. 3, pp. 1–40, 2016.

[2] D. E. Jones, H. Ghandehari, and J. C. Facelli, "A review of the applications of data mining and machine learning for the prediction of biomedical properties of nanoparticles," *Computer Methods and Programs in Biomedicine*, vol. 132, pp. 93–103, 2016.

[3] L. Zhu, Y. Zhang, Z. Pan, R. Wang, S. Kwong, and Z. Peng, "Binary and multi-class learning based low complexity optimization for HEVC encoding," *IEEE Transactions on Broadcasting*, vol. 63, no. 3, pp. 547–561, 2017.

[4] Z. Pan, J. Lei, Y. Zhang, and F. L. Wang, "Adaptive fractional-Pixel motion estimation skipped algorithm for efficient HEVC motion estimation," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 14, no. 1, pp. 1–19, 2018.

[5] J. Wang, J. Wang, Y. Wu et al., "A Machine Learning Framework for Resource Allocation Assisted by Cloud Computing," *IEEE Network*, vol. 32, no. 2, pp. 144–151, 2018.

[6] J. Zhu and X. Li, "Scheduling for multi-stage applications with scalable virtual resources in cloud computing," *International Journal of Machine Learning and Cybernetics*, vol. 8, no. 5, pp. 1633–1641, 2017.

[7] D. C. Nascimento, C. E. Pires, and D. G. Mestre, "Applying machine learning techniques for scaling out data quality algorithms in cloud computing environments," *Applied Intelligence*, vol. 45, no. 2, pp. 530–548, 2016.

[8] Y. Liu, H. Peng, and J. Wang, "Verifiable Diversity Ranking Search Over Encrypted Outsourced Data," *Computers, Materials & Continua*, vol. 55, no. 1, pp. 37–57, 2018.

[9] S. H. Albakri, B. Shanmugam, G. N. Samy, N. B. Idris, and A. Ahmed, "Security risk assessment framework for cloud computing environments," *Security and Communication Networks*, vol. 7, no. 11, pp. 2114–2124, 2014.

[10] J. Cheng, R. Xu, X. Tang, V. Sheng, and C. Cai, "An Abnormal Network Flow Feature Sequence Prediction Approach for DDoS Attacks Detection in Big Data Environment," *Computers, Materials & Continua*, vol. 55, no. 1, pp. 95–119, 2018.

[11] R. Agrawal, J. Kiernan, R. Srikant, and Y. R. Xu, "Order preserving encryption for numeric data," in *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD '04)*, pp. 563–574, ACM, New York, NY, USA, June 2004.

[12] A. Boldyreva, N. Chenette, Y. Lee, and A. O'Neill, "Order-preserving symmetric encryption," in *Advances in Cryptology-EUROCRYPT 2009*, vol. 5479, pp. 224–241, Springer, Berlin, Germany, 2009.

[13] V. Jaiman and G. Somani, "An order preserving encryption scheme for cloud computing," in *Proceedings of the 7th International Conference on Security of Information and Networks, SIN 2014*, pp. 211–216, Glasgow, UK, September 2014.

[14] Z. Liu, X. Chen, J. Yang, C. Jia, and I. You, "New order preserving encryption model for outsourced databases in cloud environments," *Journal of Network and Computer Applications*, vol. 59, pp. 198–207, 2016.

[15] M. Naveed, S. Kamara, and C. V. Wright, "Inference attacks on property-preserving encrypted databases," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, CCS 2015*, pp. 644–655, New York, NY, USA, October 2015.

[16] C. Aguilar-Melchor, S. Fau, C. Fontaine, G. Gogniat, and R. Sirdey, "Recent advances in homomorphic encryption: A possible future for signal processing in the encrypted domain," *IEEE Signal Processing Magazine*, vol. 30, no. 2, pp. 108–117, 2013.

[17] A. Chatterjee, M. Kaushal, and I. Sengupta, "Accelerating Sorting of Fully Homomorphic Encrypted Data," in *Progress in Cryptology – INDOCRYPT 2013*, vol. 8250 of *Lecture Notes in Computer Science*, pp. 262–273, Springer, 2013.

[18] A. Chatterjee and I. Sengupta, "Searching and sorting of fully homomorphic encrypted data on cloud," *IACR Cryptology ePrint Archive*, p. 981, 2015.

[19] A. Chatterjee and I. Sengupta, "Translating Algorithms to Handle Fully Homomorphic Encrypted Data on the Cloud," *IEEE Transactions on Cloud Computing*, vol. 6, no. 1, pp. 287–300, 2018.

[20] W. Fu, B. Yan, and X. Wu, "Data possession provability on semi-trusted cloud storage," in *Proceedings of the 4th International Conference Cloud Computing*, pp. 199–209, Springer, 2013.

[21] C. Gentry and S. Halevi, "Implementing Gentry's fully-homomorphic encryption scheme," *IACR Cryptology ePrint Archive*, p. 520, 2010.

[22] H. Dai, H. Ren, Z. Y. Chen et al., "Privacy-Preserving Sorting Algorithms based on Logistic Map for Clouds," in *Proceedings of the 4th International Conference on Cloud Computing and Security (ICCCS 2018)*, 2018.

[23] E. N. Lorenz, "Deterministic nonperiodic flow," *Journal of the Atmospheric Sciences*, vol. 20, no. 2, pp. 130–141, 1963.

[24] J. Banks, J. Brooks, G. Cairns, G. Davis, and P. Stacey, "On Devaney's definition of chaos," *The American Mathematical Monthly*, vol. 99, no. 4, pp. 332–334, 1992.

[25] B. Yang and X. Liao, "Period analysis of the Logistic map for the finite field," *Science China Information Sciences*, vol. 60, no. 2, p. 22302, 2017.

[26] Y. Deng, H. Hu, W. Xiong, N. N. Xiong, and L. Liu, "Analysis and Design of Digital Chaotic Systems with Desirable Performance via Feedback Control," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45, no. 8, pp. 1187–1200, 2015.

[27] S. C. Phatak and S. S. Rao, "Logistic map: a possible random-number generator," *Physical Review E: Statistical, Nonlinear, and Soft Matter Physics*, vol. 51, no. 4, pp. 3670–3678, 1995.

[28] G. Ye and X. Huang, "A secure image encryption algorithm based on chaotic maps and SHA-3," *Security and Communication Networks*, vol. 9, no. 13, pp. 2015–2023, 2016.

[29] N. K. Pareek, V. Patidar, and K. K. Sud, "Image encryption using chaotic logistic map," *Image and Vision Computing*, vol. 24, no. 9, pp. 926–934, 2006.

[30] B. Murugan, A. G. Nanjappa Gounder, and S. Manohar, "A hybrid image encryption algorithm using chaos and Conway's game-of-life cellular automata," *Security and Communication Networks*, vol. 9, no. 7, pp. 634–651, 2016.

[31] S. G. Mallat, "Theory for multiresolution signal decomposition: the wavelet representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 7, pp. 674–693, 1989.

[32] V. Patidar, K. K. Sud, and N. K. Pareek, "A pseudo random bit generator based on chaotic logistic map and its statistical testing," *Informatica*, vol. 33, no. 4, pp. 441–452, 2009.

[33] Wikipedia, "Pearson correlation coefficient," 7 April 2018, https://en.wikipedia.org/wiki/Pearson_correlation_coefficient.