Cascade Chaotic System With Applications

Yicong Zhou, *Senior Member, IEEE,* Zhongyun Hua, *Student Member, IEEE,*
Chi-Man Pun, *Senior Member, IEEE,* and C. L. Philip Chen, *Fellow, IEEE*

***Abstract*—Chaotic maps are widely used in different applications. Motivated by the cascade structure in electronic circuits, this paper introduces a general chaotic framework called the cascade chaotic system (CCS). Using two 1-D chaotic maps as seed maps, CCS is able to generate a huge number of new chaotic maps. Examples and evaluations show the CCS's robustness. Compared with corresponding seed maps, newly generated chaotic maps are more unpredictable and have better chaotic performance, more parameters, and complex chaotic properties. To investigate applications of CCS, we introduce a pseudo-random number generator (PRNG) and a data encryption system using a chaotic map generated by CCS. Simulation and analysis demonstrate that the proposed PRNG has high quality of randomness and that the data encryption system is able to protect different types of data with a high-security level.**

***Index Terms*—Cascade chaotic system (CCS), chaotic map, data encryption, pseudo-random number generator (PRNG).**

I. Introduction

A DYNAMICAL system is a concept in mathematics where a fixed rule describes the time dependence of a point in a geometrical space [1]. In the past decades, researchers paid increasing attentions to dynamical systems [2], especially to chaotic maps [3] that are traditional dynamical systems. Chaotic maps have properties of ergodicity and unpredictability. They can generate totally different chaotic sequences using different parameters or initial values. Thanks for these significant properties, chaotic maps are useful tools in applications of mathematics, computer science, and engineering. Especially in security applications, chaotic maps show excellent performance in pseudo-random number generators (PRNGs) [4], [5], data and image encryption [6]–[8].

Recently, many chaotic maps were developed [9], [10]. They can be classified into two categories: 1) 1-D and 2) high-dimensional (HD) chaotic maps. 1-D chaotic maps are mathematical systems that simulate the evolutions of a single variable over discrete steps in time. Examples include the Logistic map, Tent map, Gauss map, and Dyadic transformation [9]. These 1-D chaotic maps usually have simple structures and are easy to implement. They have excellent chaotic properties and were used for different security applications [7]. However, they have several security weaknesses: 1) their chaotic ranges are limited [8]; 2) they have small number of parameters; and 3) their outputs are easy to predict with low computation costs [11]–[13].

On the other hand, HD chaotic maps model the evolutions of at least two variables. Examples are the Hénon map [14], Lorenz system [14], Chen and Lee system [15], and hyperchaotic systems [16]. Compared with 1-D chaotic maps, HD chaotic maps usually have better chaotic performance and their chaotic orbits are more difficult to predict [17]. However, HD chaotic maps have high-computation costs and are difficult to implement in hardware. These weaknesses restrict their performance in some chaos-based applications, especially in real-time applications.

To overcome the limited chaotic performance of 1-D chaotic maps and implementation difficulty of HD chaotic maps, this paper proposes a cascade chaotic system (CCS) as a general 1-D chaotic framework. CCS connects two 1-D chaotic maps (seed maps) in series. The output of the first seed map is linked to the input of the second seed map. The output of the second one is fed back into the input of the first one for recursive iterations, and it is also the output of CCS. As a general cascade framework, CCS is able to produce new chaotic maps (NCMs) using any two 1-D chaotic maps as seed maps. Three examples of NCMs are provided. Evaluations and analysis results show that these NCMs have more parameters and better chaotic performance than their corresponding seed maps.

Using one chaotic map generated by CCS as an example, we further propose a PRNG and a data encryption system. The National Institute of Standards and Technology (NIST) SP800-22 and TestU01 test results are provided to show excellent randomness of PRNG. Simulations and security analysis are provided to demonstrate that the data encryption system can encrypt different data with a high level of security and outperform several state-of-the-art algorithms.

In summary, our main contributions in this paper are as follows.

1) We introduce the CCS that is a general chaotic framework with a simple and effective structure.
2) Chaotic performance of CCS is studied theoretically and experimentally.
3) We propose a CCS-based PRNG.
4) Random properties of the proposed PRNG are evaluated using two test standards.
5) We also develop a data encryption system using CCS.

Manuscript received July 19, 2014; revised October 2, 2014; accepted October 3, 2014. Date of publication October 30, 2014; date of current version August 14, 2015. This work was supported in part by the Macau Science and Technology Development Fund under Grant FDCT/017/2012/A1, and in part by the Research Committee at University of Macau under Grant MYRG2014-00003-FST, Grant MRG017/ZYC/2014/FST, Grant MYRG113(Y1-L3)-FST12-ZYC, and Grant MRG001/ZYC/2013/FST. This paper was recommended by Associate Editor R. Lynch.

The authors are with the Department of Computer and Information Science, University of Macau, Macau 999078, China (e-mail: yicongzhou@umac.mo).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2014.2363168

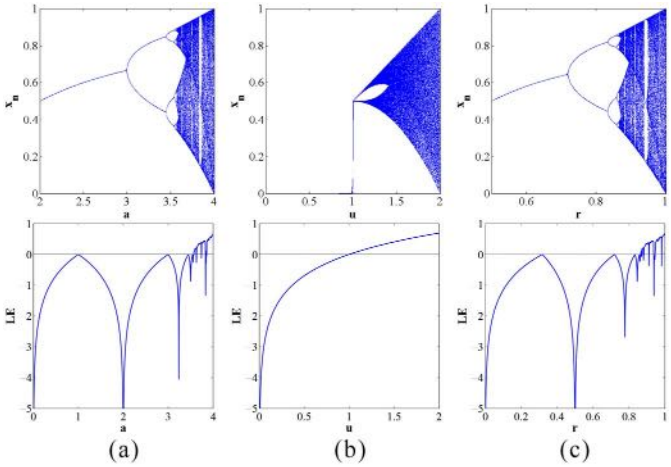

Fig. 1. Chaotic performance of three chaotic maps. The first and second rows are bifurcation diagrams and Lyapunov exponents of (a) logistic map, (b) tent map, and (c) sine map.

6) Encryption and security performance of the proposed data encryption system are analyzed.

The rest of this paper is organized as follows. Section II will review three existing 1-D chaotic maps and their properties. Section III will introduce CCS and discuss its properties. Section IV will present three chaotic maps generated by CCS and discuss their properties. Section V will quantitatively evaluate the performance of these chaotic maps. Using one chaotic map as an example, Section VI will propose a PRNG and evaluate its randomness performance using several tests, and then Section VII will introduce a data encryption system and provide simulation results and security analysis to show its encryption performance. Section VIII will give the conclusion.

II. Background

This section briefly reviews three traditional chaotic maps. They will be used as seed maps for the proposed CCS to generate NCMs in Section IV.

A. Logistic Map

The Logistic map is a 1-D discrete chaotic map widely used in many applications. It was proved to have good chaotic performance [18] and can generate chaotic sequences with range of $[0, 1]$ by stretching out and drawing back an initial input value within $[0, 1]$. Mathematically, the Logistic map is defined

$$x_{n+1} = ax_n(1 - x_n) \tag{1}$$

where a is a parameter with range of $[0, 4]$.

The bifurcation diagram plots output sequences of a chaotic map along with the change of its parameter(s). The Lyapunov exponent (LE) [19], [20] is a measure to describe chaotic behaviors of a dynamical system. A positive LE value indicates that the dynamical system is chaotic.

The bifurcation diagram and LE values of the Logistic map are shown in Fig. 1(a). From its bifurcation diagram, one can see that the Logistic map is chaotic when $a \in [3.57, 4]$, and that it has better chaotic behaviors when a is close to 4.

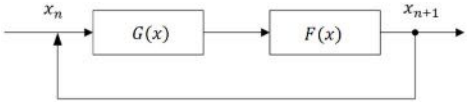

Fig. 2. Structure of CCS.

B. Tent Map

The Tent map is another 1-D discrete chaotic map that performs the stretching and folding operations. When its input is smaller than 0.5, it stretches the output into range of $[0, 1]$. Otherwise, when its input is bigger than or equal to 0.5, the Tent map folds its input value into range of $[0, 0.5]$, and then stretches it into range of $[0, 1]$ to generate its output. Its mathematical representation is defined

$$x_{n+1} = \begin{cases} ux_n & \text{for } x_n < 0.5 \\ u(1 - x_n) & \text{for } x_n \geq 0.5 \end{cases} \tag{2}$$

where parameter u is in the range of $[0, 2]$.

Fig. 1(b) shows the bifurcation diagram and LE values of the Tent map. As can be seen, the Tent map has good chaotic performance when parameter $u \in (1, 2]$. When u is close to 2, its output sequences distribute almost in the whole data range of $[0, 1]$.

C. Sine Map

The Sine map is another commonly used chaotic map that has chaotic behaviors similar to the Logistic map. But its mathematical definition is totally different, as shown

$$x_{n+1} = r \sin(\pi x_n) \tag{3}$$

where parameter r is between 0 and 1, x_n is the iterative outputs/inputs with range of $[0, 1]$.

When $r \in [0.867, 1]$, the Sine map has chaotic behaviors. Its bifurcation diagram and LE values are shown in Fig. 1(c). From its bifurcation diagram, the Sine map shows better chaotic behaviors when parameter r is close to 1.

III. CCS

Motivated by cascade structures in electronic circuits, this section proposes a CCS. It can generate new 1-D chaotic maps using a combination of two existing 1-D chaotic maps.

A. CCS

Fig. 2 shows the structure of CCS, where $G(x)$ and $F(x)$ are two seed maps. CCS connects two seed maps in series. The output of $G(x)$ is fed into the $F(x)$'s input, and the $F(x)$'s output is then fed back into the $G(x)$'s input for recursive iterations.

Mathematically, the proposed CCS is defined in the following, where $G(x)$ and $F(x)$ are two seed maps:

$$x_{n+1} = \Gamma(x_n) = F(G(x_n)). \tag{4}$$

Any existing 1-D chaotic map can be used as the seed map of CCS. Users have the flexibility to set seed maps $F(x)$ and $G(x)$ as the same or different chaotic maps.

1) When $F(x)$ and $G(x)$ are the same 1-D chaotic maps, namely $F(x) = G(x)$, CCS in (4) changes to

$$x_{n+1} = F(F(x_n)) \quad \text{or} \quad x_{n+1} = G(G(x_n)). \tag{5}$$

CCS becomes a structure that a 1-D chaotic map is cascaded with itself. For example, when $F(x)$ and $G(x)$ are two Sine maps, CCS is the Double-Sine map that will be discussed in Section IV-C in detail.

2) When $F(x)$ and $G(x)$ are selected as different chaotic maps, namely $F(x) \neq G(x)$, CCS becomes another 1-D chaotic structure defined by (4) or by

$$x_{n+1} = G(F(x_n)). \tag{6}$$

Changing settings of $F(x)$ and $G(x)$ or even the order of two seed maps, CCS yields a different 1-D chaotic map. For example, the Tent-Logistic and Logistic-Tent maps are completely different ones to be discussed in Sections IV-A and IV-B.

CCS offers users the great flexibility to generate a large number of NCMs using different settings of $F(x)$ and $G(x)$. Compared with their corresponding seed maps, chaotic maps generated by CCS are completely different, and have more parameters and more complex chaotic behaviors. This will be verified by analysis results in Section V.

Moreover, the structure of CCS in Fig. 2 can be further extended into three or more cascaded seed maps. This offers users even more flexibility of selecting seed maps. The resulting chaotic maps have much more complicated chaotic behaviors and more parameter settings, and thus they may have much better chaotic performance and generate more random and unpredictable output sequences. On the other hand, however, cascading more seed maps may result in many side effects including significant time delay, difficulty in hardware implementation, and complexity of performance analysis.

B. Chaotic Behavior Analysis

Connecting two chaotic maps $G(x)$ and $F(x)$ in series, the output sequences of CCS have the structure of $G(x)$, $F(x)$, or both. From the definition in (4), CSS contains all parameters of its seed maps. Thus, it has more parameters and complex properties than its seed maps.

Because CCS is a generalized framework of chaotic systems, using different chaotic maps as its seed maps, or even changing the order of its seed maps, CCS yields totally different chaotic maps. Directly analyzing or proving the chaotic performance of CCS becomes extremely difficult. Because the LE [19], [20] gives a quantitative description of changes between two neighboring output values of a dynamical system, it can be used to describe chaotic behaviors of a chaotic system. We hereby use LE to analyze chaotic performance of the proposed CCS.

Assume x_0 and y_0 are two extremely close initial values of CCS in (4). After the first iteration, the difference $|x_1 - y_1|$ is defined by

$$\begin{aligned} |x_1 - y_1| &= |\Gamma(x_0) - \Gamma(y_0)| \\ &= \frac{|F(G(x_0)) - F(G(y_0))|}{|G(x_0) - G(y_0)|} \frac{|G(x_0) - G(y_0)|}{|x_0 - y_0|} |x_0 - y_0|. \end{aligned}$$

For $x_0 \to y_0$, we have $G(x_0) \to G(y_0)$, then

$$\left|\frac{dF}{dx}|_{G(x_0)}\right| \approx \lim_{G(x_0)\to G(y_0)} \frac{|F(G(x_0)) - F(G(y_0))|}{|G(x_0) - G(y_0)|}$$
$$\left|\frac{dG}{dx}|_{x_0}\right| \approx \lim_{x_0 \to y_0} \frac{|G(x_0) - G(y_0)|}{|x_0 - y_0|}.$$

Thus

$$|x_1 - y_1| \approx \left|\frac{dF}{dx}|_{G(x_0)}\right| \left|\frac{dG}{dx}|_{x_0}\right| |x_0 - y_0|.$$

Similarly, after the second iteration, the difference $|x_2 - y_2|$ is defined by

$$\begin{aligned} |x_2 - y_2| &= |\Gamma(x_1) - \Gamma(y_1)| \\ &= \frac{|F(G(x_1)) - F(G(y_1))|}{|G(x_1) - G(y_1)|} \frac{|G(x_1) - G(y_1)|}{|x_1 - y_1|} |x_1 - y_1| \\ &\approx \left|\frac{dF}{dx}|_{G(x_1)}\right| \left|\frac{dG}{dx}|_{x_1}\right| \left|\frac{dF}{dx}|_{G(x_0)}\right| \left|\frac{dG}{dx}|_{x_0}\right| |x_0 - y_0|. \end{aligned}$$

After the nth ($n \to \infty$) iteration, the difference between x_n and y_n is defined by

$$\begin{aligned} |x_n - y_n| &= |\Gamma(x_{n-1}) - \Gamma(y_{n-1})| \\ &\approx \left|\prod_{i=0}^{n-1} \frac{dF}{dx}|_{G(x_i)}\right| \left|\prod_{i=0}^{n-1} \frac{dG}{dx}|_{x_i}\right| |x_0 - y_0|. \end{aligned}$$

Then the average change in each iteration from $|x_0 - y_0|$ to $|x_n - y_n|$ is

$$\Delta_{\Gamma(x)} \approx \left\{\left|\prod_{i=0}^{n-1} \frac{dF}{dx}|_{G(x_i)}\right| \left|\prod_{i=0}^{n-1} \frac{dG}{dx}|_{x_i}\right|\right\}^{\frac{1}{n}}.$$

Therefore, LE of $\Gamma(x)$ is defined by

$$\begin{aligned} \lambda_{\Gamma(x)} &= \ln(\Delta_{\Gamma(x)}) \\ &= \lim_{n\to\infty} \frac{1}{n} \sum_{i=0}^{n-1} \ln\left|\frac{dF}{dx}|_{G(x_i)}\right| + \lim_{n\to\infty} \frac{1}{n} \sum_{i=0}^{n-1} \ln\left|\frac{dG}{dx}|_{x_i}\right|. \end{aligned} \tag{7}$$

Similarly, LEs of $F(x)$ and $G(x)$ are defined by

$$\lambda_{F(x)} = \lim_{n\to\infty} \frac{1}{n} \sum_{i=0}^{n-1} \ln\left|\frac{dF}{dx}|_{x_i}\right| \tag{8}$$

$$\lambda_{G(x)} = \lim_{n\to\infty} \frac{1}{n} \sum_{i=0}^{n-1} \ln\left|\frac{dG}{dx}|_{x_i}\right|. \tag{9}$$

Then, (7) becomes

$$\lambda_{\Gamma(x)} = \lambda_{F(x)} + \lambda_{G(x)}. \tag{10}$$

Therefore, LE of CCS is a combination of the LE values of its two seed maps. When $\lambda_{\Gamma(x)} > 0$, the trajectories of two output sequences of CCS dramatically diverge as the number of iterations increases, and CCS becomes chaotic. A bigger positive LE value means the faster divergence of two trajectories, resulting in better chaotic performance. Chaotic behaviors of CCS are summarized as follows.

1) When $\lambda_{F(x)} > 0$ and $\lambda_{G(x)} > 0$, $\lambda_{\Gamma(x)} > 0$, $\lambda_{\Gamma(x)} > \lambda_{F(x)}$ and $\lambda_{\Gamma(x)} > \lambda_{G(x)}$. When both seed maps are chaotic,

CCS is chaotic and has better chaotic performance than its seed maps.

2) When $\lambda_{G(x)} \leq 0$ and $\lambda_{F(x)} \leq 0$, $\lambda_{\Gamma(x)} \leq 0$. CCS does not have any chaotic behavior when neither seed map is chaotic.
3) When $\lambda_{G(x)} > 0$ and $\lambda_{F(x)} \leq 0$, or $\lambda_{F(x)} > 0$ and $\lambda_{G(x)} \leq 0$, we have

$$\lambda_{\Gamma(x)} \begin{cases} > 0 & \text{if } \lambda_{F(x)} + \lambda_{G(x)} > 0 \\ \leq 0 & \text{if } \lambda_{F(x)} + \lambda_{G(x)} \leq 0. \end{cases}$$

When there is only one seed map that is chaotic, CCS will be chaotic if and only if $\lambda_{F(x)} + \lambda_{G(x)} > 0$.

In general, CCS is chaotic when there is at least one seed map in the chaotic range. It has better chaotic performance when both seed maps are chaotic. This will be further verified by experimental results in Section V-B.

IV. Examples of NCMs

Using different 1-D chaotic maps as seed maps, CCS is able to generate a large number of NCMs. To show the robustness of CCS, this section provides three examples of those NCMs and discusses their chaotic performance.

A. Tent-Logistic Map

When $G(x)$ is set to be the Tent map and $F(x)$ is set to be the Logistic map in Fig. 2, CCS becomes an NCM called the Tent-Logistic map. Mathematically, the Tent-Logistic map is defined

$$x_{n+1} = \begin{cases} aux_n(1 - ux_n) & \text{for } x_n < 0.5 \\ au(1 - x_n)(1 - u(1 - x_n)) & \text{for } x_n \geq 0.5 \end{cases} \quad (11)$$

where parameters a and u come from its two seed maps, the Logistic and Tent maps. Therefore, the range of two parameters are $a \in [0, 4]$ and $u \in [0, 2]$.

The 2-D and 1-D bifurcation diagrams of the Tent-Logistic map are shown in the first row of Fig. 3. Fig. 1 shows that the Logistic and Tent maps have chaotic behaviors when $a \in [3.57, 4]$ and $u \in (1, 2]$, respectively. As can be seen in Fig. 3, the Tent-Logistic map has wider chaotic ranges along with parameters a and u. This means that chaotic maps generated by CCS can inherit and enhance the chaotic performance of their seed maps.

B. Logistic-Tent Map

When changing the positions of two seed maps in the Tent-Logistic map, namely $G(x)$ to be the Logistic map and $F(x)$ to be the Tent map in Fig. 2, another chaotic map is generated, called the Logistic-Tent map. Its mathematic representation is defined

$$x_{n+1} = \begin{cases} uax_n(1 - x_n) & \text{for } ax_n(1 - x_n) < 0.5 \\ u(1 - ax_n(1 - x_n)) & \text{for } ax_n(1 - x_n) \geq 0.5 \end{cases} \quad (12)$$

where parameters a and u also have the same settings with the Logistic and Tent maps, $a \in [0, 4]$ and $u \in [0, 2]$.

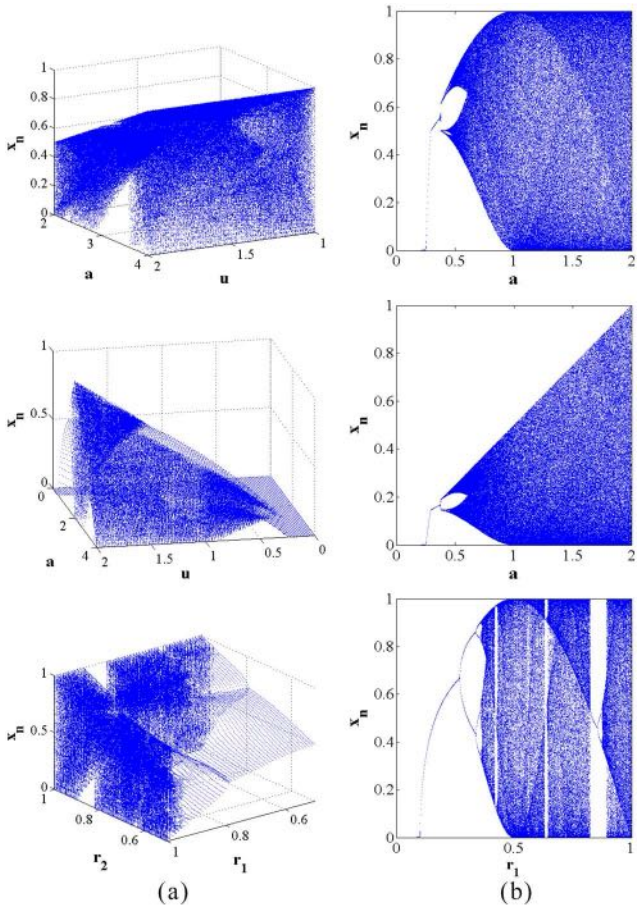

Fig. 3. Bifurcation diagrams of three NCMs. The first, second, and third rows show (a) 2-D and (b) 1-D bifurcation diagrams of the tent-logistic, logistic-tent, and double-sine maps, respectively.

The second row in Fig. 3 shows the 2-D and 1-D bifurcation diagrams of the Logistic-Tent map. Compared with the bifurcation diagrams of the Tent-Logistic map, we can observe that using two seed maps with different orders yields two chaotic maps with completely different chaotic behaviors.

C. Double-Sine Map

In CCS, two seed maps could be the same chaotic map. In this scenario, the chaotic map cascades itself. When both seed maps are selected as the Sine map, CCS generates a chaotic map, called the Double-Sine map. Its mathematic representation is defined

$$x_{n+1} = r_2 \sin(\pi r_1 \sin(\pi x_n)) \quad (13)$$

where r_1 and r_2 are two parameters, and $r_1, r_2 \in [0, 1]$.

Its 2-D and 1-D bifurcation diagrams are shown in the third row of Fig. 3. As can be seen, even simply linking two Sine maps in series, chaotic behaviors of the Double-Sine map are totally different and much better than the Sine map.

V. Performance Analysis

This section analyzes chaotic performance of three chaotic maps presented in Section IV. The comparison results show that these NCMs have better chaotic performance than their corresponding seed maps.

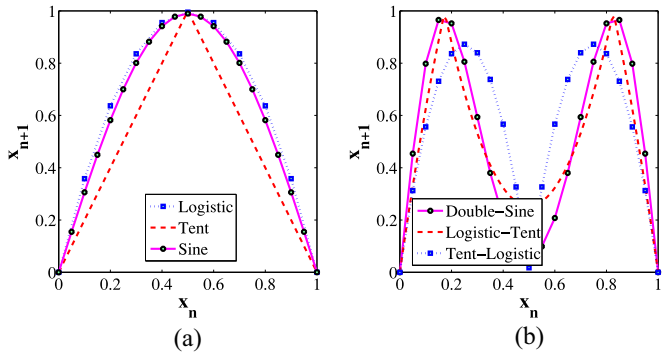

Fig. 4. Iteration function diagrams of (a) logistic, tent, and sine maps and (b) double-sine, logistic-tent, and tent-logistic maps.

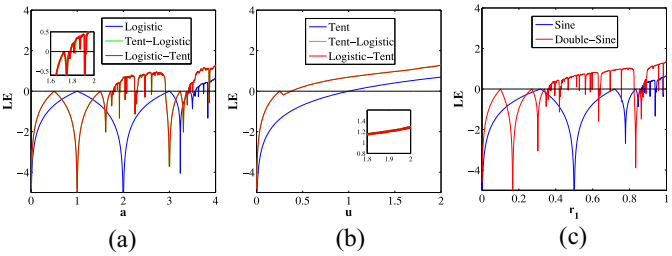

Fig. 5. LE comparison of (a) logistic, tent-logistic, and logistic-tent maps; (b) tent, tent-logistic, and logistic-tent maps, and (c) sine and double-sine maps, respectively.

A. Iteration Function Diagram

For an iterative dynamical system like $x_{n+1} = f(x_n)$, the iteration function diagram describes the output x_{n+1} along with the input x_n.

The iteration function diagrams of these NCMs in Fig. 4(b) have more complex patterns than those of their seed maps in Fig. 4(a). This is because their outputs are combinations of chaotic orbits of two seed maps. These complex iteration function diagrams have security benefits because they are difficult to predict. NCMs are more suitable for security applications than their corresponding seed maps.

B. Lyapunov Exponent

As mentioned in Section III-B, a positive LE value of a dynamical system indicates that the trajectories of its two output sequences generated from extremely close initial inputs diverge dramatically in each iteration. The bigger positive LE values indicates faster divergences of the trajectories of outputs, and thus better chaotic performance.

The LE values of NCMs and their seed maps are plotted in Fig. 5. One can observe that, under the same parameter settings, NCMs have larger LE values than their corresponding seed maps in most parameter ranges.

Even the Tent-Logistic and Logistic-Tent maps are two chaotic maps with different definitions and trajectories, they have similar chaotic characteristics due to the fact that they are derived from the same seed maps and that their LE distributions are similar.

C. Kolmogorov Entropy

The Kolmogorov entropy (KE) is also known as Metric entropy, Kolmogorov–Sinai entropy, or K entropy. It is a measure that describes how much information on average is needed to predict the trajectory of a dynamical system in each unit time [21]. Mathematically, KE is defined

$$KE = \lim_{\tau \to 0} \tau^{-1} \lim_{\varepsilon \to 0} \lim_{m \to \infty} K_{m,\tau}(\varepsilon) \tag{14}$$

where m is the embedding dimension; $K_{m,\tau}(\varepsilon)$ is defined by

$$K_{m,\tau}(\varepsilon) = - \sum_{i_1, i_2, \ldots, i_m \le n(\varepsilon)} p(i_1, i_2, \ldots, i_m) \times \log p(i_1, i_2, \ldots, i_m) \tag{15}$$

where $\phi_{i_1}, \phi_{i_2}, \ldots, \phi_{i_m}$ are nonoverlapping partitions of the phase plane of a dynamical system, $p(i_1, i_2, \ldots, i_m)$ is the joint probability of finding the trajectory in partition ϕ_{i_1} at time τ, in partition ϕ_{i_2} at time 2τ, ..., in partition ϕ_{i_m} at time $m\tau$.

A dynamical system with a positive KE value is unpredictable, and a larger positive KE value indicates more unpredictability and better chaotic performance [22].

In our experiments, we choose 12 000 continuous points from the trajectories of NCMs and their seed maps under the same parameter settings. Fig. 6 plots the KE results calculated by the method in [23]. Fig. 6(a) and (b) shows that the Tent-Logistic and Logistic-Tent maps have much bigger positive KE values than the Tent and Logistic maps in all parameter ranges. Fig. 6(c) shows that the Double-Sine map has larger positive KE values than the Sine map in most parameter settings. Therefore, NCMs have more unpredictability and better chaotic performance than their seed maps.

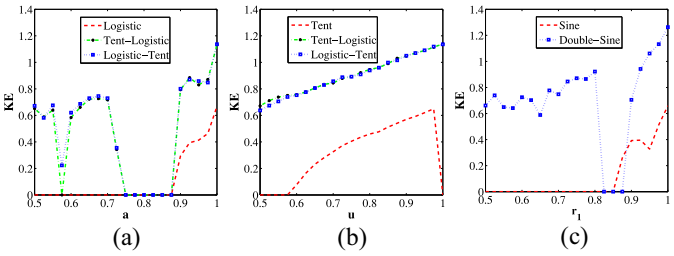

Fig. 6. KE comparison of (a) logistic, tent-logistic, and logistic-tent maps; (b) tent, tent-logistic, and logistic-tent maps, and (c) sine and double-sine maps, respectively.

D. Correlation Test

The distance of two data sequences can be evaluated by the correlation test. Mathematically, the correlation of two data sequences is defined

$$Co = \frac{E[(X_t - \mu_X)(Y_t - \mu_Y)]}{\sigma_X \sigma_Y} \tag{16}$$

where X_t and Y_t are two data sequences, μ and σ are the mean value and standard deviation, $E[.]$ is the expectation function.

If two data sequences X_t and Y_t are close to each other, the correlation value is close to 1; if X_t and Y_t are totally different, the correlation value is close to 0, indicating two sequences have an extremely low relationship.

Here, we use correlation to evaluate how a chaotic map is sensitive to its initial values and parameters. The test results of different chaotic maps are shown in Fig. 7 and Table I. The output sequence pairs S_1 and S_2, S_3, and S_4 are generated by

TABLE I
CORRELATION COMPARISONS OF THE OUTPUT SEQUENCES OF NCMS AND THEIR SEED MAPS

Parameters (u, a)	(1.46, 3.64)		(1.74, 3.96)	
	Correlation of S_1, S_2	Correlation of S_3, S_4	Correlation of S_1, S_2	Correlation of S_3, S_4
Logistic map (a)	0.833013	0.815053	0.057409	-0.127028
Tent map (u)	0.066046	0.140041	0.013925	0.022016
Logistic-Tent map (u, a)	-0.029632	0.026886	-0.005645	0.014180
Tent-Logistic map (u, a)	0.010853	0.104041	-0.006496	0.011341
Parameters (r_1, r_2)	(0.895, 0.913)		(0.902, 0.948)	
	Correlation of S_1, S_2	Correlation of S_3, S_4	Correlation of S_1, S_2	Correlation of S_3, S_4
Sine map (r_1)	-0.050228	0.223317	-0.075275	0.029231
Sine map (r_2)	-0.030918	0.012340	0.220003	0.136564
Double-Sine map (r_1, r_2)	0.000578	0.009010	-0.027488	0.017769

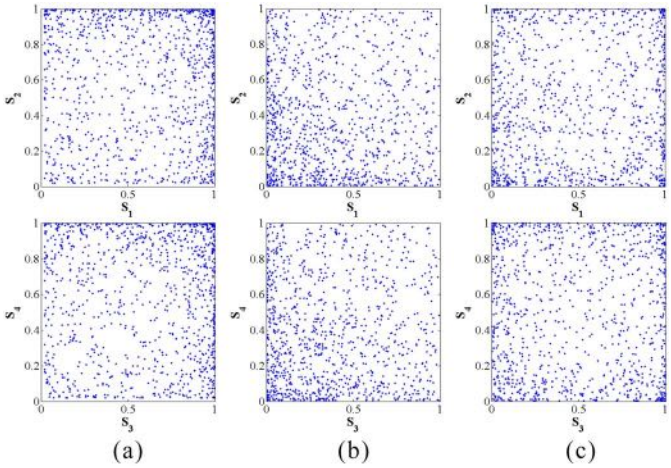

Fig. 7. Correlation plots of the output sequences generated by (a) tent-logistic, (b) logistic-tent, and (c) double-sine maps with a tiny change applied to initial values (the top row) and parameters (the bottom row).

applying a tiny change to initial values and parameters, respectively. As can be seen in Fig. 7, tiny changes in initial values or parameters of NCMs yield the output sequences distributing dynamically in the entire data range. This means that the output sequences have no relationship with each other. Thus, these NCMs are extremely sensitive to their initial values and parameters.

The quantitative comparisons in Table I show that the output sequences of NCMs have smaller absolute correlation values than their seed maps. This shows that NCMs are more sensitive to initial values and parameters than their seed maps.

VI. PROPOSED PRNG

Because chaotic maps have properties of ergodicity, unpredictability, and sensitivity to initial values/parameters, they are ideal candidates for designing a PRNG. Recently, many chaotic map-based PRNGs were proposed [5], [24]. The chaotic performance of chaotic maps determines the randomness performance of PRNGs. As discussed in previous sections, chaotic maps generated by CCS show better chaotic performance than existing seed maps, and thus they are more suitable for PRNGs.

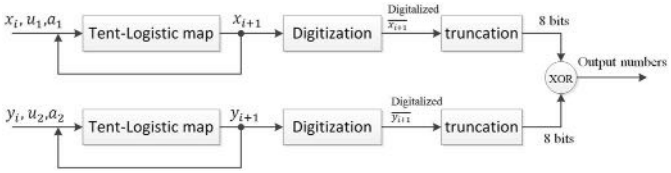

Fig. 8. Structure of TLPRNG.

This section uses the Tent-Logistic map as an example of NCMs generated by CCS to design a new PRNG, and then analyzes its random property.

A. PRNG

The proposed PRNG is called the Tent-Logistic map-based PRNG (TLPRNG). Its block diagram is shown in Fig. 8. Suppose $\{x_i, i = 1, 2, \ldots, N\}$ and $\{y_i, i = 1, 2, \ldots, N\}$ are two chaotic sequences generated by the Tent-Logistic map with different initial values and parameters, TLPRNG is defined

$$\text{TLPRNG}(i) = X(i) \oplus Y(i) \tag{17}$$

where $\oplus$ is the XOR operation, $X(i)$ and $Y(i)$ are 8-bit binary numbers defined by

$$\begin{aligned} X(i) &= T\lceil \overline{x_i} \rfloor_{k_1:(k_1+7)} \\ Y(i) &= T\lceil \overline{y_i} \rfloor_{k_2:(k_2+7)} \end{aligned} \tag{18}$$

where $T\lceil m \rfloor_{k_1:(k_1+7)}$ is a function to truncate the binary bit stream m from the bit location k_1 to location $(k_1 + 7)$. $\overline{x_i}$ and $\overline{y_i}$ are 52-bit binary streams converted by outputs x_i and y_i with the IEEE 754 standard [25]. k_1 and k_2 are two integers defined by

$$\begin{aligned} k_1 &= (x_i \times 10^{10} \bmod 6) + 39 \\ k_2 &= (y_i \times 10^{10} \bmod 6) + 39. \end{aligned} \tag{19}$$

In each iteration, the 8-bit binary sequence is generated by TLPRNG.

In TLPRNG, we use two outputs of the Tent-Logistic map with different initial values and parameters to generate pseudo-random numbers. As can be seen in Fig. 8, using two outputs from different chaotic orbits to generate pseudo-random numbers can ensure that TLPRNG is able to

TABLE II
RANDOMNESS TESTS OF TLPRNG USING NIST SP800-22

	P-$value$	Pass rate	P-$value_T$
Sub-tests	⩾0.01	⩾0.9602	⩾ 0.0001
Frequency	0.1538	0.9800	0.2145
Block freq.	0.7792	0.9900	0.8541
Cumulative*	0.5087	0.9800	0.5055
Runs	0.4944	1.0000	0.6200
Longest run	0.5141	0.9900	0.6390
Rank	0.8832	0.9900	0.9217
FFT	0.0428	0.9800	0.0512
Nonoverlop.*	0.4974	0.9838	0.5740
Overlap.	0.2493	1.0000	0.3445
Universal	0.1917	1.0000	0.2677
Apen	0.4373	0.9900	0.5627
Random e.*	0.4506	0.9889	0.0028
Random e.v.*	0.2544	0.9901	0.0011
Serial*	0.2523	0.9850	0.2335
Linear Comp.	0.1154	0.9800	0.1587
Success Counts	15/15	15/15	15/15

*the average values of multiple tests.

generate pseudo-random numbers with sufficiently large sizes and excellent randomness.

B. Randomness Analysis

Here, two PRNG test standards, NIST SP800-22 [26] and TestU01 [27], are used to evaluate the randomness of TLPRNG.

1) NIST SP800-22 Test: The NIST statistical test suite, SP 800-22, is a software package with 15 sub-tests to comprehensively evaluate the randomness of binary sequences generated by PRNGs. These sub-tests are designed to find the nonrandomness areas in all sides within a test sequence. The bit length of the testing binary sequence is suggested to be 10^6.

The significance level in NIST SP800-22 is preset as $\alpha = 0.01$. Because the sample size of binary sequences in each test should be not less than α^{-1}, we use 100 binary sequences with length of 10^6 bits generated by TLPRNG. Each sub-test will generate a P-value. Empirical results can be interpreted in three ways: 1) P-value; 2) pass rate; and 3) P-value$_T$.

P-value is to check whether the P-values generated by sub-tests are in the range of [α, 1], [0.01, 1] in our experiments. The pass rate indicates the pass rate of P-value. It takes the statistic proportion of sequences that pass the P-value test among all sequences involving the test. The minimum pass rate in our experiments is 0.9602, which can be calculated by the method in [26] with the test significance level and sample size. P-value$_T$ is a statistic of the P-value distribution. If P-value$_T \geq 0.0001$, sequences in the test are uniformly distributed and pass the corresponding sub-test.

Table II shows the results of each sub-test. As can be seen, the binary sequences generated by TLPRNG pass all tests. Thus, these binary sequences are truly random. Table III compares the P-value$_T$ results of TLPRNG with different PRNGs in [28] and [29]. Among four PRNGs, TLPRNG and Old CI can pass all 15 P-value$_T$ sub-tests.

TABLE III
P-value$_T$ RESULTS OF DIFFERENT PRNGS

PRNGs	New CI [28] ($m^n = y^n \ mod$ N)	New CI [28] (no mark)	Old CI [29]	TLPRNG
Success counts	8/15	11/15	15/15	15/15

TABLE IV
TESTU01 TEST OF BINARY SEQUENCES GENERATED BY TLPRNG

Lengths	$Rabbit$ test	$Alphabit$ test	$BlockAlphabit$ test
2^{20} bits	38/38	17/17	17/17
2^{30} bits	37/38	17/17	17/17

2) TestU01 Test: TestU01 is a software library for empirical statistic tests of PRNGs [27]. To test TLPRNG, we use the software package of TestU01-1.2.3 with three binary sequence test batteries: *Rabbit*, *Alphabit*, and *BlockAlphabit*. The *Rabbit*, *Alphabit*, and *BlockAlphabit* tests contain 38, 17, and 17 statistic sub-tests, respectively. Each statistic test generates a P-value, which is expected in range of [0.001, 0.999] to pass the test.

We use TLPRNG to generate two binary sequences with length of 2^{20} and 2^{30} bits, and then they are applied with the TestU01 test. The results are shown in Table IV. As can be seen, two binary sequences pass all sub-tests except for only one failed case in the *Rabbit* test. This means that TLPRNG is able to generate various lengths of binary sequences with excellent randomness.

VII. PROPOSED DATA ENCRYPTION SYSTEM

As a straightforward data security technology, data encryption attracts increasing attentions. It transforms data into a meaningless data format. In the past few decades, many data encryption technologies were developed. Examples include the digital encryption standard (DES), advanced encryption standard (AES), networked data encryption [30], and many other encryption algorithms [7], [31]. Due to the properties of sensitivity to parameters and initial values, ergodicity, and unpredictability, chaotic maps are good tools for data encryption. Chaotic maps with excellent chaotic behaviors have security benefits to data encryption. Because the proposed CCS has good chaotic performance, it is suitable for data encryption.

In this section, using the Tent-Logistic map as an example of CCS, we introduce a new Tent-Logistic map-based data encryption algorithm (TL-DEA). Many existing DEAs are designed to encrypt data in the binary format such as DES and AES. Data with other formats should be transformed into the binary format before encryption. This may be inefficient for some data with a large size such as high resolution images/videos. But TL-DEA can directly encrypt different types of data. Simulations and security analysis are provided to demonstrate its encryption performance.

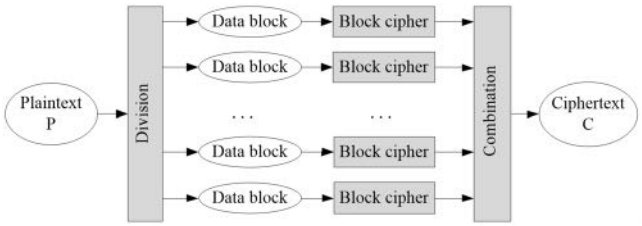

Fig. 9. Proposed TL-DEA.

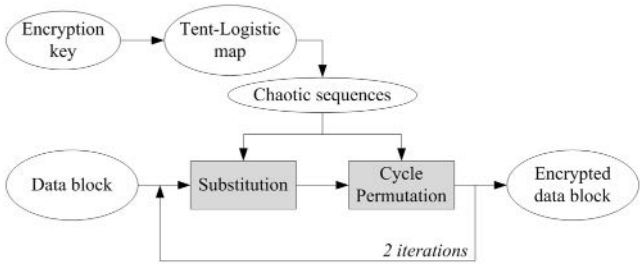

Fig. 10. Block diagram of the block cipher.

Algorithm 1 Generation of Initial Values and Parameters

Input: Security key K with length of 256 bits

Initial value $x_0 \leftarrow (\sum_{i=1}^{52} K_i 2^{52-i})/2^{52}$
Parameter $u \leftarrow (\sum_{i=53}^{104} K_i 2^{104-i})/2^{52}$
Parameter $a \leftarrow (\sum_{i=105}^{156} K_i 2^{156-i})/2^{52}$
$T \leftarrow (\sum_{i=157}^{208} K_i 2^{208-i})/2^{52}$
$R_1 \leftarrow \sum_{i=209}^{232} K_i 2^{232-i}$
$R_2 \leftarrow \sum_{i=233}^{256} K_i 2^{256-i}$
for $i = 1$ to 2 **do**
  $x_{0i} \leftarrow (x_0 + R_i T)\ mod\ 1$
  $u_i \leftarrow 1.8 + (u + R_i T)\ mod\ 0.2$
  $a_i \leftarrow 3.8 + (a + R_i T)\ mod\ 0.2$
end for

Output: Initial conditions (x_{01}, u_1, a_1) and (x_{02}, u_2, a_2).

A. TL-DEA

The block diagram of the proposed TL-DEA is shown in Fig. 9. The plaintext P denotes the original data while the cipertext C means the encrypted data. The division operation is to divide the plaintext into many data blocks with a fixed length. The block cipher is then used to encrypt each data block individually. The combination operation is to combine all encrypted data blocks into a encrypted data sequence to obtain the ciphertext.

The block cipher is shown in Fig. 10. The encryption key is to provide initial conditions of the Tent-Logistic map. Two rounds of substitution and permutation processes in TL-DEA are to guarantee good confusion and diffusion properties.

1) Key Analysis: The security key in TL-DEA is with length of 256 bits. It is used to produce two groups of initial values and parameters as described in Algorithm 1. The Tent-Logistic map then uses them to generate two different chaotic sequences.

Algorithm 2 Cycle Permutation

Input: Data block H and chaotic sequence S. Both are with length of L
Rearrange H, S with size of $M \times N$, where $L = M \times N$.
Sort each row of S and get the row index matrix I. Then $Sorted\_S_{m,n} = S_{m,\ I_{m,n}}$, where $m, n \in [1, M] \times [1, N]$
for $j = 1$ to N **do**
  for $i = 1$ to M **do**
    Find value j in ith row of I, get its position (i, j_i).
  end for
  Connect values of H in positions $(1, j_1)$, $(2, j_2)$, …, (M, j_M) into a circle, and shift them by j positions to upper direction.
end for
Rearrange the permutation result into length of L

Output: The permuted result C.

2) Substitution: The substitution process is designed to change the data values in the plaintext using its two previous neighboring data values and a random value from the chaotic sequence.

Suppose a data block P is with length of L, a chaotic sequence S with length of L is generated by the Tent-Logistic map, $S = \{x_1, x_2, \ldots, x_L\}$. Connecting each data with its previous one and linking the first data with the last one are to make the data block as a circle. Then the substitution process for each data block is defined as

$$H_i = \begin{cases} (P_i + P_L + P_{L-1} + \lfloor S \times 2^{20} \rfloor_i) \bmod F & \text{if } i = 1 \\ (P_i + C_{i-1} + P_L + \lfloor S \times 2^{20} \rfloor_i) \bmod F & \text{if } i = 2 \\ (P_i + C_{i-1} + C_{i-2} + \lfloor S \times 2^{20} \rfloor_i) \bmod F & \text{if } i \in [3, L] \end{cases} \tag{20}$$

where F is the number of allowed intensity scales in the plaintext. For example, $F = 2$ if the plaintext contains only binary data and $F = 256$ if the plaintext is represented as 8-bit decimals. $\lfloor . \rfloor$ is the floor operation.

3) Cycle Permutation: The cycle permutation is to shuffle all data positions, as shown in Algorithm 2.

For example, suppose the row index matrix I is as follows:

$$I = \begin{vmatrix} 2\ 1\ 4\ 3 \\ 1\ 3\ 2\ 4 \\ 3\ 2\ 4\ 1 \\ 1\ 4\ 3\ 2 \end{vmatrix}.$$

Fig. 11 shows the detailed operations using the index matrix I. First, we search the index value 1 in all rows in I, and get positions (1, 2), (2, 1), (3, 4), and (4, 1), then we connect the data in these positions in the data block H into a circle and shift them by 1 position to the upper direction. Second, we search the index value 2 in I, and get positions (1, 1), (2, 3), (3, 2), and (4, 4), connect the data in these positions in H into a circle, and then shift them by 2 positions to the upper direction. Repeating the same procedures until the maximum index value in I. After one cycle permutation, the data can be separated from all its neighboring data.

Repeating the substitution and circle permutation one more time with another chaotic sequence, the encrypted data block is obtained.

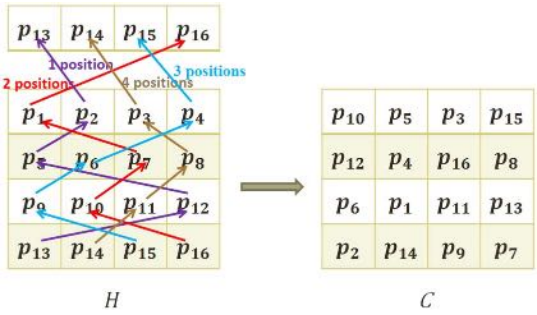

Fig. 11. Example of cycle permutation.

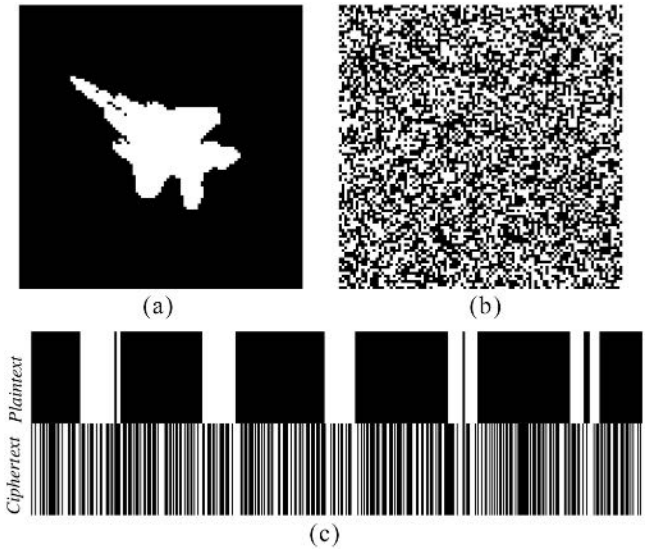

Fig. 12. Encryption results of binary data. (a) Plaintext. (b) Ciphertext. (c) Segmented data sequences from the plaintext and cipertext.

B. Simulation Results

A good cryptography algorithm should be able to encrypt different types of plaintext into the noise-like ciphertext. In our experiments, the binary data and 8-bit decimal data (e.g., images) are used as the plaintext to test the encryption performance of the proposed TL-DEA. The simulation is done with MATLAB R2010a in Windows 7 operating system.

To encrypt binary data, we use a binary image as an example. Because a binary image is represented as a 2-D matrix, it can be treated as a data block and applied with the block cipher directly. The encryption results are shown in Fig. 12. We can see that the binary data 0 and 1 in the ciphertext distribute randomly in all positions. There is no information about the original data.

TL-DEA can also encrypt data with other formats such as digital images and videos. Their pixels are usually represented by 8 or more bits. TL-DEA can directly encrypt them in the pixel level, which is more efficient and convenient than those in the bit level. Fig. 13 shows the encryption results of digital images. As can be seen, the encrypted images are visually noise-like with uniform data distributions. The original data are protected with a high level of security.

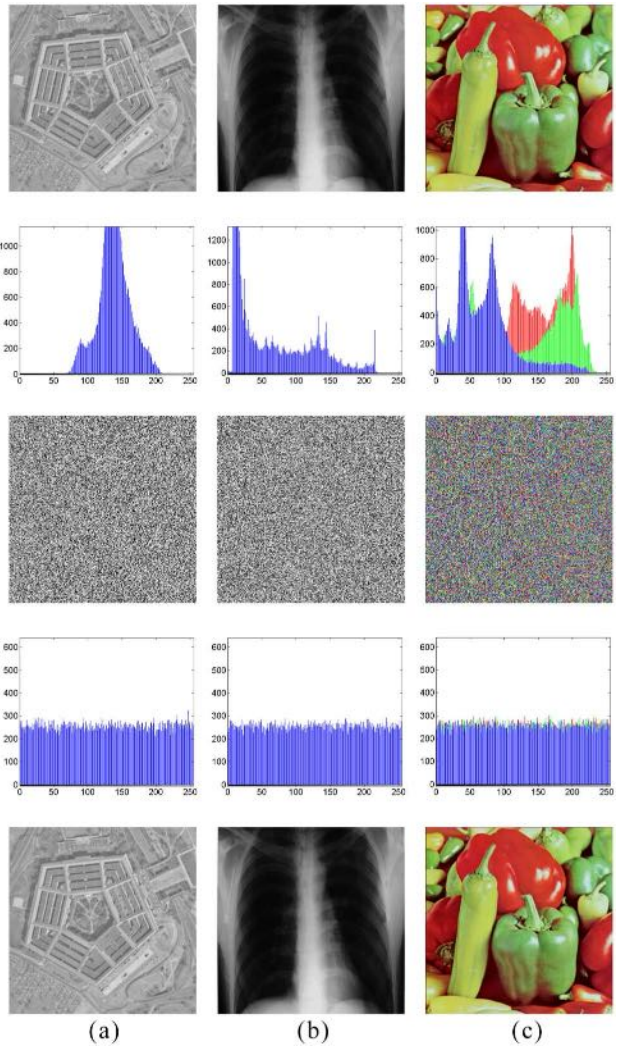

Fig. 13. Encryption of different images. (a) Grayscale image. (b) Medical image. (c) Color image.

C. Security Analysis

Security is the most important property of a cryptography system. A good cryptography system should have the ability to resist different well-known attacks.

To show the security performance of the proposed TL-DEA, we use digital images represented by 8 bits as examples to perform security analysis including the key sensitivity test, differential attack analysis, as well as noise, and data loss attacks. Most test images are selected from the USC-SIPI "Miscellaneous" image database.

1) Key Sensitivity Test: An encryption algorithm should be sensitive to its security keys. The key sensitivity can be tested in the encryption and decryption processes: 1) encryption key sensitivity, which means that a slight change in the encryption keys will yield a completely different ciphertext and 2) decryption key sensitivity, which means that the original plaintext can be recovered only when the correct security keys are being utilized, and that a light change of security keys will result in an unrecognized decryption result.

The key sensitivity analysis results are shown in Fig. 14. K_2 and K_3 are two different security keys which are generated

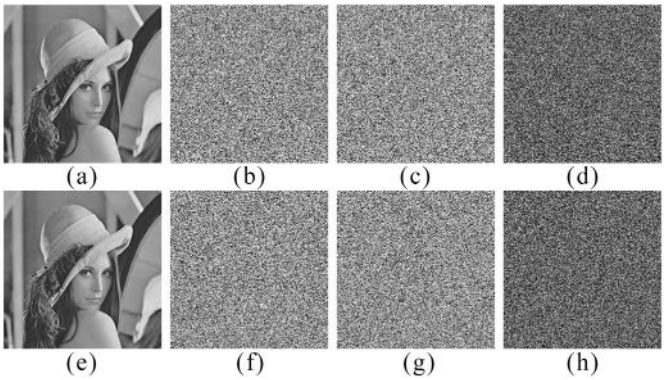

Fig. 14. Key sensitivity analysis. (a) Plaintext image P. (b) Cipertext image C_1 with K_1. (c) Cipertext image C_2 with K_2. (d) Difference between cipertext images, $|C_1 - C_2|$. (e) Decrypted image D_1 from C_1 with K_1. (f) Decrypted image D_2 from C_1 with K_2. (g) Decrypted image D_3 from C_1 with K_3. (h) Difference between decrypted images, $|D_2 - D_3|$.

from the security key K_1 with one bit change. As can be seen, when a plaintext image P [Fig. 14(a)] is encrypted using K_2 and K_3 with only one bit difference, we obtain two totally different encrypted results as shown in Fig. 14(b) and (c). Fig. 14(d) shows their differences. On the other hand, when a cipertext image [Fig. 14(b)] is decrypted by two security keys with one bit difference, we will also obtain two totally different decrypted results as shown in Fig. 14(f) and (g). Only the correct security key can reconstruct the original plaintext as shown in Fig. 14(e). Therefore, the proposed TL-DEA is sensitive to its security keys in the encryption and decryption processes.

2) Differential Attack Analysis: A cryptography system with an excellent diffusion property can resist differential attacks. To quantitatively evaluate the diffusion property of TL-DEA, we use the number of pixel change rate (NPCR) and unified averaged changed intensity (UACI). Mathematically, the NPCR and UACI of two images C_1 and C_2 are defined as

$$NPCR(C_1, C_2) = \sum_{i=1}^{M}\sum_{j=1}^{N}\frac{D(i,j)}{L} \times 100\% \tag{21}$$

$$UACI(C_1, C_2) = \sum_{i=1}^{M}\sum_{j=1}^{N}\frac{|C_1(i,j) - C_2(i,j)|}{T \times L} \times 100\% \tag{22}$$

$$D(i,j) = \begin{cases} 0, & \text{if } C_1(i,j) = C_2(i,j) \\ 1, & \text{if } C_1(i,j) \neq C_2(i,j) \end{cases} \tag{23}$$

where C_1 and C_2 are two encrypted images that are generated from two plaintext images with one pixel difference, T denotes the largest allowed pixel intensity and L denotes the total number of pixels in the image. NPCR measures the percentage of different pixels between two encrypted images while UACI tests the changed intensities.

The plaintext images are from the USC-SIPI Miscellaneous image dataset. For each test image, it is set one pixel to zero to generate a new test image, and then TL-DEA with the same security key is applied to both images. The two encrypted results are then measured by NPCR and UACI. Table V shows the measure results. As can be seen, the average values of

TABLE V
NPCR AND UACI RESULTS OF TL-DEA WITH THE PLAINTEXT IMAGES FROM THE USC-SIPI IMAGE DATASET

File Name	NPCR %	UACI %	File Name	NPCR %	UACI %
5.1.09.tiff	99.6002	33.4808	7.1.05.tiff	99.5945	33.3839
5.1.10.tiff	99.6094	33.5094	7.1.06.tiff	99.6044	33.5048
5.1.11.tiff	99.6140	33.3509	7.1.07.tiff	99.5922	33.4697
5.1.12.tiff	99.6277	33.4461	7.1.08.tiff	99.6052	33.4525
5.1.13.tiff	99.5880	33.3572	7.1.09.tiff	99.6113	33.4049
5.1.14.tiff	99.6689	33.3726	7.1.10.tiff	99.6151	33.4732
5.2.08.tiff	99.5972	33.4266	7.2.01.tiff	99.6171	33.4676
5.2.09.tiff	99.6040	33.4163	boat.512.tiff	99.6254	33.5182
5.2.10.tiff	99.6105	33.4703	elaine.512.tiff	99.6227	33.3764
5.3.01.tiff	99.6097	33.4312	gray21.512.tiff	99.6166	33.3966
5.3.02.tiff	99.6111	33.4747	house.tiff	99.6082	33.4581
7.1.01.tiff	99.6094	33.4403	numbers.512.tiff	99.5983	33.4245
7.1.02.tiff	99.59991	33.4458	ruler.512.tiff	99.5945	33.3957
7.1.03.tiff	99.5895	33.4568	Testpat.1k.tiff	99.6172	33.4547
7.1.04.tiff	99.6216	33.4550	**Mean**	**99.6098**	**33.4384**

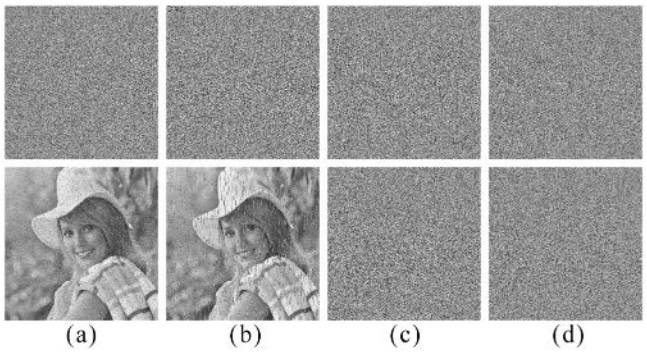

Fig. 15. Noise attack. The top and bottom rows show the encrypted images with 1% Salt&Pepper noise and reconstructed images by (a) proposed TL-DEA, (b) AES [34], (c) Liao *et al.*'s algorithm [35], and (d) Wu *et al.*'s algorithm [36], respectively.

NPCR and UACI are 99.6098% and 33.4384%, respectively. They are extremely close to the theoretically ideal values of NPCR and UACI (99.609% and 33.464%) proved in [32]. This demonstrates that TL-DEA has excellent diffusion property and is able to resist differential attack.

3) Noise and Data Loss Attacks: Almost all data transmission channels are noise channels [33]. When data are being transmitted over noise channels, they are easily contaminated by noise. There also exists data loss during data transmission and storage. Therefore, it is important for an encryption algorithm with ability to withstand noise and data loss attacks.

To test the performance in against the noise and data loss attacks, we compare TL-DEA with three existing encryption algorithms: the AES [34], Liao *et al.*'s [35], and Wu *et al.*'s [36] algorithms. We first encrypt a plaintext image

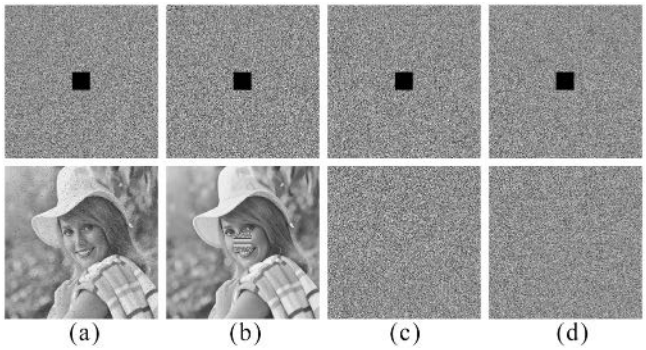

Fig. 16. Data loss attack. The top and bottom rows show the encrypted images with a square data loss and the reconstructed images by (a) proposed TL-DEA, (b) AES [34], (c) Liao *et al*.'s algorithm [35]; and (d) Wu *et al*.'s algorithm [36], respectively.

using these algorithms, apply the noise attack (adding 1% Salt&Pepper noise to the ciphertext images) or data loss attack (performing a data cutting with size of 60×60 to the cipertext images), and then reconstruct the original plaintext from the attacked images. Figs. 15 and 16 show the results of the noise and data loss attacks, respectively. As can seen, The Liao *et al.*'s and Wu *et al.*'s algorithms fail in both attacks. Their reconstructed images are completely unrecognized as shown in the bottom row in Fig. 15(c) and (d) and Fig. 16(c) and (d). TL-DEA and AES are able to reconstruct images as shown in the bottom row in Fig. 15(a) and (b) and Fig. 16(a) and (b). However, the reconstructed results of TL-DEA have much better visual quality than those of AES because the results of AES contain more noise in the noise attack and miss all visual information within the location of the cutting block in the data loss attack. The results of TL-DEA reconstruct all original information with the best visual quality. These demonstrate that TL-DEA outperforms these existing algorithms in resisting noise and data loss attacks.

VIII. Conclusion

This paper has proposed a new CCS. The system is able to generate a large number of different 1-D chaotic maps using a combination of any two existing 1-D chaotic maps. Three examples of NCMs generated by CCS have been introduced and analyzed. The evaluation and comparison results have shown that the newly generated chaotic maps are more unpredictable and have better chaotic performance, more parameters and more complex chaotic properties than existing chaotic maps.

To demonstrate how the proposed CCS can benefit the chaos-based applications, using the Tent-Logistic map as an example of NCMs of CCS, we have introduced TLPRNG and TL-DEA. The excellent randomness of TLPRNG has been demonstrated by the test results of NIST SP800-22 and TestU01. We have also evaluated the performance of TL-DEA with respect to data encryption and security analysis. The results have shown that TL-DEA is able to protect different types of data with a high level of security and to withstand differential attack, as well as noise and data loss attacks. Our future work will extend the proposed CCS into HD chaotic systems to generate hyperchaotic maps.

Acknowledgment

The authors would like to thank the anonymous reviewers for their valued comments which helped to improve this paper.

References

[1] E. Ott, *Chaos in Dynamical Systems*. New York, NY, USA: Cambridge Univ. Press, 2002.

[2] Q. Tao, Z. Sun, and K. Kong, "Developing learning algorithms via optimized discretization of continuous dynamical systems," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 42, no. 1, pp. 140–149, Feb. 2012.

[3] K.-Y. Lian, T.-S. Chiang, C.-S. Chiu, and P. Liu, "Synthesis of fuzzy model-based designs to synchronization and secure communications for chaotic systems," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 31, no. 1, pp. 66–83, Feb. 2001.

[4] S.-L. Chen, T. Hwang, and W.-W. Lin, "Randomness enhancement using digitalized modified logistic map," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 57, no. 12, pp. 996–1000, Dec. 2010.

[5] T. Addabbo *et al.*, "A class of maximum-period nonlinear congruential generators derived from the Rényi chaotic map," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 54, no. 4, pp. 816–828, Apr. 2007.

[6] R. Bose and S. Pathak, "A novel compression and encryption scheme using variable model arithmetic coding and coupled chaotic system," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 53, no. 4, pp. 848–857, Apr. 2006.

[7] K.-W. Wong, Q. Lin, and J. Chen, "Simultaneous arithmetic coding and encryption using chaotic maps," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 57, no. 2, pp. 146–150, Feb. 2010.

[8] Y. Zhou, L. Bao, and C. L. P. Chen, "A new 1D chaotic system for image encryption," *Signal Process.*, vol. 97, pp. 172–182, Apr. 2014.

[9] R. C. Hilborn, *Chaos and Nonlinear Dynamics: An Introduction for Scientists and Engineers*, 2nd ed. New York, NY, USA: Oxford Univ. Press, 2001.

[10] Y. Wu, Y. Zhou, and L. Bao, "Discrete wheel-switching chaotic system and applications," *IEEE Trans. Circuits Syst. I, Reg. Papers*, to be published.

[11] D. Arroyo, R. Rhouma, G. Alvarez, S. Li, and V. Fernandez, "On the security of a new image encryption scheme based on chaotic map lattices," *Chaos*, vol. 18, no. 3, Sep. 2008, Art. ID 033112.

[12] H. C. Papadopoulos and G. W. Wornell, "Maximum-likelihood estimation of a class of chaotic signals," *IEEE Trans. Inf. Theory*, vol. 41, no. 1, pp. 312–317, Jan. 1995.

[13] X. Wu, H. Hu, and B. Zhang, "Parameter estimation only from the symbolic sequences generated by chaos system," *Chaos Soliton. Fract.*, vol. 22, no. 2, pp. 359–366, Oct. 2004.

[14] G. Chen and X. Yu, *Chaos Control: Theory and Applications*, vol. 292. Berlin, Germany: Springer, 2003.

[15] H.-K. Chen and C.-I. Lee, "Anti-control of chaos in rigid body motion," *Chaos Soliton. Fract.*, vol. 21, no. 4, pp. 957–965, 2004.

[16] C. Shen, S. Yu, J. Lu, and G. Chen, "A systematic methodology for constructing hyperchaotic systems with multiple positive Lyapunov exponents and circuit implementation," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 61, no. 3, pp. 854–864, Mar. 2014.

[17] T. Gao and Z. Chen, "A new image encryption algorithm based on hyper-chaos," *Phys. Lett. A*, vol. 372, no. 4, pp. 394–400, Jan. 2008.

[18] Y. Zhou, L. Bao, and C. L. P. Chen, "Image encryption using a new parametric switching chaotic system," *Signal Process.*, vol. 93, no. 11, pp. 3039–3052, 2013.

[19] G. Jakimoski and K. P. Subbalakshmi, "Discrete Lyapunov exponent and differential cryptanalysis," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 54, no. 6, pp. 499–501, Jun. 2007.

[20] J. M. Amigo, L. Kocarev, and J. Szczepanski, "Discrete Lyapunov exponent and resistance to differential cryptanalysis," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 54, no. 10, pp. 882–886, Oct. 2007.

[21] A. A. Muchnik and S. Y. Positselsky, "Kolmogorov entropy in the context of computability theory," *Theor. Comput. Sci.*, vol. 271, no. 12, pp. 15–35, 2002.

[22] R. Frigg, "In what sense is the Kolmogorov–Sinai entropy a measure for chaotic behaviour? Bridging the gap between dynamical systems theory and communication theory," *Brit. J. Philos. Sci.*, vol. 55, no. 3, pp. 411–434, 2004.

[23] J. Gao, J. Hu, and W.-W. Tung, "Entropy measures for biological signal analyses," *Nonlinear Dyn.*, vol. 68, no. 3, pp. 431–444, 2012.

[24] C.-Y. Li, J.-S. Chen, and T.-Y. Chang, "A chaos-based pseudo random number generator using timing-based reseeding method," in *Proc. IEEE Int. Symp. Circuits Syst.*, pp. 3277–3280, Island of Kos, Greece, 2006.
[25] *IEEE Standard for Floating-Point Arithmetic*, IEEE Standard 754-2008, 2008, pp. 1–70.
[26] I. Lawrence *et al.*, "SP 800-22 Rev. 1a. A statistical test suite for random and pseudorandom number generators for cryptographic applications," Nat. Inst. Stand. Technol., Gaithersburg, MD, USA, Tech. NIST Rep. SP 800-22 , 2010.
[27] P. L'Ecuyer and R. Simard, "TestU01: A C library for empirical testing of random number generators," *ACM Trans. Math. Softw.*, vol. 33, no. 4, p. 22, 2007.
[28] J. M. Bahi, X. Fang, C. Guyeux, and Q. Wang, "Randomness quality of CI chaotic generators: Applications to internet security," in *Proc. 2nd Int. Conf. Evol. Internet (INTERNET)*, 2010, pp. 125–130.
[29] Q. Wang, C. Guyeux, and J. M. Bahi, "A novel pseudo-random number generator based on discrete chaotic iterations," in *Proc. 1st Int. Conf. Evol. Internet (INTERNET)*, 2009, pp. 71–76.
[30] J. Lu and G. Chen, "A time-varying complex dynamical network model and its controlled synchronization criteria," *IEEE Trans. Autom. Control*, vol. 50, no. 6, pp. 841–846, Jun. 2005.
[31] Y. Zhou, K. Panetta, S. Agaian, and C. L. P. Chen, "(n, k, p)-Gray code for image systems," *IEEE Trans. Cybern.*, vol. 43, no. 2, pp. 515–529, Apr. 2013.
[32] C. Fu *et al.*, "A chaos-based digital image encryption scheme with an improved diffusion strategy," *Opt. Express*, vol. 20, no. 3, pp. 2363–2378, 2012.
[33] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 3rd ed. Harlow, U.K.: Prentice-Hall Inc., 2007.
[34] *Advanced Encryption Standard (AES)*, FIPS PUB 197, 2001.
[35] X. Liao, S. Lai, and Q. Zhou, "A novel image encryption algorithm based on self-adaptive wave transmission," *Signal Process.*, vol. 90, no. 9, pp. 2714–2722, 2010.
[36] Y. Wu, G. Yang, H. Jin, and J. P. Noonan, "Image encryption using the two-dimensional logistic chaotic map," *J. Electron. Imaging*, vol. 21, no. 1, 2012, Art. ID 013014.

Yicong Zhou (M'07–SM'14) received the B.S. degree from Hunan University, Changsha, China, in 1992, and the M.S. and Ph.D. degrees from Tufts University, Medford, MA, USA, in 2008 and 2010, respectively, all in electrical engineering.

He is currently an Assistant Professor with the Department of Computer and Information Science, University of Macau, Macau, China. His current research interests include multimedia security, image/signal processing, pattern recognition, and medical imaging.

Zhongyun Hua (S'14) received the B.S. degree from Chongqing University, Chongqing, China, and the M.S. degree from the University of Macau, Macau, China, in 2011 and 2013, both in software engineering. He is currently pursuing the Ph.D. degree from the Department of Computer and Information Science, University of Macau.

His current research interests include chaos algebra, chaos-based applications, multimedia security, and signal/image processing.

Chi-Man Pun (M'09–SM'10) received the B.Sc. and M.Sc. degrees in software engineering from the University of Macau, Macau, China, in 1995 and 1998 respectively, and the Ph.D. degree in computer science and engineering from the Chinese University of Hong Kong, Hong Kong, in 2002.

He is currently an Associate Professor and the Head of the Department of Computer and Information Science, University of Macau. He has investigated several funded research projects and published over 120 refereed scientific papers in international journals, books, and conference proceedings. His current research interests include digital image processing, multimedia security and digital watermarking, pattern recognition and computer vision, and intelligent systems and applications.

Dr. Pun has served as the General Chair for the 10th and 11th International Conference Computer Graphics, Imaging and Visualization (CGIV2013 and CGIV2014), and a Program/Session Chair for several other international conferences. He has also served as an Editorial Member/Referee for several international journals such as the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, the IEEE TRANSACTIONS ON IMAGE PROCESSING, and *Pattern Recognition*.

C. L. Philip Chen (S'88–M'88–SM'94–F'07) received the Ph.D. degree from Purdue University, West Lafayette, IN, USA, and the M.S. degree from the University of Michigan, Ann Arbor, MI, USA, in 1988 and 1985, all in electrical engineering.

He was a Tenured Professor, the Department Head, and an Associate Dean in the Wright State University and University of Texas at San Antonio at U.S. for 23 years. He is currently the Dean of the Faculty of Science and Technology and a Chair Professor of the Department of Computer and Information Science, University of Macau, Macau, China.

Dr. Chen is a fellow of the American Association for the Advancement of Science and the Hong Kong Institution of Engineers. He is currently a Junior Past President of the IEEE Systems, Man, and Cybernetics Society and an Editor-in-Chief of the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS.