Available online at www.sciencedirect.com
SCIENCE DIRECT
Physics Letters A 356 (2006) 333–338
PHYSICS LETTERS A
www.elsevier.com/locate/pla

Cryptography based on delayed chaotic neural networks ☆

Wenwu Yu, Jinde Cao *

Department of Mathematics, Southeast University, Nanjing 210096, China

Received 1 February 2006; received in revised form 10 March 2006; accepted 28 March 2006

Available online 17 April 2006

Communicated by A.R. Bishop

Abstract

In this Letter, a novel approach of encryption based on chaotic Hopfield neural networks with time varying delay is proposed. We use the chaotic neural network to generate binary sequences which will be used for masking plaintext. The plaintext is masked by switching of chaotic neural network maps and permutation of generated binary sequences. Simulation results were given to show the feasibility and effectiveness in the proposed scheme of this Letter. As a result, chaotic cryptography becomes more practical in the secure transmission of large multi-media files over public data communication network.

Keywords: Synchronization; Time varying delay; Neural network; Chaos; Encryption; Chaotic cryptosystem

1. Introduction

Good private communication methods has aroused the interest of many researchers. A lot of encryption research works are based on using chaotic synchronization in private communications. Chaotic communication systems have attracted more and more attentions due to the broadband power spectrum of chaotic signals, high speed of information transmission, and tolerance to sufficiently high levels of noise. Besides, many chaotic communication schemes are simply realized and demonstrate a rich variety of different oscillating regimes. Networks of chaotic coupled maps are considered as string and language generators in [6]. The secret key provides the network parameters, such as the coupling strengths. Extracting information masked by the chaotic signal of a time-delay system has been considered in [7]. A method in which a chaotic signal is used to mask a message securely is stated in [12]. However, we use a more complicated Hopfield neural networks with switching chaotic maps and time varying delay.

Practically all known communication schemes using chaotic signals are based on synchronization of chaotic systems [8–11, 14–18]. Different approaches for the transmission of information signals using chaotic dynamics have been proposed, for example, chaotic masking, chaotic modulation, nonlinear mixing, chaotic switching, and others. However, not all chaotic communication schemes are as secure as expected. For example, some communication schemes using low-dimensional chaotic signals can be unmasked by the dynamical reconstruction of the chaotic system from the time series or by using suitable return maps.

The purpose of a cryptography system, which is often called a cryptosystem, is to transmit confidential messages secretively. However, in recent years the actual and potential applications of cryptography have expanded to include many other areas such as remote log-in protocols, shared control schemes, democratic voting schemes, authenticated distributed computing, electronic money, distributed management of data bases, and so on. In general, synchronous chaotic ordinary differential equations are easy to be implemented. The cryptosystems are easy to set up with chaotic systems. The chaotic signals are randomlike signals and considered as pseudorandom. The secret keys generated from these chaotic signals can be considered as pseudorandom signals.

In this Letter, cryptography based on chaotic Hopfield neural network [13] with time varying delay is proposed. As our first

☆ This work was jointly supported by the National Natural Science Foundation of China under Grant No. 60574043, and the 973 Program of China under Grant 2003CB317004.
* Corresponding author.
E-mail address: jdcao@seu.edu.cn (J. Cao).

0375-9601/$ – see front matter
doi:10.1016/j.physleta.2006.03.069

contribution, we use the Hopfield neural network to generate binary sequences. As our second contribution, A binary value in the binary sequence is supposed to choose the chaotic map which is used to generate binary sequences in the next step. As a result, this value is considered as a random switching function for choosing chaotic maps. As our third contribution, plaintext is masked by permutation and calculation of generated binary sequences. An algorithm is designed for encryption. There are some advantages using this cryptosystem. Firstly, note that the generated binary sequences are a part of trajectory of chaotic neural network. Secondly, the trajectory we choose is random for the switching value of generated binary sequences. Thirdly, to the best of our knowledge, it is difficult to synchronize the unknown chaotic neural networks unless you know the neural network clearly. Fourthly, the secret keys of initial value and the delay are both functions. We will discuss security analysis to see that even small perturbations of key parameters may result in the wrong decryption. Fifthly, we will see the distribution of ciphertext is flat and the information entropy is high, which means less information of the plaintext is revealed. Also, the algorithm is fast and efficient. So it is more secure to use the approach proposed in this Letter.

2. Chaotic neural networks

In this section, we consider the following Hopfield neural networks [13] which exhibit chaotic phenomenon:

$$\dot{x}(t) = -Cx(t) + Af\big(x(t)\big) + Bf\big(x\big(t-\tau(t)\big)\big) + I, \tag{1}$$

or

$$\dot{x}_i(t) = -c_i x_i(t) + \sum_{j=1}^{n} a_{ij} f_j\big(x_j(t)\big) + \sum_{j=1}^{n} b_{ij} f_j\big(x_j\big(t-\tau_{ij}(t)\big)\big) + I_i, \quad i = 1, 2, \ldots, n, \tag{2}$$

where n denotes the number of units in a neural network, $x(t) = (x_1(t), x_2(t), \ldots, x_n(t))^T \in R^n$ is the state vector associated with the neurons, $I = (I_1, I_2, \ldots, I_n)^T \in R^n$ is external input vector, $f(x(t)) = (f_1(x_1(t)), f_2(x_2(t)), \ldots, f_n(x_n(t)))^T \in R^n$ corresponds to the activation functions of neurons, $\tau(t) = \tau_{ij}(t)$ $(i, j = 1, 2, \ldots, n)$ are the time delays, the initial conditions of (1) are given by $x_i(t) = \phi_i(t) \in \mathcal{C}([-r, 0], R)$ with $r = \max_{1\leqslant i,j\leqslant n, t\in R}\{\tau_{ij}(t)\}$, where $\mathcal{C}([-r, 0], R)$ denotes the set of all continuous functions from $[-r, 0]$ to R. $C = \operatorname{diag}(c_1, c_2, \ldots, c_n)$ is a diagonal matrix, $A = (a_{ij})_{n\times n}$ and $B = (b_{ij})_{n\times n}$ are the connection weight matrix and the delayed connection weight matrix, respectively. As is known to all that (1) can exhibit chaotic phenomenons [1]. In order to show it clearly, we give the following example:

$$\begin{pmatrix} \frac{dx_1(t)}{dt} \\ \frac{dx_2(t)}{dt} \end{pmatrix} = -C \begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix} + A \begin{pmatrix} \tanh(x_1(t)) \\ \tanh(x_2(t)) \end{pmatrix} + B \begin{pmatrix} \tanh(x_1(t-\tau(t)) \\ \tanh(x_2(t-\tau(t)) \end{pmatrix}, \tag{3}$$

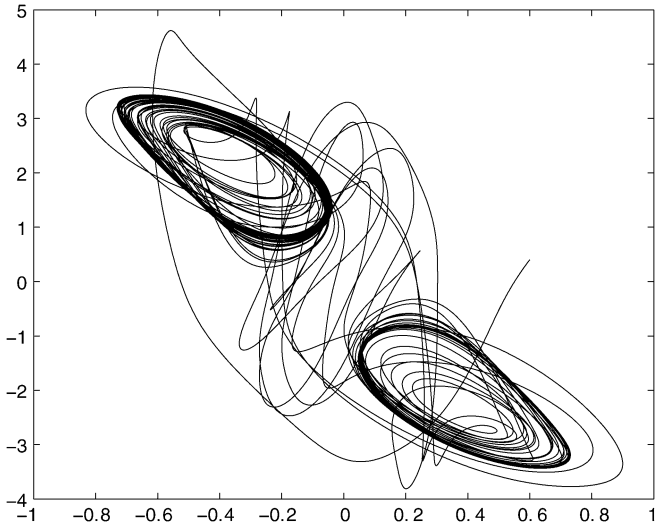

Fig. 1. Trajectories of state variables $x_1(t)$ and $x_2(t)$.

where $f_i(x_i(t)) = \tanh(x_i(t))$. $C = \left(\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right)$, $A = \left(\begin{smallmatrix} 2.0 & -0.1 \\ -5.0 & 3.0 \end{smallmatrix}\right)$, $B = \left(\begin{smallmatrix} -1.5 & -0.1 \\ -0.2 & -2.5 \end{smallmatrix}\right)$. $\tau(t) = 1 + 0.1\sin(t)$. Trajectories of state variables $x_1(t)$ and $x_2(t)$ are shown in Fig. 1. It is chaotic which is useful in encryption.

Pseudorandom number sequences with "good" properties are frequently used in secure communications and cryptosystems [2–4].

Remark 1. In [2–4], they all used the logistic map as follows:

$$x_{k+1} = bx_k(1 - x_k), \tag{4}$$

where b is a positive key parameter. However in this Letter, we used a more complicated neural network model (1). We have more unknown secret keys $f(x), C, A, B, \tau(t)$. Generally, we say that an encryption algorithm is secure if it is resistant to all known attacks under the assumption that the cryptanalyst has the details of the algorithm. Even if the cryptanalyst obtains the whole trajectories $x(t)$, it is difficult for him to estimate the unknown secret keys since there is little work about the adaptive synchronization of the unknown delayed neural networks.

There are four drawbacks in [2–4]. In [2,3], firstly, the distribution of ciphertext is not flat enough to ensure high security since the occurrence probability of cipher blocks decays exponentially as the number of iterations increases. Secondly, the cryptographic scheme is too slow to make it suitable for practical use such as the secure transmission of large multi-media files through the Internet. Thirdly, the length of ciphertext is at least twice that of plaintext, a byte of message may result in several tens of thousands of iterations that need two bytes to carry. Fourthly, the neural network model is used in this paper to ensure more high security than logistic map.

3. Random binary sequence generation

In this section, the approach to generate a sequence of independent and identical (i.i.d.) binary random variables from a class of ergodic chaotic maps were proposed in [5].

For any map $x(\cdot)$ defined on the interval $I = [d, e]$, we can give the value of $(x-d)/(e-d) \in [0,1]$ in a binary representation

$$\frac{x-d}{e-d} = 0.b_1(x)b_2(x)\cdots b_i(x)\cdots,$$
$$x \in [d,e], \quad b_i(x) \in \{0,1\}. \tag{5}$$

The ith bit $b_i(x)$ can be expressed as

$$b_i(x) = \sum_{r=1}^{2^i-1} (-1)^{r-1} \Theta_{(e-d)(r/2^i)+d}(x), \tag{6}$$

where $\Theta_t(x)$ is a threshold function defined by

$$\Theta_t(x) = \begin{cases} 0, & x < t, \\ 1, & x \geqslant t. \end{cases} \tag{7}$$

We can obtain a binary sequence $B_i^k = \{b_i(x_k)\}_{k=0}^{\infty}$ (where x_k is the kth iteration of the chaotic neural network which will be demonstrated later) can be obtained. It is composed of independent and identical distributed binary random variables [5].

Remark 2. In this Letter, we use the generated binary sequences B_i^k. We will see many advantages for choosing such binary sequences. Firstly, permuting each plaintext block with obtained binary subsequence before it is encrypted. Secondly, encrypting plaintext by employing the chaotic masking technique. Thirdly, determining the trajectory of the next iteration. Finally, determining the number of iterations in the chaotic map.

4. Novel encryption algorithm

In this section, we will give our main result about how to encrypt based on the proposed chaotic neural networks and generated binary sequences.

Firstly, we give the approach on the iterating of chaotic neural networks. Here we use fourth-order Runge–Kutta method to solve the delayed differential equation (1). We choose $n = 2$ for simplicity. The time step size h here is 0.01. Supposing $x_1(t)$ and $x_2(t)$ are the trajectories of the delayed neural network (3). The ith iterating of chaotic neural networks are $x_{1i} = x_1(ih)$, $x_{2i} = x_2(ih)$. In this Letter, we use the simulation example (3) for encrypted chaotic maps. The initial functions of (3) are chosen as $\phi(t) = (0.4, 0.6)^T$ on time interval $[-1.1, 0]$.

Secondly, a mapping of 8-bit message bytes to different regions (256 sites) in the interval $[-1,1]$ or $[-5,5]$ of the phase space of the neural network map in Fig. 1 is defined. To avoid the transient effect, the first $N_0 = 1000$ iterations are considered. In the proposed paper, each plaintext block length is 32-bit. This is different from the 8-bit length usually stated in [3,4].

Next, we give the main algorithm in this Letter which is given in Fig. 2.

Step 1. Get the start point x_0 from the last N_0 transient time iterations, $x_0 = x_1(N_0 h)$.

Step 2. Divide the message m into subsequences m_j of length l bytes (in our scheme $l = 4$):

$$m = \underbrace{p_0, p_1, \ldots, p_{l-1}}_{m_0}, \underbrace{p_l, p_{l+1}, \ldots, p_{2l-1}}_{m_1}, p_{2l}, \ldots. \tag{8}$$

Get four bytes of plaintext $p_j, p_{j+1}, p_{j+2}, p_{j+3}$ and combine them to form a binary message block P_j with 32-bits $P_j = p_j, p_{j+1}, p_{j+2}, p_{j+3}$.

Step 3. Based on the method described in Section 3, we can obtain binary sequences $A_j = B_i^1 B_i^2 \cdots B_i^{32}$, $A_j^1 = B_i^{33} B_i^{34} \cdots B_i^{37}$, $A_j^2 = B_i^{38}$ supplied by all the fourth bits, i.e., $i = 4$ in Eq. (6), through 38 iterations of the neural network (3). An integer D_j is computed as the decimal value of A_j^1. This value will be used to iterate the neural network successively for D_j times after the current block has been encrypted.

Step 4. Permute the message block P_j with left cyclic shift D_j bits and message block A_j with right cyclic shift D_j bits according to the approach illustrated in Fig. 3. Thus the message block P_j' and A_j' are obtained.

Step 5. If $A_j^2 = 0$, we use trajectory $x_1(t)$ for the successive block iterations in Step 3. If $A_j^2 = 1$, we use trajectory $x_2(t)$ for the successive block iterations in Step 3. Here, A_j^2 can be considered as a switching function for choosing which trajectory will be used in the next block iterations.

Step 6. Calculate the following manipulation with sequence A_j' and P_j':

$$C_j = P_j' \bigoplus A_j', \tag{9}$$

where $\bigoplus$ is the XOR operation. Thus we obtain the ciphertext C_j for the message block P_j. Dividing the ciphertext C_j into 8-bits partitions and we obtain $c_j, c_{j+1}, c_{j+2}, c_{j+3}$ of the plaintext bytes $p_j, p_{j+1}, p_{j+2}, p_{j+3}$, respectively.

Step 7. If all the plaintext have already been encrypted, the encryption process is finished. Otherwise, let $x_0 = x_{A_j^2+1}((38 + D_j)h)$, where A_j^2 is the sign for choosing trajectory in (3) as illustrated in Step 3. Then go to Step 2.

The decryption process is the same as the encryption one. We just need the following equation:

$$P_j' = C_j \bigoplus A_j', \tag{10}$$

we can obtain the permuted message block P_j'. By inverse permutation we obtain the message block P_j based on the value of D_j. Separate the block message into bytes, the plaintext is recovered. Note that the ciphertext has the same size as the plaintext corresponding twice in [3,4].

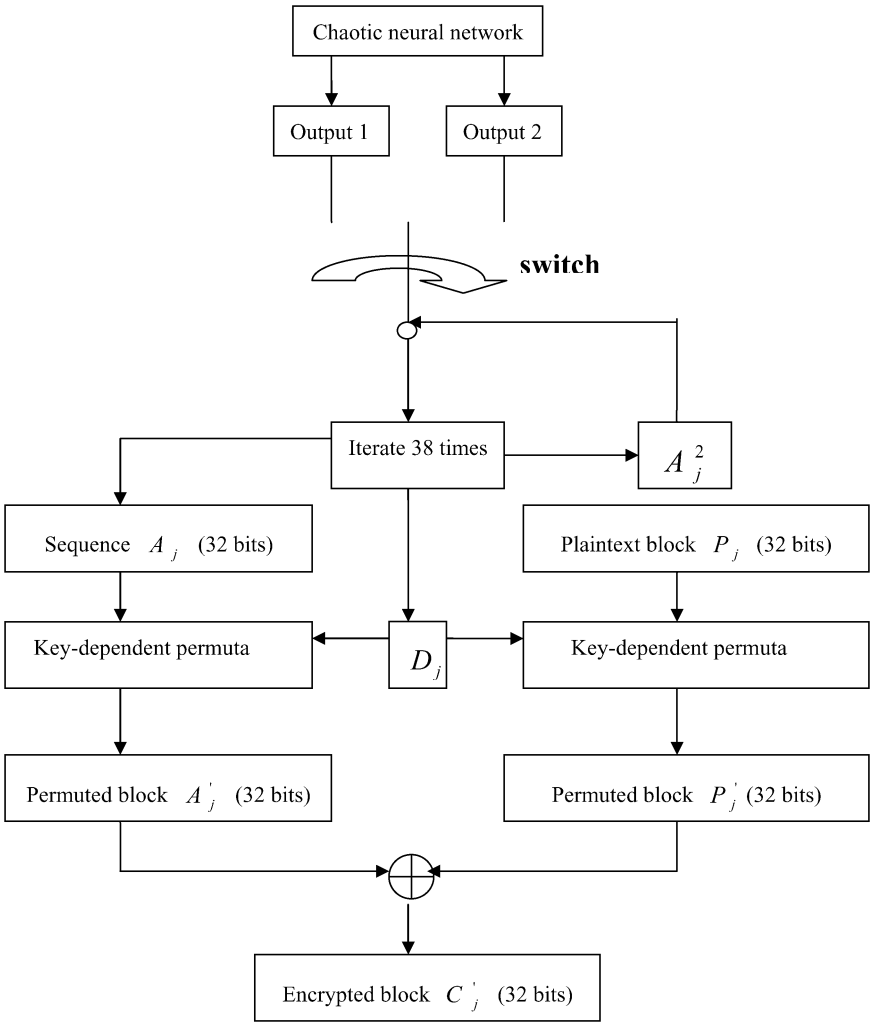

Fig. 2. Block diagram of the proposed scheme.

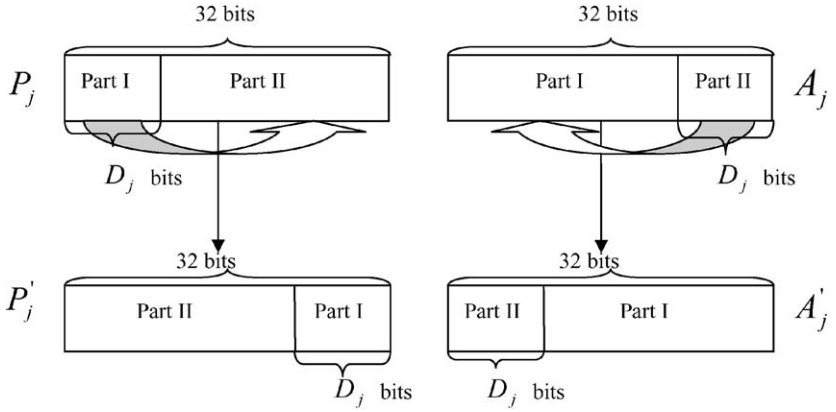

Fig. 3. Decryption of key-dependent permutation.

5. Security analysis

Security is a major issue of a cryptosystem. In this section, the evaluations mainly rest on the investigation of the performance of the above algorithm in Section 4.

Remark 3. The generated binary sequence is from chaotic map which means that the binary sequence is random. Also, it is easy to see that the iterations of chaotic map are also random based on the generated binary sequence. Therefore, D_j and sign for choosing trajectory A_j^2 are both random.

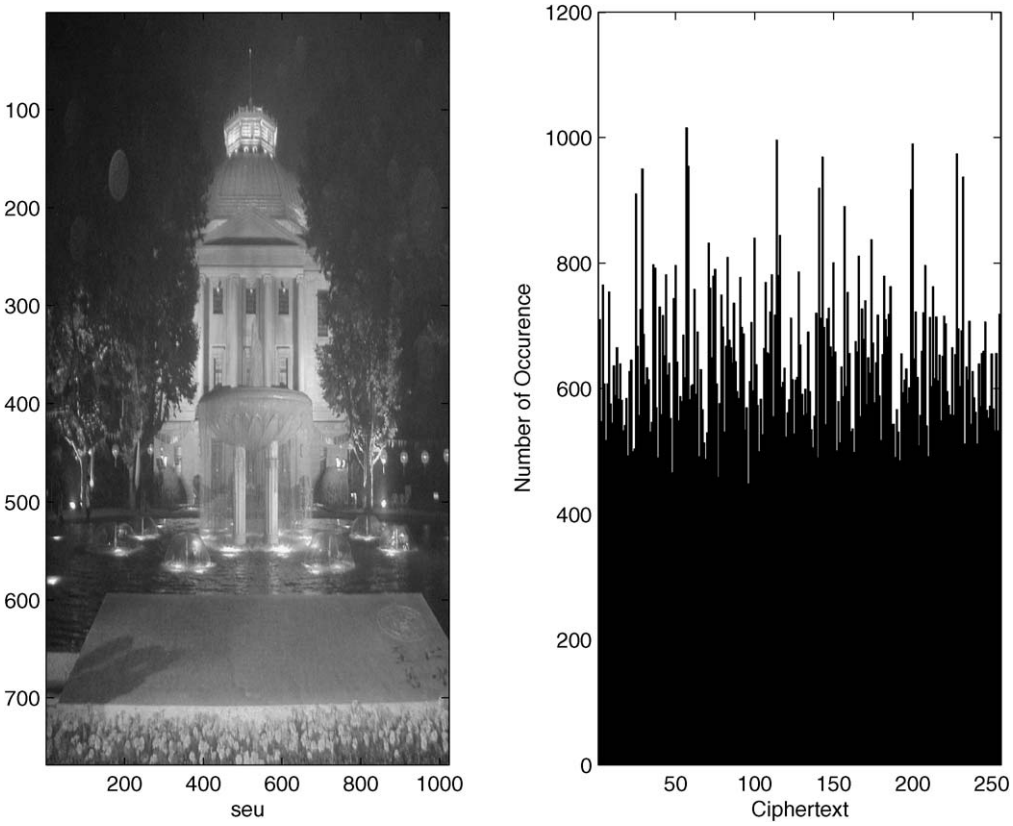

Fig. 4. Plaintext and distribution of ciphertext.

Remark 4. The single byte plaintext is mixed with four bytes, this means that a small variation of any one changes the outputs considerably. Also, one bit change in plaintext results in at most 32-bits changes in the ciphertext.

Remark 5. Note that $b_i(x)$ is the ith bit of x, only a part of the trajectories $x_j(t)$ $(j = 1, 2)$ of the chaotic map is used in the generating of binary sequences. Also, the chaotic maps are chose randomly. Even if cryptanalyst obtains the whole binary sequences B_i^k, it is difficult for him to recover the trajectories $x_j(t)$ $(j = 1, 2)$.

Remark 6. To the best of our knowledge, there is little results about the synchronization of unknown delayed system. Even if the cryptanalyst knows the structure (Hopfield neural netwok) of the used neural network model, the estimation of unknown initial function $\phi(t)$, time varying delay $\tau(t)$, the parameter matrices C, A, B and nonlinear function f are impossible.

Remark 7. Note the points of trajectories must be translated into discrete ones for the computation of computers. The chaotic signals used by us are $x(ih)$ $(i = 1, 2, \ldots)$ in (3). So the time step h is also a key parameter. Choosing different time steps h may result in wrong decryption. We will give a simulation example to show this later. Also, the numerical algorithm (fourth-order Runge–Kutta) we choose is also important for the process of encryption.

It is well known that the information entropy $H(m)$ of a plaintext message m can be calculated as

$$H(m) = -\sum_{i=0}^{2^N-1} p(m_i) \log \frac{1}{p(m_i)}. \tag{11}$$

As is known to all that the information entropy is defined to express the degree of uncertainties in the system. Let us suppose that if every symbols with equal probability, i.e., $m = \{m_1, m_2, \ldots, m_{2^8}\}$. After calculating, we obtain its entropy $H(m) = 8$, corresponding a random source. Actually, a practical information source seldom generates random messages, its information entropy is smaller than the ideal one. However, when designing an algorithm for encryption, we would like to expect the entropy to be close to the ideal one.

Here, one file are used for encryption and decryption by the proposed scheme. The file is: Image (.jpg) file named seu of size 164KB.

It is easy to see the distribution of ciphertext using the cryptosystem (3) is flat from Fig. 4, and the information entropy $H(m) = 7.9608$. The distribution of cipertext is a major concern in the process of encryption plaintext. If it is not flat enough, certain amount of information can be guessed by the opponent. Therefore, a flat distribution is desirable in cryptography. Also, it is a fast scheme in the process.

In order to show the small perturbations of unknown key parameters, we will choose some different parameters corre-

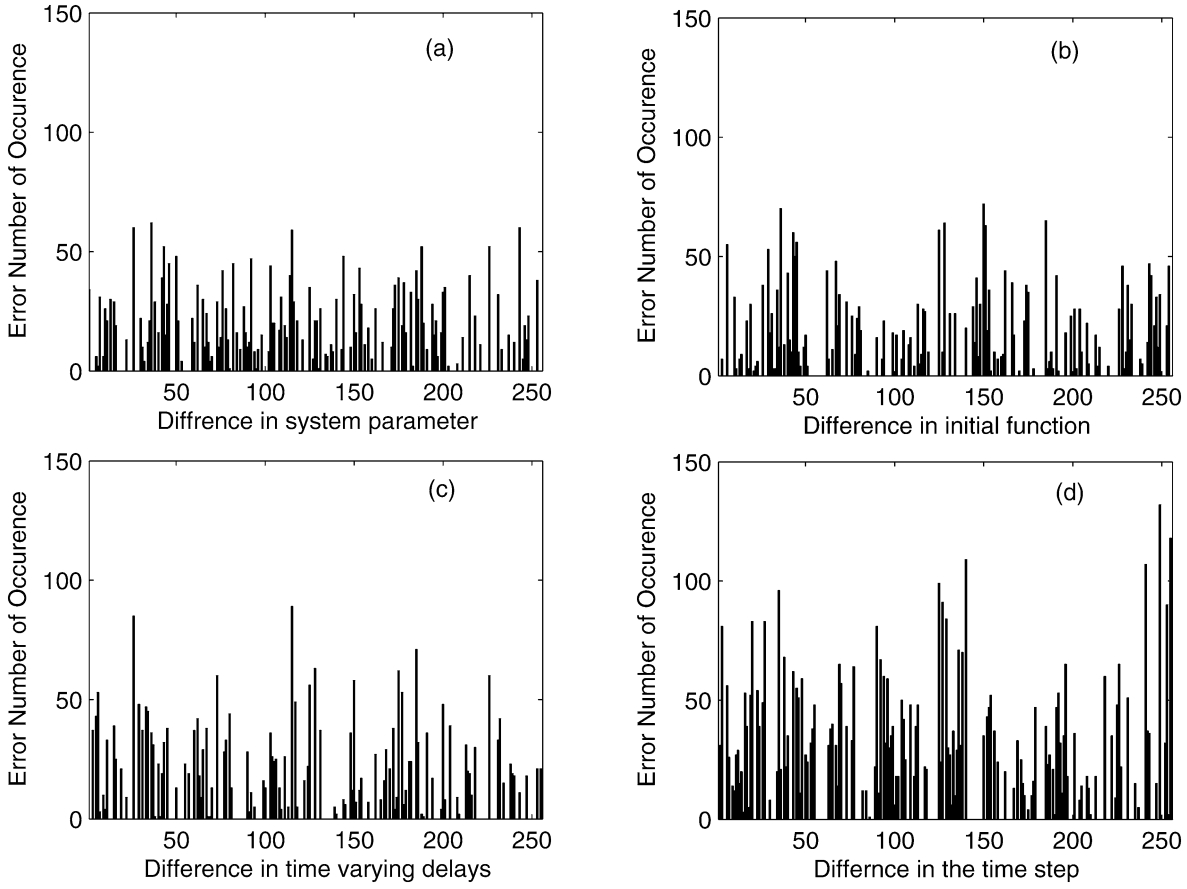

Fig. 5. Distribution of error ciphertext.

sponding to the system (3) listed as follows:

(a) $C = \begin{pmatrix} 0.99999 & 0 \\ 0 & 1 \end{pmatrix}$, $H(m) = 7.9608$,

(b) $\phi(t) = (0.39999, 0.6)^T$, $H(m) = 7.9612$,

(c) $\tau(t) = 0.99999 + 0.1\sin(t)$, $H(m) = 7.9605$,

(d) $h = 0.09999$, $H(m) = 7.9766$.

The distributions of Error Ciphertext which means the error number occurrences between the system listed above and the system (3) are shown in Fig. 5. We will see even a small variation of system parameters may result in the wrong encryption and decryption. Thus, it is really hard to reveal the original cryptosystem. Thus the Hopfield neural networks we used in this Letter is a good cryptosystem for encryption.

6. Conclusion

In this Letter, we proposed a novel approach of encryption based on chaotic neural networks with time varying delay. The chaotic neural network is used for generating binary sequences which will be used in masking plaintext, permutation of plaintext and choosing iterations of trajectory. It is easy to see the distribution of ciphertext is flat from simulation result. It is more secure since the difficult synchronization of chaotic neural networks with time varying delay.

References

[1] H. Lu, Phys. Lett. A 298 (2002) 109.
[2] G. Tang, X. Liao, Y. Chen, X. Zhang, K. Wong, Phys. Lett. A 349 (2005) 109.
[3] M.S. Baptista, Phys. Lett. A 240 (1998) 50.
[4] K.W. Kong, Phys. Lett. A 298 (2002) 238.
[5] T. Kohda, A. Tsuneda, IEEE Trans. Inform. Theory 43 (1) (1997) 104.
[6] A. Parravano, M.G. Cosenza, J. Jiménez, A. Marcano, Phys. Rev. E 65 (2002) 045201.
[7] V.I. Ponomarenko, M.D. Prokhorov, Phys. Rev. E 66 (2002) 026215.
[8] R. He, P.G. Vaidya, Phys. Rev. E 57 (1998) 1532.
[9] R. Mislovaty, E. Klein, I. Kanter, W. Kinzel, Phys. Rev. Lett. 91 (2003) 118701.
[10] E. Klein, R. Mislovaty, I. Kanter, W. Kinzel, Phys. Rev. E 72 (2005) 016214.
[11] B. Fraser, P. Yu, T. Lookman, Phys. Rev. E 66 (2002) 017202.
[12] H.D.I. Abarbanel, M.B. Kennel, Phys. Rev. Lett. 80 (1998) 3153.
[13] J.J. Hopfild, Proc. Natl. Acad. Sci. USA 81 (1984) 3088.
[14] J. Cao, P. Li, W. Wang, Phys. Lett. A 353 (4) (2006) 318.
[15] J. Cao, J. Lu, Chaos 16 (2006) 013133.
[16] W. Wang, J. Cao, Physica A 366 (2006) 197.
[17] J. Lu, J. Cao, Chaos 15 (2005) 043901.
[18] J. Cao, K. Yuan, D.W.C. Ho, J. Lam, Chaos 16 (2006) 013105.