# Image encryption algorithm for synchronously updating Boolean networks based on matrix semi-tensor product theory

Xingyuan Wang*, Suo Gao

(School of Information Science and Technology, Dalian Maritime University, Dalian 116026, China)

**Abstract:** This paper studies chaotic image encryption technology and an application of matrix semi-tensor product theory, and a Boolean network encryption algorithm for a synchronous update process is proposed. A 2D-LASM chaotic system is used to generate a random key stream. First, a Boolean network is coded, and a Boolean matrix is generated. If necessary, the Boolean network matrix is diffused in one round so that the Boolean matrix can be saved in the form of an image. Then, three random position scramblings are used to scramble the plaintext image. Finally, using a matrix semi-tensor product technique to generate an encrypted image in a second round of diffusion, a new Boolean network can be generated by encoding the encrypted image. In secure communications, users can choose to implement an image encryption transmission or a Boolean network encryption transmission according to their own needs. Compared with other algorithms, this algorithm exhibits good security characteristics.

**Keywords:** Chaotic image encryption; Matrix semi-tensor product; Boolean network; 2D-LASM chaotic system

## 1 Introduction

Chaos is considered to be the third great scientific revolution of the 20th century after relativity and quantum mechanics [14, 24, 34, 42]. Because a chaotic system has the characteristics of pseudorandomness, initial value sensitivity, parameter sensitivity and unpredictability [1, 12, 22, 37], it is very suitable for image encryption [8, 23, 35, 38]. An increasing number of chaotic systems have been proposed, and an increasing number of image encryption algorithms have been produced [18, 39, 44].

Considering a synchronously updating Boolean network as a gene regulatory network in which the genes are confidential materials in every country, it becomes necessary to protect the Boolean network from theft in the transmission process. Semi-tensor product theory is a generalization of general matrix theory and has the properties that are needed for symmetric encryption, so it can be used in encryption [15, 41, 45]. On the basis of chaotic image encryption, this paper presents a synchronously updating Boolean network encryption algorithm using matrix semi-tensor product theory.

In recent years, many studies have been conducted on complex networks [2, 26, 28]. Similar to a neural network, a gene regulatory network is also a kind of special complex system based on a network structure [19, 20, 30]. First proposed by Kauffman, the Boolean network is a kind of gene network. It is a discrete system based

*Corresponding authors. E-mail addresses: xywang@dlmu.edu.cn (X. Wang), 1418159118@qq.com (S. Gao).

on a directed graph [9]. Each gene node has two states, on and off, which are commonly represented by 0 and 1, respectively, and the gene can only be in one state at a given time [27]. The state of each node in a Boolean network is determined by one or more nodes of the previous moment. The state of the previous node is used as an input parameter, and after a series of logical operations, the state of the next node is obtained [3]. In operations, logical operators include NOT, AND, OR, XOR, EQV, and IMP [16, 17].

The matrix semi-tensor product is a new type of matrix multiplication [43] proposed by Cheng Daizhan. The matrix semi-tensor product is an operation between a matrix tensor product and ordinary matrix multiplication, which is a generalization of common matrix multiplication [11, 31]. The matrix semi-tensor product overcomes the drawbacks of traditional matrices and makes it possible for two matrices of different sizes to perform multiplication operations. At the same time, the semi-tensor product preserves all the main properties of traditional matrix multiplication, and it also has some better mathematical properties, such as pseudocommutativity, making it a good mathematical tool [10].

In traditional encryption algorithms, attention is mostly focused on the encryption of a two-dimensional image [6, 21, 25]. In this paper, based on matrix semi-tensor product theory, an algorithm for synchronously updating Boolean network encryption is proposed, and the transmission of a network is encrypted. There are many ways for attackers to crack an encryption algorithm. Attackers usually intercept the secret key and then find a common encryption algorithm in a database to crack the algorithm [5, 13]. The matrix semi-tensor product technology proposed in this paper is applied to the field of image encryption for the first time. The encryption algorithm cannot be found in a database, so it is very difficult to crack. In this paper, the matrix semi-tensor product technique is applied in the context of diffusion. Different from the diffusion process adopted by a fixed formula by using a simple XOR operation [4, 7, 29], the algorithm proposed in this paper can use a plaintext matrix of different sizes in the diffusion process, which increases the diversity of the diffusion and also increases the security of the algorithm. In addition, while Refs. [36, 40] requires multiple cycles of encryption to achieve a secure effect, this paper only needs requires round of encryption to achieve a secure effect.

The remainder of this paper is organized as follows. Section 2 introduces the relevant theory in detail. Section 3 introduces some of the functions and the preparation before the encryption process. Section 4 presents the encryption algorithm proposed in this paper. Section 5 presents a simulation of a Boolean network and a simulation of common images. Section 6 focuses on security analysis. Section 7 presents a summary of this paper and proposes future work.

# 2 Relevant Knowledge

In this section, we introduce matrix semi-tensor product theory, the Boolean network model and the 2D-LASM system.

## 2.1 Matrix Semi-tensor Product

Matrix theory is not only used in mathematics but is also widely used in various cross-disciplinary fields; it is a very important mathematical tool. Traditional matrix theory faces many problems in practical applications. For example, it is difficult to use traditional matrix multiplication in high-dimensional array operations. To apply matrix multiplication to the calculation of multilinear and even nonlinear problems, Cheng proposed the concept of the matrix semi-tensor product [10, 11].

In recent years, the tensor product has been widely studied [32, 33]. The semi-tensor product has not only attracted extensive attention in the fields of control and biology, but it has also achieved many results. However, this theory has not been applied in the field of chaotic image encryption. In matrix semi-tensor product theory, as long as the guarantee matrix is reversible, the multiplication process is reversible, and the characteristics of the symmetric encryption are satisfied; thus, in this case, the theory can be applied to the field of image encryption.

The following is a brief introduction to matrix semi-tensor product theory.

**Definition 1** [11] Suppose $A=(a_{ij})\in M_{m\times n}$ and $B=(b_{ij})\in M_{s\times t}$. The tensor product of $A$ and $B$ can be expressed as:

$$A\otimes B=\begin{bmatrix} a_{11}B & a_{12}B & \cdots & a_{1n}B \\ a_{21}B & a_{22}B & \cdots & a_{2n}B \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1}B & a_{m2}B & \cdots & a_{mn}B \end{bmatrix},$$

where "$\otimes$" is the symbol used to represent a matrix tensor product.

**Definition 2** [11] (1) Let $X=[x_1,x_2,\cdots,x_s]$ be a row vector and $Y=[y_1,y_2,\cdots,y_t]^{\mathrm{T}}$ be a column vector.

Scenario 1: If $t$ is a factor of $s$ and $s=t\times n$, then the $n$-dimensional row vector

$$\left\langle X,Y\right\rangle_L:=\sum_{k=1}^{t}X^k y_k\in\mathbb{R}^n$$

is called the left matrix semitensor product of $X$ and $Y$, where $X=[X^1,X^2,\cdots,X^t]$ and $X^i\in\mathbb{R}^n, i=1,2,\cdots,t.$

Scenario 2: If $s$ is a factor of $t$ and $t=s\times n$, then the $n$-dimensional row vector

$$\left\langle X,Y\right\rangle_L:=(\left\langle Y^{\mathrm{T}},X^{\mathrm{T}}\right\rangle_L)^{\mathrm{T}}\in\mathbb{R}^n$$

is also called the left matrix semi-tensor product of $X$ and $Y$.

(2) Suppose $M\in M_{m\times n}$ and $N\in M_{p\times q}$. If $n$ is a factor of $p$ or $p$ is a factor of $n$, $C=M\ltimes N$ is the left matrix semi-tensor product of $M$ and $N$. If $C$ consists of $m\times p$ blocks, $C=(C^{ij})$ and

$$C^{ij}=\left\langle M^i,N_j\right\rangle_L,\ i=1,2,\cdots,m,\ j=1,2,\cdots,q\ . \tag{1}$$

**Proposition 1** [11] (1) If $A\in M_{m\times np}$ and $B\in M_{p\times q}$, define

$$A\ltimes B=A(B\otimes I_n). \tag{2}$$

(2) If $A\in M_{m\times n}$ and $B\in M_{np\times q}$, define

$$A\ltimes B=(A\otimes I_p)B.$$

$I_n$ is a unit matrix of order $n$, and $I_p$ is a unit matrix of order $p$.

**Theorem 1** Suppose $C = A_{m\times np} \ltimes B_{n\times p}$. If $\left|B \otimes I_p\right| \neq 0$,

$$A = C(B \otimes I_p)^{-1}. \tag{3}$$

**Prove:** Because $C = A \ltimes B = A(B \otimes I_p)$ and $\left|B \otimes I_p\right| \neq 0$, $B \otimes I_p$ is reversible.

Because $C(B \otimes I_p)^{-1} = A(B \otimes I_p)(B \otimes I_p)^{-1}$, $A = C(B \otimes I_p)^{-1}$.

The proof is complete.

**Example 1** Let $X = \begin{bmatrix} 1 & 2 & 3 & 1 \\ 1 & 7 & 4 & 6 \\ 3 & 0 & 8 & 5 \end{bmatrix}$ and $Y = \begin{bmatrix} 1 & 3 \\ 2 & 1 \end{bmatrix}$.

With the application of Eq. (1), we can obtain the following:

$$X \ltimes Y = \begin{bmatrix} (1 & 2)\times 1 + (3 & 1)\times 2 & (1 & 2)\times 3 + (3 & 1)\times 1 \\ (1 & 7)\times 1 + (4 & 6)\times 2 & (1 & 7)\times 3 + (4 & 6)\times 1 \\ (3 & 0)\times 1 + (8 & 5)\times 2 & (3 & 0)\times 3 + (8 & 5)\times 1 \end{bmatrix}$$

$$= \begin{bmatrix} (7 & 4) & (6 & 7) \\ (9 & 19) & (7 & 27) \\ (19 & 10) & (17 & 5) \end{bmatrix} = \begin{bmatrix} 7 & 4 & 6 & 7 \\ 9 & 19 & 7 & 27 \\ 19 & 10 & 17 & 5 \end{bmatrix}.$$

## 2.2 Boolean Network

A Boolean network is a simplified logical model that is used to describe gene regulatory networks. In Boolean networks, each node represents a biological gene state, and the values are limited to binary numbers 0 and 1 corresponding to a transformation [9]. The value 0 indicates that the node is in a suppressed state, and the value 1 indicates that the node is in an active state. Each node is given a Boolean function, which is generated by Boolean operations between multiple nodes [27]. In other words, the state of each gene in a Boolean network is determined by the adjacent associated genes [3].

According to the node update rules of the Boolean network, Boolean networks can be divided into synchronously updating Boolean networks and asynchronously updating Boolean networks. In a synchronously updating Boolean network, each node is updated at the same time. In an asynchronously updating Boolean network, any node has two states: update and do not update, and the nodes may not be updated at the same time.

Because a Boolean network completely describes the dynamics of a gene regulatory network and a gene regulation process, it is used to simulate gene regulatory networks, neural networks and biological evolution networks. Boolean networks have potential application value in genetics and other fields. For example, a Boolean network can be used for gene recombination, gene mutation and gene transcription. Genetic resources are strategic resources of countries and play an important role in national health, ecological protection and the development of biological industry. One gene can enable a country to prosper. It is important to protect genes from being stolen during transmission. Therefore, to solve this problem, this paper proposes an encryption algorithm for synchronously updating Boolean networks.

The equations of Boolean networks are:

$$\begin{cases} x_1(t+1) = f_1(x_1(t), x_2(t), \cdots, x_n(t)) \\ x_2(t+1) = f_2(x_1(t), x_2(t), \cdots, x_n(t)) \\ \vdots \\ x_n(t+1) = f_n(x_1(t), x_2(t), \cdots, x_n(t)) \end{cases}, \tag{4}$$

where $x_i(t)$ represents the state of each gene point at time $t$, and $f_i$, $i = 1,2,3,\cdots,n$, is a logical operation function; $f_i$ is also the updated calculation rule for the status of each gene node. In the operation, the logical operators include NOT ( $\neg P$ ), AND ( $P \wedge Q$ ), OR ( $P \vee Q$ ), XOR ( $P \oplus Q$ ), EQV ( $P \leftrightarrow Q$ ), and IMP ( $P \rightarrow Q$ ).

**Example 2** In a synchronously updating Boolean network,

$$\begin{cases} x_1(t+1) = x_3(t) \oplus x_4(t) \wedge \neg x_5(t) \\ x_2(t+1) = x_1(t) \leftrightarrow x_3(t) \\ x_3(t+1) = \neg x_2(t) \rightarrow x_3(t) \\ x_4(t+1) = x_2(t) \rightarrow x_4(t) \vee x_5(t) \\ x_5(t+1) = x_1(t) \end{cases}.$$

If the values of $x_1(0), x_2(0), x_3(0), x_4(0), \text{and } x_5(0)$ at $t = 0$ are given, the state of each node at $t = 1$ can be obtained, and further, the state of the node at $t = n$ can be obtained.

## 2.3 2D-LASM System

In this paper, a two-dimensional chaotic system is selected — the 2D-LASM system. This system has extremely complex dynamic behavior and is widely used in the field of secure communication. Its mathematical expression is as follows [13]:

$$\begin{cases} x_{i+1} = \sin(\pi\mu(y_i + 3)x_i(1 - x_i)) \\ y_{i+1} = \sin(\pi\mu(x_{i+1} + 3)y_i(1 - y_i)) \end{cases}. \tag{5}$$

When $\mu \in [0.37,0.38] \cup [0.4,0.42] \cup [0.44,0.93] \cup \{1\}$, 2D-LASM maps to a chaotic state, and the resulting sequence $\{(x_n, y_n), n = 0,1,2,3...\}$ is aperiodic, nonconvergent and very sensitive to initial values. To avoid the periodic window, this article uses the parameter $\mu = 0.9$.

# 3 Early Preparation

In this section, we introduce the preparation of the required function, the key stream generated by 2D-LASM chaotic system and the preparation steps before the Boolean network encryption.

## 3.1 Function Declaration

**Definition 3** Define a summation function $S(P)$, where $P \in M_{m \times n}$. Suppose

$$P = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1n} \\ p_{21} & p_{22} & \cdots & p_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ p_{m1} & p_{m2} & p_{m3} & p_{mn} \end{bmatrix},$$

and $S(P)$ can be defined as follows:

$$S(P)=\sum_{i=1}^{m}\sum_{j=1}^{n}p_{ij}\,.$$

**Definition 4** Define a function $f(x)$ that represents the last three bits of $x$, where $x$ is an arbitrary real integer and $x>100$.

**Example 3** Let $x=98562136$ ;thus, $f(x)=136$.

**Definition 5** Define a sort function

$$[Q,index]=sort(P),$$

where $Q$ is the $P$ sorted vector, and $index$ is the index of each item in $Q$ corresponding to the item in $P$.

**Example 4** Let $P=[1,7,8,2,4,3,9,5,6]$; then, $[Q,index]=sort(P)$ as follows:

$$Q=[1,2,3,4,5,6,7,8,9],$$
$$index=[1,4,6,5,8,9,2,3,7].$$

**Definition 6** Define a function that can change the size of a matrix:

$$G=reshape(P,\ a,\ b),$$

where $P_{m\times n}$ represents the matrix, $a$ and $b$ are two positive integers and $m\times n=a\times b$. This function represents the scale of the changed matrix $P$, which is changed from the original $m\times n$ to $a\times b$, and the output matrix is $G_{a\times b}$.

**Definition 7** Define a residual function $[E,J]=mod(P,a)$ that represents each component of the matrix and the quotient operation of a positive integer, where $P$ is a dividend and a matrix, $a$ is a divisor, and the value of $a$ is a positive integer. The function return values $E$ and $J$ represent the quotient's integer matrix and remainder matrix, respectively.

## 3.2 Key Stream Generated By A 2D-LASM Chaotic System

The 2D-LASM chaotic system is used in this paper, its mathematical expression is shown in Eq. (5), and here we select the parameter $\mu=0.9$ and the iteration initial values $x_0=0.3842$ and $y_0=0.9625$. In the 2D-LASM chaotic system, when a sequence reaches a sufficiently chaotic state, the system begins to retain the chaotic sequence. Therefore, it is necessary to discard the previous $(K-1)$ points and keep the chaotic sequences from the $K$ point; in general, $K\geq 200$ can satisfy the requirements. The $K$ value is related to the pixel value of the original image, and different images can be intercepted to produce different chaotic sequences.

The plaintext image pixel matrix is $P_{m\times n}$:

$$P=\begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1n} \\ p_{21} & p_{22} & \cdots & p_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ p_{m1} & p_{m2} & p_{m3} & p_{mn} \end{bmatrix}, \tag{6}$$

where $n=2^t$, $t=3,4,5,\cdots$. (This provision is not necessary but is intended to facilitate diffusion). Then,

according to Definition 3 and Definition 4, we can set

$$k = f(S(P)^2),$$

$K$ is then defined as:

$$K = \begin{cases} k, & k \geq 200 \\ k + 200, & k < 200 \end{cases}.$$

According to the above description, two chaotic sequences $X'$ and $Y'$ are generated by iterating the sequences. Remove the first $K$ point, and select the previous $m \times n$ points to generate two chaotic sequence records as follows:

$$X = [x_1, x_2, x_3, ..., x_{m \times n}],$$
$$Y = [y_1, y_2, y_3, ..., y_{m \times n}]. \tag{7}$$

To generate the required scale matrix for the following encryption process, the resulting two chaotic sequences are processed as follows:

**Step 1:** Take the former $n$ elements of $X$ and apply the sorting function of Definition 5 to sort the sequence. The function returns the value of $index = A_1$ and $A_1 \in M_{1 \times n}$.

**Step 2:** Then, Step 1 takes $m$ elements from the remaining elements of $X$ and applies the sorting function of Definition 5 to sort the sequence. The function returns the value of $index = A_2$ and $A_2 \in M_{1 \times m}$.

**Step 3:** Record the previous $2^{t-3} \times 2^{t-3}$ elements of $X$ as $A_3$, where $A_3 \in M_{1 \times 2^{t-3} \cdot 2^{t-3}}$. Then, apply the function of Definition 6 to $A_3$:

$$A_4 = reshape(A_3, 2^{t-3}, 2^{t-3}). \tag{8}$$

Finally, obtain the matrix $A_4$, which we call the diffusion matrix, where $A_4 \in M_{2^{t-3} \times 2^{t-3}}$. ( $A_4 \otimes I_8$ is reversible, but if it is irreversible, we can use a scrambling algorithm to make it reversible.)

**Step 4:** Record the previous $2^{t-2} \times 2^{t-2}$ elements of $Y$ as $A_5$, where $A_5 \in M_{1 \times 2^{t-2} \cdot 2^{t-2}}$. Then, apply the function of Definition 6 to $A_5$:

$$A_6 = reshape(A_5, 2^{t-2}, 2^{t-2}). \tag{9}$$

Finally, obtain the matrix $A_6$, also a diffusion matrix, where $A_6 \in M_{2^{t-2} \times 2^{t-2}}$. ( $A_6 \otimes I_4$ is reversible, but if it is irreversible, we can use a scrambling algorithm to make it reversible.)

**Step 5:** Sort $Y'$ by applying the sort function of Definition 5, which returns a value of $index = A_7$, where $A_7 \in M_{1 \times mn}$. Then, apply the function of Definition 6 to $A_7$:

$$A_8 = reshape(A_7, m, n), \tag{10}$$

And finally obtain the matrix $A_8$, where $A_8 \in M_{m \times n}$. In this way, images of the same size can produce the same scrambling matrix, which greatly reduces the encryption time.

## 3.3 Boolean Network Coding

The form of the equations of Boolean networks is shown in Eq. (4). The following scheme encodes a Boolean network so that it can be saved in the form of a matrix that we call it the Boolean matrix, as shown in Table 1:

Table 1 Boolean network coding rules

| Symbolic name | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | $x_6$ | $x_7$ | $x_8$ | … | $x_n$ | $x_0$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Code rule | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | … | $n$ | 0 |
| Symbolic name | $\wedge$ | $\vee$ | $\oplus$ | $\rightarrow$ | $\leftrightarrow$ | $\wedge\neg$ | $\vee\neg$ | $\oplus\neg$ | $\rightarrow\neg$ | $\leftrightarrow\neg$ | $\neg$ |
| Code rule | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 0 |

Boolean networks with 16 gene nodes and the Boolean network coding are shown in Example 5 of Appendix A.

In addition, there are the following rules for converting Boolean networks into Boolean matrices:

(1) We stipulate that the number of rows of a matrix is $2^t$, where $t = 3,4,5,6\cdots$. This step is not necessary but facilitates the following diffusion process.

(2) The $1 \sim n$ line of the matrix represents a Boolean function of $n$ nodes.

(3) The $1 \sim m/2$ column of the matrix represents the determining gene for each Boolean function. The column value according to the coding rule is an integer in $[0, n]$, where 0 simply represents the off state.

(4) The $m/2+1$ column of the matrix indicates whether the first gene of each Boolean function is in an $x$ state or a $\neg x$ state. If the gene is in an $x$ state, according to the above coding rule, the corresponding column is 1; in the case of a $\neg x$ state, the corresponding column is 0.

(5) The $(m/2+1) \sim m$ column of the matrix represents the logical relation of adjacent genes of a Boolean function. According to the encoding rule, the column value is an integer in $[0,10]$. Here, 0 stands for no logical definition.

According to the above coding description, the number of columns in Boolean networks can only be even. The generated matrix is transposed, and the Boolean matrix is generated. The Boolean matrix is represented by the symbol $B$.

We use Boolean networks in the coding process, the Boolean networks operate in the form of matrices, and the time of the coding process is not within the scope of our consideration. Some other operations of the Boolean network, such as synchronization and antisynchronization, are not within the scope of our study, and we are only responsible for the encryption of the Boolean network, which prevents the network from being intercepted in the transmission process.

# 4 Algorithm Description

Based on the chaotic characteristics of the 2D-LASM system, an efficient and high-security image encryption algorithm is proposed. In contrast to the traditional image encryption algorithm, this algorithm uses a matrix with a size that is different from that of the original image to perform the multiplication operation, thereby achieving the diffusion effect. Three random position scramblings are used in the scrambling process. This algorithm can encrypt images $m \times n$ in size, and images of different sizes can generate different diffusion matrices. Encryption is a process of diffusion, scrambling and diffusion. This method achieves good encryption and better security than the traditional method. The encryption and decryption procedures are described below.

## 4.1 A Round of Diffusion

According to the coding rules of Section 3.3, Boolean networks can eventually be converted into Boolean matrices, but the pixel range of the image is an integer between 0 and 255; therefore, the generated Boolean matrix $B_{m\times n}$ should be processed by one diffusion step so that the value of the Boolean matrix is an integer between 0 and 255. The objective is to store the Boolean matrix in the form of an image. Of course, if the Boolean matrix values are less than 255, then this diffusion step can be omitted.

The function of Definition 7 is applied to the cell matrix $B_{m\times n}$. Finally, the pixel matrix $P$ of the plaintext image is obtained as follows:

$$[E_1, P] = mod(B, 256). \tag{11}$$

At this point, the value of the plaintext image pixel matrix is an integer between 0 and 255. The Boolean matrix is stored in the form of an image. The next step is chaotic image encryption, which is a process involving scrambling and diffusion.

## 4.2 Three-random-scrambling Algorithm

Three random scramblings are performed on the plaintext pixel matrix $P_{m\times n}$ that is obtained from Eq. (11).

The "three-random-scrambling algorithm" is described with the following steps:

**Step 1:** First, provide the initial values $i = 1, j = 1$.

**Step 2:** Apply the formula $P_1(1, A_8(i, j)) = P(A_2(1, i), A_1(1, j))$ to scrambling, where $A_1, A_2, A_8$ are shown in Section 3.3.

**Step 3:** Order $i = i + 1$; if $i = m + 1$, end this step, and continue to the next step; otherwise go back to Step 2.

**Step 4:** Order $i = 1$, $j = j + 1$; if $j = n + 1$, end this step, and continue to the next step; otherwise, go back to Step 2.

**Step 5:** Apply the function of Definition 6 to $P_1$, and finally generate a scrambled matrix $P_2$, which is

$$P_2 = reshape(P_1, m, n).$$

## 4.3 Two Rounds of Diffusion

In this part, the matrix $P_2$ produced in Section 4.2 is diffused. In this section, the theory of the matrix semi-tensor product is used for diffusion, which is divided into two parts.

The diffusion matrix $A_4$ is generated by Section 3.2, and the matrix semi-tensor product operation between $A_4$ and $P_2$ is carried out. The matrix semi-tensor product formula is shown in Eq. (1), and the resulting matrix $P_3$ is calculated as follows:

$$P_3 = P_2 \ltimes A_4. \tag{12}$$

In Eq. (12), $P_3 \in M_{m\times n}$.

Then, $P_3$ is diffused in the next step, and the method of the matrix semi-tensor product is also applied. The

diffusion matrix $A_6$ generated in Section 3.2 is selected, and the matrix semi-tensor product operation between $A_6$ and $P_3$ is carried out. The diffusion matrix of this step is different in size from that of the first step, and the purpose of this step is to make the generated ciphertext image more disordered and more secure. The specific diffusion processes are as follows:

$$P_4 = P_3 \ltimes A_6 . \tag{13}$$

In Eq. (13), $P_4 \in M_{m\times n}$ .

## 4.4 Generating the Image

Because the pixel value of the image is an integer between 0 and 255, a step is required to generate the image $P_5$ so that each component of the $P_5$ is an integer between 0 and 255, as follows:

Use the functions in Definition 7 to retrieve $P_4$; if the plaintext image is $P \in M_{m\times n}$ and $m \geq 256$, then

$$\begin{aligned} [E_2, J_2] &= mod(P_4, 256) \\ P_5 &= floor(J_2) \end{aligned} . \tag{14}$$

If $m < 256$, then

$$\begin{aligned} [E_2, J_5] &= mod(P_4, m+1) \\ P_5 &= floor(J_5) \end{aligned} . \tag{15}$$

where *floor*() is rounding down function and finally generates the encrypted image $P_5$.

## 4.5 Encryption Image Coding

The encrypted Boolean networks are based on the following steps:

**Step 1:** Using the function of Definition 7, the $m/2+1$ line of $P_5$ is processed so that the $m/2+1$ line of $P_5$ is 0 or 1; this line is called $p_2$.

$$[E_3, J_3] = mod(p_1, 2), \quad p_1 = J_3 . \tag{16}$$

**Step 2:** The $m/2+2 \sim m$ line of $P_5$ is processed with a function of Definition 7 so that this behavior is an integer between 1 and 10, and the $m/2+2 \sim m$ line of $P_5$ is recorded as $p_2$:

$$[E_4, J_4] = mod(p_2, 10), \quad p_2 = J_4 + 1. \tag{17}$$

A new Boolean network can be formed by encoding $P_5$.

Users can accept encrypted images or encrypted Boolean networks based on their own needs. The encryption algorithm is complete.

## 4.6 Decryption Algorithm

A decryption algorithm is the inverse process of an encryption algorithm. From the inverse process of Section 4.5, the encrypted image $P_5$ can be obtained and then decrypted. The procedure is described as follows:

**Step 1:** Let $C_1 = J_2 - P_5$ be a secret key, according to Eq. (14) and Eq. (15). When $m \geq 256$,

$$P_4 = E_2 \times 256 + (C_1 + P_5), \tag{18}$$

and when $m < 256$,

$$P_4 = E_2 \times (m+1) + (C_1 + P_5) . \tag{19}$$

**Step 2:** From $P_4$ based on Step 1 and the inverse process of the matrix semi-tensor product proposed by Theorem 1, it can be obtained that:

$$P_3 = P_4 \times (A_6 \otimes I_4)^{-1}, \tag{20}$$

$$P_2 = P_3 \times (A_4 \otimes I_8)^{-1}. \tag{21}$$

**Step 3:** According to the inverse process of the matrix $P_2$ scrambling, the matrix $P$ can be obtained.

**Step 4：** The known matrix $P$ can decrypt the clear text image according to the inverse process of diffusion. The flow chart is as follows: Matrices $E_1$ and $J_1$ are generated from Eq. (11), and then

$$B = E_1 \times 256 + P. \tag{22}$$

According to the encoding rules of the Boolean network, $B$ can be decoded, and the original Boolean network equation can be obtained.

Fig. 1 shows the decryption flow chart of the encryption algorithm proposed in this paper.

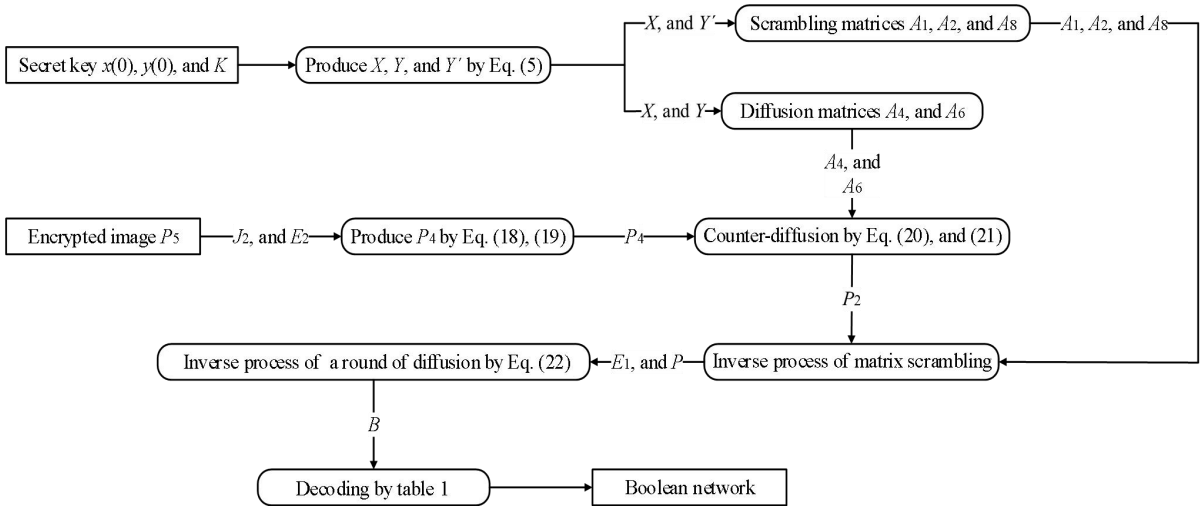


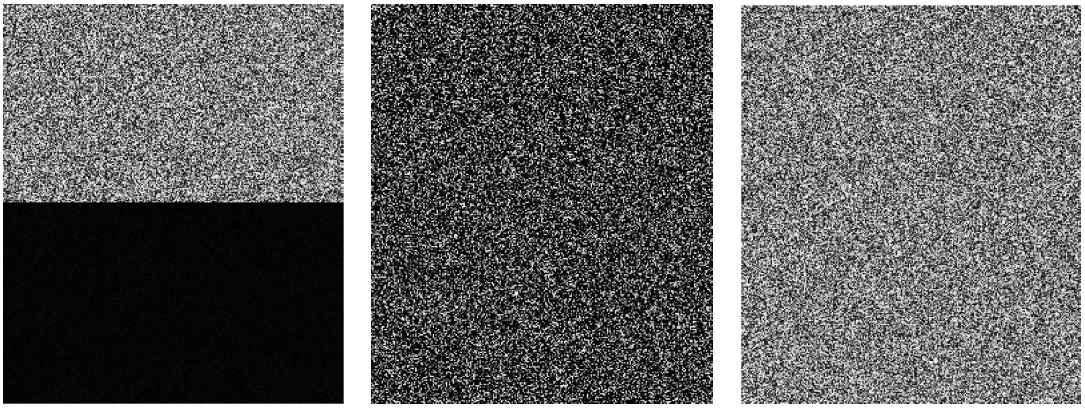

Fig. 1 Decryption flow chart

# 5 Simulation

The simulation in this section is divided into two parts. The first part generates a large Boolean network with 256 nodes randomly to demonstrate the practicability of the algorithm for Boolean network encryption. The second part uses some common images to test the performance of the algorithm mentioned in this paper.

## 5.1 Boolean Network Simulation

In this paper, we randomly generate a large Boolean network with 256 nodes, generating an image that we call a "Boolean" image. The size of the "Boolean" image is $P \in M_{300\times256}$. Then, the chaotic image encryption step is carried out. The user can choose to transmit a Boolean network in the form of an image and a Boolean network. Fig. 2(c) is processed by Section 4.5, and a new Boolean network can be formed. The experimental results are shown in Fig. 2.



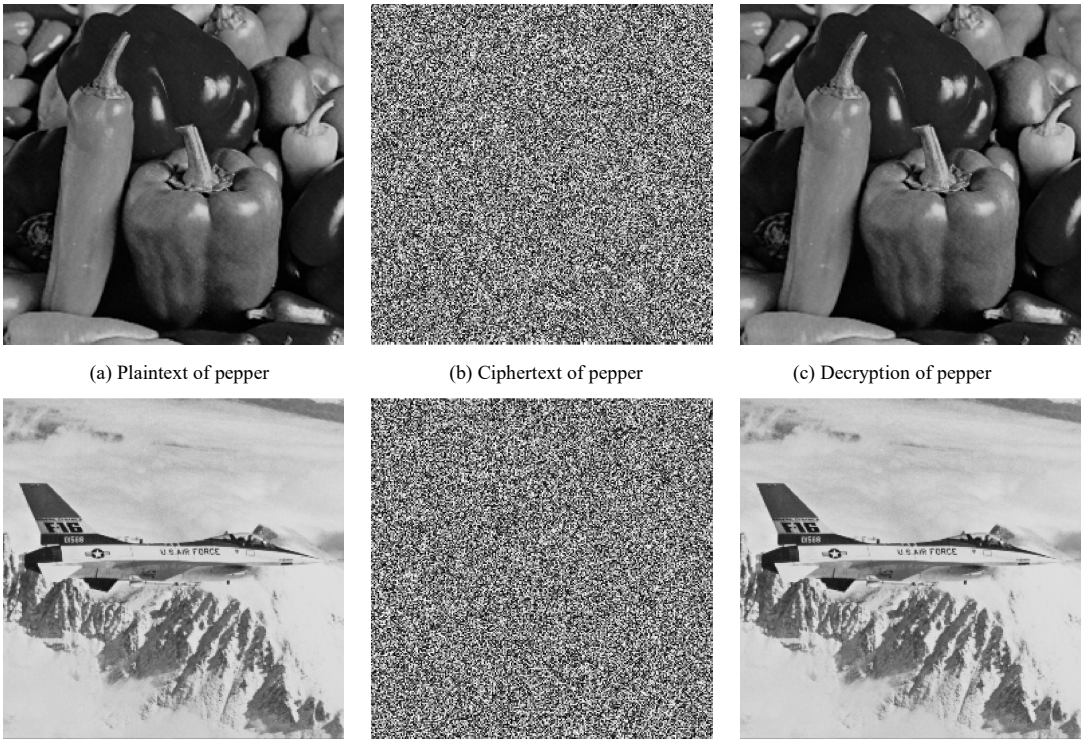
(a) Plaintext Boolean (b) Scrambled Boolean (c) Encrypted Boolean

Fig. 2 Encryption Boolean

In a further demonstration, take the Boolean network of Example 5 in Appendix A as an example and present the encryption process in the form of matrix. The result is shown in Appendix B.

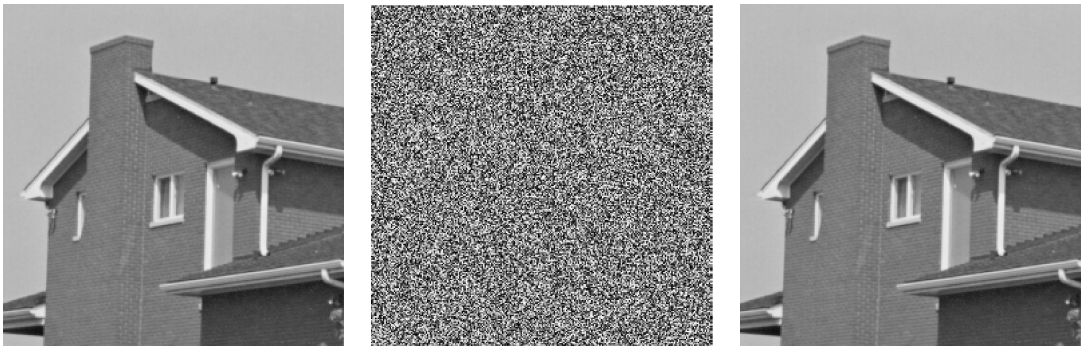
## 5.2 Common Image Simulation

This section describes the encryption of some common images, "pepper", "plane" and "house", which are $256 \times 256$ in size. The detailed encryption and decryption processes are shown in Fig. 3. The encryption method adopted in this paper is a symmetric encryption algorithm, and each part is reversible so that the decrypted image is the same as the original image.



(a) Plaintext of pepper (b) Ciphertext of pepper (c) Decryption of pepper

(d) Plaintext of plane (e) Ciphertext of plane (f) Decryption of plane



(g) Plaintext of house (h) Ciphertext of house (i) Decryption of house

Fig. 3 Encryption and decryption of grayscale images

# 6 Safety Analysis

A good chaotic encryption algorithm should be able to withstand a variety of common attacks. In this regard, a violent attack, a statistical attack, information entropy, the key space, sensitivity analysis, and the correlation between the adjacent pixels of the ciphertext are all considered. This section carries out some security analysis.

## 6.1 Secret Key Space Analysis

The algorithm secret key contains:

$$key = ((x(0), y(0)), E_2, J_2, E_1, F),$$

where the initial value of the chaotic system is $(x(0), y(0))$, the encoding of the Boolean network is $F$, $E_2$ and $J_2$ are generated by Eq. (14), and $E_1$ is generated by Eq. (11).

If only the initial value of the chaotic system is considered as the secret key, the accuracy of the initial value is $10^{-16}$, and the initial value is between $[-10,10]$, the size of the secret key space is approximately $10^{36}$, which is sufficient to resist various brute force attacks.

## 6.2 Differential Attack

By changing a pixel value of the plaintext image, an attacker can observe the pixel values of the two encrypted images and find the correlation between the plaintext image and the ciphertext image, thus cracking the encryption algorithm. To resist a differential attack, when we change a pixel value of the plaintext image, the encrypted image should exhibit a qualitative transformation. To examine the effects of changing a single pixel value of a plaintext image, we often use the following two methods of detection: the unified average changing intensity (UACI) and the number of pixels change rate (NPCR). In theory, the closer the values of the NPCR and UACI are to 99.6093% and 33.4635%, respectively, the better the algorithm. The exact formulas are as follows [37]:

$$NPCR = \frac{\sum_{i,j} D(i,j)}{W \times H} \times 100,$$

$$UACI = \frac{1}{W \times H}[\sum_{i,j} \frac{|c_1(i,j) - c_2(i,j)|}{255}] \times 100,$$

where $W$ and $H$ represent the width and height of the image, respectively. $c_1$ and $c_2$ are two cipher images in which the value of a pixel of the plaintext image changes. If $c_1(i,j) \neq c_2(i,j)$, $D(i,j)=1$, or $D(i,j)=0$.

"Boolean", "pepper", "plane" and "house" are randomly selected, and their pixel points are changed. The average values of the NPCR and UACI are obtained. The results of the NPCR and UACI are compared with Refs. [15, 21, 40, 41] and are shown in Table 2.

Through the comparison, it is found that the NPCR and UACI of the proposed algorithm are closer to the theoretical values, so the algorithm proposed in this paper has a better ability to resist a differential attack.

Table 2 The average NPCR and UACI and a comparison with other algorithms

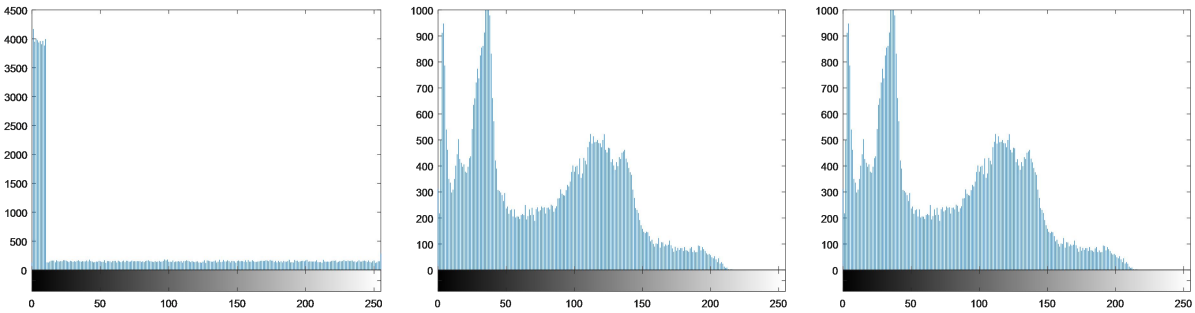
| Image | NPCR (%) | UACI (%) |
|---|---|---|
| Boolean | 99.5946 | 33.4318 |
| Pepper | 99.6002 | 33.4531 |
| Plane | 99.6216 | 33.3845 |
| House | 99.5819 | 33.3777 |
| Ref. [40] | 99.6080 | 33.4420 |
| Ref. [41] | 99.6200 | 33.4169 |
| Ref. [15] | 99.6521 | 33.3438 |
| Ref. [21] | 99.61548 | 31.1022 |

## 6.3 Statistical Analysis

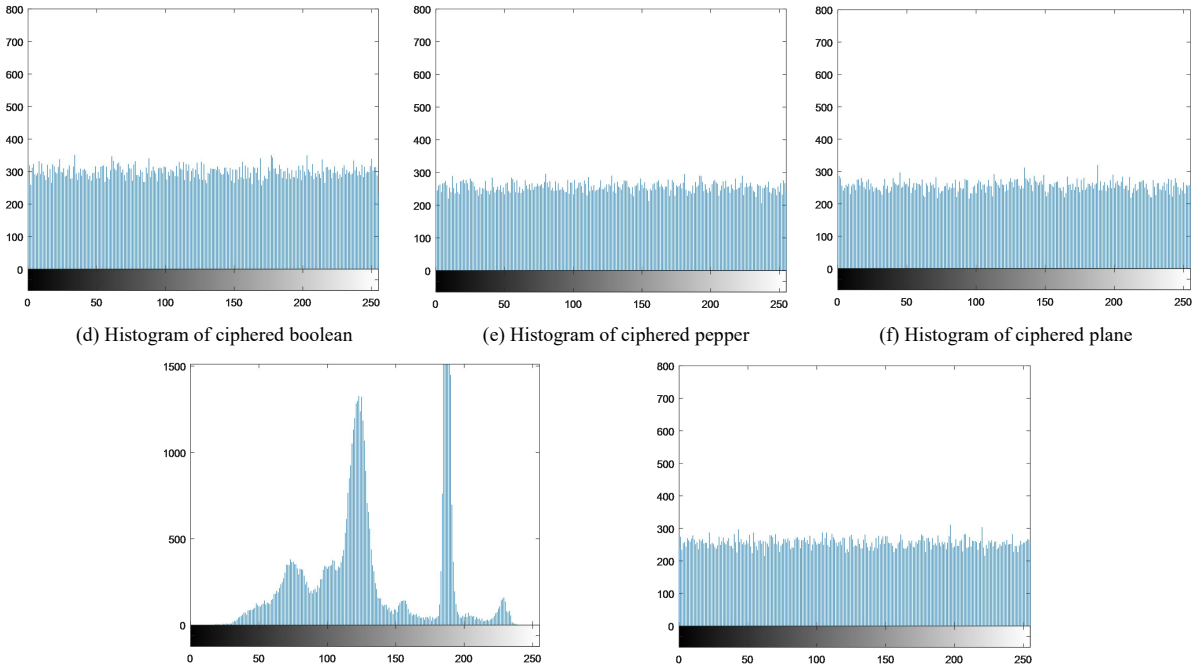
It is well known that statistical analysis of ciphertext images is very important, and a good algorithm should resist all kinds of statistical attacks. To prove that the algorithm has a good resistance to statistical analysis, kinds of tests are considered.

(1) Histogram analysis

Histogram analysis involves counting the number of pixels per pixel in an image. In general, the more evenly distributed the pixel values of the encrypted image are, the better the encryption effect [38]. Otherwise, an attacker can easily select ciphertext attacks by analyzing the statistical properties of ciphertext images. The pixel values of the "boolean", "pepper", "plane", "house" plaintext and ciphertext images are calculated as shown in Fig. 4.



(a) Histogram of boolean (b) Histogram of pepper (c) Histogram of plane



(d) Histogram of ciphered boolean (e) Histogram of ciphered pepper (f) Histogram of ciphered plane

(g) Histogram of house (h) Histogram of ciphered house

Fig. 4 Histograms of plaintext images and ciphered images

Fig. 4(a) corresponds to an image generated by a Boolean network. Because of the coding requirements and the random generation of the Boolean network, the pixel values are uniformly distributed between 0 ~ 10, and the pixel values are also uniformly distributed between 11 ~ 255. Through the encryption algorithm proposed in this paper, the pixel values of the ciphertext image are evenly distributed between 0 and 255. From the other graphs in Fig. 4, we can see that the histogram distribution of the original image is uneven, but with the encryption algorithm in this paper, the histogram distribution of the ciphertext image is uniform, so the algorithm proposed in this paper has a good ability to resist statistical analysis.

(2) Correlation analysis

The correlations between adjacent pixel points include horizontal, vertical, and diagonal adjacent pixel correlations. If the correlation between adjacent pixels is very high, encryption algorithm can easily be attacked by statistics [12]. Therefore, a good encryption algorithm should have as low a correlation as possible between the adjacent pixels of the encrypted image.

To test the correlation between adjacent pixels, the following procedure is carried out.

First, 10000 pairs of horizontally adjacent pixels are randomly selected from an image to calculate the correlation coefficients for each pair of pixels, as follows [35]:

$$r_{xy} = \frac{\operatorname{cov}(x, y)}{\sqrt{D(x)}\sqrt{D(y)}},$$

where

$$\operatorname{cov}(x, y) = \frac{1}{N}\sum_{i=1}^{N}(x_i - E(x))(y_i - E(y)), \quad D(x) = \frac{1}{N}\sum_{i+1}^{N}(x_i - E(x))^2, \quad E(x) = \frac{1}{N}\sum_{i=1}^{N}x_i.$$

In this paper, the plaintext pixels and ciphertext pixels of "boolean", "pepper", "plane" and "house" are chosen to test the correlations in the horizontal, vertical and diagonal directions, as shown in Fig. 5, Fig. 6, Fig. 7 and Fig. 8.

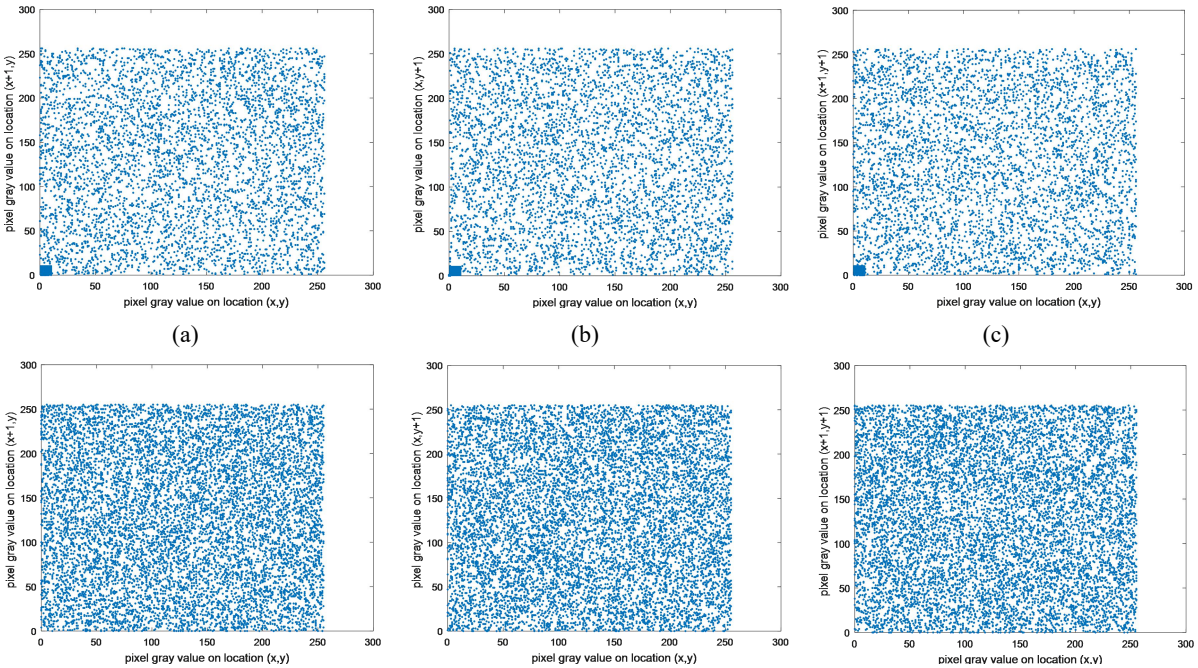



Fig. 5 Correlation coefficients of boolean: (a) horizontal correlation of boolean, (b) vertical correlation of boolean, (c) diagonal correlation of boolean, (d) horizontal correlation of ciphered boolean, (e) vertical correlation of ciphered boolean, and (f) diagonal correlation of ciphered boolean.

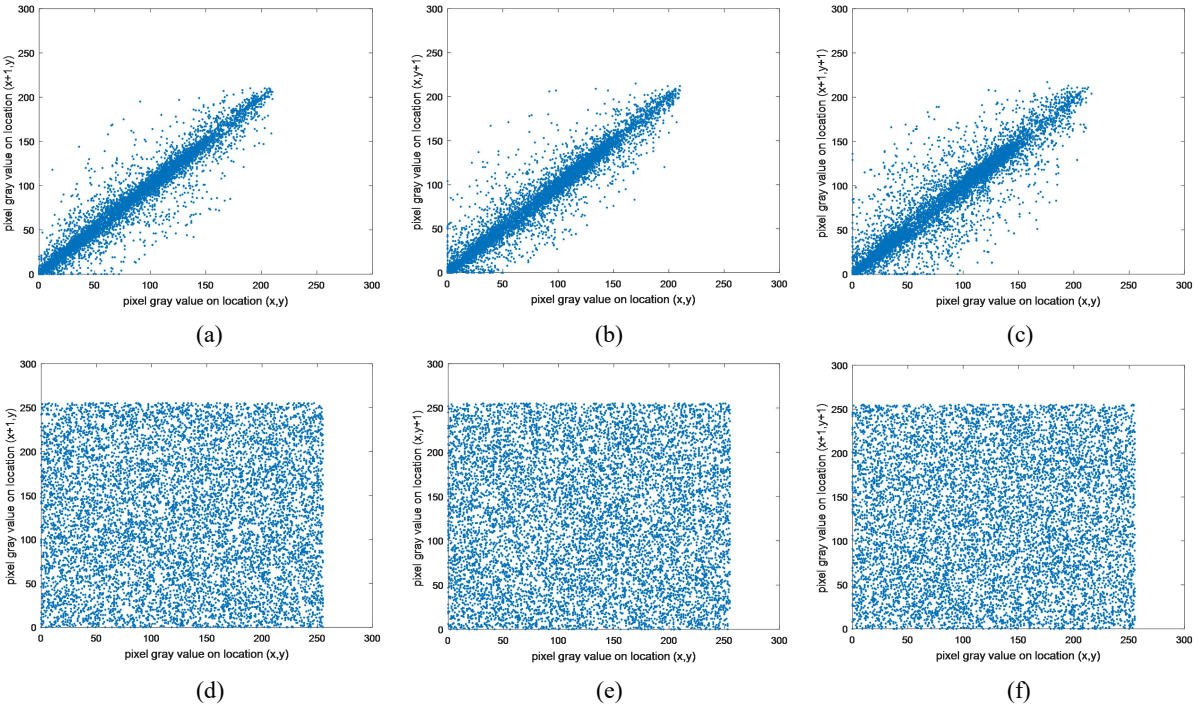


Fig. 6 Correlation coefficients of pepper; (a) horizontal correlation of pepper, (b) vertical correlation of pepper, (c) diagonal correlation of pepper, (d) horizontal correlation of ciphered pepper, (e) vertical correlation of ciphered pepper, and (f) diagonal correlation of ciphered pepper.

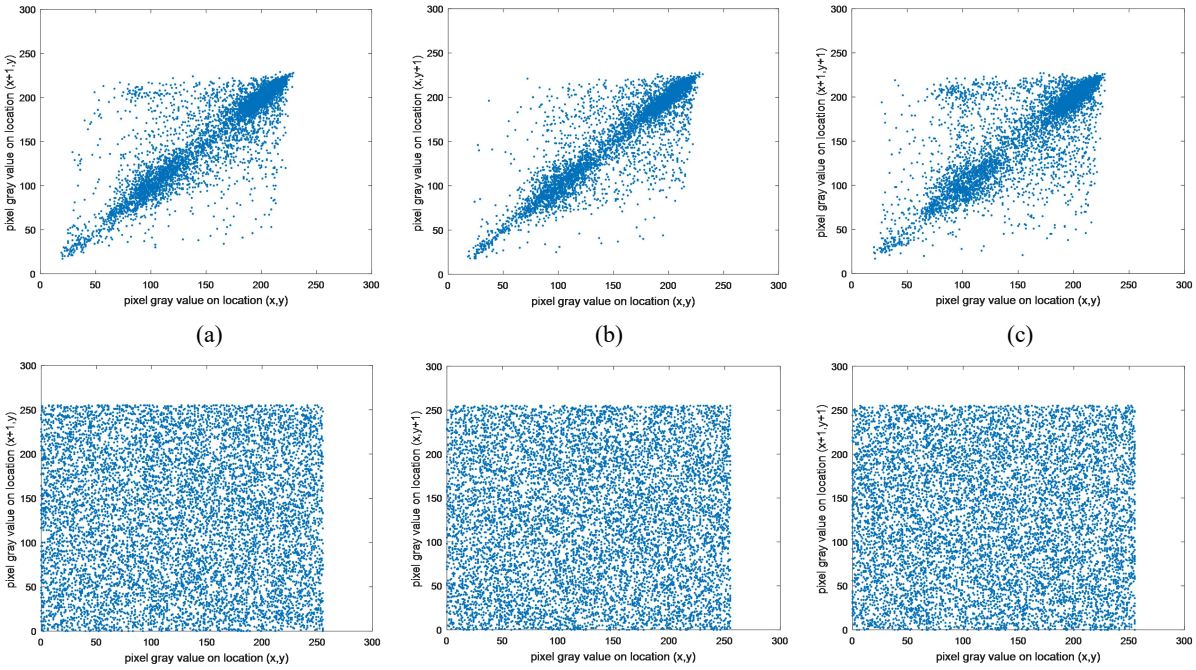



Fig. 7 Correlation coefficients of plane: (a) horizontal correlation of plane, (b) vertical correlation of plane, (c) diagonal correlation of plane, (d) horizontal correlation of ciphered plane, (e) vertical correlation of ciphered plane, and (f) diagonal correlation of ciphered plane.

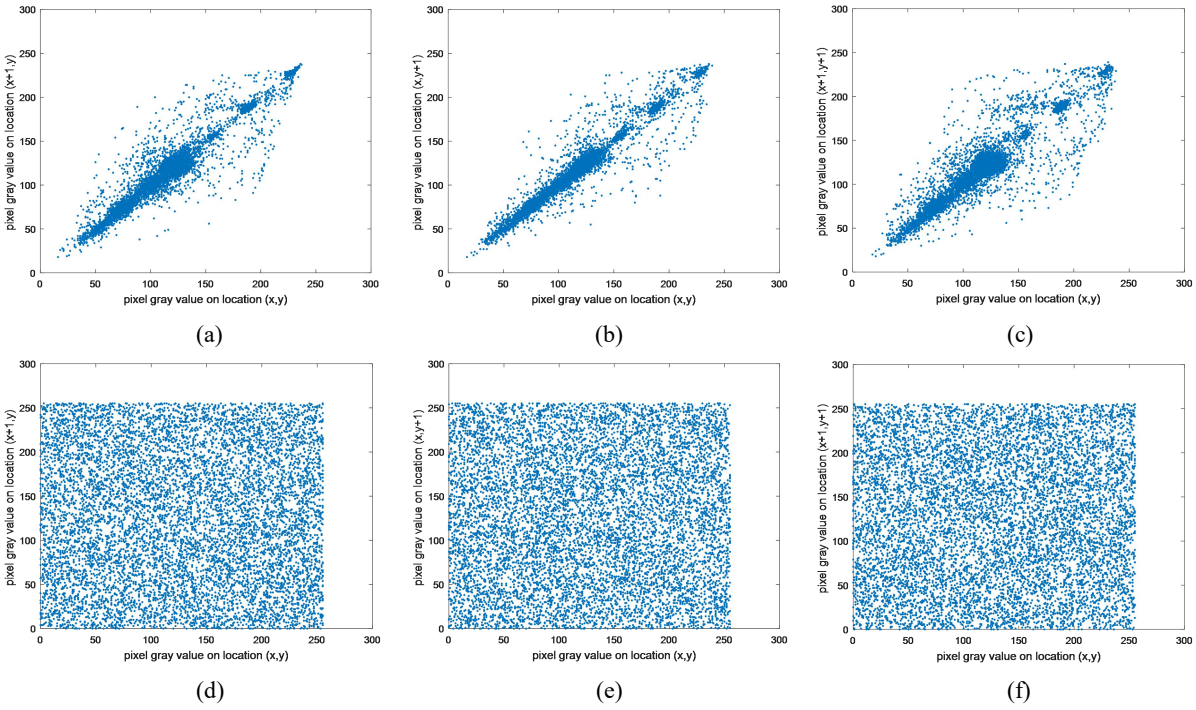


Fig. 8 Correlation coefficients of house; (a) horizontal correlation of house, (b) vertical correlation of house, (c) diagonal correlation of house, (d) horizontal correlation of ciphered house, (e) vertical correlation of ciphered house, and (f) diagonal correlation of ciphered house.

Furthermore, we use the data to more realistically describe the problem. Table 3 shows the correlation between adjacent pixels in the plaintext and ciphertext images. The study shows that the closer this value is to 0, the smaller the correlation between adjacent pixels and the better the resistance to a statistical analysis attack. This paper also compares the proposed method to some other algorithms, see Refs. [15, 21, 40, 41].

Table 3 Correlation coefficients of the images

| Image | Plain | | | Proposed | | |
|---|---|---|---|---|---|---|
| | Horizontal | Vertical | Diagonal | Horizontal | Vertical | Diagonal |
| Boolean | 0.5818 | 0.5812 | 0.5744 | 0.00033 | 0.00021 | 0.00297 |
| Pepper | 0.9675 | 0.9735 | 0.9453 | −0.00200 | 0.00044 | 0.00400 |
| Plane | 0.9186 | 0.9011 | 0.8529 | −0.00110 | −0.00290 | 0.00090 |
| House | 0.9779 | 0.9651 | 0.9480 | 0.00327 | −0.00071 | 0.00304 |

Table 4 Comparison of the correlation coefficients of the images

| Image | Boolean | Pepper | Plane | House | Ref. [40] | Ref. [41] | Ref. [15] | Ref. [21] |
|---|---|---|---|---|---|---|---|---|
| H | 0.00033 | 0.00200 | −0.00110 | 0.00327 | 0.00071 | 0.00560 | −0.07964 | −0.00908 |
| V | 0.00021 | 0.00044 | −0.00290 | −0.00071 | 0.00217 | 0.00370 | 0.01661 | −0.00052 |
| D | 0.00297 | 0.00400 | 0.00090 | 0.00304 | 0.01489 | 0.00320 | 0.00328 | −0.00594 |

The comparison results are shown in Table 4. We find that the correlation of the plaintext is very high, but with the encryption algorithm of this article, the correlation has decreased, close to 0. Thus, the proposed algorithm exhibits good performance in resisting statistical attacks.

## 6.4 Information Entropy Analysis

Information entropy is a statistical measure of disorder that reflects the randomness of information. To evaluate the randomness, the information entropy of cryptographic images is measured by the following formula [44]:

$$H(s)=\sum_{i=0}^{2^L-1}p(s_i)\log_2\frac{1}{p(s_i)}.$$

Where $p(s_i)$ represents the probability of an $s_i$ occurrence [14]. Theoretically, the closer the information entropy is to 8, the higher the degree of pixel confusion the less likely an information leak.

Table 5 shows the information entropy of "boolean", "pepper", "plane", "house" and ciphertext images and compares the entropies with the information entropy of Refs. [15, 21, 40, 41]. With the encryption algorithm in this paper, the information entropy is close to 8, so the algorithm proposed in this paper has good security characteristics.

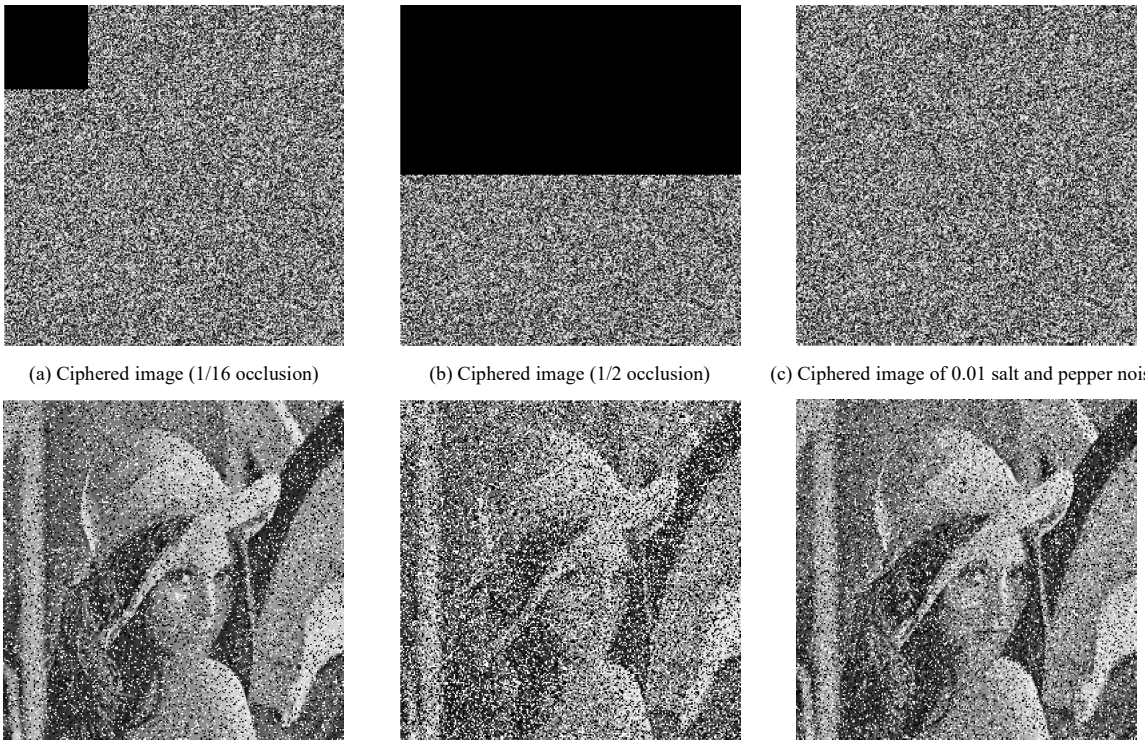
Table 5 Information entropies of the images

| Image | Plain | Proposed |
|---|---|---|
| Boolean | 6.5335 | 7.9971 |
| Pepper | 7.3800 | 7.9974 |
| Plane | 6.6992 | 7.9969 |
| House | 6.4971 | 7.9974 |
| Ref. [40] | -- | 7.9992 |
| Ref. [41] | -- | 7.9976 |
| Ref. [15] | -- | 7.9895 |
| Ref. [21] | -- | 7.9957 |

## 6.5 Robustness Detection

An image may lose some information in encryption, and there may also be some noise attacks. Therefore, it is very important for an algorithm to have good robustness [25].

In this paper, two forms of clipping attacks and noise attacks are used to test the robustness of the algorithm. Fig. 9 shows the results of a ciphertext clipping attack and a salt and pepper noise attack. For a comparison with the clipping attack of Refs. [5, 23, 45], we block half of the ciphertext and restore the graph as shown in Fig. 10. Through comparative experiments, it can be concluded that the algorithm proposed in this paper can lose half of the information, and some of the information of the original image can be obtained; the lost information is randomly distributed, so the algorithm proposed in this paper has good robustness.



(a) Ciphered image (1/16 occlusion) (b) Ciphered image (1/2 occlusion) (c) Ciphered image of 0.01 salt and pepper noise

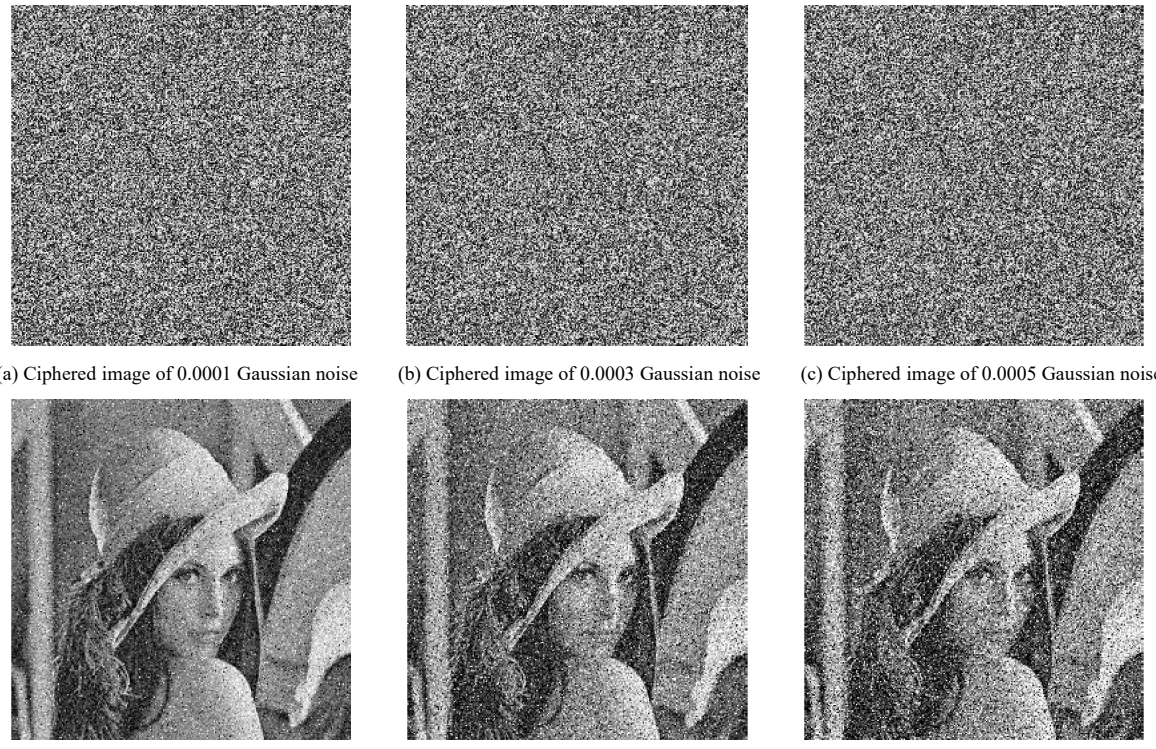
(d) Decrypted image of (a) (e) Decrypted image of (b) (f) Decrypted image of (c)

Fig. 9 Block attack and salt and pepper noise

(a) Clipping attacks in Ref. [5] (b) Clipping attacks in Ref. [23] (c) Clipping attacks in Ref. [45]

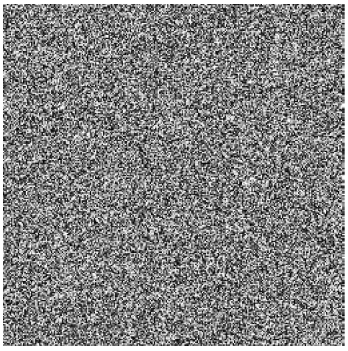
Fig. 10 Comparison of the clipping attacks (1/2)

Fig. 11 shows the results of a Gaussian attack noise and Poisson noise attack. Compared with the Gaussian noise attack in Ref. [5], the results shown in Fig. 12 show that the algorithm proposed in this paper has better clarity under the same Gaussian noise attack; thus, the algorithm proposed in this paper can better resist a noise attack.



(a) Ciphered image of 0.0001 Gaussian noise (b) Ciphered image of 0.0003 Gaussian noise (c) Ciphered image of 0.0005 Gaussian noise

(d) Decrypted image of (a) (e) Decrypted image of (b) (f) Decrypted image of (c)



(g) Ciphered image of Poisson noise

(h) Decrypted image of (g)

Fig. 11 Gaussian noise and Poisson noise

(d) Decrypted image of 0.0001 Gaussian noise

(d) Decrypted image of 0.0003 Gaussian noise

(f) Decrypted image of 0.0005 Gaussian noise

Fig. 12 Gaussian noise attack in Ref. [5]

## 6.6 Computational Complexity Analysis

To better illustrate the applicability of the proposed algorithm, we analyze the algorithm from two aspects: time complexity and spatial complexity.

The encryption algorithm in this paper consists of two parts: three random scramblings and matrix semi-tensor product diffusion. The principle of the encryption algorithm is based on matrix operations. Therefore, this paper proposes that the spatial complexity of the algorithm is related to the size of the plaintext images. The spatial complexity of the algorithm proposed in this paper can be regarded as $O(mn)$. Since the spatial complexity is not convenient for a comparison, we further consider the time complexity.

Table 6 Encryption time

| Image size | Proposed | Ref. [12] | Ref. [18] | Ref. [25] |
|---|---|---|---|---|
| 128×128 | 0.07740 | 0.0579 | 0.0323 | 0.0902 |
| 256×256 | 0.11794 | 0.2224 | 0.1440 | 0.3440 |
| 512×512 | 0.27444 | 0.9731 | 0.5510 | 1.3357 |
| 1024×1024 | 0.78902 | 3.9377 | 2.0864 | 5.3223 |

The proposed algorithm is implemented in Matlab 2017 on a computer with an Intel Core i5-7500 CPU, 8 GB

of memory and the Windows 10 operation system. We select image sizes of 128×128, 256×256, 512×512, and 1024×1024 for testing the encryption time, as shown in Table 6. The unit of time is seconds.

Compared with Refs. [12, 18, 25] as shown in Ref. [14], it can be seen that the running time of the proposed algorithm is very short, so the proposed encryption algorithm is very fast and has good applicability.

# 7 Conclusion

Based on chaotic image encryption, an algorithm for Boolean network encryption is proposed in this paper. A three-random-scrambling algorithm is used to scramble the plaintext image, and matrix semi-tensor product theory is used in the diffusion process. This encryption scheme can not only encrypt an image, but it can also encrypt a complex network. We also perform some security analyses, such as an information entropy analysis, a statistical analysis, and a differential attack analysis. The experimental results show that the matrix semi-tensor product can be used in the field of image encryption and produces good results. Because this method employs two matrices of different sizes, which increases the diversity of the diffusion, and the attacker does not know the size of the two matrices, the encryption scheme is more difficult to crack. Because of the good security characteristics of this method, the method is applied in Boolean network encryption to protect the privacy of genes to a large extent.

In this article, we only consider encryption for synchronously updating Boolean networks. In fact, asynchronously updating Boolean networks are closer to real systems. However, asynchronously updating Boolean networks have a complex update mechanism, which is difficult to operate. In future work, we wish to overcome this difficulty and find a way to encrypt an asynchronously updating Boolean network.

**Acknowledgment:** This research is supported by the National Natural Science Foundation of China (Nos. 61672124 and 61370145) and the Password Theory Project of the 13th Five-Year Plan National Cryptography Development Fund (No. MMJJ20170203).

# References

[1] M. Alawida, A. Samsudin, J. Sen Teh, R.S. Alkhawaldeh, A new hybrid digital chaotic system with applications in image encryption. Signal Process., 160 (2019), pp. 45-58.

[2] P. Anbalagan, R. Ramachandran, J.D. Cao, G. Rajchakit, C.P. Lim. Global Robust Synchronization of Fractional Order Complex Valued Neural Networks with Mixed Time Varying Delays and Impulses. Int. J. Control Autom. Syst., 17 (2) (2019), pp. 509-520.

[3] G. Ashkenasy, M.R. Ghadiri. Boolean logic functions of a synthetic peptide network. J. Am. Chem. Soc., 126 (36) (2004), pp. 11140-11141.

[4] A. Belazi, A.A. Abd El-Latif, S. Belghith. A novel image encryption scheme based on substitution-permutation network and chaos. Signal Process., 128 (2016), pp. 155-170.

[5] X.L. Chai, Y.R. Chen, L. Broyde. A novel chaos-based image encryption algorithm using DNA sequence operations. Opt. Lasers Eng., 88 (2017), pp. 197-213.

[6] G.R. Chen, Y.B. Mao, C.K. Chui. A symmetric image encryption scheme based on 3D chaotic cat maps. Chaos Solitons Fractals, 21 (3) (2004), pp. 749-761.

[7] J.X. Chen, Z.L. Zhu, H. Yu. A fast chaos-based symmetric image cryptosystem with an improved diffusion scheme. Optik, 125 (11) (2014), pp. 2472-2478.

[8] J.X. Chen, Z.L. Zhu, L.B. Zhang, Y.S. Zhang, B.Q. Yang. Exploiting self-adaptive permutation-diffusion and DNA random encoding for secure and efficient image encryption. Signal Process., 142 (2018), pp. 340-353.

[9] D.Z. Cheng, H.S. Qi. Controllability and observability of Boolean control networks. Automatica, 45 (7) (2009), pp. 1659-1667.

[10] D.Z. Cheng, H.S. Qi. Principle and range of possible applications of semi-tensor product of matrices. Journal of Systems Science and Mathematical Sciences. 32 (12) (2012), pp. 1488-1496.

[11] D.Z. Cheng, Y. Zhao. Semi-tensor product of matrices-A convenient new tool. Chin. Sci. Bull., 56 (32) (2011), pp. 2664-2674.

[12] A.V. Diaconu. Circular inter–intra pixels bit-level permutation and chaos-based image encryption. Inf. Sci., 355 (2016), pp. 314-327.

[13] Z.Y. Hua, Y.C. Zhou. Image encryption using 2D Logistic-adjusted-Sine map. Inf. Sci., 339 (2016), pp. 237-253.

[14] Z.Y. Hua, Y.C. Zhou, H.J. Huang. Cosine-transform-based chaotic system for image encryption. Inf. Sci., 480 (2019), pp. 403-419.

[15] X.L. Huang, G.D. Ye. An image encryption algorithm based on hyper-chaos and DNA sequence. Multimed. Tools Appl., 72 (1) (2014), pp. 57-70.

[16] R. Li, T.G. Chu. Complete Synchronization of Boolean Networks. IEEE Trans. Neural Netw. Learn. Syst., 23 (5) (2012), pp. 840-846.

[17] R. Li, M. Yang, T.G. Chu. State Feedback Stabilization for Boolean Control Networks. IEEE Trans. Autom. Control, 58 (7) (2013), pp. 1853-1857.

[18] X.F. Liao, S.Y. Lai, Q. Zhou. A novel image encryption algorithm based on self-adaptive wave transmission. Signal Process., 90 (9) (2010), pp. 2714-2722.

[19] C. Maharajan, R. Raja, J. D. Cao, G. Rajchakit. Fractional delay segments method on time-delayed recurrent neural networks with impulsive and stochastic effects: An exponential stability approach. Neurocomputing, 323 (2019), pp. 277-298.

[20] C. Maharajan, R. Raja, J. D. Cao, G. Rajchakit, A. Alsaedi. Novel results on passivity and exponential passivity for multiple discrete delayed neutral-type neural networks with leakage and distributed time-delays. Chaos Solitons Fractals, 115 (2018), pp. 268-282.

[21] O. Mannai, R. Bechikh, H. Hermassi, R. Rhouma, S. Belghith. A new image encryption scheme based on a simple first-order time-delay system with appropriate nonlinearity. Nonlinear Dyn., 82 (1-2) (2015), pp. 107-117.

[22] F. Ozkaynak. Brief review on application of nonlinear dynamics in image encryption. Nonlinear Dyn., 92 (2) (2018), pp. 305-313.

[23] Z. Parvin, H. Seyedarabi, M. Shamsi. A new secure and sensitive image encryption scheme based on new substitution with

chaotic function. Multimed. Tools Appl., 75 (17) (2016), pp. 10631-10648.

[24] L.M. Pecora, T.L. Carroll, G.A. Johnson, D.J. Mar, J.F. Heagy. Fundamentals of synchronization in chaotic systems, concepts, and applications. Chaos, 7 (4) (1997), pp. 520-543.

[25] P. Ping, F. Xu, Y.C. Mao, Z.J. Wang. Designing permutation–substitution image encryption networks with Henon map. Neurocomputing, 283 (2018), pp. 53-63.

[26] A. Pratap, R. Raja, J.D. Cao, G. Rajchakit, H.M. Fardoun. Stability and synchronization criteria for fractional order competitive neural networks with time delays: An asymptotic expansion of Mittag Leffler function. J. Frankl. Inst.-Eng. Appl. Math., 356 (4) (2019), pp. 2212-2239.

[27] P. Ramo, S. Kauffman, J. Kesseli, O. Yli-Harja. Measures for information propagation in Boolean networks. Physica D, 227 (1) (2007), pp. 100-104.

[28] R. Saravanakumar, G. Rajchakit, C.K. Ahn, H.R. Karimi. Exponential Stability, Passivity, and Dissipativity Analysis of Generalized Neural Networks With Mixed Time-Varying Delays. IEEE Trans. Syst. Man Cybern. -Syst., 42 (2) (2019), pp. 395-405.

[29] C.Y. Song, Y.L. Qiao, X.Z. Zhang. An image encryption scheme based on new spatiotemporal chaos. Optik, 124 (18) (2013), pp. 3329-3334.

[30] C. Sowmiya, R. Raja, Q.X. Zhu, G. Rajchakit. Further mean-square asymptotic stability of impulsive discrete-time stochastic BAM neural networks with Markovian jumping and multiple time-varying delays. J. Frankl. Inst.-Eng. Appl. Math., 356 (1) (2019), pp. 561-591.

[31] L.Z. Sun, B.D. Zheng, Y.M. Wei, C.J. Bu. Generalized inverses of tensors via a general product of tensors. Front. Math. China, 13 (4) (2018), pp. 893-911.

[32] J.H. Tang, X.B. Shu, Z.C. Li, Y.G. Jiang, Q. Tian. Social Anchor-Unit Graph Regularized Tensor Completion for Large-Scale Image Retagging. IEEE Trans. Pattern Anal. Mach. Intell., 41 (8) (2019), pp. 2027-2034.

[33] J.H. Tang, X.B. Shu, G.J. Qi, Z.C. Li, M. Wang, S.C. Yan, R. Jain. Tri-clustered tensor completion for social-aware image tag refinement. IEEE Trans. Pattern Anal. Mach. Intell., 39 (8) (2016), pp. 1662-1674.

[34] X. Wang, G.R. Chen. A chaotic system with only one stable equilibrium. Commun. Nonlinear Sci. Numer. Simul., 17 (3) (2012), pp. 1264-1272.

[35] X.Y. Wang, L. Feng, H.Y. Zhao. Fast image encryption algorithm based on parallel computing system. Inf. Sci., 486 (2019), pp. 340-358.

[36] X.Y. Wang, K. Guo. A new image alternate encryption algorithm based on chaotic map. Nonlinear Dyn., 76 (4) (2014), pp. 1943-1950.

[37] X.Y. Wang, L.T. Liu, Y.Q. Zhang. A novel chaotic block image encryption algorithm based on dynamic random growth technique. Opt. Lasers Eng., 66 (2015), pp. 10-18.

[38] X.Y. Wang, L. Teng, X. Qin. A novel colour image encryption algorithm based on chaos. Signal Process., 92 (4) (2012), pp. 1101-1108.

[39] X.Y. Wang, M.J. Wang. Hyperchaotic Lorenz system. Acta Phys. Sin., 56 (9) (2007), pp. 5136-5141.

[40] Y. Wang, K.W. Wong, X.F. Liao, G.R. Chen. A new chaos-based fast image encryption algorithm. Appl. Soft. Comput., 11(1) (2011), pp. 514-522.

[41] J.H. Wu, X.F. Liao, B. Yang. Image encryption using 2D Henon-Sine map and DNA approach. Signal Process., 153 (2018), pp. 11-23.

[42] Z.Y. Yan. Controlling hyperchaos in the new hyperchaotic Chen system. Appl. Math. Comput., 168 (2) (2005), pp. 1239-1250.

[43] J. Yao, J.E. Feng, M. Meng. On solutions of the matrix equation AX=B with respect to semi-tensor product. J. Frankl. Inst.-Eng. Appl. Math, 353 (5) (2016), pp. 1109-1131.

[44] Y.Q. Zhang, X.Y. Wang. A symmetric image encryption algorithm based on mixed linear-nonlinear coupled map lattice. Inf. Sci., 273 (2014), pp. 329-351.

[45] Y.S. Zhang, D. Xiao. Double optical image encryption using discrete Chirikov standard map and chaos-based fractional random transform. Opt. Lasers Eng., 51 (4) (2013), pp. 472-480.

# Appendix A

**Example 5:** Boolean networks with 16 gene nodes are described by:

$$
\begin{cases}
x_1(t+1) = x_1 \vee x_{13} \vee \neg x_3 \wedge \neg x_7 \leftrightarrow x_3 \wedge x_{12} \oplus x_4 \wedge x_5 \oplus x_{13} \wedge \neg x_9 \rightarrow x_{15} \\
x_2(t+1) = x_7 \leftrightarrow \neg x_{10} \wedge x_2 \rightarrow x_{14} \oplus x_3 \leftrightarrow x_5 \vee x_1 \leftrightarrow x_{16} \vee \neg x_{13} \wedge x_{12} \\
x_3(t+1) = x_3 \oplus \neg x_8 \leftrightarrow x_{13} \vee x_{11} \oplus x_{12} \wedge x_1 \wedge \neg x_2 \oplus \neg x_{14} \wedge x_{11} \wedge x_3 \rightarrow x_{10} \\
x_4(t+1) = x_{15} \leftrightarrow x_2 \wedge \neg x_6 \rightarrow x_{10} \wedge x_8 \wedge x_{13} \vee \neg x_9 \vee x_4 \vee \neg x_{12} \rightarrow x_3 \oplus \neg x_{11} \\
x_5(t+1) = x_4 \oplus x_3 \wedge \neg x_1 \vee x_2 \vee \neg x_{10} \wedge x_4 \vee x_1 \vee x_{16} \rightarrow \neg x_{13} \wedge x_{12} \vee x_5 \oplus \neg x_9 \\
x_6(t+1) = \neg x_{14} \leftrightarrow x_{15} \wedge \neg x_4 \leftrightarrow \neg x_{11} \oplus x_{16} \vee x_6 \rightarrow x_2 \oplus x_3 \vee \neg x_{12} \wedge x_1 \wedge \neg x_2 \vee x_8 \\
x_7(t+1) = \neg x_2 \wedge x_{16} \vee x_1 \oplus x_9 \wedge x_{14} \leftrightarrow \neg x_{13} \oplus \neg x_4 \rightarrow x_{15} \leftrightarrow x_{12} \oplus x_2 \vee \neg x_9 \rightarrow x_3 \\
x_8(t+1) = x_3 \rightarrow x_{15} \wedge x_3 \rightarrow x_8 \wedge \neg x_7 \leftrightarrow x_6 \vee \neg x_5 \wedge \neg x_{14} \rightarrow \neg x_{12} \wedge x_8 \\
x_9(t+1) = \neg x_5 \wedge \neg x_4 \vee x_{11} \wedge x_{13} \oplus x_{14} \rightarrow \neg x_7 \leftrightarrow x_2 \vee x_3 \vee x_6 \wedge x_{14} \oplus \neg x_4 \vee x_{13} \\
x_{10}(t+1) = x_{11} \oplus \neg x_1 \vee x_{13} \rightarrow x_{15} \wedge x_8 \wedge x_{14} \vee \neg x_3 \oplus x_{13} \leftrightarrow x_{15} \\
x_{11}(t+1) = \neg x_9 \leftrightarrow x_7 \wedge x_8 \rightarrow x_{16} \vee x_{10} \wedge x_6 \oplus x_8 \vee \neg x_3 \vee x_{13} \vee \neg x_{12} \rightarrow x_8 \wedge x_1 \\
x_{12}(t+1) = x_2 \oplus \neg x_{13} \rightarrow x_{11} \rightarrow x_1 \rightarrow x_9 \oplus x_{12} \leftrightarrow x_2 \vee x_4 \wedge x_{13} \wedge \neg x_3 \oplus x_{12} \vee \neg x_{15} \\
x_{13}(t+1) = x_1 \wedge x_7 \rightarrow x_3 \vee x_9 \rightarrow x_2 \vee \neg x_1 \leftrightarrow x_{15} \wedge x_{16} \wedge \neg x_6 \oplus \neg x_{11} \oplus x_2 \wedge x_4 \\
x_{14}(t+1) = \neg x_1 \rightarrow x_8 \oplus \neg x_3 \rightarrow \neg x_2 \wedge x_7 \wedge x_{16} \vee x_{12} \oplus \neg x_{10} \leftrightarrow \neg x_9 \vee x_{12} \leftrightarrow x_1 \leftrightarrow x_{13} \\
x_{15}(t+1) = \neg x_{15} \vee \neg x_1 \oplus \neg x_{14} \oplus x_8 \vee x_{16} \vee \neg x_4 \oplus \neg x_{15} \rightarrow \neg x_{11} \leftrightarrow \neg x_2 \leftrightarrow x_5 \rightarrow \neg x_{13} \\
x_{16}(t+1) = x_2 \vee x_8 \rightarrow \neg x_4 \vee \neg x_3 \rightarrow x_6 \oplus x_{14} \oplus x_5 \leftrightarrow \neg x_{15} \vee x_4 \wedge x_{13} \wedge x_1 \oplus x_{11}
\end{cases}.
$$

Due to article length constraints, the right side of the equal sign $x(t)$ is abbreviated as $x$. According to the coding rules, Boolean network is converted to Boolean matrix form as follows:

$$
B = \begin{bmatrix}
1 & 7 & 3 & 15 & 4 & 14 & 2 & 3 & 5 & 11 & 9 & 2 & 1 & 1 & 15 & 2 \\
13 & 10 & 8 & 2 & 3 & 15 & 16 & 15 & 4 & 1 & 7 & 13 & 7 & 8 & 1 & 8 \\
3 & 2 & 13 & 6 & 1 & 4 & 1 & 3 & 11 & 13 & 8 & 11 & 3 & 3 & 14 & 4 \\
7 & 14 & 11 & 10 & 2 & 11 & 9 & 8 & 13 & 15 & 16 & 1 & 9 & 2 & 8 & 3 \\
3 & 3 & 12 & 8 & 10 & 16 & 14 & 7 & 14 & 8 & 10 & 9 & 2 & 7 & 16 & 6 \\
12 & 5 & 1 & 13 & 4 & 6 & 13 & 6 & 7 & 14 & 6 & 12 & 1 & 16 & 4 & 14 \\
4 & 1 & 2 & 9 & 1 & 2 & 4 & 5 & 2 & 3 & 8 & 2 & 15 & 12 & 15 & 5 \\
5 & 16 & 14 & 4 & 16 & 3 & 15 & 14 & 3 & 13 & 3 & 4 & 16 & 10 & 11 & 15 \\
13 & 13 & 11 & 12 & 13 & 12 & 12 & 12 & 6 & 15 & 13 & 13 & 6 & 9 & 2 & 4 \\
9 & 12 & 3 & 3 & 12 & 1 & 2 & 8 & 14 & 0 & 12 & 3 & 11 & 12 & 5 & 13 \\
15 & 0 & 10 & 11 & 5 & 2 & 9 & 0 & 4 & 0 & 8 & 12 & 2 & 1 & 13 & 1 \\
0 & 0 & 0 & 0 & 9 & 8 & 3 & 0 & 13 & 0 & 1 & 15 & 4 & 13 & 0 & 11 \\
1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\
2 & 10 & 8 & 5 & 3 & 5 & 1 & 4 & 6 & 8 & 5 & 8 & 1 & 4 & 7 & 2 \\
7 & 1 & 5 & 6 & 6 & 6 & 2 & 1 & 2 & 2 & 1 & 4 & 4 & 8 & 8 & 9 \\
6 & 4 & 2 & 4 & 2 & 10 & 3 & 4 & 1 & 4 & 4 & 4 & 2 & 9 & 3 & 7 \\
5 & 3 & 3 & 1 & 7 & 3 & 1 & 6 & 3 & 1 & 2 & 4 & 4 & 1 & 2 & 4 \\
1 & 5 & 1 & 1 & 1 & 2 & 10 & 5 & 9 & 1 & 1 & 3 & 7 & 1 & 7 & 3 \\
3 & 2 & 6 & 7 & 2 & 4 & 8 & 7 & 5 & 7 & 3 & 5 & 5 & 2 & 8 & 3 \\
1 & 5 & 8 & 2 & 1 & 3 & 4 & 6 & 2 & 3 & 7 & 2 & 1 & 8 & 9 & 10 \\
3 & 7 & 1 & 7 & 9 & 7 & 5 & 9 & 2 & 5 & 2 & 1 & 6 & 10 & 10 & 2 \\
6 & 1 & 1 & 4 & 1 & 1 & 3 & 1 & 1 & 0 & 7 & 6 & 8 & 2 & 5 & 1 \\
4 & 0 & 4 & 8 & 2 & 6 & 7 & 0 & 8 & 0 & 4 & 3 & 3 & 5 & 9 & 1 \\
0 & 0 & 0 & 0 & 8 & 2 & 4 & 0 & 2 & 0 & 1 & 7 & 1 & 5 & 0 & 3
\end{bmatrix}.
$$

# Appendix B

For a further demonstration, take the Boolean network of Example 5 in Appendix A as an example. The encryption process in matrix form is:

$$B_1 = \begin{bmatrix}
9 & 1 & 0 & 9 & 6 & 3 & 8 & 4 & 9 & 1 & 8 & 0 & 4 & 7 & 13 & 0 \\
8 & 5 & 15 & 13 & 7 & 12 & 15 & 10 & 13 & 10 & 6 & 2 & 12 & 14 & 15 & 14 \\
14 & 3 & 10 & 6 & 14 & 8 & 6 & 0 & 15 & 6 & 3 & 5 & 1 & 13 & 1 & 6 \\
2 & 7 & 3 & 10 & 7 & 5 & 6 & 7 & 7 & 12 & 15 & 6 & 10 & 13 & 4 & 6 \\
4 & 0 & 5 & 12 & 16 & 11 & 0 & 2 & 8 & 2 & 3 & 2 & 9 & 16 & 1 & 12 \\
15 & 9 & 7 & 5 & 16 & 10 & 3 & 0 & 1 & 0 & 5 & 14 & 10 & 13 & 2 & 5 \\
8 & 4 & 10 & 11 & 6 & 0 & 0 & 9 & 11 & 8 & 3 & 9 & 7 & 12 & 14 & 7 \\
9 & 4 & 13 & 9 & 8 & 0 & 3 & 13 & 14 & 6 & 6 & 14 & 16 & 12 & 9 & 3 \\
11 & 0 & 8 & 14 & 2 & 2 & 12 & 14 & 15 & 2 & 0 & 12 & 16 & 4 & 1 & 6 \\
0 & 8 & 14 & 13 & 11 & 15 & 5 & 4 & 5 & 13 & 11 & 7 & 5 & 13 & 9 & 5 \\
15 & 16 & 0 & 4 & 0 & 10 & 1 & 6 & 1 & 0 & 4 & 8 & 4 & 1 & 1 & 1 \\
2 & 13 & 2 & 12 & 15 & 15 & 11 & 11 & 5 & 0 & 11 & 1 & 5 & 8 & 2 & 7 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\
8 & 7 & 4 & 3 & 5 & 1 & 4 & 1 & 8 & 3 & 2 & 1 & 7 & 6 & 2 & 1 \\
7 & 10 & 4 & 6 & 4 & 9 & 3 & 1 & 6 & 3 & 8 & 4 & 5 & 5 & 2 & 6 \\
4 & 2 & 4 & 6 & 7 & 2 & 4 & 9 & 7 & 4 & 7 & 1 & 2 & 3 & 9 & 5 \\
5 & 6 & 3 & 1 & 10 & 8 & 3 & 7 & 5 & 6 & 4 & 6 & 7 & 8 & 3 & 2 \\
4 & 7 & 6 & 2 & 7 & 1 & 7 & 1 & 2 & 3 & 8 & 7 & 6 & 8 & 3 & 3 \\
7 & 4 & 2 & 6 & 10 & 6 & 1 & 7 & 5 & 6 & 8 & 10 & 3 & 4 & 7 & 9 \\
5 & 1 & 5 & 7 & 5 & 4 & 6 & 8 & 6 & 8 & 7 & 6 & 4 & 7 & 6 & 7 \\
3 & 1 & 3 & 10 & 6 & 7 & 2 & 6 & 7 & 8 & 5 & 10 & 7 & 4 & 1 & 5 \\
6 & 3 & 7 & 5 & 2 & 1 & 9 & 6 & 9 & 3 & 5 & 6 & 6 & 2 & 10 & 10 \\
1 & 5 & 1 & 1 & 3 & 7 & 8 & 9 & 6 & 7 & 2 & 7 & 4 & 10 & 7 & 10 \\
1 & 10 & 8 & 6 & 8 & 7 & 4 & 7 & 1 & 1 & 5 & 5 & 7 & 4 & 2 & 1
\end{bmatrix}.$$

The cryptographic Boolean network generated by the encoding operation is:

$$\begin{cases}
x_1(t+1) = x_9 \oplus \neg x_8 \vee \neg x_{14} \to x_2 \leftrightarrow x_4 \to x_{15} \vee \neg x_8 \leftrightarrow x_9 \oplus x_{11} \wedge \neg\neg x_0 \wedge x_{15} \wedge x_2 \\
x_2(t+1) = x_1 \vee \neg x_5 \leftrightarrow \neg x_3 \vee x_7 \wedge \neg\neg x_0 \vee \neg\neg x_9 \to x_4 \wedge x_4 \wedge x_0 \oplus x_8 \leftrightarrow x_{16} \leftrightarrow \neg x_{13} \\
x_3(t+1) = x_0 \to x_{15} \to x_{10} \to x_3 \oplus x_5 \wedge \neg x_7 \vee x_{10} \leftrightarrow x_{13} \oplus x_8 \vee \neg x_{14} \wedge x_0 \oplus \neg x_2 \\
x_4(t+1) = x_9 \oplus x_{13} \wedge \neg x_6 \wedge \neg x_{10} \wedge x_{12} \vee x_5 \wedge \neg x_{11} \vee \neg x_9 \leftrightarrow \neg x_{14} \leftrightarrow x_{13} \wedge x_4 \wedge \neg x_{12} \\
x_5(t+1) = x_6 \leftrightarrow x_7 \to x_{14} \vee \neg x_7 \leftrightarrow \neg x_{16} \vee \neg x_{16} \leftrightarrow \neg x_6 \leftrightarrow x_8 \wedge \neg x_2 \vee x_{11} \oplus x_0 \oplus \neg x_{15} \\
x_6(t+1) = x_3 \wedge x_{12} \to \neg x_8 \vee x_5 \oplus \neg x_{11} \wedge x_{10} \wedge \neg x_0 \to x_0 \vee \neg x_2 \wedge x_{15} \vee \neg x_{10} \vee \neg x_{15} \\
x_7(t+1) = x_8 \to x_{15} \oplus x_6 \to x_6 \oplus x_0 \vee \neg x_3 \wedge x_0 \wedge \neg x_3 \vee x_{12} \to \neg x_5 \oplus \neg x_1 \to x_{11} \\
x_8(t+1) = \neg x_4 \wedge x_{10} \wedge x_0 \to \neg x_7 \vee \neg x_2 \wedge x_0 \vee \neg x_9 \oplus \neg x_{13} \wedge \neg\neg x_{14} \wedge \neg x_4 \to \neg x_6 \vee \neg x_{11} \\
x_9(t+1) = x_9 \oplus \neg x_{13} \wedge \neg x_{15} \vee \neg x_7 \leftrightarrow x_8 \vee x_1 \leftrightarrow x_{11} \wedge \neg x_{14} \vee \neg x_{15} \to \neg x_5 \wedge \neg x_1 \wedge x_5 \\
x_{10}(t+1) = x_1 \oplus x_{10} \oplus x_6 \to x_{12} \wedge \neg x_2 \oplus x_0 \wedge \neg x_8 \oplus \neg x_6 \oplus \neg x_2 \oplus x_{13} \vee \neg x_0 \wedge x_0 \\
x_{11}(t+1) = \neg x_8 \vee x_6 \oplus \neg x_3 \vee \neg x_{15} \to x_3 \oplus \neg x_5 \oplus \neg x_3 \vee \neg x_6 \leftrightarrow x_0 \leftrightarrow x_{11} \vee x_4 \leftrightarrow x_{11} \\
x_{12}(t+1) = x_0 \wedge x_2 \to x_5 \wedge x_6 \wedge \neg x_2 \vee \neg x_{14} \leftrightarrow \neg x_9 \wedge \neg x_{14} \leftrightarrow \neg x_{12} \wedge \neg x_7 \vee \neg x_8 \leftrightarrow x_4 \\
x_{13}(t+1) = \neg x_4 \vee \neg x_{12} \leftrightarrow x_1 \vee x_{10} \vee \neg x_9 \wedge \neg x_{10} \oplus x_7 \to x_{16} \vee \neg x_{16} \wedge \neg x_5 \to x_4 \vee \neg x_5 \\
x_{14}(t+1) = x_7 \wedge \neg x_{14} \leftrightarrow x_{13} \oplus x_{13} \oplus \neg x_{16} \oplus \neg x_{13} \to x_{12} \vee \neg x_{12} \to x_4 \vee x_{13} \leftrightarrow \neg x_1 \to x_8 \\
x_{15}(t+1) = x_{13} \vee x_{15} \vee x_1 \to \neg x_4 \oplus x_1 \oplus x_2 \vee \neg x_{14} \wedge \neg x_9 \wedge x_1 \leftrightarrow \neg x_9 \vee \neg x_1 \vee x_2 \\
x_{16}(t+1) = \neg x_0 \wedge x_{14} \wedge \neg x_6 \leftrightarrow x_6 \vee x_{12} \oplus x_5 \to \neg x_7 \vee \neg x_3 \leftrightarrow x_6 \leftrightarrow \neg x_5 \leftrightarrow \neg x_1 \wedge x_7
\end{cases},$$

where $x_0 = 0$ and $\neg x_0 = 1$. Because of the article length limit, the right side of the equal sign $x(t)$ is

abbreviated as $x$.

$x_1, x_2, \cdots, x_n$ are the Boolean network nodes, replacing the element of 0 in the system of equations with $x_0$. Each logical symbol is represented as follows: $\wedge\neg$ is $P \wedge \neg Q$; $\vee\neg$ is $P \vee \neg Q$; $\oplus\neg$ is $P \oplus \neg Q$; $\rightarrow\neg$ is $P \rightarrow \neg Q$; and $\leftrightarrow\neg$ is $P \leftrightarrow \neg Q$.