Ljupco Kocarev
Shiguo Lian (Eds.,)

Chaos-Based Cryptography

Theory, Algorithms and Applications

Springer

Ljupco Kocarev and Shiguo Lian (Eds.)

Chaos-Based Cryptography

Studies in Computational Intelligence, Volume 354

Editor-in-Chief

Ljupco Kocarev and Shiguo Lian (Eds.)

Chaos-Based Cryptography

Theory, Algorithms and Applications

Springer

Prof. Ljupco Kocarev
Macedonain Academy of Sciences and Arts
bul. Krste Misirkov 2, P.O. Box 428
1000 Skopje, Republic of Macedonia
E-mail: lkocarev@manu.edu.mk,
    lkocarev@ucsd.edu

Dr. Shiguo Lian
France Telecom R&D Beijing
2 Science Institute South Rd, Haidian District
Beijing, 100080, China
E-mail: shiguo.lian@orange-ftgroup.com

Preface

Chaos is an interesting phenomenon that often happens in some systems in various fields, e.g., physics, psychology, biology, etc. Chaos theory provides the means to explain chaos phenomenon, control chaotic dynamic systems and make use of chaos properties. Now, chaos has been used in physics, chemistry, neurophysiology, engineering, etc. Especially, chaos' properties, such as randomness and ergodicity, have been proved to be suitable for designing the means for data protection. During the past decade, many chaos-based cryptographic techniques have been studied, such as the chaos-based secret communication, chaos-based block/stream cipher, chaos-based random number generation, chaos-based hash, etc. Additionally, some secure applications based on chaos have been investigated, e.g., chaos-based image encryption or authentication, video/audio scrambling, multimedia copyright protection, etc.

To the best of our knowledge, this is the first book edited on chaos applications in cryptography. Chaos-based cryptography is a new research field across two fields, i.e., chaos (nonlinear dynamic system) and cryptography (computer and data security). To access the latest research related to chaos applications in cryptography, we launched the book project where researchers from all over the world provide the necessary coverage of the mentioned field. The primary objective of this project was to assemble as much research coverage as possible related to the field by defining the latest innovative technologies and providing the most comprehensive list of research references.

The book includes eleven chapters highlighting current concepts, issues and emerging technologies. Distinguished scholars from many prominent research institutions around the world contribute to the book. The book covers various aspects, including not only some fundamental knowledge and key techniques, but also typical applications and open issues. For example, the following topics are investigated in detail: fundamentals of chaos, relation between chaos and cryptography, Pseudo-Random Number Generation (PRNG) based on digitized chaos, cipher design based on high-dimensional chaotic maps, chaos-based hash function, and chaos-based video encryption. Additionally, the cryptanalysis of chaotic cipher and the corresponding lessons are presented in a thorough manner. Finally, some hardware implementations of chaotic ciphers and the performance evaluation compared with traditional ciphers are proposed in deep. For each of the topics, both the latest research results and open issues or hot topics are reviewed and analyzed.

The diverse and comprehensive coverage of multiple disciplines in the field of chaos based cryptography will contribute to a better understanding of all topics, research, and discoveries in this emerging and evolving field. Furthermore, the contributions included in this book will be instrumental in the expansion of the body of knowledge in this field. The coverage of this book provides strength to this reference resource for both researchers and also decision makers in obtaining a greater understanding of the concepts, issues, problems, trends, challenges and opportunities related to this field of study. It is our sincere hope that this publication and its great amount of information and research will assist our research colleagues, all faculties, their students, and our organizational decision makers in enhancing their understanding of this research field. Perhaps this publication will even inspire its readers to contribute to the current discoveries in this immense field.

Editors

Prof. Ljupco Kocarev
University of California, USA

Dr. Shiguo Lian
France Telecom R&D (Orange Labs) Beijing, China

Acknowledgments

Contents

Chapter 1
Introduction to Chaos

Dimitar Solev[1], Predrag Janjic[2], and Ljupco Kocarev[3]

[1] Macedonian Academy of Sciences and Arts, Skopje, Macedonia
   `dimitar.solev@gmail.com`
[2] Ericsson Telecommunications Macedonia, BS-Networks
   `pjanjic@netscape.net`
[3] Macedonian Academy of Sciences and Arts, Skopje, Macedonia,
   Department of Computer Science and Engineering, University "Kiril i Metodij" Skopje,
   Macedonia, BioCircuits Institute, University of California San Diego, USA
   `lkocarev@ucsd.edu`

1 Introduction

Chaos is a fascinating phenomenon that has been observed in nature (weather and climate, dynamics of satellites in the solar system, time evolution of the magnetic field of celestial bodies, and population growth in ecology) and laboratory (electrical circuits, lasers, chemical reactions, fluid dynamics, mechanical systems, and magneto-mechanical devices). Chaotic behavior has also found numerous applications in electrical and communication engineering, information and communication technologies, biology and medicine. This was mainly due to the wideband character of the chaotic signals, easy experimental control of chaos and all that being achieved with an inexpensive lab realization of either the electric circuits or corresponding algorithms if only number series were in focus. Communication and signal processing applications of chaos, as areas of permanent interest, were roughly established since 1990, after the theories of chaos synchronization and chaos control were worked out in more details. Today, sound engineering applications of quasi random sequence generation, modeling of communication channels using chaos, chaotic cryptography, digital image encoding, and chaotic transport phenomena in complex networks all represent areas of permanent research with commercially viable engineering solutions [Kocarev et al 2009].

Here we will try to introduce the reader to the concepts and very basic theory of *non-linear dynamics and chaos* so that some of the material to read further is easier to grasp and understand. Our main guiding assumption is that applications in the later chapters use or reflect to *complex, sometimes chaotic sequences* generated most often by *discrete dynamical system*.

Without loss of generality or missing anything important we will limit to discrete dynamical systems, as those are easier to relate to signal processing theory and the mathematical formalism used there. Where necessary the distinction will be made with *continuous dynamical systems*. To keep the text within a reasonable length, we are putting focus on understanding the basic features of complex dynamics and

mechanisms along which changes of the parameters and initial conditions in simple systems take predictive solutions and simple oscillations into very complex ones.

2 Dynamical Systems

2.1 Basic Concepts

We approach a model or experimental setup by defining or selecting an *observable*. An observable is generated by a *dynamical system* - set of equations describing the dynamical evolution of the quantities or observables we are trying to model and different dynamical systems can generate the same observable. Mathematicians often insist on lowest dimensionality of the system when defining its *generic* dynamical properties. As an example, we will try to find out which one-dimensional dynamical law is realised with the observable i.e. function $f(x) = e^{\lambda x}$ (in general, it can be any N-dimensional law). For $x \in \{1, 2, ...\}$ we have

$$f(1) = e^{\lambda}, \, f(2) = e^{2\lambda}, \, \text{etc} \tag{1}$$

We choose two adjacent values for x, K and $K+1$, so we can write:

$$\frac{f(K+1)}{f(K)} = e^{\lambda} \rightarrow \lambda = ln\frac{f(K+1)}{f(K)} \tag{2}$$

where $K = x \in [1, 2, \ldots]$ Thus we can reconstruct the dynamical law $\dot{x} = \lambda x$ that generated the function. Now, we can explain what the relation $\dot{x} = F(x)$ actually means. For a continuous observable, or continually changing phenomenon, a *dynamical system* is a set of coupled ordinary differential equations which determine how the state of a system evolves over time. When time is integer-valued i.e. we observe the system in discrete time, a dynamical system's evolution is governed by a set of difference equations. A continuous dynamical system can be described as the set of first-order differential equations:

$$\dot{\mathbf{x}}(t) = \mathbf{F}(\mathbf{x}(t)), \tag{3}$$

where $\dot{x} = \frac{d}{dt}\mathbf{x}(t)$, $x \in \mathbb{R}$. The mapping $\mathbf{F} : \mathbb{R}^n \rightarrow \mathbb{R}^n$ governs the evolution of the system and it is called a *vector field*. This vector field is such that at each point $\mathbf{x}_s(t)$ the vector $\mathbf{F}(\mathbf{x}_s(t))$ is tangent to the curve of the solution $\mathbf{x}_s(t)$. Simply put, equation (1) desribes an N-dimensional *flow*. If we were to define a *map* (where time is integer-valued), equation (1) becomes:

$$\mathbf{x}_{n+1} = \mathbf{F}(\mathbf{x}_n), \tag{4}$$

where $n \in \mathbb{Z}$ or $n \in \mathbb{Z}^+$. The N-dimensional vector $\mathbf{x}(t) \in \mathbb{R}^n$ represents the *state of the system* and the constituents of $\mathbf{x}(t) = (x_1, x_2, \ldots, x_m)$ are called *state variables*. Usually, $\mathbf{F}$ depends on a set of parameters $\mathbf{p} = (p_1, p_2, ..., p_k)$, $\mathbf{p} \in \mathbb{C}^k$, $\mathbb{C}^k \subseteq \mathbb{R}^n$ but most of the time there is no need to explicitly state this dependence. The space determined by $\mathbf{x}$ is called *state space* or *phase space* and is considered to be Euclidian space, but generally it could be an N-dimensional manifold.

The state of the system $\mathbf{x}(0)$ or x_0 when $t = 0$ is called the *initial conditions* and the set of all points starting from this state is called a *trajectory* or an *orbit*. Note that there are differences between orbits of a map and those of a flow. For a flow, the orbit is a continuous curve, but for a map, the orbit is a set of disconnected points, like a set of consecutive stroboscopic snapshots of an orbit of a flow with the same evolution rule (see Fig. 1.1).

The system we encountered earlier on: $\dot{x}(t) = \lambda x$ is a classical example of a *linear system*. Solving $dx/dt = \lambda x$, gives $x(t) = x_0 e^{\lambda t}$ and explains why we used the exponential function as an example. We are more interested, however, in studying *nonlinear* systems simply because most systems we observe, or try to model are in fact nonlinear: fluid dynamics, neural dynamics, general relativity etc.

The keen reader would have already noticed that we did not say whether $\mathbf{F}$ changes over time. So, we define an *autonomous dynamical system* as a system where $\mathbf{F}(\mathbf{x})$ does not explicitly depend on time. If $\mathbf{F}$ is indeed non-autonomous, such that $\mathbf{F}(\mathbf{x}, t)$, the system should be regarded as a second-order system, and its analysis is beyond the scope of this chapter.

Poincaré maps give us the ability to analyze N-dimensional flow using associated $N - 1$ dimensional map. $N - 1$ dimensional Poincaré map of a system is an invertible discrete mapping between the successive points on the *surface of section* obtained using $N - 1$ dimensional hyperplane to intersect the N-dimensional flow in $\mathbb{R}^n$. In other words, the Poincaré map maps the $N - 1$ coordinates of the n-th crossing, to those of the $(n - 1)$-th crossing of the flow of the continuous system. Assuming we can reconstruct the map or approximate it with a known one, from analysis point of view, we benefit from:

- simplicity of mostly algebraic manipulations of the map
- dealing with system of lower dimension
- invertible map, if we need to iterate backward in time

As a simple example, see (Fig. 1).
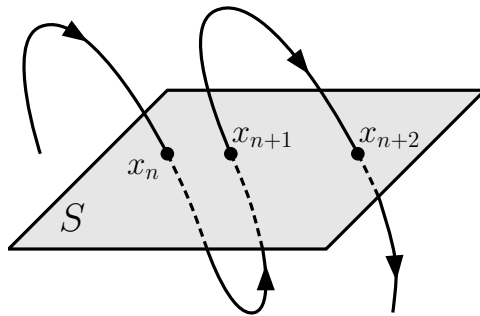
Fig. 1 The curve represents the orbit of an arbitrary flow and the intersection points are the 'stroboscopic captures' of the orbit of the flow, giving us the orbit of the map. The Poincaré surface of section (denoted S) in this example is a 2-dim plane that intersects the arbitrary 3-dim flow. Thus from a N dimensional flow, we get a N-1 dimensional map.

An interesting and useful property of Poincaré maps is that the characteristic multipliers of the map, corresponding either to a fixed point or a periodic orbit of the N-dimensional flow, does not depend on the selection of the surface of section S, or the local coordinates on it.

2.2 Complex Behaviour in One-Dimensional Maps

One-dimensional discrete systems have been quite extensively studied because they are the simplest systems capable of complex and chaotic behaviour. The usual notation for a one-dimensional map is:

$$x_{n+1} = f(x_n) \tag{5}$$

where $f : X \to X$ is the mapping from the state space to itself. Since a one-dimensional map is a discrete system we can say that it can be *iterated*. This means that for a seed state x_0, simply by applying the evolution rule over and over again (i.e. iterating) we can obtain every other future state of the system that follows from those specific *initial conditions*: $x_1 = f(x_0), x_2 = f(x_1) = f(f(x_0)), \ldots, x_n = f^n(x_0)$, where f^n is the N-fold iteration of the map or equivalently the composition:

$$f^n = \underbrace{f \circ f \circ \cdots \circ f}_{n \text{ times}} \tag{6}$$

As an example, entering a value on the calculator and pressing the sine button over and over again represents the iteration of the map $x_{n+1} = sin(x_n)$. The results from the n-th iteration, become the input for the n+1-th iteration and so on and so forth (a dynamical system is just like a recursive function in programming).

Before we can precisely define what an orbit of a map is, it is important that we define the notions of *homeomorphisms* and *diffeomorphisms* (which also hold for dynamical systems in general). For a mapping $M : X \to Y$ to be a homeomorphism it must be a bijection (one-to-one and onto), be continuous and also M^{-1} must be continuous. Similarly, we say that $M : X \to Y$ is a diffeomorphism if M is a homeomorphism and differentiable and the inverse M^{-1} is also differentiable. One interesting remark is that if there exists a diffeomorphism between two n-dimensional dynamical systems then they are equivalent.

Now for a given map $x_{n+1} = f(x_n)$, we can define the forward orbit $\mathscr{O}^+(x)$ of the state x as the set of points: $\mathscr{O}^+(x) = \{x, f(x), f^2(x), \ldots\} = f^n(x), n \in \mathbb{Z}^+$. If f is a homeomorphism, we may define the full orbit of x, $\mathscr{O}(x)$, as the set of points $f^n(x)$ for $n \in \mathbb{Z}$, and the backward orbit $\mathscr{O}^-$, as the set of points $x, f^{-1}(x), f^{-2}(x), \ldots$ The reason we distinguished between these types of orbits is that when we study maps, we can not always follow the backward orbits i.e. go backwards in time. *Noninvertible maps* are an example of such systems and in the next section we provide a thorough discussion of the issue of *invertibility* and how it is related to chaos.

One extremely convenient way of depicting orbits in discrete iterated maps is by *cobwebbing*. This graphical technique consists of overlaying the plot $y = x$ over

the plot of the map $x_{n+1} = f(x_n)$. Beginning from some initial value x_0 we draw a vertical line to the graph of the map (the parabola in our case) and from this point we draw a horizontal line to the graph of $y = x$. Thus we obtain the results from the first iteration of the map and the starting point for the next one. We repeat the same procedure for as many times as we need. For an example, see figure 1.2(a).

2.3   Example: The Logistic and the Hénon maps

One of the most studied examples of a one-dimensional system capable of various dynamical regimes including chaos is the *logistic map*:

$$x_{n+1} = rx_n(1 - x_n) \tag{7}$$

where r is the control parameter. The logistic map represents nothing more than an idealized population model [May, 1976]. Crucial to the behavior of the map is the control parameter r and we will examine the qualitative changes in the map's dynamics by varying the value of r. Similarly to the logistic map, the Hénon map is probably the most popular example of an *invertible* two-dimensional map. The Henon map is the system:

$$\begin{cases} x_{n+1} = a - x_n^2 + y_n \\ y_{n+1} = bx_n \end{cases} \tag{8}$$

where a and b are dimensionless parameters. Both maps are invaluable to the study of chaotic motion and we will rely on them in the subsequent sections.

2.4   Issue of Invertibility

We say that a map $f(x_n)$ is invertible if for a given state x_{n+1} there exists a unique pre-image x_n such that $x_n = f^{-1}(x_{n+1})$, where f^{-1} is the inverse of f. The logistic map is quite clearly non-invertible because there exist two pre-images x_n for any arbitrary x_{n+1} (except for the critical point at $x = 0.5$). From the relation $x_{n+1} = rx_n(1 - x_n)$ for x_n we obtain:

$$x_n = \frac{r \pm \sqrt{r^2 - 4rx_{n+1}}}{2r} \tag{9}$$

which means that we have two preimages for x_{n+1} thus we can confirm our statement about the non-invertibility of the logistic map. For comparison lets have another look of the Henon map we saw earlier:

$$\begin{cases} x_{n+1} = a - x_n^2 + y_n \\ y_{n+1} = bx_n \end{cases} \tag{10}$$
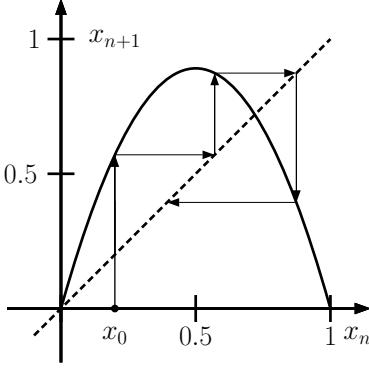
(a) Cobweb of the logistic map for $r = 3.565$ and initial conditions $x_0 = 0.2$
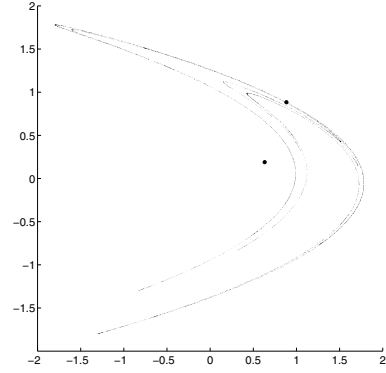
(b) Depiction of the chaotic regime in the Hénon map with $a = 1.4$ and $b = 0.3$

Fig. 2 Figure (a) depicts a high period orbit for the logistic map and the map is still in the periodic regime. The first three iterations i.e. how the cobweb is created, are marked with arrows. On Figure (b) is depicted the chaotic Hénon map. Both are examples of complex orbits, with the only difference that the logistic map depicted here is en-route to chaos.

We make the simple substitution:

$$\begin{cases} x_n = f(x_n) - Jy_n \\ y_{n+1} = bx_n \end{cases} \tag{11}$$

where $f(x_n) = a - x_n^2$ and $J = -1$. Now we can express the pre-images as:

$$\begin{cases} x_n = \frac{y_{n+1}}{b} \\ y_n = J^{-1} \left[f\left(\frac{y_{n+1}}{b} \right) - x_{n+1} \right] \end{cases} \tag{12}$$

This shows that the Henon map is invertible. We will see later that in addition to invertibility, dimensionality of the map also puts a limit on the repertoire of solutions we can observe. From the logistic map we saw that the minimum dimensionality for non-invertible maps to be chaotic is $N = 1$, whereas the Henon map is an example that we need at least two degrees of freedom for invertible maps to be chaotic. For flows i.e. systems of differential equations $N \geq 3$ must hold for the system to be chaotic [Hirsch and Smale, 1974].

2.5 Fixed Points and Attracting Sets

A *fixed point* x_* of the map $x_{n+1} = f(x_n)$ is such that $f(x_*) = x_*$. If there exists $n > 0$ such that $f^n(x_*) = x_*$ and $f^k(x_*) \neq x_*$ where $0 \leq k \leq n$ then we say that the point x_* is a *periodic point* of period n. The smallest period for which a point is periodic is called the *prime period*. A period-n orbit is made up of n periodic points and if

the state of the system belongs to a periodic orbit, it will alternate between those periodic points in succession. There exists another type of orbits one that neither corresponds to the steady states (fixed points) of the system, nor are periodic. This type of chaotic, irregular orbits will be discussed in detail in the next section.

Finding the fixed points in a one-dimensional map is quite simple, for the logistic map we only need to solve $x = rx(1 - x)$, so the solutions are:

$$x_{1/2} = \frac{-(1 - r) \pm (1 - r)}{2r} \tag{13}$$

An alternative way of finding the fixed points of a one-dimensional map is to find the intersection points of its graph with the diagonal $y = x$. A central aspect of fixed points is their character: they can be *attracting* or *repelling*, which means that nearby orbits respectively converge to or diverge from them. For discrete iterated maps, attracting fixed points are the simplest examples of *attracting sets* or *attractors* (the next simplest being period-n orbits). An *attracting set* for a map is best described as a closed subset of the map's phase space, such that solutions for many different initial conditions converge/asymptote to it as time increase.

2.6   Stability of Fixed Points

In general terms, the stability of a fixed point depends on the derivative of the map at that fixed point. To see this, we inject a small change or perturbation to the fixed point $x_* = f(x_*)$ which we label δ_n. We want to compute the perturbation in the next iteration, so we have $\delta_{n+1} = f(x_* + \delta_n) - x_*$. By using Taylor expansion we get:

$$\delta_{n+1} = f(x_* + \delta_n) - x_* = f(x_*) + f'(x_*)\delta_n - x_* + O(\delta_n^2) \tag{14}$$

Because δ is sufficiently small, the term $O(\delta_n^2)$ does not influence the character of the stability so it is correct to approximate it to zero. Thus, the perturbation after n iterations is: $\delta_n \approx (\mu_*)^n \delta_0$, where μ_* is the multiplier of the fixed point:

$$\mu_* = f'(x_*) \tag{15}$$

For $|\mu_*| < 1$ (resp. $|\mu_*| > 1$) we say that the fixed point x_* is *stable* (resp. *unstable*) and we refer to μ as the multiplier of the fixed point. If $|\mu| \neq 1$ then we say that the fixed point is *hyperbolic*. On the other hand, for $\mu = 1$ we are unsure of the character of the fixed point and for this value of μ the system undergoes a change we will explain in details later on. The notion of hyperbolicity also applies for periodic points and is explained in the next section. For $\mu = 0$ we say that the fixed point is *superstable* since all the perturbations are dampened quicker than exponentially.

2.7   Stability of Periodic Orbits

Let us consider the period-p orbit $O(x_p)$ and we know that $x_i = f^p(x_i)$ holds for $i = 0, 1, \ldots, p - 1$ Again, using the same technique, we introduce a slight perturbation

to x_i and take that value as initial conditions for our analysis: $x_0 = x_i + \delta_0$. Because of these perturbations, the p-th iterate will differ from x_i for some δ_p so we have: $x_i + \delta_p = f^p(x_i + \delta_0)$. Again, we use Taylor expansion to obtain:

$$x_i + \delta_p = f^p(x_i + \delta_0) = f^p(x_i) + (f^p)'(x_i)\delta_0 + O(\delta_0^2) \qquad (16)$$

Using the chain rule of derivatives (because f^p is a composition of functions) and not taking into account the higher orders of δ_0 (because δ_0 is sufficiently small), we arrive at: $\delta_p = \lambda_p \delta_0$, where λ_p is:

$$\lambda_p = f'(x_0)f'(x_1)\dots f'(x_{p-1}) \qquad (17)$$

and it is equal for all points x_i, $i = 0, 1, \dots, p-1$ belonging to the period-p orbit. When we follow the perturbed point $x_i + \delta_p$ another p iterates around the orbit the result is $x_i + \lambda_p\delta_p = x_i + \lambda_p^2\delta_0$, so we can generalize it to: $\delta_{np} = \lambda_p^n\delta_0$. This equation quantifies the deviation as we go along the periodic orbit. For $|\lambda_p| > 1$ this deviation grows by a factor of λ_p for every circle around the orbit and the periodic orbit acts as a repeller, we move further away from it. When $|\lambda_p| < 1$ every time round the periodic orbit the deviation decreases i.e. the orbit acts as an attractor since all initial conditions in its neighbourhood asymptote to it. For $\lambda_p = 0$ we say that the orbit is super-stable, since there is neither converges to it, neither diverges from it. λ_p can be referred to as the *stability coefficient* or multiplier for the periodic orbit. Generalizing our discussion of hyperbolicty, we can safely say that a fixed point is a special case of a periodic point i.e its prime period is 1. If the multiplier $|\lambda_p| \neq 1$ for a point of period p, then we say that it is hyperbolic and if $|\lambda_p| = 1$ is satisfied, then that periodic point is non-hyperbolic and we cannot say for certain whether it is stable or not. When this latter condition is met, a system is at a 'turning point', which is followed by a change in the dynamics.
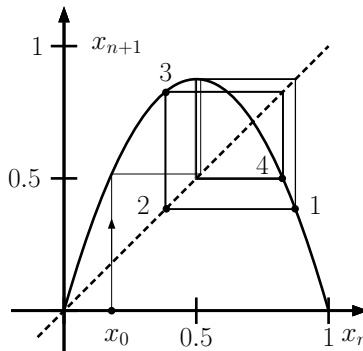
Fig. 3 Stable period-4 orbit of the logistic map, for initial conditions $x = 0.15$. An example of an asymptotic set, after the transients have died out, as the map is iterated further, the orbit will successively visit the points $1 \rightarrow 2 \rightarrow 3 \rightarrow 4$ which also can be marked on the diagonal $y = x$. Note that for any initial conditions, the asymptotic set would be the same, since it only depends on the control parameter r

2.8 Invariant Sets and Manifolds

All interesting phenomena in nonlinear dynamical systems occur on *invariant sets*, i.e. subsets of the phase space which remains as they are and constitute in a sense the structure of more complex attracting sets. So we attribute such invariant sets to a given attractor, either a fixed point, an orbit or a chaotic attractor.

   More precisely, we can define an invariant set of the dynamical system $x_{n+1} = f(x_n)$ as the subset S of the phase space X such that from $x_0 \in S$ follows $f^n(x_0) \in S$ for all n. With this formulation we actually say that the subset S will be mapped into itself for any arbitrary number of iterations $f^n S \subseteq S$. A logical conclusion is that any individual orbit $\mathcal{O}(x_0)$ is also an invariant set. Obviously, the simplest examples of invariant sets are fixed points.

   Manifolds represent special class of invariant sets with unique properties. The notion of a manifold can easily be explained simply by looking at the set of trajectories of neighbouring points. The *stable manifold* W^s of a point x_0 can be defined as an invariant set of points whose image after an arbitrary long time (i.e. infinite number of iterations) would be the same as x_0's:

$$W^s(x_0) = \{x : f^k(x) \to x_0, k \to +\infty\} \tag{18}$$

Similarly, an *unstable manifold* W_u of a point x_0 is an invariant set of points whose orbits converge to that of x_0 when we iterate backward in time i.e. x_0's and x's preimages had been the same before an infinite number of iterations:

$$W^u(x_0) = \{x : f^k(x) \to x_0, k \to -\infty\} \tag{19}$$

As the orbits follow closely geometry of the invariant sets, once getting close to them, one can realize the origin of the very irregular temporal dynamic of the
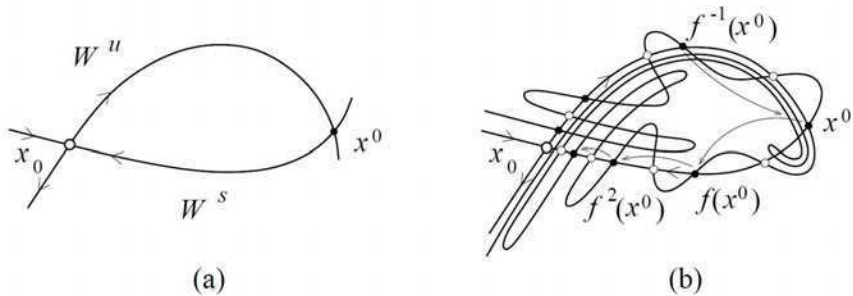
Fig. 4 (a)The manifolds W^u and W^s intersect at a point other than x_0, this is the homoclinic point x^0 (b) One homoclinic point implies that there exist infinite such points, i.e. intersections of the unstable manifold with the stable one. This is correlated to the folding action of a map: as we perform infinite iterations of the map $f^k(x^0) \to x_0$ as $k \to \pm\infty$, W^u is infinitely folded, creating an infinite number of intersections with W^s a result emerging from the complex dynamics of the system. Image taken from [Kuznetsov,1998]

solutions of the given system. The homoclinic structures are typical for parameter values for which the originally simple invariant sets has gone along certain scenario of structural changes.

A good deal of the theory of invariant sets and manifolds is common for both flows and maps, especially for hyperbolic equilibria and periodic orbits. However, not everything form the theory of smooth manifolds in differential dynamical systems is as such applicable in abstract sense to a manifold defined for a discrete map, or in certain cases the statements are valid for the close vicinities of the fixed points. For more detailed discussion of the the differences, in the context of complex dynamical behavior the reader can refer to [Gilmore 2002] [Kuznetsov, 1998].

3   Chaotic Behavior

3.1   *Main Features of Chaotic Dynamics*

Instead of going into strict mathematical definition of chaotic behaviour, we will discuss in more detail the main features and manifestations of chaos, as well as the transitions from more *regular* to chaotic solutions. Main features of a nonlinear dynamical system, exhibiting *deterministic chaos* for given values of the parameters, are the following:

- *Sensitive dependence on initial conditions* - where small changes in the initial values of variables grows in time, and produce unpredictable difference as we compute further, the orbit or path
- *Irregular motion in phase space* - illustrated by very complex, sometimes noise-like patterns of oscillations of the solutions within a bounded, compact sets. The particularity of chaos is that such complex oscillations are fully reproducible for same numeric precision in the initial conditions and parameter values. Such quasi-stochastic behavior can be qualified by the specific character of the associated *measures and invariant densities*
- *Qualitative change of the character of the solutions* - illustrated by a one or more subsequent *bifurcations*, structural changes of the phase set to which chaotic solutions converge as we evolve the system in time. Such attractors, i.e. *chaotic attractors* sometimes do not resemble at all the topological structure of other solutions, for example the period orbits. This is result of a global structural change of the phase space. Compact, simply connected subsets, along certain parameter ranges undergo series of non-smooth changes in their geometry and topology, mainly due to subsequent *stretching and folding* cascades. Such attractors are also called strange attractors, due to their specific geometry and self-similar structure on different time-scales.

Lets start with an example of chaotic orbit in logistic map. We will define necessary terms and concepts, related to chaos, as we come across them. As we mentioned earlier we are interested in seeing how orbits qualitatively change as the control

(a) $r = 3.565$

(b) $r = 4$
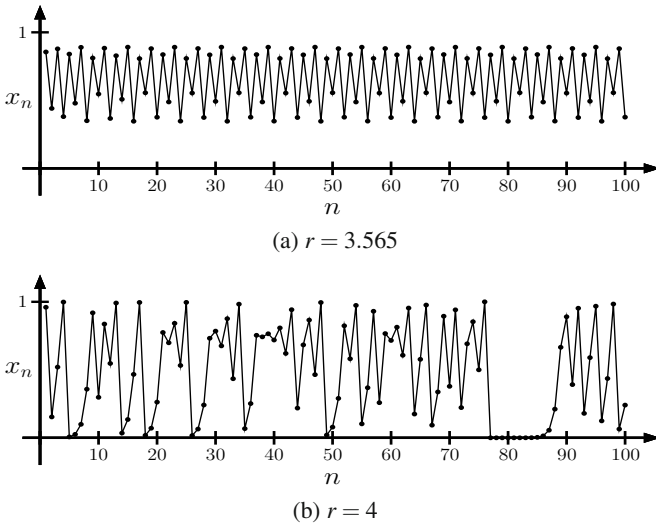
Fig. 5 Two distinct regimes for the logistic map are shown through 100 iterations of the logistic map (x_n, n), where n is the number of the iteration. The plots reveal the inherently different behaviors of the map. In the periodic regime, a repetition of the states i.e. periodicity can clearly be seen, whereas in the chaotic regime we witness irregular, random-like aperiodic motion

parameter r is varied and we will explain the concepts of bifurcations and chaotic motion. As we stated earlier, the logistic map has two fixed points:

$$x_1 = 0, \, x_2 = 1 - \frac{1}{r} \tag{20}$$

and the corresponding multipliers of these fixed points are:

$$\mu_1 = r, \, \mu_2 = 2 - r \tag{21}$$

Using the formulae for the multipliers we can determine how r affects the dynamics of the logistic map. For $r > 1$ we can clearly see that x_1 is unstable. On the other hand, for $1 < r < 3$ we have that x_2 is stable since $|2 - r| < 1$. So x_2 is an attracting fixed point in this range for r. It can easily be shown that for $r \in (1, 3)$, orbits with period $p \geq 2$ do not exist, and that any initial condition $0 \leq x_0 \leq 1$ converges to the attractor x_2, so we can say that the interval $x \in [0, 1]$ is the *basin of attraction* of x_2. What happens for $0 < r < 1$? Or with initial conditions $x_0 < 0$ or $x_0 > 1$? For $r \in (0, 1)$ we have that the fixed point $x = 0$ is stable ($|\mu_x = r| < 1$), so that any arbitrary initial conditions in the interval [0,1] will converge to it. For this value, the fixed point $x = 1 - r^{-1}$ is unstable. Also, if $r > 1$, and we have negative initial conditions $x_0 < 0$ or initial conditions such as $x_0 > 1$ then $M(x) < x$, so we conclude that such initial conditions generate orbits which tend to $-\infty$ as we go forward in time.

(a) Bifurcation diagram of the logistic map for $r \in [2.9, 4]$

(b) A zoomed in section of the chaotic region $r > 3.57$ from (a), exposing the self-similar structure of the diagram
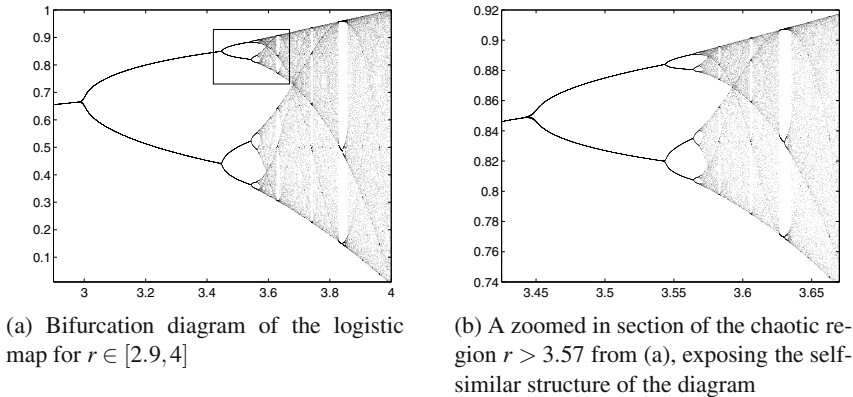
Fig. 6 The period-doubling bifurcation leads to chaos in the logistic map. From the diagram itself we can see how the map behaves for different values of r and the transition from periodic regime to chaos: sensitive dependence on initial conditions is manifested in mapping a small interval onto the whole domain of the map, coupled with the existence of orbits of infinite period (irregular motion). Finally, the self-similar structure explains the fractal arrangement and the folding process of the chaotic attractor.

Let us discuss the diagrams in Fig. 1.6 for a while. As discussed earlier, we can explicitly state the parameter dependence of a dynamical system, so we can alternatively represent it in the form:

$$(x, r), \; x \in \mathbb{R}^n, r \in \mathbb{C}^k. \tag{22}$$

For the logistic map:

$$(x, r) \in \mathbb{R}^1 \times \mathbb{C}^1, \tag{23}$$

hence the two-dimensional diagram. One of the most notable things in the diagrams is that by increasing the parameter r, orbits split in an ordered fashion. This splitting/pitch-forking represents a qualitative change in the dynamic behaviour of the map. The period-1 orbit "bifurcates" into a period-2 orbit (right diagram). The reason we used quotation marks is that by bifurcation we don't just assume splitting, but any qualitative change in the dynamics of a system and a proper definition will follow shortly after. This types of diagrams, where we observe the changes of the character of the solution, as parameter changes, are called *bifurcation diagrams*. Note that this splitting continues indefinitely, and it is called the *period-doubling cascade* i.e. as $n \in \mathbb{Z}, n \to \infty$ a period-2^n orbit is created. This cascade is the main culprit of the emergence of *chaos* in the logistic map. A quick glance at the left diagram, at $r_\infty = 3.57$, the diagram becomes fuzzy. At this point, orbits of infinite period are created, impliying infinitely many bifurcations. This accumulation point is closely related to a constant named the Feigenbaum number which is explained in a subsequent section (where we explain the period-doubling bifurcation in more detail).

From r_∞ to $r = 4$ we say that the map is in chaotic regime. But as we can see there is more to the bifurcation diagram than the simple distinction between periodic and chatoic behavior. There are "windows" that are easily noticeable and correlate to whole intervals of $[0, 1]$ being mapped to only 3 values, period-3 windows. There's an endless cascade of period-3^n orbits as well, finely interlaced with the period-2^n orbits. For values $r > 4$ all the orbits escape to infinity.

We now focus on more in detail of the main characteristics of chaos, as outlined at the beginning of this chapter.

3.2   Sensitive Dependence on Initial Conditions

We say that the map $f : X \to X$ has sensitive dependence on initial conditions if there exists $\delta > 0$ such that, for any $x \in X$ and any neighborhood σ of x, there exists $y \in \sigma$ and $n \geq 0$ such that $|f^n(x) - f^n(y)| > \delta$. For a map this means that for a given point x, there exist at least one arbitrarily close point whose image after n iterations will differ by δ from the image of x. Using our previous analysis of stability we concluded that an infinitesimal perturbation will behave like $|\delta x_n| = |\mu^n \delta x_0|$. For example, the *tent map* (equation 1.18, figure 1.12) has a constant slope $|\mu| = 2$ throughout, so for it, $|\mu^n| = 2^n$. Let's say that we have two point $x, y \in A = (0, 1)$ and that $y = x + \delta_0$ holds, where $\delta_0 > 0$ and it will be our initial separation. Following a similar analysis to that of the stability of periodic orbits, for the separation δ_m after m iterations of the map, we can write $\delta_m = |\mu^m|\delta_0$, from which it is obvious that $\delta_m > \delta_0$, because $\mu, m > 1$. We can think of the sensitivity to initial conditions in another way. We assume that a point x_0 belongs to a subinterval of A with a diameter of 2^{-k}. Under the action of the map, after k iterations the subinterval will be mapped onto the entire state space of the map (the whole interval A) and the state of the system x_k could be anywhere in A. Although at the beginning we know that the initial point is within a small subinterval, for x_n, $n \geq k$ we are absolutely uncertain about the location of x_n.

This sensitivity to initial conditions can be quantified by *Lyapunov exponents*. Similarly as above, we have some initial conditions $x_0 + \delta_0$, where δ_0 is very small. If we can express the separation δ_n after n iterations as $\delta x_n \approx e^{\lambda n}|\delta x_0|$, then λ is a Lyapunov exponent. For any one-dimensional map there is only one Lyapunov exponent:

$$\lambda = \lim_{n \to \infty} \frac{1}{n} \sum_{i=0}^{n-1} \ln \frac{|\delta_{n+1}|}{|\delta_n|} = \lim_{n \to \infty} \frac{1}{n} \sum_{i=0}^{n-1} \ln |(f^i)'(x_0)| \tag{24}$$

Sensitivity to initial conditions (and chaotic motion in general) is associated with a strictly positive Lyapunov exponent because the distance between nearby points grows exponentially. If we try to compute the Lyapunov exponent of the tent map, we easily arrive at $\lambda = \log 2$. In essence, Lyapunov exponents give us a measure of the divergence of infinitesimally close orbits.

3.3 Bifurcations

Bifurcation is a qualitative change in the dynamics of a given dynamical system as a control parameter is varied. The logistic map exhibits two characteristic bifurcations: a *transcritical* and a *period-doubling* bifurcation, which is also known as a *flip* bifurcation and this bifurcation is the main culprit for chaotic motion in the logistic map. We say that a bifurcation occurs if the phase portrait of a dynamical system change for some parameter r. In other words, if the dynamics of the system for $r_1 = r - \psi$, $\psi > 0$ are no longer the same as those for $r_2 = r + \psi$, $\psi > 0$, we say that for that distinct value of r a bifurcation occurred, resulting in qualitatively different phase portraits for the respective values of r. For a bifurcation to happen, a certain number of conditions have to be met, out of which the most important are the values of the parameters. In the logistic map, we only tweak one parameter, so we can say that bifurcations in the logistic map are *first-order* or *codimension-1*.

One extremely convenient way to describe the dynamics of a system is through phase portraits. This type of graphical representation does not require the computation of higher iterates and the subsequent plotting of each one of their graphs. For one-dimensional maps we can depict the orbits in $\mathbb{R}^1$ instead of $\mathbb{R}^2$ for the graph plot. On this phase line (obviously for 2-dim systems we would have a phase plane) all the information about all the iterates can be shown simultaneously.
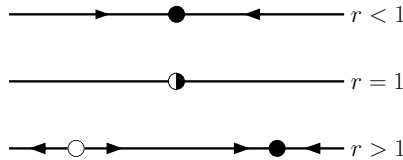
Fig. 7 Phase portraits of the logistic map for various values of r. The white circle denotes an unstable (repelling) fixed point and the black circle denotes a stable (attracting) fixed point. As the parameter is increased, we can see how one fixed point ($x_* = 0$) loses its stability and another fixed point ($x_* = 1 - r^{-1}$) is created. This actually represents the *transcritical bifurcation* explained in the section that follows.

In addition to the bifurcation diagram of the logistic map, phase portraits will help us illustrate the different types of bifurcations that occur in the map and we will see how fixed points "move" along the curve of the map resulting in different dynamics.

3.4 Transcritical bifurcation

When we discussed the stability of the fixed point $x_* = 0$, we saw that for $r = 1$ which we will denote (r_{tc}), it loses its stability. As we increase r from 0, at $r_{tc} = 1$ the diagonal is tangent to the parabola and up until this point the multiplier $\mu = f'(x_*) = r - 2rx_*$ for $x_* = 0$ is below 1. As we further increase r, the parabola rises

(a) $r < 1$ (b) $r = 1$ (c) $r > 1$
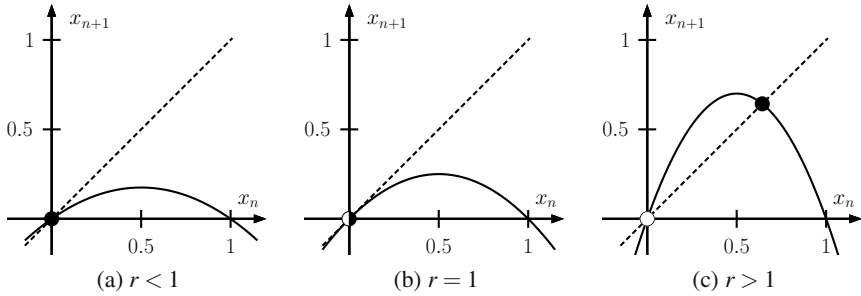
Fig. 8 Transcritical bifurcation: (a) for $r < 1$ the parabola is under the diagonal $y = x$ and the fixed point x_* is stable. (b)The diagonal is tangent to the parabola and x_* is non-hyperbolic. (c) For $r > 1$ the parabola is over the diagonal and there are two fixed points: $x_* = 0$ lost its stability and there is a new stable fixed point $x_* = 1 - r^{-1}$

above the diagonal and this gives birth to a new fixed point $x_* = 1 - r^{-1}$ (which is stable for $1 < r < 3$) at the expense of the stability of $x_* = 0$ which becomes unstable. This is an example of a *transcritical* bifurcation. See figure (transcritical.png) as an example.

3.5   *Period Doubling Bifurcation*

We actually began the analysis of the period-doubling bifurcation when we discussed the emergence of chaotic behaviour earlier on. We saw that for $r < 3$ a period-1 regime is present, but actually there are two period-1 orbit At $r = 3$ something interesting happens. The period-1 orbit loses its stability and gives rise to a stable period-2 orbit. this is called the period doubling bifurcation. In general, at a period-doubling bifurcation any period-n orbit will become unstable and give birth to a period-2n orbit, and remain present as an unstable orbit. Let's analyze the stability coefficients for $r = r_{pd} = 3$. $\mu_1 = 3$ and $\mu_2 = -1$. What this means is that we are left with two orbits, the former being an unstable period-1 orbit and the latter being stable period-2 orbit. This unstable orbit does not vanish, but is present in all subsequent dynamical regimes. The period-2 orbit can also be considered as a fixed point of the twofold iteration of the logistic map i.e.

$$f^2 = f(f(x)) = [rx(1-x)][1 - rx(1-x)] \tag{25}$$

So to compute the two periodic points x_1, x_2 we need to solve:

$$x = f^2(x) = r[rx(1-x)][1 - rx(1-x)] \tag{26}$$

Both fixed points of f are solutions to the equation, so we reduce it from quartic to quadratic:

$$b(x) = \frac{f(f(x)) - x}{f(x) - x} = r^2 x^2 - (r^2 + r)x + r + 1 = 0 \tag{27}$$

for which we can compute the solutions:

$$x_{1/2} = \frac{r+1 \pm r\sqrt{(r-3)(r+1)}}{2r} \tag{28}$$

Since r must be positive for the logistic map, that leaves us with $r_{pd} = 3$ for the value at which the period-2 orbit appears and the orbit exists for any value $r > r_{pd}$ but as we later see, its stability changes. The multiplier of the period-2 orbit can be obtained as calculating either one of the multipliers of the fixed points of f^2 since they are equal:

$$\mu_{1,2} = [f^2(x_1)]' = [f^2(x_2)]' = f'(x_2)f'(x_1) = r^2(1-2x_1)(1-2x_2) \tag{29}$$

which for $r = r_{pd} = 3$ evaluates to $\mu = 1$. The logistic map continues to act in this way as we further increase r. As we saw earlier, x_{fix2} is stable for $1 < r \leq r_{pd} = 3$ and as r goes beyond 3, the orbit gradually loses its stability, the multiplier changes from 1 to -1, and at some point r_{pd4} it bifurcates, thus giving birth to a stable period-4 orbit. So this behaviour repeats itself over and over again, like and endless cascade, generating orbits such that for the range $r_{n-1} < r \leq r_n$, the period 2^n orbit is stable, for $n \to \infty$. The range of r for which a period 2^n orbit is stable decreases almost geometrically with n. Interestingly enough, it is a constant and it is called the Feigenbaum number [Feigenbaum, 1978, 1980a]:

$$\frac{r_n - r_{n-1}}{r_{n+1} - r_n} \to 4.669201 = \delta, \text{ as } n \to \infty \tag{30}$$

Period doubling bifurcation sequence leading to a chaotic solution, represents only one of the scenarios for transition to chaos, or a route to chaos, also called Feigenbaum route or scenario.
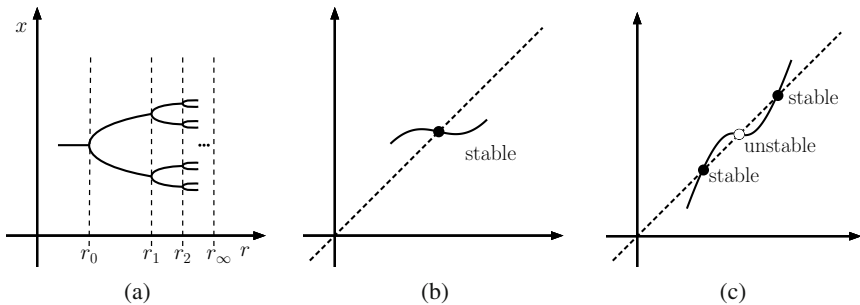
Fig. 9 (a) A simplistic display of the period-doubling bifurcation. (b) and (c) Represent a section from the second iterate of the logistic map f^2. It becomes more curved as r is increased and intersects the diagonal at three points instead of one. At $r = 3$ the fixed point $x_* = 1 - r^{-1}$ loses its stability and gives birth to two stable fixed points. As the parameter is increased further, the same happens to those fixed points as well.

3.6   *Local vs. Global Analysis*

The bifurcations we discussed are examples of *local bifurcations*. These bifurcations are caused by the variation of the value of a parameter such that it changes the stability of a fixed point i.e. at this point the multiplier of the fixed point is $|\mu| = 1$. Also, by varying the 'bifurcation' parameter around the bifurcation point allows us to confine the changes in the phase portrait of the system to arbitrarily small neighbourhood of the bifurcating fixed points, so that is why these types of bifurcations are called local. On the other hand, *global bifurcations* occur when we have larger attracting sets (like periodic orbits) collide with fixed points. The result is that now the changes in the phase portrait cannot be limited to the near vicinity of the bifurcation point. Because of the arbitrarily large influence of these changes, these bifurcations are referred to as 'global'.

The *Hartman-Grobman* theorem is one of the invaluable tools used in the local analysis of dynamical systems. This theorem can be written in a form convenient for iterated maps and it states that a discrete map $x_{n+1} = f(x_n)$ sufficiently near a fixed point $x = x_*$ *locally* has the same phase portrait as the linear map

$$x_{n+1} = \mu(x_n - x_*) \tag{31}$$

provided that the multiplier $\mu = f'(x_*)$ at the fixed point is such that $\mu \neq 1$ (i.e. it is hyperbolic).
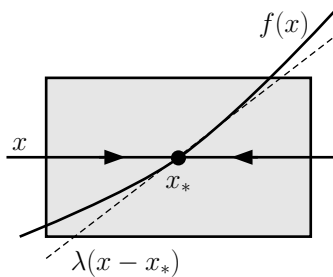
Fig. 10 The Hartman-Grobman theorem states that the dynamics around a hyperbolic point of a non-linear map can be approximated to the dynamics of a simpler linear map. This greatly facilitates the analysis of complex systems.

A bifurcation is said to be *generic* if the character of the bifurcation cannot be changed by introducing arbitrarily small perturbations to either the phase space or the parameter space. Let us denote the mapping $f : M \to M$ as $M(x, r)$, where x is the state and r is the control parameter. If we substitute $M(x, r) + \psi g(x, r)$ for $M(x, r)$, where g is smooth, for sufficiently small ψ the qualitative behaviour of the bifurcation does not change. There are only three generic bifurcation for one-dimensional maps: the period-doubling bifurcation, the tangent bifurcation and the transcritical bifurcation. Non-generic bifurcations require special conditions to be met in order for them to occur.

3.7   Stretching and Folding

In general, the asymptotic dynamics of many chaotic systems which are *dissipative*, is limited to an arbitrarily small segment or region of the state space. Therefore, any *stretching* that occurs must be counteracted with an antagonistic *squeezing* so that orbits must remain in a bounded region (expansion $\Longleftrightarrow$ contraction). The stretching and the squeezing can be combined by folding processes and one of the best examples is the Smale's horseshoe map [Smale 1967]. We take a square S and stretch it along the vertical direction (fig(a)), while compressing it in the horizontal direction (fig(b)). Then we fold the deformed rectangle in the middle and place the horseshoe on top of the original square S during which, the striped area of the horseshoe is mapped outside the square S. The intersection of the original square with the horseshoe is the two vertical strips:

$$f(s) \cap S = V_0 \cup V_1 \tag{32}$$

Iterating f^{-1}, we take the same steps only in reverse order (d-a). The conclusion is that the vertical strip V_i actually comes from a horizontal strip $H_i = f^{-1}(V_i)$ and thus

$$f^{-1}(S) \cap S = H_0 \cup H_1 \tag{33}$$

Now that we know how the stretching and folding processes work, we transfer our discussion to one-dimensional maps. We saw earlier that for $r = 4$ the logistic map
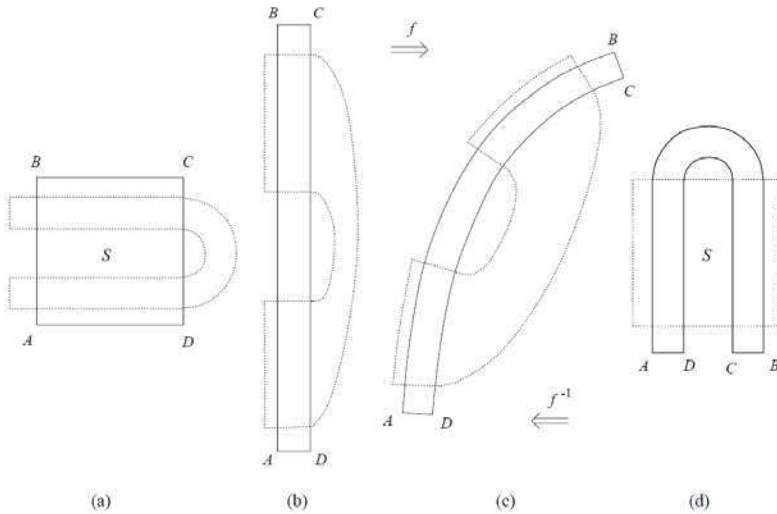
Fig. 11 (a)The square is stretched at more than twice its height. (b) The square is squeezed at less then a half of its width. (c) The resulting rectangle is folded at the middle. (d)The horseshoe is placed over the original square. This completes one iteration of the Smale's horseshoe map. Image taken from [Kuznetsov,1998]
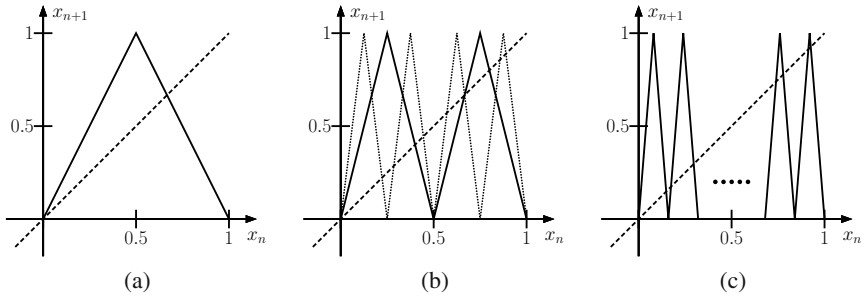
Fig. 12 (a) Graph of the tent map (1.18) alongside the diagonal $y = x$. (b) Graphs of high-order iterates of the tent map: the twin-peak structure represents the second iterate h^2 and the more complex structure is the fourth iterate h^4.

is chaotic. For this value the whole interval $A = (0, 1)$ is mapped onto itself i.e. the map is surjective. Moreover, every $x \in A$ is actually the image of two different point $x_1, x_2 \in A$. Because of the surjectivity, A is also an invariant set $(f(A) \rightarrow A)$. To simplify our analysis we will use a transformation of the logistic map. We will resort to a change of coordinates, namely $x = \sin^2(\pi y)$ and using some trigonometric substitutions we arrive at:

$$y_{n+1} = h(y_n) = 1 - 2\left|y_n - \frac{1}{2}\right| \tag{34}$$

which is a piecewise linear map and it is called the *tent map* because of its graph (fig ...). What is immediately noticeable is that the tent map looks like a spiky version of the logistic map and this feature means that computation of the higher-order iterates is greatly simplified. Because it is piecewise linear and symmetric at $x = 1/2$, the slope is constant $|h'(y)| = 2$ on the whole interval $A = (0, 1)$ on which the map is defined. We can split the interval A into two subintervals $A = A_0 \cup A_1$, such that $h(A_0) = h(A_1) = A$.

From Fig. 1.12 (b) we can see that the iterations of the tent map resemble the ribs of an accordion watched through a square window. The higher the iterate, the more ribs are "folded in". This is due to the *stretching* and *folding* mechanisms of the tent map. The two peaks in the graph of h^2 are a result of how h maps each of A_0 and A_1 onto the whole interval A, so the graphs of h^2 of A_i $i = 1, 2$ are the same as the graph h on A, with the obvious restriction that they are defined on twice as narrow interval: $|A_1| = |A_2| = 1/2|A|$. Basically, this means that a h^n iterate of the map will uniformly stretch a subinterval $A_i |A_i| = 1/2^n$ to the whole interval A in $N \leq n$ iterations, doubling the length A_i at each iteration. As we iterate further, the doubling of the length of the interval continues, but the orbits must stay bounded within the invariant set A. What we have is that all the time the interval A will continue to map onto itself, a process known as *folding*. From now on, the motion of the map will follow a pattern consisting of alternating stretching and folding:

- The whole interval will be stretched twice its length
- The resulting interval will be folded in half, so it fits the invariant set on which the map is defined.

Now we can clearly see two very important features (which we mentioned earlier) of chaotic maps. First, sensitivity to initial conditions simply because the stretching means exponential divergence of neighbouring orbits (factor of two for each iterate of the tent map). The folding process keeps the orbits bounded. Second, non-invertibility, which is caused by the folding action, because the whole invariant interval is folded onto itself resulting into an image having two distinct preimages.
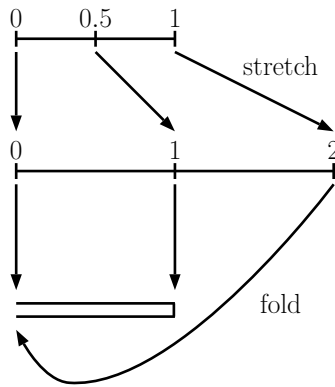
Fig. 13 The stretching and folding processes in the tent map. First the interval is stretched to twice its length, and then it is folded onto itself. These two properties are inherent to chaos: for a map to be chaotic it must be expanding, but for the orbits to remain bounded there must be folding as well

3.8   Strange Attractors

We saw earlier that the tent and logistic maps are chaotic on their entire phase space, but this is not the case with other systems, which are only chaotic within a small bounded subset of their phase space. In general, a map is said to be *dissipative* if it does not preserve phase space volumes on each iterate. This means that a bounded region of the phase space will shrink as the map is iterated and this is associated with the presence of an attractor within that region. All three mechanisms: stretching, folding and squeezing are inherent to chaotic dynamics and they cause the geometry of an attractor to be quite more complex than that of the attractors we have seen so far. To illustrate this, we will use the Hénon map and will treat it in a similar fashion to [Ott, 1993]. Figures (a) through (c) show that as we iteratively zoom in the attractor and observe it at smaller scales, each of the structures we observe are similar to one another (a number of parallel lines). If we continue to zoom in, we will conclude that the attractor has a *self-similar* structure on an *arbitrarily small scale*. This is in fact the main feature of a *fractal* which is a geometrical object that
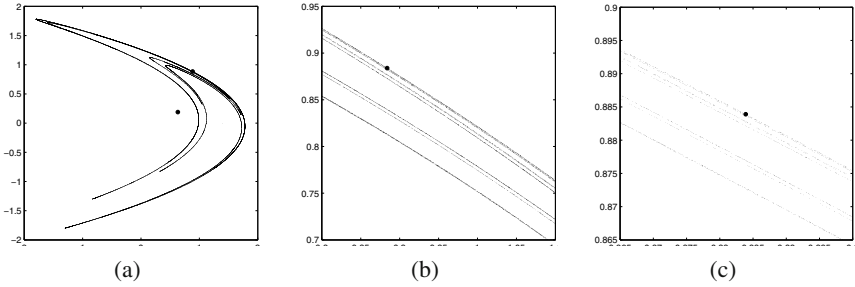
Fig. 14 (a) The Hénon attractor. (b) A magnification of the region around the attractor [0.883896,0.883896] i.e. the upper dot. (c) A further magnification of the region the attractor. From (b) and (c) we can see the self similar structure of fractal attractors.

possesses a non-integer dimension (hence the name). Fractal attractors are called *strange attractors* and often the dynamics on a strange attractor are chaotic, but there are also example of non-chaotic strange attractors. Strange non-chaotic attractors are actually self-similar fractal sets for which sensitive dependence on initial conditions does not hold. The Hénon attractor is a set of infinite number of lines, so on one hand it has a dimension greater than one, but on the other it doesn't have a surface so it cannot be of dimension two. The actual value of the fractal dimension of the Hénon attractor is $D_0 \simeq 1.26$.

3.9   Symbolic Dynamics

The other reasons of introducing the Smale's horseshoe map is that it is very important (actually it was originally conceived for) in the study of *symbolic dynamics*. In the rest of the section we follow the formalism as in [Gilmore and LeFranc, 2002]

Symbolic dynamics is a useful technique for analyzing the motion of dynamical systems. It relies on following orbits and marking regions of the state space they visit and associating different symbols to these different *partitions* of the state space. Let us assume that a unimodal one-dimensional map is defined on the interval A. The most simple splitting of this interval would be to divide it in two parts A_0 and A_1. Since the splitting points should be chosen such that the boundaries of the partitions are the critical points of the map, this binary partitioning is quite appropriate (unimodal maps have only one critical point). But in the general case, we decompose the interval A into N disjoint intervals A_α, $\alpha = 0 \ldots N-1$, ordered from left to right, such that:

$$A = A_0 \cup A_1 \cup \ldots A_{N-1} \qquad (35)$$

Note that this kind of partitioning only holds for one-dimensional maps. After splitting the interval into a union of disjoint A_α intervals, like in Fig. 15 (b), at each iteration, we adhere/adjoin a symbol $\alpha \in \mathscr{A} = 0, \ldots, N-1$ depending on which interval the current state/point belongs to. Note that $\mathscr{A}$ is actually an alphabet of the N symbols that α can assume. Now we can define the coding function $s(x)$:

$$s(x) = \alpha \iff x \in A_\alpha \tag{36}$$

Now, we can create a representation $\Sigma(x)$ of an arbitrary orbit $\mathcal{O} = f^n(x) = \{x, f(x), f^2(x), \dots\}$ such that $\Sigma(x)$ is the infinite sequence of the symbols denoting which intervals the orbits visits in succession:

$$\Sigma(x) = s(x), s(f(x)), s(f^2(x)), \dots, s(f^i(x)), \dots \tag{37}$$

This sequence is called the *itinerary* of x. We denote the set of all possible sequences in the alphabet $\mathscr{A}$ as $\mathscr{A}^{\mathbb{N}}$, and

$$\Sigma(A) = \{\Sigma(x); \; x \in A\}$$

We can define the shift operator σ as

$$\Sigma = \{s_0, s_1, s_2, \dots, s_i, \dots\} \xrightarrow{\sigma} \{s_1, s_2, s_3, \dots, s_i, \dots\} = \sigma\Sigma \tag{38}$$

So, applying the shift operator σ on the symbolic sequence $\Sigma(x) \in \Sigma(A)$ is the same as applying f on x:

$$\Sigma(f(x)) = \sigma\Sigma(x) \tag{39}$$

Visually, we can draw a commutative diagram such as:

$$
\begin{array}{ccc}
x & \xrightarrow{\quad f \quad} & f(x) \\
\downarrow {\scriptstyle \Sigma} & & \downarrow {\scriptstyle \Sigma} \\
\{s_i\}_{i \in \mathbb{N}} & \xrightarrow{\quad \sigma \quad} & \{s_{i+1}\}_{i \in \mathbb{N}}
\end{array}
$$

There is one important remark to be made. Note that we only shifted the symbolic sequences to the right. This corresponds to moving forwards in time, and for non-invertible maps is quite sufficient (the only orbits we can compute are $\mathcal{O}^+ = \{f^n(y)_{n \geq 0}\}$. So the sequences are one-sided and extend to $+\infty$ (forward in time). Thus, σ is non-invertible. We can in fact define operators that are inverse to σ i.e. σ_α^{-1} that insert a symbol (α in this case) to the head of the symbolic sequence:

$$\Sigma = \{s_0, s_1, s_2, \dots, s_i, \dots\} \xrightarrow{\sigma_\alpha^{-1}} \{\alpha, s_0, s_1, s_2, \dots, s_{i-1}, \dots\} = \sigma_\alpha^{-1}\Sigma \tag{40}$$

In general, $f^{-1}(f(x)) = f(f^{-1}(x)) = x$, but for the shift operator this is not the case, since first we lose a symbol (which can take N-values) and then we replace it with a fixed symbol α, so we end up with: $\sigma(\sigma_\alpha^{-1} = \mathrm{Id} \neq \sigma_\alpha^{-1}(\sigma)$. The way we decided to symbolize the orbits, presents quite a convenient way to mark periodic orbits, so we have a periodic sequence such that: $\Sigma = \{s_i\}$ with $s_i = s_{i+p}$ for all $i \in \mathbb{N}$, and it can easily be concluded that they satisfy $\sigma^p\Sigma = \Sigma$, which we can see that using the commutative diagram is: $f^p(x) = x$.
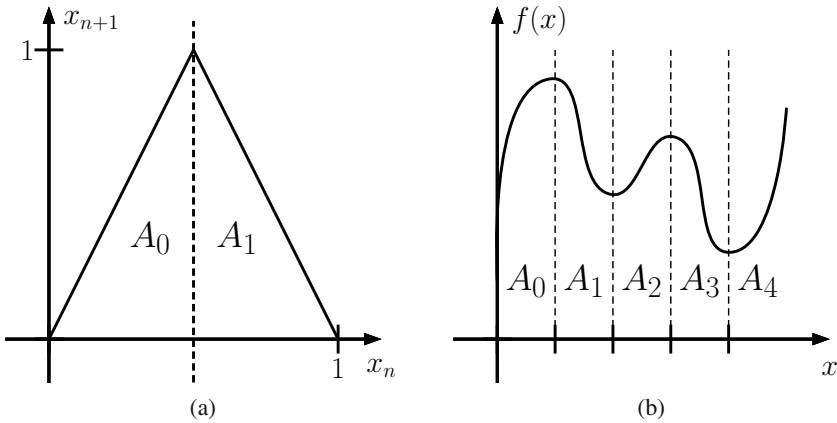
Fig. 15 (a) Partitioning of the tent map. Since it has only one critical point at $x = 0.5$ we can divide its domain in two disjoint intervals and associate the symbols of 0 and 1 to them. (b) presents a more complex case where the map has more than one maxima and minima so additional symbols have to be used. Note that the symbols are adhered to each subinterval from left to right

3.10 Invariant Densities and Measures

We discussed in several occasions the complex, random-like and irregular character of chaotic solutions, or signals, if we encounter them as a time-series in experiments. Due to the distinct properties of the geometry of their attracting sets, attempting to obtain the asymptotic solutions in the full phase space is obviously important, but when temporal sequences become very complicated and the dimensionality is three or higher, one can not say much about the *time-averaged behaviour* of the solution, if finite length sequences of such solutions would eventually appear along the motion or response of realistic apparatus or a device.

This practically means that despite we deal with a system which is essentially governed by deterministic laws, we will benefit if we can obtain *statistical characterization* of the behavior for the whole range of equally probable initial conditions and specific parameter values. We would like to stress, that here we do not talk of purely random and stochastic components like transient disturbances, additive noise or channel noise, but the irregular behavior of the chaotic solutions.

The focus of statistical analysis of dynamical systems is on *invariant probability measures* and *invariant density*, if the later can be defined.

By defining density $\rho(x)$ we assign to any of the points x_0 of the range of $x(t)$ a scalar frequency representing how often chaotic trajectory comes arbitrary close to x_0. When we define the density on the attracting set, and especially on chaotic attractor we talk of *invariant density*. There are some times singularities in defining the invariant densities, typically only on countably many points for a critical value of the map.

A good example is the logistics map for $r = 4$ where critical point $x = 1/2$ is iterated to the unstable fixed point $x_p = 1$, which is then mapped to the stable $x_p = 0$, and stays there. Figure 1.16.a shows the density of logistic map after change of coordinates used in Ch.3.7, for $r = 4$, with singularities at the ends of the interval. In this case with the mentioned change of the coordinates it is possible to get $\rho(x)$ in analytic form. In practice we compute the densities as histograms of sufficiently long trajectories, well after the initial transients, and eventually smooth them out numerically.

For values lower than $r = 4$ where the logistic map or its higher order iterates are along the period-doubling route to fully developed chaos, the number of unstable fixed points or unstable periodic orbits is growing, fractioning the interval $[0, 1]$ into subintervals. On the ends of such subintervals invariant density will always show a singular behavior, as approximately shown on Fig. 1.16.b, for $r = 3.8$ [Shaw. 1981]. For the case of fully developed chaos one can expect erratic looking nowhere smooth graph of peak values, due to the invariant set reducing to a strange attractor of Lebesgue measure zero (having no parts with positive natural length). In such limiting case, for r=4, we are getting the smooth density of fully developed chaos, i.e. Fig. 16.a.

Instead of dealing with invariant densities which are essentially a point-wise property we can analyse the corresponding *invariant measures* μ, in general case related by

$$d\mu(x) = \rho(x)dx \qquad (41)$$

when the density can be defined. There are different ways to define appropriate measure, although we can say that both invariant measures and invariant densities are proven to exist for chaotic attractors of one-dimensional maps, under very general conditions. For more detailed treatment of measures in chaotic dynamical systems please refer to [Ruelle, 1989]. In higher-dimensional dynamical systems the existence of invariant measures is more complicated, and it is generally an open problem.

The central idea is to relate *time averages* of arbitrary long time-series of an experimental observable $\phi[x(t)]$, to the *phase space averages* at the chaotic attractor, weighted by the invariant measure, i.e.

$$\lim_{T \to \infty} \frac{1}{T} \sum_{n=0}^{T} \phi(M^n(x_i)) = \int \phi[x(t)]d\mu(x). \qquad (42)$$

Furthermore, where the above holds *almost everywhere*, or in other words for typically chosen x_i from the invariant set, except for the countably many points of Lebesgue measure zero, we say that dynamical system is *ergodic*. For quite complete discussion of properties of invariant measures and densities of the logistic map refer to [Ott, 1993] while more complete treatment of invariant measures on chaotic attractors can be found in [Ruelle, 1989].
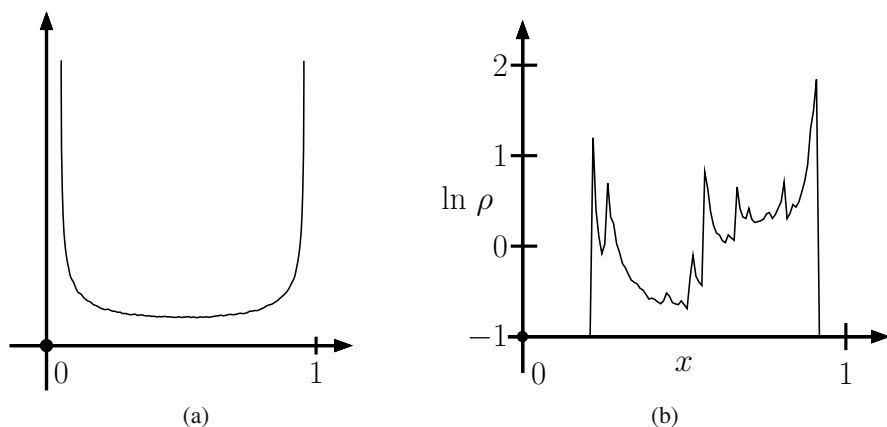
Fig. 16 Examples of invariant densities. a) Invariant density of logistic map for $r = 4$, showing a typical smooth behavior of fully developed chaos except at the singularities on the ends of the interval. b) Erratic character of the density for $r = 3.8$ [Shaw, 1981], along period-doubling cascade, where $[0, 1]$ interval is being fragmented to subintervals resulting into countably infinite singular points (the end s of each subinterval).

References

1. Kocarev, L., Galias, Z., Lian, S. (eds.): Intelligent Computing Based on Chaos Series: Studies in Computational Intelligence, vol. 184. Springer, Heidelberg (2009)
2. Devaney, R.L.: An Introduction to Chaotic Dynamical Systems. Benjamin/Cummings, Menlo Park (1986)
3. Guckenheimer, J., Holmes, P.: Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields. Springer, New York (1983)
4. Hirsch, M.W., Smale, S.: Differential Equations, Dynamical Systems and Linear Algebra. Academic Press, New York (1974)
5. Kuznetsov, Y.A.: Elements of Applied Bifurcation Theory, 3rd edn. Springer, Heidelberg (2004)
6. Gilmore, R., Lefranc, M.: The Topology of Chaos, Alice in Stretch and Sqeezeland. Wiley, NY (2002)
7. Edward, O.: Chaos in Dynamical Systems. Cambridge University Press, New York (2002)
8. Feigenbaum, M.J.: Quantitative Universality for a Class of Nonlinear Transformations. J. Stat. Phys. 19(25), A978
9. Feigenbaum, M.J.: The Metric Universal Properties of Period Doubling Bifurcations and the Spectrum for a Route to Turbulence. Ann. New York. Acad. Sci. 357, 330–336 (1980)
10. Smale, S.: Differentiable dynamical systems. Bull. Amer. Math. Soc. 73, 747–817 (1967)
11. Grebogi, C., Ott, E., Yorke, J.: Chaos, Strange Attractors and Fractal Basin Boundaries in Nonlinear Dynamics. Science 238(585), A987d
12. Shaw, R.: Strange Attractors, Chaotic Behavior, and Information Flow. Z. Naturforsch A 36(80), A981
13. Ruelle, D.: Chaotic Evolution and Strange Attractors. Cambridge University Press, New York (1989)

Chapter 2
Chaos-Based Public-Key Cryptography

Igor Mishkovski[1] and Ljupco Kocarev[2]

[1] Department of Electronics, Politecnico di Torino, Corso Duca degli Abruzzi 24,
I-10129, Turin, Italy
`igor.mishkovski@polito.it`

[2] Institute for Nonlinear Science, University of California, San Diego, 9500 Gilman Drive,
La Jolla, CA 92093-0402, USA
`lkocarev@ucsd.edu`

Abstract. In this chapter we give an overview and the state of the art in the field of Chaos-based cryptography. The public key cryptosystems based on Chebyshev polynomials enjoy some nice chaotic properties, which makes them suitable for use in both encryption and digital signature. The cryptosystem can work either on real or integer numbers. The cryptosystem that works on real numbers is not secure and permits to recover the corresponding plaintext from a given ciphertext. In addition, it also allows forgeries if the cryptosystem is used for signing messages. On the other hand, ElGamal-like and RSA-like algorithms when using Chebyshev polynomials on integer numbers are secure as the aforementioned encryption algorithms. The chaos-based cryptography is discussed from a point of view which we believe is closer to the spirit of both cryptography and chaos theory than the way the subject has been treated recently by many researchers.

1 Introduction

The study of chaotic systems and their possible applications to Cryptography has received considerable attention during the last years in a part of the scientific community. Chaotic systems are characterized by sensitive dependence on initial conditions, similarity to random behavior, and continuous broad-band power spectrum. Chaos has potential applications in several functional blocks of a digital communication system: compression, encryption, and modulation. In early days (from 1992 to 1996) the main research goal was to develop schemes in which a single chaotic system is used for both modulation and encryption. This approach eventually evolved into two distinct research areas: chaos-based modulation [1], [2] and chaos-based cryptography [3], [4].

Cryptography is generally acknowledged as the best method of data protection against passive and active fraud [5]. An overview of recent developments in the design of conventional cryptographic algorithms is given in [5]. The tight relationship between chaos and cryptography is given in the Shanon's paper on cryptography [6]:

"Good mixing transformations are often formed by repeated products of two simple non-commuting operations. Hopf has shown, for example, that pastry dough can be mixed by such a sequence of operations. The dough is first rolled out into a thin slab, then folded over, then rolled, and then folded again, etc."

More detailed relationship between chaos and cryptography, is given in [7]. Table 1 contains a partial list of these properties. Despite the given relationship, a deep relation between chaos and cryptography has not yet been establish. In Table 2 we give the main similarities and differences between chaotic systems and cryptographic algorithms. Table 2 summarizes similarities and differences between chaotic maps and cryptographic algorithms. Chaotic maps and cryptographic algorithms (or more generally maps defined on finite sets) have some similar properties: sensitivity to a change in initial conditions and parameters, random-like behavior and unstable periodic orbits with long periods. Encryption rounds of a cryptographic algorithm lead to the desired diffusion and confusion properties of the algorithm. Iterations of a chaotic map spread the initial region over the entire phase space. The parameters of the chaotic map may represent the key of the encryption algorithm. An important difference between chaos and cryptography is that encryption transformations are defined on finite sets, while chaos has meaning only on real numbers. Moreover, for the time being, the notions of cryptographic security and performance of cryptographic algorithms have nocounterpart in chaos theory.

Table 1 Comparison between chaos and cryptography properties.

Chaotic property	Cryptographic property	Description
Ergodicity	Confusion	The output has the same distribution for any input
Sensitivity to initial conditions/control parameter	Diffusion with a small change in the plaintext/secret key	A small deviation in the input can cause a large change at the output
Mixing property	Diffusion with a small change in one plain-block of the whole plaintext	A small deviation in the local area can cause a large change in the whole space
Deterministic dynamics	Deterministic pseudo-randomness	A deterministic process can cause a pseudo-random behavior
Structure complexity	Algorithm (attack) complexity	A simple process has a very high complexity

Short summary of the three types of cryptographic objects and their chaos-based implementation follows. Three most common cryptographic objects are: block-encryption algorithms (private-key algorithms), pseudo-random number generators (additive stream ciphers), and public-key algorithms.

Table 2 Similarities and differences between chaotic systems and cryptographic algorithms.

Cryptographic algorithms	Chaotic systems
Phase space: finite set of integers	Phase space: (sub)set of real numbers
Algebraic methods	Analytic methods
Rounds	Iterations
Key (Boolean) - Discrete keyspace	Parameters (real) - Continuous keyspace
Diffusion	Sensitivity to a change in initial condition/parameters
Digital realizations by integer arithmetic	Digital realization by non integer arithmetic which approximates continuous continuous-value systems
Security and performance	?

Block ciphers transform a relatively short string (typically 64, 128, or 256 bits) to a string of the same length under control of a secret key. Several block encryption ciphers based on chaotic maps have been proposed in the literature, in which a discretization (a process that describes the way a chaotic map is implemented in the computer) is not realized by rounding the chaotic map according to the computer arithmetic, but rather is constructed explicitly. Pichler and Scharinger [8] proposed cryptographic systems based on chaotic permutations constructed by explicitly discretizing the two-dimensional bakers map. Fridrich [9] extended their ideas to chaotic permutations on any size of two-dimensional lattices. Her permutations benefit from the expanding property along one axis, technically avoiding the contracting property along the other axis. The authors of [10] used two well-known chaotic maps, exponential and logistic, to construct a class of block encryption algorithms. In [11], they analytically derived the lower bound of a number of active S-boxes in their algorithms, computed upper bounds for differential and linear probabilities, and therefore, proved the resistance of the algorithms proposed [10] to differential and linear attacks. Masuda and Aihara [12] considered a discrete version of the skew-tent map, which exploits important chaotic properties such as the sensitive dependence on initial conditions and the exponential information decay. They discussed the difference between the discretized map and the original map, explaining the ergodic- and chaotic-like properties of the discretized map. In [13] authors propose software base approach of chaos based cryptography superior to the realization of Lorenz dynamics on electronics circuit. These approach does not suffer from parameter drifts, stability and it is not prone to the reconstruction of dynamics, i.e. is not vulnerable to attacks as the circuit based approach. The systems dynamics are hidden using trajectory folding and their results show that both the encryption and the decryption are fast and implementable.

A pseudo-random number generator is a deterministic method, usually described with a mapping, to produce from a small set of "random" numbers, called the seed, a larger set of random-looking numbers called pseudo-random numbers. Chaotic systems may be used to generate pseudo-random numbers. For example, in a series of papers [14], the authors proposed a chaos-derived pseudo-random number

generator. They numerically observed that the average cycle and transient lengths grow exponentially with the precision of implementation, and from this fact deduced that using high-precision arithmetic one can obtain pseudo-random number generators (PRNGs) which are still of cryptographic interest. Statistical properties of binary sequences generated by a class of ergodic maps with some symmetrical properties are discussed in [15]. The authors derived a sufficient condition for this class of maps to produce a sequence of independent and identically distributed binary random variables. However, the authors did not discuss the implementation of these maps on finite-state machines and the consequences this implementation may have on the randomness of the generated sequences. In [16], the authors proposed a class of chaos-based pseudo-random bit generators.

Certain applications in cryptography require the use of a truly random number generator (RNG), which is a device which outputs a sequence of statistically independent and unbiased numbers. It is widely accepted that the core of any RNG must be an intrinsically random physical process. Thus, it is no surprise that the proposals and implementations of RNGs range from tossing a coin, to measuring thermal noise from a resistor and shot noise from a Zener diode or a vacuum tube, measuring radioactive decay from a radioactive source, and sampling a stable high-frequency oscillator with an unstable low-frequency clock, to mention only a few proposals. For chaos-based generators of truly random numbers, see, for example, [17], [18], [19], [20]. Papers [18], [19] are devoted to the analysis of the application of a chaotic piecewise-linear one-dimensional map as an RNG. Piecewise linearity of the map enables the authors to mathematically find parameter values for which a generating partition is Markov and the RNG behaves as a Markov information source, and then to mathematically analyze the information generation process and the RNG. The map is implemented in a 0.8m standard complementary metal oxide semiconductor (CMOS) process utilizing switched current techniques.

Public-key algorithms [5], also called asymmetric algorithms, are designed so that:

1. the encryption key is different from the decryption key;
2. the encryption key can be made public; and
3. the decryption key cannot, at least in any reasonable amount of time, be calculated from the encryption key.

There are many public-key algorithms; the three most widely used public-key cryptosystems are: RSA, ElGamal, and Rabin [5]. In this chapter we investigate public-key encryption algorithms using Chebyshev maps [21], [22], define on the set [-1, 1], and implemented using floating-point arithmetic, which in [23] are shown that are not secure. Furthermore, we show the Elgamal-like and RSA-like public-key algorithm using Chebyshev maps, presented in [24]. In this case, our analysis of the periodic orbits in sequences of integers generated by Chebyshev maps is based on the arithmetic properties of toral automorphisms, another well-known class of chaotic maps. This kind of chaotic-based cryptography is secure and practical, and can be used for both encryption and digital signature.

However, despite the huge number of papers published in the field of chaos-based cryptography (some of them cited here), the impact that this research has made on conventional cryptography is rather marginal. This is due to two reasons:

- First, almost all chaos-based cryptographic algorithms use dynamical systems defined on the set of real numbers, and therefore are difficult for practical realization and circuit implementation.
- Second, security and performance of almost all proposed chaos-based methods are not analyzed in terms of the techniques developed in cryptography. Moreover, most of the proposed methods generate cryptographically weak and slow algorithms.

2   Public-Key Encryption

Cryptography has come to be understood to be the science of secure communication. The publication in 1949 by C. E. Shannon of the paper "Communication Theory of Secrecy Systems" [6] ushered in the era of *scientific secret-key cryptography*. However, Shannons 1949 paper did not lead to the same explosion of research in cryptography that his 1948 paper had triggered in information theory [25]. The real explosion came with the publication, in 1976, by W. Diffie and M. E. Hellman of their paper, "New Directions in Cryptography" [26]. Diffie and Hellman showed for the first time that secret communication was possible without any transfer of a secret key between sender and receiver, thus establishing the turbulent epoch of *public-key cryptography*. Moreover, they suggested that computational complexity theory might serve as a basis for future research in cryptography. In a public-key encryption system [5] Alice has a *public key e* and a corresponding *private key d*. In secure systems, the task of computing d given e is computationally infeasible. The public key defines an encryption transformation E_e, while the private key defines the associated decryption transformation D_d. Bob, wishing to send a message m to Alice, obtains an authentic copy of Alice's public key e, uses the encryption transformation to obtain the cipher-text $c = E_e(m)$, and transmits c to Alice. To decrypt c, Alice applies the decryption transformation to obtain the original message $m = D_d(c)$.

Since 1976, numerous public-key algorithms have been proposed; the three most widely used public-key crypto-systems are: RSA, Rabin, and ElGamal. The security of the RSA system, named after its inventors R. Rivest, A. Shamir, and L. Adleman, is based on the intractability of the integer factorization problem. In the Rabin public-key encryption scheme, the problem faced by a passive adversary is computationally equivalent to factoring. The security of the ElGamal public-key system is based on the intractability of the discrete logarithm problem. Public-key encryption schemes are typically substantially slower than symmetric-key encryption algorithms. For this reason, public-key encryption is most commonly used in practice for encryption of small data items and/or for transport of keys, subsequently used for data encryption by symmetric-key algorithms.

Recall first the basic ElGamal algorithm. The ElGamal public-key algorithm can be viewed as Diffie-Hellman key agreement in key transfer-mode [5]. Consider a class of functions defined as $\pi(x) = x^p(\mathrm{mod}\,N)$, where N is a prime number, x is a generator of the multiplicative group $\mathbb{Z}_N^*$, and $1 \leq p \leq N-2$. Any two functions π and π_q commute under composition:

$$\pi_p(\pi_q(x)) = \pi_{pq}(x) \tag{1}$$

The Diffie-Hellman key agreement protocol describes how Alice and Bob agree on their common secret key. Alice generates a number p, computes $y = \pi(x)$ and sends (x, y) to Bob. Bob creates a number q, computes $z = \pi_q(x)$ and sends z to Alice. The secret key, which can be shared by both Alice and Bob, is computed as follows. Alice computes the secret key k as $k = \pi(z)$. Bob computes the secret key k as $k = \pi_q(y)$.

In the ElGamal public-key scheme, Alice generates a large random prime N and a generator x of the multiplicative group $\mathbb{Z}_N^*$ of integers modulo N. She also generates a random integer $s \leq N-2$ and computes $A = x^s(\mathrm{mod}\,N)$. Alice's pubic key is (x, N, A); Alice's private key is s. To encrypt a message m, Bob selects a random integer $r \leq N-2$, computes $B = x^r(\mathrm{mod}\,N)$ and $X = mA^r(\mathrm{mod}\,N)$, and sends the cipher-text $c = (B, X)$ to Alice. To recover the message m from c, Alice uses the private key s to recover m by computing $m = B^{-s}X(\mathrm{mod}\,N)$. The decryption allows recovery of the original message because $B^{-s}mA^r \equiv x^{-rs}mx^{rs} \equiv m(\mathrm{mod}\,N)$.

Recall now the RSA algorithm. Let $N = pq$ and $\phi = (p-1)(q-1)$, where p and q are two large random (and distinct) primes p and q. Alice selects a random integer e, $1 < e < \phi$, such that $\gcd(e, \phi) = 1$ and computes the unique integer d, $1 < d < \phi$, such that $ed \equiv 1(\mathrm{mod}\,\phi)$. Alice's public key is (N, e); Alice's private key is d. To encrypt a message m, Bob computes $c = m^e(\mathrm{mod}\,N)$ and sends to Alice. To recover the message m from c, Alice should use the private key d to recover $m = c^d(\mathrm{mod}\,N)$. Let $\pi_p(x) = x^p(\mathrm{mod}\,N)$. The decryption in the RSA algorithm works for two reasons: the functions π_e and π_d commute under composition, and p is a periodic point of the function π_{Ed} for every m: $m^{ed} \equiv m(\mathrm{mod}\,N)$. The last follows from the following observation. Since $ed \equiv 1(\mathrm{mod}\,\phi)$, there exists an integer k such that $ed = 1 + k\phi$. Now, if $\gcd(m, p) = 1$, then by Fermat's theorem $m^{p-1} \equiv 1(\mathrm{mod}\,p)$. Raising both sides of this congruence to the power of $k(q-1)$ and then multiplying both sides by m yields $m^{ed} \equiv m(\mathrm{mod}\,p)$. By the same argument $m^{ed} \equiv m(\mathrm{mod}\,q)$. Finally, since p and q are distinct primes, it follows that $m^{ed} \equiv m(\mathrm{mod}\,N)$.

3   Chaotic Maps

3.1   Chebyshev Maps

A Chebyshev polynomial map $T_p : R \rightarrow R$ of degree p is defined using the following recurrent relation:

$$T_{p+1}(x) = 2xT_p(x) - T_{p-1}(x). \tag{2}$$

with $T_0 = 1$ and $T_1 = x$. The first few Chebyshev polynomials are

$$T_2(x) = 2x^2 - 1,$$
$$T_3(x) = 4x^3 - 3x,$$
$$T_4(x) = 8x^4 - 8x^2 + 1.$$

One of the most remarkable properties of the Chebyshev polynomials is thesemi-group property [27]:

$$T_r(T_s(x)) = T_{rs}(x). \tag{3}$$

An immediate consequence of this property is that Chebyshev polynomials commute under composition, i.e.,

$$T_s(T_r) = T_r(T_s). \tag{4}$$

The interval $[-1,1]$ is invariant under the action of the map $T_p : T_p([-1,1]) = [-1,1]$. Thus, the Chebyshev polynomial restricted to the interval $[-1,1]$ is the well-known chaotic map for all $p > 1$: it has a unique absolutely continuous invariant measure

$$\mu(x)dx = \frac{dx}{\pi\sqrt{1-x^2}},$$

with positive Lyapunov exponent $\lambda = \ln p$. For $p = 2$, the Chebyshev map reduces to the well-known logistic map.

For both ElGamal and RSA algorithms, property (1) is crucial for encrypting and decrypting the information. Now we address the following question: Are there other functions with the semigroup property (1)? We consider only polynomials. Two polynomials, P and Q, are called permutable if $P(Q(x)) = Q(P(x))$ for all x. If we write $P \circ Q$ to indicate composition $P(Q(x))$, then P and Q are permutable if $P \circ Q = Q \circ P$. A sequence of polynomials, each of positive degree, containing at least one of each positive degree and such that every two polynomials are permutable, is called a chain. The Chebyshev polynomials $T_1(x), T_2(x), \dots$, form a chain. The powers $\pi_j(x) \equiv x^j, j = 1, 2, \dots$, form a chain as well. Suppose that $\lambda(x) = ax + b, a \neq 0$, so that $\lambda^{-1}(x) = (x - b)/a$. If P and Q commute, it is clear that $\lambda^{-1} \circ P \circ \lambda$ and $\lambda^{-1} \circ Q \circ \lambda$ also commute. We say that P and $\lambda^{-1} \circ P \circ \lambda$ are *similar*.

The answer to the above question for polynomials is given by the following theorem [27]: If P and Q commute, either both are iterates of the same polynomial or both are similar, with respect to the same λ, to either Chebyshev polynomials or powers. Thus, the sequences T_j and π_j are the only chains, up to similarities.

3.2   Torus Automorphisms

In this section we briefly discuss some general properties of automorphisms of the two-dimensional torus. An automorphism of the 2-torus is implemented by a 2×2

matrix M with integer entities and determinant ± 1. The requirement that the matrix M has integer entities ensures that M maps torus into itself. The requirement that the determinant of the matrix M is ± 1 guarantees invertibility. Here we consider only strictly unimodular automorphisms, for which $\det M = 1$.

Let M be a 2-torus automorphism

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = M \begin{bmatrix} x \\ y \end{bmatrix} (\bmod 1), \tag{5}$$

where $x, y \in [0, 1]$. Let k be a trace (which is an integer) of the automorphism M, $f(z) = z^2 - kz + 1$ its characteristic polynomial, and λ one of its roots (say, the largest one):

$$\lambda = \frac{k + \sqrt{k^2 - 4}}{2}.$$

It is well known that for $k > 2$ (we will consider only positive k) the automorphism M has strong chaotic properties and, in particular, it has a dense set of unstable periodic orbits. The detailed structure of periodic orbits of the 2-torus automorphisms has been studied by Percival and Vivaldi [28].

Periodic orbits of a toral automorphism consist of those points having rational coordinates $\xi = p1/q1$, $\eta = p2/q2$, p_i, q_i integers. Let p_i, q_i be co-primes (their greatest common divisor is 1) and let q be the least common multiple of q_1 and q_2. Clearing denominators, we let M act on Z^2, the lattice of integral vectors, and then take into account the periodicity of the torus by identifying points whose coordinates differ by multiplies of q, i.e., we consider the factor group Z^2/gZ^2. Thus, the dynamics of periodic orbits is dynamics over a finite set of integers.

The work [28] illustrated the close link existing between arithmetic in algebraic number fields and strongly chaotic dynamics. The main conclusions of [28] may be summarized as follows:

- A 2-torus automorphism has three different types of (periodic) orbit structure, according to the classification of rational primes: inert, split, and ramified primes [29].
- The orbits which correspond to inert primes are almost without structure. The split primes have two distinct ideal factors, which correspond to orbits confined to invariant sublattices. For this reason, two ideal orbits which exist on split prime lattices are the "most ergodic" orbits and, thus, equilibrium averages computed with them minimize statistical fluctuations.
- Both inert and split prime lattices are found infinitely often and, moreover, with the same frequency in both cases. These are consequences of Dirichlet's theorem on the existence of infinity many primes in any arithmetic progression [30].
- The ramified prime lattices support orbits which are exceptionally regular. However, there is only a finite number of ramified primes, so that this apparently contradictory phenomenon of regularity in chaos is in fact very rare.

4 Floating Point Implementation of Cryptosystem Based on Chebyshev Polynomials

This public-key cryptosystem based on Chebyshev polynomials can be viewed as a generalization of the ElGamal public-key cryptosystem [31] and it is proposed in [21]. The floating point implementation of this cryptosystem, the prove of its correctness, its implementation and the security analysis are provided in [23].

4.1 Cryptosystem

The cryptosystem is composed of three algorithms: a Key Generation algorithm, an Encryption algorithm, and a Decryption algorithm.

Key Generation Algorithm: Key generation is done in three steps:

Alice, in order to generate the keys, does the following:

1. Generates a large integer s.
2. Selects a random number $x \in [-1, 1]$ and computes $T_s(x)$.
3. *Alice* sets her public key to $(x, T_s(x))$ and her private key to s.

Encryption Algorithm: Encryption requires five steps:

Bob, in order to encrypt a message, does the following:

1. Obtain Alice's authentic public key $(x, T_s(x))$.
2. Represents the message as a number $M \in [-1, 1]$.
3. Generates a large integer r.
4. Computes $T_r(x), T_{r \cdot s(x)} = T_r(T_s(x))$ and $X = M \cdot T_{r \cdot s}(x)$.
5. Sends the ciphertext $C = (T_r(x), X)$ to *Alice*.

Decryption Algorithm: Decryption requires two steps:

Alice, to recover the plaintext M from the ciphertext C, does the following:

1. Uses her private key s to compute $T_{s \cdot r} = T_s(T_r(x))$.
2. Recovers M by computing $M = X / T_{s \cdot r}(x)$.

4.2   Correctness of the Cryptosystem

The algorithm is correct due to the semi-group property of the Chebyshev polynomials. Indeed, encryption provides:

$$X = M \cdot T_r(T_s(x)).$$

Since Chebyshev polynomials commute under composition, it follows that

$$X = M \cdot T_s(T_r(x)).$$

Therefore

$$M = X / T_{s \cdot r}(x).$$

4.3   Implementation

Both encryption and decryption involve the evaluation of Chebyshev polynomials. If we evaluate Chebyshev polynomials directly, applying the recursive definition, then the computation of $T_n(x)$ takes linear time in n. However, it is possible to further reduce the computation to a logarithmic number of steps [32], by noticing that

$$T_{2 \cdot n}(x) = T_2(T_n(x))$$
$$T_{2 \cdot n+1}(x) = 2 \cdot T_{n+1}(x) \cdot T_n(x) - x$$

and re-organizing the computation. More precisely, we can use the recursive relation for evaluating Chebyshev polynomials

$$
\begin{aligned}
T_0 &= 1 \\
T_1 &= x \\
T_n(x) &= \begin{cases} 2 \cdot T_{n/2}^2(x) - 1, & \text{if } n \text{ is even} \\ 2 \cdot T_{(n-1)/2}(x) \cdot T_{(n+1)/2}(x) - x, & n \text{ odd}. \end{cases}
\end{aligned}
$$

Another important issue that must be considered when implementing the above cryptosystem is the finite precision of the arithmetics. In [21] the authors pointed out that the semi-group property of Chebyshev polynomials, stated by (3), holds only if the values s and r, chosen by Alice and Bob, are such that $s < s_0$ and $r < r_0$, where s_0 and r_0 are constant values depending on the arithmetics precision used in implementing the encryption and decryption algorithms. They gave a table where, for certain precisions, expressed in terms of bits, some possible upperbound for and hold. For example, a 2048-bit precision implies constants s_0 and r_0 smaller than 2^{970}. Such upperbounds where empirical determined. No general relation linking the arithmetic precision of the operations to the values of s_0 and r_0 is currently known.

4.4   Security Analysis of Cryptosystem

In this section, we show that the above cryptosystem is not secure. Given a cipher-text an adversary, by exploiting the same definition of Chebyshev polynomials and after some algebra, can recover the cleartext. In [21], it was presumed to be secure based on the following observation: as pointed out the scheme resembles ElGamal encryption scheme. The security of ElGamal encryption scheme is based on the intractability of the discrete logarithm problem in $\mathbb{Z}_n^*$, i.e., given n, x and x^p, find p. In the above scheme, given x and $T_p(x)$, the value $T_p(x)$ is the value of a polynomial of order p, *not just* a power x^p. Hence, computing the order of the polynomial p, given only one pair $(x, T_p(x)$ seems to be much harder than computing p from a power. Thus, recovering s given x and $T_s(x)$ seems only possible by computing $T_p(x)$ for all $p > 2$ and, then, comparing for which p the equality $T_p(x) = T_s(x)$ holds.

Unfortunately, there are some fundamental differences between the two schemes: the ElGamal scheme is implemented over $\mathbb{Z}_n^*$ and uses modular arithmetic. Then, given x and x^p the discrete logarithm is uniquely determined while, as we will show later, there are several Chebyshev polynomials passing through the same point.

4.4.1   How to Recover the Plaintext

In this section, we present an attack which enables an adversary to recover from a given ciphertext the corresponding cleartext.

First of all, we will use the trigonometric functions $\cos(x)$ and $\arccos(x)$ defined as

$$\cos : R \rightarrow [-1,1] \text{ and } \arccos : [-1,1] \rightarrow [0,\pi].$$

The $\cos(x)$ function has period 2π.

Notice that Chebyshev polynomials can be alternatively defined as follows.

Definition IV.1: Let n be an integer, and let x be a variable taking value over the interval [-1, 1]. The polynomial $T_n(x) : [-1,1] \rightarrow [-1,1]$ is defined as

$$T_n(x) = \cos(n \cdot \arccos(x)). \tag{6}$$

A simple trigonometric argument shows that this is equivalent to (2).

Description of the Attack: Let $(x, T_s(x))$ be Alice's public key. In order to encrypt a message M, Bob chooses a large integer r and computes

$$T_r(x), \quad T_{r \cdot s}(x) = T_r(T_s(x)), \qquad and \qquad X = M \cdot T_{r \cdot s}(x).$$

Then, he sends the cipher-text $C = (T_r(x), X)$ to Alice.

Unfortunately an adversary, given Alice's public key $(x, T_s(x))$ and the ciphertext $(T_r(x), X)$, can recover M by doing the following steps:

1. Compute an r' such that $T_{r'}(x) = T_r(x)$.
2. Evaluate $T_{r' \cdot s}(x) = T_{r'}(T_s(x))$.
3. Recover $M = (X / T_{r' \cdot s}(x))$.

The attack is always successful because, if r' is such that $T_{r'}(x) = T_r(x)$, then:

$$
\begin{aligned}
T_{r \cdot s}(x) &= T_{s \cdot r} \\
&= T_s(T_r(x)) = T_s(T_{r'}(x)) \\
&= T_{s \cdot r'}(x) = T_{r' \cdot s}(x) \\
&= T_{r'}(T_s(x)).
\end{aligned}
$$

Let us show how such an r' can be computed. Let $\mathbb{N}$ be the set of natural numbers and let $\mathbb{Z}$ be the set of integers. According to Definition IV.1, it holds that $T_r(x) = \cos(r \cdot \arccos(x))$. Let

$$
\mathbb{P} = \left\{ \left. \frac{\pm \arccos(T_r(x)) + 2k\pi}{\arccos(x)} \right| \; k \in \mathbb{Z} \right\}.
$$

Notice that some r' belonging to the set $\mathbb{P}$ might not be integers. However, the following result shows that $\mathbb{P}$ contains *all* possible integers r' defining polynomials $T_{r'}(x)$ passing through $T_r(x)$.

Lemma IV.1: For each pair $(x, T_r(x))$, the integer r' satisfies $T_{r'}(x) = T_r(x)$ if and only if $r' \in \mathbb{P} \cap \mathbb{N}$.

Proof: Let $r' \in \mathbb{P} \cap \mathbb{N}$. Assume that

$$
r' = \frac{\arccos(T_r(x)) + 2k'\pi}{\arccos(x)}
$$

for a certain k'. By using Definition IV.1, it holds that

$$
\begin{aligned}
T_{r'}(x) &= \cos(r' \arccos(x)) \\
&= \cos\left(\frac{\arccos(T_r(x)) + 2k'\pi}{\arccos(x)} \cdot \arccos(x) \right) \\
&= \cos(\arccos(T_r(x)) + 2k'\pi) \\
&= \cos(\arccos(T_r(x))) \\
&= T_r(x).
\end{aligned}
$$

Hence, if $r' \in \mathbb{P} \cap \mathbb{N}$, then $T_{r'}(x) = T_r(x)$. If $r' = (-\arccos(T_r(x)) + 2k'\pi)/(\arccos(x))$ we can apply exactly the same argument.

On the other hand, assume that $T_{r'}(x) = T_r(x)$ for a certain $r' \in \mathbb{N}$. Then,

$$
T_{r'}(x) = cos(r' \arccos(x)) = T_r(x).
$$

Applying the arccos function to both members, we get

$$\arccos(\cos(r' \arccos(x))) = \arccos(T_r(x)). \tag{7}$$

Let $y = \arccos(w)$. Due to the equality $\cos(-\beta) = \cos(\beta)$, for every angle β, and due to the periodicity of the cos function, all angles β such that $\cos(\beta) = w$ are given by $\beta = \pm y + 2k\pi$, for $k \in \mathbb{Z}$. Therefore, identity (7) holds if and only if

$$r' \arccos(x) = \pm \arccos(T_r(x)) + 2k'\pi$$

where $k' \in \mathbb{Z}$. Dividing both members by $\arccos(x)$, we get:

$$r' = \frac{\pm \arccos(T_r(x)) + 2k'\pi}{\arccos(x)}$$

i.e., $r' \in \mathbb{P} \cap \mathbb{N}$. Thus, the lemma holds.

Using the above result, denoting by

$$a = \frac{\arccos(T_r(x))}{\arccos(x)} \quad \text{and} \quad b = \frac{2\pi}{\arccos(x)} \tag{8}$$

the adversary has to find an integer $k \in \mathbb{Z}$ and a positive integer $u \in \mathbb{N}$ solutions to one of the two equations

$$a + k \cdot b = u \quad or \quad -a + k \cdot b = u \tag{9}$$

given a and b.

Let $(a \bmod 1)$ and $(b \bmod 1)$ be the fractional parts of and a and b. The actual problem becomes solving

$$(a \bmod 1) + k \cdot (b \bmod 1) = z$$

or

$$-(a \bmod 1) + k \cdot (b \bmod 1) = z$$

How to find k in a real implementation. Assume that we use a finite precision implementation in base $B \geq 2$, and that L is the maximum number of digits of $(a \bmod 1)$ and $(b \bmod 1)$. Then, multiplying all terms by B^L, we can rewrite the above equations in equivalent form as

$$(a \bmod 1) \cdot B^L + k \cdot (b \bmod 1) \cdot B^L = z \cdot B^L$$

and

$$-(a \bmod 1) \cdot B^L + k \cdot (b \bmod 1) \cdot B^L = z \cdot B^L$$

Denoting by a' the integer $(a \bmod 1) \cdot B^L$ and by b' the integer $(b \bmod 1) \cdot B^L$, the solutions to the above equations are exactly the solutions to the linear modular equations

$$b' \cdot k \equiv a' \mod B^L \quad \text{and} \quad b' \cdot k \equiv -a' \mod B^L. \tag{10}$$

However, notice that we can restrict our attention to just one of the above modular equations. Indeed, since $b' \cdot k \equiv -a' \mod B^L$ is equivalent to $b' \cdot (-k) \equiv a' \mod B^L$, once we have solved $b' \cdot k \equiv a' \mod B^L$, we easily derive the solutions to the second one. More precisely, if k is solution to $b' \cdot k \equiv a' \mod B^L$ then $-k$ is solution to $b' \cdot k \equiv -a' \mod B^L$.

We can get efficiently the set of solutions to linear modular equations of the form $b'k \equiv a' \mod B^L$ (see, for example, [9, ch. 33]). Denoting by $\langle b' \rangle = \{b'j \mod B^L | j \in Z_{B^L}\}$ the subgroup of elements of $Z_{B^L}^*$ generated by b', it is easy to see that the modular equation has solutions if and only if $a' \in \langle b' \rangle$. Moreover, denoting by d the $\gcd(b', B^L)$, the above membership condition is equivalent to $d|a'$. The set of distinct solutions to $b'k \equiv a' \mod B^L$ (if there exist) has cardinality d and is given by

$$x_j = x_0 + j \cdot \frac{B^L}{d} \mod B^L, \quad \text{for } j = 1, ..., d-1$$

where the first solution x_0 can be obtained directly by applying the extended Euclidean algorithm. Indeed, such an algorithm, on input (b', B^L), outputs a triple (d, s', t') of integers where $d = b's' + B^L t'$, and it is easy to check that $x_0 = s'(a'/d)$ is solution to $b'k \equiv a' \mod B^L$. From a computational point of view, the above procedure is efficient since the running time of the extended Euclidean algorithm requires $O(\log B^L)$ steps in the worst case.

Coming back to our setting, notice that the equations given in (10) have solutions by *construction*. More precisely, there are $d = \gcd(b', B^L)$ exactly distinct solutions for each of them, which can be easily found applying the above method. Clearly, just one solution suffices to the adversarys goal.

4.4.2   An Example

We show how an adversary, given Alices public key $(x, T_s(x))$ and the ciphertext $C = (T_r(x), X)$, where $X = M \cdot T_{rs}(x)$, constructed by Bob in order to send M to Alice, computes the value $T_{rs}(x)$. Then, dividing X by $T_{rs}(x)$, he recovers M.

Let us start by generating Alices public-key parameters.

Let $B = 10, \pi = 3.141592654, x = 0.64278761$ and $s = 106000$. Then, $\arccos(x) = (5/18)\pi$, and $T_s = \cos(s \cdot \arccos x) = \cos(106000 \cdot (5/18)\pi) = 0.173648178$. Hence, Alices public key is given by the pair

$$(x, T_s(x)) = (0.64278761, 0.173648178).$$

Assume that Bob, in order to encrypt a message M, chooses $r = 81500$. Then

$$T_r(x) = \cos(r \cdot \arccos x)$$
$$= \cos\left(81500 \cdot \frac{5}{15}\pi\right)$$
$$= -0.939692621$$

and

$$T_r(T_s(x)) = \cos(r \cdot \arccos(T_s(x)))$$

$$= \cos\left(81500 \cdot \frac{4}{9}\pi\right)$$

$$= 0.766044443$$

By applying the strategy described before, an adversary computes an r' such that $T_{r'}(T_s(x)) = T_r(T_s(x))$. Since it holds that

$$\arccos(T_r(x)) = \frac{8\pi}{9} \quad \text{and} \quad \arccos(x) = \frac{5\pi}{18}$$

the set of possible integer indexes r' is given by

$$\mathbb{P} = \left\{ \frac{\pm \arccos(T_r(x))}{\arccos(x)} + \frac{2\pi k}{\arccos(x)} \,\middle|\, k \in \mathbb{Z} \right\}$$

$$= \{\pm 3.2 + 7.2k \mid k \in \mathbb{Z}\}.$$

Hence, the adversary has to find a solution to one of the following two equations:

$$3.2 + 7.2k_1 = u_1 \quad \text{and} \quad -3.2 + 7.2k_2 = u_2 \tag{11}$$

where $u_1, u_2 \in \mathbb{N}$. By considering only the fractional parts, the problem becomes solving one of

$$0.2 + 0.2k_1 = z_1 \quad \text{or} \quad -0.2 + 0.2k_2 = z_2$$

where $z_1, z_2 \in \mathbb{N}$. Since $L = 1$, then $B^L = 10$, and the above equations are equivalent to

$$2 + 2k_1 = 10z_1 \quad \text{and} \quad -2 + 2k_2 = 10z_2$$

whose solutions are exactly the solutions to the modular equations

$$2k \equiv 8 \bmod 10 \quad \text{and} \quad 2k \equiv 2 \bmod 10. \tag{12}$$

Let us consider the first one. This equation has solutions since $\gcd(2, 10) = 2$ and $2 | 8$. Precisely, there are 2 solutions, given by $k = 4 + i5$, for $i = 0, 1$, where 4 is the solution x_0 obtained directly by means of the Extended Euclidean Algorithm. By choosing one of them, for example 4, the corresponding index r', computed evaluating the first one of (11) is 32. Then, it holds that

$$T_{32}(T_s(x)) = \cos(32 \cdot \frac{4}{9}\pi) = 0.766044443.$$

Hence, the adversary has computed $T_{rs}(x)$. The cleartext sent by Bob is computed by the adversary as $X/T_{32s}(x)$.

For completeness, notice the two solutions to the second equation are $\{6,1\}$ and are obtained by computing $-4 \bmod 10$ and $-9 \bmod 10$. By choosing one of them, for example 1, the corresponding index r', computed evaluating the second of (11) is 4. Then, it holds that

$$T_4(T_s(x)) = \cos(4 \cdot \frac{4}{9}\pi) = 0.766044443.$$

Hence, the adversary has computed $T_{rs}(x)$. The cleartext sent by Bob is computed by the adversary as $X/T_{4s}(x)$.

5 Floating Point Implementation of Cryptosystem Based on Jacobian Elliptic Chebyshev Rational Maps

As suggested in [21], instead of using Chebyshev polynomials, the cryptosystem we have previously analised can be also realized by using the *Jacobian Elliptic Chebyshev Rational Maps*, studied in [33] and [34]. This section is dedicated to the implementation of this type of cryptosystem and attacks that can be identified for these cryptosystems. This work was done and explained in details in [23].

5.1 Jacobian Elliptic Chebyshev Rational Maps

The Jacobian Elliptic Chebyshev Rational Maps are rational functions defined as follows [34].

Definition 5.1: Let p be a positive integer, let $\omega \in [-1,1]$ be a real number, and let $k \in [0,1]$ be a real number called modulus. Jacobian Elliptic Chebyshev Rational Maps are defined by

$$R_{p+1}(\omega,k) = \frac{2\omega}{1 - k^2(1 - R_p(w,k)^2)(1 - \omega^2)} R_p(w,k) - R_{p-1}(w,k)$$

where $R_0(w,k) = 1$ and $R_1(w,k) = \omega$.

Notice that, when the modulus $k = 0$ the Jacobian Elliptic Chebyshev Rational Map $R_p(\omega,0)$ is exactly a Chebyshev polynomial, i.e., $R_p(\omega,0) = T_p(\omega)$.

Jacobian Elliptic Chebyshev Rational Maps enjoy the *semigroup property*. Indeed, for each integers $r,s \geq 2$, and for each w,k, it holds that

$$R_r(R_s(\omega,k),k) = R_{r \cdot s}(\omega,k). \tag{13}$$

Hence, these maps commute under composition, i.e.,

$$R_r(R_s(\omega,k),k) = R_s(R_r(\omega,k),k).$$

5.2 *Cryptosystem*

The cryptosystem is composed of three algorithms: a Key Generation algorithm, an Encryption algorithm, and a Decryption algorithm.

Key Generation Algorithm: Key Generation takes place in three steps:

Alice, in order to generate the keys, does the following:

1. Generates a large integer s.
2. Selects two random numbers $\omega \in [-1,1]$ and $k \in [0,1]$, and computes $R_s(\omega,k)$.
3. *Alice* sets her public key to $(\omega,k,R_s(\omega,k))$ and her private key to s.

Encryption Algorithm: Encryption requires five steps:

Bob, in order to encrypt a message, does the following:

1. Obtain Alice's authentic public key $(\omega,k,R_s(\omega,k))$.
2. Represents the message as a number $M \in [-1,1]$.
3. Generates a large integer r.
4. Computes $R_r(\omega,k), R_{r \cdot s(\omega,k)} = R_r(R_s(\omega,k),k)$ and $X = M \cdot R_{r \cdot s}(\omega,k)$.
5. Sends the ciphertext $C = (R_r(\omega,k),X)$ to *Alice*.

Decryption Algorithm: Decryption requires two steps:

Alice, to recover the plaintext M from the ciphertext C, does the following:

1. Uses her private key s to compute $R_{s \cdot r}(\omega,k) = R_s(R_r(\omega,k),k)$.
2. Recovers M by computing $M = X/R_{s \cdot r}(\omega,k)$.

Notice that, the value of k, which defines the form of the map, could be the same for all users of the system.

5.3 *Correctness of the Cryptosystem*

The cryptosystem is correct due to the semi-group property of the Jacobian Elliptic Chebyshev Rational Maps. Indeed, encryption provides:

$$X = M \cdot R_r(R_s(\omega,k),k).$$

Since the maps commute under composition, it follows that

$$X = M \cdot R_s(R_r(\omega, k), k).$$

Therefore

$$M = X / R_{s \cdot r}(\omega, k).$$

5.4 Jacobian Elliptic Functions and Jacobian Elliptic Chebyshev Rational Maps

Jacobian elliptic Chebyshev rational maps can be equivalently defined by means of the Jacobian elliptic functions [34].

Let $\omega \in [-1, 1]$, let $k \in [0, 1]$, and let $\rho \in [0, 2\pi]$ be the angle, referred to as the *amplitude* of ω, defined by

$$\omega = \int_0^\rho \frac{d\theta}{(1 - k^2 \cdot \sin^2(\theta))^{\frac{1}{2}}}.$$

Then, the Jacobian elliptic functions $sn(\omega, k)$ and $cn(\omega, k)$ are defined as follows:

$$sn(\omega, k) = \sin(\rho) \text{ and } cn(w, k) = \cos(\rho)$$

and Let $k' = \sqrt{(1 - k^2)}$. The above functions are doubly-periodic, having a real period and an imaginary one. More precisely, denoting by

$$K = \int_0^{\frac{\pi}{2}} \frac{d\theta}{(1 - k^2 \cdot \sin^2(\theta))^{\frac{1}{2}}}$$

and

$$iK' = i \int_0^{\frac{\pi}{2}} \frac{d\theta}{(1 - k'^2 \cdot \sin^2(\theta))^{\frac{1}{2}}}$$

where i is the imaginary unit, we get that $sn(\omega, k)$ has periods $4K$ and $2iK'$; while $cn(\omega, k)$ has periods $4K$ and $2K + 2iK'$. We restrict our attention to the real periodicity.

For any fixed k, the function $cn^{-1}(v, k)$, inverse of the Jacobian elliptic function $cn(\omega, k)$, relatively to the interval $[0, 2K]$, is given by

$$cn^{-1}(v, k) = \int_0^\rho \frac{d\theta}{(1 - k^2 \cdot \sin^2(\theta))^{\frac{1}{2}}}$$

where $\rho = \arccos(v)$.

Then, we can state the following alternative definition for the Jacobian elliptic Chebyshev rational maps.

Definition 5.2: Let $p \geq 2$ be an integer, let $k \in [0, 1]$ be a real number, and let $\omega \in [-1, 1]$. The Jacobian elliptic Chebyshev rational maps with modulus k are defined by

$$R_p(\omega, k) = cn(p \cdot cn^{-1}(\omega, k), k).$$

5.5   *Efficient Computation of* $cn(\omega,k)$, $sn(\omega,k)$ *and* $cn^{-1}(v,k)$

The functions $cn(\omega,k)$, $sn(\omega,k)$, and $cn^{-1}(v,k)$, all defined in terms of elliptic integrals, can be efficiently computed by means of the *Arithmetic-Geometric Method*, (A.G.M. method, for short). Roughly speaking, such a method works as follows: starting with (a_0,b_0), it proceeds to determine number triples

$$(a_1,b_1,c_1),(a_2,b_2,c_2),...,(a_n,b_n,c_n)$$

according to the following scheme of arithmetic and geometric mean:

$$a_{j+1} = (1/2)(a_j+b_j)$$
$$b_{j+1} = (a_j \cdot b_j)^{(1/2)}$$
$$c_{j+1} = (1/2)(a_j-b_j).$$

Assume that we use an arithmetic in base B with N-digit precision of the operations. The procedure stops at the n-th step when $a_n = b_n$, i.e., when $c_n = 0$. Notice that such an equality is achieved when the relative error $\varepsilon_n = 1 - (b_n/a_n)$ is less than the degree of accuracy fixed by the implementation i.e., B^{-N}. It has been estimated (see, for example [35]) that the relative error $\varepsilon_j = 1 - (b_j/a_j)$ decays approximatively as $\varepsilon_j \approx (1/8)e^{-2^j}$, from which it easily follows that the method converges after roughly $\log N$ steps.

To compute the functions $cn(\omega,k)$ and $sn(\omega,k)$, we apply the A.G.M method starting with $a_0 = 1$, and $b_0 = k'$. Once the A.G.M method stops, we compute the angle (in degrees) $\phi_n = 2^n a_n \omega(180/\pi)$. Then, applying, for $j = n,...,1$, the recurrence relation $\sin(2\phi_{j-1} - \phi_j) = (c_j/a_j)\sin\phi_j$, we compute the angles $\phi_{n-1},\phi_{n-2},...,\phi_0$. Finally,

$$sn(\omega,k) = \sin\phi_0 \quad \text{and} \quad cn(\omega,k) = \cos\phi_0.$$

On the other hand, to evaluate $cn^{-1}(v,k)$, $for j = 0,...,n-1$, by applying the recurrence relation $\tan(\gamma_{j+1} - \gamma_j) = (b_j/a_j)\tan\gamma_j$, where $\gamma_0 = \rho$, we compute the angles $\gamma_1,...,\gamma_n$, and then

$$cn^{-1}(v,k) = \frac{\gamma_n}{2^n a_n}.$$

Notice that the quarter-period K can be easily computed as well, since it is a special case of the computation of $cn(\omega,k)$ (just set the angle $\rho = 2\pi$). The reader is referred to [36] for further details on the A.G.M method, and on the computation of $sn(\omega,k)$, $cn(\omega,k)$ and $cn^{-1}(v,k)$. Moreover, an efficient implementation of the above functions can be found in [37].

5.6   *Security Analysis of Cryptosystem*

Apart the complexity of the mathematical objects we are dealing with, the attack we have applied against the public-key scheme based on Chebyshev polynomials still works against the cryptosystem based on Jacobian elliptic Chebyshev rational maps.

5.6.1 How to Recover the Plaintext

Let $(\omega,k,R_s(\omega,k))$ be Alices public key. In order to encrypt a message M, Bob chooses a large integer r and computes

$$R_r(\omega,k), \quad R_{r\cdot s}(\omega,k) = R_r(R_s(\omega,k),k) \quad \text{and} \quad X = M \cdot R_{r\cdot s}(\omega,k).$$

Then, he sends the ciphertext $C = (R_r(\omega,k),X)$ to Alice.

Unfortunately an adversary, given Alice's public key $(\omega,k,R_s(\omega,k))$ and the ciphertext $C = (R_r(\omega,k),X)$, can recover M as follows:

> The adversary, to get the message, does the following
>
> 1. Compute an r' such that $R_{r'}(\omega,k) = R_r(\omega,k)$.
> 2. Evaluate $R_{r'\cdot s}(\omega,k) = R_{r'}(R_s(\omega,k),k)$.
> 3. Recover $M = (X/R_{r'\cdot s}(\omega,k))$.

The attack is always successful because, if r' is such that $R_{r'}(\omega,k) = R_r(\omega,k)$, then:

$$\begin{aligned}
R_{r\cdot s}(\omega,k) &= R_{s\cdot r}(\omega,k) \\
&= R_s(R_r(\omega,k),k) \\
&= R_s(R_{r'}(\omega,k),k) \\
&= R_{s\cdot r'}(\omega,k) \\
&= R_{r'\cdot s}(\omega,k) \\
&= R_{r'}(R_s(\omega,k),k).
\end{aligned}$$

Let us show how such an r' can be computed. According to Definition V.2, it holds that

$$R_r(\omega,k) = cn(r \cdot cn^{-1}(\omega,k),k).$$

Hence applying the cn^{-1} function to both members of the equality, and using the periodicity of $cn(\omega,k)$ and the property $cn(\omega,k) = cn(-\omega,k)$ we get that

$$\pm cn^{-1}((R_r(\omega,k),k) + z \cdot 4K = r \cdot cn^{-1}(\omega,k)$$

for $z \in \mathbb{Z}$. Notice that we are only considering the real periodicity, since we are not interested in imaginary solutions. Let

$$\mathbb{P} = \left\{ \left. \frac{\pm cn^{-1}(R_r(\omega,k),k) + z \cdot 4K}{cn^{-1}(\omega,k)} \right| \ z \in \mathbb{Z} \right\}.$$

We can show that $\mathbb{P}$ contains *all* possible integers r' defining maps $R_{r'}(\omega,k)$ passing through $R_r(\omega,k)$, for certain r, ω and k. The proof proceeds along the same lines of proof provided for Lemma IV.1. We omit it since it is essentially the same.

Lemma 5.1: For each triple $(\omega,k,R_r(\omega,k))$, the integer r' satisfies $R_{r'}(\omega,k) = R_r(\omega,k)$ if and only if $r' \in \mathbb{P} \cap \mathbb{N}$.

Setting $a = (cn^{-1}(R_r(\omega,k),k))/(cn^{-1}(\omega,k))$ and $b = (4K/cn^{-1}(\omega,k),k)$ as in (8), we apply *exactly* the same steps we have done in Subsection 4.4.1 describing the attack against the cryptosystem based on Chebyshev polynomials. Hence, an adversary can recover the plaintext from the ciphertext.

5.6.2   An Example

We show how an adversary, given Alices public key $(\omega,k,T_s(\omega,k))$ and the ciphertext $C = (R_r(\omega,k),X)$, where $X = M \cdot R_{rs}(\omega,k)$, constructed by Bob in order to send M to Alice, computes the value $R_{rs}(\omega,k)$. Then, dividing X by $R_{rs}(\omega,k)$, he recovers M.

Let us start by generating Alices public-key parameters.

Let $B = 10$, $\omega = 0.435946$, $k = 0.3$ and $s = 2342$. Then, $R_s(\omega,k) = cn(s \cdot cn^{-1}(\omega,k),k) = 0.245756$. Hence, Alices public key is given by the triple

$$(\omega,k,R_s(\omega,k)) = (0.435946, 0.3, 0.245756). \tag{14}$$

Assume that Bob, in order to encrypt a message M, chooses $r = 1876$. Then

$$R_r(\omega,k) = cn(r \cdot cn^{-1}(\omega,k),k) = -0.938538$$

and

$$R_r(R_s(\omega,k),k) = cn(r \cdot cn^{-1}(R_s(\omega,k),k),k) = 0.613408.$$

By applying the strategy described in Subsection 5.6.1, an adversary computes an r' such that $R_{r'}(R_s(\omega,k),k) = R_r(R_s(\omega,k),k)$. The set of possible integer indexes r' is given by

$$\mathbb{P} = \left\{ \frac{\pm cn^{-1}(R_r(\omega,k),k) + z \cdot 4K}{cn^{-1}(\omega,k)} \;\middle|\; z \in \mathbb{Z} \right\}$$
$$= \{\pm 2.6 + 5.8k \mid k \in \mathbb{Z}\}.$$

Hence, the adversary has to find a solution to one of the following two equations:

$$2.6 + 5.8k_1 = u_1 \quad \text{and} \quad -2.6 + 5.8k_2 = u_2 \tag{15}$$

where $u_1, u_2 \in \mathbb{N}$. By considering only the fractional parts, the problem becomes solving one of

$$0.6 + 0.8k_1 = z_1 \quad \text{or} \quad -0.6 + 0.8k_2 = z_2$$

where $z_1, z_2 \in \mathbb{N}$.

Since $L = 1$, then $B^L = 10$, and the above equations are equivalent to

$$6 + 8k_1 = 10z_1 \quad \text{and} \quad -6 + 8k_2 = 10z_2 \tag{16}$$

whose solutions are exactly the solutions to the modular equations

$$8k \equiv 4 \bmod 10 \quad \text{and} \quad 8k \equiv 6 \bmod 10.$$

Let us consider the first one. This equation has solutions since $\gcd(8,10) = 2$ and $2|4$. Precisely, there are 2 solutions, given by $k = 3 + i5$, for $i = 0,1$, where 3 is the solution x_0 obtained directly by means of the Extended Euclidean Algorithm. By choosing one of them, for example 3, the corresponding index r', computed evaluating the first one of (15) is 20. Then, it holds that

$$R_{20}(R_s(\omega,k),k) = cn(20 \cdot cn^{-1}(R_s(\omega,k),k),k) = 0.613408.$$

Hence, the adversary has computed $R_{rs}(\omega,k)$. The cleartext sent by Bob is computed by the adversary as $X/R_{20s}(\omega,k)$.

5.7 Key Agreement by Using Rational Maps

Rational maps enjoying the semi-group property can be also used to design a Diffie-Hellman like key agreement scheme. Umeno [38] was the first author who suggested such a method. Let us briefly recall the following definitions, given in [5].

Definition 5.3: Key establishment is any process whereby a shared secret key becomes available to two or more parties, for subsequent cryptographic use.

Definition 5.4: A key agreement protocol or mechanism is a key establishment technique in which a shared secret is derived by two or more parties as a function of information contributed by, or associated with, each of these, ideally such that no party can predetermine the resulting value.

Let us look at the following key agreement protocol.

Let X be a public real value, and let $F(\cdot,\cdot)$ be a rational map enjoying the semi-group property, i.e., $F(p,F(q,x)) = F(pq,x)$.

Bob, in order to agree on a common key with Alice, does the following:

1. Generates a large integer p.
2. Computes $Y = F(p,X)$.
3. Sends Y to Alice.

   Alice, in order to agree on a common key with *Bob*, does the following:

1. Generates a large integer q.
2. Computes $Y' = F(q,X)$.
3. Sends Y' to Bob.

Then, *Alice* and *Bob* compute the common value $Z = F(q,Y) = F(q,F(p,X)) = F(p,F(q,X)) = F(p,Y')$.

It is easy to check that if the rational map used in the above scheme is a Chebyshev Polynomial or a Jacobian Elliptic Chebyshev Rational map then, since X is public and $F(p,X)$ and $F(q,X)$ are sent in clear over the channel, an adversary who taps the channel, with no knowledge of the secret values p and q, can employ the same attack we have described before for the public-key cryptosystem, and compute the common key.

5.8   Entity Authentication Based on Chebyshev Polynomials

Chebyshev Polynomials have also been used to design an authentication scheme. Entity authentication is defined as follows [5].

Definition 5.5: Entity authentication is the process whereby one party is assured (through acquisition of corroborative evidence) of the identity of a second party involved in a protocol, and that the second has actually participated (i.e., is active at, or immediately prior to, the time evidence is acquired).

In [39], a scheme based on Chebyshev Polynomials, by means of which a user can efficiently authenticate himself to a server in order to log in, was proposed. It strongly resembles the public-key cryptosystem described in [21]. Apart minor implementation details, the scheme works as follows.

Let $m \in [-1, 1]$ be a real value, and denote by $T_s^i(\cdot)$ the map $T_s(\cdot)$ iterated i times, i.e., $T_s^{i'}(\cdot) = T_s(T_s(T_s...T_s(\cdot))...) = T_{s^i}(\cdot)$.

Setup Phase - Server Side

1. The server generates a random number r.
2. Computes and sends $T_r(m)$ to the user.

Setup Phase - User Side

1. The user chooses a random number s.

i-th Authentication Phase

1. The user computes $T_s^i(m)$, and $auth = T_s^i(T_r(m))$, and sends both values to the server.
2. The server computes $auth' = T_r(T_s^i(m))$ and checks whether $auth = auth'$. Then, if check is satisfied, the access is granted.

It is easy to see that, if m and $T_r(m)$ are public, an adversary who gets the messages associated with the first log in request, can apply the same attack we have described before in order to get an integer s' such that $T_{s'}(m) = T_s(m)$. Then, at the i-th session, he can authenticate himself as the real user by computing $T_s^i(m)$, and $auth = T_{s'}^i(T_r(m))$. Indeed, it is easy to show, arguing by induction on i, that $T_{s'}^i(m) = T_s^i(m)$.

Therefore, it holds that

$$\begin{aligned} auth &= T^i_{s'}(T_r(m)) \\ &= T_r(T^i_{s'}(m)) \\ &= T_r(T^i_s(m)) \\ &= T^i_s(T_r(m)) \\ &= auth'. \end{aligned}$$

Thus, the scheme is not secure. One way to avoid the above attack is to make m and $T_r(m)$ private to the user and the server. Unfortunately, the scheme is not secure *even if* m and $T_r(m)$ are private. Indeed, even in this scenario, an adversary with no knowledge of the private values m and $T_r(m)$, who just listen to *two consecutive authentication phases*, can subsequently authenticate himself to the server as it were the real user. More precisely, assume that the adversary gets $T^{i-1}_s(m)$, $T^{i-1}_s(T_r(m))$ and $T^i_s(m)$, $T^i_s(T_r(m))$. Then, the attack works as follows:

The adversary does the following:
1. Computes an integer w such that $T_w(T^{i-1}_s(m)) = T^i_s(m)$.
2. For any $l \geq 1$, to authenticate himself at the $(i+l)$-th session,
    (a) Computes $T^{i+l}_s(m) = T^l_w(T^i_s(m))$ and $auth = T^{i+l}_s(T_r(m)) = T^l_w(T^i_s(T_r(m)))$.
    (b) Sends the pair $(T^{i+l}_s(m), auth)$.

Notice that the adversary *does not* need to know the index i of the session. He just needs two consecutive authentication messages.

In order to understand why the attack works, notice that an integer w such that can $T_w(T_{s^{i-1}}(m)) = T_{s^i}(m)$ be computed by applying the same attack we have described before against the cryptosystem. Then, we can proceed by induction on l to show that

$$T^{i+l}_s(m) = T^l_w(T^i_s(m))$$

and

$$auth = T^{i+l}_s(T_r(m)) = T^l_w(T^i_s(T_r(m))).$$

Let $l = 1$. It is easy to see that

$$\begin{aligned} T^{i+1}_s(m) &= T_s(T_{s^i}(m)) \\ &= T_s(T_w(T_{s^{i-1}}(m))) \\ &= T_w(T_s(T_{s^{i-1}}(m))) \\ &= T_w(T_{s^i}(m)). \end{aligned}$$

Then, notice that

$$T_w(T_{s^{i-1}}(T_r(m))) = T_{s^i}(T_r(m)).$$

Indeed

$$
\begin{aligned}
T_w(T_{s^{i-1}}(T_r(m))) &= T_w(T_r(T_{s^{i-1}}(m))) \\
&= T_r(T_w(T_{s^{i-1}}(m))) \\
&= T_r(T_{s^i}(m)) \\
&= T_{s^i}(T_r(m)).
\end{aligned}
$$

Therefore

$$
\begin{aligned}
T_s^{i+1}(T_r(m)) &= T_s(T_{s^i}(m)) \\
&= T_s(T_w(T_{s^{i-1}}(T_r(m)))) \\
&= T_w(T_s(T_{s^{i-1}}(T_r(m)))) \\
&= T_w(T_{s^i}(T_r(m))).
\end{aligned}
$$

Assume that

$$
T_{s^{i+(l-1)}}(m) = T_{w^{(l-1)}}(T_{s^i}(m))
$$

and

$$
T_{s^{i+(l-1)}}(T_r(m)) = T_{w^{(l-1)}}(T_{s^i}(T_r(m))).
$$

By applying the inductive hypothesis, it holds that

$$
\begin{aligned}
T_s^{i+l}(m) &= T_s(T_{s^{i+(l-1)}}(m)) \\
&= T_s(T_{w^{(l-1)}}(T_{s^i}(m))) \\
&= T_s(T_{w^{(l-1)}}(T_w(T_{s^{i-1}}(m)))) \\
&= T_{w^l}(T_{s^i}(m)).
\end{aligned}
$$

and

$$
\begin{aligned}
T_s^{i+l}(m) &= T_s(T_{s^{i+(l-1)}}(T_r(m))) \\
&= T_s(T_{w^{(l-1)}}(T_{s^i}(T_r(m)))) \\
&= T_s(T_{w^{(l-1)}}(T_w(T_{s^{i-1}}(T_r(m))))) \\
&= T_{w^l}(T_{s^i}(T_r(m))).
\end{aligned}
$$

Thus, the attack works.

6 Integer Implementation of Cryptosystem Based on Chebyshev Polynomials

In this section we give three reasons why to use integer arithmetic instead of using floating-point arithmetic when implementing cryptosystem based on Chebyshev polynomials. In order to do this we present a modified Chebyshev polynomial, given in [24], and show how this polynomial can be implemented in software when its parameters are large integers. Furthermore, using the work in [24] we present

an implementation and give examples of ElGamal and RSA public-key encryption using modified Chebyshev polynomials.

6.1   *Floating-Point Arithmetic versus Integer Arithmetic*

Chaotic systems are defined on real numbers. Any encryption algorithm which uses chaotic maps when implemented on a computer (finite-state machine) becomes a transformation from a finite set onto itself. Because of its wide dynamic range, the floating-point implementation seems to be the most appropriate for software realizations (implementation) of Chebyshev polynomials. However, there are three reasons for not using floating-point arithmetic in public-key encryption.

First, floating-point numbers are not uniformly distributed over any given interval of the real axis [40]. Furthermore, one may observe the existence of redundant number representations. Indeed, due to the normalized calculations in floating-point arithmetic, some floating-point numbers represent the same real signal value.

Second, noninvertibility of Chebyshev polynomials and their floating-point implementation imply a restriction on the length of the message. Indeed, the public-key encryption scheme proposed recently in [21], [22] can be viewed as a generalization of ElGamal public-key scheme using Chebyshev polynomials. We summarize the algorithm as follows. Alice generates a large integer s, selects a random number $x \in [-1, 1]$, and computes $T_s(x)$. Alice's public key is $(x, T_s(x))$, while her private key is s. Bob represents the message as a number $M \in [-1, 1]$, generates a large integer r, and computes $T_r(x)$, $T_{rs} = T_r(T_s(x))$, and $X = MT_{rs}$. He sends the ciphertext $c = (T_r(x), X)$ to Alice. To recover plain-text M from c, Alice should use the private key s to compute $T_{sr} = T_s(T_r(x))$, and recovers M by computing $M = X/T_{sr}$. Let l_s, l_r, l_M be the lengths (in bits) of s, r, and M, respectively, and let N-bit precision arithmetic be used in a software implementation of the algorithm. Then $lm \leq N - ls - lr$ [21], [22].

Third, the authors think that the most important reason is that there are no analytical tools for understanding the periodic structure of the periodic orbits in the floating-point implementation of chaotic maps (when implemented on a computer all chaotic maps are periodic: all trajectories are eventually periodic). On the other hand, when using integers one may hope that a possible link between number theory and chaos theory has been established, as in the case of the toral automorphisms, to understand the structure of the orbits.

6.2   *Modified Chebyshev Polynomials*

In this section we use the following map, $T_p : 0, 1, ..., N-1 \rightarrow 0, 1, ..., N-1$ defined as

$$y = T_p(x)(\bmod N) \tag{17}$$

where x and N are integers, to extend ElGamal and RSA public-key algorithms to Chebyshev maps. We call (17) a modified Chebyshev polynomial.

The modified Chebyshev polynomials can replace powers in ElGamal and/or RSA public-key algorithms only if they commute under composition and if one can compute the period of their orbits. The following two theorems show that these properties hold for modified Chebyshev polynomials.

Theorem 6.1. *Modified Chebyshev polynomials commute under composition, that is,*

$$T_p(T_q(x)(\mathrm{mod}\,)N))(\mathrm{mod}\,N) = T_{pq}(x)(\mathrm{mod}\,N).$$

Theorem 6.2. *Let N be an odd prime and let $x \in Z$ such that $0 \le x < N$. Then the period of the sequence $T_n(x)(\mathrm{mod}\,N)$, for $n = 0, 1, 2...$, is a divisor of $N^2 - 1$.*

The first theorem can easily be verified; the proof of the second theorem is given in the Appendix.

We now present an example. Several trajectories of the map (17), when $N = 19$, are given below:

$$x = 0, 1, 0, 18, 0, 1, 0, 18, 0, ...,$$
$$x = 1, 1, 1, 1, 1, ...,$$
$$x = 2, 1, 2, 7, 7, 2, 1, 2, 7, 7, 2, ...,$$
$$x = 3, 1, 3, 17, 4, 7, 0, 12, 15, 2, 16, 18, 16, 2, 15, 12, 0, 7, 4, 17, 3, ...,$$
$$x = 4, 1, 4, 12, 16, 2, 0, 17, 3, 7, 15, 18, 15, 7, 3, 17, 0, 2, 16, 12, 4, ...,$$
$$x = 5, 1, 5, 11, 10, 13, 6, 9, 8, 14, 18, 14, 8, 9, 6, 13, 10, 11, 5, ...$$

The periods of all trajectories of the map (17), with $N = 19$, are listed in Table 3. They are always divisors of $18 \times 20 = 2^3 3^2 5$. One can easily show that for any odd prime N the periods of the trajectories starting from the initial points $x = 0$, $x = 1$, and $x = N - 1$ are always 4, 1, 2, respectively.

Table 3 Periods of the sequences $T_n(x)(\mathrm{mod}\,19)_{n \ge 0}$ for each $x = 0, 1, 2, ..., 18$

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Period	4	1	5	20	20	18	18	5	18	3	6	9	10	9	9	20	20	10	2

6.3   Software Implementation

In a public-key algorithm encryption, decryption, signing, and verifying signatures all involve multiplying with a large number. We now present an algorithm for computing $T_p(x)(\mathrm{mod}\,N)$ when N and p are large numbers. Equation (2) can be rewritten as

$$\begin{bmatrix} T_p \\ T_{p+1} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & 2x \end{bmatrix} \begin{bmatrix} T_{p-1} \\ T_p \end{bmatrix} = A \begin{bmatrix} T_{p-1} \\ T_p \end{bmatrix}, \tag{18}$$

or, after some algebra, as

$$\begin{bmatrix} T_p \\ T_{p+1} \end{bmatrix} = A^p \begin{bmatrix} T_0 \\ T_1 \end{bmatrix}. \tag{19}$$

Matrix exponentiation can be done effectively by the *square and multiply* algorithm. Using notation

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$

the pseudo-algorithm for calculating the matrix exponent A^p is

$A^p = I$;
for(i = p.numBits(); i > 0; i–)
{
    $A^p = (A^p)^2$;
    (p.bitAt(i) == 1)
        $A^p = A^p A$;
}.

Bit positions are enumerated starting at 1. The algorithm represents the matrix version of the number exponentiation algorithm that is used in the commercial asymmetric encryption algorithms.

The $T_p(x)(\bmod N)$ calculation speed is tested on an Intel Pentium 1700 MHz processor with 512 MB RAM, using equation (19). The test includes Java [41] and GNU multiple precision library [42] implementation. For N and p of order 1024 bits, calculating $T_p(x)(\bmod N)$ takes

Java : $\approx$ 700 ms
GMP : $\approx$ 70 ms.

6.4 ElGamal Public-Key Encryption with Chebyshev Polynomials

The ElGamal public-key encryption scheme can be viewed as a DiffieHellman key agreement in key transfer-mode [5]. Its security is based on the intractability of the discrete logarithm-problem and the DiffieHellman problem. The basic ElGamal and generalized ElGamal encryption schemes are described in [5]. Here we generalize the ElGamal encryption scheme for modified Chebyshev polynomials.

6.4.1 Description of the Algorithm

The ElGamal public-key cryptographic system consists of two algorithms: an algorithm for key generation and an algorithm for encryption.

Algorithm for key generation.
Alice should do the following:

1. Generate a large random prime N and an integer x such that $x < N$.
2. Generate a random integer $s < N$ and compute $A = T_s(x)(\bmod N)$.
3. Alices pubic key is (x, N, A); Alices private key is s.

Algorithm for ElGamal public-key encryption.

1. Encryption. To encrypt a message m, Bob should do the following:
   a. Obtain Alices authentic public key (x, N, A).
   b. Represent the message as an integer m in the range $\{0, 1, ..., N-1\}$.
   c. Select a random integer $r < N$.
   d. Compute $B = T_r(x)(\bmod N)$ and $X = mT_r(A)(\bmod N)$.
   e. Send the cipher-text $c = (B, X)$ to Alice.
2. Decryption. To recover the message m from c, Alice should do the following:
   a. Use the private key s to compute $C = T_s(B)(\bmod N)$.
   b. Recover m by computing $m = XC^{-1}(\bmod N)$.

Proof that decryption works. This follows from the fact that

$$T_s(B) = T_s(T_r(x)) = T_r(T_s(x)) = T_r(A).$$

6.4.2   Example

We now present an example with artificially small parameters.

Key generation. Alice chooses the prime $N = 1749940627$, integers $x = 25749480$ and $s = 7207480595$, and computes $A = 173359085$. Alices public key is $(N = 1749940627, x = 25749480, A = 173359085)$, while her private key is $s = 7207480595$.

Encryption. To encrypt a message $m = 11223344$, Bob chooses an integer $r = 6431562606$ and computes $B = 1399079193$ and $X = 878048528$. He sends the cipher text $c = (B, X) = (1399079193, 878048528)$ to Alice.

Decryption. To recover the message m from c, Alice computes $C = 1376040233$ and $m = 11223344$.

6.4.3   Security

If $x > 1$, the Chebyshev polynomial $T_n(x)$ can be written as

$$T_n(x) = \cosh(n \cosh -1(x)).$$

Thus, if $y = T_n(x)(\bmod N)$, then, after some algebra, we find $n = \log_{x+\sqrt{x^2-1}}(y + \sqrt{y^2-1})$. In the case where both square roots, $\sqrt{x^2-1}$ and $\sqrt{y^2-1}$, exist in $GF(N)$, one has a conventional discrete log problem. On the other hand, if at least one of the square roots exists in the quadratic extension field $GF(N^2)$, this leads to a quadratic extension field generalization of the discrete log problem. Thus, the security of our modified ElGamal public-key algorithm is the same as the security of the original ElGamal algorithm.

6.5 RSA Public-Key Encryption with Chebyshev Polynomials

The RSA cryptosystem, named after its inventors, R. Rivest, A. Shamir, and L. Adleman, is the most widely used public-key cryptosystem. It may be used to provide both secrecy and digital signatures and its security is based on the intractability of the integer factorization problem. This section describes the generalization of an RSA encryption scheme for the modified Chebyshev polynomials. As in the case of an RSA cryptosystem, our system can be used for both encryption and digital signature and its security is based on the intractability of the integer factorization problem.

6.5.1 Description of the Algorithm

The RSA public-key cryptographic system consists of two algorithms: an algorithm for key generation and an algorithm for encryption.

Algorithm for key generation.
Alice should do the following:

1. Generate two large random (and distinct) primes p and q, each roughly the same size.
2. Compute $N = pq$ and $\phi = (p^2 - 1)(q^2 - 1)$.
3. Select a random integer e, $1 < e < \phi$, such that $\gcd(e, \phi) = 1$.
4. Compute the unique integer d, $1 < d < \phi$, such that $ed \equiv 1 \pmod{\phi}$.
5. Alice's public key is (N, e); Alice's private key is d.

Algorithm for encryption.

1. Encryption. To encrypt a message m, Bob should do the following:
   a. Obtain Alices authentic public key (N, e).
   b. Represent the message as an integer in the range $[1, N-1]$.
   c. Compute $c = T_e(m) \pmod{N}$ and send to Alice.
2. Decryption. To recover the message m from c, Alice should do the following:
   a. Use the private key d to recover m by computing $m = T_d(c) \pmod{N}$.

The integers e and d in RSA key generation are called the *encryption exponent* and the *decryption exponent*, respectively, while N is called the *modulus*.

Proof that decryption works. It was shown in Section 6.2 that if p is an odd prime number and $0 \le g < p$, then the period of the sequence $T_n(g) \pmod{p}$, $n = 0, 1, ...,$ is a divisor of $p^2 - 1$. Since $ed \equiv 1 \pmod{\phi}$, there exists an integer k such that $ed = 1 + k\phi$. Thus, we find

$$T_d(T_e(x)) \equiv T_{de}(x) \equiv T_{1+k\phi}(x) \equiv T_1(x) \equiv x(\bmod p).$$

By the same argument,

$$T_d(T_e(x)) \equiv T_{de}(x) \equiv T_{1+k\phi}(x) \equiv T_1(x) \equiv x(\bmod q).$$

Finally, since p and q are distinct primes, we may use the Chinese remainder theorem to show that

$$T_d(T_e(x)) \equiv T_{de}(x) \equiv T_{1+k\phi}(x) \equiv T_1(x) \equiv x(\bmod N).$$

6.5.2   Example

We now present an example with artificially small parameters.

Key generation. Alice chooses the primes $p = 21787$ and $q = 3793$ and computes $N = 82638091$ and $\phi = 6829053595064064$. Alice chooses $e = 65537$ and, using the extended Euclidean algorithm, finds $d = 2150406320724737$. Alices public key is the pair $(N = 82638091, e = 65537)$, while her private key is $d = 2150406320724737$.

Encryption. To encrypt a message $m = 11223344$, Bob computes

$$c = T_{65537}(11223344)(\bmod 82638091) = 12355612.$$

Decryption. To decrypt c, Alice computes

$$T_d(c)(\bmod N) = T_{2150406320724737}(12355612)(\bmod 82638091) = 11223344.$$

7   Conclusion

In this chapter we have shown different chaotic cryptosystems based on real and integer numbers, proposed in [21, 23] and [24]. The public-key cryptosystem based on real numbers even if it is efficient and based on fascinating and elegant idea, we have shown that is not secure, since an adversary can efficiently recover the plaintext from a given ciphertext. The cryptosystem proposed in [21] can be implemented by using any chaotic map for which F can be written as $F_p(x) = f(p \cdot f^{-1}(x))$, and such that $F_p(F_s(x)) = F_{p \cdot s}(x)$, i.e., it enjoys the semi-group property. Jacobian Elliptic Chebyshev Rational Maps represent another class of maps enjoying such a property. We have shown that the attack described in **Section 5** can still be applied if these maps are used. In [23] it was shown that the attack described in **Section 5** can still be applied if these maps are used. Moreover, in [23] the authors analyzed a DiffieHellman like key agreement scheme based on rational maps and they pointed out that if Jacobian Elliptic Chebyshev Rational Maps are used, then the scheme is not secure, in the sense that a passive adversary can compute the common key. We stress here that for the class of rational mappings enjoying the semi-group property, which comes from

multiplication formulas of periodic functions, Weierstrasss theorem [43] says that this class is limited to Jacobian elliptic functions and Chebyshev polynomials [44]. Therefore, the conclusion of the first part of this chapter has a certain universality: rational maps enjoying the semi-group property, which comes from multiplication formulas of periodic functions, are not secure for public-key encryption.

In the next part of this chapter we are concerned with public-key encryption algorithms using modified Chebyshev polynomials, which are both secure and practical and can be used for both encryption and digital signature. In [24] it was shown that El-Gamal and RSA algorithms can be extended for Chebyshev polynomials. A fast algorithm for computing Chebyshev polynomials is suggested. The public-key algorithms and their properties depend, in a crucial way, on the properties of the discretized versions of two well-known chaotic maps: Chebyshev maps and toral automorphisms.

References

1. Kolumban, G., Kennedy, M.P., Kis, G., Jako, Z.: FM-DCSK: A novel method for chaotic communications. In: Proceedings of the 1998 IEEE International Symposium on Circuits and Systems, ISCAS 1998, vol. 4, pp. 477–480 (1998)
2. Sushchik, M., Rulkov, N., Larson, L., Tsimring, L., Abarbanel, H., Yao, K., Volkovskii, A.: Chaotic pulse position modulation: A robust method of communicating with chaos. IEEE Commun. Lett. 4(4), 128–130 (2000)
3. Kocarev, L.: Chaos-based cryptography: A brief overview. IEEE Circuits Systems Magazine 1(3), 6–21 (2001)
4. Dachselt, F., Schwarz, W.: Chaos and cryptography. IEEE Trans. Circuits Systems I: Fund. Theory Appl. 48(12), 1498–1509 (2001)
5. Menezes, A., van Oorschot, P., Vanstone, S.: Handbook of Applied Cryptography. CRC Press, Boca Raton (1997)
6. Shanon, C.E.: Communication theory of secrecy systems. Bell Sys. Tech. J. 28, 656–715 (1949)
7. Alvarez, G., Li, S.: Some basic cryptographic requirements for chaos based cryptosystems. International Journal of Bifurcation and Chaos 16, 2129–2151 (2006)
8. Pichler, F., Scharinger, J.: Finite dimensional generalized Baker dynamical systems for cryptographic applications, Lect. In: Albrecht, R., Moreno-Díaz, R., Pichler, F. (eds.) EUROCAST 1995. LNCS, vol. 1030, pp. 465–476. Springer, Heidelberg (1996)
9. Fridrich, J.: Symmetric ciphers based on two-dimensional chaotic maps. Internat. J. Bifur. Chaos 8(6), 1259–1284 (1998)
10. Jakimoski, G., Kocarev, L.: Chaos and cryptography: block encryption ciphers based on chaotic maps. IEEE Trans. Circuits Systems I: Fund. Theory Appl. 48(2), 163–169 (2001)
11. Jakimoski, G., Kocarev, L.: Differential and linear probabilities of a block-encryption cipher. IEEE Trans. Circuits Systems I: Fund. Theory Appl. 50(1), 121–123 (2003)
12. Masuda, N., Aihara, K.: Cryptosystems with discretized chaotic maps. IEEE Trans. Circuits Systems I: Fund. Theory Appl. 49(1), 28–40 (2002)
13. Lawande, Q.V., Ivan, B.R., Dhodapkar, S.D.: Chaos Based Cryptography: A New Approach to Secure Communications, BARC Newsletter, Bombay (2005)
14. Matthews, R.A.J.: On the derivation of a 'chaotic' encryption algorithm. Cryptologia 13, 29–42 (1989); Wheeler, D.D.: Problems with chaotic cryptosystems. Cryptologia 13, 243–250 (1989); Wheeler, D.D., Matthews, R.A.J.: Supercomputer investigations of a chaotic encryption algorithm. Cryptologia 15(2), 140–152 (1991)

15. Kohda, T., Tsuneda, A.: Statistics of chaotic binary sequences. IEEE Trans. Inform. Theory 43, 104–112 (1997)
16. Kocarev, L., Jakimoski, G.: Pseudorandom bits generated by chaotic maps. IEEE Trans. Circuits Systems I: Fund. Theory Appl. 50(1), 123–126 (2003)
17. Petrie, C.S., Connelly, J.A.: A noise-based IC random number generator for applications in cryptography. IEEE Trans. Circuits Systems I: Fund. Theory Appl. 47(5), 615–621 (2000)
18. Stojanovski, T., Kocarev, L.: Chaos-based random number generators-part I: Analysis. IEEE Trans. Circuits Systems I: Fund. Theory Appl. 48(3), 281–288 (2001)
19. Stojanovski, T., Pihl, J., Kocarev, L.: Chaos-based random number generators PART II: Practical realization. IEEE Trans. Circuits Systems I: Fund. Theory Appl. 48(3), 382–385 (2001)
20. Gerosa, A., Bernardini, R., Pietri, S.: A fully integrated chaotic system for the generation of truly random numbers. IEEE Trans. Circuits Systems I: Fund. Theory Appl. 49(7), 993–1000 (2002)
21. Kocarev, L., Tasev, Z.: Public-key encryption based on Chebyshev maps. In: 2003 IEEE International Symposium on Circuits and Systems, ISCAS 2003, Bangkok, Thailand, May 25-28 (2003)
22. Kocarev, L., Tasev, Z., Makraduli, J.: Public-key encryption and digital-signature schemes using chaotic maps. In: 16th European Conference on Circuits Theory and Design, ECCTD 2003, Krakow, Poland, September 1-4 (2003)
23. Bergamo, P., D'Arco, P., De Santis, A., Kocarev, L.: Security of Public-Key Cryptosystems Based on Chebyshev Polynomials. IEEE Transactions on Circuits and Systems—I: Regular Papers 52(7) (2005)
24. Kocarev, L., Makraduli, J.: Public-key Encryption Based on Chebyshev Polynomials. Circuits Systems Signal Processing 24(5), 497–517 (2005)
25. Shannon, C.E.: Bell Syst. Tech. J., 27(379) (1948); 27(623) (1948)
26. Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Trans. Inform. Theory 22, 644–654 (1976)
27. Rivlin, T.J.: Chebyshev Polynomials. Wiley, New York (1990)
28. Percival, I., Vivaldi, F.: Arithmetical properties of strongly chaotic motions. Physica D 25(1-3), 105–130 (1987)
29. Cohn, H.: A Second Course in Number Theory. Wiley, New York (1962)
30. Hasse, H.: Number Theory. Springer, Berlin (2002)
31. El Gamal, T.: A public-key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans. Inf. Theory IT-31(4), 469–472 (1985)
32. Fateman, R.J.: Lookup tables, recurrences, and complexity. In: Proc. Int. Symp. Symbolic and Algebraic Computation. ISSAC 1989, pp. 68–73 (1989)
33. Umeno, K.: Method of constructing exactly solvable chaos. Phys. Rev. E 55, 5280–5284 (1997)
34. Kohda, T., Fujisaki, T.: Jacobian elliptic Chebyshev rational maps. Phys. D 148, 242–254 (2001)
35. Yacas (Yet Another Computer Algebra System), http://homepage.mac.com/yacas/manual/Algochapter4.html
36. Abramowitz, M., Stegun, I.A.: Handbook of Mathematical Functions. Dover, New York (1970)
37. Wachspress, E.L.: Evaluating elliptic functions and their inverses. Computers and Mathematics with Applications (39), 131–136 (2000)
38. Wachspress, E.L.: System, apparatus, and method for outputting pseudorandom noise sequences, and data recording medium, Japanese Patent no. 3 455 483 (July 25, 2003), see also the US Patent no. 6 654 404, http://www.uspto.gov

39. Xiao, D., Liao, X., Tang, G., Li, C.: Using chebyshev chaotic map to construct infinite length hash chains. In: Proc. Int. Conf. Communications, Circuits and Systems, vol. 1, pp. 11–12 (2004)
40. Knuth, D.E.: The Art of Computer Programming, vol. 2. Addison-Wesley, Reading (1998)
41. http://java.sun.com
42. http://www.swox.com/gmp/
43. Whittaker, E.T., Watson, G.N.: A Course of Modern Analysis. Cambridge Univ. Press, Cambridge (1935)
44. Umeno, K.: Exactly solvable chaos and addition theorems of elliptic functions, RIMS Kokyuroku no. 1098 (1999)

A Appendix

A.1 Ideal Theory in Quadratic Fields

In this Appendix we briefly summarize the ideal theory in quadratic fields, following [29], [30].

Quadratic integers. The solutions of the linear equations with integral coefficient, $ax + b = 0$, form the field of rational numbers. If the leading coefficient is equal to 1, $a = 1$, the solutions are integers. Following Dedekind, quadratic irrationals are defined as the solutions of quadratic equations with integral coefficients, whereas quadratic equations whose leading coefficient is 1 yield *quadratic integer*. Thus $(1 + \sqrt{5})/2$ and $2i$ are quadratic integers, since they satisfy the equations $x^2 - x - 1 = 0$ and $x^2 + 4 = 0$, respectively. Quadratic integers coincide with the eigenvalues of 2×2 integral matrices. Sometimes, when the possibility of confusion arises, ordinary integers will be called *rational* integers.

Norm, units, and primes. By analogy with complex conjugates, we define the conjugate of a quadratic irrational $z = (a + b\sqrt{D})/c$ as $z' = (a - b\sqrt{D})/c$. The number $zz' = N(z)$ is called the *norm* of z. Then $N(z) = N(z')$ and $N(zv) = N(z)N(v)$. We shall be interested exclusively in real fields, where the norm of the number has nothing to do with its actual magnitude, and can even be negative. The norm of a quadratic integer is a rational integer.

The divisors of all rational integers are just 1 and -1, which are called *units*. The units of a quadratic field are precisely those quadratic integers of the field having unit norm. In real fields there is an infinity of units, forming a cyclic multiplicative group. So every unit can be expressed as a power of the generator of the group, which is called the *fundamental unit*. For instance, the golden mean $(1 + \sqrt{5})/2$ and $2143295 + 221064\sqrt{94}$ are fundamental units in their respective fields.

A quadratic integer z that is not a unit is called a prime if a factorization $z = uv$ is possible only when one of the two factors is a unit. For instance, $z = 2 + \sqrt{7}$ is a prime. Then one would hope that any integer can be factored in essentially only one way as a product of primes. The richness and difficulty of the arithmetic of quadratic integers depends largely on the fact that unique factorization generally fails.

Quadratic residues. The values of a for which the congruence in x,

$$x^2 \equiv a \,(\mathrm{mod}\, p), \tag{20}$$

is solvable are called *quadratic residues* of the odd prime p. The quadratic residue character is denoted by the Legendre symbol $(\frac{a}{p})$ [also written (a/p)], where

$$
\begin{aligned}
(a/p) &= 1, \text{ if } x^2 \equiv a\,(\mathrm{mod}\, p) \text{ solvable and } (a,p) = 1,\\
(a/p) &= 0, \text{ if } (a,p) = p,\\
(a/p) &= -1, \text{ if } x^2 \equiv a\,(\mathrm{mod}\, p) \text{ unsolvable.}
\end{aligned}
\tag{21}
$$

Thus, $[1 + (a/p)]$ is the number of solutions to equation (20) for any a.

Modules. We define a *module* as a set of quantities closed under addition and subtraction. Thus, when a module contains an element ξ, it contains $0 (= \xi - \xi)$ as well as negatives $-\xi (= 0 - \xi)$ and integral multiples ($\xi + \xi$ written as 2ξ, $\xi + \xi + \xi$ written as 3ξ, etc.). We shall use capital letters $\mathbb{M}$, $\mathbb{N}$, $\mathbb{D}$, etc., to denote modules. We consider combinations of a finite set of vectors V_i ,

$$u = x_1 V_1 + x_2 V_2 + \dots + x_s V_s \tag{22}$$

where the x_i range over all integers. The set of those u forms a module $\mathbb{M}$ and the vectors V_i are called a basis of the module, written

$$M = [V_1, V_2, ..., V_s].$$

Field. A field is a set of quantities taken from the complex numbers closed under the rational operations, namely addition, subtraction, multiplication, and division (excluding division by zero). In quadratic number theory, the field we consider is taken to be the set of surds $(a + b\sqrt{D})/c$ for a, b, c integral, D fixed and not a perfect square, and $c \neq 0$. It can be seen that addition, subtraction, multiplication, and division of such quantities lead to quantities of the same form. This field is written symbolically as $R(\sqrt{D})$, meaning that the set of surds is *generated* by adjoining $\sqrt{D}$ to the rationals. The field $R(\sqrt{D})$ is called a field over rationals. We now extract from D its (positive or negative) square-free kernel D_0, so that $D = m^2 D_0$. Note that $\sqrt{D}$ and $\sqrt{D_0}$ generate the same field. We define

$$
\omega_0 = \begin{cases} \sqrt{D_0}, & \text{if } D_0 \not\equiv (\mathrm{mod}\, 4),\\ (1 + \sqrt{D_0})/2, & \text{if } D_0 \equiv (\mathrm{mod}\, 4). \end{cases}
\tag{23}
$$

Thus, the basis of quadratic integers in $R(\sqrt{D})$ is $[1, \omega_0]$. This module is designated by the symbol

$$D = [1, \omega_0].$$

For example, the basis of $R(\sqrt{2})$ is $[1, \sqrt{2}]$, the basis of $R(\sqrt{5})$ is $[1, (1+\sqrt{5})/2]$, while $R(\sqrt{8})$ has the same basis as $R(\sqrt{2})$. In general, the field $R(\sqrt{m^2 D_0})$ is independent of m, and so is $\mathbb{D}$ and its basis.

The rational integer d, defined as

$$d = \begin{cases} D_0, & \text{if } D_0 \equiv (\bmod 4), \\ 4D_0, & \text{if } D_0 \not\equiv (\bmod 4). \end{cases} \tag{24}$$

is called a *field discriminant*. All numbers sharing the same discriminant d form a field.

Integral domain. A set of quantities taken from complex numbers which is closed under addition, subtraction, and multiplication (ignoring division) is called a *ring*. If a ring contains the rational integers, it is called an *integral domain*. The quadratic integers of a fixed field $R(\sqrt{D_0})$ form a domain which we call $\mathbb{D}$.

If the integral domain $\mathbb{D}$ of all quadratic integers of $R(\sqrt{D})$ contains an integral domain $\mathbb{D}*$. which does not consist wholly of rationals, then $\mathbb{D}*$ is characterized by some fixed positive rational integer n as the set of integers of $\mathbb{D}$ which are congruent to a rational integer modulo n. The integral domain $\mathbb{D}*$ corresponding to n is written $\mathbb{D}^n$. Thus $\mathbb{D}_1 = \mathbb{D}$. Note also that $\mathbb{D}_n = [1, n\omega_0]$.

Ideals. We start with $\mathbb{D}_n$, a quadratic integral domain. We define an ideal $\mathbb{A}$ in $\mathbb{D}_n$ as a module in $\mathbb{D}_n$ with a special property that if $\alpha, \beta \in \mathbb{A}$ and $\xi \in \mathbb{D}_n$, then $\alpha \pm \beta \in \mathbb{A}$ (property valid for modules) and $\alpha\xi \in \mathbb{A}$ (property distinguishing ideals). Starting with α, a fixed element of $\mathbb{D}_n$, we define the principal ideal in $\mathbb{D}_n$

$$\mathbb{A} = (\alpha)$$

as the set of $\alpha\xi$ where ξ in $\mathbb{D}_n$. The ideal (1) is called the *unit* ideal. We define the *sum* of ideals as the ideal $\mathbb{A} + \mathbb{B} = \alpha + \beta$, where $\alpha \in \mathbb{A}$ and $\beta \in \mathbb{B}$. We next define the *product* of two ideals $\mathbb{A}$ and $\mathbb{B}$ as the ideal $\mathbb{C}$ "generated by all products" $\alpha\beta$. We now say *ideal $\mathbb{A}$ divides ideal $\mathbb{C}$* in $\mathbb{D}_n$ (or $\mathbb{A}|\mathbb{C}$ if and only if an ideal $\mathbb{B}$ exists in $\mathbb{D}_n$ for which $\mathbb{C} = \mathbb{A}\mathbb{B}$).

An *indecomposable ideal* in $\mathbb{D}_n$ is an ideal $\mathbb{Q}$ in $\mathbb{D}_n$ other than the unit ideal, which has no ideal in $\mathbb{D}_n$ as a divisor other than $\mathbb{Q}$ and $\mathbb{D}_n$. The integral domain $\mathbb{D}_1$ has unique factorization into indecomposables if and only if all ideals are principals.

A *prime ideal* in $\mathbb{D}_n$ is an ideal $\mathbb{P}$ in $\mathbb{D}_n$ other than the unit ideal, with the property that for any two ideals in $\mathbb{D}_n$, $\mathbb{A}$ and $\mathbb{B}$, if $\mathbb{P}|\mathbb{A}\mathbb{B}$, then $\mathbb{P}|\mathbb{A}$ or $\mathbb{P}|\mathbb{B}$. Every prime ideal $\mathbb{P}$ belongs to a rational prime p determined uniquely by $\mathbb{P}|(p)$.

The rational prime p factors in the quadratic field $R(\sqrt{D})$ (D is a square-free integer), according to the following rules based on d, the discriminant of the field, and (d/p), the Kronecker symbol,

$(p) = (p)$ or p is inert (does not factor) if and only if $(d/p) = -1$,

$(p) = \mathbb{P}_1\mathbb{P}_2$ or p splits into two different factors if and only if $(d/p) = 1$, (25)

$(p) = \mathbb{P}^2$ or p ramifies if and only if $(d/p) = 0$.

A.2   Dynamics and Arithmetics

In this section we briefly summarize the arithmetic properties of toral automorphisms, following [28]. Consider the dynamics of the following map:

$$\begin{bmatrix} x_{n+1} \\ y_{n+1} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & k \end{bmatrix} \begin{bmatrix} x_n \\ y_n \end{bmatrix} (\bmod N). \tag{26}$$

where x, y, k are integers and N, is prime. We further assume that $0 < x0$, $y0 < N$, and $2 < k < N$.

Let us consider a fixed value of the trace k, and let

$$\lambda = \frac{k + \sqrt{k_2 - 4}}{2}$$

be the eigenvalue of the matrix in equation (26). This determines an integral domain $\mathbb{D}_1$ to which the eigenvalue belongs. Let d be the field discriminant, i.e., let $d = D_0 \equiv (\bmod 4)$ or $d = 4D_0 \not\equiv (\bmod 4)$ where D_0 is the square-free kernel of $k^2 - 4$.

Consider now the unit ideal in $\mathbb{D}_1$ (i.e., $\mathbb{D}_1$ itself): $\mathbb{D}_1 = (1) = [1, \omega_0]$ where ω_0 is given by equation (23). Multiplying this ideal by λ we obtain the same ideal, but with a different basis. Its elements are integral linear combinations of the basis elements 1 and ω_0 given by the equation

$$\lambda[1, \omega_0] = [m_{11} + \omega_0 m_{21} + \omega_0 m_{22}],$$

where the numbers m_{ij} are rearranged as a matrix

$$M' = \begin{bmatrix} m_{11} & m_{22} \\ m_{12} & m_{22} \end{bmatrix}. \tag{27}$$

Since λ is a unit of norm $+1$, the matrix M' is strictly unimodular (its determinant is equal to $+1$). We now identify the point $(x, y) \in Z^2$ with $z = x + y\omega_0$, i.e., z is a quadratic integer in the ideal (1). From (27) one obtains

$$\lambda z = \lambda x + \lambda y \omega_0 = x' + y' \omega_0$$

One can see that multiplication by λ corresponds to the action of the transpose M of M' on Z^2: $M(x, y) = (x', y')$. In constructing the matrix M from (27), we have used the largest solution λ of the equation $\lambda^2 - k\lambda + 1 = 0$. This choice is not restrictive, since the smallest solution λ', which is conjugate to λ, would just correspond to the inverse matrix M^{-1}, as is easily verified. Note also that one can derive an explicit expression for M. Let $k^2 - 4 = m^2 D_0$ and let D_0 be a square-free kernel. Thus, for k odd, M reads

$$M = \begin{bmatrix} h & (h^2 + mh - 1)/m \\ m & h + m \end{bmatrix},$$

where $h = (k - m)/2$, while for k even, M reads

$$M = \begin{bmatrix} h & (h^2 - 1)/m \\ m & h \end{bmatrix},$$

where $h = m/2$.

We now determine, for each value of k, the properties of the orbits generated by M, a task which is greatly simplified by our choice of identifying Z^2 with the unit ideal in $\mathbb{D}_1$. Then, one can determine the properties of the orbits generated by other 2×2 matrices with integer entries and determinant $+1$. It turns out, however, that the orbit structure depends to a great extent on the eigenvalue λ alone, which depends only on one parameter, the trace k.

In order to take into account the periodicity of the torus, we use a "two-dimensional" modular arithmetic, identifying quadratic integers which differ by elements of the ideal $(N) = [N, N\omega_0]$. In other words, we identify the points of square lattices with side N. To do so, we need a generalization of the concept of congruence, since if $z = x + y\omega_0$ both x and y must be taken modulo N. We say that two quadratic integers v, z are congruent module an ideal $\mathbb{A}$, and write $v \equiv z(\bmod A)$, if $v - z$ is contained in $\mathbb{A}$.

The period of an orbit through the point (x, y) is given by the smallest integer T satisfying the congruence

$$\lambda^T \equiv z(\bmod (N)), \quad z = x + y\omega_0.$$

Note that since λ is a unit, $(\lambda)\mathbb{A} = \mathbb{A}$ for any ideal $\mathbb{A}$; thus, $\mathbb{A}$ is an invariant sublattice of Z^2. On the other hand, since one performs arithmetic modulo (N), the only invariant ideals on the torus are divisors of (N). To perform the ideal factorization of (N) (if N is an integer), we first determine its rational prime factors, $N = p_1 p_2 ... p_n$, where p_i are rational primes. This corresponds to the ideal factorization $(N) = (p_1)(p_2)...(p_n)$. However, in our case, N is a prime number. In the following, an orbit which belongs to some ideal factor of (N) different from (1) will be called an *ideal orbit*, otherwise we shall speak of a *free orbit*. Bellow we state some results, which are proved in [28].

1. If $(d/N) = -1$, (N) is inert. All orbits are free and have the same period T, which is a divisor of $N + 1$. If $T = (N + 1)/m$, then there are $m(N - 1)$free orbits.
2. If $(d/N) = -1$, (N) splits. All orbits have the same period T, which divides $N - 1$. If $T = (N - 1)/m$, then there are $m(N - 1)$ free orbits and $2m$ ideal orbits.
3. If $(d/N) = -1$, (N) ramifies. The periods of orbits are computed as follows. Let $\lambda = (k + b\sqrt{D0})$ (with k and b both even if $D_0 \not\equiv 1(\bmod 4)$). We have two cases:

   a. If $k \equiv 2(\bmod N)$, there are $N - 1$ ideal fixed points and $N - 1$ free orbits of period N.
   b. If $k \equiv -2(\bmod N)$, there are $(N - 1)/2$ ideal orbits of period 2 and $(N - 1)/2$ free orbits of period $2N$.

A.3 Proof of Theorem 6.2

In this section we give a proof of Theorem 6.2. Consider the following matrix:

$$C = \begin{bmatrix} 0 & 1 \\ -1 & 2g \end{bmatrix}.$$

We write $\lambda = g + g^2 - 1$ for its largest eigenvalue. Let $g^2 - 1 = m^2 D_0$, where D_0 is a square-free kernel. We define an integer d as follows:

$$d = \begin{cases} D_0, & \text{if } D_0 \equiv (\bmod 4), \\ 4D_0, & \text{if } D_0 \not\equiv (\bmod 4). \end{cases}$$

The proof of Theorem 6.2 follows directly from the following theorem:

Let N be an odd prime and let $g \in Z$ be such that $0 \le g < N$. Let T be the period of the sequence $T_n(g)(\bmod N)$ for $n = 0, 1, 2....$ Then:

(i) *if $x^2 \equiv d(\bmod N)$ is solvable, then T is a divisor of $N - 1$; otherwise*
(ii) *if $x^2 \equiv d(\bmod N)$ is unsolvable, then T is a divisor of $N + 1$.*

The proof of this theorem, however, follows from the results of the previous Section A.2 if $g \ge 2$. We need only to consider the cases $g = 0$ and $g = 1$. As mentioned in Section 6.2 the periods of the trajectories starting from the initial points $g = 0$ and $g = 1$ are always 4 and 1, respectively. Thus, for all odd primes N, the period of the sequence $T_n(g)(\bmod N)$ is a divisor of $N^2 - 1$.

Chapter 3
Digitized Chaos for Pseudo-random Number Generation in Cryptography

Tommaso Addabbo, Ada Fort, Santina Rocchi, and Valerio Vignoli

Information Engineering Department
University of Siena – Italy
`addabbo@dii.unisi.it`

1 Introduction

Random numbers play a key-role in cryptography, since they are used, e.g., to define enciphering keys or passwords [1]. Nowadays, the generation of random numbers is obtained referring to two types of devices, that are often properly combined together: True Random Number Generators (TRNGs), and Pseudo Random Number Generators (PRNGs). The former are devices that exploit truly stochastic physical phenomena [2, 3, 4, 5, 6], such as the electronic noise or the chaotic dynamics of certain nonlinear systems: for these devices the output sequences have an intrinsic degree of unpredictability, that is measured referring to the theoretical tools provided by Information Theory (e.g., in terms of the Shannon entropy) [7, 4]. On the other hand, PRNGs are deterministic periodic finite state machines whose aim is to emulate, within the period, the random behavior of a truly random source of numbers. From a theoretical point of view, due to their deterministic nature, PRNGs are potentially predictable by observing their generated sequences [8, 9, 10, 1]. Nevertheless, in literature some families of PRNGs are classified to be 'secure', meaning that their algorithmic structure involves calculations that *in average*, referring to the prediction task, require an amount of computation time that is *asymptotically* unfeasible with the size of the problem, when referring to both the computational equipment at disposal and the known computing fastest algorithms [1, 11]. It is worth noting that a given generator, even if belonging to an asymptotically secure family of PRNGs, can generate short periodic (and unsecure) sequences for several values of the initial seed[1]. Therefore, apart from the cryptographic robustness of their algorithmic structure, a cryptographic PRNG must generate sequences that are acceptable from a statistical point of view, i.e., that pass a certain number of standard statistical tests [1, 12].

---

[1] E.g., the well known Blum, Blum, Shub generator $k_{n+1} = k_n^2 \mod N$ can generate short periodic sequences (short compared to N).

In this work we propose to take certain chaotic systems as a reference for the design of PRNGs based on nonlinear congruences. In detail, in Section 2 we report a brief comparison between linear and nonlinear PRNGs. Since our aim is to derive nonlinear congruential generators from certain chaotic maps, in Section 3 we overview some theoretical fundamentals about TRNGs based on statistically stable mixing dynamical systems, focusing on the family of the Rényi maps. In Section 4 we discuss the link that exists between the dynamics of chaotic and pseudo-chaotic systems: to explain how the two dynamics are related it is necessary to project some results achieved within the Ergodic Theory (valid for chaotic systems) on the world of digital pseudo-chaos. To this aim, we have proposed a weaker and more general interpretation of the Shadowing Theory proposed by Coomes et al. [13], focusing on probability measures, rather than on single chaotic trajectories. In Section 5 we study how to digitize the Rényi maps, discussing how to set a minimum period length of the digitized trajectories. In Section 6 we present two alternative methods for the design of PRNGs based on nonlinear recurrences derived from the Renyi map, reporting the results of the NIST SP800.22 standard statistical test suite [12].

2   Linear vs. Nonlinear Congruential Generators

Conventional cryptographic systems are based on finite state machines, and the problem of generating random numbers can be analyzed referring to finite subsets of integers. Accordingly, let $\Lambda_M = \{0, \ldots, M\}$ be the set of the first $(M+1)$ non-negative integers. We define the j-plet $k_0, \ldots, k_{j-1} \in \Lambda_M$ as the *initial seed* of the generator, and we define a congruential generator as an iterative method for generating the sequence $\{k_i \in \Lambda_M, i \in \mathbb{N}\}$, where

$$k_n = G(k_{n-1}, \ldots, k_{n-j}) \bmod M, \qquad n > j, \qquad (1)$$

for a certain function $G : \Lambda_M^j \to \mathbb{N}$. The congruential generator is called linear if the function G is a linear combination of the previous j numbers in the sequence (with coefficients in Λ_M), otherwise it is said nonlinear. The simplest example of a linear generator of the form (1) is the Linear Congruential Generator (LCG) $k_n = ak_{n-1} + c \bmod M$, whereas for the Linear Feedback Shift Register (LFSR) with primitive polynomial $x^3 + x + 1$ we have $M = 2$, $\Lambda_2 = \{0, 1\}$ and $G(k_{n-1}, k_{n-2}, k_{n-3}) = k_{n-1} + k_{n-3}$ [10]. Examples of nonlinear congruential generators are the Nonlinear Feedback Shift Registers (NLFSRs), the polynomial congruential generators in which $G(k_{n-1}) = a_p k_{n-1}^p + a_{p-1} k_{n-1}^{p-1} + \ldots + a_0$, and the Inversive Congruential Generator with $G(k_{n-1}) = ak_{n-1}^{-1} + c$ [14, 15, 16, 1, 17]. Alternatively, we will show that the function G can be obtained by digitizing a chaotic map.

Regardless of the linearity of G, a generator with finite memory like those of the form (1) can be implemented in a finite state machine, being the state of
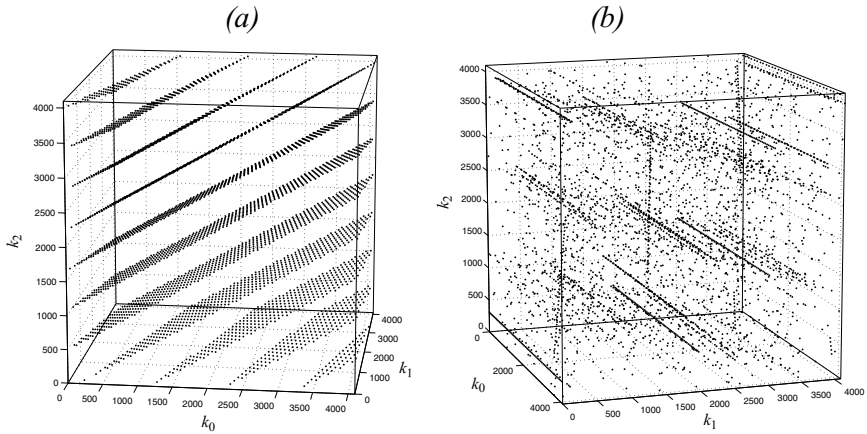
Fig. 1 The distribution of vectors (k_2, k_1, k_0) for the linear congruential generator $k_n = 1333k_{n-1} + 3 \bmod 2^{12}$ (subplot (a)) and for the nonlinear congruential generator $k_n = k_{n-1}^3 - k_{n-1}^2 + 7 \bmod 2^{12}$ (subplot (b)).

the machine at the time-step n the j-tuple $\sigma_n = (k_{n-1}, \ldots, k_{n-j}) \in \Sigma = \Lambda_n^j$. Since the cardinality of Σ is finite and due to the deterministic nature of the machine evolution, for any initial seed $\sigma_0 \in \Sigma$ the sequence $\{\sigma_i, i \in \mathbb{N}\}$ is eventually periodic with period $\mu(\sigma_0)$, and it enters the loop after a transient of $\eta(\sigma_0)$ steps [10]. As a result, the same happens to the sequences of pseudo-random numbers, being the generated number at the time-step n dependent of the state σ_n. A generator that for any σ_0 generates a sequence with period equal to the cardinality of Σ is called a maximum cycle generator.

The problem of relating the length of the cycles μ with the initial seed σ_0 and the generator parameters (e.g., for a linear generator the coefficients in the linear combination G), is an issue of high interest in cryptography. Indeed, it is desirable to know *a priori* that a given sequence of pseudo-random numbers will not enter a too-short predictable cycle. For generators based on linear recurrences the problem has been studied in depth, and well known design criteria are at disposal to obtain maximum cycle devices. On the other hand, for most families of nonlinear generators the problem seems to be intractable, with few exceptions [18, 1].

When dealing with cryptographic applications, linear methods for generating pseudo-random sequences (like LFSRs, LCGs or their proper combinations) are highly not recommended, since efficient algorithms are at disposal to predict the sequence on the basis of a relatively short sequence observation [8, 9]. This cryptographic weakness is reflected by the fact that methods based on linear recurrences may generate numbers lying on regular lattices [19]. In detail, $(d+1)$-tuples of generated numbers $(k_n, k_{n-i_1}, \ldots, k_{n-i_d})$ in the sequence form vectors that belong to a lattice structure in $\mathbb{N}^{d+1}$, as

shown in the subplot *(a)* of Fig. 1. In the subplot *(b)* of the same figure it is shown that this does not happen for nonlinear generators, for which the distributions of generated points can reveal irregular structures that – it must be underlined – in most cases are far from being uniformly distributed.

Concluding this brief comparison overview, linear recurrences allow for the definition of maximum cycle generators, with regular distributions of generated numbers (even 'too regular' for some applications [19,15]), involving very efficient hardware or software implementations, and they are highly not recommended in cryptography [1]. These drawbacks can be overcome by nonlinear generators, but in this case only for few exceptions maximum cycles generators could be designed [18,15], their hardware or software implementations are less efficient and the distributions of generated numbers can be particularly not uniform. Nevertheless, as shown in this work, solutions derived from chaotic systems can help in defining nonlinear cryptographic PRNGs with both efficient implementation and good statistical properties.

3 Statistically Stable Mixing Systems

Since our aim is to derive nonlinear congruential generators from certain chaotic maps, in this Section we overview some fundamentals that will be used afterwards. The theory presented hereafter has been simplified to match the aim of this work, adopting the following notation and terminology. With reference to the Lebesgue integration theory, the notation $L^p(I)$ denotes the set of functions $f : I \rightarrow \mathbb{R}$ such that $\int_I |f(x)|^p dx < \infty$, with $0 < p \in \mathbb{N}$, whereas $L^\infty(I)$ is the set of almost everywhere bounded measurable functions. We recall that $L^p(I)$ and $L^\infty(I)$ can be made Banach spaces with reference to the norms $\|f\|_p = (\int_I |f(x)|^p dx)^{\frac{1}{p}}$ and $\|f\|_\infty = \inf\{M \in \mathbb{R}^+$ such that $\{x \in I : |f(x)| > M\}$ has zero measure$\}$, respectively. We define Π as the set of all finite partitions of intervals of $[0,1)$, i.e., if $Q \in \Pi$ then $Q = \{I_i, i = 1, \ldots, q,$ with $q > 0\}$, with $I_i \cap I_j = \varnothing$ for $i \neq j$ and $\cup_{i=1}^q I_i = [0,1)$. Among the elements of Π we highlight the partition $\mathcal{P}_n = \{I_0, \ldots, I_{2^n-1}\}$ made of the 2^n equal and disjoint intervals that divide $[0,1)$.

We define the special set of probability density functions (pdfs) of bounded variations [20] as

$$D_{BV} = \{f \in L^1([0,1)) : \|f\|_1 = 1, f \geq 0, f \text{ is of bounded variation}\}. \quad (2)$$

The set D_{BV} is a wide set containing all those pdfs of practical interest and physical meaning.

The chaotic systems taken into account in this work are those ruled by the maps defined as in the following

Definition 1. *The map* $S : [0,1) \rightarrow [0,1)$ *is piecewise affine expanding (PWAE) if and only if it is onto and there exists a partition* $Q \in \Pi$,

with $Q = \{I_0, \ldots, I_{q-1}\}$, such that $S|_{I_i}$ is a linear function of the form $S|_{I_i}(x) = \gamma_i x + \beta_i$, with $\gamma, \beta_i \in \mathbb{R}$ and $|\gamma_i| > 1$ $(i = 0, \ldots, q-1)$.

3.1   Statistical Stability and Correlation Decay

We say that a pdf $\phi^* \in L^1$ is invariant for the map S if for any subset $A \subseteq [0, 1)$ it results

$$\int_A \phi^*(x)dx = \int_{S^{-1}(A)} \phi^*(x)dx,$$

which implies $P(S(x) \in A) = P(x \in A)$. By assuming $x_0 \in [0, 1)$ as a random variable with pdf ϕ_0, let us focus on the sequence $\{S^p(x_0), p \in \mathbb{N}\}$. Even if S is deterministic $x_p = S^p(x_0)$ is a stochastic variable and we denote with ϕ_p its associated pdf. In this paper we refer to the following

Definition 2. *A PWAE map S is said to be statistically stable if $\forall \phi_0 \in D_\Pi$ there exists an unique invariant pdf $\phi^* \in L^1([0, 1))$ such that*

$$\lim_{p \to \infty} \|\phi_p - \phi^*\|_\infty = 0. \tag{3}$$

According to the above definition, for a statistically stable PWAE map as far as $p \to \infty$ the pdf of the random variable $S^p(x_0)$ approaches an invariant (stationary) pdf ϕ^* that only depends on the map S, regardless of the distribution of the initial condition x_0. The evolution of densities $\{\phi_p, p \in \mathbb{N}\}$ can be analyzed by means of the Frobenious-Perron operator $\Theta_S : D_\Pi \to D_\Pi$

$$\phi_{p+1} = \Theta_S \phi_p(x) = \sum_{y = S^{-1}(x)} \frac{\phi_p(y)}{\left|\frac{dS}{dy}(y)\right|}. \tag{4}$$

An efficient numerical method for analyzing the convergence rate of the sequence of pdfs can be found in [21, 22]. As shown in the next sections, in practical cases that are of interest for random number generation the transient (3) vanishes in few steps.

A consequence of statistical stability is the decay of the autocorrelation function associated to the stochastic process $\{S^p(x_0), p \in \mathbb{N}\}$ [23]. Once assuming the process stabilized on its invariant density ϕ^* the autocorrelation function r_{xx} is given by

$$r_{xx}(m) = E\{xS^m(x)\} = \int_0^1 xS^m(x)\phi^*(x)dx. \tag{5}$$

3.2 True Random Number Generation with Rényi Maps

Let us consider a special case of PWAE maps, i.e., the family of Rényi transformations

$$S_\beta(x) = \beta x \bmod 1, \qquad \beta > 1, \beta \in \mathbb{R}, \tag{6}$$

where we assume the modulus operator extended to the real numbers, i.e., $\beta x \bmod 1 = \beta x - \lfloor \beta x \rfloor$. In the following, we denote with $b = \lfloor \beta \rfloor$ the integer part of β, whereas we denote with $\gamma = \beta \bmod 1$ its fractional part. If the parameter β assumes integer values $b \in \mathbb{N}$ (i.e., $\gamma = 0$) the Rényi map has the uniform pdf as its unique stationary pdf [20]. In such case the Frobenious-Perron operator can be explicitly written as

$$\phi_{p+1} = \Theta_{S_b}\phi_p(x) = \frac{1}{b}\sum_{i=0}^{b-1}\phi_p\left(\frac{x+i}{b}\right), \tag{7}$$

and it can be easily used for evaluating the limit (3) starting from an arbitrary pdf ϕ_0. Moreover, since $S_b^m(x) = b^m x \bmod 1$, the autocorrelation function (5) results equal to

$$r_{xx}(p) = E\{x_i S_b^p(x_i)\} = \sum_{i=0}^{b^p-1}\int_{\frac{i}{b^p}}^{\frac{i+1}{b^p}} x(b^p x - i)dx =$$

$$= \sum_{i=0}^{b^p-1}\left(\frac{(i+1)^3}{2b^{2p}} - \frac{i(i+1)^2}{2b^{2p}} - \frac{i^3}{3b^{2p}} + \frac{i^2}{2b^{2p}}\right) = \frac{3b^p+1}{12b^p}. \tag{8}$$

According to the previous result, as far as $b^m \to \infty$ the autocorrelation approaches 0.25, i.e., the sequence of numbers $\{x_{km}, k \in \mathbb{N}\}$ becomes a sequence of uncorrelated random variables uniformly distributed in $[0,1)$. The hyperbolic curve

$$r_{xx}(m) - \frac{1}{4} = \frac{1}{12b^m} \tag{9}$$

represents the correlation decay associated to the chaotic sequence, assuming the chaotic state pdf stabilized on its invariant uniform pdf.

Ideal Random Number Generators

If the Rényi map S_b is stabilized on its invariant uniform pdf, it is possible [20, 24] to generate a sequence $\{\psi_i, i \in \mathbb{N}\}$ of i.i.d. uniform random integers $\psi_i \in \{0, \dots, b-1\}$ by partitioning the domain with the b intervals

$$J_i = \left[\frac{i}{b}, \frac{i+1}{b}\right), \qquad i = 0, \dots, b-1, \tag{10}$$

and coding the sequence $\{S_b^i(x_0), i \in \mathbb{N}\}$ with the rule

$$\psi_i = m \Leftrightarrow S_b^i(x_0) \in J_m. \tag{11}$$

The partition made of the intervals (10) is called the *natural generating symbolic partition*.

The above technique defines the way to generate a symbolic sequence from any chaotic trajectory, obtaining an information source of i.i.d. uniformly distributed b random integers, i.e., the above technique defines an ideal TRNG. We recall that for an ideal TRNG having as alphabet the first M nonnegative integers it must result:

$$E\{\psi\} = \sum_{i=0}^{M-1} \frac{i}{M} = \frac{M-1}{2}, \tag{12}$$

and in general, for $k \geq 1$,

$$E\{\psi_{m_1} \ldots \psi_{m_k}\} = \frac{(M-1)^k}{2^k}. \tag{13}$$

It is interesting noting that if we apply the generation rule (11) referring to a symbolic partition made of M intervals that are different from those of the form (10), in general we obtain a Markovian information source that generates sequences biased and affected by memory [20,24,4]. In other words, we obtain a not ideal TRNG not satisfying (12) and (13), and we stress that in general the obtained Markovian source can have infinite memory [23,25].

Nevertheless, since the uniform pdf is the invariant pdf for the Rényi map S_b, the biasing between numbers can be eliminated using a symbolic partition made of intervals with equal length, obtaining (12) to be satisfied. Nevertheless, as shown in the following example, this trick may not help in eliminating the correlation between the generated numbers.

Example

Let us consider the Rényi chaotic map

$$S_3(x) = 3x \bmod 1. \tag{14}$$

If the state x_0 is not uniformly distributed, a vanishing transient occurs for the state pdf to reach the stationary invariant pdf. More in detail, regardless of the pdf $\phi_0 \in D_\Pi$ associated to x, the greater is p and the more the random variable $y = S_3^p(x)$ is uniformly-distributed, as stated in (3). In other words, $\forall \varepsilon > 0$ there exists p such that

$$\|\phi_p - u\|_\infty < \varepsilon, \tag{15}$$

where $u : [0,1) \rightarrow \{1\}$ is the uniform pdf. The number of iterations p necessary to satisfy the above inequality in general depends on ϕ_0 and ε, but
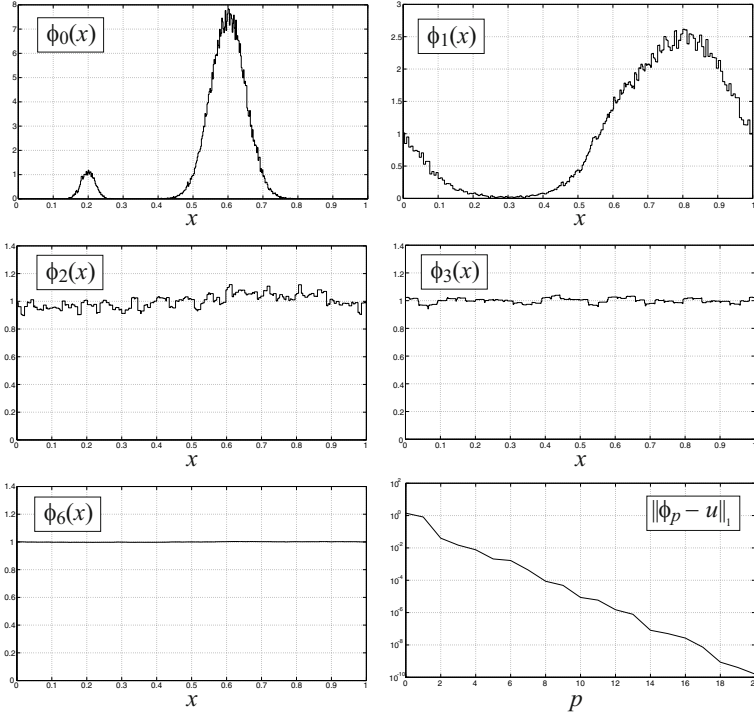
Fig. 2 The pdfs associated to the random variables $y = S_3^p(x)$, for $p = 0, 1, 2, 3, 6$. After 6 iterations the distance $\|\phi_5 - u\|_\infty$ is lower than 10^{-3}: the convergence rate is exponential, as confirmed in the log-scale plot of $\|\phi_p - u\|_\infty$. The pdfs were calculated exploiting a modified version of the accurate approach described in [21, 22].

in cases of practical interest, since the convergence rate is exponential, few iterations suffice to obtain a reasonably accurate approximation of the uniform pdf u [3, 20]. Indeed, 8 iterations of the map S_3 suffices to satisfy the inequality (15) with values of ε in the order of 10^{-4}, even when ϕ_0 is quite different from the uniform pdf (see, e.g., Fig. 2).

As far as the map S_3 is used to define a TRNG, the statistical characteristics of the obtained information source depends on the chosen symbolic partition. To make clear this idea, in Fig. 3, we represented the two Markov chains that model the symbolic dynamics obtained by partitioning the domain $[0, 1)$ with the partition $\mathcal{P}_1 = \left\{ \left[0, \frac{1}{2} \right), \left[\frac{1}{2}, 1 \right) \right\}$ – case (a) – and the natural generating symbolic partition $\left\{ \left[0, \frac{1}{3} \right), \left[\frac{1}{3}, \frac{2}{3} \right), \left[\frac{2}{3}, 1 \right) \right\}$ – case (b). In the case (a) the obtained TRNG is not ideal. Indeed, by denoting with I_1 the interval $\left[\frac{1}{2}, 1 \right)$ we have

$$E\{\psi_{m_1}\ldots\psi_{m_k}\} = \sum_{\substack{\psi_{m_1}\in\{0,1\}\\ \cdots \\ \psi_{m_k}\in\{0,1\}}} \psi_{m_1}\ldots\psi_{m_k}P(\psi_{m_1}\ldots\psi_{m_k}) =$$

$$= P(\psi_{m_1}=1,\ldots,\psi_{m_k}=1) = P\left(x\in\bigcap_{i=1}^{k}S_3^{-m_i}(I_1)\right) = \qquad (16)$$

$$= \int_{\bigcap_{i=1}^{k}S_3^{-m_i}(I_1)} u(x)dx = \lambda\left(\bigcap_{i=1}^{k}S_3^{-m_i}(I_1)\right),$$

where λ represents the Lebesgue measure. When $k=1$ in (16), we obtain the mean value

$$E\{\psi\} = \frac{1}{2}, \qquad (17)$$

that satisfies (12) for $M=2$. When $k=2$ in (16), by denoting with $m = m_2 - m_1$, we obtain the autocorrelation function

$$R_{xx}(m) = E\{\psi_i\psi_{i+m}\} = \frac{1}{4} + \frac{1}{2\cdot 3^m}, \qquad (18)$$

that only asymptotically satisfies (13) for $M=2$, if $m\to\infty$, i.e., if the generated symbols becomes progressively uncorrelated. In general, the same can be shown for any order $k\geq 2$.
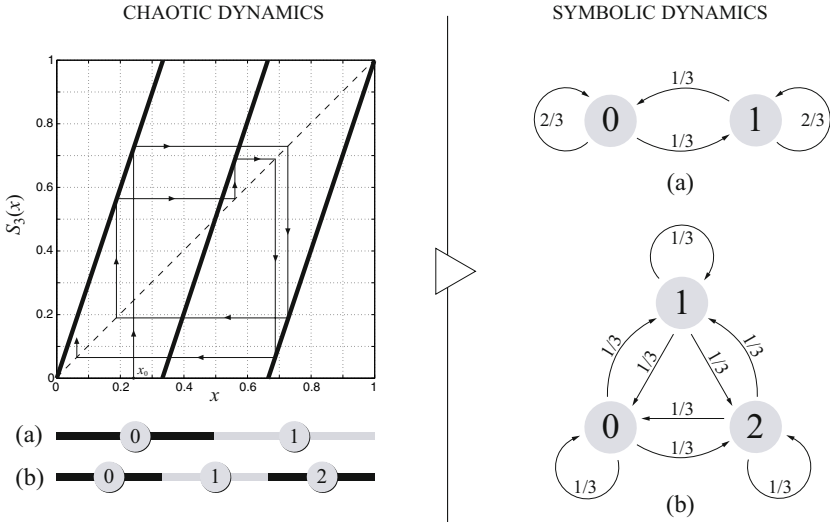
Fig. 3 The Rényi map $S_3(x) = 3x \bmod 1$ and the Markov chains modeling the symbolic dynamics obtained by partitioning the domain with the partition $\mathcal{P}_1 = \left\{\left[0,\frac{1}{2}\right),\left[\frac{1}{2},1\right)\right\}$– case (a) – and the natural generating symbolic partition $\left\{\left[0,\frac{1}{3}\right),\left[\frac{1}{3},\frac{2}{3}\right),\left[\frac{2}{3},1\right)\right\}$ – case (b). The state transition probabilities of the Markov chains are reported beside the arrows.
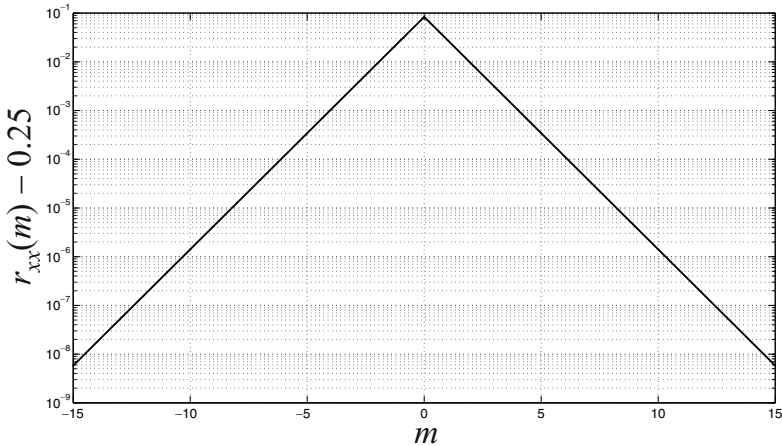
Fig. 4 The autocorrelation decay (9) for the sequence $\{S_3^p(x), p \in \mathbb{N}\}$, plotted in logarithmic scale.

On the other hand, when the Rényi map S_3 is partitioned with its natural generating symbolic partition, adopting the same theoretical approach it can be easily shown that the obtained TRNG is ideal: the three integers that may be generated are i.i.d. and their generation probabilities are equal to $1/3$, with a k-th order joint probability $P(\psi_1, \psi_2, \ldots, \psi_k) = \prod_{i=1}^k \frac{1}{3} = \frac{1}{3^m}$, satisfying both (12) and (13) for any order k.

It is worth noting that the result (18) is in accordance with the correlation decay property (9) between x and $S_3^m(x)$. Indeed, it is expected that if x and $S_3^m(x)$ are not correlated, also the two numbers generated with the rule (11) are not correlated. The autocorrelation function depicted in Fig. 4 shows that a few under-sampling steps suffice to have a negligible residual correlation between x and $S_3^m(x)$.

Preliminary conclusions

The above discussion lead to two different strategies for generating random numbers with the chaotic Rényi maps:

A)   To divide the domain $[0, 1)$ with the natural generating symbolic partition associated to the map S_b, in order to obtain an *ideal* TRNG issuing b random integers;

B)   To divide the domain $[0, 1)$ with an arbitrary partition $\mathcal{P}_n$ of 2^n intervals, in order to obtain a *not-ideal* TRNG issuing 2^n random integers. In such case the symbols are uniformly distributed, but they are affected by a correlation of any order that has a vanishing decay. Accordingly, an ideal TRNG issuing 2^n random integers can be approximated with any accuracy by under-sampling the sequence $\{S_b^i(x), i \in \mathbb{N}\}$ by a factor p, that is referring to the sequence $\{S_b^{ip}(x), i \in \mathbb{N}\}$.
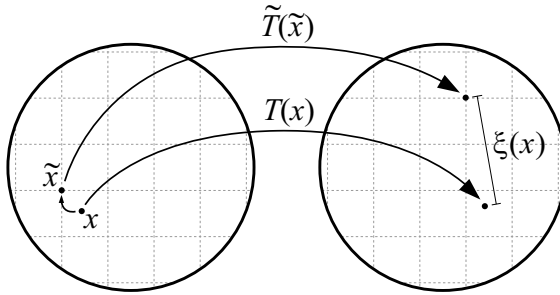
Fig. 5 The effect of the finite-precision computation of a generic function T.

Furthermore, we stress that if b can be divided by q, then b/q unions of q partitioning intervals can be used to obtain an ideal TRNG that can generate random integers in the set $\{0, \ldots, b/q\}$. For example, any Rényi map S_b with b even can be used to define an ideal TRNG with alphabet $\{0, 1\}$, referring to the partition $\mathcal{P}_1$. The process is similar to generating even or odd numbers throwing a fair six-sided die.

4 Pseudo-chaotic Systems

In this work we propose to exploit the ergodic dynamics of PWAE chaotic maps for the design of nonlinear PRNGs with good statistical properties and that involve low-complexity implementations. To this aim, we start analyzing the links that exist between chaos and pseudo-chaos. Pseudo-chaos is obtained when a chaotic dynamical system is simulated using finite precision arithmetic algorithms [26,27,28]. In detail, referring to a generic chaotic map $T : [0, 1) \to [0, 1)$, the pseudo-chaotic approximation of T is obtained in two steps. In the first step any point $x \in [0, 1)$ is represented by a finite-precision point $\tilde{x}$ belonging to a finite set $\Lambda \subset [0, 1)$, called the discrete domain. Accordingly we have the definition of the finite-precision point $\tilde{x} : [0, 1) \to \Lambda$, as a function of x. In the second step the function T is approximated by a finite-precision approximating function $\tilde{T} : \Lambda \to \Lambda$ such that

$$\left| T(x) - \tilde{T}(\tilde{x}) \right| = \xi(x) < 1, \tag{19}$$

where the function ξ quantifies the quality of the approximation. Summarizing, the overall pseudo-chaotic approximation is defined by the composition $\tilde{T} \circ \tilde{x} : [0, 1) \to \Lambda$. If the function ξ assumes reasonably small values for all x, it is often said that $\tilde{T}$ 'shadows' T.

Trying to relate the properties of the two systems T and $\tilde{T}$ can be a difficult task, depending on their nonlinear functional forms. For example, adopting a dynamical evolution point of view, Coomes et al. in [13] proved

some major results valid for hyperbolic diffeomorphisms, that represent a fundamental reference within the Shadowing Theory for chaotic systems. We adopt a different point of view, since we are interested in relating the probability measures associated to T and $\tilde{T}$. In detail, for a given subset $I \subseteq [0,1)$, we are interested in the quantity

$$\mathcal{E}(I,\xi) = \left| P(T(x) \in I) - P(\tilde{T}(\tilde{x}) \in I) \right|, \tag{20}$$

where the set I is understood to be a sub-interval of $[0,1)$. Since T is measurable, the previous quantity can be also written as

$$\mathcal{E}(I,\xi) = \left| P(x \in T^{-1}(I)) - P(\tilde{x} \in \tilde{T}^{-1}(I \cap \Lambda)) \right|. \tag{21}$$

Due to the calculating approximation, in general it can happen that

$$\tilde{T}^{-1}(I \cap \Lambda) - (T^{-1}(I) \cap \Lambda) \neq \varnothing, \tag{22}$$

i.e., there are points mapped in I by the function $\tilde{T}$ that do not belong to $T^{-1}(I)$. About this issue, we have the following

Proposition 1 (Proved in [29]). *Let $I \subseteq [0,1)$ be an interval with endpoints $a < b$, and let us assume that $\forall x \in T^{-1}(I)$ it results*

$$\left| T(x) - \tilde{T}(\tilde{x}) \right| < \alpha = \sup_{x \in T^{-1}(I)} \xi(x). \tag{23}$$

Accordingly, by denoting with ϕ the pdf of $T(x)$, if $b - a \geq 2\alpha$ then it results

$$\int_{a+\alpha}^{b-\alpha} \phi(x)dx \leq P\left(\tilde{T}(\tilde{x}) \in I\right) \leq \int_{a-\alpha}^{b+\alpha} \phi(x)dx. \tag{24}$$

It is worth noting that the above Proposition has a general validity, and that if the condition (23) holds in the whole interval $[0,1)$, then the result (24) holds for any interval $I \subseteq [0,1)$ with length greater than 2α. This is the case provided by the Shadowing Theory, that is valid for certain continuous maps [13]; in this work we allow the chaotic map to be *not* continuous, as discussed in the following.

4.1 Almost-Uniform Measure-Preserving Chaotic Transformations

Under the hypotheses of Proposition 1, in this subsection we refine the theoretical result (24) assuming the random variable $T(x)$ almost uniformly-distributed, i.e., we assume

$$\|\phi - u\|_\infty \leq \varepsilon,$$

where $u : [0, 1) \rightarrow \{1\}$ is the uniform pdf over $[0, 1)$. Accordingly, for any interval $I \subseteq [0, 1)$ with endpoints $a < b$ it results

$$(b - a)(1 - \varepsilon) \leq \int_I \phi(x)dx \leq (b - a)(1 + \varepsilon). \tag{25}$$

If we now assume the interval I to have Lebesgue measure $\frac{1}{2^k}$, i.e., $b - a = \frac{1}{2^k}$, using (25) the inequality (24) can be rewritten as

$$\left(\frac{1}{2^k} - 2\alpha \right)(1 - \varepsilon) \leq P\left(\tilde{T}(\tilde{x}) \in I \right) \leq \left(\frac{1}{2^k} + 2\alpha \right)(1 + \varepsilon),$$

which implies

$$-\frac{\varepsilon}{2^k} - 2\alpha(1 - \varepsilon) \leq P\left(\tilde{T}(\tilde{x}) \in I \right) - \frac{1}{2^k} \leq \frac{\varepsilon}{2^k} + 2\alpha(1 + \varepsilon).$$

Focusing on the worst case, we finally have

$$\left| P\left(\tilde{T}(\tilde{x}) \in I \right) - \frac{1}{2^k} \right| \leq \frac{\varepsilon}{2^k} + 2\alpha(1 + \varepsilon), \tag{26}$$

which leads to the relative error

$$\frac{\left| P\left(\tilde{T}(\tilde{x}) \in I \right) - \frac{1}{2^k} \right|}{\frac{1}{2^k}} \leq \varepsilon + 2^{k+1}\alpha(1 + \varepsilon). \tag{27}$$

4.2   Random Perturbations

We stress that the above results hold even assuming the function $\tilde{T}$ to be stochastic, i.e., perturbed by a small additive random noise. In such case the quantity $|T(x) - \tilde{T}(\tilde{x})|$ is a random variable, and the Proposition 1 and (26) hold provided to define α as the supremum of $|\xi(x) + \nu|$, being ν the stochastic fluctuation. As it will be made clearer in the following, we will consider pseudo-random perturbations of the pseudo-chaotic map $\tilde{T}$, to make the resulting nonlinear generator immune from short periodic cycles.

5   Nonlinear Recurrences Derived from the Rényi Map

There is an infinite number of different ways to define the digitized version of a chaotic system. In this section we propose a method for digitizing the Rényi maps, in order to obtain PRNGs based on nonlinear recurrences. Assuming $n \in \mathbb{N}$, with $n > 1$, we define the discrete domain as the following set of dyadic rationals

$$\Lambda_{2^n} = \left\{ \frac{q}{2^n} \in \mathbb{Q} : 0 \leq q < 2^n \right\}.$$
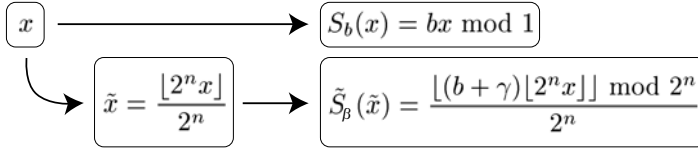
Fig. 6 The definition rules for the pseudo-chaotic Rényi map.

Given the Rényi map $S_b(x) = bx \bmod 1$, for any arbitrary small positive $\gamma \ll 1$ we define the digitized map $\tilde{S}_\beta : \Lambda_{2^n} \to \Lambda_{2^n}$ as

$$\tilde{S}_\beta \left(\frac{q}{2^n} \right) = \frac{1}{2^n} \left(\lfloor (b+\gamma)q \rfloor \bmod 2^n \right) = \frac{1}{2^n} \left(bq + \lfloor \gamma q \rfloor \bmod 2^n \right) \qquad (28)$$

The link between x and $\tilde{x}$ is defined assuming the truncation strategy to approximate x, i.e., if $x \in [0,1)$ it results

$$\tilde{x} = \frac{q(x)}{2^n} = \frac{\lfloor 2^n x \rfloor}{2^n}. \qquad (29)$$

Accordingly, any point x can be written as $x = \tilde{x} + \xi(x) = \frac{q(x)}{2^n} + \xi(x)$, where $\xi(x) \in [0, 2^{-n})$. Summarizing, S_b and $\tilde{S}_b$ are linked by the definition rules shown in Fig. 6. We spend few words on the role of the above introduced parameters γ and n. First of all, we stress that the digital architecture implementing (28) is a n-bit state machine, and for this reason we call the parameter n the digital resolution of the pseudo-chaotic system [28, 30]. Furthermore, it can be noticed that due to the modular calculations in (28) for any integer b the pseudo-chaotic version of $S_b(x) = bx \bmod 1$ agrees with the pseudo-chaotic version of $S_{b+2^n}(x) = (b+2^n)x \bmod 1$ (for a same value of γ in (28)). Moreover, it must be highlighted that if $0 \leq \gamma < \frac{1}{2^n-1}$ then for any $0 \leq q < 2^n$ the quantity $\lfloor \gamma q \rfloor$ is equal to zero: in such case the expression (28) defines a linear congruential generator. Accordingly, in order to make (28) a nonlinear congruential generator it must be

$$\frac{1}{2^n-1} \leq \gamma \ll 1. \qquad (30)$$

As it will be shown in the next Section, adopting the point of view discussed in Section 4 the greater are n and $1/\gamma$ and the better the dynamics of S is shadowed by $\tilde{S}$.

5.1 Properties of the Digitized Rényi Maps

In this Section we theoretically discuss some of the relationships that exist between the Rényi chaotic maps and their pseudo-chaotic versions (28). We face this problem in two steps. First, we discuss the link between the Rényi
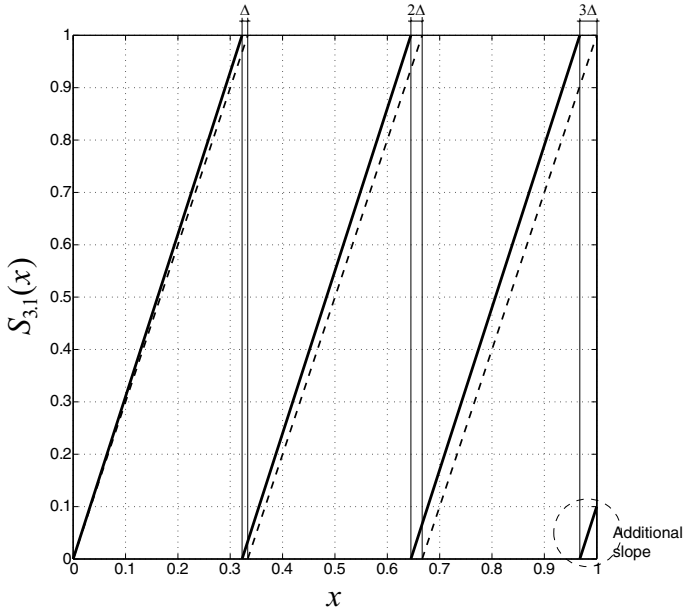
Fig. 7 The Rényi map $S_{3.1}$ (solid line) and the Rényi map S_3 (dashed line).
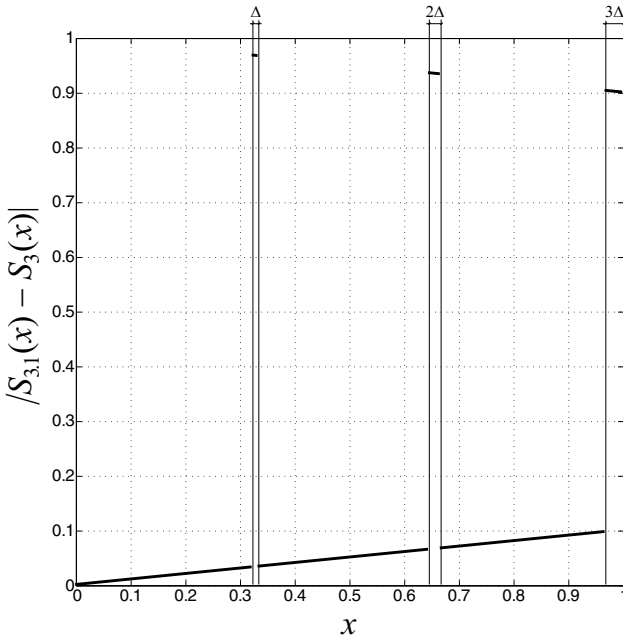
Fig. 8 The absolute error $|S_{3.1}(x) - S_3(x)|$ between the two Rényi maps $S_{3.1}$ and S_3.

map $S_\beta = (b + \gamma)x$ mod 1 and the Rényi map $S_b(x) = bx$ mod 1, and then we analyze how (28) approximates $S_b(x)$.

Accordingly, let us begin to analyze the link between S_β and S_b.

Referring to the Figs.7 and 8, it can be noticed that the higher absolute error $|S_\beta(x) - S_b(x)|$ is made around the discontinuity points of the maps, i.e., within the union of intervals

$$M = \bigcup_{i=1}^{b} \left[\frac{i}{\beta}, \frac{i}{b} \right).$$

(31)

The length of the i-th of these intervals is $i \cdot \Delta$, where $\Delta = \frac{\beta - b}{\beta b}$, and it can be easily verified that the absolute error in $[0, 1)/M$ is not greater than $b \left(1 - \frac{b}{\beta} \right)$. Despite the effects of the discontinuities, for small values of γ the dynamics of $S_\beta(x)$ and $S_b(x)$ within a limited time period follow trajectories that are close to each other for most values of x, in the sense specified by the following

Theorem 1. *Let us consider the interval*

$$K_1 = [0, 1 - \delta_1),$$

where $\delta_1 = \left(1 - \frac{b}{\beta} \right) \frac{b^{p+1} - b}{b - 1}$ *and* $p \in \mathbb{N}$. *For any* $x \in S_b^{-p}(K_1)$ *it results*

$$\left| S_\beta^p(x) - S_b^p(x) \right| \leq \left(1 - \frac{b}{\beta} \right) \frac{b^{p+1} - b}{b - 1}.$$

(32)

For proving the theorem, we first need the

Lemma 1. *Let us consider two points* x_1, x_2 *belonging to the same interval* $I = [\frac{i}{b}, \frac{i+1}{\beta})$, *with* $0 \leq i \leq b$. *Accordingly the two points do not belong to* M *and the two maps* S_b *and* S_β *have constant slope in* I. *It results*

$$|S_\beta(x_1) - S_b(x_2)| \leq b|x_1 - x_2| + b \left(1 - \frac{b}{\beta} \right).$$

(33)

Proof. We have $|S_\beta(x_1) - S_b(x_2)| = |S_\beta(x_1) - S_b(x_1) + S_b(x_1) - S_b(x_2)| \leq |S_\beta(x_1) - S_b(x_1)| + |S_b(x_1) - S_b(x_2)|$. The first term is not greater than the maximum absolute error $|S_\beta(x) - S_b(x)|$ in $[0, 1)/M$, that is $b \left(1 - \frac{b}{\beta} \right)$. On the other hand, since S_b is linear over I, we have $|S_b(x_1) - S_b(x_2)| = b|x_1 - x_2|$, and the lemma is proved.   □

We can now prove the Theorem 1.

Proof of Theorem 1. Let us assume that the two points $S_\beta^j(x)$ and $S_b^j(x)$ for $j = 0, 1, \ldots, p - 1$ belong to same intervals $I_j = [\frac{i_j}{b}, \frac{i_j+1}{\beta})$, with $0 \leq i_j \leq b$. From the previous Lemma we have

$$\left| S_\beta^{j+1}(x) - S_b^{j+1}(x) \right| \leq b \left| S_\beta^j(x) - S_b^j(x) \right| + b \left(1 - \frac{b}{\beta} \right),$$

and proceeding by induction it is easy to prove that for $j > 0$

$$\left| S_\beta^j(x) - S_b^j(x) \right| \leq \left(1 - \frac{b}{\beta} \right) \sum_{r=1}^j b^r = \left(1 - \frac{b}{\beta} \right) \frac{b^{j+1} - b}{b - 1}. \qquad (34)$$

The above inequality agrees with (32) by setting $j = p$, and the proof is completed if we show that if $x \in S_b^{-p}(K_1)$ then the two points $S_\beta^j(x)$ and $S_b^j(x)$ for $j = 0, 1, \ldots, p - 1$ belong to same intervals $I_j = [\frac{i_j}{b}, \frac{i_j+1}{\beta})$, with $0 \leq i_j \leq b$.

Accordingly, we define the set $P_b = \{\frac{i}{b}, 0 < i \leq b\}$ and we note that for $j = 0$ the previous condition is satisfied if the distance of x from the greater nearest point in P_b is not smaller than $(1 - \frac{b}{\beta})$: this is a sufficient condition for x not to belong to the set M defined in (31). For the second step, since we know from (34) that $|S_\beta(x) - S_b(x)| < b(1 - \frac{b}{\beta})$, a sufficient condition to have both of the points $S_\beta(x), S_b(x)$ in a same interval I_1 is to have $S_b(x)$ not closer than $b(1 - \frac{b}{\beta}) + (1 - \frac{b}{\beta})$ to the greater nearest point in P_b. Generalizing, for $j = 0, \ldots, p - 1$ both of the points $S_\beta^j(x), S_b^j(x)$ lye in a same interval I_j if $S_b^j(x)$ is not closer than

$$\left(1 - \frac{b}{\beta} \right) \sum_{r=0}^j b^r = \left(1 - \frac{b}{\beta} \right) \frac{b^{j+1} - 1}{b - 1}$$

to the greater nearest point in P_b.

Let us now focus on the set K_1: we will show that for $m = 1, \ldots, p$ the set $S_b^{-m}(K_1)$ contains only points that satisfy the previous sufficient conditions. In detail, we note that the set $S_b^{-1}(K_1)$ is made of b intervals of the form $[\frac{i}{b}, \frac{i}{b} + \frac{1 - \delta_1}{b})$ for $0 \leq i < b$, that is, each point in $S_b^{-1}(K_1)$ is not closer than $\frac{\delta_1}{b}$ to the greater nearest point in P_b. Following the same reasoning, it is easy to check that each point in $S_b^{-m}(K_1)$ is not closer than $\frac{\delta_1}{b^m}$ to the greater nearest point in P_b. Accordingly, in order to satisfy the previously discussed sufficient conditions, it suffices that

$$\frac{\delta_1}{b^{p-j}} \geq \left(1 - \frac{b}{\beta} \right) \frac{b^{j+1} - 1}{b - 1}, \qquad (35)$$

that is satisfied for $j = 0, \ldots, p - 1$ if

$$\delta_1 = (1 - \frac{b}{\beta}) \frac{b^{p+1} - b}{b - 1}, \qquad (36)$$

concluding the proof.   □
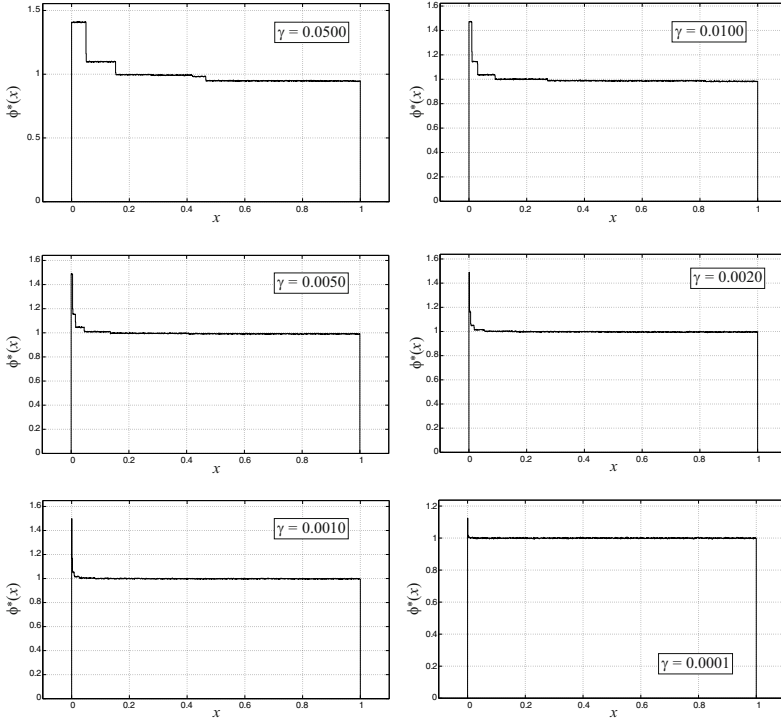
Fig. 9 The estimation of the invariant pdf for the Rényi maps $S_{3+\gamma}$, for several values of γ. From the Ergodic Theory of dynamical systems it results $\lim_{\beta \to b} \left\| \phi_\beta^* - u \right\|_1 = 0$, where u is the uniform pdf in $[0, 1)$ [21, 20].

The previous result indicates that for any $p \in \mathbb{N}$ if $\gamma \to 0$ the trajectory of $\{S_\beta^j(x)\}$ converges to the trajectory $\{S_b^j(x)\}$ uniformly over $\{0, \ldots, p\}$. Moreover, we stress that the invariant measure induced by the uniform pdf of the Rényi map S_b agrees with the Lebesgue measure of intervals, and in such case the Lebesgue measure of $S_b^{-p}(K_1)$ is equal to $1 - \delta_1$. Accordingly, for $\gamma \to 0$ δ_1 goes to zero and the property (32) holds almost everywhere in $[0, 1)$.

This fact is somehow reflected by the shape of the invariant pdf associated to S_β. In detail, the Ergodic Theory [20, 7] provides the tools for showing that

$$\lim_{\beta \to b} \left\| \phi_\beta^* - u \right\|_1 = 0, \tag{37}$$

i.e., the invariant pdf associated to S_β converges in $L^1([0, 1))$ to the invariant pdf of S_b, that is the uniform pdf. As it can be seen in Fig. 9, the presence of the additional slope highlighted in Fig. 7 causes an accumulation of the pdf around the point 0, and the effect vanishes as soon as γ goes to zero. A

rigorous theoretical approach to analyze this topic and to demonstrate the above limit is reviewed and discussed in [21, 20].

As a second step of our analysis, we investigate the link between $\tilde{S}_\beta$ and S_b. The procedure is similar to that one previously discussed, with the difference that we have to take into account, defining the map $\tilde{S}_\beta$, the effects of the truncations. In detail, we have the following

Theorem 2. *Let us consider the interval*

$$K_2 = [\delta_2, 1 - \delta_2),$$

with

$$\delta_2 = \frac{2\beta^{p+1}}{(\beta-1)2^n} + \left(1 + \frac{b}{\beta}\right)\frac{b^{p+1} - b}{b - 1}, \tag{38}$$

and $p, n \in \mathbb{N}$. For any $x \in S_b^{-p}(K_2)$ it results

$$\left|S_b^p(x) - \tilde{S}_\beta^p(\tilde{x})\right| < \frac{\beta^{p+1} - 1}{(\beta-1)2^n} + \left(1 - \frac{b}{\beta}\right)\frac{b^{p+1} - b}{b - 1}. \tag{39}$$

Proof. See the Appendix.

The previous theorem represent the main result of this work, since it explicitly relates the dynamics of the pseudo-chaotic map $\tilde{S}_\beta$ with the dynamics of the original chaotic Rényi map S_b. Interestingly, we highlight two aspect. First, by increasing n the effect of the digitization vanishes exponentially. Moreover, we recall from the discussion at the beginning of this Section that to have a nonlinear congruential generator the parameter $\gamma = \beta - b$ must be not smaller than $\frac{1}{2^n - 1}$. Accordingly, the larger is n and the more the parameter β can be set close to b. The larger is n, the more β can be set close to b and the more the interval K_2 covers the entire domain. Accordingly, by playing with n and β one can design a pseudo-chaotic map whose dynamics is arbitrarily close the the original chaotic Rényi map S_b, obtaining nonlinear recurrences.

Example

Let us consider the Rényi map S_3, and let us define its digitization by means of the 32 bit pseudo-chaotic map

$$\tilde{S}_{3+\gamma}\left(\frac{m}{2^{32}}\right) = \frac{1}{2^{32}}\left\lfloor (3+\gamma)m \bmod 2^{32}\right\rfloor, \tag{40}$$

where $\gamma = 6.886664777994155883789625 \cdot 10^{-6}$ and $\beta = $ `0x3.0000738A`. Adopting the binary fixed point representation, β can be written as

$$11.0000\ 0000\ 0000\ 0000\ 0111\ 0011\ 1000\ 1010. \tag{41}$$

As discussed in the next Section, the binary representation of β plays a key-role to determine the computation complexity involved by the PRNG based on the pseudo-chaotic Rényi maps. From Theorem 2 we have that

$$\left| S_3^p(x) - \tilde{S}_{3+\gamma}^p(\tilde{x}) \right| \leq \frac{(3+\gamma)^{p+1} - 1}{(3+\gamma)2^{32}} + \left(1 - \frac{3}{3+\gamma}\right) \frac{3^{p+1} - 3}{2} \approx$$

$$\approx \begin{cases} 2.328 \cdot 10^{-10}, & \text{if } p = 0, \\ \quad \vdots & \\ 2.755 \cdot 10^{-5}, & \text{if } p = 2, \\ \quad \vdots & \\ 7.528 \cdot 10^{-3}, & \text{if } p = 7 \ldots \end{cases} \tag{42}$$

The above inequalities hold for any point x such that $S^p(x) \in K_2 \approx [0.0075, 0.9925)$, that is calculated for $p = 7$. As a result, if we set $T = S_3^7$ and $\alpha = 7.528 \cdot 10^{-3}$, referring to the partition $\mathcal{P}_3$, from (27) we obtain that for any interval I_j of the partition that also belongs to K_2

$$\frac{\left| P\left(\tilde{T}(\tilde{x}) \in I_j\right) - \frac{1}{8} \right|}{\frac{1}{8}} \leq \varepsilon + 16 \cdot 7.528 \cdot 10^{-3}(1+\varepsilon). \tag{43}$$

If we assume ε in the order of 10^{-3} (i.e., according to a worst case indicated by the numerical analysis, see Fig. 2), the above inequality yields that the probability for the state $\tilde{x}$ to belong to any interval $I_j \subset K_2$ of the partition $\mathcal{P}_3$ after 7 iterations of the digitized Rényi map differs from $1/8$ by a relative error that is smaller than 12%, even if, in the worst case, the distribution of the initial state was very different from the uniform distribution. If the initial state is better uniformly distributed, the number of iterations decreases and the bounds decrease exponentially.

Theorem 2 does not tell us what may happen outside the interval K_2, at the borders of the phase space $[0, 1)$: if $x \notin T^{-p}(K_2)$ the 'shadowing error' made by the digitized map $\tilde{T}$ after p iterations can be much greater than the upper bound (24), due to the effects of the map discontinuities. Actually, in practical cases this fact has negligible consequences on the short-term statistical behavior of the digitized map, since its dynamics is typically observed adopting a coarse-grained resolution, greater than α (e.g., in this example we adopted an observation interval with length $1/8$, being α in the order of 10^{-3}).

Nevertheless, we stress that our point of view is statistical: we are not interested in following trajectories exactly; rather, we are interested in emulating the long-term statistical behavior of the original system. To this aim, we must introduce the pseudo-random perturbation of the digitized state trajectories, in order to emulate the trajectory instability of chaotic systems and to avoid the digitized system entering periodic stable orbits.

5.2 *Pseudo-random Perturbation of Digitized Chaotic Systems: Setting the Period Length*

It is well known that periodic trajectories are dense in chaotic attractors [31]. However, they are instable and the probability for a chaotic motion to enter a periodic orbit is zero. That is not what happens in a digitized system, for which all the trajectories are eventually periodic. In this work we propose to emulate the instability of chaotic trajectories by pseudo-randomly perturbing the digitized state $\tilde{x}$. To this aim, we remark that if the perturbation magnitude remains particularly small the shadowing property previously discussed still holds, as highlighted in Section 4.2.

Referring to the approach of Lasota in [25] the presence of noise in a chaotic system can be modeled as it follows. At each time step the chaotic sample x_{p+1} deviates from its ideal value due to a statistically independent noise ν_p added to the sample x_p, i.e.,

$$x_{p+1} = T(x_p + \nu_p). \tag{44}$$

As a result, in a chaotic system the presence of noise can cause the trajectories to diverge, e.g., bringing the perturbed state $x_p + \nu_p$ outside the basin of attraction of the chaotic attractor: this is not the case for a chaotic map like the Rényi map, defined as in (6). In general, the resulting process describes a discrete-time random walk in the phase space $[0, 1)$, that may not admit a stable stationary probability density function [25]. If the noisy samples $\{\nu_p\}$ are i.i.d. random variables statistically independent from the $\{x_m\}$ samples, it results that

$$\phi_{p+1} = \tilde{\Theta}_T(\phi_p \otimes f_\nu), \tag{45}$$

where f_ν is the pdf associated to the noisy samples, whereas the operator $\tilde{\Theta}_T$ is the extension of the Frobenius Perron operator Θ_T defined in (4) to the whole set of densities of bounded variations in $L^1([0, 1))$. It is worth noting that x_p only depends on the noisy samples $\nu_{p-1}, \nu_{p-2}, \ldots$ and if $\phi = \tilde{\Theta}_T(\phi \otimes f_\nu)$ then the pdf ϕ is stationary and invariant for the stochastic dynamical system (44). The characterization of the evolution of densities induced by (45) for systems like (44) is still an open theoretical problem, and depending on the considered case even the existence of an invariant density can be an undetermined issue [32, 25]. Nowadays, the consequence of the stochastic perturbations on the dynamics is typically studied resorting to 'computer simulations', and this is exactly what we do by digitizing the Rényi map and by adding a small perturbation noise to the digitized state.

Since our aim is to reproduce the instability of chaotic trajectory, we set the noise magnitude as small as possible, i.e., equal to the digitized resolution $1/2^n$. Accordingly, as discussed in the following Section, we perturb the less significant bit (LSB) of the digitized state $\tilde{x}$ by performing the XOR operation with the output bit of a LFSR. Simulation results show that the chief statistical behavior of the original Rényi map are preserved (See, e.g., Fig. 10).
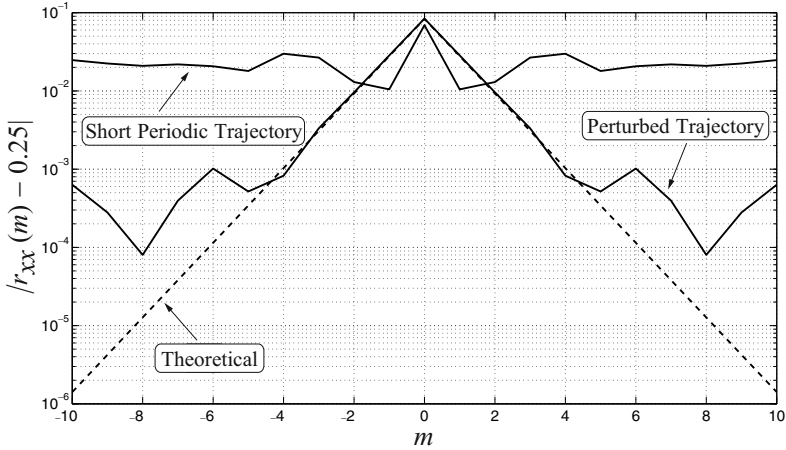
Fig. 10 The effect on the correlation decay of the LSB perturbation. The two trajectories were obtained using the digitized map (40), starting from the same initial condition. The autocorrelation functions were estimated on 100000 samples.

Even considering the pseudo-random perturbation, the resulting overall computing method is deterministic and the digitized dynamics is still eventually periodic with a period length that can be set greater than a minimum, according to the following

Proposition 2. *Let us consider the binary periodic sequences $y(p)$, $z(p)$ and $w(p) = y(p) \oplus z(p)$, for $p \in \mathbb{N}$, and let P_y, P_z, P_w be their respective periods. If $P_z > 1$ is prime, then $P_w = mP_z$, with $m \geq 1$.*

Proof. Since $P_z > 1$, it is immediate to verify that $P_w > 1$. If $P_w < P_y P_z$ an integer q exists such that $P_y P_z = qP_w + P_y P_z \bmod P_w$, with $0 \leq P_y P_z \bmod P_w < P_w$. On the other hand, $w(p) = y(p) \oplus z(p) = y(p + P_y P_z) \oplus z(p + P_y P_z) = w(p + P_y P_z) = w(p + qP_w + P_y P_z) = w(p + P_y P_z \bmod P_w)$. Since P_w is the smallest integer such that $w(p) = w(p + P_w)$, it must be $P_y P_z \bmod P_w = 0$. Accordingly it results $P_y P_z = hP_w$, that is $P_w = \frac{P_y P_z}{h}$. If P_z is prime, h divides P_y and $P_w = mP_z$.   □

Referring to the above notation, we can set $w(p)$ as the LSB of $\tilde{x}_p$, $y(p)$ as the less significant bit of $\tilde{S}(\tilde{x}(p-1))$ and $z(p)$ as the output bit of an LFSR. According to the above proposition, if the LFSR has prime period P_z, then the perturbed trajectories of $\tilde{x}$ have period that is not smaller than P_z. By recalling that a LFSR of order k has period $P_z = 2^k - 1$, we remark that P_z is a Mersenne prime if we set $k = 31$ or $k = 61$, obtaining P_z equal to $2147483647 \approx 2.15 \cdot 10^9$ and $2305843009213693951 = 2.3 \cdot 10^{18}$, respectively. These choices assure that the period of the generated digitized sequences can be made long enough for our cryptographic purposes, regardless of the initial condition of the nonlinear recurrence (28).

5.3   On the Hardware/Software Implementation of the Digitized Rényi Map

We spend few comments about the hardware/software implementation of the digitized Rényi map, noting that the only operations involved in the calculation are 2^n-modular additions (to calculate the multiplication), with a truncation of the result to the n most significant bits. The number of additions is equal to the number of '1' in the binary representation of β, and the addition operations can be easily performed referring to efficient solutions like carry-save adders [30]. In the example previously discussed, ten 2^n-modular additions suffices.

6   PRNGs Derived from the Rényi Map: Design and Testing

When dealing with PRNGs and statistical tests, one point must be clear: for any given PRNGs it is always possible to define a 'never-passing' test. This is because of the deterministic and periodic nature of PRNGs, that always introduces some undesired defects in the statistical characteristics of the generated sequences. Nevertheless, the results discussed in this paper indicate that for any given finite-time statistical test there is an infinite number of PRNGs based on the pseudo-chaotic Rényi map (28) that pass that test. This is because of the two following remarks:

1. with any Rényi chaotic map S_b (with integer b) it is always possible to obtain an ideal TRNG, by adopting a proper symbolic coding of its dynamics [20];
2. by playing with the parameters n and γ one can use a map $\tilde{S}_\beta$ to approximate the chaotic Rényi map S_b according to any arbitrary accuracy over finite time-windows.

The above point 2. can be faced adopting the two different strategies identified in the preliminary conclusions A) and B), discussed in Section 3.2, taking in mind that the *required accuracy of the approximation depends on the target application* (e.g., it depends on the statistical tests taken into account). Before introducing two specific examples related to these two mentioned approaches, it is worth discussing how to perform the domain partitioning.

6.1   Domain Partitioning in Digitized Chaotic Systems

The simplest approximated way to generate numbers referring to the rule (11) is to compare the digitized state $\tilde{x}$ with the endpoints of a symbolic partition: in general, up to $M - 1$ comparisons are required for a partition made of M intervals. This operation introduces the problem of comparing dyadic

rationals, since even the partition endpoints must be represented according to a digitization strategy. For example, the endpoints $1/3$ and $1/6$ of the natural generating symbolic partition of the map S_3 can not be represented adopting a conventional binary fixed point representation of numbers, and the same happens for the endpoints associated to other maps S_b with $b \neq 2^m$.

On the other hand, it is very easy to apply the generation rule (11) when the partition is of the form $\mathcal{P}_k$, i.e., a partition made of 2^k equal intervals. In such case, no comparison is necessary since the most k significant bits of the state $\tilde{x}$ indicate the number of the belonging interval in the partition, i.e., the generated number of the PRNG. Adopting the same reasoning discussed in the preliminary conclusions of the Section 3.2, the partition $\mathcal{P}_k$ can be used without introducing correlation between symbols if b is of the form $q2^k$, with $q \in \mathbb{N}^+$.

PRNGs design – approach (A)

According to the approach A) discussed in Section 3.2, we divide the domain $[0, 1)$ adopting a refinement of the natural generating symbolic partition associated to the map S_b, in order to approximate an *ideal* TRNG issuing b random integers. Since we refer to partitions of the form $\mathcal{P}_k$, we have to digitize maps S_b with $b = m \cdot 2^k$, for any $m \in \mathbb{N}, m > 1$. As an example, in Table 1 we reported the NIST SP800.22 test [12] results for a PRNG based on $\tilde{T} = \tilde{S}_\beta$, with $n = 64$ bits and $\beta = $ `0x4.000000000001838A` ($b = 4$). At each time step the 2 most significant bits of $\tilde{x}$ were output ($\mathcal{P}_k = \mathcal{P}_2$). The minimum pass rate for each statistical test with the exception of the Random Excursion Variant test is approximately 0.960150 for a sample size of 100 binary sequences (2 million bits each). The minimum pass rate for the Random Excursion Variant test is approximately 0.954323 for a sample size $= 70$ binary sequences (2 million bits each).

PRNGs design – approach (B)

According to the approach b) discussed in Section 3.2, we divide the domain $[0, 1)$ with an arbitrary partition $\mathcal{P}_n$ of 2^n intervals, in order to approximate a *not-ideal* TRNG. In such case the 2^n random integers are uniformly distributed, but they are affected by a correlation that has a vanishing decay. Accordingly, an ideal TRNG issuing 2^n random integers can be approximated with any accuracy by under-sampling the sequence $\{S^i(x), i \in \mathbb{N}\}$ by a factor p, that is referring to the sequence $\{S^{ip}(x), i \in \mathbb{N}\}$. As an example, we report the NIST SP800.22 test results for a PRNG based on $\tilde{T} = \tilde{S}_\beta^8$, with $n = 32$ bits and $\beta = $ `0x7.000000AA0` ($b = 7, p = 8$). Referring to the map $\tilde{T}$, at each time step the 8 most significant bits of $\tilde{x}$ were output ($\mathcal{P}_k = \mathcal{P}_8$). The tests were performed on a sample size of 100 binary sequences (2 million bits each), as in the previous example.

Table 1 NIST800.22 test results for the PRNG based on $\tilde{T} = \tilde{S}_\beta$, with $n = 64$ bits and $\beta = $ `0x4.000000000001838A`. At each time step the 2 most significant bits of $\tilde{x}$ were output.

```
-------------------------------------------------------------------------------
RESULTS FOR THE UNIFORMITY OF P-VALUES AND THE PROPORTION OF PASSING SEQUENCES
-------------------------------------------------------------------------------
    generator is <N64K2P1_4.000000000001838A_2Mb_100Seq.dat>
-------------------------------------------------------------------------------
C1   C2   C3   C4   C5   C6   C7   C8   C9 C10  P-VALUE  PROPORTION  STATISTICAL TEST
-------------------------------------------------------------------------------
 7    6   15   14   13   14    9   14    5    3  0.032923   0.9900    Frequency
11   14   10    7    5   13   10   14   10    6  0.419021   0.9800    BlockFrequency
11    6    6   14   13   10   14   11    5   10  0.350485   0.9900    CumulativeSums
 8    9    8    6   14   13   15    7   10   10  0.494392   0.9800    CumulativeSums
 8    8   14    8    7    8    9   16   12   10  0.514124   1.0000    Runs
 9    7   13   11    8    7    8   14   11   12  0.759756   1.0000    LongestRun
 4   10   11   16   13   11   11   10    6    8  0.319084   1.0000    Rank
 7    6   16    9   11   14    7   11   13    6  0.249284   0.9900    FFT
                                     mean:      0.459514   0.9916    NonOverlTemplate (*)
                       standard deviation:      0.292429   0.0095
 8    9   10    7   15    7   12   12   10   10  0.779188   0.9900    OverlappingTemplate
 9    8   11   10    5   16   15   15    5    6  0.071177   0.9800    Universal
10   13   11   10    9    9    7   10    9   12  0.978072   0.9900    ApproximateEntropy
                                     mean:      0.441598   0.9911    RandomExcursions (*)
                       standard deviation:      0.278924   0.0106
                                     mean:      0.612595   0.9960    RandomExcVariant (*)
                       standard deviation:      0.225489   0.0082
 8    8   14   10    7   13    8   11    8   13  0.739918   1.0000    Serial
12   10   13   12   10    7    5    8   12   11  0.739918   0.9800    Serial
11   11    8    8   11    6   18    7   12    8  0.289667   0.9900    LinearComplexity
-------------------------------------------------------------------------------
(*) Test with multiple results: mean value and standard deviation were reported.
-------------------------------------------------------------------------------
```

Comments on the test results

We stress that the above presented good results are typical, even if changing the parameter γ to a different small value, provided to satisfy (30). We have tried at random dozens of PRNGs with $n = 32, 64$ and $b = 3, 4, 5, 6, 7, 8$, with different values of γ, obtaining similar results. If $b = 4$ (or 8) we output the 2 (or 3) most significant bits of $\tilde{x}$ at each time step. For the other values, we set an under-sampling rate equal to $p = 8$, whereas outputting the most significant byte (8 bits). As a general trend, it seems that for the kind of generators proposed in this paper the most sensitive test is the Non Overlapping Template. The focus of this test, according to the default parameters, is the number of occurrences of 148 different non-periodic patterns of 9-bits. In some cases it happened that the passing rate for some patterns dropped to 96.0% when analyzing 100 sequences of 2 million bits each (the theoretical requested minimum passing rate is 96.0150%). Nevertheless, apart from this minor and occasional statistical imperfection, in all of the evaluated PRNGs none of the tests in the NIST SP800.22 standard were badly failed (the worst PRNG found had a 94.0% passing rate in one test).

Table 2 NIST800.22 test results for the PRNG based on $\tilde{T} = \tilde{S}_\beta^8$, with $n = 32$ bits and $\beta = $ `0x7.000000AA0`. Every eight iterations of $\tilde{S}_\beta^8$ the 8 most significant bits of $\tilde{x}$ were output.

```
--------------------------------------------------------------------------------
RESULTS FOR THE UNIFORMITY OF P-VALUES AND THE PROPORTION OF PASSING SEQUENCES
--------------------------------------------------------------------------------
   generator is <N32K8P8_7.000000AA0_2Mb_100Seq.dat>
--------------------------------------------------------------------------------
C1  C2  C3  C4  C5  C6  C7  C8  C9 C10  P-VALUE  PROPORTION  STATISTICAL TEST
--------------------------------------------------------------------------------
 7  10  10   7  10  14  10   6  15  11  0.574903   0.9900    Frequency
10  11   9  10   8   9  12  10   8  13  0.983453   1.0000    BlockFrequency
 8   6  14  10  10   8  10  15   5  14  0.304126   1.0000    CumulativeSums
 8  11   6  11   8  11   8  12   9  16  0.616305   0.9900    CumulativeSums
15   9   8   5  17   7  12  10  11   6  0.145326   1.0000    Runs
15  12  11   9   9   7   9  13   9   6  0.657933   0.9800    LongestRun
 9  10  15   8   8   9   8   3  18  12  0.075719   0.9800    Rank
14   7   7  13  10  10  10   5  17   7  0.181557   0.9900    FFT
                            mean:  0.495993   0.9895    NonOverlTemplate (*)
                standard deviation:  0.295015   0.0094
10  16  13   8   9   8  16   7   9   4  0.137282   0.9700    OverlappingTemplate
12  10  11  13   9  13   9  12   9   2  0.401199   0.9900    Universal
 9   8   9   9  10  10  11  17   8   9  0.719747   0.9700    ApproximateEntropy
                            mean:  0.368254   0.9896    RandomExcursions (*)
                standard deviation:  0.281558   0.0126
                            mean:  0.368254   0.9896    RandomExcVariant (*)
                standard deviation:  0.281559   0.0126
11  11  11  12  10  12   6   4  12  11  0.657933   0.9900    Serial
11  11  14   7  10   9   6  12  11   9  0.834308   1.0000    Serial
 8  14   7   9  10  10  12  14  11   5  0.574903   1.0000    LinearComplexity
--------------------------------------------------------------------------------
(*) Test with multiple results: mean value and standard deviation were reported.
--------------------------------------------------------------------------------
```

7   Conclusions

In this work we have proposed the design of PRNGs based on nonlinear recurrences derived from the Rényi chaotic map. Starting from a weaker interpretation of the Shadowing Theory proposed by Coomes et al. we have theoretically framed the relationship that exists between the Rényi chaotic maps and their digitized versions. Exploiting the ergodic properties of the original systems we have proposed a method for the design of nonlinear recurrences whose statistical behavior can be analyzed in terms of chaotic dynamics approximation. In order to overcome the problem of the shortness in the period length of the digitized trajectories, we have proposed to perturb the pseudo-chaotic dynamics in such a way to emulate the orbit instability peculiar to chaotic systems. Statistical tests confirm the validity of the approach, that is general and define an infinite family of PRNGs that can be used to approximate the statistical behavior of an ideal TRNG with an arbitrary accuracy.

References

1. Menezes, A., Van Oorschot, P., Vanstone, S.: Handbook of Applied Cryptography. CRC Press, Boca Raton (1997)
2. Petrie, C., Connelly, A.: A noise-based IC random number generator for applications in cryptography. IEEE Transaction on Circuits and Systems I 47(5), 615–621 (2000)
3. Addabbo, T., Alioto, M., Fort, A., Rocchi, S., Vignoli, V.: A variability-tolerant feedback technique for throughput maximization of TRBGs with predefined entropy. Journal of Circuits, Systems and Computers 19(4), 1–17 (2010)
4. Addabbo, T., Alioto, M., Fort, A., Rocchi, S., Vignoli, V.: A feedback strategy to improve the entropy of a chaos-based random bit generator. IEEE Transaction on Circuits and Systems – part I 53(2), 326–337 (2006)
5. Callegari, S., Rovatti, R., Setti, G.: Embeddable ADC-based true random number generator for cryptographic applications exploiting nonlinear signal processing and chaos. IEEE Trans. on Signal Processing 53(2), 793–805 (2005)
6. Bucci, M., Germani, L., Luzzi, R., Tommasino, P., Trifiletti, A., Varanonuovo, M.: A high-speed IC random-number source for smartcard microcontrollers. IEEE Transaction Circuits and Systems I 50(11), 1373–1380 (2003)
7. Walters, P.: An Introduction to Ergodic Theory. Springer, Heidelberg (1982)
8. Boyar, J.: Inferring sequences produced by pseudo-random number generators. Journal of the ACM 36(1), 129–141 (1989)
9. Plumstead, J.B.: Inferring a sequence produced by a linear congruence. In: CRYPTO, pp. 317–319 (1982)
10. Knuth, D.: The art of computer programming, 2nd edn., vol. 2. Addison-Wesley, Reading (1981)
11. Blum, L., Blum, M., Shub, M.: A simple unpredictable pseudo-random number generator. SIAM Journal on Computing 15(2), 364–383 (1986)
12. NIST Special Publication 800-22 Rev.1a: A statistical test suite for random and pseudorandom number generators for cryptographic applications (April 2010)
13. Coomes, B., Kocak, H., Palmer, K.: Shadowing in Discrete Dynamical Systems. In: Six Lectures on Dynamical Systems, pp. 163–211. World Scientific, Singapore (1996)
14. Eichenauer-Herrmann, J.: Pseudorandom number generation by nonlinear methods. International Statistical Reviews 63, 247–255 (1995)
15. Tezuka, S.: Uniform Random Numbers: Theory and Practice. Kluwer Academic Publishers, Dordrecht (1995)
16. Eichenauer-Herrmann, J.: Inversive congruential pseudorandom numbers avoid the planes. Mathematics of Computation 56, 297–301 (1991)
17. Golomb, S.W.: Shift Register Sequences. Aegean Park, Laguna Hills (1982)
18. Eichenauer, J., Topuzŏglu, A.: On the period length of congruential pseudo-random number sequences generated by inversions. Journal of Computational and Applied Mathematics 31, 87–96 (1990)
19. Beyer, W.A., Roof, R.B., Williamson, D.: The lattice structure of multiplicative congruential pseudo-random vectors. Mathematics of Computation 25(114), 345–363 (1971)
20. Boyarsky, A., Góra, P.: Laws of Chaos. Birkhäuser, Basel (1997)

21. Addabbo, T., Fort, A., Rocchi, S., Papini, D., Vignoli, V.: Invariant measures of tunable chaotic sources: Robustness analysis and efficient computation. IEEE Transactions on Circuits and Systems - I 56(4), 806–819 (2009)
22. Addabbo, T., Fort, A., Papini, D., Rocchi, S., Vignoli, V.: An efficient and accurate method for the estimation of entropy and other dynamical invariants for piecewise affine choatic maps. International Journal of Bifurcation and Chaos 19(12), 4175–4195 (2009) (accepted)
23. Setti, G., Mazzini, G., Rovatti, R., Callegari, S.: Statistical modeling of discrete-time chaotic processes: basic finite-dimensional tools and applications. Proc. of the IEEE 90(5), 662–690 (2002)
24. Stojanovski, T., Kocarev, L.: Chaos-based random number generator – part I: Analysis. IEEE Transactions on Circuits and Systems I 48(3), 281–288 (2001)
25. Lasota, A., Mackey, M.C.: Chaos, Fractals and Noise - Stochastic Aspects of Dynamics, 2nd edn. Springer, Heidelberg (1994)
26. Amigó, J., Kocarev, L., Tomovski, I.: Discrete entropy. Physica D 228, 77–85 (2007)
27. Kocarev, L., Szczepanski, A.J.: Discrete chaos–I: Theory. IEEE Transaction on Circuits and Systems – I 53(6), 1300–1309 (2006)
28. Addabbo, T., Alioto, M., Fort, A., Pasini, A., Rocchi, S., Vignoli, V.: A class of maximum-period nonlinear congruential generators derived from the Rényi chaotic map. IEEE Transactions on Circuits and Systems - I 54(4), 816–828 (2007)
29. Addabbo, T., Fort, A., Kocarev, L., Rocchi, S., Vignoli, V.: Pseudo-chaotic lossy compressors for true random number generation. IEEE Transaction on Circuits and Systems I (2010) (accepted) doi: 10.1109/TCSI.2011.2108050
30. Addabbo, T., De Caro, D., Fort, A., Petra, N., Rocchi, S., Vignoli, V.: Efficient implementation of pseudochaotic piecewise linear maps with high digitization accuracies. International Journal of Circuit Theory and Applications 39(4) (April 2010)
31. Devaney, R.: An Introduction to Chaotic Dynamical System, 2nd edn. Addison-Wesley, Reading (1989)
32. Pareschi, F., Setti, G., Rovatti, R.: Noise robustness condition for chaotic maps with piecewise constant invariant density. In: Malek, M., Reitenspiess, M., Kaiser, J. (eds.) ISAS 2004. LNCS, vol. 3335, pp. 681–684. Springer, Heidelberg (2005)

Appendix

Proof of Theorem 2

We first need the following

Lemma 2. *Let $\frac{q_0}{2^n} \in \Lambda_n$ and $z > \frac{q_0}{2^n}$. If the restriction of the Rényi map S_β to the interval $J = \left[\frac{q_0}{2^n}, z\right)$ is continuous, for any $x \in J$ and for any $m \in \mathbb{N}$, with $q_0 \leq m \leq \lfloor 2^n z \rfloor$, it results*

$$\left| S_\beta(x) - \tilde{S}_b\left(\frac{m}{2^n}\right) \right| < \beta \left| x - \frac{m}{2^n} \right| + \frac{1}{2^n}.$$

Proof. We note that if S_β is continuous over J, then it has a constant slope: for any $x_1, x_2 \in J$ we have $|S_\beta(x_2) - S_\beta(x_1)| = \beta|x_2 - x_1|$. Moreover, we note that for any $\frac{m}{2^n} \in \Lambda_n$ it results $\left|S_\beta\left(\frac{m}{2^n}\right) - \tilde{S}_\beta\left(\frac{m}{2^n}\right)\right| < \frac{1}{2^n}$, indeed

$$
\tilde{S}_\beta\left(\frac{m}{2^n}\right) = \frac{\lfloor \beta m \bmod 2^n \rfloor}{2^n} \leq \frac{\beta m \bmod 2^n}{2^n} = S_\beta\left(\frac{m}{2^n}\right) <
$$
$$
< \frac{\lfloor \beta m \bmod 2^n \rfloor + 1}{2^n} = \tilde{S}_\beta\left(\frac{m}{2^n}\right) + \frac{1}{2^n}. \tag{46}
$$

Accordingly,

$$
\left|S_\beta(x) - \tilde{S}_\beta\left(\frac{m}{2^n}\right)\right| = \left|S_\beta(x) - S_\beta\left(\frac{m}{2^n}\right) + S_\beta\left(\frac{m}{2^n}\right) - \tilde{S}_\beta\left(\frac{m}{2^n}\right)\right| \leq
$$
$$
\leq \left|S_\beta(x) - S_\beta\left(\frac{m}{2^n}\right)\right| + \left|S_\beta\left(\frac{m}{2^n}\right) - \tilde{S}_\beta\left(\frac{m}{2^n}\right)\right| < \beta\left|x - \frac{m}{2^n}\right| + \frac{1}{2^n},
$$

concluding the proof. $\square$

We can now prove the main Theorem.

Proof of Theorem 2. We will prove the theorem focusing on the inequality

$$
\left|S_b^p(x) - \tilde{S}_\beta^p(\tilde{x})\right| \leq \left|S_b^p(x) - S_\beta^p(x)\right| + \left|S_\beta^p(x) - \tilde{S}_\beta^p(\tilde{x})\right|, \tag{47}
$$

and noting that $K_2 \subset K_1$. Let us assume the Rényi map continuous on intervals containing the points $S_\beta^j(x)$ and $\tilde{S}_\beta^j(\tilde{x})$, for $j = 0, 1, \ldots, p-1$. From the Lemma 2 we have

$$
\left|S_\beta^{j+1}(x) - \tilde{S}_\beta^{j+1}(\tilde{x})\right| < \beta\left|S_\beta^j(x) - \tilde{S}_\beta^j(\tilde{x})\right| + \frac{1}{2^n},
$$

and proceeding by induction it is easy to prove that

$$
\left|S_\beta^j(x) - \tilde{S}_\beta^j(\tilde{x})\right| < \beta^j|x - \tilde{x}| + \frac{1}{2^n}\sum_{r=0}^{j-1}\beta^r = \beta^j|x - \tilde{x}| + \frac{\beta^j - 1}{(\beta - 1) \cdot 2^n}. \tag{48}
$$

Due to the truncation strategy (29) we have $|\tilde{x} - x| < 1/2^n$, and the previous inequality becomes

$$
\left|S_\beta^j(x) - \tilde{S}_\beta^j(\tilde{x})\right| < \frac{\beta^{j+1} - 1}{(\beta - 1)2^n}. \tag{49}
$$

Since $K_2 \subset K_1$ we can use the result (32), and by setting $j = p$ the ineq. (47) is verified. The proof is completed if we show that if $x \in S_b^{-p}(K_2)$ then some intervals exist containing the points $S_\beta^j(x)$ and $\tilde{S}_\beta^j(\tilde{x})$, for $j = 0, 1, \ldots, p-1$, and in which the Rényi map is continuous (this allows the use of the previous Lemma).

Accordingly, we define the set $P_\beta = \{\frac{i}{\beta}, 0 < i \le b\}$ and we note that for $j = 0$ the previous condition is satisfied if the distance of x from the nearest point in P_β is not smaller than $\frac{1}{2^n}$: this is a sufficient condition to avoid a discontinuity point of S_β lying between x and $\tilde{x}$. For the second step, since we know from (49) that $|S_\beta(x) - S_b(x)| < \frac{\beta+1}{2^n}$, a sufficient condition to have both of the points $S_\beta(x), \tilde{S}_\beta(\tilde{x})$ in a same interval between two discontinuity points of S_β is to have $S_\beta(x)$ not closer than $\frac{1}{2^n} + \frac{\beta+1}{2^n}$ to the nearest point in P_β. Generalizing, for $j = 0, \ldots, p-1$ both of the points $S_\beta^j(x), S_b^j(x)$ lye in a same interval between two discontinuity points of S_β if $S_\beta^j(x)$ is not closer than

$$\frac{1}{2^n}\left(1 + \sum_{r=0}^{j} b^r\right) = \frac{\beta^{j+1} + \beta - 2}{(\beta - 1)2^n} \tag{50}$$

to the nearest point in P_β.

By using the result (32), the above condition is verified if $S_b^j(x)$ is not closer than

$$\frac{\beta^{j+1} + \beta - 2}{(\beta - 1)2^n} + \left(1 - \frac{b}{\beta}\right)\frac{b^{j+1} - b}{b - 1} \tag{51}$$

to the nearest point in P_β. Again, the latter condition is verified if $S_b^j(x)$ is not closer than

$$\left(1 - \frac{b}{\beta}\right) + \frac{\beta^{j+1} + \beta - 2}{(\beta - 1)2^n} + \left(1 - \frac{b}{\beta}\right)\frac{b^{j+1} - b}{b - 1} =$$
$$= \frac{\beta^{j+1} + \beta - 2}{(\beta - 1)2^n} + \left(1 - \frac{b}{\beta}\right)\frac{b^{j+1} - 1}{b - 1} \tag{52}$$

to the nearest point in P_b.

Let us now focus on the set K_2: we will show that for $m = 1, \ldots, p$ the set $S_b^{-m}(K_2)$ contains only points that satisfy the previous sufficient conditions. In detail, we note that the set $S_b^{-1}(K_2)$ is made of b intervals of the form $[\frac{i}{b} + \frac{\delta_2}{b}, \frac{i}{b} + \frac{1-\delta_2}{b})$ for $0 \le i < b$, that is, each point in $S_b^{-1}(K_2)$ is not closer than $\frac{\delta_2}{b}$ to the nearest point in P_b. Following the same reasoning, it is easy to check that each point in $S_b^{-m}(K_2)$ is not closer than $\frac{\delta_2}{b^m}$ to the nearest point in P_b. Accordingly, in order to satisfy the previously discussed sufficient conditions, it suffices that

$$\frac{\delta_2}{b^{p-j}} \ge \frac{\beta^{j+1} + \beta - 2}{(\beta - 1)2^n} + \left(1 - \frac{b}{\beta}\right)\frac{b^{j+1} - 1}{b - 1}, \tag{53}$$

that is

$$\delta_2 \ge b^{p-j}\frac{\beta^{j+1} + \beta - 2}{(\beta - 1)2^n} + \left(1 - \frac{b}{\beta}\right)\frac{b^{p+1} - b^{p-j}}{b - 1}. \tag{54}$$

On the other hand for $j = 0, \ldots, p - 1$,

$$\frac{b^{p+1} - b^{p-j}}{b - 1} < \frac{b^{p+1} - b}{b - 1},\tag{55}$$

and

$$b^{p-j} \frac{\beta^{j+1} + \beta - 2}{(\beta - 1)2^n} \leq \beta^{p-j} \frac{\beta^{j+1} + \beta}{(\beta - 1)2^n} = \frac{\beta^{p+1} \left(1 + \beta^{-j}\right)}{(\beta - 1)2^n} < \frac{2\beta^{p+1}}{(\beta - 1)2^n}.\tag{56}$$

As a result, if $\delta_2 = \frac{2\beta^{p+1}}{(\beta-1)2^n} + \left(1 + \frac{b}{\beta}\right) \frac{b^{p+1}-b}{b-1}$ the inequality (54) is verified and the proof is completed. $\square$

Chapter 4
Formation of High-Dimensional Chaotic Maps and Their Uses in Cryptography

Wallace K.S. Tang[1] and Ying Liu[2]

[1] Department of Electronic Engineering, City University of Hong Kong
[2] Department of Information Science and Electronic Engineering, Zhejiang University

Abstract. Being a particular class of nonlinearity, chaos nowadays becomes one of the most well known and potentially useful dynamics. Although a chaotic system is only governed by some simple and low order deterministic rules, it possesses many distinct characteristics, such as deterministic but random-like complex temporal behavior, high sensitivity to initial conditions and system parameters, fractal structure, long-term unpredictability and so on. These properties have been widely explored for the last few decades and found to be useful for many engineering problems such as cryptographic designs, digital communications, network behaviour modeling, to name a few. The increasing interests in chaos-based applications have also ignited tremendous demand for new chaos generators with complicate dynamics but simple designs. In this chapter, two different approaches are described for the formation of high-dimensional chaotic maps and their dynamical characteristics are studied. As reflected by the statistical results, strong mixing nature is acquired and these high-dimensional chaotic maps are ready for various cryptographic usages. Firstly, it is used as a simple but effective post-processing function which outperforms other common post-processing functions. Based on such a chaos-based post-processing function, two types of pseudo random number generators (PRNGs) are described. The first one is a 32-bit PRNG providing a fast and effective solution for random number generation. The second one is an 8-bit PRNG system design, meeting the challenge of low bit-precision system environment. The framework can then be easily extended for some practical applications, such as for image encryption. Detailed analyses on these applications are carried out and the effectiveness of the high-dimensional chaotic map in cryptographic applications is confirmed.

1 Introduction

Chaotic maps are with a long history in nonlinear dynamical studies, closely relating to the modeling of natural processes. For example, logistic map is a discrete-time version of the logistic equation describing the population growth

proposed by Pierre Verhulst in 1845 [38,39]; Hénon map is a simplified model [13] of the Poincaré section of Lorenz system, a model of natural convection rolls derived by Edward Lorenz in 1963; the horseshoe map was introduced by Stephen Smale [34] when Van der Pol oscillator was studied.

The distinct properties of chaos, especially its extreme sensitivity to tiny variations of initial conditions and system parameters, have granted it to be a good candidate for cryptographic algorithms. In fact, chaos-based algorithm (CBA) has shown some exceptionally good properties in many aspects regarding security, complexity, speed, computing power and computational overhead, etc [22,44]. Unlike conventional cryptographic algorithms which are mainly based on discrete mathematics, CBA is relied on the complex dynamics of chaotic maps which are deterministic but simple in structures.

The huge potential of chaotic maps in various applications has also ignited great demands of new chaotic maps with more complicate and nicer dynamical nature. By referring to the dimensionality of the maps, they are generally categorized as one-dimensional, two-dimensional and so on (some examples of chaotic maps are given in Appendices). The designs of low-dimensional chaotic maps are basically relied on the non-linear or piecewise-linear transformations, so that the necessity expansion and contraction can be obtained in the same invariant set. In contrast, the formation of high-dimensional chaotic maps is rather ad hoc. Therefore, it is desirable if some systematic ways can be obtained for the derivation of chaotic maps with any desired dimension.

One of the possible approaches is based on the anti-control or chaotification of non-chaotic system. It is proved that an n-dimensional discrete-time linear system can be driven to chaos by feeding nonlinear controls, such as sinusoidal function [45], hyperbolic tangent function [18], cubic function and so on.

High-dimensional chaotic maps can also be obtained by weakly coupling some low-dimensional ones [12], or by the formation of the coupled map lattices [16,37,42,43] and the globally coupled map [2,26,29,46]. However, the characteristics of the resultant maps may largely vary from their corresponding low-dimensional ones.

In this chapter, we are interested in extending the low-dimensional chaotic mapping function to high-dimensional space. It is suggested in [11] that, high dimensional chaotic map can be obtained by replacing the scalar values of the transformation matrix of a conventional Cat map by symmetric m-matrices with natural entries. A more general approach, known as multi-dimensional generalization, is to be presented with the proof of some important properties, such as area-preservative and positive Lyapunov exponent. Another possible way to form a chaotic map with any dimension is known as spatial extension. It is inspired by the design given in [6,23] where 3-dimensional Cat map and 3-dimensional baker map are described. The new design concept is to extend the space domain of some low-dimensional maps, and apply the mapping in a sub-space manner over the entire high-dimensional space. The dynamical characteristics of the high-dimensional chaotic maps formed by multidimensional generalization and spatial extension are then investigated and their practical usages in cryptography are also explored.

2 Formulation of High-Dimensional Chaotic Maps

2.1 Multidimensional Generalization

Recall the conventional two-dimensional Cat map given as below:

$$f: \left\{ \begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} 1 & a \\ b & 1+ab \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} \right\} \bmod 1 \tag{1}$$

where $0 < a, b \leq 1$ are real-value parameters, an m-dimensional Cat map can be devised as follows:

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \\ \vdots \\ x_m(k+1) \end{bmatrix} = M \begin{bmatrix} x_1(k) \\ x_2(k) \\ \vdots \\ x_m(k) \end{bmatrix} \bmod 1 \tag{2}$$

where

$$M = \begin{bmatrix} I_p & M_a \\ M_b & I_q + M_b M_a \end{bmatrix} \tag{3}$$

with $q=(m\text{-}p)$; I_p and I_q are identity matrices of sizes p and q, respectively; M_a and M_b are $p \times q$ and $q \times p$ non-zero matrices. The entries of M_a and M_b can be freely chosen, providing that M is full rank. It is remarked that the formation of M in (3) can be considered as replacing 1, a and b in (1) by the corresponding matrix elements I, M_a and M_b, respectively. Just like the conventional Cat map, it can be proved that the high-dimensional map given in (2) is area-preservative and its largest Lyapunov exponent (LE) is always positive.

Proposition 3.1: The determinant of

$$M = \begin{bmatrix} A & B \\ C & D \end{bmatrix}$$

equals to $\det(M) = \det(A) \times \det(D - CA^{-1}B)$ where A and D are square matrices.

Proof: The proof can be found in [5].

Theorem 1: The mapping defined in (2) is area preserving.

Proof: Based on *Proposition 3.1*, the determinant of M can be expressed as:

$$\begin{aligned} \det(M) &= \det(I_n) \times \det((I_l + M_b M_a) - M_b I_n^{-1} M_a) \\ &= \det(I_n) \times \det(I_l + M_b M_a - M_b M_a) \\ &= \det(I_n) \times \det(I_l) \\ &= 1 \end{aligned}$$

Hence, the determinant of (3) is equal to 1 and the map is area preserving.

Theorem 2: The largest LE of (2) is positive.

Proof: It is shown that there exists a positive LE if $M_a \neq 0$, $M_b \neq 0$ and M is full rank. As M is full rank, there are m linear independent eigenvectors $p_1, p_2, \cdots, p_m$ corresponding to the eigenvalues $\lambda_1, \lambda_2, \cdots, \lambda_m$. There exist a modal matrix P and a spectral matrix D, such that $M = PDP^{-1}$, where

$$P = [p_1 \quad p_2 \quad \cdots \quad p_m] \text{ and } D = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \lambda_m \end{bmatrix}$$

As $\det|M| = 1$, the products of the eigenvalues $\prod_{i=1}^{m} \lambda_i = 1$. If $\lambda_1 = 1$ for all i, $M = PDP^{-1} = PP^{-1} = I$. Since $M_a \neq 0$ and $M_b \neq 0$, $M \neq I$, and hence contradiction. As a result, $D \neq I$, and there exists at least one $\lambda_j > 1$. Therefore, there is a positive LE $\sigma_j = \ln|\lambda_j| > 0$.

Example 1: let $M_a = \begin{bmatrix} 0.1 & 0.2 & 0.3 \\ 0.4 & 0.5 & 0.6 \end{bmatrix}$ and $M_b = \begin{bmatrix} 0.25 & 0.35 \\ 0.45 & 0.55 \\ 0.65 & 0.75 \end{bmatrix}$, from (3), one ob-

tains $M = \begin{bmatrix} 1 & 0 & 0.1 & 0.2 & 0.3 \\ 0 & 1 & 0.4 & 0.5 & 0.6 \\ 0.25 & 0.35 & 1.165 & 0.225 & 0.285 \\ 0.45 & 0.55 & 0.265 & 1.365 & 0.465 \\ 0.65 & 0.75 & 0.365 & 0.505 & 1.645 \end{bmatrix}$ and hence a five-dimensional Cat

map is obtained. The phase portrait of x_1 and x_2 is depicted in Fig. 1, and the largest LE is computed as 1.0356.
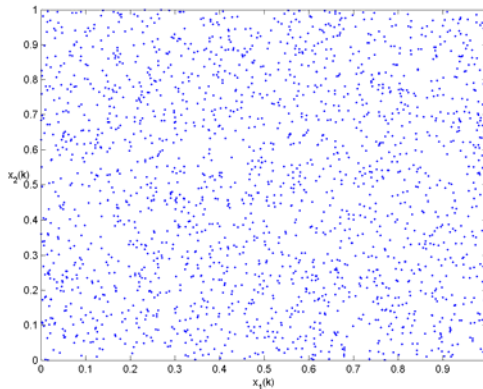

Fig. 1. The phase portrait of x_1 and x_2 of the 5-dimensional map obtained by multidimensional generalization

2.2 Spatial Extension

The main design concept of spatial extension is to apply the mapping in all the combinations of sub-spaces over the entire high-dimensional space. Consider an n-dimensional map defined by:

$$f_{d_1 d_2 \cdots d_n} : T \rightarrow T \tag{4}$$

where $T = S_{d_1} \times S_{d_2} \cdots \times S_{d_n}$, a m-dimensional map, g, with $m > n$ can then be constructed based on f,

$$g_{d_1 d_2 \cdots d_n} : S \rightarrow S \tag{5}$$

with $S = S_{d_1} \times S_{d_2} \times \cdots \times S_{d_n} \times \cdots \times S_{d_m}$, such that g performs similar mapping as f on the space T while the other dimensions kept unchanged. An m-dimensional map, $G_f : S \rightarrow S$, is hence defined as

$$G_f = g_{1,2,\cdots,n} \circ g_{1,2,\cdots,n-1,n+1} \circ \cdots \circ g_{1,2,\cdots,n-1,m} \circ g_{2,\cdots,n,n+1} \circ \cdots \circ g_{(m-n+1),\cdots,m} \tag{6}$$

It should be noticed that this presents a general method which is applicable for designing different kinds of high-dimensional maps. In addition, the nice properties in the corresponding low-dimensional maps can be maintained. For illustration, some examples are described below.

Example 2: In this example, a high-dimensional Cat map is derived by the spatial extension technique. Referring to the two-dimensional Cat map given in (1), an m-dimensional Cat map can then be obtained by:

$$x(k+1) = G_f(x(k)) = A_m x(k) \bmod 1 \tag{7}$$

where

$$A_m = \begin{bmatrix} 1 & a_{12} & 0 & \cdots & 0 \\ b_{12} & 1+a_{12}b_{12} & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & a_{13} & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ b_{13} & 0 & 1+a_{13}b_{13} & \cdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \cdots \begin{bmatrix} 1 & 0 & 0 & \cdots & a_{1m} \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 1 & 0 \\ b_{1m} & 0 & 0 & 0 & 1+a_{1m}b_{1m} \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & a_{23} & \cdots & 0 \\ 0 & b_{23} & 1+a_{23}b_{23} & \cdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \cdots \begin{bmatrix} 1 & \cdots & 0 & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & \cdots & 1 & 0 & 0 \\ 0 & \cdots & 0 & 1 & a_{m-1,m} \\ 0 & \cdots & 0 & b_{m-1,m} & 1+a_{m-1,m}b_{m-1,m} \end{bmatrix}. \tag{8}$$

A_m is a $m \times m$ matrix constructed by multiplying C_2^m matrices, $A_{12}A_{13}\cdots A_{1m}A_{23}\cdots A_{m-1,m}$ where $0 < a_{ij}, b_{ij} \leq 1$ and A_{ij} describes the transformation

*i*th- and *j*th-dimensions of the space. The following theorems prove that the map (7) is area-preservative and its largest LE is always positive.

Theorem 3: The map in (7) is area-preservative.

Proof: Consider the matrix A_{12} with $a_{ij} = a$ and $b_{ij} = b$, that is:

$$A_{12} = \begin{bmatrix} 1 & a & 0 & \cdots & 0 \\ b & 1+ab & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{9}$$

The determinant of A_{12} is computed as:

$$\det(A_{12}) = |A_{12}| = 1 + ab - ab = 1$$

by the use of cofactor expansion.

Notice that any matrix A_{ij} can be obtained by interchanging the rows and columns of the matrix A_{12}. For example, A_{23} can be obtained by swapping rows and columns of A_{12} for four times. It can be hence deduced that $\det(A_{ij}) = 1$ (According to the fact that interchange of any two rows and any two columns only changes the sign of the determinant for all $i, j \in [1, m]$). Therefore, the determinant of A_m is given as:

$$|A_m| = |A_{12}||A_{13}| \cdots |A_{m-1,m}| = 1$$

implying that the map (7) is area-preservative. It should also be emphasized that the determinant of A_m is independent of a_{ij} and b_{ij}.

Theorem 4: The largest LE of the map (7) is positive.

Proof: To determine the LE of the map, the eigenvalues of A_m are considered. Since the product of the eigenvalues of a matrix is equal to the determinant of the matrix, by assuming that the eigenvalues of A_m are λ_i for $i = 1, 2, \ldots, m$, we have

$$\prod_{i=1}^{m} \lambda_i = |A_m| = 1 \tag{10}$$

Based on the formulation of A_m, $\lambda_i = 1$ if and only if $a_{ij} = b_{ij} = 0$ or $A_m = I$. If there exist a_{ij}, $b_{ij} \neq 0$ for $i, j \in [1, m]$, there is at least one eigenvalue, say $\lambda_k > 1$. Otherwise, if $\lambda_i < 1$ for all i, $\prod_{i=1}^{m} \lambda_i < 1$ which contradicts (10). The largest LE σ_k of the map is then obtained by:

$$\sigma_k = \ln |\lambda_k| > 0 \tag{11}$$

For illustration, a 3-dimensional Cat map can be expressed as:

$$x(k+1) = G_f(x(k)) = A_3 \, x(k) \bmod 1 \qquad (12)$$

where

$$A_3 = \begin{bmatrix} 1 & a_{12} & 0 \\ b_{12} & 1+a_{12}b_{12} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & a_{13} \\ 0 & 1 & 0 \\ b_{13} & 0 & 1+a_{13}b_{13} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & a_{23} \\ 0 & b_{23} & 1+a_{23}b_{23} \end{bmatrix} \qquad (13)$$

Let $a_{ij} = b_{ij} = 1$ for all $i, j \in [1, 3]$, we have $|A_m|=1$ and the LEs are 1.9719, -1.4146, -0.5573. It should also be highlighted that a larger positive LE is obtained as compared with that of the two dimensional map. Moreover, no eigenvalue of (13) equals to ±1, implying that it is also a hyperbolic toral automorphism. Figure 2 shows the extension and contraction effects on different subspaces while Fig. 3 shows the results after iterating the map for k-times over a unit cubic.
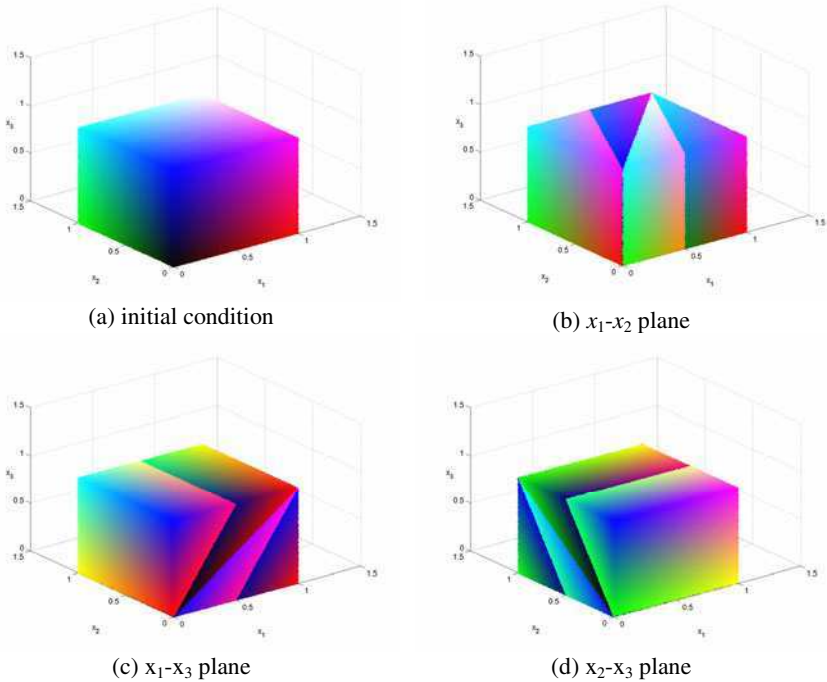

(a) initial condition   (b) x_1-x_2 plane

(c) x_1-x_3 plane   (d) x_2-x_3 plane

Fig. 2 Mapping of a 3-dimensional Cat map (12) on each plane

Example 3: The advantage of using spatial extension is that it can be applied to low-dimensional map without the expression of transformation matrix. This is illustrated by the following example with the generation of high-dimensional Baker map. Again, the basic concept is to apply the two-dimensional Baker map on the $m(m-1)/2$ subspaces. For example, a 3-dimensional Baker map can be considered as applying the Baker mapping onto each plane of a unit cube, as shown in Fig. 4. The evolution of the map is illustrated in Fig. 5 for reference.
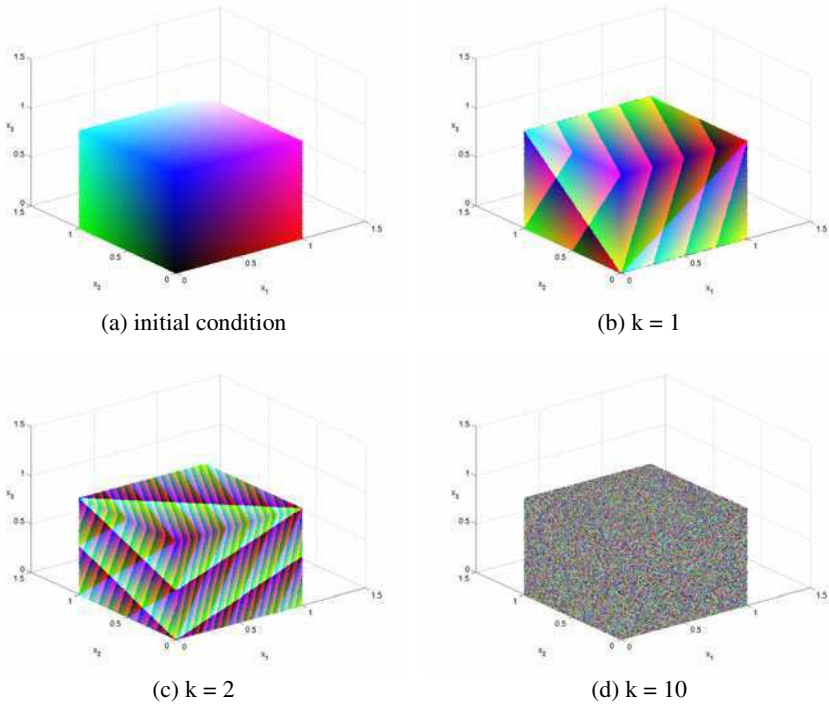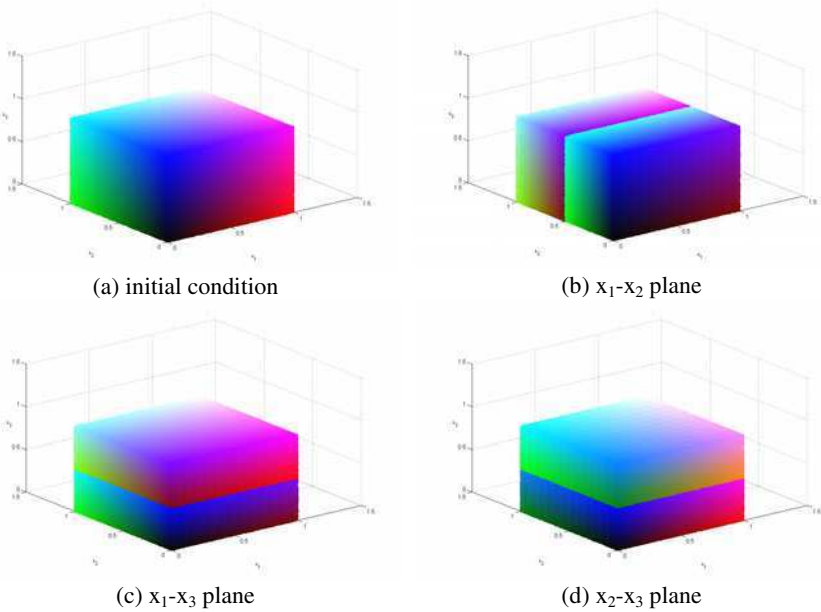
(a) initial condition

(b) k = 1

(c) k = 2

(d) k = 10

Fig. 3 Iterations of 3-dimensional Cat map

(a) initial condition

(b) x_1-x_2 plane

(c) x_1-x_3 plane

(d) x_2-x_3 plane

Fig. 4 Mapping of a 3-dimensional Baker map on each plane

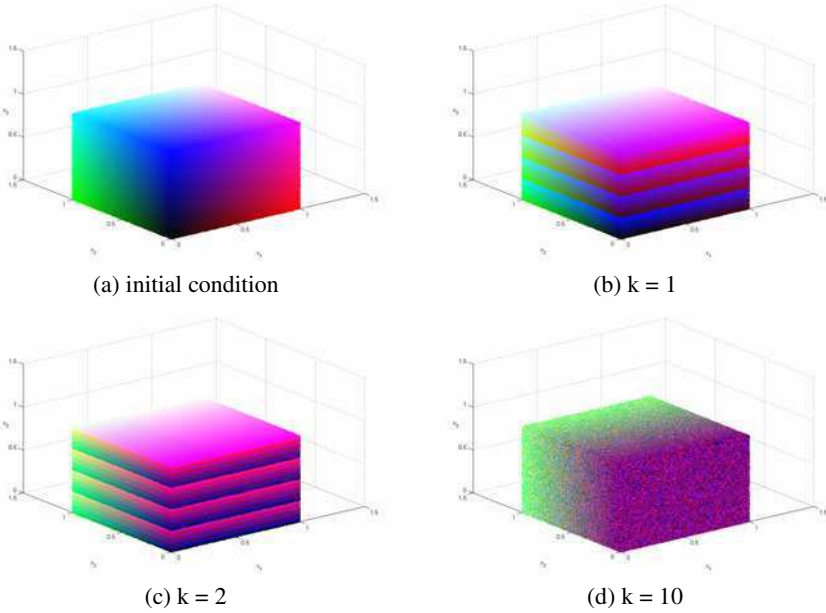(a) initial condition

(b) k = 1

(c) k = 2

(d) k = 10

Fig. 5 Iterations of a 3-dimensional Baker map

3 Properties of the High-Dimensional Chaotic Maps

Many of the interests in chaos research are motivated by its potential applications for which its distinct characteristics have crucial contributions. Although the high-dimensional maps generated by multidimensional generalization and spatial extension in Sect. 2 exhibit similar chaotic characteristics as their low-dimensional counterparts, however, due to the differences between their formations, the corresponding dynamical behaviours may not be the same, and they will be discussed below. Note that herein, Types I and II denote the high dimensional Cat map constructed by spatial extension and multidimensional generalization, respectively.

3.1 Recurrence Plot

A recurrence plot (RP) is a plot showing for a given moment at which a phase space trajectory visits roughly the same area in the phase space [4]. It is useful to visually check the periodic properties of the map and to reveal the correlations between data, serving as an evaluating measure of randomness.

Considering a sequence $\{ x_0, x_1 \cdots x_n \}$, a vector y_i of dimension $m \geq 2$ and delay $d \geq 1$ can be constructed by

$$y_i = (x_i, x_{i+d}, x_{i+2d}, \cdots, x_{i+(m-1)d}) \tag{14}$$

The RP is then obtained by plotting a point if the following condition is satisfied

$$\left\| y_j - y_i \right\| < t \tag{15}$$

where t is a predefined threshold distance.

The RPs of the proposed Type I and Type II 3D chaotic maps are shown in Fig. 6 with $t = 0.15$ and $d = 1$ and there are total 5000 data points each. No clear pattern is noticed in either type, indicating that consecutive samples are much far apart and uncorrelated. To further investigate the complexity of the RP, recurrence qualification analysis based on the toolbox provided in [40] can be used. Based on the data sets, there are 49 epochs with window size $L = 100$ and a data shifting of 100 points. The results show that the percent recurrences and the longest diagonal line segments in each epoch are all equal to zero for both sequences, meaning that the percent determinism and the entropy go to infinity. Therefore, these chaotic maps are of high complexities.
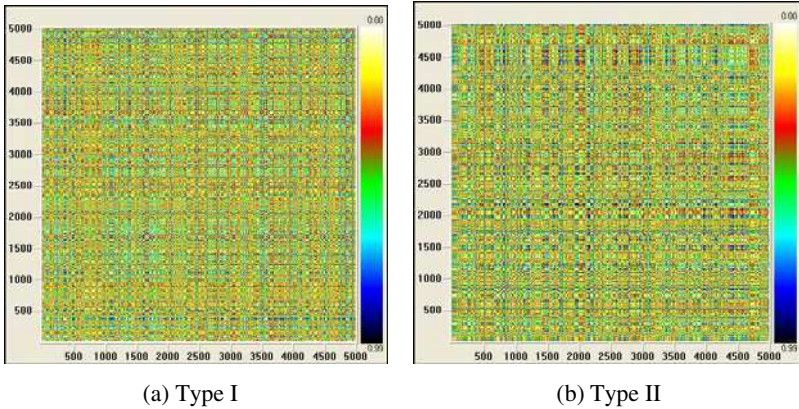

(a) Type I   (b) Type II

Fig. 6 Recurrence Plot with $t=0.15$ (a) 3D Cat map generated by spatial extension (b) 3D Cat map generated by multidimensional generalization.

For comparison, Fig. 7 depicts RPs obtained by some other chaotic maps including:

Type A: Chaotification of a discrete-time system [18]

Example: $x(k+1) = Ax(k) + bu(k)$ where $u(k) = \sum_{i=1}^{3} -\alpha_i \tanh(\beta_i x_i(k))$,

with $A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -0.91 & 0.78 & 0.78 \end{bmatrix}, b = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$, α=[-0.619 1.022 -0.462] and $\beta = 10 \times \mathbf{1}_{1\times3}$.

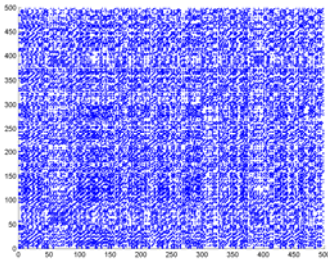**Type B:** Coupling of two chaotified discrete-time systems (six-dimension) [12]

$$\textit{Example:}\quad\begin{cases} x_1(k+1) = x_2(k) \\ x_2(k+1) = x_3(k) - \varepsilon_1(x_3(k) - x_6(k)) \\ x_3(k+1) = -a_1 x_1(k) - a_2 x_2(k) - a_3 x_3(k) + u_A(k) \\ x_4(k+1) = x_5(k) \\ x_5(k+1) = x_6(k) - \varepsilon_2(x_6(k) - x_3(k)) \\ x_6(k+1) = -a_4 x_4(k) - a_5 x_5(k) - a_6 x_6(k) + u_B(k) \end{cases}$$

where $u_A(k) = \sum_{i=1}^{3} -\alpha_i \tanh(\beta_i x_i(k))$, $u_B(k) = \sum_{i=4}^{6} -\alpha_i \tanh(\beta_i x_i(k))$
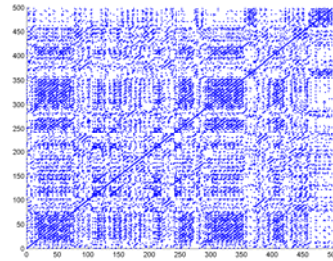
a = [0.91 -0.78 -0.78], b = [0.99 -0.84 -0.86], $\alpha_{1\text{-}3}$= [-0.619 1.022 -0.462], $\alpha_{4\text{-}6}$= [-0.627 1.028 -0.454], $\beta_{1\text{-}6}$ = 10, and ε_1, ε_2 =0.01.

Type C: Globally coupled map (GCM) based on logistic map [26]

Example: $x_i(k+1) = (1-\varepsilon)f(x_i(k)) + \dfrac{\varepsilon}{3}\sum_{j=1}^{3} f(x_j(k))$ for $i = 1, 2, 3$ and $\varepsilon = 0.01$

(a) Type A   (b) Type B
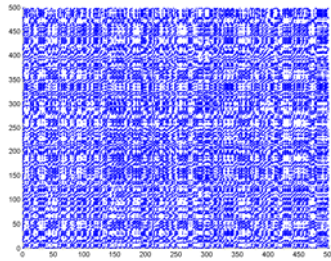
(c) Type C

Fig. 7 RPs generated by (a) chaotification of discrete-time system (b) coupling of two chaotic systems (c) GCM

From Fig. 7, some patterns are clearly observed. Moreover, Types A to C usually process lower LEs, as shown in the Table I, although they all possess chaotic properties such as sensitivity to the initial condition and exhibiting "random-like" dynamics.

Table 1 LEs of various high-dimensional chaotic maps

Chaotic maps	Calculation approach	Largest LE	Remarks
Type A, B	$\sigma_i = \lim\limits_{T\to\infty} \dfrac{1}{T} \sum\limits_{0}^{T-1} \ln\left|f'(x)\right|$	0.2067, 0.2200	Numerical calculations
Type C	$\sigma_1 = \sigma_0, \sigma_i = \ln(1-\varepsilon) + \sigma_0$	0.6931	Derived from logistic map [46]
Type I	$\sigma_i = \ln\left|\lambda_i\right|$	1.0356	Derived from eigenvalues
Type II	$\sigma_i = \ln\left|\lambda_i\right|$	1.9719	Derived from eigenvalues

3.2 Complexity Analysis

Entropy is a measure to evaluate the complexity or irregularity of a time series based on statistics. It quantifies an aspect of the time series under consideration in a robust and statistical manner. In the following study, two complexity measures, namely the sample entropy (SampEn) and the symbolic entropy (SyEn), are applied. The computation of these two entropies can be referred to [21,31] and references therein.

Figures 8 (a) and (b) depict the obtained SampEn and SyEn of Type I and II with different dimensions, respectively. Each result is the mean value based on 10 sets of time series, each with 10000 samples.

By comparing with the SampEn and SyEn for traditional 2D Cat map which are 0.5783 and 0.8228, respectively, the high-dimensional chaotic maps generated by the proposed methods possess higher complexity. It is also observed that both SampEn and SyEn demonstrate similar trend against the dimensions for both types of chaotic maps. Their complexities increase significantly with the system dimension, and finally reach maximums.
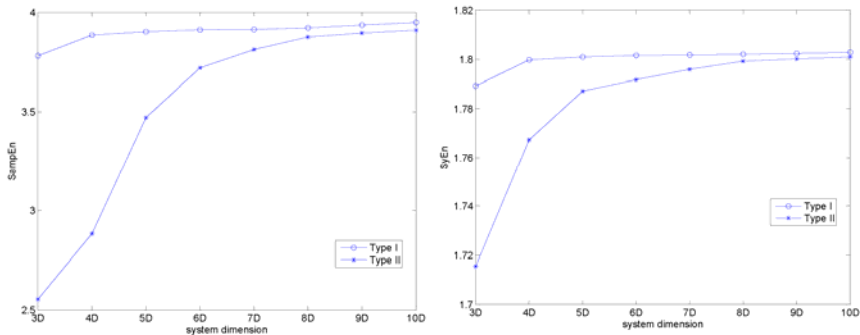

Fig. 8 (a) SampEn of different chaotic maps with m=2, r=0.01 (b) SyEn of different chaotic maps with m=8, L=5.

3.2 Mixing Nature

In addition to the above properties, another nice and important chaotic feature is its mixing nature. Here, it is verified in a statistical approach. Considering the

chaotic map as a black box and randomly choosing a set of initial conditions $x_1(0)$ (with $x_{2,3,\cdots,m}(0)$ fixed) which is normal distributed, it is possible to categorize the final values of state $x_1(k)$ after k iterations into b non-overlapping classes. If the frequency counts of these classes are uniformly distributed, the chaotic map under test is considered to possess nice mixing property.

For illustration, a set of initial conditions of x_1, denoted as S_1, is assumed as shown in Fig. 9(a). The data is randomly generated with mean 0.8 and standard deviation (SD) 0.001.

Let N=20000, b=100, after iterating a 5-dimensional Cat map (spatial one) with different number of iterations (k=1, 10, 20), the frequency count of each bin are depicted in Fig. 9(b)-(d). From the results, it is observed that, even though the initial states are limited within only a few bins, the final states tends to be uniformly distributed among all bins after some iterations, that is

$$\frac{\#\{x_1^{(n)}(k)\in Y_i, n=1,2,...,N\}}{N} \to \frac{1}{b} \tag{16}$$

The independence relationship between the initial and final states is further confirmed by comparing the results with several set of S_1 with different mean values, as shown in Fig. 10.

It should be remarked that the mixing nature of the high-dimensional Cat maps is much better than those obtained by other methods. To further confirm the
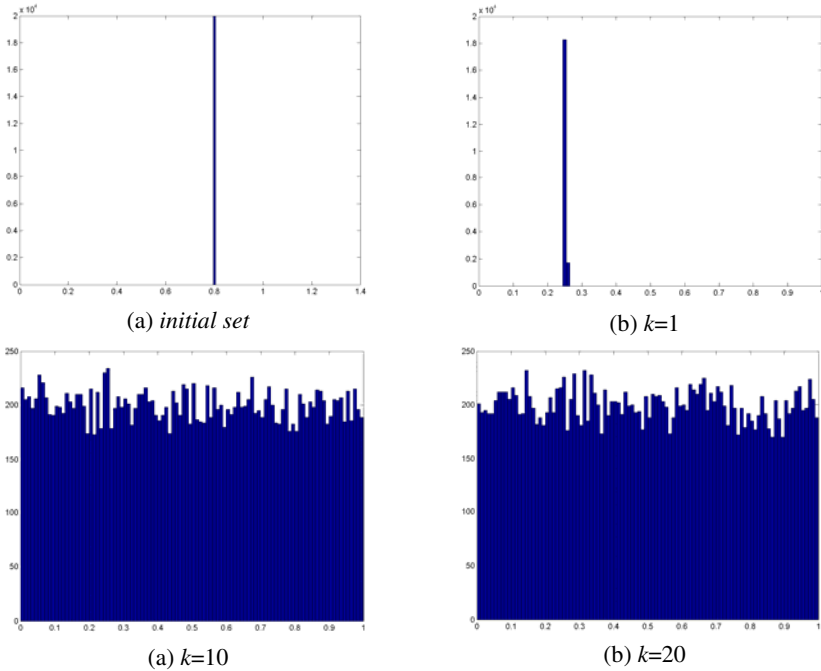

(a) *initial set*

(b) k=1

(a) k=10

(b) k=20

Fig. 9 Distribution of S2 after k iterations

mixing effect, the final histograms are testified by the chi-squared statistics with the hypothesis stated as below:

H0: The frequency count histogram after k iterations is uniformly distributed over the bins.

Consider

$$\chi_{test}^2 = \sum_{i=1}^{b} \frac{(o_i - e_i)^2}{e_i} \tag{17}$$

where b is the number of classes, o_i and e_i are the observed and expected occurrence frequencies in a particular class. For example, with a significance level of 0.05 and $b = 100$, $\chi_{99,0.05}^2$ is equal to 123.23. If $\chi_{test}^2 < \chi_{99,0.05}^2$, the null hypothesis is not rejected and hence the distribution of the histogram is assumed to be uniform.
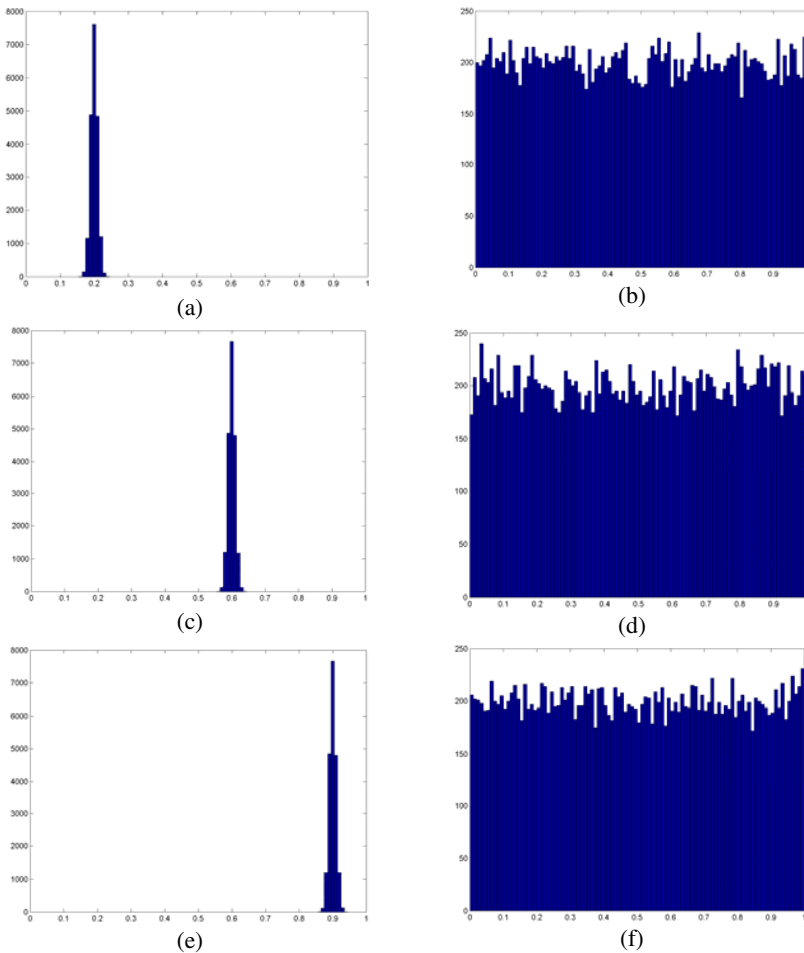

Fig. 10 (a), (c) and (e) are the initial condition sets with mean 0.2, 0.6 and 0.9, respectively; and (b), (d) and (f) are the corresponding states after ten iterations.

Tables 2 and 3 show the testing results with 100 sets of initial values for $b=100$ and $b=200$ $(\chi^2_{199,0.05} = 232.91)$, respectively.

Table 2 Mean of χ^2_{test} from 100 sets of initial values with $b=100$

Dimension	k=1	k=10	k=20
2	169840.72	50670.65	20738.67
3	179346.79	1944.62	213.90
4	183508.05	136.79	106.20
5	172629.90	107.16	109.77

Table 3 Mean of χ^2_{test} from 100 sets of initial values with $b=200$

Dimension	k=1	k=10	k=20
2	317202.71	60802.03	19007.91
3	305006.44	3121.64	395.48
4	308887.50	295.412	202.36
5	303624.27	202.09	204.67

As shown in Tables 2 and 3, $\chi^2_{test} < \chi^2_{\alpha,df}$ which is independent of the choice of b, when $k \geq 20$ and the dimension of map is greater than 3. The hypothesis is NOT rejected, or in other words, it is considered as uniform. With such uniformity, it is believed that the statistical relationship between the initial conditions and final values is broken by the transformation, and the proposed high-dimensional is considered to be mixing, with a finite number of iterations and particular dimensions.

Obviously, the mixing effect of the high-dimensional chaotic map is dependent on the dimensions of the map and the number of iterations. Such a relationship is revealed with a 3-dimensional view of test results given in Figs. 11 for $b=100$. For the ease of referencing, a horizontal plane indicating $\chi^2_{\alpha,df}$ is drawn. Any bar above this reference plane implies that the distribution of the histogram is non-uniform, i.e., the mixing effect is not good enough. Similar results can be obtained if different values of b are used.

Figure 12 shows a magnified version of Fig. 11. It clearly shows that for a certain dimension, a better mixing effect can be obtained by increasing the number of iterations. Similarly, when the number of iterations is fixed, the mixing nature depends on the dimension of the chaotic map.

Table 4 shows the appropriate dimensions and the corresponding numbers of iterations for which sufficient mixing effect can be achieved. For example, a 4-dimensional Cat map will need at least 11 iterations to achieve an acceptable mixing effect. While for a 5-dimensional Cat map, only 6 iterations are needed.
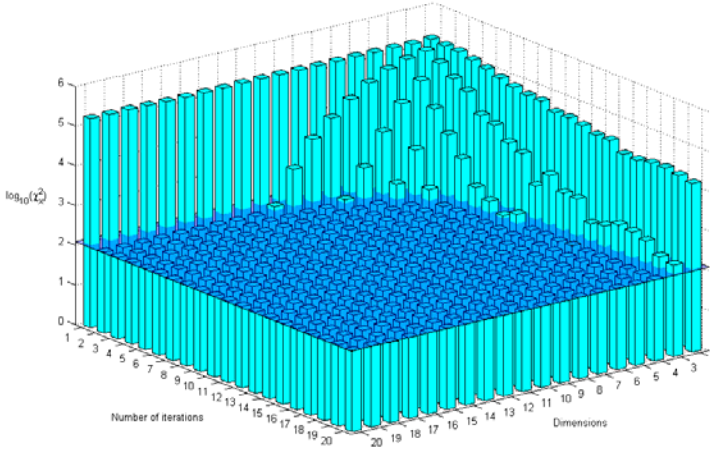
Fig. 11 The chi-square test results with b=100

To demonstrate the mixing effects, some illustrations are given in Fig. 14 and 15. Consider an arbitrary 5-dimesional Cat map, with initial values randomly selected in the domain of [0, 0.1], Fig. 13 depicts the final values after six iterations (i.e. k=6) for which an acceptable effect is demonstrated. Another example is also shown in Fig. 15 with m= 9 and k= 3, which is the most effective one in terms of computational cost.
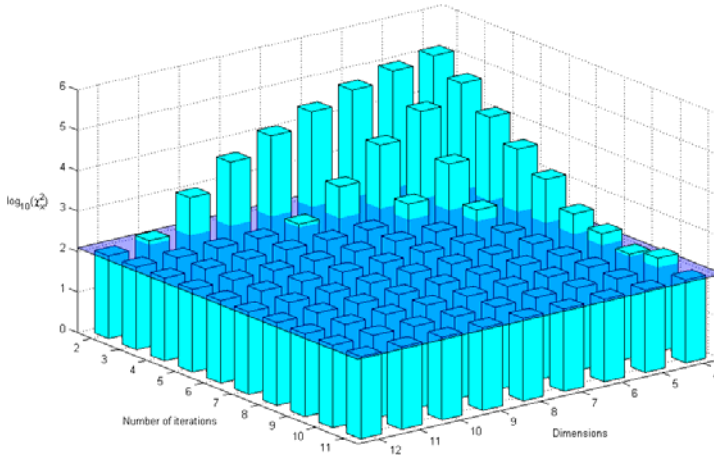
Fig. 12 The chi-square test result with b=100

Table 4 Parameters of the maps with sufficient mixing effect

Dimension (m)	No. of iterations (k)	χ_x^2 (b = 100)	χ_x^2 (b = 200)
4	11	110.62	206.33
5	6	115.43	209.98
6	5	112.49	206.74
7	4	113.61	204.63
8	4	109.25	201.30
9	3	109.75	205.95
10	3	109.48	201.92
11	3	107.49	203.25
12	2	110.12	202.08
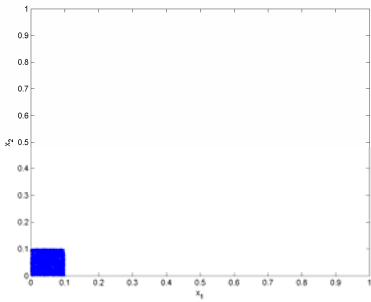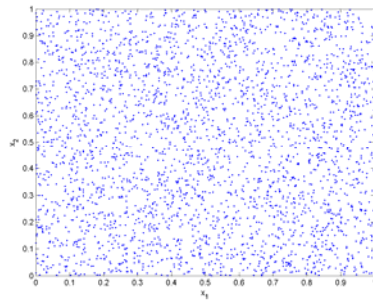
(a) initial states   (b) *K*=6

Fig. 13 Mixing effect of a five-dimensional Cat map
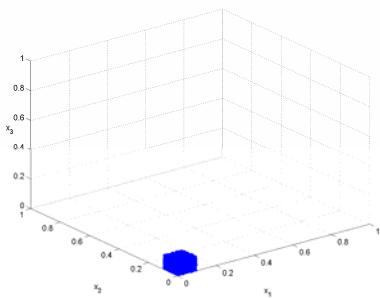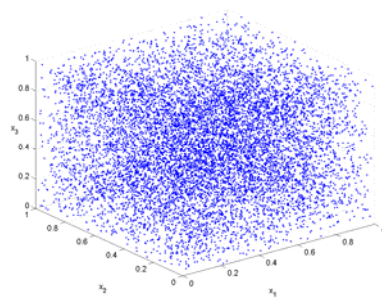
(a) initial states   (b) *K*=3

Fig. 14 Mixing effect of a nine-dimensional Cat map

For comparison, Tables 5 and 6 show the results of its counterpart, namely, high-dimensional Cat map constructed by multidimensional generalization and high-dimensional Baker map constructed by spatial extension. For consistency, the parameters of all the maps are randomly chosen between 0 and 1.

Table 5 Results of the statistical test of high-dimensional Cat map constructed by multidimensional generalization

| Dimension (m)	No. of iterations (k)	χ_x^2 (b = 100)	χ_x^2 (b = 200)
4	27	113.19	208.27
5	17	112.99	209.75
6	13	119.08	218.56
7	10	119.89	215.13
8	9	117.98	209.42
9	8	113.51	212.78
10	8	120.55	216.66
11	7	122.79	218.21
12	6	123.20	219.74

Table 6 Results of the statistical test of high-dimensional baker map constructed by spatial extension

Dimension (m)	No. of iterations K	χ_x^2 (b = 100)	χ_x^2 (b = 200)
4	10	109.55	203.78
5	7	105.51	201.89
6	7	107.40	202.12
7	5	110.19	201.48
8	3	109.38	204.70
9	3	109.64	204.39
10	2	106.31	202.71
11	2	111.44	204.32
12	2	106.75	202.57

By comparing the results in Tables 4 and 5, a larger number of iterations is noticed if high-dimensional Cat map constructed by multidimensional generalization is considered. This is mainly due to the formation of the transformation matrix. The inclusion of identity matrix in multidimensional generalization will result a weaker mixing effect as some dimensions may not be involved in the mapping. Therefore, spatial extension is more appropriate if a good mixing nature is necessary.

From Table 6, it is observed that the high-dimensional Baker map requires similar number of iterations to achieve a good mixing effect, as compared with the

high-dimensional Cat map. However, the implementation of high-dimensional Cat map is much easier than the high-dimensional baker map.

4 Cryptographic Applications

As illustrated in Sect. 3, the high-dimensional Cat map generated by spatial extension process better characteristics and can be easily implemented. Therefore, it is adopted in the design of the following cryptographic applications.

4.1 32-Bit Chaos-Based Random Number Generator

The first application is for the design of a PRNG and the block diagram of the proposed design is given in Fig. 15. It mainly consists of two parts: a chaotic skew tent map for the generation of bit sequence, and the high-dimensional Cat map for post-processing.
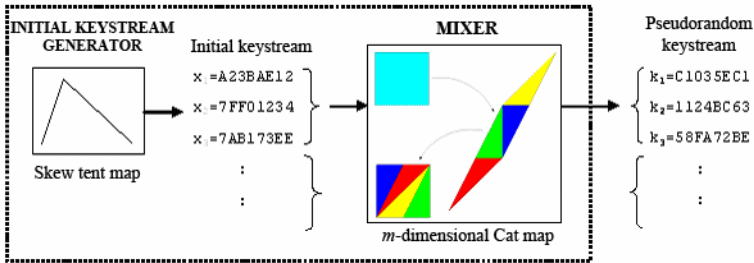
Fig. 15 A cascade structure for chaos- based PRNG

The skewed tent map is chosen as it is simple and fast in computation. Consider the below discrete tent map:

$$\hat{x}_{n+1} = \begin{cases} g\left(\dfrac{\hat{x}_n}{\hat{p}}\right) & \text{if} \quad 0 < \dfrac{\hat{x}_n}{2^L+1} \le \hat{p} \\ g\left(\dfrac{2^L+1-\hat{x}_n}{1-\hat{p}}\right) & \text{otherwise} \end{cases} \tag{18}$$

where $\hat{p} = (2k-1)/2(2^L+1)$ with $k = 1,2\cdots,2^{L-1},2^{L-1}+2,\cdots,2^L+1$ to ensure $\hat{p} \ne \dfrac{\hat{x}_n}{2^L+1}$ and $p \ne 0,0.5,1$, $\hat{x}_n \in \left\{1,2,\cdots,2^L\right\}$, $\hat{x}_0$ is the initial value and g(•) is the approximation function (rounding, flooring or ceiling). The bit sequence is generated by directly expressing the value of $\hat{x}_n$ in binary number of L bits:

$$\hat{x}_n = \{ b_1 b_2 \cdots b_j \cdots b_L \} \tag{19}$$

where $b_j \in \{0,1\}$ and $j = 1,2,\cdots,L$.

Since the bit sequence generated from the skewed tent map is unavoidably affected by the quantization process, some kinds of post-processing is required. The sequence is hence divided into blocks, each contains m's L-bit integers. The blocks are then transformed by the map (7) and the output is used as the random keystream. Based on the result shown in Table 5, a 5-dimensional Cat map is chosen. Some further improvements can also be made by adding a data feedback described as below:

$$\begin{bmatrix} z_{6+4(c-1)} \\ \vdots \\ z_{5+4c} \\ d_c \end{bmatrix} = A \begin{bmatrix} d_{c-1} \\ x_{6+4(c-1)} \\ \vdots \\ x_{5+4c} \end{bmatrix} \mod 2^L \tag{20}$$

where $A = A_{12}A_{13}\cdots A_{1s}A_{23}A_{24}\cdots A_{25}\cdots A_{45}$ is a 5×5 matrix constructed by multiplying $_5C_2$ matrices, each A_{ij} mixes the i-th and j-th dimensions as in the extended Cat Map and c is the block's identity number ($c = 1,2,\cdots$).

In order to testify the randomness of the keystream, the statistic test suite, designed by the National Institute of Standards and Technology (NIST) in Special Publication 800-22 [33], is used. It contains an extensive set of statistical tests, evaluating three major categories of random natures:

1. Random Walk Nature: frequency test (FT), block frequency test (BFT), cumulative sums test (CST), random excursions test (RET), random excursions variant test (REVT).
2. Pattern Checking: runs test (RT), longest run of ones test (LROT), nonoverlapping template matching test (NTMT), overlapping template matching test (OTMT), Maurer's universal statistical test (MUST), approximate entropy test (AET), serial test (ST).
3. Complexity and Compression: Marsaglia's rank test (MRT), spectral test (SPT), linear complexity test (LCT).

The acceptable percentage of passing a test is determined by:

$$P_a = (1-\alpha) \pm \sigma \sqrt{\frac{\alpha(1-\alpha)}{S}}$$

where $\alpha \in (0.001, 0.01]$, σ is the standard deviation and S is the sample size. In our experiments, 300 sequences ($S = 300$), each with 1,000,000-bit long, are tested. Together with the chosen standard parameters, $\alpha = 0.01$ and $\sigma = 3$, we have $P_a = 97.28\%$. If the successive percentage of any test is smaller than P_a, the sequences are considered to be not good enough.

Table 7 summarizes the passing rates of the sequences with and without post-processed by the high-dimensional Cat map. It can be observed that 10 out of 16 of the tests are failed if the sequence is tested directly. However, a noticeable improvement is observed after post-processing, and all the tests are passed.

Table 7 Passing rate of the random sequences generated by system in Fig .15

	Without post-processing	With post-processing					
Tests	tent bit sequence	VNC	XOR	SR	SHA-1	5DC	5DCFB
FT	96.67%	98.67%	98.67%	98.67%	98.67%	99.00%	99.67%
BFT	64.67%	83.67%	53.00%	99.00%	99.00%	99.33%	99.00%
CST(a)	95.00%	98.67%	98.67%	98.67%	98.33%	99.00%	99.33%
CST(b)	94.33%	98.67%	98.00%	98.67%	99.00%	99.00%	99.33%
RET	99.26%	98.81%	98.90%	98.41%	99.18%	98.57%	98.70%
REVT	99.01%	98.91%	99.72%	98.98%	98.82%	99.22%	99.22%
RT	97.33%	98.00%	99.00%	99.67%	99.00%	98.67%	98.67%
LROT	98.33%	97.33%	97.00%	98.67%	99.67%	98.67%	99.33%
NTMT	98.00%	97.35%	92.82%	98.98%	99.00%	99.14%	98.95%
OTMT	96.00%	98.00%	96.33%	98.33%	99.00%	98.00%	100.00%
MUST	87.33%	96.33%	96.67%	97.33%	99.00%	99.00%	99.33%
AET	97.67%	74.33%	33.67%	99.00%	98.67%	99.33%	98.67%
ST	98.17%	96.50%	86.50%	99.00%	98.67%	99.00%	99.17%
MRT	88.33%	98.67%	89.67%	99.00%	98.33%	99.00%	99.33%
SPT	61.00%	98.67%	89.67%	99.33%	98.33%	98.33%	99.33%
LCT	99.67%	99.00%	99.67%	98.67%	99.33%	98.00%	98.67%

VNC: Von Neumann corrector [15]; XOR: XOR based post-processing [30].
SR: shift-register based post-processing [30]; 5DC: 5-D Cat map (7).
5DCFB: 5-D Cat map with feedback (19).

The bit output rates in a 2.6GHz Pentium-4 machine with different post-processing function are compared in Table 8. It is noticed that a quite slow rate is obtained when Von Neumann corrector or XOR-based post-processing function is used because of the decimation rate of 1:4. The shift-register based post-processing is also slow when it is implemented in software since it is mainly a hardware design. Of course, the speed may be greatly improved if the whole design is implemented in hardware.

With a rather complex structure, SHA-1 has a moderate bit rate. The 5-D Cat map is the fastest bit rate because only 25 multiplication and 20 addition operations are needed for each 160 bits output. The speed reduces slightly when data feedback is introduced, but it is still very fast as compared with the others.

Table 8 Bit output rate of Tent bit sequence with different post-processing

	Tent's map bit sequence with post-processing					
	VNC	XOR	SR	SHA-1	5DC	5DCFB
Bit rate(Mbps)	13.92	34.02	21.43	59.97	141.51	135.02

4.2 8-Bit Chaos-Based Random Number Generator Design

A similar design as given in Fig. 15 is also applicable for low precision environment, except that the quantization error makes the usage of chaotic map as initial keystream become inappropriate. Taking 8-bit precision as an example, there exists only 256 states in such a finite machine and hence it is impossible to generate a fair enough bit sequence for usage and the short cycle length problem becomes very serious.

On the other hand, chaotic circuit can be adopted for bit generation. Figure 16 shows the block diagram of the design and the operational procedures are as follows. Firstly, bit stream is obtained by sampling a system state of a chaotic circuit, and then after post-processing, a random bit sequence can be obtained [36].
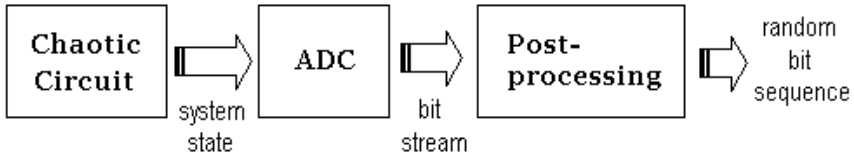
Fig. 16 Block diagram of chaos-based RNG design

In this example, a very simple chaotic circuit proposed in [10] is adopted. It is a modified Chua's circuit, as expressed below:

$$
\begin{cases}
C_1 \dfrac{dV_1}{dt} = \dfrac{1}{R}(V_2 - V_1) - f(V_1) \\[2mm]
C_2 \dfrac{dV_2}{dt} = \dfrac{1}{R}(V_1 - V_2) + I_3 \\[2mm]
L \dfrac{dI_3}{dt} = -V_2 - I_3 R_0
\end{cases}
\tag{21}
$$

where $f(V) = G_a V - \dfrac{G_a}{V_{max}^2} V^3$ denotes the cubic-like voltage-current characteristic

of the nonlinear resistor with slopes G_a and $\pm V_{max}$ are the outer zeroes.

The electronic circuitry proposed in [10] is redrawn in Fig. 17 where the shaded part is to realize $f(V)$. The double-scroll chaotic attractor can be obtained as shown in Fig. 18.
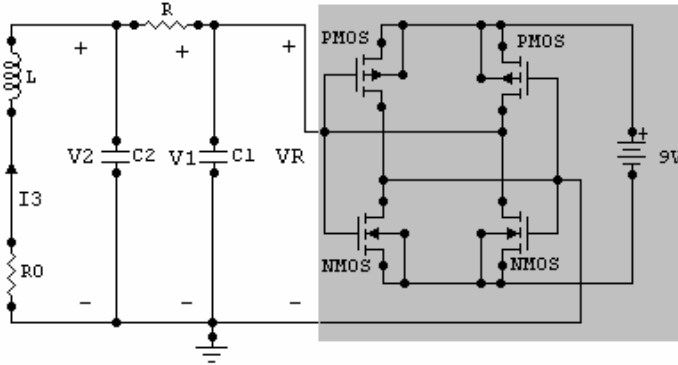
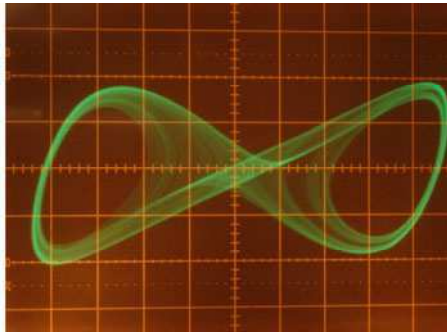**Fig. 17** Realization of modified Chua's circuit in electronic circuit

Fig. 18 Double scroll attractor (horizontal axis: 1V/dic; vertical axis: 0.5V/div)

The use of this circuit has the following advantages:

1. The circuit is extremely simple and only consists of a few passive compo-
   nents and two pairs of dual complementary MOS transistors.
2. A chaotic attractor is observed, and hence non-periodic bit sequence can be
   obtained after data sampling.
3. Based on the work of [8] for the implementation of Chua's circuit in inte-
   grated circuit, it is possible to realize this modified circuit using the similar
   technology. Moreover, the cubic function implemented in MOS transistors
   can also be easily included.

In our design, the state of V_2 in (21) is sampled by analog-to-digital converter
(MAX118) with a sampling rate of 50KHz and used as the bit sequence. However,
even though the sampled data is non-periodic, the data sequence is still biased and
not good enough as a PRNG due to the existence of fundamental frequency in the
oscillator. This is confirmed with the non-uniform frequency spectrum as shown
in Fig. 19.

Therefore, the sampled data is postprocessing by a 5-dimensional Cat map (19).
The block diagram of the entire system and a prototype are shown in Figs 20 (a)

and (b), respectively. It should be remarked that the software design for postprocessing is light-weighted. The program size is just 266 bytes, including the data control and communication via RS232. In fact, the post-processing function only occupies 126 bytes of program memory and 35 bytes internal RAM.
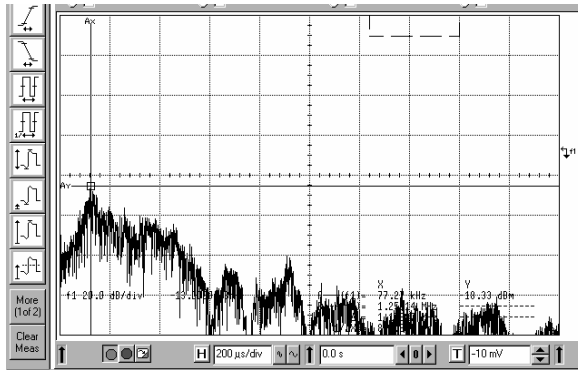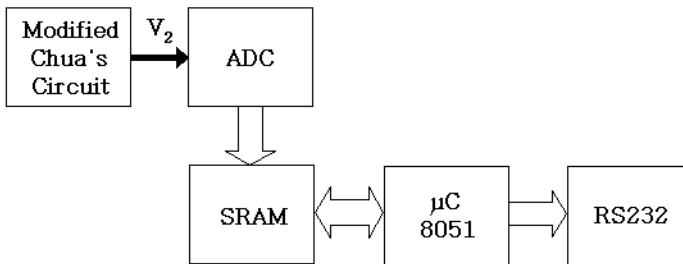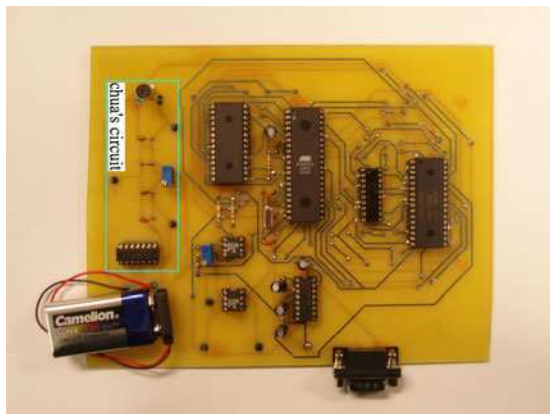

Fig. 19 FFT of Sampled data of V_2

(a)

(b)

Fig. 20 (a) Block diagram (b) prototype of RNG with 8-bit controller

Again, to testify the statistical performance of the output, NIST tests are carried out and the results are tabulated in Table 10. A significant improvement is noticed after post-processing and all the tests are now passed.

Table 10. Statistical performance of bit sequences obtained by system in Fig. 20

Tests	Before post-processing	After post-processing
FT	0.00%	99.00%
BFT	0.00%	98.33%
CST(a)	0.00%	99.00%
CST(b)	0.00%	99.00%
RET	0.00%	99.05%
REVT	0.00%	98.79%
RT	0.00%	99.33%
LROT	0.00%	99.33%
NTMT	12.70%	98.91%
OTMT	0.00%	99.33%
MUST	0.00%	99.00%
AET	0.00%	99.00%
ST	0.00%	99.67%
MRT	94.00%	97.67%
SPT	1.00%	99.00%
LCT	99.33%	99.33%

4.3   Chaos-Based Image Encryption

Based on the 32-bit chaos-based PRNG given in Sect. 4.1, a fast image encryption algorithm in the stream cipher architecture [20] can be designed as shown in Fig. 21. As an image usually consists of a large number of bytes, for example, an image in 24-bit true color with 512×512 pixels has a size of about 786 Kbytes, a fast encryption scheme and hence a fast keystream generator are demanded. Therefore, finite precision representation and fixed point arithmetic are adopted. This kind of implementation will also favor for the future hardware development of the algorithm.

Since the chaos-based PRNG is governed by a skew tent map and a m-dimensional Cat map, the output sequence is depended on the values of p, x_0, A_m (i.e. $\{a_{ij}\}$ and $\{b_{ij}\}$). These values can be derived from the user key by a chaos-based hash function, constructed by another high-dimensional Cat map. Through iterations, a hash value, which is highly dependent on the input (i.e. the user key), is obtained and used to determine the system parameters needed in the PRNG. This hash function will produce an avalanche change in the obtained hash value and the process increases the complexity of deducing the key from the pseudorandom keystream.

1. Construct the parameters needed for PRNG with a 192-bit user key.
2. Transform the image into a 32-bit data stream (refer to Fig. 21). Taking an image in 24-bit true color with 512×512 pixels as an example, if each pixel consists of 24-bit in RGB format, a stream of total 196,608 data (each data is in 32 bits and denoted as d_i) is formed by cascading the bits obtained from pixels.
3. Mask each data d_i by the keystream with the following encryption function:

$$c_i = (d_i + k_i + c_{i-1}) \bmod 2^{32} \tag{22}$$

where k_i is the output from PRNG and c_i is the encrypted data output with c_0 =0. The inclusion of c_{i-1} further changes the encrypted keystream based on previous data. The function (21) may be repeated by continuing k_i to increase the complexity of confusion after all pixels in the plain image are processed. In our design, $k_i=2$ (this will be justified later in Sect. 4.3.1-A)
4. Convert the encrypted data stream back into RGB format for storage or transmission. At this moment, the original image is encrypted to a cipher-image.

The procedures of decryption are similar to those of encryption, where the major difference is to replace (22) by the following function:

$$\hat{d}_i = (c_i - k_i - \hat{d}_{i-1}) \bmod 2^{32} \tag{23}$$

where $\hat{d}_i$ is the decrypted sequence. By iterating with the same number of times as in encryption, the original image should be obtained correctly ($\hat{d}_i = d_i$) with the same k_i.
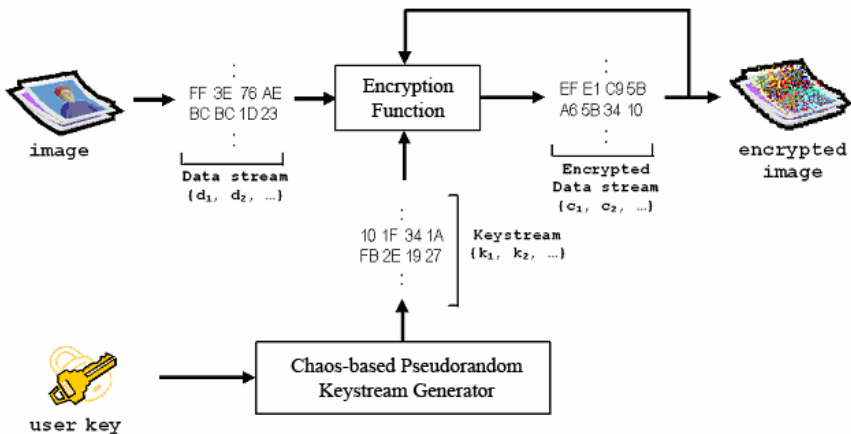

Fig. 21 Chaos-Based Image Encryption Scheme

4.3.1 Performance Analysis

A good encryption scheme should resist all kinds of known attacks. In this section, some security analyses are performance where most of the tests follow the suggestions given in [6,27].

A. Cipher cycle

As a general requirement for all the image encryption schemes, the encrypted image should be greatly different from its original form. To quantify this requirement, two measures, including the number of pixel change rate (NPCR) and unified average changing intensity (UACI) [6,27], are adopted.

The NPCR$_{R,G,B}$ is used to measure the number of pixels in difference of a color component between two images. Let $C(i, j)$ and $C'(i, j)$ be the i-th row and j-th column pixel of two images C and C', respectively, the NPCR$_{R,G,B}$ can be defined as:

$$\text{NPCR}_{R,G,B} = \frac{\sum_{i,j} D_{R,G,B}(i, j)}{N} \times 100\% \tag{24}$$

where N is the total number of pixels in the image and $D_{R,G,B}(i, j)$ is defined as

$$D_{R,G,B}(i, j) = \begin{cases} 0 & C_{R,G,B}(i, j) = C'_{R,G,B}(i, j) \\ 1 & C_{R,G,B}(i, j) \neq C'_{R,G,B}(i, j) \end{cases} \tag{25}$$

with $C_{R,G,B}(i, j)$ and $C'_{R,G,B}(i, j)$ representing the values of the corresponding color component Red (R), Green (G) or Blue (B) in the two images, respectively.

Considering two random images, the expected value of NPCR$_{R,G,B}$ is found to be:

$$\mathcal{E}\left[\text{NPCR}_{R,G,B}\right] = (1 - 2^{-L_{R,G,B}}) \times 100\% \tag{26}$$

where $L_{R,G,B}$ is the number of bits used to represent the color component of Red, Green or Blue. For example, for two random images with 512×512 pixels and 24-bit true color (8 bit for each RGB color component, and hence $L_R = L_G = L_B = 8$), $\mathcal{E}[\text{NPCR}_R] = \mathcal{E}[\text{NPCR}_G] = \mathcal{E}[\text{NPCR}_B] = 99.609375\%$.

Another quantity, UACI$_{R,G,B}$, is to measure the average intensity differences in a color component and can be defined as:

$$\text{UACI}_{R,G,B} = \frac{1}{N}\left(\sum_{i,j} \frac{\left|C_{R,G,B}(i, j) - C'_{R,G,B}(i, j)\right|}{2^{L_{R,G,B}} - 1}\right) \times 100\% \tag{27}$$

In the case of two random images, the expected value of $UACI_{R,G,B}$ can be computed by:

$$\mathcal{E}\left[UACI_{R,G,B}\right]=\frac{\dfrac{1}{2^{2L_{R,G,B}}}\left(\displaystyle\sum_{i=1}^{2^{L_{R,G,B}}-1}i\left(i+1\right)\right)}{2^{L_{R,G,B}}-1}\times100\% \tag{28}$$

and $\mathcal{E}\left[UACI_{R}\right]=\mathcal{E}\left[UACI_{G}\right]=\mathcal{E}\left[UACI_{B}\right]=33.46354\%$, assuming each color component is coded with 8 bits.

The $NPCR_{R,G,B}$ and $UACI_{R,G,B}$ of two cipher images, from two images with one bit difference in the corresponding color component using the same password, are obtained. Their typical values are plotted against the number of iterations, as shown in Figs. 22(a) and (b), respectively. It can be observed that a relatively good result can be obtained in all three color components if the number of cipher rounds is two or larger. Therefore, in the following experiments, two cipher cycles are adopted in order to obtain the fastest encryption scheme.

(a)   (b)
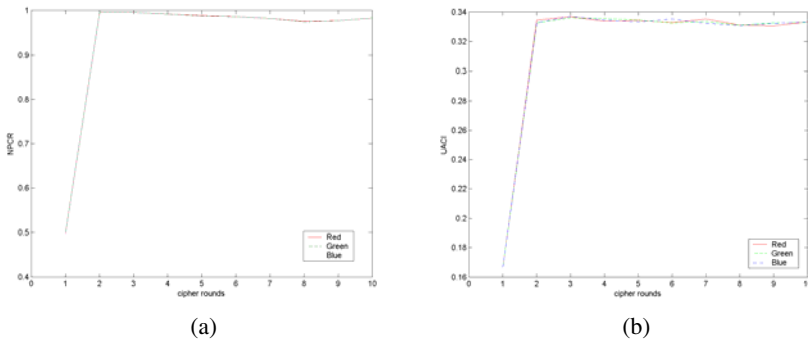
Fig. 22 (a) NPCR and (b) UACI for R,G, B color components

B. Visual Testing

Firstly, we demonstrate the performance of the algorithm with a visual test. Figure 23 depicts three examples, where each image is in 24-bit color with 512×512 pixels. By comparing the original and the encrypted images in Fig. 22, no visual information can be observed in the encrypted image. Moreover, the encrypted images are visually indistinguishable even with a big difference in the color tone found in the original images.

(a) Original image (Peppers)
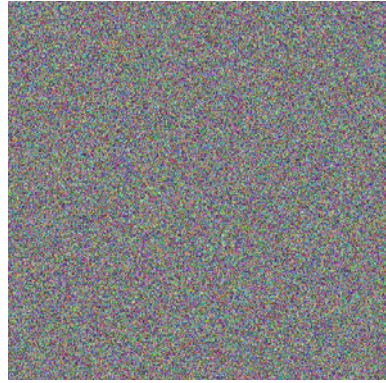
(b) Encrypted image (Peppers)

(c) Original image (Sailboat)
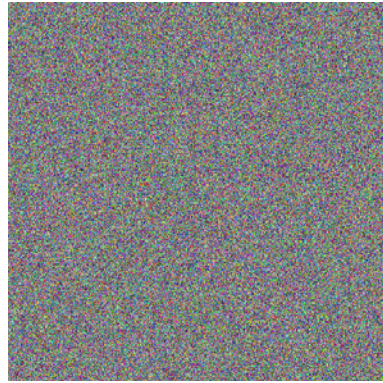
(d) Encrypted image (Sailboat)
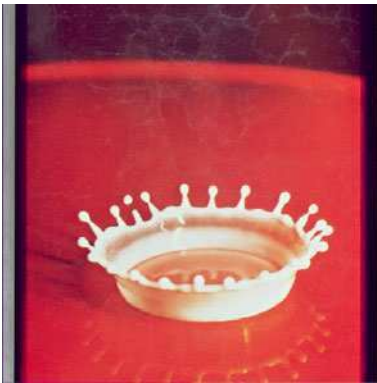
(e) Original image (Splash)

(f) Encrypted image (Splash)

Fig. 23 Examples of the original and the encrypted images

C. Statistical Analysis

In order to resist the statistical attacks, which are quite common, the encrypted images should possess certain random properties. Some more sophisticated tests suggested in [6,27] are then carried out.

Firstly, the color histograms of the encrypted image are checked. To appear random, the histograms should be uniform distributed in all three color components (RGB). Figures 24 and 25 depict the histograms for Peppers and its encrypted image (Similar histograms are obtained for other encrypted images). A flat color histogram is resulted from the encrypted image based on the proposed encryption scheme in all the cases. Its uniformity is further justified by the Chi-squared test similar to the one described in Sect. 3.2. Consider the number of levels is 256 in each color component, it is found that $\chi^2_{test} < \chi^2_{255,0.05} = 293.25$ for a significance level of 0.05. The testing result is given in Table 11.

(a)   (b)

(c)
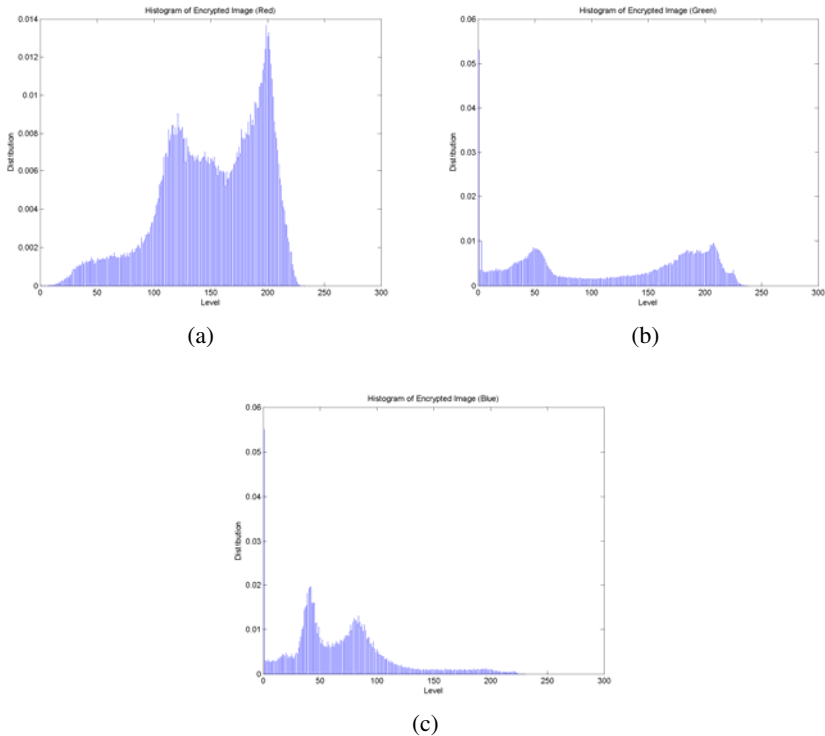
Fig. 24 Color histograms of the original image Peppers (a) Red (b) Green (c) Blue

(a)   (b)

(c)
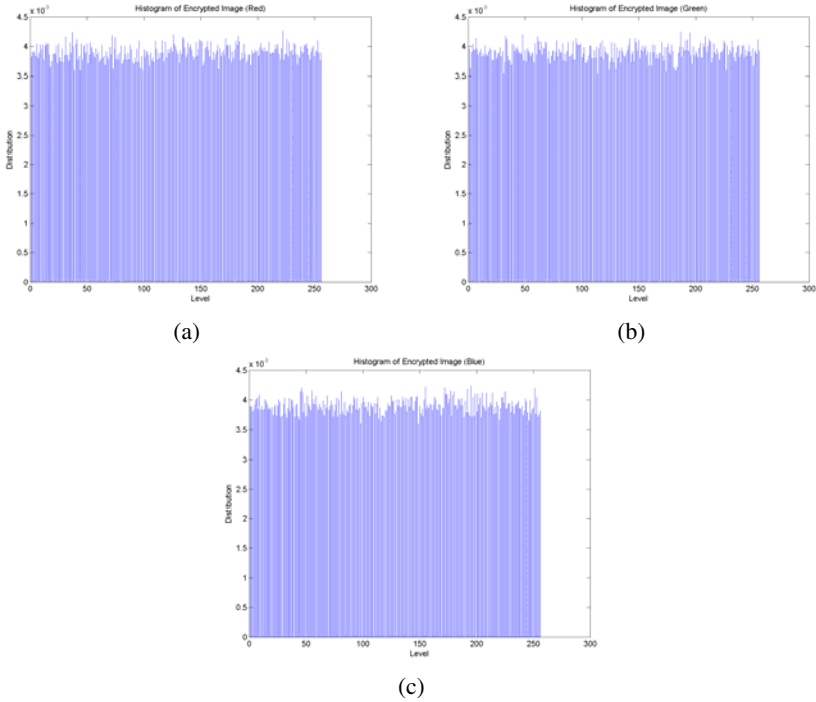
Fig. 25 Color histograms of the encrypted image Peppers (a) Red (b) Green (c) Blue

Table 11 Chi-squared testing results of color histograms of encrypted images

Color	Encrypted Images		
	Peppers	Sailboat	Splash
Red	261.79	218.93	242.85
Green	250.87	266.01	254.53
Blue	234.13	244.81	251.57

For an ordinary image, pixel is usually highly correlated with its adjacent pixels either in horizontal, vertical or diagonal directions. These high-correlation properties can be quantified as the correlation coefficient for comparison. Taking the horizontal correlation as an example, for each pixel of the image, a duplet (p_i, q_i), can be found, where q_i is the horizontal adjacent pixel of p_i. Obviously, there may be more than one duplet for each pixel, and the horizontal correlation coefficient [6] is computed as:

$$r = \frac{\text{cov}(p,q)}{\sqrt{D(p)}\sqrt{D(q)}} \tag{29}$$

where $D(p) = \dfrac{1}{M} \sum\limits_{i=1}^{M} (p_i - \overline{p})^2$, $\mathrm{cov}(p,q) = \dfrac{1}{M} \sum\limits_{i=1}^{M} (p_i - \overline{p})(q_i - \overline{q})$, M is the total

number of duplets (p_i, q_i) obtained from the image, and $\overline{p}$ and $\overline{q}$ are the mean values of p_i and q_i, respectively. For an image with $r \times s$ pixels (r and s indicate the number of pixels in horizontal and vertical directions, respectively), $M_d = 2r(s-1)$. By the similar approach, the vertical and diagonal correlation coefficients of an image can be defined.

Table 12 shows the three correlation coefficients of the encrypted images. It can be observed that the encrypted image obtained from the proposed scheme retains small correlation coefficients in all directions. The results are also compared with a random image (the value of each pixel is randomly generated by the rand() function in C) and similar values are noted.

Table 12 Correlation coefficients of adjacent pixels of encrypted images

Correlation Coefficient	Random Image	Peppers	Sailboat	Splash
Horizontal	0.0006326	0.0006604	0.0006860	0.0006644
Vertical	0.0006110	0.0006141	0.0006620	0.0006768
Diagonal	0.0019646	0.0019644	0.0018721	0.0020112

In order to avoid the known-plaintext attack and the chosen-plaintext attack, the changes in the cipher image should be significant even with a tiny change in the original one. According to the proposed encryption process, this small difference should be diffused to the whole ciphered data. They can in fact be reflected by the $\mathrm{NPCR}_{R,G,B}$ as shown in Fig. 22 and hence it is omitted here.

D. Security Key Analysis

The user key is in 192 bits and the key space is about 6.277×10^{57}, which is large enough to resist the brute-force attack with the current computer technology. Moreover, the key space can easily be expanded in order to satisfy the future requirement with the use of higher dimensional map.

Table 13 Pixel difference between images encrypted by keys with 1-bit difference

Image	Pixel difference (mean NPCR)
Peppers	99.60352%
Sailboat	99.60772%
Splash	99.60645%

To have a good encryption scheme, it should also be key-sensitive, meaning that a tiny change in the key will cause a significant change in the encrypted output. The average pixel differences of the encrypted images over 100 random keys

(for each key, all the cases with one-bit difference are computed) are tabulated in Table 13. It can be observed that the values are very close to the expected value of pixel difference on two randomly generated images (99.609375%). Figure 26 illustrate an example of two encryped images generated from two security keys with only 1-bit difference.

(a)   (b)

(c)   (d)
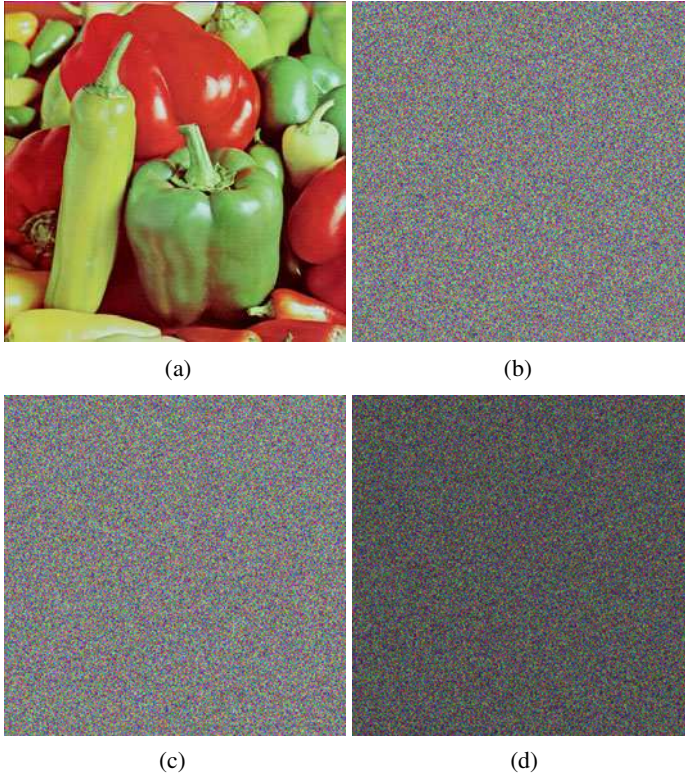
Fig. 26 Difference between encrypted images (Peppers) (a) Original image (b) and (c) Encrypted images with user key with 1-bit difference (d) difference between (b) and (c)

E. Speed Performance

The speed of the proposed encryption algorithm is much faster than those existing algorithms. Its average speed is about 40 MB/s with a Pentium IV 2.8 GHz personal computer. Table 14 shows the result of the average encryption speed as compared with some well-known encryption algorithms in Crypto++ Library [7], running on the same machine. It is also compared with Chen's method [6], which is originally developed in the MATLAB platform. On the same MATLAB platform, the speed of Chen's method is about 1.816 MB/s while the speed of the proposed system is up to 13.36 MB/s. With such a speed, this image encryption scheme can be used in Internet applications over broadband network, where the encryption and decryption time should be short relative to the transmission time.

Table 14 Encryption speed of the chaos-based encryption scheme and other well-known algorithms

Algorithm	Speed (MB/s)
Chaos-based encryption scheme	40.029
DES	6.483
AES (192-bit key)	11.645
AES (256-bit key)	11.380

5 Conclusions

In this chapter, two different methods, namely multidimensional generalization and spatial extension, have been introduced for the formation of high-dimensional chaotic maps. Based on some existing low-dimensional chaotic maps, it is demonstrated that chaotic maps with any dimensions can be obtained. The useful features, such as area-preservation and possession of positive Lyapunov exponents, are also inherited and also confirmed with mathematical proofs.

The studies of those generated high-dimensional chaotic maps are continued by investigating their chaotic characteristics. Yet, all of chaotic maps possess the typical chaotic nature, such as sensitive dependence of parameters and initial conditions and possessing at least one positive Lyapunov exponents, it is noticed that the newly generated maps possess more complex dynamics and better mixing nature.

The nice mixing characteristics of these high-dimensional chaotic maps have granted it to be an effective post-processing function. This in turns provides a major element for the design of the 32-bit chaos-based random number generator. By generating the bit sequence with a discretized tent map, and after post-processing, a very nice random number sequence is obtained. Its randomness is verified with different statistical criteria and also the stringent NIST test suite which commonly used to verify the random number sequences used in cryptographical applications. Similar structure can also be used to meet another challenge of designing a random number generator for very low-precision system. By replacing the discretized tent map with a simple chaotic circuit, a nice random number can still be obtainable. The experimental results show that a very nice random number generator can be realized and the generated bit sequences fulfill all the statistical requirements, including the tests in NIST.

Finally, an image encryption scheme is developed. A stream cipher is designed by cascading a one-dimensional map for the initial sequence generation and a high-dimensional Cat map for post-processing. Thorough analyses have been carried out and a fast and effective chaos-based image encryption scheme is confirmed.

Acknowledgments

References

[1] Arnold, V.I., Aver, A.: Ergodic problems of classical mechanics. W. A. Benjamin, New York (1968)

[2] Atay, F.M., Jost, J.: Delays, connection topology, and synchronization of coupled chaotic maps. Phys. Rev. Lett. 92(4), 144101 (2004)

[3] Billings, L., Bollt, E.M.: Probability density functions of some skew tent maps. Chaos, Solitons and Fractals 12, 365–376 (2001)

[4] Bourke, P.: Recurrence Plots (1998),
http://local.wasp.uwa.edu.au/pbourke/fractals/recurrence/

[5] Brookes, M.: Matrix Reference Manual (2005),
http://www.ee.ic.ac.uk/hp/staff/dmb/matrix/intro.html

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

[7] Crypto++ Library, http://www.cryptopp.com

[8] Cruz, J.M., Chua, L.O.: An IC chip of Chua's circuit. IEEE Trans.Circuits and Systems II: Analog and Digital Processing 40(10), 614–625 (1993)

[9] Dai, S., Wu, Q., Pei, W., Yang, L., He, Z.: Lissajous chaotic map. In: Int. Conf. on Neural Networks and Signal Processing, vol. 1, pp. 772–775 (2003)

[10] Donoghue, K.O., Kennedy, M.P., Forbes, P., Wu, M., Jones, S.: A fast and simple implementation of Chua's oscillator with cubic-like nonlinearity. Int. J. Bifurcation and Chaos 15(9), 2959–2971 (2005)

[11] Falcioni, M., Palatella, L., Pigolotti, S., Vulpiani, A.: Properties making a chaotic system a good pseudo random number generator. Phys. Rev. E 72, 16220 (2005)

[12] Grassi, G., Mascolo, S.: A systematic procedure for synchronizing hyperchaos via observer design. J. of Circuits, Systems and Computers 11(1), 1–16 (2002)

[13] Hénon, M.: A two-dimensional mapping with a strange attractor. Communications Math. Phys. 50, 69–77 (1976)

[14] Holmgren, R.A.: A first course in discrete dynamical systems. Springer, New York (1996)

[15] Jun, B., Kocher, P.: The Intel random number generator. Cryptography Research, White Paper Prepared for Intel Corporation (April 1999),
http://www.cryptography.com/resources/whitepapers/
IntelRNG.pdf

[16] Kaneko, K.: Theory and applications of coupled map lattices. John Wiley & Sons, Chichester (1993)

[17] Kohda, T., Tsuneda, A.: Statistics of chaotic binary sequences. IEEE Trans. Information Theory 43, 104–112 (1997)

[18] Kwok, H.S., Tang, K.S.: Chaotification of discrete-time systems using neurons. Int. J. of Bifurcation and Chaos 14(4), 1405–1411 (2004)

[19] Kwok, H.S., Tang, K.S.: Applications of high-dimensional cat map and its mixing nature. In: Int. Symp. Nonlinear Theory and its Applications, pp. 687–690 (2006)

[20] Kwok, H.S., Tang, K.S.: A fast image encryption scheme based on high-dimensional chaotic map. Chaos, Solitons and Fractals 32(4), 1518–1529 (2007)

[21] Li, S., Li, Q., Li, W., Mou, X., Cai, Y.: Statistical properties of digital piecewise linear chaotic maps and their roles in cryptography and pseudo-random coding. In: Honary, B. (ed.) Cryptography and Coding 2001. LNCS, vol. 2260, pp. 205–221. Springer, Heidelberg (2001)

[22] Li, S., Zheng, X.: Cryptanalysis of a Chaotic Image Encryption Method. In: IEEE Int. Symp. Circuits and Systems, vol. 2, pp. 708–711 (2002)

[23] Lian, S., Mao, Y., Wang, Z.: 3D extensions of some 2D chaotic maps and their usage in data encryption. In: The Fourth Int. Conf. on Control and Automation, pp. 819–823 (2003)

[24] Liu, Y., Sun, L., Zhu, Y., Beadle, P.: Novel method for measuring the complexity of schizophrenic EEG based on symbolic entropy analysis. IEEE-EMBS 1, 37–40 (2005)

[25] Lorenz, E.N.: The essence of chaos. University of Washington Press, Seattle (1993)

[26] Lu, H., Wang, S., Wu, G.: Pseudo-random number generator based on coupled map lattices. Int. J. of Modern Phys. B 18(17-19), 2409–2414 (2004)

[27] Mao, Y.B., Chen, G., Lian, S.G.: A novel fast image encryption scheme based on the 3D chaotic baker map. Int. J. of Bifurcation and Chaos 14(10), 3613–3624 (2004)

[28] Misiurewicz, M.: Strange attractors for the Lozi-mappings, nonlinear dynamic. New York Acad. of Sciences 357, 348–358 (1980)

[29] Morita, S.: Bifurcations in globally coupled chaotic maps. Phys. Lett. A 211(5), 258–264 (1996)

[30] Pareschi, F., Rovatti, R., Setti, G.: Simple and effective post-processing stage for random stream generated by a chaos-based RNG. In: Int. Symp. Nonlinear Theory and its Applications, pp. 383–386 (2006)

[31] Richman, J.S., Moorman, J.R.: Physiological time-series analysis using approximate entropy and sample entropy. Am. J. Physiol. 278, 2039–2049 (2000)

[32] Schuster, H.G., Just, W.: Deterministic chaos: an introduction. Wiley-VCH, Weinheim (2005)

[33] Security Requirements for Cryptographic Modules, Federal Information Processing Standards 140-2 (2001)

[34] Smale, S.: Differentiable dynamical systems. Bulletin of the Amer. Math. Soc. 73, 747–817 (1967)

[35] Sprott, J.C.: Chaos and time-series analysis. Oxford University Press, Oxford (2003)

[36] Tang, K.W., Kwok, H.S., Tang, K.S., Man, K.F.: A chaos-based random number generator for 8-bit micro-controller system. Int. J. Bifurcation and Chaos 18(3), 851–867 (2008)

[37] Tian, C., Chen, G.: Chaos in the sense of Li-Yorke in coupled map lattices. Physica A 376, 246–252 (2007)

[38] Verhulst, P.F.: Recherches mathématiques sur la loi d'accroissement de la population. Nouv. mém. de l'Academie Royale des Sci. et Belles-Lettres de Bruxelles 18, 1–41 (1845)

[39] Verhulst, P.F.: Deuxième mémoire sur la loi d'accroissement de la population. Mém. de l'Academie Royale des Sci., des Lettres et des Beaux-Arts de Belgique 20, 1–32 (1847)

[40] Visual Recurrence Analysis (2007),
     http://www.myjavaserver.com/nonlinear/vra/download.html

[41] Weisstein, E.W.: Baker's map, MathWorld- A Wolfram Web Resource,
     http://mathworld.wolfram.com/BakersMap.html

[42] Willeboordse, F.H.: The spatial logistic map as a simple prototype for spatiotemporal chaos. Chaos 13(2), 533–540 (2003)

[43] Yang, W., Ding, E., Ding, M.: Universal scaling law for the largest Lyapunov exponent in coupled map lattices. Phys. Rev. Lett. 76(11), 1808–1811 (1996)

[44] Yen, J., Guo, J.: A new chaotic key-based design for image encryption and decryption. In: IEEE Int. Conf. Circuits and Systems, vol. 4, pp. 49–52 (2000)

[45] Zhang, H., Chen, G.: Single-input multi-output state-feedback chaotification of general discrete systems. Int. J. of Bifurcation and Chaos 14(9), 3317–3323 (2004)

[46] Zhao, L., Elbert, E.N.: A network dynamically coupled chaotic maps for scene segmentation. IEEE Trans. on Neural Networks 12(6), 1375–1385 (2001)

Appendix I: Examples of One-Dimensional Chaotic Maps

One-Dimensional Chaotic Maps	Illustration
Logistic [14]: $x(k+1) = ax(k)(1-x(k))$ where $x \in [0, 1]$	
Skew tent [3]: $x(k+1) = \begin{cases} \dfrac{x(k)}{p} & \text{if } 0 \le x(k) \le p \\ \dfrac{1-x(k)}{1-p} & \text{if } p < x(k) \le 1 \end{cases}$ where $x \in [0, 1]$	
Chebyshev [17]: $x(k+1) = \cos(a\cos^{-1}(x(k)))$ where $x \in [-1, 1]$	
Sine [35]: $x(k+1) = A\sin(\pi\, x(k))$ where $x \in [0, 1]$	
Sawtooth [32]: $x(k+1) = 2x(k) \bmod 1$ where $x \in [0, 1]$	
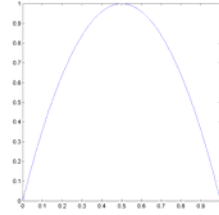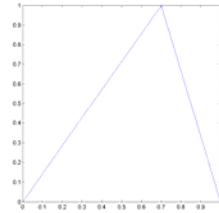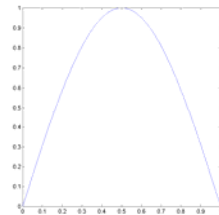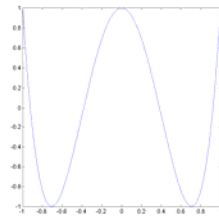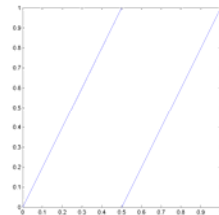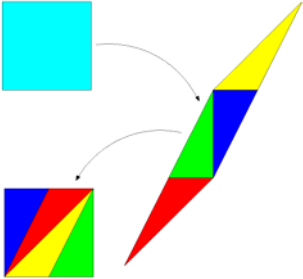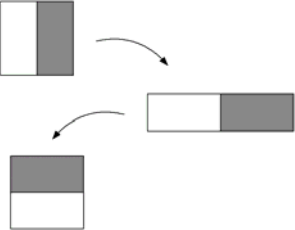
Appendix II: Examples of Two-Dimensional Chaotic Maps

Two-Dimensional Chaotic Maps	Illustration

Cat map [1]:

$$\begin{bmatrix} x(k+1) \\ y(k+1) \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix}\begin{bmatrix} x(k) \\ y(k) \end{bmatrix} \mathrm{mod}\, 1$$

where $x, y \in [0,1)$

Baker map [41]:

$$\begin{bmatrix} x(k+1) \\ y(k+1) \end{bmatrix} = \begin{cases} \begin{bmatrix} 2x(k) \\ y(k)/2 \end{bmatrix} & \text{if } 0 \le x(k) < \frac{1}{2} \\ \begin{bmatrix} 2x(k)-1 \\ y(k)/2+1/2 \end{bmatrix} & \text{if } \frac{1}{2} \le x(k) \le 1 \end{cases}$$

where $x, y \in [0,1)$

Hénon [13]:
$$x(k+1) = 1 - ax(k)^2 + by(k)$$
$$y(k+1) = x(k)$$

Lozi [28]:
$$x(k+1) = 1 - a|x(k)| + by(k)$$
$$y(k+1) = x(k)$$

Appendix III: Examples of Three-Dimensional Chaotic Maps

Three-Dimensional Chaotic Maps

Lorenz three-dimensional [25,35]:
$$x(k+1) = x(k)y(k) - z(k)$$
$$y(k+1) = x(k)$$
$$z(k+1) = y(k)$$

Modified Lissajous [9]:
$$x(k+1) = \sin(a \sin^{-1}(x(k)))$$
$$y(k+1) = \sin(b \sin^{-1}(x(k+1)))$$
$$z(k) = c \sin^{-1}(y(k))$$

Chapter 5
Chaos Based Hash Function

Di Xiao, Xiaofeng Liao, and Shaojiang Deng

College of Computer Science and Engineering, Chongqing University,
Chongqing 400044, China

Abstract. This chapter focuses on the construction of chaos-based hash function. Hash function is a special kind of one-way function which takes a variable-length input and returns a fixed-length value. As one of the cores of Cryptography, hashing is a basic technique widely used in information security. Utilizing chaos to construct hash function is a promising direction which attracts more and more attention. In this chapter, some preliminaries on hash function are firstly given in brief. Then, the systemic descriptions of different chaos-based construction approaches are presented in the order of the simple chaotic map-based hash function, the complex map-based hash function, the composite map-based hash function, the chaotic neural network-based hash function, the parallel hash function as well as the combined chaotic cryptographic and hashing scheme. Meanwhile, the detailed analyses of some typical chaos-based hash functions are described. Finally, by borrowing some principles from classical Cryptography, we summarize some instructions on chaos-based hash function secure construction which is beneficial to the hash function design based on chaos in the future.

1 Hash Function

A hash function takes a message of arbitrary length, and creates a message digest of fixed length. Hash functions may be split into two classes: *unkeyed hash functions*, whose specification dictates a single input parameter (a message); and *keyed hash functions*, whose specification dictates two distinct inputs (a message and a secret key). The hash value is sometimes referred to as a hash-code or message digest. Since the hash value is the function result of all the bits of the input message, any tiny change in the single bit of the message will lead to huge changes in the final hash value. Hash functions play a fundamental role in the field of Information Security. For example, hash functions actually provide an error detection function, which is utilized for data integrity authentication. Besides, to smoothly realize the digital signature scheme in E-Commence or E-Government, a message is typically

hashed first, and then the hash value, as a representative of the message, is signed in place of the original message.

Generally, a hash function has the following properties [1]:

① *Compression*: its input may be of arbitrary finite bit length, and its output is of fixed bit length, such as 128-bit or 256-bit.

② *Preimage resistance*: for essentially all pre-specified outputs, it is computationally infeasible to find any input which hashes to that output, i.e., to find any preimage x' such that $h(x') = y$ when given any y for which a corresponding input is not known.

③ *2nd-preimage resistanc*e—it is computationally infeasible to find any second input which has the same output as any specified input, i.e., given x, to find a 2nd-preimage $x' \neq x$ such that $h(x') = h(x)$.

④ *Collision resistance*: it is computationally infeasible to find any two distinct inputs x, x' which hash to the same output, i.e., such that $h(x') = h(x)$.
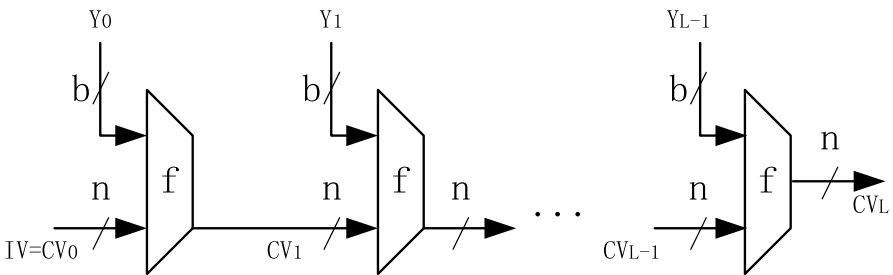
Fig. 1 The Merkle-Damgard scheme of iterated hash function

The Merkle-Damgard scheme, shown in Fig. 1, is the basis for many widely used hash functions today, such as MD5, SHA [2]. First of all, the message length and padding are appended to the message to create an augmented message which can be evenly divided into $Y_0, Y_1, \cdots, Y_{L-1}$, blocks of b bits, where b is the size of the block to be processed by the compression function f. The message length is embedded in the last block, which can increase the attack difficulty. To attack a hash function successfully, the attacker has to ensure both the hash value and the length of the forged message are equal to the original one. A compression function f is used repeatedly. In each round, the compression function f has two inputs: one is the n-bit output CV_{i-1} of the last round; the other is the b-bit input block Y_{i-1} of the current round. At the same time, the output of the current round is the n-bit CV_i, which is also used as the input of the next round. Before starting the iteration, the initial vector IV should be set. The output of the last round is the final hash value. The scheme can be described as follows:

$$CV_0 = IV = n \text{ -}bit \text{ initial value;}$$

$$CV_i = f(CV_{i-1}, Y_{i-1}), \quad 1 \le i \le L; \tag{1}$$

$$H(M) = CV_L.$$

To design a hash function, the core task is to design a compression function which is collision resistant. In the conventional hash function, the compression function is realized through the complicated method based on logical XOR operation or multi-round iterations of some available cipher. Since the security and efficiency of conventional hash function totally depend on the basic cipher, more complicated computation is required. Although each step of the former is simple, the processing round number will also be huge, even the message is very short. Since 2005, the collision research on hash function has achieved breakthrough. Some potential flaws in the collision resistance of widely used hash functions, such as MD5, SHA-1, are disclosed [3,4], which makes hash function become the common focus of both academia and enterprises.

As a ubiquitous phenomenon in nature, chaos is a kind of deterministic random-like process generated by nonlinear and deterministic dynamic systems. Utilizing chaos to construct hash function is a promising direction which attracts more and more attentions. The inherent merits of chaos, such as the sensitivity to tiny changes in initial conditions and parameters, mixing property, ergodicity, unstable periodic orbits with long periods and one-way iteration, have laid the potential theoretical foundation for excellent hash function construction.

Different classification criterions are summarized from different perspectives based on the systematic study on the known chaos-based hash functions:

① The processing unit of message: Partition the pending message into a number of fixed-length processing units and map them into decimal numbers by means of linear transform. In some character-wise algorithms, the processing unit includes 8 bits, and thus no extra padding is needed. While in some block-wise algorithms, the processing unit includes some specific-length bits, such as 512, and thus the pending message has to be padded in a correct manner that its length is a multiple of a specific fixed-length.

② The chaotic model: Choose a specific chaotic model, set its initial value and parameter as the algorithm secret key and start iteration. Until now, various chaotic models have been introduced into hash function construction, such as simple chaotic map, complex chaotic map, composite map and chaotic neural network, etc.

③ The modulated method: The message units are modulated into the chaotic iteration by influencing the chaotic phase space, parameter space or iteration times.

The known chaos-based hash functions are generally the combination of the above hash construction methods. In the following, we will give the systemic descriptions of different chaos-based construction approaches.

2 Simple Chaotic Map-Based Hash Function

2.1 Typical Algorithm One

2.1.1 The Algorithm Description

In [5], a simple one-way hash function is proposed based on Logistic map $f(X) = 1 - \mu * X^2$. This is a character-wise algorithm whose processing unit includes 8 bits, and thus no extra padding is needed. The message units are modulated into the chaotic parameter space and the initial value is set. The detailed procedure is as follows:

1）Translate the pending message to the corresponding ASCII numbers, and then map these ASCII numbers into a array by computing $\mu_i = 1.746 + 0.001 * Asc(m_i)$. The array length is the character number N of message. Each item of the array can be chosen as the parameter to ensure the system is chaotic.

2）Input μ_i into the chaotic system $f(X) = 1 - \mu * X^2$, and iterate it 32-time to get the sequence $\{X_i\}$, which has 32 items.

Table 1 Transform X_i into binary sequence

[7/8,1]	[6/8,7/8]	[5/8,6/8]	[4/8,5/8]	[3/8,4/8]	[2/8,3/8]	[1/8,2/8]	[0,1/8]
1111	1110	1101	1100	1011	1010	1001	1000

[-1/8,0]	[-2/8,-1/8]	[-3/8,-2/8]	[-4/8,-3/8]	[-5/8,-4/8]	[-6/8,-5/8]	[-7/8,-6/8]	[-1,-7/8]
0111	0110	0101	0100	0011	0010	0001	0000

3）By looking up Table 1, each $\{X_i\}$ is transformed to a $4 \times 32 = 128$-bit binary sequence S_i.

4）By XORing the binary sequences corresponding to each X_i, the final 128-bit hash value can be obtained.

2.1.2 Security Analysis

In the above algorithm, the message units are transformed to the chaotic parameter, and the chaotic initial value is set. By iterating the system, the chaotic sequences are produced and transformed to the binary sequences, and then the final hash value can be generated by XORing the binary sequences. This is a very simple hash

construction, which is suitable for software implementation. Only floating number multiplication is utilized. It is not like the conventional hash function which requires complicated computation and huge memory space.

However, there exists a serious security drawback in this algorithm. The cause of this drawback lies in the fact that the processing of each message unit is independent to each other. That is to say, for i^{th} message unit, it is firstly transformed to the chaotic parameter μ_i. Then, starting from the set chaotic initial value, the chaotic map is iterated to generate the chaotic sequence $\{X_i\}$ and the corresponding binary sequence S_i. Finally, all the binary sequences are XORed together to obtain the final hash value of the paragraph of message. The commutativity of XOR operation results in the collision drawback of the algorithm: any change in the order of the message unit of the message will not affect the final hash value. For example, the final hash value of the original message-" $ABCDEFG$ " is the same as that of the modified message-" $GFEDCBA$ ".

In order to overcome this drawback, the improvement should introduce some connection among the processing of message units, or make the result have a close relation with the orders of message units. One practical idea is to use the iteration result of the previous message unit as the initial iteration value of the next message unit.

2.2 Typical Algorithm Two and Its Variant

2.2.1 The Description of the Typical Algorithm Two

In [6], a one-way hash function is proposed based on Henon map $a_n = 1 + b(a_{n-2} - a_{n-3}) - ca_{n-1}^2$. This is a character-wise algorithm whose processing unit includes 8 bits, and thus no extra padding is required. The message units are modulated into the chaotic phase space, and the parameters $b = 0.3, c = 1.08$ are set. When the parameters are set as $b = 0.3, \ 1.07 \le c \le 1.09$, and the initial value within [-1.5, 1.5], the system occurs chaotic attractor. The detailed procedure is as follows:

1) Translate the pending message to the corresponding ASCII numbers, and then map these ASCII numbers into an array S whose element is a number which belongs to [-1.2, 1.2] and length is the character number N of message. This mapping is achieved by means of linear transform.

2) The first two items S_1, S_2 are used to generate the chaotic initial values:

$$a_1 = \frac{2.4 \times S_1}{255} - 1.2, \qquad a_2 = \frac{2.4 \times S_2}{255} - 1.2. \tag{2}$$

3) Start iteration: Set the iteration time to be

$$r = R \times ([N / R] + 1), \text{ where } R = 64. \tag{3}$$

(a) j from 3 to r, compute

$$a_j = 1 + 0.3(a_{j-2} - c) - 1.08a_{j-1}^2, \tag{4}$$

If $j \le N$, set $c = 2.4S_j / 255 - 1.2$,

If $j > N$, set $c = a_{j-3}$.

(b) set $b_1 = a_{r-2}, b_2 = a_{r-1}, b_3 = a_r$, j from 4 to $3R$, compute

$$b_j = 1 + 0.3(b_{j-2} - b_{j-3}) - 1.08b_{j-1}^2. \tag{5}$$

4) Transform b_R, b_{2R}, b_{3R} to the corresponding binary format, extract 40, 40, 48 bits after the decimal point, respectively, and juxtapose them from left to right to get a 128-bit final hash value.

In this algorithm, the message array S is utilized to generate the chaotic initial value and the c value of the iteration procedure, which introduces the influence of message into the iteration procedure.

2.2.2 Security Analysis of the Typical Algorithm Two

Unfortunately, this algorithm is weak in collision resistance. In the following, we will analyze the cause of this drawback. When we modulate the message unit into the chaotic phase space (as an initial value) and start iteration, a chaotic orbit will occur. However, if we choose other orbit values from the above chaotic orbit as the new initial iteration value and start iteration, the same orbit will occur definitely. The following is a collision instance [7]:

For a three-byte message $S = s_1 s_2 s_3$, the length is $N = 3$. Let

$$f(s) = \frac{2.4s}{255} - 1.2, \tag{6}$$

then the initial iteration value is

$$a_1 = f(s_1), \qquad a_2 = f(s_2);$$

Set

$$a_j = g(a_{j-1}, a_{j-2}, c) = 1 + 0.3(a_{j-2} - c) - 1.08a_{j-1}^2, \tag{7}$$

then the iteration sequence is as follows:

$$\begin{cases} a_3 = g(a_2,\ a_1,\ c) = g(a_2,\ a_1,\ f(s_3)) \\ a_4 = g(a_3,\ a_2,\ a_1) \\ \cdots \\ a_r = g(a_{r-1},\ a_{r-2},\ a_{r-3}) \end{cases} \tag{8}$$

For a four-byte message $S = s_1 s_2 s_3 s_4 = s_1 s_2 s_3 s_1$, the length is $N = 4$. By (6), the initial value is obtained as

$$a_1 = f(s_1), \qquad a_2 = f(s_2);$$

By (7), the iteration sequence is as follows:

$$\begin{cases} a_3 = g(a_2,\ a_1,\ c) = g(a_2,\ a_1,\ f(s_3)) \\ a_4 = g(a_3,\ a_2,\ f(s_4)) = g(a_3,\ a_2,\ a_1) \\ \cdots \\ a_r = g(a_{r-1},\ a_{r-2},\ a_{r-3}) \end{cases} \tag{9}$$

By comparing (8) and (9), it is easy to see that a_1, a_2, a_3, a_4 of the two sequences are equal to each other. By using mathematical induction, it can be proven that $a_5, \cdots, a_r$ of the two sequences are also equal to each other. That is to say, the two sequences are exactly the same.

If the iteration times are set as the same, namely $R \times ([3/R]+1) = R \times ([4/R]+1)$ (it is easy to satisfy this condition.), the subsequent sequences are definitely the same. Therefore, the final hash values are the same, and this causes collision.

Actually, the above analysis can be generalized as the following collision construction proposition: Let $S = s_1 s_2 \cdots s_{N-2} s_{N-1} s_N$ be a paragraph of message, another paragraph of message $S' = s_1 s_2 \cdots s_{N-2} s_{N-1} s_N s_{N-2}$ can be constructed to have the same final hash value $H(S) = H(S')$, if their iteration times are set as the same, namely $R \times ([N/R]+1) = R \times ([(N+1)/R]+1)$.

2.2.3 The Description of the Variant Algorithm One

In [8], by introducing the extended chaotic model into the above original hash algorithm, an improved one-way hash function based on the extended chaotic map switch is proposed. The basic idea of the extended chaotic model is to use a uniform equation to represent several chaotic maps under the precondition that the value

domains of these chaotic maps are the same. For example, since the value domains of Logistic map, Cubic map and Tent map are all within [-1, 1], their extended chaotic model can be constructed as:

$$x_{n+1} = g(x_n) = a_0 + a_1 x_n + a_2 x_n^2 + a_3 x_n^3 + a_4 \mid x_n \mid + a_5 sign(x_n - b_0)$$
$$+ a_6((1 + b_1(x_{n-1} - b_2) + b_3 x_n^2) \bmod 2 - 1). \tag{10}$$

Let $a_i(i = 0, 1, \cdots, 6)$ and $b_j(j = 0, 1, 2, 3)$ be different values, Eq. (10) can represent the specific chaotic map, respectively. For instance, if $a_0 = 1, a_2 = -2$ and the rest $a_i, b_j = 0$, $x_{n+1} = g(x_n)$ is actually Logistic map; if $a_1 = 3, a_3 = -4$ and the rest $a_i, b_j = 0$, $x_{n+1} = g(x_n)$ actually Cubic map.

The detailed procedure of the one-way hash function based on the extended chaotic map switch is as follows:

1）Translate the pending message to the corresponding ASCII numbers, and then map these ASCII numbers into an array S by means of linear transform whose element is a number which belongs to [-1, 1] and length is the character number N of message.

2）The first two items S_1, S_2 are used to generate the chaotic initial values:

$$a_1 = \frac{2 \times S_1}{k-1} - 0.8, \qquad a_2 = \frac{2 \times S_2}{k-1} - 0.8, \tag{11}$$

where $k = 256$.

3）Start iteration: Set the iteration time to be

$$r = R \times ([N / R] + 1), \tag{12}$$

where $R = 32$.

(a) j from 3 to r, compute as follows:

If $j \le r/5$, set $a_6 = 1, b_1 = 0.3, b_3 = 2917$, and the rest $a_i, b_j = 0$, then the extended chaotic model is actually

$$x_j = (1 + 0.3(x_{j-2} - c) + 2917 x_{j-1}^2) \bmod 2 - 1, \tag{13}$$

If $j \le N$, set $c = 2S_j / (k-1) - 0.8$,

If $j > N$, set $c = x_N$.

Similarly, when $r/5 \le j \le 2r/5$, $2r/5 \le j \le 3r/5$, $3r/5 \le j \le 4r/5$, $4r/5 \le j \le r$, different values of a_i, b_j are set to obtain different chaotic maps, respectively. For other extended chaotic models, the above method can also be utilized to modulate message into the iteration parameter.

(b) Continue iterating: j from 3 to $3R$, compute as follows:

If the current extended model is (13), set $c = x_{r-2}, x_1 = x_{r-1}, x_2 = x_r$, compute

$$x_j = (1 + 0.3(x_{j-2} - c) + 2917x_{j-1}^2) \bmod 2 - 1. \tag{14}$$

For other extended chaotic models, it is similar.

4) Transform x_R, x_{2R}, x_{3R} to the corresponding binary format, extract 40, 40, 48 bits after the decimal point, respectively, and juxtapose them from left to right to get a 128-bit final hash value.

2.2.4 Security Analysis of the Variant Algorithm One

By carefully comparing the above original algorithm and the variant one, it is easy to find out that their essential ideas are the same. The difference lies in the iteration procedure. In the variant algorithm, the extended chaotic map switch is carried out based on iteration times. Unfortunately, this variant algorithm is still weak in collision resistance. The cause of this drawback is similar to that of the original one. The following is a collision instance [7]:

For a two-byte message $S = s_1 s_2$, the length is $N = 2$. Let

$$f(s) = \frac{2 \times s}{255} - 0.8, \tag{15}$$

then the initial iteration value $a_1 = f(s_1), a_2 = f(s_2)$;

Set

$$a_j = g(a_{j-1}, a_{j-2}, c), \tag{16}$$

where $g(*)$ is the current chaotic map (the uniform equation of the five chaotic maps in [8]). The iteration sequence is as follows:

$$\begin{cases} a_3 = g(a_2, \ a_1, \ c) = g(a_2, \ a_1, \ f(s_2)) \\ a_4 = g(a_3, \ a_2, \ f(s_2)) \\ \dots \\ a_r = g(a_{r-1}, \ a_{r-2}, \ f(s_2)) \end{cases}. \tag{17}$$

For a three-byte message $S = s_1 s_2 s_3 = s_1 s_2 s_2$, the length is $N = 3$, where its 3rd byte is set as $s_3 = s_2$. By (15), the initial value is derived as

$$a_1 = f(s_1), \qquad a_2 = f(s_2);$$

By (16), the iteration sequence is as follows:

$$
\begin{cases}
a_3 = g(a_2,\ a_1,\ c) = g(a_2,\ a_1,\ f(s_3)) = g(a_2,\ a_1,\ f(s_2)) \\
a_4 = g(a_3,\ a_2,\ f(s_2)) \\
\dots \\
a_r = g(a_{r-1},\ a_{r-2},\ f(s_2))
\end{cases}
. \quad (18)
$$

Since the iteration subsection in step 3) of the variant algorithm one is only related to $r = R \times ([2/R]+1) = R \times ([3/R]+1)$, the iteration subsections of the above two conditions are the same. Furthermore, the chaotic map in each subsection is also the same to each other. By comparing (17) and (18), it is easy to see that the two sequences are equal to each other. And the subsequent sequences are definitely the same. Therefore, the final hash values are the same, and this causes collision.

2.3 Typical Algorithm Three

In the proposed algorithm of [9], two-dimensional and four-dimensional Cat Chaotic Maps（CATCM）are utilized.

Two-dimensional generalized CATCM is defined as:

$$
\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = A \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} (\text{mod}\,1) = \begin{bmatrix} 1 & a \\ b & 1+ab \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} (\text{mod}\,1) \quad (19)
$$

where $a, b, x_1(k), x_2(k)$ are integers in $[0,\ 2^L - 1]$.

Ten-dimensional generalized CATCM is defined as:

$$
\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \\ x_3(k+1) \\ \vdots \\ x_{10}(k+1) \end{bmatrix} = A \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \\ \vdots \\ x_{10}(k) \end{bmatrix} (\text{mod}\,2^L), \quad (20)
$$

where $A = A_{12}A_{13}\cdots A_{1\ 10}A_{23}A_{24}\cdots A_{2\ 10}\cdots A_{9\ 10}$ is a 10×10 matrix and

$$A = \begin{bmatrix} 1 & a_{12} & 0 & \cdots & 0 \\ b_{12} & 1+a_{12}b_{12} & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & a_{13} & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ b_{13} & 0 & 1+a_{13}b_{13} & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix} \cdots \begin{bmatrix} 1 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 1 & 0 & 0 \\ 0 & \cdots & 0 & 1 & a_{9,10} \\ 0 & \cdots & 0 & b_{9,10} & 1+a_{9,10}b_{9,10} \end{bmatrix},$$

where $a_{ij}, b_{ij}, x_1(k), x_2(k)\cdots, x_{10}(k)$ are integers in $[0,\ 2^L-1]$.

The 128-bit secret key of the algorithm is divided to four 32-bit numbers (a,b,c,d). Two-dimensional CATCM is used to generate the parameters a_{ij}, b_{ij} of ten-dimensional CATCM as follows:

$$\begin{bmatrix} y_1(k+1) \\ y_2(k+1) \end{bmatrix} = \begin{bmatrix} 1 & c \\ d & 1+cd \end{bmatrix}\begin{bmatrix} y_1(k) \\ y_2(k) \end{bmatrix}(\mathrm{mod}\,2^L), \tag{21}$$

where $y_1(0) = a, y_2(0) = b$ and $L = 32$.

2.3.1 The Algorithm Description

The parameters of the ten-dimensional CATCM are generated based on the algorithm key, and the partial initial values are also set. The message is divided into n-bit block, where $n = 32*(m-4)=32*6=192$. This is a block-wise algorithm and the message is modulated into the chaotic phase space.

1) Append the bit pattern of "10" repetitively to the message until its length is in a multiple of n. Separate the padded message into blocks, B_1, B_2, $\cdots, B_p$, each with n-bit and np is the total length of the padded message.

2) During the processing of the 1st block B_1, set the initial values: $x_1(1)=01\ 23\ 45\ 67$, $x_2(1)=89\ ab\ cd\ ef$, $x_3(1)= fe\ dc\ ba\ 98$, $x_4(1)=76\ 54\ 32\ 10$, divide B_1 into six sub-blocks and use them as the initial values of $x_5(1)$, $x_6(1)$, $x_7(1)$, $x_8(1)$, $x_9(1)$, $x_{10}(1)$. Iterate (20) once to obtain $x_1(2)$, $x_2(2),\cdots, x_{10}(2)$.

3) During the processing of the 2nd block B_2, keep the values of $x_1(2)$, $x_2(2)$, $x_3(2)$, $x_4(2)$, divide B_2 into six sub-blocks and use them as the initial values of $x_5(2)$, $x_6(2)$, $x_7(2)$, $x_8(2)$, $x_9(2)$, $x_{10}(2)$. Iterate (20) once to obtain $x_1(3)$, $x_2(3),\cdots, x_{10}(3)$.

4) Repeat the above process until the last block B_p has been processed. Juxtapose $x_1(p+1)$, $x_2(p+1)$, $x_3(p+1)$, $x_4(p+1)$ from left to right to get a 128-bit final hash value.

2.3.2 The Performance Analysis and Improvement

According to the performance data provided in [9], this algorithm can generally meet the performance requirement of hash function. But 1-bit change in message leads to only 58.353-bit change in the 128-bit final hash value, which is far from the ideal number 64-bit. Ref. [10] gave the detailed analysis of this point. The modulus operation is used in the original algorithm, which may cause different impact on the high and low bits in the binary sequence of message. As we know, the modulus operation is insensitive to high bits of the binary sequence but sensitive to low bits, so it may lead to collision. For two big numbers with different low bits, its modulus results are different; while for two big numbers with different high bits, its modulus results may be the same. Therefore, the collision resistance of this algorithm is low.

In order to improve the collision resistance of this algorithm, the sensitivity of hash value to different message bits should be made approximately even. Ref. [10] proposed the following improved algorithm to overcome the above potential flaws. For the convenience of comparison, only the different parts between the original algorithm and the improved one are stressed as follows:

The first modification is in step 1): Append the bit pattern of "10" repetitively with length l, such that $(m + l) \bmod N = N - 64$ (N is the length of hash value) and then append 64-bit denoting the length of the original message.

The second modification is embedded in step 2) and 3): For each $B_i\,(i=1,\cdots,p)$, after generating $x_1(i), x_2(i), \cdots, x_{10}(i), i=1,2,\cdots,p$, each elements of 32-bit length, the following sub-steps are processed as the post-processing:

(a) For each 32-bit $x_j(i), j=1,2,\cdots,10$, get the corresponding $x_j'(i), j=1,2,\cdots,10$, which is the reverse binary format of $x_j(i)$, and then compute $x_j''(i) = x_j(i) \oplus x_j'(i)$.

(b) For each 32-bit $x_j''(i), j=1,2,\cdots,10$, the XOR operation between the last 16-bit binary format of $x_j''(i), j=1,2,\cdots,10$ and the top 16-bit binary format of $x_j''(i), j=1,2,\cdots,10$ is processed as the top 16-bit binary format of $x_j''(i), j=1,2,\cdots,10$ and cascade the last 16-bit binary format

of $x_j^{"}(i), j = 1, 2, \cdots, 10$ to generate the new $x_j^{"}(i), j = 1, 2, \cdots, 10$. Replace the current values of $x_j(i), j = 1, 2, \cdots, 10$ with $x_j^{"}(i), j = 1, 2, \cdots, 10$.

Simulation results have shown that the improved performance is significant.

2.4 Typical Algorithm Four

In [11], Yi combined chaotic iteration with DM(Davies-May) scheme and used 75-time iterations of a chaotic tent map as a block cipher to generate the hash value. This algorithm divides message into blocks for further processing. The message units are modulated into the chaotic iteration by influencing both the chaotic phase space and parameter space.

In this algorithm, a chaotic tent map $F_\alpha(x_i)$ with parameter α is defined as follows:

$$F_\alpha(x_i) = \begin{cases} \dfrac{x_{i-1}}{\alpha}, & 0 \le x_{i-1} \le \alpha \\ \dfrac{1 - x_{i-1}}{1 - \alpha}, & \alpha \le x_{i-1} \le 1 \end{cases}, \tag{22}$$

where $0 < \alpha < 1$.

By using $F_\alpha(x_i)$, a map G_α is defined as follows:

$$G_\alpha : x_i = \begin{cases} F_{A(\alpha)}(x_{i-1}), & 0 < x_{i-1} < 1 \\ \beta, & others \end{cases}, \tag{23}$$

where $A(\alpha)$ is an affine mapping from $[0,1)$ to a sufficiently small interval around 0.5, and $0 < \beta < 1$ is a constant. G_α^n represents $\overbrace{G_\alpha \circ G_\alpha \cdots \circ G_\alpha}^{n}$.

2.4.1 Algorithm Description

1) First of all, a message M with arbitrary length is padded so that the size of the message is a multiple of l. Then the padded message is broken into blocks M_0, M_1, $\cdots, M_{r-1}$, each has l bits (where the last l-bit block M_r represents the length of M in bits).

Transform l-bit block $M_i = (p_{i1} p_{i2} \cdots p_{il})$ (where $i = 0, 1, \cdots, r$) into a pair of binary fractions $(m_i, \tilde{m}_i)$, where $m_i = 0.p_{i1} p_{i2} \cdots p_{il}$ and $\tilde{m}_i = 0.p_{il} p_{i(l-1)} \cdots p_{i1}$.

2) Let $i = 0$ and input a pair of common initial binary fractions $(s_0, \ t_0)$, where $s_0 = 0.s_{01}s_{02}\cdots s_{0l}$, $t_0 = 0.t_{01}t_{02}\cdots t_{0l}$. They need not to be kept secret.

3) Hash the quadruple binary fractions $(s_i, \ t_i, \ m_i, \ \tilde{m}_i)$ into a pair of binary fractions $(s_{i+1}, \ t_{i+1})$ with a hash round function H. In order to describe the hash round function, two symbols $\boxed{+}$ and $\otimes$ are defined as

$$
\begin{aligned}
x\boxed{+}y &= (x+y)(\mathrm{mod}\,1)\\
x\otimes y &= G_{\min(x,y)}(\max(x,y))
\end{aligned}
\tag{24}
$$

In the hash round function H, $G_\alpha^n(x_0)$ is used as a block cipher, and in $G_\alpha^n(x_0)$, $x_0 = s_i\boxed{+}m_i, \alpha = t_i\boxed{+}\tilde{m}_i$. Let

$$
z_i = G_\alpha^n(x_0) = G_{t_i\boxed{+}\tilde{m}_i}^n(s_i\boxed{+}m_i)
\tag{25}
$$

the hash round function H can be decomposed into two hash round functions H_1, H_2 as:

$$
\begin{aligned}
H_1 : t_{i+1} &= s_i\boxed{+}z_i\\
H_2 : s_{i+1} &= t_i\otimes(\tilde{m}_i\boxed{+}z_i)
\end{aligned}
\tag{26}
$$

4) Let $i = i+1$. If $i \le r$, then go to step 3).

5) Output $2\,l$-bit hash value $h = (a_1a_2\cdots a_l b_1 b_2\cdots b_l)$, according to the pair of binary fractions $(s_{r+1} = 0.a_1a_2\cdots a_l\cdots, \ t_{r+1} = 0.b_1b_2\cdots b_l\cdots)$ which are the output of the last iteration.

2.4.2 Performance Analysis

1) Statistical analysis of $G_\alpha^n(x_0)$

$G_\alpha^n(x_0)$ plays a critical role in the hash round function. It is evolved from the chaotic tent map and its features are determined by two parameters α, n.

(a) Uniform distribution
It is proved in [11]: For any $0 \le \alpha < 1$, the distribution of $x_1 = G_\alpha(x_0)$ for randomly chosen $0 < x_0 < 1$ is the standard uniform distribution $U(0,1)$. It is also proved in [11]: For any $0 \le \alpha < 1$ and any n, the distribution of $x_n = G_\alpha^n(x_0)$ for randomly chosen $0 < x_0 < 1$ is the standard uniform distribution $U(0,1)$.

(b) The determination of the minimum number of iterations

Ref. [11] has proved that the distribution of $G_{\alpha+\Delta\alpha}^n(x_0)$ for even tiny $\Delta\alpha$ is independent of the distribution of $G_\alpha^n(x_0)$ if the number of iterations $n \geq 73$. Therefore, $n = 75$ is chosen as the the minimum number of iterations in this hash algorithm.

2) Security analysis

This hash algorithm essentially belongs to the iterated hash function. Any attack on the hash round function implies an attack of the same type on the iterated hash function with the same computation complexity. In this algorithm, $G_\alpha^n(x_0)$ is thought as a "block cipher," and two sub-round functions H_1, H_2 are both similar to the well-known DM scheme. Therefore, H_1, H_2 are both believed to have almost the same computational complexities for the attack as DM scheme. In other words, this hash function has at least the same computational security against the attack as DM scheme.

2.5 Typical Algorithm Five

We proposed a simple algorithm for one-way Hash function construction based on the chaotic map with changeable-parameter in [12]. A piecewise linear chaotic map with changeable-parameter P is chosen, and the message is modulated into the chaotic iteration by affecting the chaotic parameter space. The initial chaotic value is set and the processing unit is 8-bit character. In this algorithm, cipher block chaining mode (CBC) is introduced to ensure that the parameter P in each iteration is dynamically determined by both the last-time iteration value and the corresponding message bit in different positions. The final hash value is derived by means of the linear transform on the iteration sequence. Theoretical analysis and computer simulation indicate that this algorithm can meet the performance requirements of hash function in an efficient and flexible manner.

2.5.1 Algorithm Description

One dimension piecewise linear chaotic system is defined as:

$$X(t+1) = F_P(X(t)) = \begin{cases} X(t)/p, & 0 \leq X(t) < p \\ (X(t)-P)/(0.5-P), & P \leq X(t) < 0.5 \\ (1-X(t)-P)/(0.5-P), & 0.5 \leq X(t) < 1-P \\ (1-X(t))/P, & 1-P \leq X(t) \leq 1, \end{cases} \tag{27}$$

where $X \in [0, 1]$, $P \in (0, 0.5)$ (see Fig.2). By [13], $\{X(t)\}$ is ergodic in [0, 1], and the auto-correlation function of $\{X(t)\}$ is δ-like. The Frobenius-Perron operator of invariant density $f^*(x)$ in the system is
$$P_r(f^*(x)) = pf^*(xp) + (0.5-p)f^*(p+x(0.5-p)) + (0.5-p)f^*(0.5+(1-x)*(0.5-p)) + pf^*(1-xp).$$
Since $f^*(x) = 1$, $\{X(t)\}$ is uniformly distributed in [0, 1]. These inherent merits lay the theoretical foundation for hash function construction.
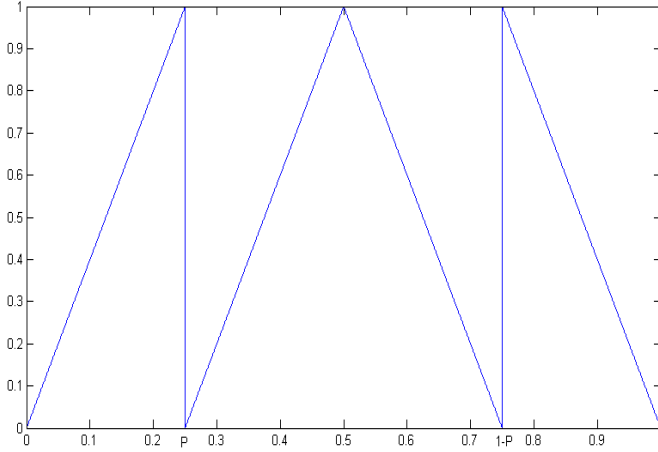
Fig. 2 A piecewise linear chaotic map

Secretly chooses $X_0 \in [0, 1], H_0 \in (0, 1)$ as the keys. CBC mode is introduced into the algorithm to dynamically change the parameter P during iteration on the joint foundation of the last-time iteration value and the corresponding different message bit. The 3-unit iterations, $1^{st} \sim N^{th}, (N+1)^{th} \sim 2N^{th}, (2N+1)^{th} \sim 3N^{th}$, ensure that each bit of the final hash value will be related to all the bits of message. The detailed algorithm can be described as follows:

1) Translate the pending message to the corresponding ASCII numbers, and then map these ASCII numbers into an array C by means of linear transform whose element is a number which belongs to [0, 1] and length is the character number N of message.

2) The iteration process is as follows:

1st: $P_1 = (C_1 + H_0)/4 \in (0, 0.5)$, $X_1 = F_{P1}(X_0) \in (0, 1)$;

$2^{nd} \sim N^{th}$: $P_i = (C_i + X_{i-1})/4 \in (0, 0.5)$, $X_i = F_{Pi}(X_{i-1}) \in (0, 1)$;

$(N+1)^{th}$: $P_{N+1} = (C_N + X_N)/4 \in (0, 0.5)$, $X_{N+1} = F_{P_{N+1}}(X_{i-1}) \in (0, 1)$;

$(N+2)^{th}$~$2N^{th}$: $P_i = (C_{(2N-i+1)} + X_{i-1})/4 \in (0,0.5),$ $X_i = F_{Pi}(X_{i-1}) \in (0,1);$

$(2N+1)^{th}$: $P_{2N+1} = (C_1 + H_0)/4 \in (0,0.5),$ $X_{2N+1} = F_{p_{2N+1}}(X_{i-1}) \in (0,1);$

$(2N+2)^{th}$~$3N^{th}$: $P_i = (C_{(i-2N)} + X_{i-1})/4 \in (0,0.5),$ $X_i = F_{pi}(X_{i-1}) \in (0,1).$

Note that, if a certain iteration value X_i is equal to 0 or 1, then an extra iteration is carried out. The property of chaos can ensure that this kind of extra iteration time is very few.

3) Transform X_N, X_{2N}, X_{3N} to the corresponding binary format, extract 40, 40, 48 bits after the decimal point respectively, and combine them to get a 128-bit final Hash value.

2.5.2 Performance Analysis

For different chaotic hash algorithms, their performance analysis methods are similar to each other. We take this algorithm as a sample to demonstrate the detailed hash performance analysis. Due to page limitation, for other hash algorithms, we only emphasize its unique performance and omit the rest performances.

1) Hash results of messages

Hash simulation experiments have been done under the following different 6 conditions:

Condition 1: The original message is "*As a ubiquitous phenomenon in nature, chaos is a kind of deterministic random-like process generated by nonlinear dynamic systems. The properties of chaotic cryptography includes: sensitivity to tiny changes in initial conditions and parameters, random-like behavior, unstable periodic orbits with long periods and desired diffusion and confusion properties, etc. Furthermore, benefiting from the deterministic property, the chaotic system is easy to be simulated on the computer. Unique merits of chaos bring much promise of application in the information security field.*".

Condition 2: Change the first character *A* in the original message into *B*.

Condition 3: Change the word *unstable* in the original message into *anstable*.

Condition 4: Change the full stop at the end of the original message into a comma.

Condition 5: Add a blank space to the end of the original message.

Condition 6: Change the secret key X_0=0.232323 to X_0=0.232325.

The corresponding hash values in hexadecimal format are:

Condition 1: 64AF14F9F6A10F6218247D71E4B73C2C

Condition 2: B65812A2AF70B57C38901D9C8A88C62D

Condition 3: E52461DF5D1D93313040D7778F269B6B

Condition 4: 6575ABF7BE7527B75E6F95E55F1A40AE

Condition 5: B641F329EB27D9EE75184D8CA08C0BB0

Condition 6: 8AE38465D60B268062F512C9D58EB170.

The graphical display of binary sequences is shown in Fig.3.
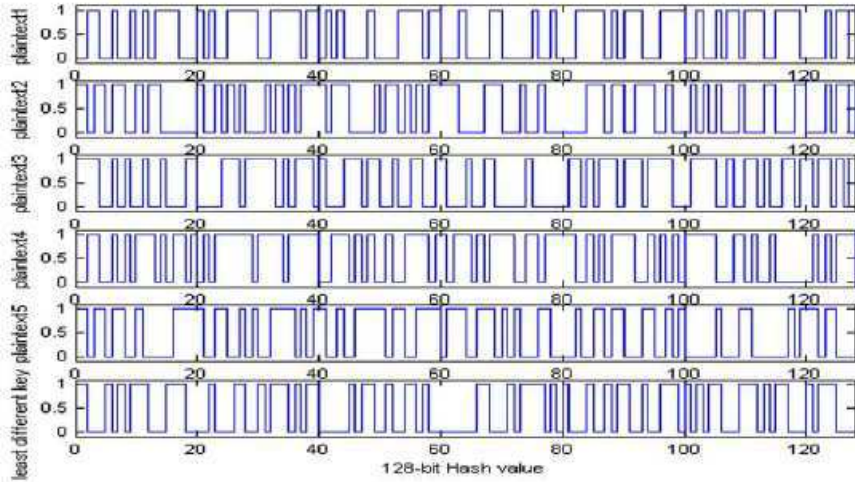
Fig. 3 Hash values under different conditions

The simulation result indicates that the one-way property of the proposed algorithm is so perfect that any difference of the message or key will cause huge changes in the final Hash value.

2) Statistic analysis of diffusion and confusion

In order to hide message redundancy, Shannon introduced diffusion and confusion. Diffusion means spreading out of the influence of a single plaintext bit over many ciphertext bits so as to hide the statistical structure of the plaintext. Confusion refers to the use of transformations that complicate dependence of the statistics of ciphertext on the statistics of plaintext. They are two general principles to guide the design of a practical cipher, including Hash function. For the Hash value in binary format, each bit is only 1 or 0. So the ideal diffusion effect should be that any tiny changes in initial conditions lead to the 50% changing probability of each bit.

We have performed the following diffusion and confusion test: A paragraph of message is randomly chosen and hash value is generated; then a bit in the message is randomly selected and toggled and a new hash value is generated. Two hash values are compared and the number of changed bit is counted as B_i. This kind of
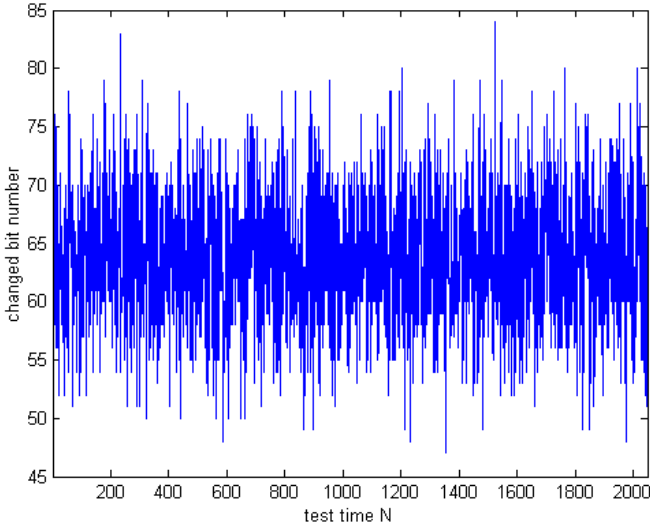
Fig. 4 Distribution of changed bit number

test is performed N times, and the corresponding distribution of changed bit number is shown as Fig.4, where $N = 2048$.

Obviously, the changed bit number corresponding to 1 bit changed message concentrates around the ideal changed bit number-64 bit. It indicates that the algorithm has very strong capability for diffusion and confusion. Usually, four statistics are defined as follows:

Mean changed bit number:

$$\overline{B} = \frac{1}{N} \sum_{i=1}^{N} B_i, \tag{28}$$

Mean changed probability:

$$P = (\overline{B}/128) \times 100\%, \tag{29}$$

Mean changed bit number of B:

$$\Delta B = \sqrt{\frac{1}{N-1} \sum_{i=1}^{N} (B_i - \overline{B})^2}, \tag{30}$$

Mean changed bit number of P:

$$\Delta P = \sqrt{\frac{1}{N-1} \sum_{i=1}^{N} (B_i/128 - P)^2} \times 100\%. \tag{31}$$

Through the tests with $N = 256, 512, 1024, 2048$, respectively, the corresponding data are listed in Table 2.

Table 2 Number of changed bit B_i

	N=256	N=512	N=1024	N=2048	Mean
$\overline{B}$	64.4414	63.8008	63.8398	63.8501	63.9830
P/%	50.34	49.84	49.87	49.88	49.9825
ΔB	5.5218	5.7081	5.7078	5.7898	5.6819
ΔP /%	4.31	4.46	4.46	4.52	4.4375

Based on the analysis of data in Table 2, we can draw the conclusion: the mean changed bit number $\overline{B}$ and the mean changed probability P are both very close to the ideal value 64 bit and 50%. While ΔB and ΔP are very little, which indicates the capability for diffusion and confusion is very stable.

3) Analysis of collision resistance and birthday attacks resistance

Collision resistance and birthday attacks are similar in idea. They are essentially a probability problem that two random input data $x' \neq x$ are found such that they are hashed to the same output. In our proposed algorithm, CBC mode is introduced to ensure that the parameter P in each-time iteration is decided by both the last-time iteration value and the corresponding message bit in different positions. This inherent structure expedites the avalanche effect, which will ensure that each bit of the final hash value will be related to all the bits of message, and even a single bit change in message or key will be diffused and result in great changes in the final hash value.

We have performed the following test to do quantitative analysis on Collision resistance [14]: first, the hash value for a paragraph of message randomly chosen is generated and stored in ASCII format. Then a bit in the message is selected randomly and toggled. A new hash value is then generated and stored in ASCII format. Two hash values are compared and the number of ASCII character with the same value at the same location in the hash value is counted. Moreover, the absolute difference of two hash values is calculated using the formula: $d = \sum_{i=1}^{N} |t(e_i) - t(e_i')|$, where e_i and e_i' be the ith ASCII character of the original and the new Hash value, respectively, and the function $t(*)$ converts the entries to their equivalent decimal values. This kind of collision test is performed 2048 times, with the secret key $x_0 = 0.232323, H_0 = 0.858485$. The maximum, mean, minimum values of d and Mean/character are listed in Table 3. A plot of the

Table 3 Absolute differences of two Hash values

Absolute difference d	Maximum	Minimum	Mean	Mean/character
File1	2221	696	1506	94.125

distribution of the number of ASCII characters with the same value at the same location in the Hash value is given in Fig.5. Notice that the maximum number of equal character is only 3 and the collision is very low.
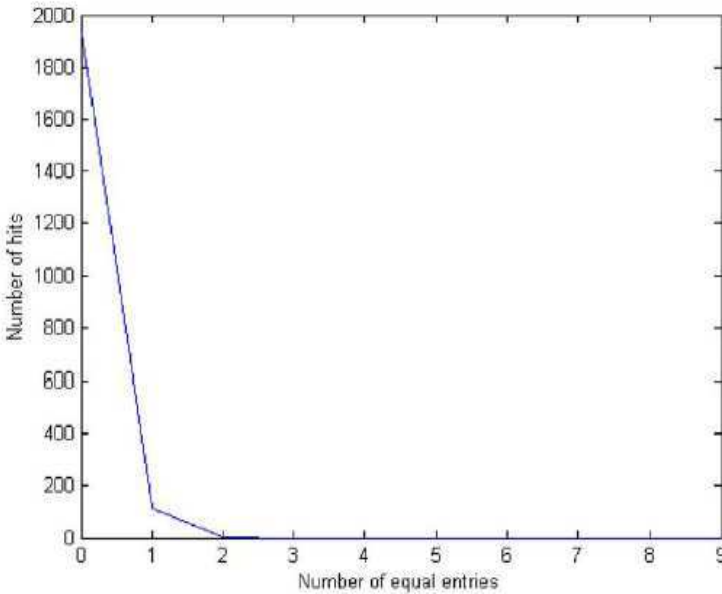
Fig. 5 Distribution of the number of ASCII characters with the same value at the same location in the Hash value

4) Security of Key

In the proposed algorithm, $X_0 \in [0, 1], H_0 \in (0, 1)$ of the piecewise linear chaotic map are chosen as the secret keys. The key space is huge enough to resist any exhaustive key search. Moreover, for the sensitivity to tiny changes in initial conditions and parameters, it is absolutely impossible to inversely deduce the value of X_0, H_0 from the iteration values.

5) Analysis of Speed

First, the proposed algorithm is proportional to the length of the original message, and no padding is required, while most of the other known algorithms always need

to do padding for the length requirement. When the original message is very short, the other algorithms still have to do lots of computation while the proposed algorithm only need a little.

Second, the proposed algorithm has the parellel property. Three iteration values X_N, X_{2N}, X_{3N} generate 40, 40, 48 bits of the final Hash value, respectively, which corresponds to the parallel computation of 5, 5, 6 Bytes.

Furthermore, one dimension piecewise linear chaotic map is chosen in our algorithm. On the one hand, its dynamical property is enough for the algorithm security; on the other hand, its structure is so simple that only one multiplication (division) and several additions (subtractions) are operated in each-iteration, which reduces the algorithm complexity and guarantees the high efficiency.

Generally, the numbers of the required multiplicative operations for each ASCII character (8-bit) message during the hash process are computed for speed's comparison among different algorithms. Since each multiplicative operation consumes much more time than each additive operation, this kind of comparison is objective, in spite of different implementing platforms. The required multiplicative operation for each character in this algorithm is six times. Compared with other similar algorithms, it is the fastest one.

6) Flexibility

Through simply modifying the way to process X_N, X_{2N}, X_{3N}, the length of the final Hash value will be easily changed. Compared with the conventional hash algorithm such as MD5 with fixed 128-bit length, the proposed algorithm can adapt to the practical demand better.

3   Complex Map-Based Hash Function

3.1   Typical Algorithm One (Using Hyper-Chaotic Map)

3.1.1   Algorithm Description

In [15], based on the basic idea of the algorithm in Section 2.2, by replacing Henon map in the original algorithm with the hyper-chaotic map, a one-way hash function based on two-dimensional hyper-chaotic map is proposed. This is still a character-wise algorithm whose processing unit includes 8 bits, and thus no any extra padding is required. The message units are modulated into the chaotic phase space.

1）Translate the pending message to the corresponding ASCII numbers, and then map these ASCII numbers into an array S by means of linear transform whose element is a number which belongs to [-1, 1] and length is the character number N of message.

2）The first two items S_1, S_2 are used to generate the chaotic initial values:

$$a_1 = \frac{2 \times S_1}{255} - 0.8, \qquad a_2 = \frac{2 \times S_2}{255} - 0.8. \tag{32}$$

3）Start iteration: set the iteration time to be

$$r = R \times ([N/R] + K), \tag{33}$$

where R, K are both integers larger than 0.

(a) j from 3 to r, compute (Using the extended chaotic model)

$$a_j = ((1 + 0.3(a_{j-2} - c) + 2917a_{j-1}^2) \bmod 2) - 1, \tag{34}$$

If $j \le N$, set $c = 2S_j / 255 - 0.8$,

If $j > N$, set $c = a_{j-3}$.

(b) By means of linear transform, a_{r-1}, a_r are transformed to be $a'_{r-1} = 1.045a_{r-1} - 0.55$, $a'_r = 1.245a_r + 0.615$；（Using the hyper-chaotic map）

Set $x_1 = a'_{r-1}, x_2 = a'_r$, m from 2 to $3R$, compute

$$\begin{aligned} x_{m+1} &= 1.66y_m - 1.3y_m^2 \\ y_{m+1} &= -1.1x_m + 0.3y_m \end{aligned}. \tag{35}$$

4）Transform $x_R, x_{2R}, x_{3R}, y_R, y_{2R}, y_{3R}$ to the corresponding binary format, extract 40, 40, 48, 40, 40, 48 bits after the decimal point, respectively. Then by extracting 15th-34th bits from each 40 bits and extractihg 13th-36th bits from each 48 bits, 20, 20, 24, 20, 20, 24 bits are obtained and juxtaposed from left to right to get a 128-bit final hash value.

3.1.2 Security Analysis

By carefully comparing the original algorithm in Section 2.2 and the above one, it is easy to find out that their essential ideas are the same. The difference is that the hyper-chaotic map is introduced into step 3) (b). Naturally, this algorithm is still weak in collision resistance. The cause of this drawback is similar to that of the original one-"when we modulate the message unit into the chaotic phase space (as an initial value) and start iteration, a chaotic orbit will occur. However, if we choose other orbit values from the above chaotic orbit as the new initial iteration value and start iteration, the same orbit will occur definitely." The following is a collision instance [7]:

The only thing that we need to do is to replace the above (6) and (7) with the following two formulas:

$$f(s) = \frac{2 \times s}{255} - 0.8, \tag{36}$$

$$a_j = g(a_{j-1}, a_{j-2}, c) = ((1 + 0.3(a_{j-2} - c) + 2917a_{j-1}^2) \bmod 2) - 1, \tag{37}$$

Then the subsequent analysis and conclusion are the same.

3.2 Typical Algorithm Two (By Influencing the Parameter of Spatiotemporal Chaos)

Recently, spatiotemporal chaos has been attracting more and more interests from researchers. Spatiotemporal chaos has its advantages in cryptography. Spatiotemporal chaotic system is a high-dimensional chaotic system, which has a number of positive Lyapunov exponents that guarantee the complex dynamics of system. Compared with a single chaotic system, it is more difficult or even impossible to predict the time series of spatiotemporal chaos. Besides, the lattices are coupled each other. It is more difficult to deduce the last lattice orbit value based on the known current orbit values, namely its one-way property is stronger.

In [16], a hash function is constructed by using the plaintext to affect the parameter of spatiotemporal chaos. The plaintext is partitioned into blocks and then processed block-wisely.

The chaotic system and its operations are as follows:

$$y_{n+1}(1) = (1 - \varepsilon)f[y_n(1)] + \varepsilon f[y_n(N), a(1)] \tag{38}$$

$n = 0, 1, ..., P\text{-}1$.

$$y_{n+1}(i) = (1 - \varepsilon)f[y_n(i)] + \frac{1}{2}\varepsilon\{f[y_n(i)] + f[y_n(N), a(i)]\} \tag{39}$$

$i = 2, 3, ... , N\text{-}1$.

$$y'_{n+1}(N) = (1 - \varepsilon)f[y_n(N)] + \frac{1}{N-1}\varepsilon\sum_{i=1}^{N-1}f[y_n(i), b(i)] \tag{40}$$

$$y_{n+1}(N) = \{S \cdot I[y'_{n+1}(N)]\} / 2^{32} \tag{41}$$

$$f(y) = 4y(1 - y), \qquad f(y, a) = (3.75 + a/4) * y(1 - y), \qquad a \in [0,1], \tag{42}$$

where $f[y(N), a(i)]$ denotes the coupling from the Nth map to the ith map, while $\sum_{i=1}^{N-1} f[y_n(i), b(i)]$ represents the average coupling to the Nth map from all other maps.

The operation I in Eq. (41) represents a transformation from a real number to a 32-bit integer one:

$$I(y') = 2^{50} y' \bmod 2^{32} . \tag{43}$$

The operation S denotes a transformation from a 32-bit integer to another 32-bit one. This transformation can be divided to three steps as shown in Fig. 6:

1）The integer $I(y')$ is represented in a binary form

$$Y' = \sum_{i=1}^{32} Y_i' 2^{32-i} , \quad Y_i' \in \{0,1\} , \tag{44}$$

which produces a binary sequence

$$Y' = (Y_1', Y_2', ..., Y_{32}') .$$

2）The 32-bit binary sequence Y' is divided to four blocks, each has 8 bits:

$$Y' = [Y'(1), Y'(2), Y'(3), Y'(4)],$$

where

$$Y'(1) = (Y_1', Y_2', ..., Y_8') \quad , \quad Y'(2) = (Y_9', Y_{10}', ..., Y_{16}') \quad , \quad Y'(3) = (Y_{17}', Y_{18}', ..., Y_{24}') \quad ,$$

$$Y'(4) = (Y_{25}', Y_{26}', ..., Y_{32}') .$$

3）These four blocks are transformed to four new blocks (each has 8 bits) as

$$Y(1) = Y'(1) \oplus Y'(2) \oplus Y'(3) \oplus Y'(4) , Y(i) = Y'(i), \quad i = 2, 3, 4 . \tag{45}$$

The four blocks of $Y(i)$, $i = 1, 2, 3, 4$ are combined together to form again a 32-bit integer.

3.2.1 Algorithm Description

1）Choose the initial variables for lattices. One can choose any arbitrary $y_0(1)$, $y_0(2)$, ... , $y_0(N)$ within (0, 1).

2）The message m is padded to t blocks, each has $L = 64(N-1)$ bits, and the block number t is equal to the message length divided by L. In the following, we
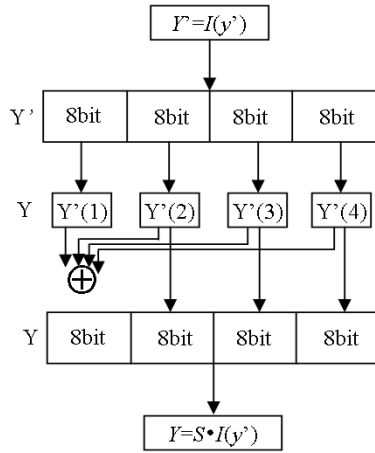
Fig. 6 *S* Operation

will consider operations for a single arbitrary block X of the L-bit: $X = (X_1, X_2, ..., X_{64(N-1)})$. The L-bit message block X is divided to 2(N−1) smaller 32-bit blocks: $X=[X(1), X(2), ..., X(2N-2)]$, where $X(i) = (X_{32(i-1)+1}, X_{32(i-1)+2}, ..., X_{32i})$, $i=1,2, ..., 2N$-2. Each $X(i)$ has the corresponding integer value:

$$X(i) = \sum_{j=1}^{32} X_{32(i-1)+j} 2^{32-j} . \tag{46}$$

3）$X(i)$ are finally transformed to real parameters $a(i)$, $b(i)$ in (39) and (40) as:

$$a(i) = X(i)/2^{32}, b(i) = X(N-1+i)/2^{32}, i = 1,2, ..., N\text{-}1. \tag{47}$$

4）We perform P iterations of Eq. (38)-(42) with the parameter set $a(i)$ and $b(i)$, and obtain the final state $y_p(i)$, $i=1, 2, ..., N$. If all the blocks of message m have not been processed, then set $y_0(i) = y_p(i)$, $i=1, 2, ..., N$, and input the next block and go to step 3).

5）The hash value is produced from the final state $y_p(i)$ of the tth block as

$$\sum_{j=1}^{32} h_{32(i-1)+j} 2^{32-j} = S \cdot I[y_p(i)]. \tag{48}$$

Every state variable outputs 32 bits, and the whole hash value has the length of 32N bits.

The above hash function is illustrated in Fig. 7.

(a)

(b)
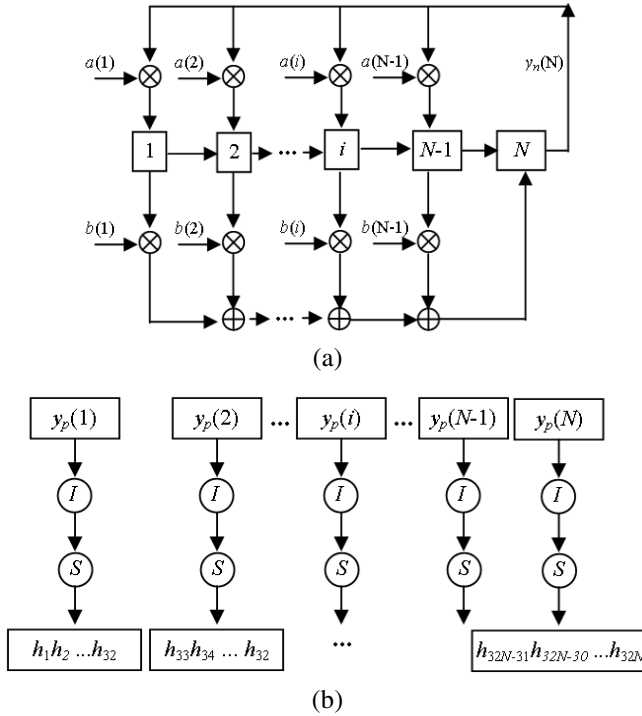
Fig. 7 Chaos-based hash function (a) Dynamical iteration (b) Hash value output

3.2.2 Security Analysis

In this algorithm, the plaintext block is transformed to be the parameter of spatiotemporal chaos. Through the iteration of the chaotic system and the coupling among the lattices, the influence of plaintext on the system phase is strengthened so that the generated hash value has close relation to the plaintext. Mapping the plaintext to the parameter of spatiotemporal chaos or using the plaintext to affect the parameter of spatiotemporal chaos is a typical scheme to construct a hash function. This scheme can construct the sensitivity of hash value to the message. It has strong security. Since the plaintext affects the chaotic state indirectly through the chaotic parameter, the difficulty to find collision is increased. As shown in the performance simulation, the above algorithm enjoys good performance and security property. However, there are also some drawbacks as follows:

1) This algorithm is not suitable for hardware implementation.
2) The choosing principle of some parameter is not clear. By optimizing the parameter range, it is promising to improve its performance and security.

3.3   Typical Algorithm Three (By Adjusting the State of Spatiotemporal Chaos)

In [17], two ideas are combined to construct hash function based on spatiotemporal chaos. The first idea is using plaintext to control or adjust the state of spatiotemporal chaos, iterating the chaotic system and extracting the final orbit values to generate hash value; the second idea is adopting the structure similar to the traditional hash function, as shown in Fig. 1. The algorithm structure of the proposed hash function based on the two-dimensional coupled map lattices (2D CML) is shown in Fig.8. Several-time iterations of 2D CML is used as the compression function; the original plaintext is divided into blocks G_1, G_2, ...,G_R, and they are transformed to be the state of some lattices. In this way, message is modulated into 2D CML. Hash value is obtained by extracting the final state of the lattices.
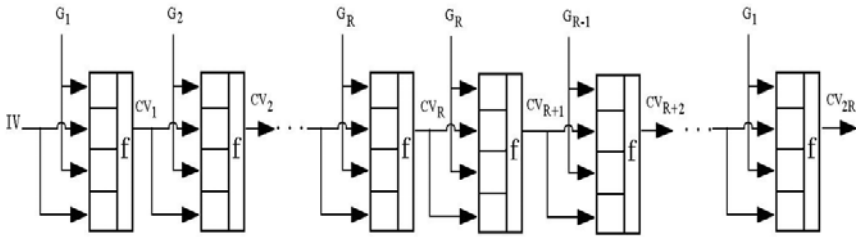
Fig. 8 The structure of Hash function based on the 2D CML

3.3.1   Analysis and Parameter Set of the 2D CML Mode

A general nearest-neighbor 2D CML can be described as

$$x_{n+1}^{i,j} = (1-\varepsilon)f(x_n^{i,j}) + \frac{\varepsilon}{4}[f(x_n^{i+1,j}) + f(x_n^{i-1,j}) + f(x_n^{i,j+1}) + f(x_n^{i,j-1})], \qquad (49)$$

where n=1, 2, … is the time index or state index; i=1,2, … , M is the lattice row index; j=1, 2, … , N is the lattice column index; f is a local chaotic map and $\varepsilon \in (0,1)$ is a coupling constant. To reduce the computational complexity, we use the following model to construct Hash function:

$$x_{n+1}^{i,j} = (1-\varepsilon)f(x_n^{i,j}) + \frac{\varepsilon}{2}[f(x_n^{i+1,j}) + f(x_n^{i,j+1})]. \qquad (50)$$

The periodic boundary conditions, $x_n^{M+i,j} = x_n^{i,j}$ and $x_n^{i,N+j} = x_n^{i,j}$, are used in 2D CML. Here the Logistic map is taken as the local map, given as $f(x) = \mu x(1-x)$, where $\mu \in (3.57,4)$ is a constant.

It is well known that the system is chaotic when the largest Lyapunov exponent (LLE) is larger than 0. And the larger the LLE is, the more chaotic the system is.

Since μ, ε, M, N may affect the LLE of system, we calculate it with one parameter varied and other three parameters fixed, and observe the influence of variable parameter on the LLE [18]. According to the result of numerical calculation, some conclusions can be drawn below:

1) The LLE increases as the parameter μ increases;

2) The parameter ε has slightly influence on LLE. And when $\mu = 4$, the LLE firstly decreases as the parameter ε increases, then keeps almost the same, finally increases slightly.

3) When $M \geqslant 4$ and $N \geqslant 8$, there is almost no effect on the LLE as the size of 2D CML increases.

Based on the conclusion drawn above, we take $\mu = 4$, $\varepsilon = 0.05$, $M = 4$, $N = 8$ in Eq.(50), that is,

$$x_{n+1}^{i,j} = 3.8 x_n^{i,j}(1 - x_n^{i,j}) + 0.1[x_n^{i+1,j}(1 - x_n^{i+1,j}) + x_n^{i,j+1}(1 - x_n^{i,j+1})]. \tag{51}$$

3.3.2 Algorithm Description and Determining the Minimum Iteration Number

INPUT: bitstring y of arbitrary bitlength;

OUTPUT: 128-bit Hash value.

1) *Definition of constants.* Define sixteen 8-bit initial values (IVs) which are expressed in hexadecimal format, that is : IV[1...16] = [01,23,45,67,89,AB,CD,EF,FE,DC,BA,98,76,54,32,10]. And set

$$x_1^{1,j} = (IV[j] + 0.8)/256, \; x_1^{3,j} = (IV[8+j] + 0.8)/256, \tag{52}$$

where $j = 1, 2, \dots, 8$.

2) *Preprocessing.* Pad y such that its bit length is a multiple of 128. Here we append some 0-bits to the last block if necessary. Then translate the padded y to the corresponding ASCII numbers and map these ASCII numbers into array C whose element is a number $\in (0,1)$. The map is achieved by linear transform:

$$C[i] = (A[i] + 0.8)/256, \tag{53}$$

where $A[i]$ is the ith ASCII number of y and $C[i]$ is the ith element of array C. Divide the array C into R groups, each group consisting of l elements(here $l = 16$):

$$C = \underbrace{C[1]C[2],...,C[l]}_{G_1}, \underbrace{C[l+1],...,C[2l]}_{G_2}, ..., \underbrace{C[(R-1)l+1],...,C[Rl]}_{G_R}. \tag{54}$$

For the convenience of description, we define $G_m = G_m[1] \; G_m[2]... \; G_m[l]$ and $G_m[i] = C[(m-1)*l+i]$, $m = 1, 2, \dots, R$.

3) *Processing.* Input G_m, $m=1,2,...,R$ into 2D CML and begin iteration. The detailed procedure is shown in Fig. 9.
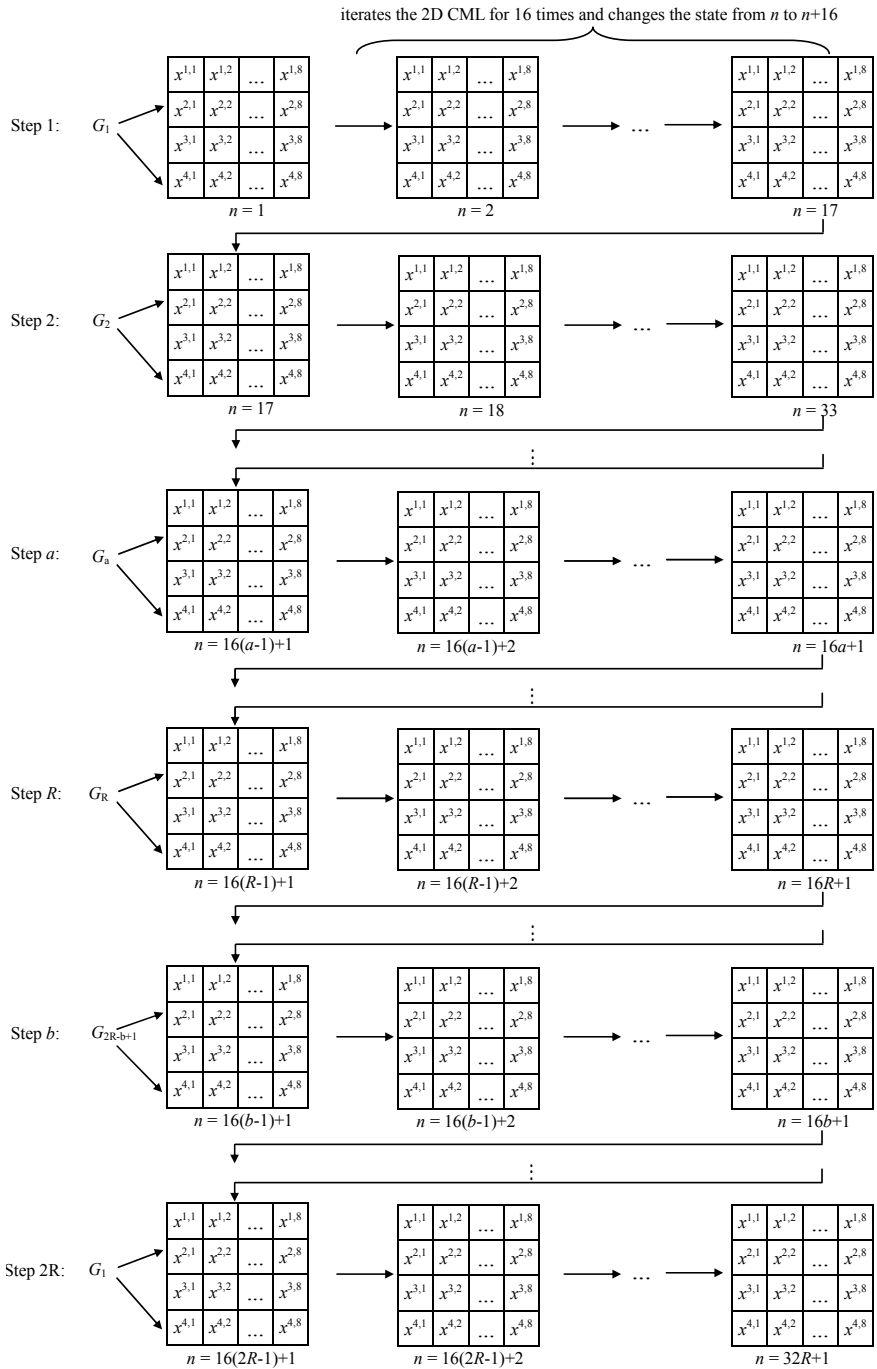
Fig. 9 The details from step 1 to step 2*R*

Step 1 ~ Step R : Define $a \in [1, R]$ as step index. In step a , the lattice values in the second and fourth rows are modified according to formula (55):

$$x^{2,j}_{K(a-1)+1} = G_a[j], \; x^{4,j}_{K(a-1)+1} = G_a[8+j] \quad (j=1, 2, \dots , 8). \tag{55}$$

Then according to Eq.(51), iterate the 2D CML for K times and change the system from state $K(a-1)+1$ to state $Ka+1$.

Step $(R+1)$ ~ Step $2R$: Define $b \in [R+1, 2R]$ as step index. In step b, the lattice values in the second and fourth rows are modified according to formula (56):

$$x^{2,j}_{K(b-1)+1} = G_{2R-b+1}[j], \; x^{4,j}_{K(b-1)+1} = G_{2R-b+1}[8+j] \quad (j = 1, 2, \dots , 8). \tag{56}$$

Then according to Eq. (51), iterate the 2D CML for K times and change the system from state $K(b-1)+1$ to state $Kb+1$. In the end, the final state of 2D CML is $2KR+1$.

4)Transform the final lattice values in the first and third rows of 2D CML, that is $x^{i,j}_{2KR+1}$ ($i = 1, 3; j=1,2,\dots,8$), to the corresponding binary format and extract 8 bits (from 9th to 16th bit after decimal point) from each x . Finally, juxtapose these bits from left to right to get a 128-bit hash value.

Let us discuss how to determine the minimum iteration number K in step 3) through χ^2 test. Let $m = 11$, $S=1000$, $\Delta\alpha = 1/256$, the result of χ^2 test is shown in Fig. 10.
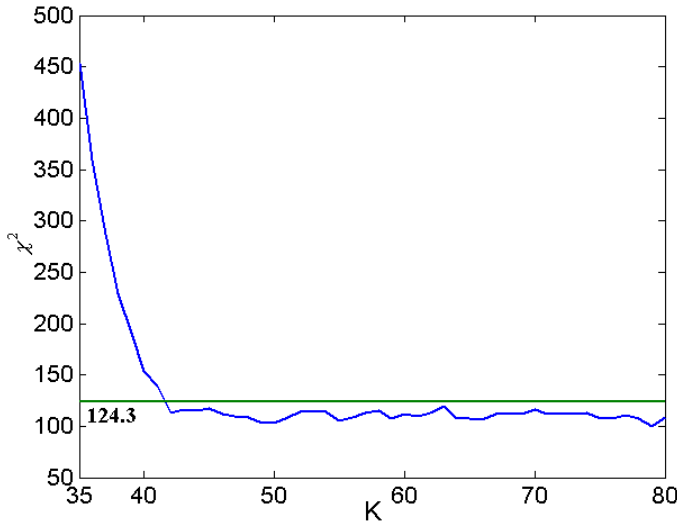
Fig. 10 Result of χ^2 test

If the χ^2 values are smaller than the upper 5% critical point of χ^2 of which the number of the degrees of freedom is $(m-1)^2$, the independence is not rejected at level of significance 0.05. From Fig. 10, we can see that the independence is not rejected when $K>43$ because the upper 5% critical point of χ^2_{100} is 124.3. Leaving an assurance margin, we choose $K=45$ as the number of iterations.

4   Composite Map-Based Hash Function

In [19], a composite discrete chaotic system is designed and a keyed hash function is proposed. In this algorithm, the initial value of the composite discrete chaotic system is set and the plaintext is divided into blocks. During the iteration, the chaotic maps are determined by the plaintext block (The message units are actually modulated into the chaotic parameter space).

First, two functions are defined within $[0, 1]$:

$$f_0(x) = \sqrt{|2x-1|}, \tag{57}$$

$$f_1(x) = 1 - \sqrt{|2x-1|}. \tag{58}$$

Two iteration systems $x_{n+1} = f_r(x_n)(r=0,1)$ and the composite discrete chaotic system $(f_0(x), f_1(x), R)$ can be obtained. Ref. [19] has proven their properties:

1）$x_{n+1} = f_r(x_n)(r=0,1)$ is a chaotic iteration system.

2）Their invariant density functions are $\rho_0(x) = 2x$ and $\rho_1(x) = 2-2x$, respectively.

3）They are complementary.

4）The auto-correlation functions of two iteration systems $x_{n+1} = f_r(x_n)(r=0,1)$ can both satisfy $C_{fr}(n) = (\frac{2}{3})^n C_{fr}(0)$.

5）The binary sequence R exists, which can ensure the auto-correlation functions of the composite discrete chaotic system $(f_0(x), f_1(x), R)$ satisfies $C(n) = \delta(0)$.

4.1   Algorithm Description

The detailed hash function algorithm based on the composite discrete chaotic system $(f_0(x), f_1(x), R)$ is as follows:

Without loss of generality, let the plaintext M be binary sequence, $N \geq 128$ be the bit-length of hash value. First of all, the original message M is padded such that its length is a multiple of N. Then after padding, M is constituted by blocks with N bits, $M = (M_1, M_2, \cdots, M_s)$, and each block is indicated as $M_i = m_i^1 m_i^2 \cdots m_i^N$. The initial vector $H_0 = \{0\}_1^N$ is set. The basic idea of the algorithm is: The chosen secret key x_0 is set as the initial value of the composite discrete chaotic system. Let $H_0 \oplus M_1$ be the composite sequence. By iterating the composite chaotic system, the orbit $\{y_i\}_1^N$ can be obtained and then transformed into a binary sequence which is the hash value H_1 of M_1. Similarly, $x_1 = y_N$ is set as the initial value and let $H_1 \oplus M_2$ be the composite sequence. Then the hash value H_2 can be obtained. By repeating the above procedure until the last plaintext block, the hash value H_s of M can be obtained. As shown in Fig. 11, the algorithm structure can be described as follows:

$$(H_i, x_i) = F(x_{i-1}, H_{i-1} \oplus M_i), i = 1, 2, \cdots, s;$$

$$H(M) = H_s,$$

(59)
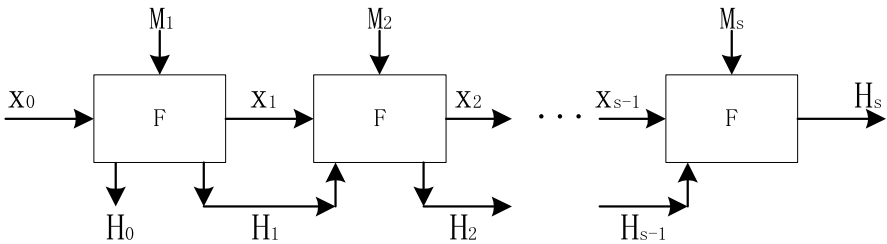
Fig. 11 The hash function based on the composite discrete chaotic system

where $F(*,*)$ represents the procedure to obtain the hash value of a plaintext block, which is essentially the iteration procedure of the composite discrete chaotic system.

During the iteration, the orbit $\{y_i\}_1^N$ is transformed to a binary sequence by computing $T_r(x) = \lfloor 2^r x \rfloor \mod 2$, where r is a natural number. F is actually a boolean function based on the composite chaotic system, which can be described further as follows:

1）From $i = 1$ to $i = s$:

a）$y_0 = x_{i-1}$.

b） $q = H_{i-1}^j \oplus m_i^j, y_j = f_q(y_{j-1}), H_i^j = T_r(y_j), j = 1, 2, \cdots, N.$

c） $y_0 = y_N.$

d） $q = H_i^j, y_j = f_q(y_{j-1}), H_i^j = T_r(y_j), j = 1, 2, \cdots, N.$

2） $H_i = H_i^1 H_i^2 \cdots H_i^N, x_i = y_N.$

4.2 The Performance Analysis

Ref. [19] has carried out the performance analysis. Because of the sensitivity and randomicity of the iteration process, there is a very complicated nonlinear connection between the hash value and the corresponding message, and then every bit of the hash value is related to all the bits of the message M . Any tiny changes in the message will lead to huge changes in the hash value. Besides, if the secret key has tiny changes, the difference is strengthened and diffused by the composite chaotic system to a totally different hash value. What's more, Ref. [19] has also proven that $H(M)$ distributes uniformly in $GF(2)^{(N)}$.

5 Chaotic Neural Network-Based Hash Function

Compared with the simple chaotic map, Chaotic Neural Network (CNN) has stronger spatiotemporal complexity. Its confusion and diffusion properties have been used to design encryption algorithms, such as the stream cipher and the block cipher. Furthermore, the structure with multi-input and single-output leads to good compression. At the same time, given the inner structure parameters, it is easy to compute the output when the input is given; while for the sensitivity to tiny changes in the initial conditions and parameters of chaotic map, it is very hard to find the input when the output is given. The inherent merits of chaotic neural network make them suitable for hash function design.

5.1 Typical Algorithm One

In [20], a hash function is proposed based on a two layer neural network whose input layer and output layer have eight neurons and four neurons, respectively. The message is divided into blocks, and the message is modulated into the chaotic phase space.

5.1.1 Algorithm Description

The original message M is padded such that its length is a multiple of 256. Then after padding, M is divided into blocks with 256 bits, $M_0, M_1, \cdots, M_{n-1}$. Each

block will be processed by CNN, respectively.   The output C_i of the ith CNN can be used as the secret key K of the $i+1^{\text{th}}$ Block Hash function to generate the weight, the bias and the transfer function parameter of this CNN. As shown in Fig. 12, the whole structure of the algorithm based on CBC (Cipher Block Chaining) mode can be described in (60).

$$\begin{cases} C_i = CNN(K_i, \ M_i), \ \ i=1,2,\cdots,n-1 \\ H(M) = G(C_{n-1}), \end{cases} \tag{60}$$

where the function G juxtaposes four 32-bit fractions of C_{n-1} from left to right to get a 128-bit hash value.
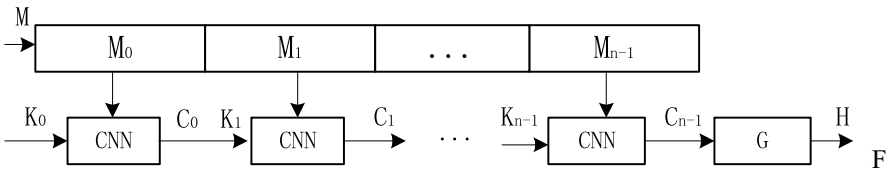

Fig. 12 CBC Hash function mode

The chaotic neural network structure of the Block Hash function is illustrated in Fig. 13. It is composed of two layers: eight-neuron input layer and four-neuron output layer. The piecewise linear chaotic map（PWLCM）（61）is utilized as the transfer function of each layer.
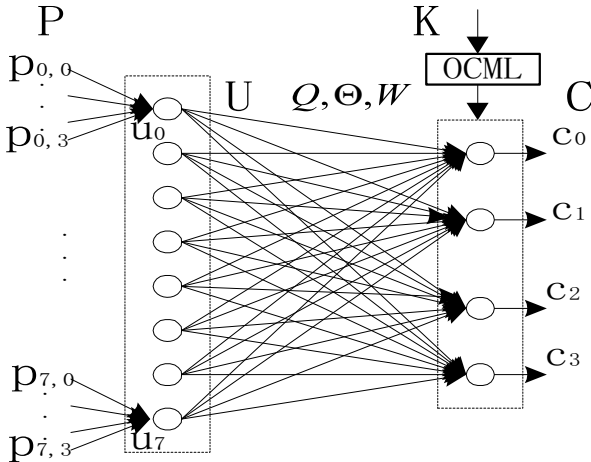

Fig. 13 The chaotic neural network structure of the Block Hash function

$$f(x,Q) = \begin{cases} x/Q, & 0 \le x < Q \\ (x-Q)/(0.5-Q), & Q \le x < 0.5 \\ (1-Q-x)/(0.5-Q), & 0.5 \le x < 1-Q \\ (1-x)/Q, & 1-Q \le x \le 1, \end{cases} \tag{61}$$

The detailed algorithm is as follows:

1）The 256-bit block M_i is re-divided into 32 units, each with 8-bit. Then four 8-bit units constitute a group. There are totally eight groups, namely, $P_{0,0}, \cdots, P_{0,3}, P_{1,0}, \cdots, P_{1,3}, \cdots, P_{7,0}, \cdots, P_{7,3}$.

2）According to the weight $W_1 = [1/2^8 \quad 1/2^{16} \quad 1/2^{24} \quad 1/2^{32}]$, four 8-bit units $P_{i,0}, P_{i,1}, P_{i,2}, P_{i,3}$ $i = 0,1,\cdots,7$ in each group can be transformed into the corresponding 32-bit fraction P_i, $i = 0,1,\cdots,7$ within [0, 1] by computing $P_i = [P_{i,0}/2^8 + P_{i,1}/2^{16} + P_{i,2}/2^{24} + P_{i,3}/2^{32}]$.

3）Input P_i, $i = 0,1,\cdots,7$ into the eight neurons of the input layer as the initial values of the transfer function, set the transfer function parameter $Q = 1/3$, and the iteration time $\tau = 40$. For the input $P = [p_{0,0}, \cdots, p_{0,3}, p_{1,0}, \cdots, p_{1,3}, \cdots, p_{7,0}, \cdots, p_{7,3}]^T$, where $p_{i,j} \in \{0,1,\cdots,2^8-1\}$, the output $U = [u_0, u_1, \cdots, u_7]^T$, $u_i \in [0,1]$ can be obtained by

$$u_i = f^\tau(P_i, 1/3), \quad i \in \{0,1,\cdots 7\}. \tag{62}$$

4) The output layer realizes good compression and makes the plaintext and secret key diffused and confused to a large extent. Each neuron i in the output layer has connection with all the neurons in the input layer. Let $W_2(i, \quad j) \in (0, \ 1)$ be the weight of the connection between Neuron j in the input layer and Neuron i in the output layer, $\Theta_i \in [0, \ 1]$ be the bias of each neuron i and $Q_i \in (0, \ 0.5)$ be the transfer function parameter. During the process of the each message block, the parameter vectors W_2, Θ, Q are generated by the current key K based on a One-way Coupled Map Lattice (OCML) as follows: The 128-bit key K is divided into four 32-bit integers and transformed into four fractions k_1, k_2, k_3, k_4 within [0, 1]. The four fractions k_1, k_2, k_3, k_4 are used as the initial values, and OCML is iterated continuously. An orbit value is obtained in every 30-time iteration, and all together ten orbit values are obtained. The former eight four-dimensional orbit

values are set as the weight W_2; the latter two four-dimensional orbit values are set as the bias Θ and the transfer function parameter Q, respectively (totally 40 values).

$$
\begin{aligned}
x_1(i+1) &= (1-\varepsilon)g(x_1(i)) + \varepsilon g(x_4(i)) \\
x_2(i+1) &= (1-\varepsilon)g(x_2(i)) + \varepsilon g(x_1(i)) \\
x_3(i+1) &= (1-\varepsilon)g(x_3(i)) + \varepsilon g(x_2(i)) \\
x_4(i+1) &= (1-\varepsilon)g(x_4(i)) + \varepsilon g(x_3(i))
\end{aligned}
\tag{63}
$$

where function g is Logistic map $x(i+1) = 4x(i)(1-x(i))$, and the coupling constant $\varepsilon = 1/3$.

   5) Let W_2^i be the ith row of the weight matrix W_2. For the output U of the input layer, the output $C = [c_0,\ \ c_1,\ \ c_2,\ \ c_3]^T$ of the four neurons in the output layer can be obtained by

$$
c_i = f^\tau(\mathrm{mod}(W_2^i \bullet U + \Theta_i,\ 1),\ Q_i),\quad i \in \{0,\ 1,\ \cdots,3\},
\tag{64}
$$

where iteration time $\tau = 40$. The obtained four outputs in the output layer are juxtaposed from left to right to get a 128-bit hash value.

5.1.2 Security Analysis

In [21], the authors noticed the chaotic map in this algorithm had symmetry property, and analyzed the collision weakness of the plaintext pair and the key pair.

1)   The plaintext pair collision
The transfer function $f(x,Q)$ of the neuron has symmetry property, namely $f(x,Q) = f(1-x,Q)$. Therefore, if two groups- a_0, a_1, a_2, a_3 and b_0, b_1, b_2, b_3 satisfy $a_0 / 2^8 + a_1 / 2^{16} + a_2 / 2^{24} + a_3 / 2^{32} = 1 - (b_0 / 2^8 + b_1 / 2^{16} + b_2 / 2^{24} + b_3 / 2^{32})$, then the plaintext pair- $a_0, a_1, a_2, a_3, P_{1,0}, P_{1,1}, P_{1,2}, P_{1,3}, \cdots, P_{7,0}, P_{7,1}, P_{7,2}, P_{7,3}$ and $b_0, b_1, b_2, b_3, P_{1,0}, P_{1,1}, P_{1,2}, P_{1,3}, \cdots, P_{7,0}, P_{7,1}, P_{7,2}, P_{7,3}$ will obtain the same hash value under the same secret key.

   The detailed analysis is as follows: For the two plaintexts, in the above step 3), the obtained orbit values after one-time iteration are the same. Similarly, the obtained orbit values after forty-time iterations are the same. Since the output of the output layer has connection only with the secret key and the output of the input layer, the final hash value will be the same under the same secret key.

2)   The key pair collision

The Logistic map $g(x) = 4x(1-x)$ has symmetry property, namely $g(x) = 4x(1-x)$. Therefore, if two 32-bit integers a, b satisfy $a/2^{32} = 1 - b/2^{32}$, then the key pair- a, k_2, k_3, k_4 and b, k_2, k_3, k_4 will obtain the same hash value for the same plaintext.

The detailed analysis is as follows: For the two secret keys, in the above step 4), the obtained orbit values after one-time iteration are the same. Therefore, the obtained parameters W_2, Θ, Q are the same. Finally, the obtained hash values are the same for the same plaintext.

5.2   Typical Algorithm Two

In [22], a hash function is proposed based on a three-layer neural network whose input layer, hidden layer and output layer have eight neurons, eight neurons and four neurons, respectively. The message is divided into blocks, and the message is modulated into the chaotic phase space.

The original message M is padded such that its length is a multiple of 1024. Then after padding, M is divided into blocks, $M_0, M_1, \cdots, M_{n-1}$, each with 1024-bit. The processing is illustrated in Fig. 14. The hash value H_{M_i} of M_i block is modulated by its key $K_{M_{i-1}}$. Thus, the final hash value is

$$H_M = K_{M_{n-2}} \oplus H_{M_{n-1}} = (K_{M_{n-3}} \oplus H_{M_{n-2}}) \oplus H_{M_{n-1}} = \cdots = (K \oplus H_{M_0}) \oplus H_{M1} \oplus \cdots \oplus H_{M_{n-1}}. \quad (65)$$
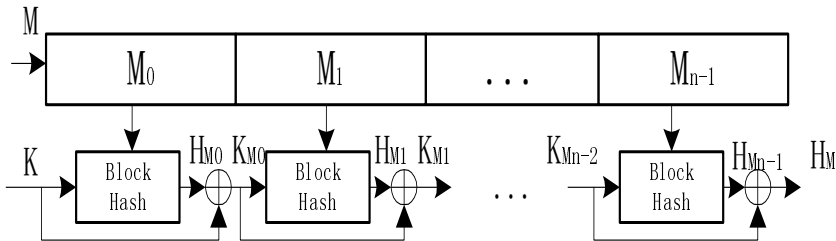
Fig. 14 The multi-block hash mode

The chaotic neural network structure of the Block Hash function is illustrated in Fig. 15. It is composed of three layers: the input layer, the hidden layer and the output layer. The piecewise linear chaotic map（PWLCM）（61）is utilized as the transfer function of each layer.
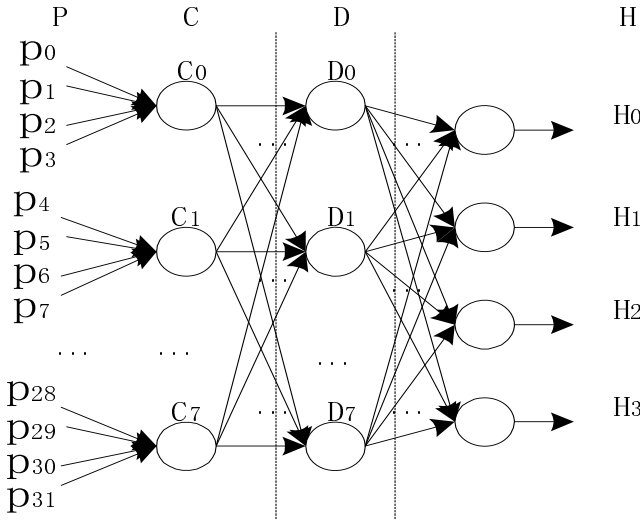
Fig. 15 The chaotic neural network structure of the Block Hash function

During the process of the ith message block M_i, the parameters $W_0, B_0, Q_0, W_1, B_1, Q_1, W_2, B_2, Q_2$ of the neural network are generated by the current key $K_{M_{i-1}}$, where W_i, B_i, Q_i are the weight, the bias and the transfer function parameter of the ith neuron layer, respectively. The total numbers of these parameters are 151. The key $K_{M_{i-1}}$ is divided into four sub-keys: $K_0 = k_0 k_1 \cdots k_{31}, K_1 = k_{32} k_{33} \cdots k_{63}, K_2 = k_{64} k_{65} \cdots k_{95}, K_3 = k_{96} k_{97} \cdots k_{127}$.

They are quantized and used to generate all the parameters as follows:

$$X_0(k) = f^{T+k}(K_0, K_1),$$
$$X_1(k) = f^{T+k}(K_2, K_3), \tag{66}$$
$$K_s(k) = (X_0(k) + X_1(k)) \bmod 1.$$

where $K_s(k)(k = 0,1,\cdots,150)$ is the kth sub-key, and the iteration time $\tau \geq 50$. The module operation is defined as

$$a \bmod 1 = \begin{cases} a, & 0 \leq a \prec 1 \\ a-1, & 1 \leq a \prec 2 \end{cases}.$$

In the input layer, there are eight neurons, and each neuron has four 32-bit inputs. By computing the following Eq. (67), the input data are transformed into the output data.

$$C = \begin{bmatrix} f^T(\sum\limits_{i=0}^{3} w_{0,i}P_i + B_{0,0}, Q_0) \\ f^T(\sum\limits_{i=4}^{7} w_{0,i}P_i + B_{0,1}, Q_0) \\ \vdots \\ f^T(\sum\limits_{i=28}^{31} w_{0,i}P_i + B_{0,7}, Q_0) \end{bmatrix} = \begin{bmatrix} C_0 \\ C_1 \\ \vdots \\ C_7 \end{bmatrix}, \tag{67}$$

where $W_0 = [w_{0,0} \quad w_{0,1} \quad \cdots \quad w_{0,31}]$ is the weight, B_0 is the bias, and Q_0 is the parameter of the transfer function. And the iteration time $\tau \geq 50$

Similarly, the following computations are carried out in the hidden layer and the output layer, respectively:

$$D = \begin{bmatrix} f^T(\sum\limits_{i=0}^{7} w_{1,0,i}C_i + B_{1,0}, Q_1) \\ f^T(\sum\limits_{i=0}^{7} w_{1,1,i}C_i + B_{1,1}, Q_1) \\ \vdots \\ f^T(\sum\limits_{i=0}^{7} w_{1,7,i}C_i + B_{1,7}, Q_1) \end{bmatrix} = \begin{bmatrix} D_0 \\ D_1 \\ \vdots \\ D_7 \end{bmatrix}, \tag{68}$$

$$H = \begin{bmatrix} f^T(\sum\limits_{i=0}^{7} w_{2,0,i}D_i + B_{2,0}, Q_2) \\ f^T(\sum\limits_{i=0}^{7} w_{2,1,i}D_i + B_{2,1}, Q_2) \\ \vdots \\ f^T(\sum\limits_{i=0}^{7} w_{2,3,i}D_i + B_{2,3}, Q_2) \end{bmatrix} = \begin{bmatrix} H_0 \\ H_1 \\ \vdots \\ H_3 \end{bmatrix}, \tag{69}$$

where W_1, W_2 are the weights, B_1, B_2 are the biases, and Q_1, Q_2 are the parameter of the transfer functions. And the iteration time $\tau \geq 50$.

The obtained four outputs in the output layer are juxtaposed to get a 128-bit hash value.

6 Parallel Keyed Hash Function Construction Based on Chaotic Maps

Although a variety of chaos-based hash functions have been proposed, none of them works efficiently in parallel computing environment. In [23, 24], we have proposed an algorithm for parallel keyed hash function construction, whose structure can ensure the uniform sensitivity of hash value to the message. By means of the mechanism of both changeable-parameter and self-synchronization, the keystream establishes a close relation with the algorithm key, the content and the order of each message block. The entire message is modulated into the chaotic iteration orbit, and the coarse-graining trajectory is extracted as the hash value. Theoretical analysis and computer simulation indicate that the proposed algorithm can satisfy the performance requirements of hash function. It is simple, efficient, practicable, and reliable. These properties make it a good choice for hash on parallel computing platform.

6.1 Algorithm Structure

The widely used hash functions are implemented in the Merkle-Damgard scheme, shown in Fig.1. The message length and padding are appended to the message to create an augmented message which can be evenly divided into $Y_0, Y_1, \cdots, Y_{L-1}$, blocks of b bits, where b is the size of the block to be processed by the compression function f. A compression function f is used repeatedly. In each round, the compression function f has two inputs: one is the n-bit output CV_{i-1} of the last round; the other is the b-bit input block Y_{i-1} of the current round. At the same time, the output of the current round is the n-bit CV_i, which is also used as the input of the next round. Before starting the iteration, the initial vector IV should be set. The output of the last round is the final hash value.

The processing of the current message unit can not start until the previous one has been processed. This restricts their applications on the platform, which supports parallel processing. Besides, the sensitivities of hash value to the message units at different positions of the message are uneven. In order to solve this problem, all the iterations have to be repeated several times as we have done in [12]. However, in this way, the efficiency has inevitably been reduced greatly. In this section, we propose a non-traditional algorithm for parallel keyed hash function construction, whose structure can ensure the uniform sensitivity of hash value to the message.

Let $N \geq 128$ be the bit-length of hash value. First of all, the original message M is padded such that its length is a multiple of N (The detail is described in the latter algorithm description). Then after padding, M is constituted by blocks, $M = (M_1, M_2, \cdots, M_s)$, each with N bits, and each block is indicated as $M_i = M_i^1 M_i^2 \cdots M_i^N$. The initial vector $H_0 = \{0\}_1^N$ is set.

The whole structure of the non-iterated type algorithm can be illustrated in Fig. 16 and described in Eq. (70).

$$\begin{cases} K_i = F(key, \ i, \ M_i), \ i = 1, 2, \cdots, s \\ H(M) = Hash \ \ Mixer(H_0, \ K_1, \ K_2, \cdots, K_s), \end{cases} \tag{70}$$
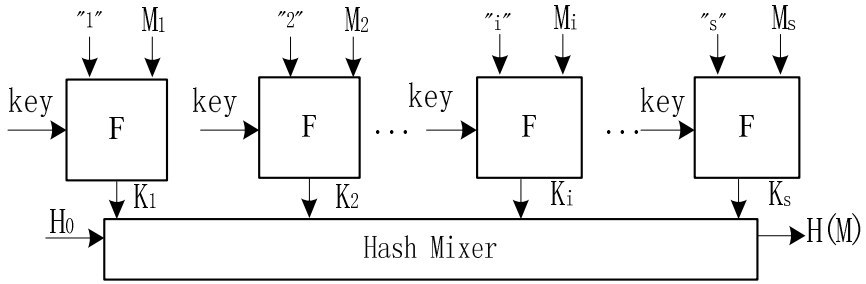
Fig. 16 Whole structure of the non-iterated type hash algorithm

where F is a round function, M_i is the ith message block, i is the order of each block, key is the secret key, K_i is the ith intermediate hash value, and $H(M)$ is the final hash value. There are two phases in this algorithm. The 1st phase is to generate the keystream K_i corresponding to each block M_i under the control of $(key, \ i, \ M_i)$. The 2nd phase is to use the hash mixer to obtain the final hash value $H(M)$ based on the generated $H_0, K_1, K_2, \cdots, K_s$. The Hash Mixer consists of two parts. The first part is to obtain the intermediate hash value H_s by $H_s = H_0 \oplus K_1 \oplus K_2 \cdots \oplus K_s$. The second part is to introduce the complicated non-linear connections among the different parts of the intermediate hash value H_s through some confusion approach.

6.2 Algorithm Description and Its Characteristics

In the proposed algorithm, Piecewise Linear Chaotic Map（PWLCM） and 4-dimensional Cat Chaotic Map（CATCM）will be utilized.

PWLCM is defined as:

$$x(k+1) = F_u(x(k)) = \begin{cases} x(k)/u, & 0 \le x(k) < u \\ (x(k)-u)/(0.5-u), & u \le x(k) < 0.5 \\ (1-x(k)-u)/(0.5-u), & 0.5 \le x(k) < 1-u \\ (1-x(k))/u, & 1-u \le x(k) \le 1 \end{cases} \tag{71}$$

where $x(k) \in [0, \ 1]$, $u \in (0, \ 0.5)$ are the iteration trajectory value and parameter of PWLCM, respectively. $\{x(k)\}$ is ergodic and uniformly distributed in $[0, 1]$, and the auto-correlation function of $\{x(k)\}$ is δ-like.

2-dimensional CATCM is defined as:

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = A \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} (\mathrm{mod}\,1) = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} (\mathrm{mod}\,1). \tag{72}$$

where $x_1(k), x_2(k)$ are real numbers in $[0, \ 1]$. The map is area-preserving since the determinant of its transformation matrix $|A|=1$.

By introducing two control parameters, a and b, the above 2-dimensional CATCM can be generalized as follows:

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = A \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} (\mathrm{mod}\,1) = \begin{bmatrix} 1 & a \\ b & 1+ab \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} (\mathrm{mod}\,1). \tag{73}$$

Similarly, 4-dimensional CATCM is defined as:

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \\ x_3(k+1) \\ x_4(k+1) \end{bmatrix} = A \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \\ x_m(k) \end{bmatrix} (\mathrm{mod}\,1), \tag{74}$$

where $A = A_{12} A_{13} A_{14} A_{23} A_{24} A_{34}$ is a 4×4 matrix and

$$A_{12} = \begin{bmatrix} 1 & a_{12} & 0 & 0 \\ b_{12} & 1+a_{12}b_{12} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad A_{13} = \begin{bmatrix} 1 & 0 & a_{13} & 0 \\ 0 & 1 & 0 & 0 \\ b_{13} & 0 & 1+a_{13}b_{13} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$\tag{75}$$

$$A_{23} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & a_{23} & 0 \\ 0 & b_{23} & 1+a_{23}b_{23} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \dots\dots A_{34} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & a_{3,4} \\ 0 & 0 & b_{3,4} & 1+a_{3,4}b_{3,4} \end{bmatrix}.$$

While reserving the mixing property and the sensitivity to initial conditions and parameters, the above 2-dimensional CATCM can be discretized as follows:

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = A\begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix}(\mathrm{mod}\,N) = \begin{bmatrix} 1 & a \\ b & 1+ab \end{bmatrix}\begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix}(\mathrm{mod}\,N). \tag{76}$$

where $a, b, x_1(k), x_2(k)$ are integers in $[0, \ N-1]$.

Similarly, 4-dimensional CATCM can be discretized in the same way.

The secret key of the algorithm includes: the initial condition $x(0) \in [0, \ 1]$ and initial parameter $u_0 \in (0, \ 0.5)$ of PWLCM; the initial condition and parameters (a, b, c, d) of 2-dimensional CATCM, which are integers in $(0, \ 16]$ and used to generate the parameters of 4-dimensional CATCM as follows:

$$\begin{bmatrix} y_1(k+1) \\ y_2(k+1) \end{bmatrix} = \begin{bmatrix} 1 & c \\ d & 1+cd \end{bmatrix}\begin{bmatrix} y_1(k) \\ y_2(k) \end{bmatrix}(\mathrm{mod}\,16), \tag{77}$$

where $y_1(0) = a, y_2(0) = b$. By iterating Eq. (77) for 6-time, the iteration values are assigned as the parameters of the 4-dimensional CATCM, $a_{ij}, b_{ij}, \ i = 1,2,3, \ j = 2,3,4, \ j > i$.

6.2.1 Algorithm Description

Let $N \geq 128$ be the bit-length of hash value without loss of generality. First of all, the original message M is padded such that its length is a multiple of N: let m be the length of the original message M; the padding bits $(100\cdots0)_2$ with length n (such that $(m+n)\,\mathrm{mod}\,N = N - 64, \ 1 \leq n \leq N$) are appended. The left 64-bit is used to denote the length of the original message M. If m is greater than 2^{64}, then $m \, \mathrm{mod} \, 2^{64}$. Then after padding, M is constituted by blocks with N bits, $M = (M_1, M_2, \cdots, M_s)$, and each block is indicated as $M_i = M_i^1 M_i^2 \cdots M_i^N$. The initial vector $H_0 = \{0\}_1^N$ is set.

The detailed process of the ith message block M_i ($i = 1, \ 2, \ \cdots, \ s$) in i^{th} round is described as follows (see Fig. 17).

M_i is re-divided into L units as Eq. (78), each unit with 8 bits (actually a character).

$$M_i = \underbrace{M_i^1 M_i^2 \cdots M_i^8}_{w_0}, \underbrace{M_i^9 M_i^{10} \cdots M_i^{16}}_{w_1}, \cdots \underbrace{M_i^{N-7} M_i^{N-6} \cdots M_i^N}_{w_L}. \tag{78}$$

1) While $N = 128$, M_i is actually divided into 16 units, $w_1, w_2, \cdots, w_{16}$, which is further assigned as 4 parallel groups for processing. Here, $w_1, w_2, \cdots, w_{16}$ should be pre-mapped into $wr_1, wr_2, \cdots wr_{16}$ in (0, 0.5) by means of linear transform $wr_i = \dfrac{u_0}{2} + \dfrac{w_i}{1024}$, $i = 1, 2, \cdots, 16$, where $u_0 \in (0, 0.5)$ is the initial parameter of PWLCM.

Group 1- wr_1, wr_2, wr_3, wr_4 : Start from the initial condition $x(0)$, and iterate PWLCM 10-time sequentially with the parameter sequence $\{wr_1, wr_2, wr_3, wr_4, wr_4, wr_3, wr_2, wr_1, u_0, u_0\}$. The final iteration value is set as $x_1(i)$;

Group 2- wr_5, wr_6, wr_7, wr_8 : Start from the initial condition $x(0)$, and iterate PWLCM 10-time sequentially with the parameter sequence $\{wr_5, wr_6, wr_7, wr_8, wr_8, wr_7, wr_6, wr_5, u_0, u_0\}$. The final iteration value is set as $x_2(i)$;

Group 3- $wr_9, wr_{10}, wr_{11}, wr_{12}$: Start from the initial condition $x(0)$, and iterate PWLCM 10-time sequentially with the parameter sequence $\{wr_9, wr_{10}, wr_{11}, wr_{12}, wr_{12}, wr_{11}, wr_{10}, wr_9, u_0, u_0\}$. The final iteration value is set as $x_3(i)$;

Group 4- $wr_{13}, wr_{14}, wr_{15}, wr_{16}$: Start from the initial condition $x(0)$, and iterate PWLCM 10-time sequentially with the parameter sequence $\{wr_{13}, wr_{14}, wr_{15}, wr_{16}, wr_{16}, wr_{15}, wr_{14}, wr_{13}, u_0, u_0\}$. The final iteration value is set as $x_4(i)$.

2) The order "i" of each message block is utilized to transform matrix A of 4-dimensional CATCM to $AA(i)$ in the following way:

Let $\alpha_j (j = 1, 2, 3, 4)$ denote the jth row vector of matrix A, then $AA(i) = (\alpha_1, \alpha_2 + \dfrac{i}{s} \times \alpha_1, \alpha_3, \alpha_4)$, namely, "$i/s$" multiple of the 1st row vector α_1 is added to the 2nd row vector α_2. For the property of determinant, $|AA(i)| = |A| = 1$, therefore $AA(i)$ is still area-preserving.

3) $x_1(i), x_2(i), x_3(i), x_4(i)$ obtained in Step 1) are set as the initial conditions, and then the transformed 4-dimensional CATCM is iterated twice sequentially.

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \\ x_3(k+1) \\ x_4(k+1) \end{bmatrix} = AA(i) \times \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_3(k) \\ x_4(k) \end{bmatrix} (\mathrm{mod}\,1) . \tag{79}$$

The final iteration values are set as $xx_1(i), xx_2(i), xx_3(i), xx_4(i)$. Transform them to the corresponding binary format, extract 32, 32, 32, 32 bits after the decimal point, respectively, and juxtapose them from left to right to get a 128-bit K_i, the keystream of the ith round.

After all the the keystreams K_i, $i = 1, 2, \cdots, , s$, are obtained, the detailed processing in the hash mixer is described as follows. First, the intermediate hash value H_s is generated by $H_s = H_0 \oplus K_1 \oplus K_2 \cdots \oplus K_s$. Then, the value H_s is re-divided into 16 integers as Eq. (80), and each integer has 8 bits.

$$H_s = \underbrace{H_s^1 H_s^2 \cdots H_s^8}_{b_0}, \underbrace{H_s^9 H_s^{10} \cdots H_s^{16}}_{b_1}, \cdots \underbrace{H_s^{121} H_s^{122} \cdots H_s^{128}}_{b_{16}} . \tag{80}$$

The 16 integers, $b_1, b_2, \cdots, b_{16}$, are first arranged in the order of left to right and top to bottom to form a 4×4 matrix B. Then compute

$$BB = A \times B (\mathrm{mod}\,256) , \tag{81}$$

where Matrix A is the 4-dimensional CATCM generated by the 2-dimensional CATCM with the secret key. If necessary, we can also make the 4-dimensional cat map A be generated independently by the 2-dimensional CATCM with the
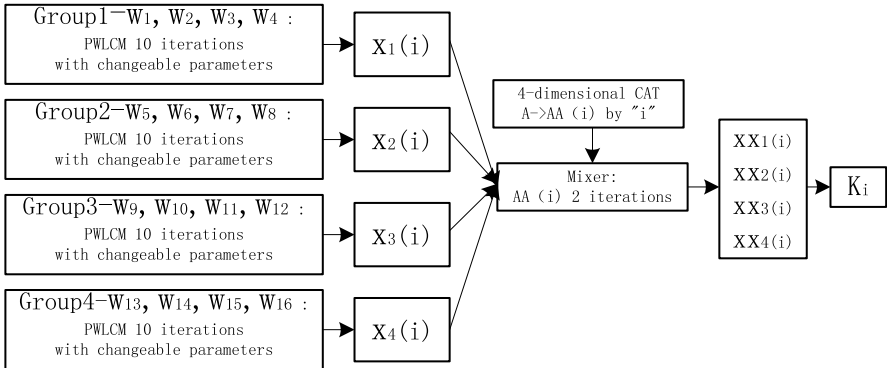

Fig. 17 The processing of the ith block M_i

different secret key. In the order of left to right and top to bottom, the 16 elements of the obtained Matrix BB are transformed to the corresponding binary format, respectively, and juxtaposed from left to right to get the 128-bit final hash value $H(M)$.

Note that the most important point is that the generation of the keystream K_i must be under the control of the corresponding $(key, \ M_i, \ i)$, namely, K_i must have a close relation with the algorithm key, the content and the order of current message block M_i, which can guarantee the security of the algorithm.

6.2.2 Characteristics of the Algorithm

The proposed algorithm has three remarkable characteristics: the parallel mode, the mechanism of both changeable-parameter and self-synchronization, and the uniform sensitivity of hash value to the message.

1) Parallel mode
The parallel mode is embodied in two aspects: on the one hand, from the whole structural point of view, after the message M is divided into blocks $M_1, M_2, \cdots, M_s$, the processes among blocks can be performed in a parallel mode (see Fig.16 and Eq. (70)). At the same time, the confusion operation by 4-dimensional CATCM can also be performed in a parallel mode. On the other hand, within each message block, the 4-group iterations of PWLCM can also be processed in a parallel mode, and 4-dimensional CATCM is utilized to be a mixer (see Fig.17).

2) Changeable-parameter and self-synchronizing keystream
During the hashing process of each message block M_i ($i = 1, \ 2, \ \cdots, \ ,s$), the message unit at different positions will cause the parameter of chaotic maps to change dynamically. On the one hand, perturbation is introduced in a simple way to avoid the dynamical degradation of chaos; on the other hand, self-synchronizing stream is realized, which ensures that the generated keystream K_i is closely related to the algorithm key, the content and order " i "of each message block M_i ($i = 1, \ 2, \ \cdots, \ ,s$). The mechanism of both changeable-parameter and self-synchronization makes each bit of the final hash value have a close relation to all the bits of the message. It lays the solid foundation for the security of the proposed algorithm: different message block M_i leads to different keystream K_i; and even the same message block M_i, when the order " i " changes, the corresponding keystream K_i will be also totally different.

3) Uniform sensitivity of hash value to the message.

Through the analysis on recurrent formula Eq.(70) of the final hash value, we can find out that the effect of each message block M_i on the final hash value $H(M)$ is equivalent, any change in message block M_i ($i = 1, 2, \cdots, , s$) will lead to the consequential change in the corresponding K_i ($i = 1, 2, \cdots, , s$) and then the final hash value $H(M)$. This thoroughly overcomes the flaw in some former hash algorithms that the sensitivities of hash value to the message units at different positions of the message are wavy.

6.3 Performance Analysis

In [23, 24], series of strict performance analysis and simulations are carried out, including distribution of hash value, sensitivity of hash value to the message and secret key, statistic analysis of diffusion and confusion, analysis of collision resistance, resistance against forgery attack, security of key, analysis of speed, and implementation and flexibility, etc.

Most of the performance analysis methods are similar to that given in Section 2.5. Due to page limitation, in this section, we only emphasize its unique performances and omit the rest performances.

6.3.1 Security of Key

The security of key includes two aspects. One is the key non-recovery property. It must be computationally infeasible to recover key, given one or more message-MAC pairs (M_j, $H(M_j)$). The other is the size of the key space, which characterizes the capability of resisting brute-force attack.

In the hashing process of our algorithm, the sensitivity to tiny changes in initial conditions and parameters, and the mechanism of both changeable-parameter and self-synchronization are fully utilized. It makes the algorithm having strong one-way property, and there exist complicated nonlinear and sensitive dependence among message, hash value and secret key. Therefore, we believe that it is immune from key recovery attack.

To investigate the key space size, the following evaluations are performed. Let the tiny change of the initial value $x(0)$ of PWLCM be larger than 10^{-16}, for example, $x(0)$ is changed from 0.232323 to 0.2323230000000001, the corresponding changed bit number of hash value obtained is around 64. Similarly, let the tiny change of the initial parameters u_0 of PWLCM be larger than 10^{-16}, for example, u_0 is changed from 0.454445 to 0.4544450000000001, the corresponding changed bit number of hash value obtained is also around 64. However, if the tiny changes of $x(0)$ and u_0 are set as 10^{-17}, no corresponding bit of the hash value changes. Therefore, the sensitivity to $x(0)$ and u_0 are both considered as 10^{-16}. As for the

other part key (a,b,c,d), the parameters and initial values of the 2-dimensional CATCM, when the change of them is 1, the corresponding changed bit number of hash value is also around 64. Considering the value ranges of components, $x(0) \in [0, \ 1]$, $u_0 \in (0, \ 0.5)$, $(a,b,c,d) \in (0, \ 16]$, it can be derived that the size of the key space is larger than 2^{121}. Although some weak key combinations $(x(0), \ u_0)$ of PWLCM and some possible combinations (a,b,c,d) of the 2-dimensional CATCM may be sacrificed, a conclusion can still be drawn that the key space is large enough to resist the brute-force attack.

6.3.2 Analysis of Speed

For speed comparison among different algorithms, the numbers of the required multiplicative operations for each ASCII character (8-bit) message during the hash process are listed in Table 4. Although the confusion approach by 4-dimensional CATCM is added after the XOR operation, the effect on the entire efficiency of the algorithm is very slight, especially with the increasing of the message block number s.

The calculation of our proposed algorithm is as follows:

Each message block M_i has 16-character message units, which are divided into four groups. Each group has four characters. First, in each group, linear transform needs 4-time multiplicative operations, and 10-time iterations of PWLCM need 10-time multiplicative operations. Therefore, the four groups need 56-time multiplicative operations. Second, to transform A to $AA(i)$ of 4-dimensional CATCM by the order "i" of each message block needs 8-time multiplicative operations. Finally, the consecutive two-iteration of the transformed 4-dimensional CATCM needs 32-time multiplicative operations. In summary, the required multiplicative operation for each character in our proposed algorithm is (56+8+32)/16=6. On the other hand, the added confusion approach by 4-dimensional CATCM needs extra 64-time multiplicative operations for the operation between two 4×4 matrixes. However, the 64-time multiplicative operations are undertaken by the total number of message block, and each block has 16- character message units. Therefore, only $4/s^*$ has to be added, which is very slight.

Table 4 Required multiplicative operation for each character of algorithms

	Ref. [12]	Ref. [11]	Ref. [25]	This algorithm
Multiplication	6	11.7	32（by software）/8 (by hardware）	$6+4/s^*$

* s represents the number of message block with the length of 128-bit.

Furthermore, since the proposed algorithm can support parallel mode, not only the parallel processes among message blocks, but also 4-group parallel iterations of PWLCM within each message block, its efficiency is predominant, especially compared with other hash algorithm in sequential mode.

7 Combined Chaotic Cryptographic and Hashing Scheme

In some security applications, both encryption/decryption and hashing are required to perform simultaneously. To fulfill this kind of requirement, we proposed a combined chaotic cryptographic and hashing algorithm based on chaos [26]. This algorithm originates from a combined chaotic cryptographic and hashing scheme based on dynamic look-up table proposed by Wong in [14], which can also be regarded as one of the variants of the cryptosystem based on the ergodicity property of chaotic systems proposed by Baptista [27] in 1998. However, shortly after Wong's scheme was proposed, G. Alvarez *et al* [28] pointed out the security problem of this scheme and described a keystream attack on it. In [26], we have analyzed the cause of vulnerability of the original dynamic look-up table based chaotic encryption scheme in detail, and then have also proposed the corresponding enhancement measures. Theoretical analysis and computer simulation indicate that the modified scheme is more secure than the original one. At the same time, it holds the combined merit of the original scheme. Besides, other idea for improvement is also presented to improve the efficiency and the ratio of ciphertext/plaintext of the combined algorithm.

7.1 Wong's Algorithm and Its Security Analysis

7.1.1 Description of Wong's Algorithm [14]

The chaotic map chosen in the scheme is the simple Logistic map governed by the following equation:

$$X_{n+1} = bX_n(1 - X_n), \tag{82}$$

where $X_n \in [0, 1]$ and the value of the coefficient b should ensure the map is running in a chaotic regime. A paragraph of plaintexts is divided into a number of message blocks. There is only one character in each block and the plaintext is thus the ASCII code of the character.

In the initialization phase, the initial value X_0 and the coefficient b should be chosen as the secret key. Moreover, an initial look-up table should be set at the same time. In Fig.18, there are 256 entries numbered from 0-255 in the look-up table, and each entry contains an ASCII character. If the range $[X_{min}, X_{max}]$ is chosen as [0.2000, 0.8000], the interval width is $\varepsilon = (X_{max}-X_{min})/256=0.0023$. Each interval is associated with a particular unit of the look-up table. For instance, there are

Fig. 18 Look-up Table

mappings of interval [0.2000, 0.2023] to the first unit, interval $[X_{min}+i\varepsilon, X_{min}+(i+1)\varepsilon]$ to the ith unit, and so on.

For the encryption of the ith message block in the encryption phase, we begin to iterate the logistic map with the current X value until the orbit first falls into the region corresponding to the ASCII code of this message block. The current number of iterations will be regarded as the ciphertext. To enhance the anti-attack capability of the scheme, the iteration will continue if the current number of iterations is smaller than a pre-defined minimum number of iterations T_{min}. This rule can be straightforwardly applied to the remaining message blocks in the plaintext. Before encrypting the next message block, we have to update the look-up table dynamically by exchanging the ith entry e_i with another entry e_j. The number j is determined by the following formula:

$$v = \left\lfloor \frac{X - X_{min}}{X_{max} - X_{min}} \times N \right\rfloor, \tag{83}$$

$$j = (i + v) \bmod N, \tag{84}$$

where $N=256$ is the entry number in the scheme. We have to perform a modulus operation so that the value of j is always smaller than N.

In the chaotic cryptosystem described above, the final look-up table will be obtained after the whole paragraph of plaintexts is encrypted. This table, while completely determined by the content of the plaintexts, is much shorter than the plaintext, and is totally different for different messages. According to the definition of the one-way hash function, the algorithm is a valid hash function. Therefore, it has essentially completed the encryption and hashing in a combined manner.

To ensure good collision resistance of the hash function, the swapping of multiple p ($p \geq 1$) pairs of entries in the look-up table is allowed during the

encryption of each message block, and the multiple r ($r \geq 1$) runs of continuous encryption on the whole message is introduced. Starting from the current entry number i, the p–pair swapping rule is as follows:

$$i \leftrightarrow ((i+v) \bmod N), ((i+v+1) \bmod N) \leftrightarrow ((i+2v+1) \bmod N), ((i+2v+2) \bmod N) \leftrightarrow ((i+3v+2) \bmod N),$$

$$\cdots, ((i+(p-1)v+p-1) \bmod N) \leftrightarrow ((i+pv+p-1) \bmod N).$$

$$(85)$$

Once the message has been encrypted, the whole process is repeated (r-1) times, $r \geq 1$. However, only the ciphertext obtained from the first run need to be transmitted to the receiver, the remaining (r-1) runs are for only the good performance of the hash function.

As the ciphertext is the number of iterations required, we can start the decryption phase by iterating the corresponding Logistic map with X_0, b and the initial look-up table for the required number of iterations. Based on the mapping of the interval to the entry in the look-up table, the first block of plaintext can be obtained. Before decrypting the next ciphertext block, we have to update the look-up table in the same way as in the encryption process. This rule can be applied to the remaining ciphertext blocks in turn. Finally, we can decrypt all the ciphertext and get the final look-up table, too. The final look-up table is the hash value and so the scheme has also completed the decryption and hashing in a combined way.

7.1.2 Attack [28]

1）**Attack On the Dynamic Look-up Table (hash function):**
The attack on the dynamic look-up table lies on the fact that the attacker can easily predict the new positions of the characters even without the exact knowledge of the value of current orbit X. For instance, if there are only two characters, $S_2 = \{s_1, s_2\}$, when the orbit falls into the first interval $[0.2, 0.5)$, $0.2 \leq x < 0.5$, then Eq. (83) becomes:

$$0 \leq v = \left\lfloor \frac{X - X_{\min}}{X_{\max} - X_{\min}} \times N \right\rfloor = \left\lfloor \frac{x - 0.2}{0.6} \times 2 \right\rfloor < 1,$$

thus $v=0$ and Eq. (84) is equivalent to: $j = i \bmod 2$.

When the orbit falls into the second interval $[0.5, 0.8)$, $0.5 \leq x < 0.8$, then $v=1$ and Eq. (84) becomes: $j = (i+1) \bmod 2$.

Similarly, if there are four characters, $S_4 = \{s_1, s_2, s_3, s_4\}$, then $v = 0$ when the orbit falls into $[0.2, 0.35)$, $v = 1$ when the orbit falls into $[0.35, 0.5)$, $v = 2$ when the orbit falls into $[0.5, 0.65)$, and $v = 3$ when the orbit falls into $[0.65, 0.8)$.

The generalized relationship for higher order characters exists as follows:

$$k_i = s_1 \leftrightarrow v = 0, k_i = s_2 \leftrightarrow v = 2, \cdots, k_i = s_n \leftrightarrow v = n-1. \tag{86}$$

In conclusion, the updated table depends solely on the plaintext, and not the key (X_0 and b). When encrypting the same plaintext using different keys, the same updating evolution will take place in the look-up table. Regardless of the key used, the same message will always yield the same hash value, which is to the advantage of the attacker.

2）**Attack on the encryption**

The scheme is essentially a stream cipher. When the secret key (X_0 and b) is given, a keystream $k = k_1 k_2 \cdots$ can be generated using (82). The plaintext string $p = p_1 p_2 \cdots$ is encrypted according to the rule:

$$c = e_{k1}(p_1) e_{k2}(p_2) \cdots = c_1 c_2 \cdots. \tag{87}$$

The ciphertext string c can be easily decrypted after getting the keystream $k = k_1 k_2 \cdots$ based on the secret key and undoing the encryption operation e_{ki}.

The focus of the attack described in [28] is to recover the keystream $k = k_1 k_2 \cdots$. G. Alvarez *et al* pointed out that the cipher is immune from Chosen ciphertext attack and Ciphertext only attack, but Chosen plaintext attack and Known plaintext attack can totally or partially break the cryptosystem. Similar to the attack on the look-up table, the cause of vulnerability is that the look-up table evolution can be easily predicted without the exact knowledge of the value of current orbit X. Details of the attack can be found in [28].

7.2 Modified Scheme and Its Performance Analysis

7.2.1 Description of the Modified Scheme

The following modified scheme is proposed to prevent the cryptanalysis based on the existing vulnerability.

Logistic map (82) is still chosen to generate the chaotic orbit, where the secret key K is chosen as X_0=0.232323, b_0=3.99995 and N=256. In the initialization phase, the same preparation will be performed as in the original algorithm.

For the encryption of the ith message block in the encryption phase, we begin to iterate the logistic map with the current X_{i-1} value until the orbit first falls into the region corresponding to the ASCII code of this message block. The current number of iterations will be regarded as the ciphertext. To enhance the anti-attack capability of the scheme, the iteration will continue if the current number of iterations is smaller than a pre-defined minimum number of iterations T_{min}. Before encrypting the next message block, we still have to update the look-up table dynamically by

exchanging the ith entry e_i with another entry e_j. However, the formula for finding j is changed as follows:

First, the 2nd, 3rd and 4th digits of the current value of X_i after the decimal point are extracted to construct a new decimal integer Y.

Then

$$v = Y \bmod 256 , \tag{88}$$

$$j = (i + v) \bmod 256. \tag{89}$$

According to the basic property of *mod* operation

$$(a(\bmod n) + b(\bmod n)) \bmod n = (a + b) \bmod n. \tag{90}$$

These two steps can be combined as one step:

$$j = (i + Y) \bmod 256. \tag{91}$$

We have to perform a modulus operation so that the value of j is always smaller than 256.

Similar to the original scheme in ensuring good collision resistance of the hash function, the modified scheme is generalized by allowing the swapping of p pairs of entries in the look-up table during the encryption of each message block, and by allowing r runs of encryption on the whole message continuously, where $r \geq 1, p \geq 1$:

$$i \leftrightarrow ((i + Y) \bmod 256), ((i + Y + 1) \bmod 256) \leftrightarrow ((i + 2Y + 1) \bmod 256)$$
$$\cdots, ((i + (p-1)Y + p - 1) \bmod 256) \leftrightarrow ((i + pY + p - 1) \bmod 256). \tag{92}$$

Only the ciphertext obtained from the first run need to be transmitted to the receiver, the remaining $(r\text{-}1)$ runs are only for the good performance of the hash function.

In this modified scheme, the final look-up table is also the hash value of the plaintext. Therefore, we can draw a conclusion that the modified scheme has kept the remarkable characteristic to perform encryption and hashing in a combined manner.

In the decryption phase, only the formula for updating the look-up table is changed as in the encryption phase. The other parts are just the same as the original scheme. As a result, the modified scheme has also completed decryption and hashing simultaneously.

7.2.2 Performance Analysis

1) Resistance analysis of the dynamic look-up table (hash function)
The attack on the dynamic look-up table of the original scheme lies on the fact that the attacker can easily predict the new positions of the characters even without the exact knowledge of the value of current orbit X. In our modified scheme, it is Y

that decides the look-up table evolution and it is obtained by extracting the 2nd, 3rd and 4th digits after the decimal point of the current X value. This improvement establishes a relationship between the exchanging entries with the current orbit value, which totally avoids the cryptanalysis described above. Since the current orbit value has close relation with the secret key (X_0 and b), the updating evolution relies not only on the plaintext but also on the secret key.

2) Resistance analysis of the encryption

Similarly, the reason why Chosen plaintext attack and Known plaintext attack can totally or partially break the original cryptosystem is that the look-up table evolution can be easily predicted without the exact knowledge of the value of current orbit X. Our improvement has established a close link between the exchanging entries and the current orbit value. It gets rid of the cryptanalysis described in [28] and so similar attacks become impossible.

3) Security of Key

In the modified scheme, $X_0 \in [0, 1]$ and the coefficient b_0 are chosen as the secret key. The key space is huge enough to resist any exhaustive key search. Moreover, it is absolutely impossible to inversely deduce the value of X_0, b_0 from the iteration values due to the high sensitivity to tiny changes in initial conditions and parameters.

4) Encryption and hashing experiments

Under two different conditions, the modified scheme is used to perform comparative experiments on a selected plaintext sample: "*Key establishment is any process whereby a shared secret key becomes available to two and more parties, for subsequent security communications. A key agreement protocol is a key establishment technique in which a shared secret key is derived by two or more parties as a function of information contributed by, or associated with, each of these, ideally such that no party can predetermine the resulting value. As a ubiquitous phenomenon in nature, chaos is a kind of deterministic random-like process generated by nonlinear dynamic systems. The properties of chaos includes: sensitivity to tiny changes in initial conditions and parameters, random-like behavior, ergodicity, unstable periodic orbits with long periods and desired diffusion and confusion properties, etc. Furthermore, the iteration process is one-way. Unique merits of chaos bring much promise of application in the information security field. Basically, there are two general ways to design chaos-based ciphers: using chaotic system to*".

Condition 1: The secret key is set as X_0=0.1777、 b_0=3.9999995, run-time r=2, swapping pairs p=10, minimum number of iterations T_{min}=200.

Condition 2: Only change the secret key b_0=3.9999995 to b_0=3.9999996.

(a)   (b)
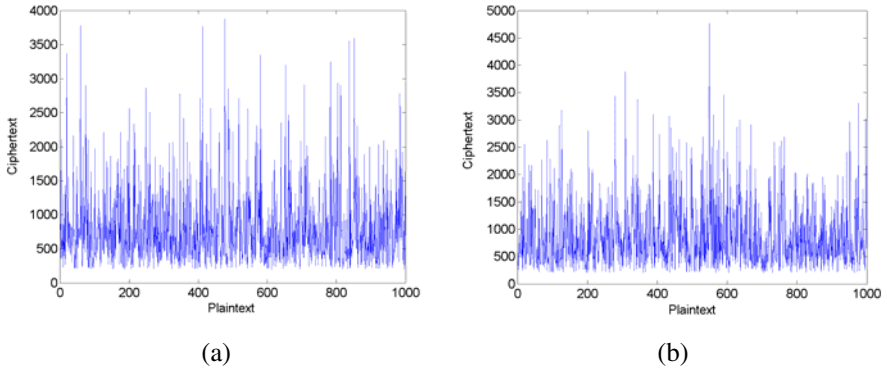
Fig. 19 Plaintext-Ciphertext from Two Conditios

(a)   (b)
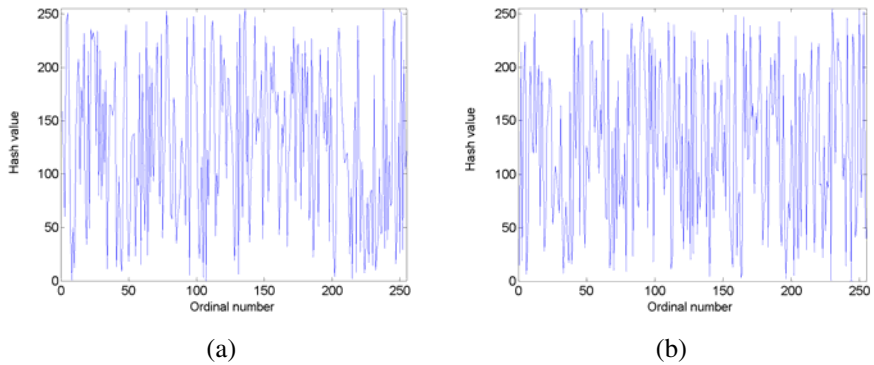
Fig. 20 Hash values from Two Conditions

The modified scheme can perform both encryption and hashing smoothly under the two different conditions. Fig. 19 is a plot of the Plaintext-Ciphertext result, where (a) is obtained from Condition 1 while (b) is from Condition 2. Fig. 20 is a plot of the hash values, where the horizontal axis represents the ordinal number 0-255 of the look-up table and the vertical axis represents the ASCII value of the character in the corresponding entry. Fig. 20(a) is obtained from Condition 1 while Fig. 20(b) is from Condition 2. Obviously, when the secret key K has tiny changes, the ciphertexts and hash values of the two experiments are totally different. In the hash values, 254 entries in the look-up table are different within all the 256 entries, which proves that the modified scheme is very sensitive to the secret key.

5) Analysis of collision resistance and birthday attacks resistance
Collision and birthday attacks are similar in idea. They are essentially a probability problem that two random input data are found to give the same hash output. In our

modified scheme, the look-up table evolution has close relation with the exact value of the current orbit X, and the swapping of multiple pairs of entries in the look-up table within multiple runs of encryption are allowed. These operations will ensure that each bit of the final hash value is related to all the bits of the message. Even a single bit change in message or key will be diffused and results in great changes in the final hash value.

Similar to [14], we have performed the following tests for the quantitative analysis on the collision resistance: first, the hash value for a paragraph of randomly chosen message is generated and stored in ASCII format. Then a bit in the message is selected randomly and toggled. A new hash value is then generated and stored in ASCII format. The two hash values are compared and the number of ASCII characters with the same value at the same location is counted. Moreover, the absolute difference of the two hash values is calculated using the formula $d = \sum_{i=1}^{N} |t(e_i) - t(e_i')|$, where e_i and e_i' be the ith ASCII character of the original and the new hash value, respectively, and the function $t(*)$ converts the entries to their equivalent decimal values. This kind of collision test is performed 2000 times, with the secret key $X_0 = 0.1777, b_0 = 3.9999995$, and run-time $r = 2$, swapping pairs $p = 10$. The maximum, mean, minimum values of d and mean/character are listed in Table. 5. A plot of the distribution of the number of ASCII characters with the same value at the same location in the hash is shown in Fig. 21. Notice that the maximum number of equal characters is only 5 and the collision level is very low.
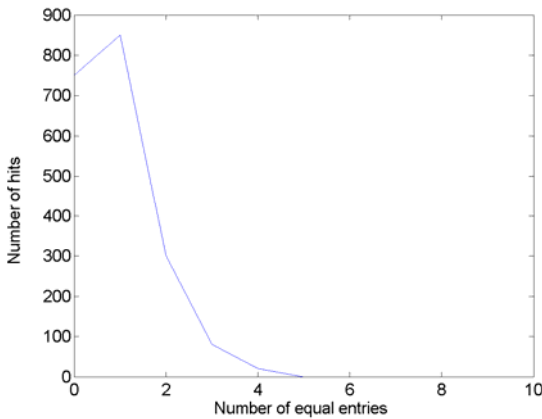

Fig. 21 Distribution of the number of ASCII characters with the same value at the same location in the hash

Table 5 Absolute differences of two hash values

Absolute difference d	Maximum	Minimum	Mean	Mean/character
	23944	18666	21946	85.7266

6) Efficiency Analysis

The difference between the Wong algorithm and the modified one lies in the way to find j when updating the look-up table dynamically by exchanging the ith entry e_i with another entry e_j.

In Wong algorithm, the number j is determined by the following formula:

$$v = \left\lfloor \frac{X - X_{\min}}{X_{\max} - X_{\min}} \times N \right\rfloor, \tag{83}$$

$$j = (i + v) \bmod N. \tag{84}$$

In the modified algorithm, the 2nd, 3rd and 4th digits of the current value of X_i after the decimal point are extracted to construct a new decimal integer Y, and then Y is used to decide j:

$$v = Y \bmod 256, \tag{88}$$

$$j = (i + v) \bmod 256. \tag{89}$$

These two steps can be combined as one step:

$$j = (i + Y) \bmod 256. \tag{91}$$

Hence, the efficiency of the modified algorithm is similar to the original one.

The actual efficiency depends on the values of p and r. They may be set flexibly to fulfill the requirement. When $r = 2, p = 10$, the encryption and decryption speed achieved is 6.75KB/sec and 12.42KB/sec by implementing the modified scheme using C++ programming language running on a personal computer with Pentium IV 1.7GHz processor and 256 MB RAM.

If we compare the encryption/decryption speed or hashing speed separately between the proposed algorithm and some traditional ones, it seems that the proposed algorithm is slower. However, the remarkable characteristic of the proposed algorithm is that it performs encryption/decryption and hashing in a combined way. There are lots of applications of cryptography where not only the encryption and decryption of the plaintext are required, but also the hashing operation. We may apply the above algorithm to these occasions so that the efficiency can be improved.

7.3 Other Ideas for Improvement

Both Wong algorithm and the modified one have two major drawbacks: The first one is their low efficiency, which inevitably limits their actual application. Second, as their ratio of ciphertext /plaintext is 2, it will increase transmission burden and shorten the unicity distance, which is harmful to the algorithm security from the perspective of information theory.

To overcome the above drawbacks, a novel combined cryptographic and hash algorithm based on chaotic control character is proposed in [29]. The control character is generated by chaotic iteration. The pre-process of plaintext is performed in accordance with the control character, and then the corresponding index value of the item in the look-up index table is taken as its ciphertext. At the same time, the chaotic trajectory is changed continuously according to the control character, which can avoid the dynamical degradation of chaos to some extent. Besides, the look-up index table is updated by utilizing the control character continuously, and the index item of the final look-up index table can be considered as the hash value of the whole paragraph of plaintext. Therefore, the proposed algorithm still keeps the remarkable characteristic to perform encryption and hash in a combined manner. Compared with Wong's and our modified algorithms, the proposed one has decreased the ratio of ciphertext/ plaintext to 1, and improved the efficiency greatly.

7.3.1 Description of the New Algorithm

The new algorithm no longer divides the chaotic attractor into ranges and assigns the association between the ranges and the plaintext blocks. A notable characteristic of its structure is to introduce a look-up index table. Fig. 22 is the initial look-up index table. The right part of the table represents the plaintext space. When the length of plaintext block is set as *nbit*, the corresponding plaintext space is $M = \{s_0, s_1, \cdots, s_{2^n - 1}\}$, where $s_i = i$. The left part of the table represents the index space $T^n = \{T_0, T_1, \cdots, T_{2^n - 1}\}$, where $T_i = i$. In nature, the look-up index table is a bijective map $\varphi : T^n \to M$ between the plaintext space and the index space.

The chaotic map chosen in the algorithm is Piecewise Linear Chaotic Map （PWLCM）governed by the following equation:

$$F : x_{i-1} = \begin{cases} x_i / b & x_i \leq b \\ (1 - x_i) / (1 - b) & x_i > b \end{cases}.$$

(93)

1) Encryption and decryption

The secret key of the algorithm is $Key = \{x_0, \ b\}$, where $x_0, \ b$ are the initial value and control parameter of PWLCM, respectively. The plaintext is composed of
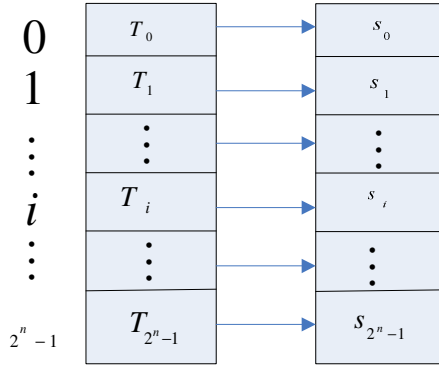
Fig. 22 The look-up index table

r blocks, each with $nbit$, namely $m = m_1, m_2, \cdots, m_r$. The detailed encryption process is as follows (see Fig.23(a)):

(1) The initialization of the look-up index table. First set $\varphi_0(T_i) = s_i$, $i = 0, 1, 2, \cdots, 2^n - 1$, where $s_i \in M$, and iterate PWLCM N-time ($N = 2^n$ is the item number of the table) with the initial value x_0 to obtain a chaotic sequence $\{x_0, x_1, \cdots, x_i, \cdots, x_{N-1}\}$, and then utilize the sequence to permute the look-up index table into a new one φ_1 in the following way: Turn the chaotic sequence $\{x_0, x_1, \cdots, x_i, \cdots, x_{N-1}\}$ into the corresponding integer sequence $\{y_0, y_1, \cdots, y_i, \cdots, y_{N-1}\}$ by round operation, then exchange the ith index item with the corresponding y_i^{th} one in turn.

(2) Set $i = 1$, $x_0^1 = x_0$, and $C_0 = g$, where $g(0 \le g \le 2^n, g \in Z^+)$ is a constant and can be chosen arbitrarily.

(3) Iterate PWLCM $it = 10$ times with the initial value x_0^i, and obtain the chaotic orbit value $\tilde{x} = F^{it}(x_0^i)$ (see Section 2.3) ;

(4) Calculate $D_i = \lfloor \tilde{x} \times 2^n \rfloor$, transform it to the corresponding binary format $D_i = (d_1 d_2 \cdots d_n)_2$, and extract its odd bits and even bits to obtain the particular control character P_i and Q_i, respectively, namely $P_i = (d_1 d_3 \cdots d_{2n-1})_2$ and $Q_i = (d_2 d_4 \cdots d_{2n})_2$, where $d_j \in \{0, 1\}$, $j = 1, 2, \cdots 2\alpha$, $\alpha = \dfrac{n}{2}$;

(5) If $P_i = Q_i$, set $x_0^i = \tilde{x}$ and go to (3); if $P_i < Q_i$, set $\tilde{m} = m_i \oplus C_{i-1}$; if $P_i > Q_i$, set $\tilde{m} = m_i \oplus D_i$;

(6) Search the look-up index table by $\tilde{m}$, and set the index value of the corresponding item $\tilde{m}$ as the ciphertext C_i, namely $T_{C_i} = \phi_i^{-1}(\tilde{m})$;

(7) If $i = r$, iterate PWLCM with the initial value x_0^i to obtain a chaotic sequence with the length of N, permute the look-up index table in a similar way in (1), and use the index item of the final look-up index table as the hash value of the whole paragraph of plaintext, then the algorithm ends. Otherwise, update the look-up index table to φ_{i+1} (see Section 2.2), set $i = i+1$, $x_0^i = \left(x + \tilde{m}/2^n\right) - \left\lfloor x + \tilde{m}/2^n \right\rfloor$, and go to (3).

The detailed decryption process is as follows (see Fig.23(b)):

(1)-(4) are the same as the counterparts in the encryption process;
(5) Set $\tilde{m} = \varphi_i(T_{C_i})$;

(6) If $P_i = Q_i$, set $x_0^i = \tilde{x}$ and go to (3); if $P_i < Q_i$, the ith plaintext block is $m_i = \tilde{m} \oplus C_{i-1}$; if $P_i > Q_i$, the ith plaintext block is $m_i = \tilde{m} \oplus D_i$;

(7) This step is the same as (7) in the encryption process. When the algorithm ends, the index item of the final look-up index table is considered as the corresponding hash value.

2) Obtain the hash value (Update the look-up index table)

In order to obtain the hash value, two kinds of updating processes of the look-up index table are necessary. The first kind of updating is within the processing of each block of the plaintext. It is essentially transforming the old φ_i to the new φ_{i+1}, where φ_{i+1}, φ_i are both bijective maps between the plaintext space and the index space. The detailed updating process is shown in Fig. 24. During the encryption of each plaintext block, starting from T_{C_i} (the index item corresponding to the current plaintext block in the look-up index table), the Q_i–pair swapping rule with the interval P_i is as follows:

$$T_{C_i \bmod N} \leftrightarrow T_{j \bmod N} \quad , \quad T_{(C_i+P_i) \bmod N} \leftrightarrow T_{(j+P_i) \bmod N} \quad , \quad \cdots \quad ,$$
$$T_{(C_i+Q_i \times P_i) \bmod N} \leftrightarrow T_{(j+Q_i \times P_i) \bmod N} \quad , \quad \text{where} \quad N = 2^n, \quad n \text{ is the length of each}$$

Fig. 23 Encryption/ decryption process
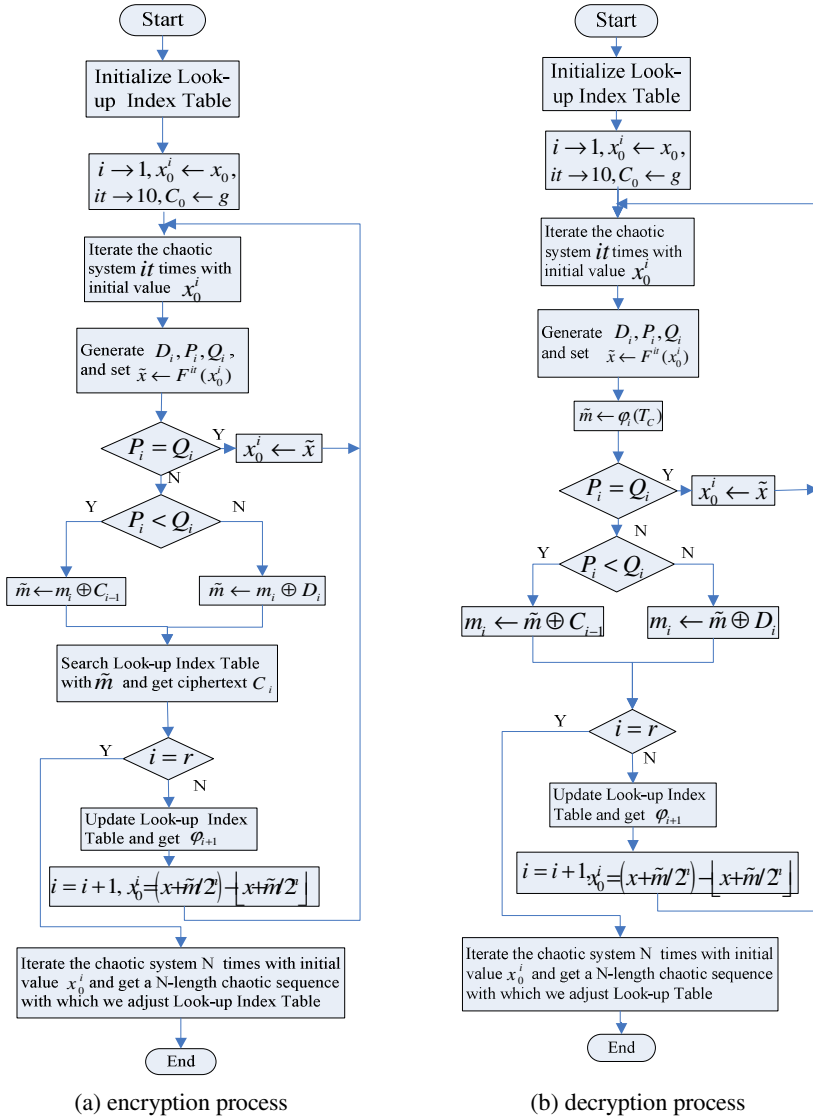
plaintext block, and $j = \begin{cases} C_i + \tilde{m} & \tilde{m} \neq 0 \\ C_i + 1 & \tilde{m} = 0 \end{cases}$. Assume that $\varphi_i(T_{C_i}) = S_{C_i}$, then after

updating, $\varphi_{i+1}(T_{C_i}) = S_j$.

The second kind of updating is run after all the plaintext blocks have been processed. Iterate PWLCM N -time ($N = 2^n$ is the item number of the table) with

the current initial value x_0^i to obtain a chaotic sequence $\{x_0, x_1, \cdots, x_i, \cdots, x_{N-1}\}$, turn the chaotic sequence $\{x_0, x_1, \cdots, x_i, \cdots, x_{N-1}\}$ into the corresponding integer sequence $\{y_0, y_1, \cdots, y_i, \cdots, y_{N-1}\}$ by round operation, and then exchange the ith index item with the corresponding y_i^{th} one in turn.

The updating process of the look-up index table is closely related to the algorithm key and the plaintext, which on the one hand ensures the algorithm security of the encryption/decryption, on the other hand results in the fact that the obtain of the look-up index table is determined by the content of the plaintext and different plaintext leads to different final look-up index table. According to the definition of one-way hash function, the index item of the final look-up index table can be used as the hash value of the whole paragraph of plaintext. Therefore, it is certainly a combined cryptographic and hash algorithm which keeps the remarkable characteristic of Wong's and our former algorithms to perform encryption/decryption and hash in a combined manner.
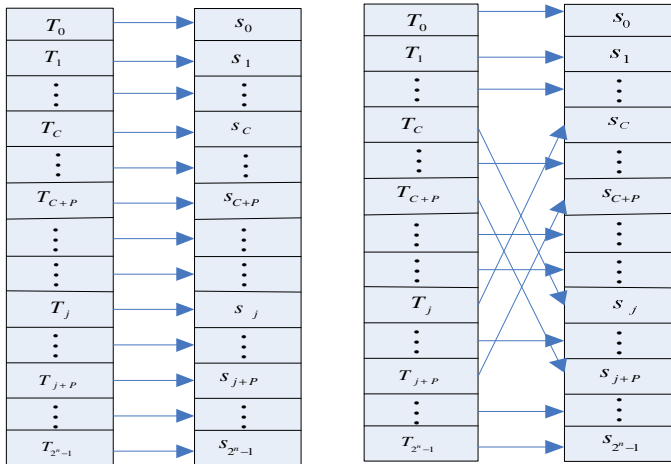
Fig. 24 Update the look-up index table

7.3.2 Performance Analysis

In [29], series of strict performance analysis and simulations are carried out. Due to page limitation, in this section, we only emphasize its unique performance and omit the rest performance.

1) The ratio of ciphertext/plaintext
In this algorithm, the control character is generated by chaotic iteration. The pre-process of plaintext is performed with the control character, and then the index value of the item in the look-up index table corresponding to the "pre-processed

plaintext" is regarded as its ciphertext. Obviously, the ratio of ciphertext/plaintext is constantly equal to 1, which is favorable for decreasing transmission burden and lengthening the unicity distance.

2) Analysis of efficiency

For speed comparison among different algorithms, the numbers of the required multiplicative operations for each ASCII character (8-bit) message during the hash process are obtained. Since each multiplicative operation consumes much more time than each additive operation, this kind of comparison is objective, in spite of different implementing platforms.

Compared with Wong's and our modified algorithms [14, 26], the proposed one keeps the remarkable characteristic of performing encryption/decryption and hash in a combined manner; moreover, it improves the efficiency greatly. The efficiencies of Wong's and our former algorithms depend on the run-time and the item-number of the look-up table. If the two parameters are set as 2 and 10 respectively, and the minimum number of iteration is set as $T_{\min} = 200$, then over 800-time multiplicative operations are needed for encrypting an 8-bit plaintext block. However, if our proposed algorithm are utilized to process a paragraph of plaintext composed of r 8-bit blocks, only $(10 + (N*2)/r)$-time (N is the item number of the table), namely slightly more than 10-time multiplicative operations are needed for encrypting a $n = 8$-bit plaintext block. The efficiency is sharply improved about 80-time.

8 Some Instructions on Chaos-Based Hash Function Construction

Although there are already many chaos-based hash functions, the systematic approach to the design and security evaluation of chaos-based hash function is rare [30]. Due to the lack of better dialogue between the chaos and cryptography communities, the disobedience of the security principles or the mis-utilization of the chaotic characteristics may result in insecure even naive design from cryptographic point of view. In the following, let us briefly review the general method of one kind of chaos-based hash functions firstly:

Step 1: Partition the pending message into a number of fixed-length processing units and map them into decimal numbers by means of linear transform. In some character-wise algorithms [12, 14], the processing unit includes 8 bits, and thus no extra padding is needed. While in some block-wise algorithms [23, 25], the processing unit includes some specific-length bits, such as 512, and thus the pending message has to be padded in a correct manner that its length is a multiple of a specific fixed-length.

Step 2: Choose a specific chaotic model, set its initial value and parameter as the algorithm secret key and start iteration. Until now, various chaotic models have been introduced into hash function construction, such as simple chaotic maps [11, 12, 14, 23], complex chaotic maps [16, 25] and chaotic neural networks [21, 22].

The message units processed in Step 1 are modulated into the chaotic iteration by affecting the chaotic phase space, parameter space or iteration times.

Step 3: Choose several specific iteration values, transform them to the corresponding binary format, extract some bits and juxtapose them to get the hash value.

From the above general construction method, we notice that the final hash value is transformed from several iteration values which are chosen from a specific chaotic orbit. If we set a certain initial iteration value and start iteration, a chaotic orbit will be obtained. However, if we choose other obit values from the above chaotic orbit as the new initial iteration value and start iteration, the same orbit will occur definitely. Therefore, it is possible for a carelessly designed chaos-based hash function to result in a collision problem. In order to securely construct one-way hash function based on chaotic map, it is necessary for us to summarize some attentions and borrow some principles from classic cryptography. They are listed as follows:

1) It is better to modulate the message into the chaotic parameter space rather than the phase space or iteration times.

2) If the pending message has to be padded, not only the repeated "1" or "0" bit, but also some information about the original message needs to be included. For example, a very standard way to have padding for hash function is appending 1 followed by a suitable size sequence of 0 and binary representation of length, which is used in the traditional hash function MD5 or SHA. Actually, there are some specific researches on suffix-free padding rules for hash function.

3) The iteration mode of the above chaos-based hash functions is similar to the well-known DM scheme. Therefore, chaos-based hash function can borrow some principles from DM scheme. In chaos-based hash function, the processing of a specific message unit should be related to that of its next message unit to some extent. To meet this rule, the iteration result of the previous message unit is usually used as the initial iteration value of the next message unit. This is a kind of sequential realization mode. Sometimes, if the high efficiency is highly desired, the order of each message unit can be utilized to introduce the connection among blocks as we have done in [23, 24]. By this way, a parallel mode is realized.

4) The constructions should give information about the performance of the proposed functions. Any crypto-primitive author should obligatorily give experimental measurements of his crypto-function. For example, the speed is measured in cycles/byte, or in Kbps, Mbps, Gbps. Other characteristics of the hash function are also good to know: How much memory is needed? Is it possible to be parallelized? Is it possible to be implemented in hardware, in RFID, etc.?

5) To make it easy for someone to repeat (or implement) the proposed design, the proposed functions should give clearly the test vectors and reference code (usually in pseudocode).

Acknowledgements

References

1. Schneier, B.: Applied Cryptography, 2nd edn. Wiley, New York (1996)
2. Stallings, W.: Cryptography and Network Security: Principles and Practice. Prentice Hall, New Jersey (1999)
3. Wang, X., Lai, X., Feng, D., Chen, H., Yu, X.: Cryptanalysis of the hash functions MD4 and RIPEMD. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. Eurocrypt, pp. 1–18. Springer, Heidelberg (2005)
4. Wang, X., Yu, H.: How to break MD5 and other hash functions. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 19–35. Springer, Heidelberg (2005)
5. Chen, Z., Huang, Y.: Chaotic one way hash function. Communications Technology (7), 96–98 (2001) (in Chinese)
6. Liu, J., Xie, J., Wang, P.: One way hash function construction based on chaotic mappings. Journal of Tsinghua University (Sci. & Tech.) 40(7), 55–58 (2000) (in Chinese)
7. Wang, J., Wang, Y., Wang, M.: The collision problem of one kind of methods for constructing one-way Hash function based on chaotic map. Acta Physica Sinica 55(10), 5048–5054 (2006) (in Chinese)
8. Wang, X., Zhang, J., Zhang, W.: One way hash function construction based on the extended chaotic maps switch. Acta Physica Sinica 52(11), 2737–2742 (2003) (in Chinese)
9. Kwok, H., Tang, W.: A Chaos Based Cryptographic Hash function for Message Authentication. International Journal of Bifurcation and Chaos 15(12), 4043–4050 (2005)
10. Deng, S., Li, Y., Xiao, D.: Analysis and improvement of a chaos-based Hash function construction. Communications in Nonlinear Science and Numerical Simulation 15(5), 1338–1347 (2010)
11. Yi, X.: Hash function based on the chaotic tent map. Transactions on Circuits and Systems 52(6), 354–357 (2005)
12. Xiao, D., Liao, X., Deng, S.: One-way Hash function construction based on the chaotic map with changeable-parameter. Chaos Solitons & Fractals 24(1), 65–71 (2005)
13. Baranousky, A., Daems, D.: Design of one-dimensional chaotic maps with prescribed statistical properties. International Journal of Bifurcation and Chaos 5(6), 1585–1598 (1995)
14. Wong, K.: A combined chaotic cryptographic and hashing scheme. Physics Letters A 307, 292–298 (2003)
15. Peng, F., Qiu, S., Long, M.: One-way Hash function construction based on two-dimensional hyper-chaotic mappings. Acta Physica Sinica 54(10), 4562–4568 (2005) (in Chinese)
16. Wang, S., Hu, G.: Hash function based on chaotic map lattices. Chaos 17, 023119 (2007)
17. Wang, Y., Liao, X., Xiao, D.: One-way hash function construction based on 2D coupled map lattices. Information Sciences 178(5), 1391–1406 (2008)

18. Wolf, A., Swift, J., Swinney, H., Vastano, J.: Determining Lyapunov exponents from a time series. Physica D 16, 285–317 (1985)
19. Li, H., Feng, D.: Composite nonlinear discrete chaotic dynamical systems and keyed hash function. Chinese Journal of Computers 26(4), 460–464 (2003) (in Chinese)
20. Liu, G., Shan, L., Dai, Y., Sun, J., Wang, Z.: One-way Hash function based on based on chaotic neural network. Acta Physica Sinica 55(11), 5688–5693 (2006) (in Chinese)
21. Wang, J., Wang, M., Wang, Y.: The collision of one keyed hash function based on chaotic map and analysis. Acta Physica Sinica 57(5), 2737–2742 (2008) (in Chinese)
22. Lian, S., Sun, J., Wang, Z.: Secure hash function based on neural network. Neurocomputing 69, 2346–2350 (2006)
23. Xiao, D., Liao, X., Deang, S.: Parallel keyed hash function construction based on chaotic maps. Physics Letters A 372(26), 4682–4688 (2008)
24. Xiao, D., Liao, X., Wang, Y.: Improving the security of a parallel keyed hash function based on chaotic maps. Physics Letters A 373(47), 4346–4353 (2009)
25. Zhang, J., Wang, X., Zhang, W.: Chaotic keyed hash function based on feedforward–feedback nonlinear digital filter. Physics Letters A 362, 439–448 (2007)
26. Xiao, D., Liao, X., Wong, K.: Improving the security of a dynamic look-up table based chaotic cryptosystem. IEEE Transactions on Circuits and Systems 53(6), 502–506 (2006)
27. Baptista, M.: Cryptography with chaos. Physics Letters A 240, 50–54 (1998)
28. Alvarez, G., Montoya, F., Romera, M., Pastor, G.: Cryptanalysis of dynamic look-up table based chaotic cryptosystems. Physics Letters A 326, 211–218 (2004)
29. Deng, S., Li, Y., Xiao, D.: A novel combined cryptographic and hash algorithm based on chaotic control character. Communications in Nonlinear Science and Numerical Simulation 14(11), 3889–3900 (2009)
30. Xiao, D., Liao, X., Wang, Y.: Collision analysis of one kind of chaos-based hash function. Physics Letters A 374(10), 1228–1231 (2010)

Chapter 6
Chaos-Based Video Encryption Algorithms

Zhaopin Su[1], Shiguo Lian[2], Guofu Zhang[1], and Jianguo Jiang[1]

[1] School of Computer and Information, Hefei University of Technology, China
   `szp@hfut.edu.cn`
[2] France Telecom R&D Beijing, China
   `shiguo.lian@ieee.org`

1 Introduction

In recent years, with the development of network technology and multimedia technology, multimedia data, especially video data, are used more and more widely in human society. Some multimedia data applied in entertainment, politics, economics, militaries, industries or education, etc., are necessary to be protected by providing confidentiality, integrity, and ownership or identity. To protect video contents, cryptology, which appears to be an effective way for information security, has been employed in many practical applications[1][2][3]. However, traditional ciphers like DES [4], IDEA [5], RSA[6] and AES[7], are often used for text or binary data, while not suitable for direct video encryption because of the following reasons.

Firstly, as digital videos are usually very large-sized and bulky, encrypting such bulky data with traditional ciphers incurs significant overhead, and it is too expensive for real-time video applications, which require real-time operations,such as displaying, cutting, copying, bit-rate control or recompression.

Secondly, in the case of digital videos, consecutive frames are similar and most likely only few pixels would differ from frame to frame, and such an extremely high data redundancy makes the conventional ciphers fail to obscure all visible information [8].

Additionally, for many practical applications, we would like to have very light encryption that preserves some perceptual information. For example, in a pay-per-view video system [9], a degraded but visible content could potentially influence a consumer to order certain paid services. This is impossible to achieve with traditional ciphers alone, which most likely degrades the data to a perceptually unrecognizable content.

Very recently, an increasing attention has been devoted to the usage of chaotic theory [10][11][12][13][14][15][16] to implement the encryption process. The main advantage of such encryption lies in the observation that a chaotic signal looks like noise for non-authorized users ignoring the mechanism for generating it. Secondly, time evolution of the chaotic signal strongly

depends on the initial conditions and the control parameters of the generating functions: slight variations in these quantities yield quite different time evolutions. In other words, this means that initial states and control parameters can be efficiently used as keys in an encryption system. What's more, generating of chaotic signal is often of low cost, which makes it suitable for the encryption of large bulky data[17].

Due to these recognized potential benefits, chaos-based video encryption algorithms are of high interest up to now, and have made great progress. This chapter focuses on chaos-based video encryption algorithms, by reviewing different types of works, investigating the state of the art progress, analyzing the performance evaluation and comparison, and presenting some challenges and open issues.

The organization of this chapter is as follows. In Section 1, backgrounds of chaos-based video encryption algorithms are given. Some requirements of video encryption are described in Section 2. Section 3 introduces the common video coding standards. Section 4 is a comprehensive review on today's chaos-based video encryption technology of different types. Section 5 and Section 6 compares and discusses the algorithms listed in Section 4, respectively. Some open issues and challenges are presented in Section 7. The last section concludes the chapter.

2   Some Requirements of Video Encryption

Due to some characteristics of digital video, such as large data volumes, high redundancy, interactive operations, and real-time responses, an ideal video encryption algorithm should satisfy some requirements.

1) Security: for video encryption, security is the primary requirement. Generally speaking,an encryption algorithm is regarded as secure if the cost for cracking it is no smaller than the one paid for the authorization of video content. For example,in broadcasting,the news may be of no value after an hour.Thus,if the attacker can not break the encryption algorithm during an hour, then the encryption algorithm may be regarded as secure in this application [18]. Therefore, the usage of chaotic maps in video encryption should guarantee the security of a video.

2) Invariance of compression ratio and constant bit rate: In some applications, it is required that the encryption transformation preserves the size of video data, which is called invariance of compression ratio. An algorithm with invariance of compression ratio can maintain the same storage space or transmission bandwidth, which is called constant bit rate. Sometimes, the encryption stage is allowed to slightly increase the size of a bit stream. In this case,video encryption algorithms should not change compression ratio or at least keep the changes in a small range.

3)Format compliance: In many applications, video data are often encoded or compressed before transmission, which produces the data streams with some

format information. The format information will be used by the decoder to recover the video data successfully. It is desired that the encryption algorithm preserves the multimedia format. In other words, after encrypting the encoded multimedia, ordinary decoders can still decode it without crashing. This property of an encryption algorithm is often called format compliance. When feeding the decoder with the format compliant encrypted data, the produced output seems distorted and randomized[8]. Generally, encrypting the data except the format information will keep the multimedia format. This will support some direct operations (e.g., decoding, playing, bit-rate conversion, etc.) and improve the error robustness in some extent.

4) Transmission error tolerance: Since the real-time transport of video data often occurs in noisy environments, which is especially true in the case of wireless channels [19][20], the delivered video is prone to bit errors. So, a perfect video encryption algorithm should be insensitive and robust to transmission errors.

5) Demand of real-time: As the demand of real-time of the video transmission and access, encryption and decryption algorithm can not bring much delay to transmission and access. Therefore, the encryption and decryption algorithm must be fast enough to meet the requirements of real-time video applications.

6) Multiple levels of security: A user may be willing to sacrifice some degree of security for the ability to perform more complex video processing. A given cryptosystem provides a certain level of security. Most available cryptographic systems are fully or partially scalable, in the sense that one can choose different security levels. Scalability is usually achieved by allowing variable key sizes or by allowing different number of iterations, or rounds. A higher level of security is achieved with larger key sizes or larger number of rounds[21][22].

7) Low overhead: Encryption techniques should have minimal overhead in terms of bandwidth requirements and processing power. A low overhead encryption algorithm will provide the end hosts with as much processing power for ensuring high quality video presentation to the user[8]. In addition, providing a secure stream should not significantly increase the bandwidth required to transmit it.

8) Allow degradation: For transmission of secure video streams, the user may be willing to accept a certain degree of degradation in quality in order to achieve a reasonable transport cost for the video session. When degradation is applied to a multimedia content, the content is usually still perceptible to some degree[8]. For instance, in some applications, such as video on demand, database search, etc., it could be desirable to encourage customers to buy the content. For this purpose, one may still see the objects in a degraded video, but the visual quality should be unacceptable for entertainment purposes, so that he prefers to pay to access the full-quality unencrypted content.

3 Some Video File Formats

Due to the huge size of digital videos, they are usually transmitted in compressed formats such as MPEG-x or H.26x. In this section, we will introduce some of them in brief.

3.1 *MPEG-x*

MPEG (Moving Picture Experts Group) compression standards[23] compress data to form small bits that can be easily transmitted and then decompressed. MPEG achieves its high compression rate by storing only the changes from one frame to another, instead of each entire frame. The video information is then encoded using a technique called Discrete Cosine Transform (DCT). MPEG uses a type of lossy compression, since some data are removed. But the diminishment of data is generally imperceptible to the human eye. MPEG has standardized the following compression standards [23]:

MPEG-1[24]: MPEG-1 is the first MPEG compression standard for audio and video. It was basically designed to allow moving pictures and sound to be encoded into the bit-rate of a Compact Disc. It is used on Video CD, SVCD and can be used for low-quality video on DVD Video. It was used in digital satellite/cable TV services before MPEG-2 became widespread. To meet the low bit requirement, MPEG-1 downsamples the images, as well as uses picture rates of only 24-30 Hz, resulting in a moderate quality.

MPEG-2[25]: MPEG-2 offers resolutions of 720x480 and 1280x720 at 60 fps (frames per second), with full CD-quality audio. This is sufficient for all the major TV standards, including NTSC, and even HDTV. MPEG-2 is used by DVD-ROMs. MPEG-2 can compress a 2 hour video into a few gigabytes. While decompressing an MPEG-2 data stream requires only modest computing power, encoding video in MPEG-2 format requires significantly more processing power.

MPEG-3[26]: Was designed for HDTV but was abandoned in place of using MPEG-2 for HDTV.

MPEG-4[27]: is a graphics and video compression algorithm standard that absorbs many of the features of MPEG-1 and MPEG-2 and other related standards, and uses further coding tools with additional complexity to achieve higher compression factors than MPEG-2. MPEG-4 files are designed to transmit video and images over a narrower bandwidth and can mix video with text, graphics and 2-D and 3-D animation layers.

3.2   H.26x

H.26x family is video compression coding standard in the domain of the ITU-T Video Coding Experts Group (VCEG), and consists of the following standards:

H.261[28]: H.261 was originally designed for transmission over ISDN (Integrated Services Digital Network) lines on which data rates are multiples of 64 kbit/s. The coding algorithm was designed to be able to operate at video bit rates between 40 kbit/s and 2 Mbit/s. The standard supports two video frame sizes: CIF (352x288 luma with 176x144 chroma) and QCIF (176x144 luma with 88x72 chroma) using a 4:2:0 sampling scheme. It also has a backward-compatible trick for sending still picture graphics with 704x576 luma resolution and 352x288 chroma resolution.

H.262[29]: is the second part of the MPEG-2 standard.

H.263[30]: originally designed as a low-bitrate compressed format for video-conferencing and developed as an evolutionary improvement based on experience from H.261, MPEG-1 and MPEG-2 standards.

H.264[31]: is a block-oriented motion-compensation-based codec standard, and contains a number of new features that allow it to compress video much more effectively than older standards and to provide more flexibility for application to a wide variety of network environments. H.264 is used in such applications as players for Blu-ray Discs, videos from YouTube and the iTunes Store, web software such as the Adobe Flash Player and Microsoft Silverlight, broadcast services for DVB and SBTVD, direct-broadcast satellite television services, cable television services, and real-time video conferences.

4   Chaos-Based Video Encryption Algorithms

In the past decade, chaos-based video encryption has been a topic of great interest. According to the relation between compression process and encryption, these proposed algorithms can be classified into two types: encrypting the raw video data, and encrypting the video data in compression process.

4.1   Encrypting the Raw Video Data

This type of chaos-based video encryption algorithm encrypts the raw video data directly with chaotic maps. Among them, some encrypt the raw data completely without considering region-of-interest, [32][33][34][35][36] and some consider the region-of-interest partially or selectively [41][42].

4.1.1   Encryption without Considering Region-of-Interest

Encryption without considering interest regions means to encrypt the video data as binary large objects, pixels, or sets of frames, without taking into con-
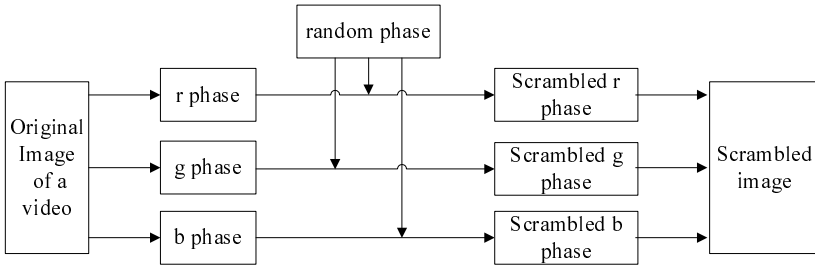
Fig. 1 Phase scrambling methods

sideration video objects or any other kind of regions of semantic information. Thus, it treats the regions fairly without special considerations.

Li et al [32] proposed a chaotic video encryption scheme (CVES) based on multiple digital chaotic systems, which is independent of any video compression algorithms. In CVES, video data is encrypted frame by frame. First, each plain-block is first XORed by a chaotic signal pseudo-randomly generated from chaotic maps based on perturbation-based algorithm [37], and then substituted by a pseudo-random S-box generated from all chaotic orbits of the chaotic maps. Their detailed analysis have shown that CVES has fair speed and security, and can be realized easily by both hardware and software. Furthermore, CVES can be easily extended to other real-time secure applications.

Ganesan et al [33] described a public key encryption (PKE) of images and videos based on chebyshev maps, shown as Eq. (1) [38]. In the work, videos in simple terms are considered as a collection of images, and each video is made up of frames and each frame is like a still image. Encrypting video is equal to encrypting each frame by Arnold scrambling [39]. If in a video, the number of frames is too large, the authors propose the use of Phase Scrambling [40] for video encryption instead of Arnold scrambling. The method of Phase Scrambling (See Fig. 1) adds the same random phase structure to the original r, g and b phase structures respectively to randomize the phase of the r, g and b layers of an image. The scrambling operation's security is not high enough to resist known-plaintext or select-plaintext attacks.

$$
\begin{cases}
T_n(x) = 2 \cdot x \cdot T_{n-1}(x) - T_{n-2}(x), n \geq 2 \\
T_0(x) = 1 \\
T_1(x) = x
\end{cases}
\tag{1}
$$

Kezia et al [34] also treated video data as a set of frames, and used a high dimensional Lorenz chaotic system to encrypt each frame by confusing the position of the pixels (hence called LCS). If the frames are large in size, then it is broken into macro-blocks for the encryption. Moreover, their concept of multi-key based on logistic map, whose values cannot be predicted in the long-run, is also employed where each frame is encrypted by a unique key

instead of changing the key for a particular number of frames. The position confusion operation is weak when it is used alone for data encryption.

Mao et al[35] and Lian et al. [36] first extended the standard two-dimensional baker map to a three-dimensional setting, then constructed a fast and secure encryption scheme (FSES) (see Fig. 2). In their scheme, to make known-plaintext attack infeasible, an XOR plus modulo (mod) operation (see Eq. (2)) is inserted between every two adjacent rounds of chaotic map based confusion. This kind of scheme combines confusion and diffusion, and aims to obey traditional block cipher's principles.

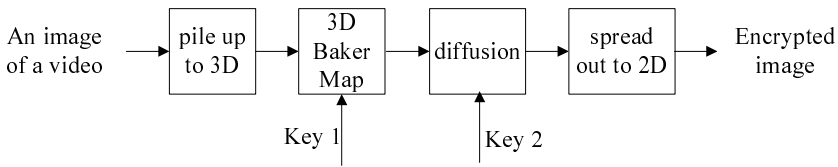$$C(k) = \phi(k) \oplus \{[I(k) + \phi(k)] \bmod N\} \oplus C(k-1) \qquad (2)$$

Fig. 2 Encryption scheme based on 3D baker map

The chaos-based video encryption algorithms mentioned above deal with video data as binary large objects, pixels, or sets of frames, without taking into account of regions-of-interest. These regions may need better protection or can be the only regions that need protection, depending on the practical applications.

4.1.2   The Encryption Considering Regions-of-Interest

Generally speaking, for the video data, a region of interest means human video objects or any other kind of regions of semantic information. In many practical applications, it is not necessary or suitable to encrypt all video data, while just regions of interest. In this issue, researchers have proposed some encryption algorithms.

Tzouveli et al. [41] proposed a human video object encryption system (HVOE) based on the chaotic logistic map (see Fig. 3). In their system, face regions are first efficiently detected, and afterwards body regions are extracted using geometric information of the location of face regions. Then, the pixels of extracted human video objects are encrypted using an iterative cipher module, which is based on logistic map and a feedback mechanism responsible for mixing the current encryption parameters with encrypted information of the previous step. The encryption of each plain pixel depends on the key, the value of the previous cipher pixel and the output of the logistic map. This method can save a great amount of computational resources and time devoted for encrypting the whole contents of a video file. This method's
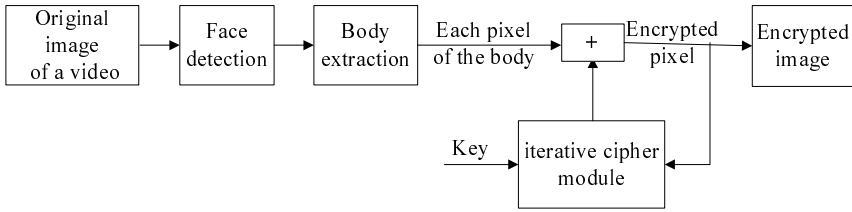
Fig. 3 Human video object encryption system

security depends on the chaotic sequence's randomness and the detection of face region.

Ntalianis et al.[42] proposed a video object based chaotic encryption system (VOCE) (Fig. 4). Initially, stereoscopic pairs are analyzed and video objects are automatically extracted based on the appropriate fusion of color information according to depth constraints. Next, for each video object, multiresolution decomposition is performed and the pixels of the lowest resolution level are encrypted using a chaotic cipher module combining a simple chaotic stream cipher and two simple chaotic block ciphers (with time variant S-boxes) to implement a complex product cipher. Finally, the encrypted regions are propagated to the higher resolution levels and the encryption process is repeated until the highest level is reached. The system presents robustness against known cryptanalytic attacks, enables layered access of multimedia content and the overall security can be enhanced due to region topology. This scheme's security depends on the encryption algorithm and the video object detection.

Encrypting the raw video data before encoding or compression process removes a lot of redundancy, which results in a very poor compression ratio. Additionally, it changes video data format, so that the encrypted video cannot

Fig. 4 Chaos-based video object encryption system in [42]

be displayed without decrypting it firstly. So, some researchers have proposed that the encryption should be implemented in compression process.
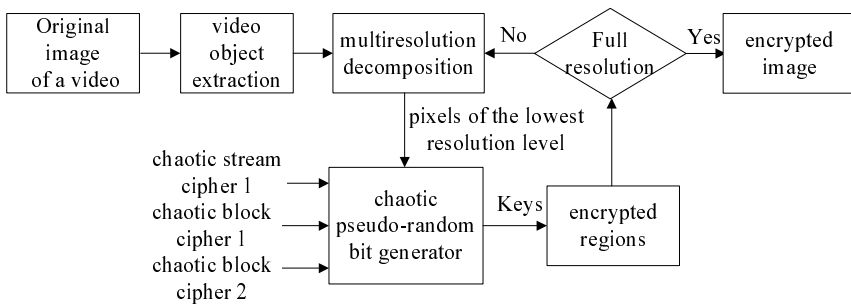
4.2 Encrypting the Video Data in Compression Process

Encrypting the video data in compression process means realizing encryption in the encoding process before entropy coding, i.e. CAVLC(Context-adaptive variable-length coding), CABAC(Context-adaptive binary arithmetic coding), VLC(variable length coding), RLC(run length coding), Golomb, Huffman, etc. Till now, some algorithms have been proposed for MPEG, and some for H.26x.

4.2.1 Encryption for MPEG

For MPEG, the representative works are done by Yang et al [43], Lian et al. [44][45] and Hamdi et al. [46].

Yang et al. [43] used double coupling logistic maps (shown as Eq. (3)) to scramble the DCT coefficients of every I-frame of the video, and then used another chaotic map (shown as Eq. (4)) to encrypt the DCT coefficients of the scrambled I-frame (hence called DCLM). The process of encryption is shown in Fig. 5. In the whole process, five keys are introduced, and the key space is large, which makes brute-force attack difficult. Moreover, as I-frames do not refer to any other frame and are the beginning of decoding, their changes can greatly influence the B-frames and P-frames, which makes the DCLM effective in video protection. Besides, DCLM, only encrypting the DCT coefficients of I-frames, brings little consumed time, and is feasible for real-time applications. However, considering that there are some macro blocks in B-frame or P-frame, which are encoded without referring to I-frame, these blocks will be left unencrypted. Thus, some video contents may be intelligible, and thus the encryption scheme is not secure enough.

$$\begin{cases} x_{n+1} = \mu_x x_n(1 - x_n) \\ y_{n+1} = \mu_y y_n(1 - y_n) \end{cases} \tag{3}$$

$$x(i + 1) = 1 - \mu x^2(i) \tag{4}$$

Lian et al. [44] constructed an efficient image/video encryption scheme (EES) based on 2D coupled map lattice (CML) [47], which is a kind of spatiotemporal chaos. The chaotic lattices are used to generate pseudo-random sequences and then encrypt some sensitive parameters during the video compression process. For example, for MPEG2 videos, only the intra-blocks in each frame are encrypted. Fig. 6 gives the architecture of the encryption scheme. In compression, after pre-encoding (i.e., color space transformation),
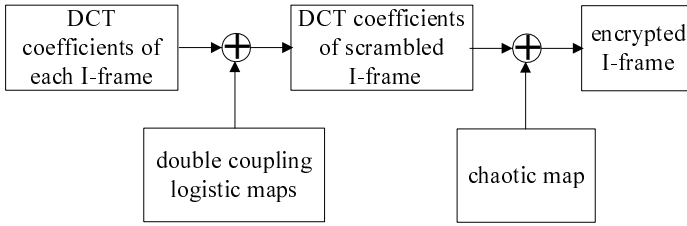
Fig. 5 The video encryption in [43]

block partitioning (each block is in 88 size), DCT transformation and quanti-zation, the blocks are encrypted by the proposed cipher one by one, and the cipher-blocks are then post-encoded (i.e., zig-zag scan and VLC). The pro-posed scheme satisfies the requirement of secure encryption principles, the encrypted videos are secure in perception, the encryption operation does not change the compression ratio apparently, and increases little computational cost compared with video compression. The scheme's cryptographic security depends on the randomness of the chaotic sequences generated by 2D coupled map lattice.
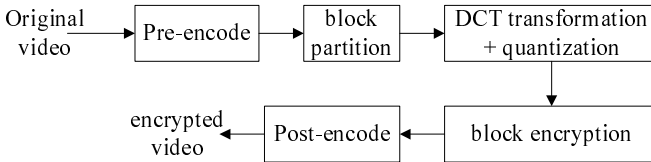
Fig. 6 Architecture of the encryption scheme

Lian et al. [45] presented a video encryption algorithm which combines encryption process with MPEG-2 encoding process (hence called VEM2). In the algorithm, I-frame, B-frame and P-frame of MPEG-2 video are encrypted with different methods, respectively. That is, for I-frames, the macroblocks are inter-permuted by color-plane confusion. For each intra-macroblock, the DCT coefficients are permuted by coefficient confusion. The DCT coefficients' signs of each intra-macroblock and motion vectors of each inter-macroblock are modulated by chaotic sequence generated by the chaotic sequence gener-ator proposed in [48]. All these encryption processes are controlled by a key generation and distribution system based on Logistic map [49]. The analysis and experiments have shown that the VEM2 is secure against brute-force attack and known-plaintext attack. Moreover, the VEM2 is of low compu-tational complexity, costs little time, supports direct bit-rate control and is more robust to transmission errors.

In addition, Hamdi et al. [46] also proposed a progressive Chaotic Video Encryption Scheme (PCVE) for MPEG-4 coding. In their scheme, multi-dimensional chaotic maps [50] are used to build chaotic multi-resolution transforms, which can introduce randomness in the selection of wavelet filters and de-correlate the video data. The major advantage of PCVE is that a client can access to multiple resolutions of the streamed video. These resolutions vary according to the security level of the client as well as the networking and processing capabilities. Moreover, the PCVE may be secure against statistical attacks and known-plaintext cryptanalysis. Additionally, it is of low computational complexity, and does not affect the compression performance ensured by the MPEG-4 coder. Besides, The bit rate control functionality makes it compatible with video transmission in wireless networks.

4.2.2   Encryption for H.26x

For H.26x, the representative works are done by Jian et al [51] and Chiaraluce et al. [52].

For H.263, Jian et al [51] proposed chaos-based encryption algorithm (CBEA) for the H.263 video-conference coding standard. A sawtooth-like chaotic map is first used to generate a pseudo-random bit sequence(PRBS). Then, according to the PRBS, all of the DC coefficients, part of the AC coefficients of I blocks as well as Motion Vectors (MVs) are encrypted in the video coding. The full encryption algorithm is shown in Fig. 7, where the cipher operations have been seamlessly integrated into the H.263 encoding process, i.e., before RLC and packaging. Although the encryption processing doesn't interfere the motion features of the videos, the algorithm may introduce slight computational overhead and slight data inflation in video encoding. The scheme's security depends on the chaotic sequence's randomness.
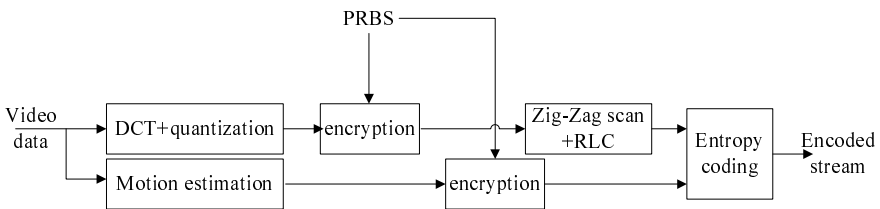
Fig. 7 Chaos-based encryption algorithm in [51]

Chiaraluce et al. [52] presented a selective encryption algorithm (SEA) for the H.263+ videos, which employs suitably arranged three different chaotic functions (see Fig. 8) to encrypt the video data selectively. In Fig. 8, video data include the most significant bit in the DC coefficients of DCT (Discrete Cosine Transform), the AC coefficients of I-MB's (Intra MacroBlocks), the sign bit of the AC coefficients of the PMB's (Predicted MacroBlocks) and the

sign bit of the Motion Vectors. And CM_1 is the skew tent map [53], CM_2 is a saw-tooth likewise map [54], and CM_3 is the logistic map [49]. The outputs of the CM_1 and CM_2 are added, and then the addition is scaled to be an integer between 0 and 255. Each scaled integer is used as the initial condition of the third map to generate a 64-size key stream to mask the plaintext with XOR operation. In order to increase the security level against known/chosen-plaintext attacks, it was suggested to change the key every 30 frames. The algorithm introduces a modest delay, offers good security and the ability to reconstruct perfectly the image, and gets a good compromise between the need to improve security while maintaining a limited additional processing time. These properties make it suitable for video applications that have real time or almost real time requirements.
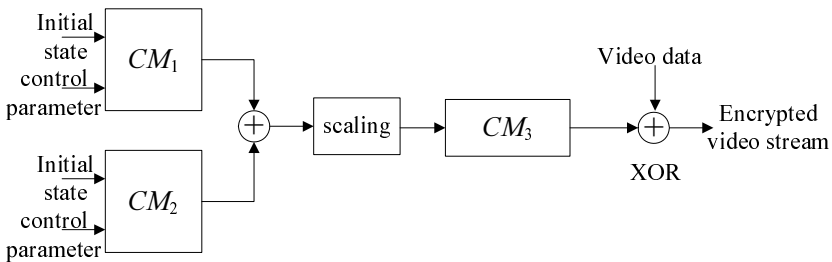
Fig. 8 Block diagram of the encryption diagram in [52]

4.3 Encrypting the Compressed Video Data

Encrypting the compressed video data means realizing encryption after entropy-encoding and before package. The representative works are done by Lian et al.[55][56] and Qian et al. [57].

Lian et al. [55] constructed a chaotic stream cipher with random feedback mode based on a discrete piecewise linear chaotic map [58], and then encrypted both the intra-macroblocks (all the macroblocks in I-frame and some intra-encoded macroblocks in P/B-frame) and the motion vectors' signs segment by segment (hence called CSCF), which is shown in Fig. 9. The whole encryption process is achieved after VLC and before packaging. The encryption scheme is of high key sensitivity, secure in perception, format compliant, and error robust. Besides, the encryption/decryption process does not affect the compression/decompression process greatly. The cryptographic security depends on the chaotic sequence's randomness.

In [56], a fast video encryption scheme is proposed combining with MPEG-4 codec (hence called VEM4). In the scheme, the file format information, such as file header, packet header, and so on, are left unencrypted in order to support such operation as bit-rate control; the motion vectors, subbands, code blocks or bit-planes are partially encrypted by a stream cipher based on a
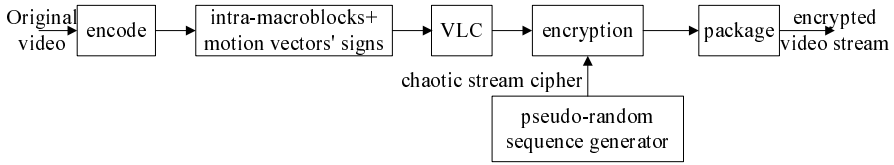
Fig. 9 Encryption scheme in [55]

modified chaotic neural network [59][60][61][62]. Moreover, for each encoding-pass, the chaotic binary sequence is generated from different initial-condition based on logistic map. Thus, if one encoding-pass cannot be synchronized because of transmission errors, the other ones can still be decrypted correctly. The encryption scheme is of high security in perception, of low computation complexity, and secure against brute-force attack, statistic attack or differential attack. And it keeps compression ratio and file format unchanged, supports direct bit-rate control, and keeps the error-robustness unchanged. The scheme's cryptographic security depends on the chaotic sequence's randomness.

Qian et al. [57] proposed a multiple chaotic system (MCS) for MPEG-2 which combines the partial encryption with block permutation and confusion. In their system, three chaotic or hyperchaotic maps, namely Logistics Map [63], 2-D Baker Map [64] and 4-D hyperchaotic Map [65], are introduced for stream partial encryptions, block permutation, confusion after block permutation, respectively. Moreover, stream ciphers encrypt only DC coefficients by XOR operation after DCT and quantization when compressing the video data, and block permutation and confusion are carried out after the video compression, respectively (See Fig. 10). The algorithm is secure, efficient, and of low computational complexity. Besides, it nearly brings no data expansion.
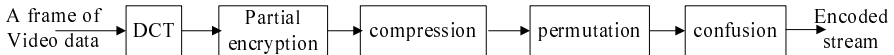
Fig. 10 multiple chaotic system in [57]

5   Performance Evaluation

Some performances are often considered to evaluate a video encryption scheme, e.g., security analysis, encryption speed, compression ratio and error robustness.

5.1 Security Analysis

Security of an algorithm is generally evaluated by the perceptual experiments, key apace analysis, key sensitivity analysis, and the ability against attacks.

The perceptual experimental result is achieved by a group of comparison between frames of the original video and those of the encrypted one. Besides, some works decrypt the encrypted video to examine the effects of their encryption.

Key space of an encryption algorithm is generally defined as the number of encryption/decryption key pairs that are available in the cryptosystem. Assume k_i denotes a key and K represents a finite set of possible keys, the key space can be expressed as $K = \{k_1, k_2, ..., k_r\}$, where r is the number of key. To make brute-force attack infeasible, the size of key space should be large enough. For chaos-based encryptions, the chaotic sequence generator should produce chaotic ciphers with good randomness, which can be tested by long period, large linear complexity, randomness and proper order of correlation immunity [67].

Key sensitivity of a chaotic cipher refers to the initial states sensitivity and control parameters sensitivity of chaotic map. A typical key sensitivity test is performed according to the following steps: First, assume a frame of a video is encrypted by using the key "K1=0123456789". Then, the same frame is encrypted by using the key "K2=1123456789", which changes the least significant bit of K1. Finally, the above two encrypted frames, encrypted by K1 and K2 respectively, are compared, and cross-correlation curve between the two encrypted frames is analyzed.

A good cipher can avoid potential attacks. In general, brute-force attack is analyzed by key apace analysis. Known-plaintext attack and chosen-plaintext attack can be tested by comparing the original frame of a video and the decrypted one. Differential attack test can be achieved through measuring the percentage p of different pixel numbers (see Eq. (5) and Eq. (6)) between two encrypted images, I_1 and I_2 (the width and height is W and H, respectively), whose corresponding plain-images have only one pixel's difference.

$$p = \frac{\sum\limits_{i,j} D(i,j)}{W \cdot H} \cdot 100\%, i = 0, 1, \cdots, W - 1 j = 0, 1, \cdots, H - 1 \qquad (5)$$

$$D(i,j) = \begin{cases} 0, I_1(i,j) = I_2(i,j) \\ 1, \text{otherwise} \end{cases} \qquad (6)$$

5.2 Encryption Speed Test

The encryption time is tested in three manners: absolute encryption time, relative encryption time ratio, and computation complexity analysis. Absolute encryption time refers to the assumed time for encrypting a video, and

its measuring unit is second. Relative encryption time ratio refers to the time ratio between encryption and compression. Computation complexity of an encryption scheme depends on the cost of the chaos-based cipher and the video data volumes to be encrypted.

If the computational cost or assumed time of a video encryption scheme is very little compared with video compression, it is considered to be suitable for real-time applications.

5.3   Compression Ratio Test

In general, the compression ratio is tested by comparing the original compressed data volumes and encrypted and compressed data volumes. Considering that the compression coder often produces the data stream with a given bit-rate, the compression ratio test may be measured by the video quality under certain bit rate. The common measurement of video quality is $PSNR$ (Peak Signal-to-Noise Ratio) shown as Eq. (7) and Eq. (8), where B is sampling frequency, I and I' represent an original mn frame and the encrypted one, respectively.

$$PSNR = 10 \cdot \log_{10} \left(\frac{(2^B - 1)^2}{MSE} \right) \tag{7}$$

$$MSE = \frac{1}{m \cdot n} \cdot \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i,j) - I'(i,j)]^2 \tag{8}$$

5.4   Error-Robustness Test

If an encryption scheme does not change file format, and a slight change in one pixel does not spread to others, it is of lower sensitivity to transmission errors.

The general test method for error-robustness is analyzing the relationship (usually expressed by a curve) between the quality PSNR of the decrypted frames and the number of bit-error happened in the encrypted frames. Besides, error-robustness can be tested through correct decryption of an encrypted video, even if a frame is corrupted or lost in its transmission.

6   Performance Comparison

In this section, we compare the performance of different encryption algorithms mentioned above. Here, various aspects listed in Section 2 are considered, and contrast results are shown in Table 1.

Table 1 Comparison of chaos-based video encryption algorithms

	Security	ICR	FC	TET	Real-Time	MLS	LO	AD
CVES[32]	H	No	No	No	Yes	No	Yes	No
PKE[33]	L	No	No	No	Yes	No	Yes	No
LCS [34]	L	No	No	Yes	Yes	No	Yes	No
FSES[35]	H	No	No	No	Yes	No	Yes	No
HVOE[41]	L	No	No	No	Yes	No	Yes	No
VOCE[42]	M	No	No	No	Yes	Yes	Yes	Yes
DCLM[43]	L	No	Yes	No	Yes	No	Yes	No
EES[44]	M	Yes	Yes	No	Yes	No	Yes	No
VEM2[45]	M	Yes	Yes	Yes	Yes	No	Yes	No
PCVE[46]	M	Yes	Yes	No	Yes	Yes	Yes	Yes
CBEA[51]	M	Yes	Yes	No	Yes	No	Yes	No
SEA[52]	M	No	Yes	No	Yes	No	Yes	No
CSCF[55]	M	Yes	Yes	Yes	Yes	No	Yes	Yes
VEM4[56]	M	Yes	Yes	Yes	Yes	No	Yes	No
MCS[57]	H	Yes	Yes	No	Yes	No	Yes	No

ICR:Invariance of compression ratio; FC: Format compliance
TET: Transmission error tolerance; MLS: Multiple levels of security
LO: low overhead; AD: Allow degradation; L: Low security; M: Middle security;
H: High security.

From Table 1, we get the following conclusions:

1) The encryption algorithms, CVES, PKE, LCS and FSES, encrypt the video data completely, without considering interest regions. Their security depends on the proposed chaotic ciphers. Generally, if the ciphers are well-designed, they are often of higher security, higher complexity, and higher overhead than other types. So, they are more suitable for secure video storing than for real-time transmission.

2) The encryption algorithms considering interest regions (HVOE, VOCE) encrypt only the interest regions, and leave the rest (such as background) unprotected. They are of lower computation complexity and lower overhead, and more suitable for real-time applications than that without considering interest regions. Their cryptographic security depends on the adopted chaotic cipher and the region selection. These algorithms that encrypt the raw video data directly can not preserve invariance of compression ratio, and change the video format, so that the encrypted video cannot be displayed without decrypting it firstly. Besides, they do not consider the compression process, especially lossy compression, which may result degradation after video decoding, and bring difficulty to decryption.

3) The algorithms that encrypt the video data in compression process belong to partial or selective encryption, and are often of lower complexity than those encrypt the raw video data directly. However, some of them (DCLM, SEA)

change the compression ratio for they change the statistical characteristics of DCT coefficients. Interestingly, some of them can keep file format unchanged. Thus, these algorithms support direct bit rate control, that is, they permit to re-compress the encoded and encrypted video before decrypting it firstly, and save much time for secure transcoding. Therefore, they are more suitable for real-time applications, such as wireless multimedia network or multimedia transmission over narrow bands.

4)The algorithms that encrypt the compressed video data can not only preserve invariance of compression ratio and format compliance, but also be of low overhead. Additionally, it is of low-cost and is easy to be realized. For these advantages, it is suitable for real-time required applications, such as video transmission or video access. However, as the video stream after entropy encoding may have a certain structure or syntax, the encryption scheme may destroy the structure of the video stream. And thus, these algorithms, without considering the rules of package before transmission, may bring error spreading when the transmission error happens.

5) For the algorithms in Table 1, many of them don't give any illustration whether their algorithms are insensitive and robust to transmission errors except LCS, VEM2, CSCF and VEM4.

7   Discussions

From the above survey on the issues of chaos-based video encryption, we can learn many valuable experiences on how to design a video encryption scheme based on chaotic maps. However, there are still some questions to be discussed:

Now, the security of different chaos-based encryption algorithms is evaluated by different means, including the perceptual results, key apace, key sensitivity, ability against attacks, and so on. How to assess a chaos-based cipher and also the video specific encryption algorithm in a general manner is still an open issue.

Most of existing chaos-based video encryption schemes provide different levels of video security. Various practical applications may demand different encryption approaches with certain security levels. Consequently, how to define the security levels in detail and how to choose the appropriate security level for certain application will be interesting topics.

Encrypting more video data results in higher level of security, meanwhile, it also produces more computational complexity and costs much more time. Therefore, how to get a good tradeoff between security and time-efficiency is another important issue.

As is known that, encryption algorithms are often sensitive to transmission errors, for a slight change in cipher text often causes great changes in the decoded data. The existing chaos-based video encryption schemes lack of error-robustness test in practical transmission environments. Thus, how to

design a chaos-based video encryption scheme with strong error-robustness is also an challenging task.

8 Conclusions

Video encryption plays a more and more important role in today's multimedia world. And chaos theory provides a fast and practical solution for the design of digital cipher for video encryption. Many efforts have been devoted to study the security issue, and some valuable algorithms can be used as the fundamentals of future research.

Although chaos-based video encryption appears to be promising, it is not yet mature. More efforts are needed for its further development toward practical applications with high security, invariance of compression ratio, format compliance, strong transmission error tolerance, real-time, multiple levels of security, low overhead and allow degradation.

References

1. Lian, S.: Multimedia Content Encryption: Techniques and Applications. Auerbach Publication, Taylor & Francis Group (2008)
2. Lian, S., Zhang, Y.: Handbook of research on secure multimedia distribution. IGI Global (formerly Idea Group, Inc.), Hershey, Pennsylvania (2009)
3. Lian, S., Wang, Z.: Compare of Several Wavelet Coefficient Confusion Methods Applied in Multimedia Encryption. In: Proceedings of the 3rd International Conference on Computer Networks and Mobile Computing (ICCNMC 2003), pp. 372–376. IEEE Computer Society, Shanghai (2003)
4. Tuchman, W.: A brief history of the data encryption standard. ACM Press, Addison-Wesley Publishing Co., New York (1997)
5. Dang, P.P., Chau, P.M.: Implementation IDEA algorithm for image encryption. In: Mathematics and Applications of Data/Image Coding, Compression, and Encryption III. Proceedings of SPIE, vol. 4122, pp. 1–9 (2000)
6. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to algorithms, 2nd edn. MIT Press, McGraw-Hill, Cambridge (2001)
7. Zeghid, M., Machhout, M., Khriji, L., Baganne, A., et al.: A modified AES based algorithm for image encryption. International Journal of Computer Science and Engineering 1(1), 70–75 (2007)
8. Furht, B., Muharemagic, E., Socek, D.: Multimedia encryption and watermarking. Springer, Heidelberg (2005)
9. Ballest, A.M.: Real-time pay-per-view of protected multimedia content v:2.0. Ph.D.Dissertation, Universitat Polit'ecnica de Catalunya, Barcelona (2004)
10. Devaney, R.L.: An introduction to chaotic dynamical systems, 2nd edn. Westview Press, San Francisco (2003)
11. Alligood, K.T., Sauer, T., Yorke, J.A.: Chaos: an introduction to dynamical systems. Springer, Heidelberg (1997)
12. Yang, T., Wu, C.W., Chua, L.O.: Cryptography based on chaotic systems. IEEE Transactions on Circuits and Systems-I: Fundamental Theory and Applications 44, 469–472 (1997)

13. Solak, E.: Cryptanalysis of observer based discrete-time chaotic encryption schemes. International Journal of Bifurcation and Chaos 15(2), 653–658 (2005)
14. He, J., Qian, H., Zhou, Y., Li, Z.: Cryptanalysis and improvement of a block cipher based on multiple chaotic systems. Mathematical Problems in Engineering 2010, 1–14 (2010)
15. Alvarez, G., Montoya, F., Romera, M., Pastor, G.: Breaking two secure communication systems based on chaotic masking. IEEE Transaction on Circuit and SystemsII: Express Briefs 51, 505–506 (2004)
16. Kocarev, L., Galias, Z., Lian, S.: Intelligent Computing Based on Chaos. Springer, Heidelberg (2009)
17. Alvarez, G., Li, S.: Some basic cryptographic requirements for chaos-based cryptosystems. International Journal of Bifurcation and Chaos 16, 2129–2151 (2006)
18. Lian, S., Sun, J., Liu, G., Wang, Z.: Efficient video encryption scheme based on advanced video coding. Multimed. Tools Appl. 38, 75–89 (2008)
19. Gschwandtner, M., Uhl, A., Wild, P.: Transmission error and compression robustness of 2D chaotic map image encryption schemes. EURASIP Journal on Information Security 2007, 1–16 (2007)
20. Lin, C.F., Chung, C.H., Chen, Z.L., et al.: A chaos-based unequal encryption mechanism in wireless telemedicine with error decryption. Wseas Transactions on Systems 7(2), 49–55 (2008)
21. Li, Y., Cai, M.: H.264-based multiple security levels net video encryption scheme. In: Proceedings of International Conference on Electronic Computer Technology, pp. 8–11. IEEE press, Macau (2009)
22. Rohwer, K., Krout, T.: Multiple levels of security in support of highly mobile tactical internets-ELB ACTD. In: Proceedings of Military Communications Conference on Communications for Network-Centric Operations: Creating the Information Force, vol. 1, pp. 81–86. IEEE press, McLean (2001)
23. Chariglione, L.: MPEG and Multimedia Communications. IEEE Transactions on Circuits and Systems for Video Technology 7(1), 5–18 (1997)
24. Srinivasan, U., Pfeiffer, S., Nepal, S., Lee, M., Gu, L., Barrass, S.: A survey of MPEG-1 audio, video and semantic analysis techniques. Multimedia Tools and Applications 27(1), 105–141 (2005)
25. Kleihorst, R.P., Vander Werf, A., Bruls, W.H.A., Verhaegh, W.F.J., Waterlander, E.: MPEG2 video encoding in consumer electronics. Journal of Vlsi Signal Processing Systems 15, 5–20 (1997)
26. Pan, D.: A tutorial on mpeg/audio compression. IEEE MultiMedia 2, 60–74 (1995)
27. Richardson Iain, E.G.: H.264 and MPEG-4 video compression: video coding for next-generation multimedia. John Wiley & Sons, Ltd., New York City (2003)
28. Girod, B., Steinbach, E., Faerber, N.: Comparison of the H.263 and H.261 video compression standards. In: Rao, K.R. (ed.) Standards and Common Interfaces for Video Information Systems. Critical Reviews of Optical Science and Technology, Philadelphia, Pennsylvania (1995)
29. Vetrivel, S., Suba, K., Athisha, G.: An overview of h.26x series and its applications. International Journal of Engineering Science and Technology 2(9), 4622–4631 (2010)
30. Girod, B., Steinbach, E., Faerber, N.: Performance of the H.263 video compression standard. The Journal of VLSI Signal Processing 17(2-3), 101–111 (1997)

31. Wiegand, T., Sullivan, G.J., Bjontegaard, G., Luthra, A.: Overview of the H.264/AVC Video Coding Standard. IEEE Transactions on Circuits and Systems for Video Technology 13(7), 560–576 (2003)
32. Li, S.J., Zheng, X., Mou, X., Cai, Y.: Chaotic encryption scheme for real-time digital video. In: Real-Time Imaging VI, San Jose, CA, USA. Proceedings of SPIE, vol. 4666, pp. 149–160 (2002)
33. Ganesan, K., Singh, I., Narain, M.: Public key encryption of images and videos in real time using chebyshev maps. In: Proceedings of the 2008 Fifth International Conference on Computer Graphics, Imaging and Visualisation, pp. 211–216. IEEE Computer Society, Washington, DC, USA (2008)
34. Kezia, H., Sudha, G.F.: Encryption of Digital Video Based on Lorenz Chaotic System. In: Proceedings of the 16th International Conference on Advanced Computing and Communications, pp. 40–45. IEEE Computer Society Press, Tamilnadu (2008)
35. Mao, Y., Chen, G., Lian, S.: A novel fast image encryption scheme based on the 3d chaotic baker map. International Journal of Bifurcation and Chaos 14(10), 3613–3624 (2004)
36. Lian, S., Mao, Y., Wang, Z.: The 3-dimension extension of baker map and its application in multimedia encryption. Chinese Journal of Control & Decision 19(6), 714–717 (2004) (Chinese)
37. Sang, T., Wang, R., Yan, Y.: Perturbance-based algorithm to expand cycle length of chaotic key stream. Electronics Letters 34(9), 873–874 (1998)
38. Bergamo, P., D'Arco, P., Santis, A., Kocarev, L.: Security of public key cryptosystems based on Chebyshev polynomials. IEEE Transactions on Circuits and Systems-I 52, 1382–1393 (2005)
39. Prasad, V.V.R., Kurupati, R.: Secure image watermarking in frequency domain using arnold scrambling and filtering. Advances in Computational Sciences and Technology 3(2), 236–244 (2010)
40. Nishchal, N.K., Joseph, J., Singh, K.: Fully phase based encryption using fractional Fourier transform. Opt. Eng. 42, 1583–1588 (2003)
41. Tzouveli, P., Ntalianis, K., Kollias, S.: Security of human video objects by incorporating a chaos-based feedback cryptographic scheme. ACM Multimedia, 10–16 (2004)
42. Ntalianis, K.S., Kollias, S.D.: Chaotic video objects encryption based on mixed feedback, multiresolution decomposition and time-variant S-boxes. In: Proceedings of ICIP (2) 2005, pp. 1110–1113. IEEE, Genoa (2005)
43. Yang, S., Sun, S.: A video encryption method based on chaotic maps in DCT domain. Progress in Natural Science 18(10), 1299–1304 (2008)
44. Lian, S.: Efficient image or video encryption based on spatiotemporal chaos system. Chaos, Solitons and Fractals 40, 2509–2519 (2009)
45. Lian, S., Wang, Z., Sun, J.: A fast video encryption scheme suitable for network applications. In: Proceedings of 2004 International Conference on Communications, Circuits and Systems (ICCCAS 2004), pp. 566–570. IEEE press, Chengdu (2004)
46. Hamdi, M., Boudriga, N.: A progressive chaotic mpeg-4 video encryption scheme for wireless networks. In: Proceedings of the 2009 IEEE International Conference on Communications, pp. 5420–5424. IEEE Press, Piscataway (2009)
47. Wang, S., Kuang, J., Li, J., et al.: Chaos-based communication in a large community. Phys. Rev. 66(6), 1–4

48. Kuo, C.J., Chen, M.S.: A new signal encryption technique and its attack study. In: Proceedings of IEEE International Conference on Security Technology, pp. 149–153. IEEE press, Taipei (1991)

49. Su, Z., Jiang, J., Lian, S., et al.: Hierarchical Selective Encryption for G.729 Speech Based on Bit Sensitivity. Journal of Internet Technology 11(5), 599–607 (2010)

50. Hamdi, M., Boudriga, N.: Four dimensional chaotic ciphers for secure image transmission. In: Proceedings of IEEE International Conference on Multimedia and Expo, pp. 437–440. IEEE press, Hannover (2008)

51. Jian, H., Mao, Y., Wang, Z.: A novel chaos-based video encryption algorithm. Applied Computational Intelligence, 641–648 (2004)

52. Chiaraluce, F., Ciccarelli, L., Gambi, E., Pierleoni, P., Reginelli, M.: A new chaotic algorithm for video encryption. IEEE Transactions on Consumer Electronics 48(4), 833–844 (2002)

53. Freitas, A.C.M.: Statistics of the maximum for the tent map. Chaos. Solitons & Fractals 42(1), 604–608 (2009)

54. Bird, N., Vivaldi, F.: Periodic orbits of the sawtooth maps. Physica D: Nonlinear Phenomena 30(1-2), 164–176 (1988)

55. Lian, S., Sun, J., Wang, J., Wang, Z.: A chaotic stream cipher and the usage in video protection. Chaos, Solitons and Fractals 34, 851–859 (2007)

56. Lian, S., Liu, Z., Ren, Z., Wang, H.: Secure Media Distribution Scheme Based on Chaotic Neural Network. In: Liu, D., Fei, S., Hou, Z., Zhang, H., Sun, C. (eds.) ISNN 2007. LNCS, vol. 4492, pp. 79–87. Springer, Heidelberg (2007)

57. Qian, Q., Chen, Z., Yuan, Z.: Video compression and encryption based-on multiple chaotic system. In: Proceedings of the 3rd International Conference on Innovative Computing Information and Control, pp. 561–564. IEEE Computer Society, Washington, DC, USA (2008)

58. Papadimitriou, S., Bountis, T., Mavroudi, S., Bezerianos, A.: A probabilistic symmetric encryption scheme for very fast secure communication based on chaotic systems of difference. Int. J. Bifurcat Chaos 11(12), 3107–3115 (2001)

59. Socek, D., Culibrk, D.: On the security of a clipped. Hopfield neural network-based cryptosystem. In: Proceedings of the 7th Workshop on Multimedia and Security, pp. 71–76. ACM press, New York (2005)

60. Xiao, D., Liao, X.-F.: A combined hash and encryption scheme by chaotic neural network. In: Yin, F.-L., Wang, J., Guo, C. (eds.) ISNN 2004. LNCS, vol. 3174, pp. 13–28. Springer, Heidelberg (2004)

61. Lian, S., Chen, G., Cheung, A., Wang, Z.: A Chaotic-Neural-Network-Based Encryption Algorithm for JPEG2000 Encoded Images. In: Yin, F.-L., Wang, J., Guo, C. (eds.) ISNN 2004. LNCS, vol. 3174, pp. 627–632. Springer, Heidelberg (2004)

62. Lian, S.: Multimedia Content Protection Based on Chaotic Neural Networks. In: Nedjah, N., Abraham, A., de Macedo Mourelle, L. (eds.) Computational Intelligence in Information Assurance and Security. Springer, Heidelberg (2007)

63. Su, Z., Jiang, J., Lian, S., et al.: Selective encryption for g.729 speech using chaotic maps. In: Proceedings of International Conference on Multimedia Information Networking and Security, vol. 1, pp. 488–492. IEEE computer society, Wuhan (2009)

64. Fridrich, J.: Symmetric ciphers based on two-dimensional chaotic maps. International Journal of Bifurcation and Chaos 8(6), 1259–1284 (1998)

65. Li, Y.X., Tang, W.K.S., Chen, G.R.: Generating hyperchaos via state feedback control. Int. J. of Bifurcation and Chaos 15(10), 3367–3375 (2005)
66. Millerioux, G., Bloch, G., Amigo, J.M., Bastos, A., Anstett, F.: Real-time video communication secured by a chaotic key stream cipher. In: Proceedings of the European Conference on Circuit Theory and Design ECCTD 2003, pp. 245–248. IEEE, Krakow (2003)
67. Rueppel, R.A.: Analysis and design of stream ciphers. Springer, Heidelberg (1986)

Chapter 7
Cryptanalysis of Chaotic Ciphers

Ercan Solak

Işık University
Istanbul, Turkey
ercan@isikun.edu.tr

1 Introduction

Cryptanalysis is an integral part of any serious effort in designing secure encryption algorithms. Indeed, a cryptosystem is only as secure as the most powerful known attack that failed to break it. The situation is not different for chaos-based ciphers. Before attempting to design a new chaotic cipher, it is essential that the designers have a thorough grasp of the existing attacks and cryptanalysis tools.

There is a large variety of chaotic ciphers proposed in the literature. Consequently, their cryptanalyses come up with equally diverse attacks. Each attack tries to exploit weaknesses that are specific to the particular chaotic cipher. Thus, it is somewhat difficult to devise common non-trivial attacks that can be applied against a range of chaotic ciphers. On the other hand, such diversity of designs works against the security of the chaos-based ciphers. Rather than using well-analyzed and tested building blocks, there seems to be a general tendency to try novel and fancier structures, thus opening new venues for attacks.

If chaos cryptography is to make serious contributions to mainstream cryptography, we need to have more of analysis and less of design. Rather than trying to come up with new and interesting ways to incorporate chaos into encryption, the research effort should try to establish ground rules and primitive building blocks for the use of chaos in cryptography. This can only come through a rigorous cryptanalysis of existing proposals and by identifying the common weaknesses and pitfalls.

There have been a few noteworthy efforts in this direction. In particular, [Alvarez and Li, 2006] offer general observations about the flaws and weaknesses found in many chaotic encryption schemes. [Amigó et al., 2007, Masuda et al., 2006, Kocarev and Jakimoski, 2003, Dachselt and Schwarz, 2001] identify the building blocks that can be used in chaotic ciphers and random number generators. [Anstett et al., 2006] draws parallels between identifiability of dynamical systems and cryptanalysis. [Li et al., 2008]

establishes general attacks that can be launched against permutation-only chaotic image encryption algorithms.

In many proposals for chaotic ciphers, we observe a common tendency to use a subset of statistics in order to demonstrate the strength of the encryption. Although a necessary condition, good statistics are far from establishing good encryption. Indeed, any mildly sophisticated function produces good confusion and diffusion when applied in enough number of rounds. What statistics can not do, however, is hide the algebraic weaknesses inherent in the cipher. For example, even a linear block cipher will pass some easy statistical tests. Yet, a linear cipher can trivially be broken.

Therefore, it is crucially important to analyze the algebraic structure of a chaotic cipher and identify weak transformations.

A particular class of attacks against chaos based ciphers aims at bypassing the chaotic part of the cryptosystem. In this class, the encryption algorithm is expressed in an equivalent form in which the chaotic subsystems are replaced by a set of secret maps or parameters. In this way, the algebraic weaknesses in the rest of the algorithm are highlighted. This approach makes the whole system more amenable to cryptanalysis. In this chapter on the cryptanalysis of chaos-based ciphers, we illustrate the power of algebraic attacks on a number of different chaotic encryption algorithms.

In the next section, we examine the case of "inadvertently" linear ciphers. Such a cipher uses the nonlinear nature of chaos to generate some key parameters. However, the transformation from the plain image to the cipher image is linear.

The final part of the chapter illustrates the power of algebraic analysis in breaking chaotic ciphers.

2   Chaotic *Linear* Ciphers

Before we identify a few chaotic ciphers that turns out to be linear, we briefly show how a linear block cipher can be trivially broken.

Assume P and C are n-bit plaintext and ciphertext blocks, respectively. If the encryption transformation from P to C is linear, then it can be represented as a binary matrix multiplication

$$C = \mathbf{A}P, \tag{1}$$

where the matrix $\mathbf{A}$ is the secret mapping. For a known plaintext block P, an attacker can construct n linear equations

$$c_1 = a_{11}p_1 \oplus a_{12}p_2 \oplus \cdots \oplus a_{1n}p_n,$$
$$c_2 = a_{21}p_1 \oplus a_{22}p_2 \oplus \cdots \oplus a_{2n}p_n,$$
$$\vdots$$
$$c_n = a_{n1}p_1 \oplus a_{n2}p_2 \oplus \cdots \oplus a_{nn}p_n$$

for the entries a_{ij} of $\mathbf{A}$.

Using a set of n distinct known plaintext-ciphertext pairs, an attacker can construct n^2 linear equations for n^2 secret entries of $\mathbf{A}$. Solving these linear equations, the attacker easily breaks the cipher.

A common weakness in many chaotic ciphers is to use a set of well-known chaotic systems with secret system parameters to generate a linear transformation $\mathbf{A}$, which is then used as in (1). This creates a complex relationship between the chaotic system parameters and the resulting linear transformation. However, the attacker bypasses this complexity by attacking the linear transformation rather than trying to reveal the secret system parameters. The situation is illustrated in Fig. 1.
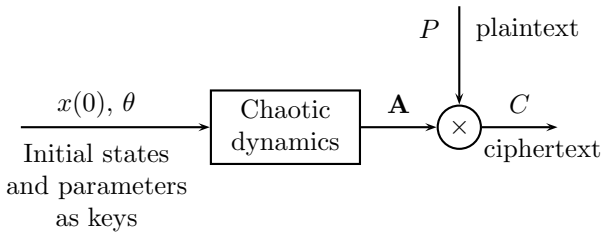
Fig. 1 A general structure of a chaotic *linear* block cipher.

Example 1. In the chaos-based image cipher proposed in [Guan et al., 2005], the encryption process consists of two parts. In the first part, the algorithm takes an image P and shuffles its pixels using Arnold Cat map. The second part of the algorithm changes the gray levels of the pixels using Chen's chaotic system.

Representing the image as a vector, the shuffling transformation can be represented as

$$S = \mathbf{A}P,$$

where $\mathbf{A}$ is a secret permutation matrix. For the second step of the encryption, Chen's chaotic system is used with secret parameters and initial values to generate a key vector, K. Thus, the encryption can be written as

$$C = \mathbf{A}P \oplus K. \tag{2}$$

Clearly, (2) is an affine linear equation. Assume that the attacker knows two plaintext-ciphertext image pairs $(P_1,\ C_1)$ and $(P_2,\ C_2)$. Let us define the differences as $\Delta P = P_1 \oplus P_2$ and $\Delta C = C_1 \oplus C_2$. Using (2), the attacker calculates

$$\Delta C = \mathbf{A}\Delta P$$

Going from ΔP to ΔC, there is only shuffling by the Arnold Cat map, which is a linear operation.

For a number of known plaintext-ciphertext differences, the attacker can find the secret $\mathbf{A}$. Once he reveals $\mathbf{A}$, he uses just one known pair (P, C) to calculate the secret K as

$$K = C \oplus \mathbf{A}P.$$

It is possible to improve the attack if one allows for chosen plaintexts. For more details, see [Çokal and Solak, 2009].

Although the attack is quite simple, it can be applied to a number of chaotic ciphers with only a few adaptations.

In [Patidar et al., 2009], a plaintext image P is encrypted in four steps. The first and the last steps involve adding chaotically generated key images K_1 and K_2. The second step linearly diffuses the pixel values in horizontal direction. The third step does the same in vertical direction. The two diffusions can be combined into one matrix multiplication. Thus, the whole encryption process becomes

$$C = \mathbf{A}(P \oplus K_1) \oplus K_2.$$

Clearly, this is a linear transformation. Moreover, the parameter A is not secret. This makes the whole scheme trivially weak. More details on the attack can be found in [Rhouma et al., 2010].

A general class of chaotic linear ciphers are shuffling-only image ciphers. In many cases, the shuffling parameters are generated by iterating one or more chaotic systems starting with secret initial conditions and parameters. In attacking these systems, the attacker aims to reveal the intermediate shuffling parameters rather than the chaotic system parameters. A recent example of such a cipher is proposed in [Huang and Nien, 2009], which is cryptanalyzed in [Solak et al., 2010b]. A general approach in attacking substitution-only image ciphers is given in [Li et al., 2008].

3   Algebraic Attacks

The mapping from the chaotic system parameters and initial conditions to its trajectories is highly nonlinear and complex. Still, when a chaotic system is used in encryption, the algebraic structures that it induces might be amenable to cryptanalysis. In the following discussion, we analyze three chaotic ciphers in order to illustrate the power of algebraic analysis in attacks.

3.1   *Reconstructing Small Permutations*

We first give a few facts about the powers of permutations over finite sets.

Definition 1. *[Fraleigh, 2002] An ordered orbit of a permutation π on a finite set is the ordered tuple $(a_0, a_1, \ldots, a_{n-1})$ such that $\pi(a_0) = a_1$, $\pi(a_1) = a_2, \cdots, \pi(a_{n-2}) = a_{n-1}$, $\pi(a_{n-1}) = a_0$. n is the length of the ordered-orbit.*

Theorem 1. *[Fraleigh, 2002] A permutation defined on a finite set partitions the set into disjoint ordered-orbits.*

Remark 1. Given a permutation π defined on a set V, determining its orbits is straightforward. We start from any element $a_0 \in V$ and form the orbit elements as $(a_0, \pi(a_0), \pi^2(a_0), \ldots, \pi^{n-1}(a_0))$ until $\pi^n(a_0) = a_0$. We then start over with an element not included in the orbits found so far. We continue forming orbits until we exhaust all the elements in the set V.

An example of a permutation over the set $\{a_0, a_1, \ldots, a_{10}\}$ is given in Fig. 2. Note that there are two orbits of lengths 5 and 6.
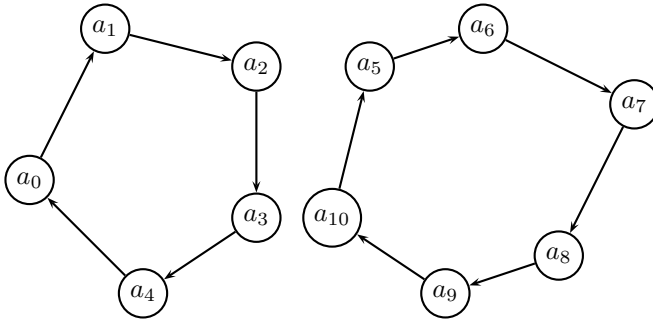
Fig. 2 A permutation with two orbits of lengths 5 and 6.

Note that if a_0 is an element in an orbit of length n in the permutation π, then, for all integers i,

$$\pi^i(a_0) = \pi^{i \bmod n}(a_0).$$

Lemma 1. *Let $\alpha = (a_0, a_1, \ldots, a_{n-1})$ be an orbit of length n in the permutation π, where $\gcd(n, r) = v$. Then, α is split into v equal length orbits in π^r.*

Lemma 2. *Let $\beta = (b_0, b_1, \ldots, b_{t-1})$ be the only orbit of length t in the permutation π^r. Then,*

$$\pi(b_j) = b_{(j+r^*) \bmod t}, \ 0 \le j < t,$$

where r^* is the multiplicative inverse of r in $\bmod\, t$, i.e. $rr^* \equiv 1 \pmod{t}$.

Remark 2. An immediate result of Lemma 1 and Lemma 2 is that if we have an orbit $\alpha = (a_0, a_1, \ldots, a_{n-1})$ in π such that $\gcd(n, r) = 1$, then in π^r, α is not split but is rather shuffled as

$$\beta = (a_0, a_{r \bmod n}, a_{(2r) \bmod n}, \ldots, a_{((n-1)r) \bmod n}).$$

Lemma 3. *Let $\beta = (b_0, b_1, \ldots, b_{t-1})$ be one of the q orbits of length t in the permutation π^r. Let v be the least divisor of r larger than 1. Assume that $q < v$. Then,*

$$\pi(b_j) = b_{(j+r^*) \bmod t}, \ 0 \le j < t, \tag{3}$$

where r^* is the multiplicative inverse of r in $\bmod\, t$, i.e. $rr^* \equiv 1 \pmod{t}$.

Lemma 4. *Let $\beta^{(1)} = (b_0^{(1)}, b_1^{(1)}, \ldots, b_{t-1}^{(1)})$ and $\beta^{(2)} = (b_0^{(2)}, b_1^{(2)}, \ldots, b_{t-1}^{(2)})$ be two orbits of length t in π^r. If $\pi(b_i^{(1)}) = b_j^{(2)}$ for some i, j then*

$$\pi(b_{(i+k) \bmod t}^{(1)}) = b_{(j+k) \bmod t}^{(2)}, \ 1 \le k < t.$$

For the proofs of these lemmas, see [Solak and Çokal, 2009].

An illustration of how orbits are shuffled and split in powers of permutation is given in Fig. 3. The graph shows the orbit structure of π^2 of the permutation π given in Fig. 2. Note that the length 5 orbit of π is only shuffled while its length 6 orbit is split into two length 3 orbits.
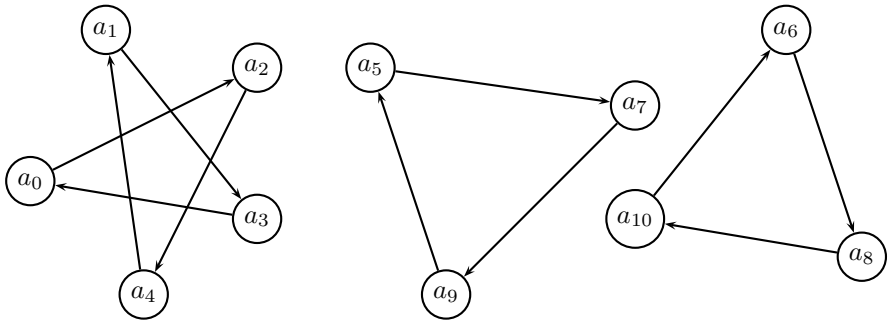
Fig. 3 Orbits of π^2 for the permutation π given in Fig. 2.

We know apply these properties of permutations to design algebraic attacks against two chaotic block ciphers.

3.2 Algebraic Attack on a Cryptosystem Based on Discretized Two-Dimensional Chaotic Maps

In the chaotic cipher proposed in [Xiang et al., 2007], plaintext and ciphertext sequences are partitioned into 16-bit blocks P_i, C_i, $1 \le i \le n$, as

$$\text{Plaintext}: \ P_1 P_2 \cdots P_n,$$
$$\text{Ciphertext}: \ C_1 C_2 \cdots C_n.$$

The key of the cryptosystem is the collection of the parameters (r, m, t, C_0, K_s, K_c). In [Xiang et al., 2007] this collection is defined as the master key. The master key is composed of the number of rounds r, the shift amount m, the number of iterations t, the initial value C_0, the subkey K_s and the collection of TDCM parameters K_c. Below, we explain how each part of the key is used in encryption.

A block key K_i is used in the encryption of plaintext block P_i. Initially, we assign

$$K_0 = K_s. \tag{4}$$

Before the encryption of block P_i, K_i is first updated as

$$K_i = \begin{cases} K_{i-1} \oplus C_{i-1} & \text{if } C_{i-1} \neq K_{i-1}, \\ K_{i-1} & \text{if } C_{i-1} = K_{i-1}. \end{cases} \tag{5}$$

The encryption of the ith block is given as

$$C_i = E(K_i, P_i), \tag{6}$$

where the function E involves the following round operations.

$$\begin{aligned} v_0 &= P_i, \\ v_j &= \sigma(v_{j-1} \oplus \text{ROL}(K_i, jm)), \ 1 \leq j \leq r, \\ C_i &= v_r. \end{aligned} \tag{7}$$

Here, v_j is the output of round j. Thus, v_r becomes the ciphertext. $\text{ROL}(\cdot, jm)$ denotes the circular left rotation of its argument by jm bits. The amount of circular left shifts depends on the number of rounds r and is given as

$$m = \begin{cases} \lfloor 16/r \rfloor & r \leq 16, \\ 1 & \text{else.} \end{cases} \tag{8}$$

The round function σ is a composition of a number maps and is given as

$$\sigma = w \circ z^{-1} \circ \text{TDCM}_{K_c}^t \circ z \circ S. \tag{9}$$

In (9), S represents the S-box substitution. S invertibly maps between 16-bit quantities. The S-box is designed to have desirable nonlinear properties, and its value is fixed (not secret) for the algorithm.

The map z is an invertible function that maps from 16-bit quantities to 2D vectors of integers. It maps the unsigned integer values corresponding to each byte of its argument to one of the integer coordinates in 2D discrete state space. The aim of z is to prepare a 2D initial state out of a given 16-bit quantity.

$\text{TDCM}_{K_c}^t$ denotes the t-times iteration of TDCM. K_c denotes the collection of the chaotic system parameters. The choice of the chaotic map is part of the algorithm design. In [Xiang et al., 2007], the standard map, the generalized

cat map, and the generalized baker map are considered. The chaotic map must be bijective in order to have an invertible encryption operation. The output of the chaotic system is passed through z^{-1} to map the final 2D state of TDCM to a 16-bit number.

The last mapping in w in (9) denotes the byte swap operation.

After the encryption of block i, the block key is once more updated as

$$K_i \leftarrow \mathrm{ROL}(K_i, rm). \tag{10}$$

Since K_i is 16-bits, the effective amount of rotation on K_i in this step is rm mod 16.

We now give a detailed cryptanalysis of the cipher.

The relation (8) fixes m once r is known. This removes the freedom in the choice of m, and effectively reduces the key length by 8 bits. Therefore, the shift amount m must be treated not as a key but rather as an internal parameter that is derived from the key.

Another reduction in effective key length is due to the way the secret parameter C_0 is used. Before the encryption of the first 16-bit block, the subkey K_s is updated by using (5). Hence, the value of K_1 used in the encryption of P_1 is $K_s \oplus C_0$. Consequently, we can treat $K_s \oplus C_0$ as one secret parameter rather than two distinct parameters, K_s and C_0. Indeed, any pair of C_0 and K_s values that yields the same XOR value results in identical encryption functions. This fact reduces the effective key length by another 16 bits. In the subsequent sections, we assume without loss of generality that $C_0 = \texttt{0x0000}$.

After noting these reductions in the effective key space, we now give an algebraic break of the cipher. We first demonstrate how an attacker can reveal K_s without having access to the rest of the key parameters.

In out attacks, we assume that the attacker knows the number of rounds r. This is not a very restrictive assumption. Since r is represented with 8 bits, it can only take one of 255 possible nonzero values. The attacks that we develop in this and the next section have very low computational requirements. In the case when the attacker does not know the value of r, he tries all 255 possible values with the attacks described here.

Revealing K_s

To illustrate the method of the attack, we only analyze the case when $rm \equiv 0 \bmod 16$. For the details of the attack for the case $rm \not\equiv 0 \bmod 16$, see [Solak and Çokal, 2008].

We assume that the attacker does not know the TDCM parameters, so he does not know the function E in (6).

Assume that the first two ciphertext blocks are the same and given as

$$C_1 = C_2 = j. \tag{11}$$

If $j = K_s$, using (4), (5), (6) and (10), we have

$$j = E(K_s, P_1), \ j = E(K_s, P_2).$$

So, by the invertibility of E for fixed K_s, we have $P_1 = P_2$.
   If $j \neq K_s$, we have

$$j = E(K_s, P_1), \ j = E(K_s \oplus j, P_2).$$

In this case, most probably $P_1 \neq P_2$. The difference in two cases indicates
that the equality of P_1 and P_2 is a good test on whether $K_s = j$.
   The attack on K_s proceeds as follows. The attacker chooses a 16-bit num-
ber j. He requests plaintexts for a two-block ciphertext $C_1 C_2$ chosen as in
(11). He compares these plaintext blocks P_1 and P_2. If they are equal, then
j is a candidate for the secret K_s. The attacker repeats this for all the 16-bit
j values and records candidates for K_s. A total of $2^{16} - 1$ trials are made.
   It may happen that the attacker obtains $P_1 = P_2$ even when $j \neq K_s$. This
is because we might have $E(K_1, P) = E(K_2, P)$ for some $K_1 \neq K_2$, and P. In
order to eliminate the false keys, the attacker performs the following further
tests.
   Assume that the attacker has two candidates j_1 and j_2 for the subkey K_s.
From his previous attempt at determining the keys, the attacker knows P_1
and P_2 which satisfy

$$j_1 = E(K_s, P_1), \ j_2 = E(K_s, P_2). \tag{12}$$

The attacker now chooses the new ciphertext blocks $\overline{C}_1$ and $\overline{C}_2$ as $\overline{C}_1 = j_1$
and $\overline{C}_2 = j_2$. He obtains the corresponding plaintext blocks $\overline{P}_1$ and $\overline{P}_2$.
There are two cases for the validity of j_1. Let us see how $\overline{P}_1$ and $\overline{P}_2$ differ
for each case.
   Case 1: $j_1 = K_s$: Using (4), (5), (6) and (10), we find that

$$j_1 = E(K_s, \overline{P}_1), \ j_2 = E(K_s, \overline{P}_2).$$

Comparing this with (12), we obtain $\overline{P}_1 = P_1$ and $\overline{P}_2 = P_2$.
   Case 2: $j_1 \neq K_s$: This time we find,

$$j_1 = E(K_s, \overline{P}_1), \ j_2 = E(K_s \oplus j_1, \overline{P}_2).$$

Comparing this with (12), we conclude $\overline{P}_1 = P_1$ and $\overline{P}_2$ is a random 16-bit
number.
   In both cases, $\overline{P}_1 = P_1$. However, only in the first case we are guaranteed
to have $\overline{P}_2 = P_2$. In the second case, we might have $\overline{P}_2 = P_2$ even when
$j_1 \neq K_s$. So, if $\overline{P}_2 \neq P_2$ the test is conclusive and $j_1 \neq K_s$. If $\overline{P}_2 = P_2$ the
test is inconclusive.
   This test gives the attacker a method to eliminate the false subkeys
among the candidates. Assume that attacker has determined q candidates,

$\{j_1, j_2, \cdots, j_q\}$ for the subkey K_s. To eliminate the false subkeys, he chooses a pair of candidates j_{i_1} and j_{i_2} and applies the test as explained. In this way, he eliminates j_{i_1} if the test is conclusive. Otherwise, he chooses a different pair and repeats the test. The attack on K_s successfully terminates when there remains only one candidate for the subkey.

Once the attacker knows K_s, he proceeds to reveal the other parameters t and K_c. We assume that the attacker already knows the number of rounds r. Hence, by the relation (8), he also knows the shift amount m. The only secret parameters to be revealed are K_c, the collection of the TDCM parameters and t, the number of times the TDCM is iterated. When the block key K_i and r are fixed, the parameters K_c and t characterize the function E.

A brute force attack on K_c and t has to try all their values against a known plaintext-ciphertext pair. We now give a general attack that requires on the order of 2^{16} chosen ciphertext/plaintext blocks and very little amount of computation. Moreover, the computational complexity of our attack does not depend on the lengths of the keys K_c and t.

Sampling E

We first note that, for a fixed K_i of his choice, the attacker can choose either one of C or P in the relation

$$C = E(K_i, P), \tag{13}$$

and obtain the other. To see how this can be done, let us write the encryption equations for a sequence of two blocks of plaintext, $P_1 P_2$.

$$C_1 = E(K_s, P_1),$$
$$C_2 = E(\mathrm{ROL}(K_s, rm) \oplus C_1, P_2). \tag{14}$$

Here, we assume that $\mathrm{ROL}(K_s, rm) \oplus C_1 \neq 0$.

If C_1 is chosen as $C_1 = K_i \oplus \mathrm{ROL}(K_s, rm)$, (14) becomes

$$C_2 = E(K_i, P_2).$$

So, the attacker first chooses a single block ciphertext with $C_1 = K_i \oplus \mathrm{ROL}(K_s, rm)$ and obtains the plaintext P_1. If he wants to choose C and obtain the corresponding P in (13), he next chooses the ciphertext sequence $C_1 C$ and obtains P_2 as his desired plaintext block P. If, instead, he wants to know C for a particular P in (13), he chooses the plaintext sequence $P_1 P$ and obtains C_2 as his desired ciphertext block C.

Thus, an attacker can freely choose K_i, and sample the function $C = E(K_i, P)$ at arbitrary points (P, C) of his choice. We will see that this ability lets the attacker determine the internal secret parameters of the encryption function E.

Since the functions w, z, S are fixed and the attacker already knows r, m, and K_i, revealing the secret parameters t, K_c is equivalent to revealing the

function σ in (7). Namely, once the attacker knows σ, he can encrypt/decrypt any plaintext/ciphertext sequences as if he knew the parameters t and K_c. Below we describe three attacks that reveal the function σ.

We first note that σ is a permutation over the set $\{0, 1, \ldots, 2^{16} - 1\}$. We now show how particular choices of K_i lets an attacker reveal portions of σ.

Permutation orbit attack

Let us choose K_i such that

$$\mathrm{ROL}(K_i, m) = K_i. \tag{15}$$

Namely, K_i is m-bit rotation invariant.

When we use (15) in (7), we obtain

$$v_j = \sigma(v_{j-1} \oplus K_i), \ 1 \leq j \leq r.$$

Defining a new permutation π as

$$\pi(x) = \sigma(x \oplus K_i) \tag{16}$$

for $x \in \{0, 1, \ldots, 2^{16} - 1\}$, we can express the relation between P and C as

$$C = \underbrace{\pi \circ \pi \circ \cdots \circ \pi}_{r \text{ times}}(P) = \pi^r(P).$$

If the attacker reveals the value Y of π at P so that $Y = \pi(P)$, he reveals that the value of σ at $P \oplus K_i$ is Y, i.e. $Y = \sigma(P \oplus K_i)$.

To illustrate the choice of K_i that turns the function E into the $r-$power of a permutation, let us take $m = 2$. In this case, the nonzero K_i values that satisfy (15) are 0101010101010101 (0x5555), 1010101010101010 (0xAAAA) and 1111111111111111 (0xFFFF) . If $m = 1$, the only nonzero K_i that satisfies (15) is (0xFFFF). Note that by (5), K_i can never be zero.

Also note that for each value of K_i that satisfies (15), we obtain a different permutation π.

Using the sampling method given above, the attacker can obtain $\pi^r(P)$ for every P in $\{0, 1, \ldots, 2^{16} - 1\}$. Hence, he can reveal the permutation π^r.

For a given m, the attacker determines the keys K_i that satisfy (15). Assume that there are k such keys. For each such K_i^j, $1 \leq j \leq k$, the attacker finds $E(K_i^j, P)$ for all $P \in \{0, 1, \ldots, 2^{16} - 1\}$. This is in fact π_j^r that corresponds to K_i^j i.e. $\pi_j^r(x) = E(K_i, x)$, for every x. The attacker then determines the orbit structure of π_j^r. Then he starts partially revealing π_j. He performs the following two steps for each K_i^j.

1. Use lone orbits in π_j^r. If there is a lone orbit of length n_1 in π_j^r, use Lemma 2 to reveal n_1 points in π_j. From those, reveal n_1 points of σ using (16).

2. Look for a collection of the same length orbits in π_j^r. If the size q of the collection is less than the least divisor of r larger than 1, then use Lemma 3 to reveal q $_2$ more points in π_j, where n_2 is the length of an orbit in the collection. Again, use (16) to reveal qn_2 points in σ.

Let $R_j \subset \{0, 1, \ldots, 2^{16} - 1\}$ be the points for which $\sigma(R_j)$ is revealed using Steps 1 and 2 above with K_i^j for $1 \le j \le k$. Let $R = R_1 \cup R_2 \cup \cdots \cup R_k$. Let $x \in R \backslash R_j$. Namely, the attacker knows $y = \sigma(x)$ but this point is revealed in either Step 1 or Step 2 for a key other than K_i^j. Then, π_j^r contains two same length orbits β^1 and β^2 such that $x \oplus K_i^j \in \beta^1$ and $y \in \beta^2$. These orbits were not used in the Step 2 for K_i^j above otherwise we would have $x \in R_j$. Hence, $\pi_j(x \oplus K_i^j) = y$. Then, the attacker uses Lemma 4 with β^1 and β^2 to reveal some more points on σ. This, in turn, adds points to R. The procedure is repeated until there are no points x satisfying $x \in R \backslash R_j$.

Furthermore, if the attacker uses any of the attacks explained below and somehow obtains the new knowledge of a sample point in π_j, and the point maps across two orbits of length n_3 in π_j^r, then he uses Lemma 4 to reveal $n_3 - 1$ more points in π_j.

Expansion attack

In the previous section we described an attack that partially reveals σ. We now describe an attack that works with a partially revealed permutation σ. This attack is applied together with the permutation orbit attack.

Assume that R and U are two disjoint subsets of $\{0, 1, \ldots, 2^{16} - 1\}$ such that $R \cup U = \{0, 1, \ldots, 2^{16} - 1\}$. Also assume that the attacker knows the value of $\sigma(x)$ for every $x \in R$ and he does not know the value of $\sigma(x)$ for any $x \in U$. In other words, R denotes the revealed portion of the domain of σ, and U denotes the unrevealed portion.

Assume that the attacker knows the triple (C, P, K_i) such that $C = E(K_i, P)$. Assume that $C \notin \sigma(R)$ i.e. he does not know the value which is mapped by σ to C. He now tries to carry out the calculation (7). He starts out with $v_0 = P$. He can calculate v_1 if $v_0 \oplus \mathrm{ROL}(K_i, m) \in R$. Once he knows v_1, he can calculate v_2 if $v_1 \oplus \mathrm{ROL}(K_i, 2m) \in R$. Assume that he continues in this fashion, reaches the penultimate step and calculates v_{r-1}. Obviously, $v_{r-1} \oplus \mathrm{ROL}(K_i, rm) \notin R$ because otherwise we would have $C = v_r = \sigma(v_{r-1} \oplus \mathrm{ROL}(K_i, rm)) \in \sigma(R)$ which contradicts the assumption. But this means that the attacker has just revealed the value of the map σ at a new point $v_{r-1} \oplus \mathrm{ROL}(K_i, rm)$ because he already knows C. Thus, if the attacker reaches the last step while staying in the partially revealed portion R, he expands R by one point and shrinks U by one point.

Every time the expansion attack succeeds and the attacker reveals a new point on the map σ, he uses Lemma 4 to check if this corresponds to mapping across two different same-length orbits in π^r. If so, the revealed portion R

is expanded even more. This, in turn, increases the probability that next application of the expansion attack succeeds.

Skipping attack

Using (8), we see that when $r \geq 9$, we have $m = 1$. So the attacker can use only $K_i = $0xFFFF in the permutation orbit attack. Moreover, when r is an even number, its smallest divisor is 2 and he can not use Lemma 3. This adversely affects the size of the revealed set R that can be used in the expansion attack. We now describe another attack that works with $r \geq 9$ and even. The attack relies on deriving a new permutation by skipping over odd rounds in the expression of E in (7).

Assume that a nonzero K_i satisfies

$$\mathrm{ROL}(K_i, 2) = K_i. \tag{17}$$

Using (17) with (7) and substituting odd round outputs into even round expressions, we obtain

$$
\begin{aligned}
v_0 &= P, \\
v_2 &= \sigma(\sigma(v_0 \oplus \mathrm{ROL}(K_i, 1)) \oplus K_i), \\
v_4 &= \sigma(\sigma(v_2 \oplus \mathrm{ROL}(K_i, 1)) \oplus K_i), \\
&\;\;\vdots \\
v_r &= \sigma(\sigma(v_{r-2} \oplus \mathrm{ROL}(K_i, 1)) \oplus K_i), \\
C &= v_r.
\end{aligned}
$$

Defining a new permutation γ as

$$\gamma(x) = \sigma(\sigma(x \oplus \mathrm{ROL}(K_i, 1)) \oplus K_i), \tag{18}$$

we can express the relation between C and P as

$$C = \gamma^{r/2}(P).$$

First, the attacker applies the permutation orbit attack with $K_i = $0xFFFF. In doing so, he obtains the permutation π^r, and using Lemma 2 and Lemma 3, he reveals a portion of σ. Let R denote the revealed portion of the map σ.

The skipping attack proceeds as follows. As in the permutation orbit attack, by choosing every $P \in \{0, 1, \ldots, 2^{16}-1\}$ and obtaining their corresponding ciphertext block with $K_i^1 = $0x5555 and $K_i^2 = $0xAAAA satisfying (17), the attacker finds the permutations $\gamma_1^{r/2}$ and $\gamma_2^{r/2}$. For each $\gamma_j^{r/2}$, $j = 1, 2$, the attacker uses its orbit structure to reveal a portion of γ_j.

Assume the attacker has determined a pair (x, y) such that $y = \gamma_j(x)$ for some j. Hence, he knows that

$$y = \sigma(\sigma(x \oplus \text{ROL}(K_i^j, 1)) \oplus K_i^j). \tag{19}$$

There are two ways the attacker can use (19) to reveal a new point on σ. If $x \oplus \text{ROL}(K_i^j, 1) \in R$ and $y \notin \sigma(R)$, the attacker reveals the value of the map σ at $\sigma(x \oplus \text{ROL}(K_i^j, 1)) \oplus K_i^j$ as y. On the other hand, if $y \in \sigma(R)$ and $x \oplus \text{ROL}(K_i^j, 1) \notin R$, the attacker reveals the value of the map σ at $x \oplus \text{ROL}(K_i^j, 1)$ as $\sigma^{-1}(y) \oplus K_i^j$.

Thus, with the skipping attack, the attacker reveals some new points on the map σ. He subsequently uses Lemma 4 to check if these new points correspond to mappings across two different orbits in π^r that were not used in the permutation orbit attack. If so, the revealed portion R is expanded even more.

Example 2. In the first example, we used the cryptosystem with secret parameters $r = 5$, $m = 3$, $t = 12$, $C_0 =$0x4ED3, $K_s =$0x8F4C. By the equivalence explained in above, this is equivalent to $K_s =$0xC19F=0x4ED3$\oplus$0x8F4C and $C_0 =$0x0000, We used the standard map as TDCM. The secret TDCM parameter is $K_c = 53246$.

Since $m = 3$, we can apply the permutation orbit attack only with $K_i^1 =$0xFFFF. We obtain the orbit structure of π_1^5 as $(1, 53712)$, $(1, 6432)$, $(5, 779)$, $(1, 699)$, $(1, 449)$, $(1, 252)$, $(1, 72)$, $(5, 5)$. Here a pair (q, n) means that there are q orbits of length n.

We apply Lemma 2 to lone orbits of length 53712, 6432, 699, 449, 252 and 72 in π_1^5 to reveal 61616 entries in σ. This corresponds to 94.02% of the map σ.

We saw that $1 \notin \sigma(R)$. So, we choose $C = 1$ in the expansion attack. We try $K_i = 1$ and find $P = 65082$. The expansion attack for these values indeed succeeds and we find $1 = \sigma(680)$.

Now, we go back to the result of permutation orbit attack. Searching for $680\oplus$0xFFFF in the cycles of π_1^5, we see that it is mapped across two cycles of length 779. Using this sample point with Lemma 4, we reveal 779 new points in σ. Thus, the revealed set R gets bigger by 779 new points. Hence, a new expansion attack is even more likely to succeed. Repeating the attack with 9 more unrevealed ciphertext blocks with the same $K_i = 1$, we reveal the whole map σ.

3.3 Algebraic Cryptanalysis of a Chaotic Cipher Based on Chaotic Map Lattices

In the image encryption algorithm proposed in [Pisarchik et al., 2006], the plaintext is the vector $c \in \mathbf{Z}_{256}^m$ obtained by the usual row-scan of an $N \times M$ image, where m is the total number of pixels, i.e. $m = NM$. Here, $\mathbf{Z}_{256}$ denotes the set $\{0, 1, 2, \ldots, 255\}$ of integers which are represented by 8-bit pixels. The algorithm encrypts plaintext c in three steps; D/A conversion, chained chaotic iteration and A/D conversion.

1. D/A conversion: each integer pixel value c_i is mapped to one of 256 distinct real values x_i in the chaotic attractor $\Omega = (x_{\min}, x_{\max})$ for the logistic map

$$f(u) = au(1 - u),$$

using

$$x_i = g_1(c_i) = x_{\min} + (x_{\max} - x_{\min})\frac{c_i}{255}, \quad 1 \leq i \leq m, \tag{20}$$

where $x_{\min} = (4a^2 - a^3)/16$ and $x_{\max} = a/4$.

2. Chained chaotic iteration: the real values x_i are transformed using repeated chaotic iteration as follows. We first initialize cycle 0 values as $y_i^{(0)} = x_i$, $1 \leq i \leq m$. The transformation for the jth cycle is given as

$$
\begin{aligned}
y_1^{(j)} &= A(f^n(y_m^{(j-1)}) + y_1^{(j-1)}), \\
y_i^{(j)} &= A(f^n(y_{i-1}^{(j)}) + y_i^{(j-1)}), \quad i \geq 2, \ 1 \leq j \leq r,
\end{aligned}
\tag{21}
$$

where the function $A : (2x_{\min}, 2x_{\max}) \rightarrow \Omega$ guarantees that the LHS of (21) falls within the attractor. The plot of A is given in Fig. 4.
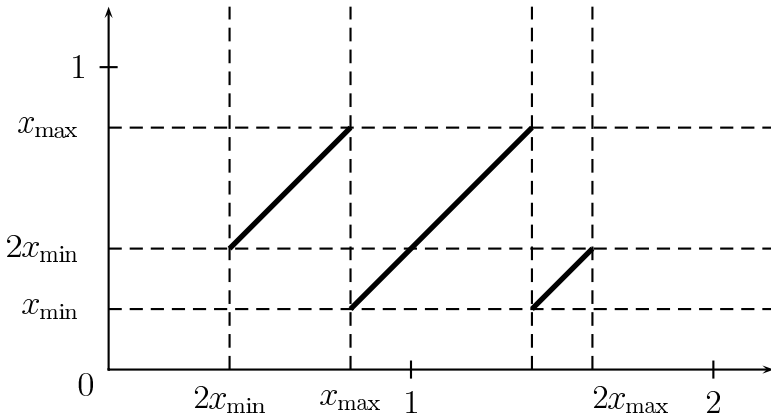

Fig. 4 The plot of the function $A : (2x_{\min}, 2x_{\max}) \rightarrow (x_{\min}, x_{\max})$. The function wraps around the attractor like modulus.

In (21), r denotes the number of cycles (rounds) in the encryption. Note that the logistic map f is iterated n times starting with the initial value $y_{i-1}^{(j)}$ for $i \geq 2$ and with $y_m^{(j-1)}$ for $i = 1$. The number of iterations n is part of the secret key.

3. A/D conversion: each $y_i^{(r)}$ is mapped back to an integer d_i in $\mathbf{Z}_{256}^m$ using

$$d_i = g_2(y_i^{(r)}) = \text{round}\left[(y_i^{(r)} - x_{\min})\frac{255}{x_{\max} - x_{\min}}\right]. \tag{22}$$

The vector $d \in \mathbf{Z}_{256}^m$ is the ciphertext.

In the subsequent discussion, we explain the attack given in [Solak and okal, 2011].

Equivalent representation

Here, we give the equivalent representation of the algorithm so that all the operations are done in $\mathbf{Z}_{256}$ and the secret quantities are some unknown permutations.

Note that g_1 and g_2 denote the D/A and A/D conversion functions in (20) and (22), respectively. For one round of encryption, we can write (21) as

$$
\begin{aligned}
d_i = g_2(y_i) &= g_2(A(f^n(y_{i-1}) + y_i)), \\
&= g_2(A(f^n(g_1(d_{i-1})) + g_1(c_i))), \quad 2 \le i \le m.
\end{aligned} \tag{23}
$$

Note that the mapping in (23) is from the pair $(d_{i-1}, c_i) \in \mathbf{Z}_{256} \times \mathbf{Z}_{256}$ to $d_i \in \mathbf{Z}_{256}$. Let us denote this map as $s : \mathbf{Z}_{256} \times \mathbf{Z}_{256} \to \mathbf{Z}_{256}$.

Given the secret quantities a and n, one can calculate the map s as

$$
s(i, j) = g_2(A(f^n(g_1(i)) + g_1(j))), \ 0 \le i, j \le 255. \tag{24}
$$

Now, we can write the single round encryption as

$$
\begin{aligned}
d_1 &= s(c_m, c_1), \\
d_i &= s(d_{i-1}, c_i), \ 2 \le i \le m.
\end{aligned} \tag{25}
$$

Similarly, we can trivially extend this expression for arbitrary number of rounds r. In this new expression of the algorithm, the equivalent secret quantities are the map s and the number of rounds r. However, the number of rounds is a small number in the range of 10. Thus, it can be safely assumed that the attacker knows r. Even when the attacker does not know r, he can try several values for r and apply the rest of the attack for the tried r. If the attack succeeds then the attacker has found the correct r.

In the next section, we give the attack that recovers the secret map s, assuming that r is known. The attack is first given in [Solak and okal, 2011].

Recovering s

Assume that the attacker chooses a two pixel image (c_1, c_2) as plaintext and obtains the corresponding ciphertext (d_1, d_2) for a single round. Using (25) with $m = 2$, we obtain

$$
\begin{aligned}
d_1 &= s(c_2, c_1), \\
d_2 &= s(d_1, c_2).
\end{aligned} \tag{26}
$$

Thus, (26) defines a function $\pi : \mathbf{Z}_{256} \times \mathbf{Z}_{256} \rightarrow \mathbf{Z}_{256} \times \mathbf{Z}_{256}$, $\pi((c_1, c_2)) = (d_1, d_2)$. Since the encryption is invertible, π is a permutation over the set $\mathbf{Z}_{256} \times \mathbf{Z}_{256}$.

Note that if attacker knows a point $(d_1, d_2) = \pi((c_1, c_2))$ on the permutation, then using (26), he can reveal the map s on two points (c_2, c_1) and (d_1, c_2).

If π is a single round encryption, then r round encryption becomes π^r. Hence, for his chosen plaintext image (c_1, c_2), the attacker observes $\pi^r((c_1, c_2))$. Choosing all of the 2^{16} possible 2-pixel plaintexts one by one and obtaining their corresponding ciphertexts, the attacker constructs the permutation π^r. Using the results given at the start of this section, the attacker reveals portions of π. Using the known points on π, the attacker finally recovers the secret map s.

We now give the details of the attack.

Permutation orbit attack

Once the attacker obtains π^r, he calculates its orbit structure using the procedure in Remark 1. Given the orbit structure of π^r, he starts by using the orbits that are shuffled going from π to π^r. The attacker uses such orbits in two distinct categories.

1. Look for lone orbits in π^r: If there is a lone orbit of length t_1 in π^r, use Lemma 2 to reveal t_1 points in π. From those, reveal at most $2t_1$ points of s using (26). Hence, if $\beta = (b_0, b_1, \ldots, b_{t_1-1})$ is a lone orbit of π^r, we can reveal some of the points on s for $0 \leq j < t_1$ as

$$s(b_{j,2}, b_{j,1}) = b_{(j+r^*) \bmod t_1, 1},$$
$$s(b_{(j+r^*) \bmod t_1, 1}, b_{j,2}) = b_{(j+r^*) \bmod t_1, 2}.$$

Note that each b_j is a pair $(b_{j,1}, b_{j,2})$, corresponding to a 2-pixel image.
2. Look for a collection of the same length orbits in π^r: If the size q of the collection is less than the least divisor of r larger than 1, then use Lemma 3 to reveal qt_2 more points in π, where t_2 is the length of an orbit in the collection. Again, use (26) to reveal at most $2qt_2$ new points in s.

Using the permutation attack, the attacker recovers a portion of the map s. If the portion is the whole, then the attack concludes successfully. If there are still unrevealed portions of s, the attacker performs the following consistency checks on the orbits not used in the permutation attack.

Consistency check

Suppose there are $q > 1$ orbits of length t_3 among the orbits of π^r and that none of these orbits were used in the permutation orbit attack. We now give consistency checks that can be applied to these orbits in order to reveal more points on the partially revealed map s.

Let β be one of those q orbits in π^r. There are two ways such a β might occur in π^r. One way is that β might have been obtained by the split of a larger orbit in π. The other possibility is that β was obtained by the shuffling of an orbit of the same length in π, see Remark 2.

We first test if latter is the case.

Assume that $\beta = (b_0, b_1, \ldots, b_{t_3-1})$ was obtained by the shuffling of an orbit of π. In this case, $\gcd(n_3, r) = 1$. Note that each b_j is a pair $(b_{j,1}, b_{j,2}) \in \mathbf{Z}_{256} \times \mathbf{Z}_{256}$. By Lemma 2, $\pi(b_j) = b_{(j+r^*) \bmod t_3}$, $0 \le j < n_3$. Thus, we conclude that, for $0 \le j < t_3$,

$$s(b_{j,2}, b_{j,1}) = b_{(j+r^*) \bmod t_3, 1},$$
$$s(b_{(j+r^*) \bmod t_3, 1}, b_{j,2}) = b_{(j+r^*) \bmod t_3, 2}.$$

Thus, on the assumption that β was obtained by shuffling, we reveal possibly $2t_3$ new points of the map s. However, if the assumption was wrong, then we expect to encounter inconsistencies. The newly revealed points might conflict with the already revealed points on s. Also, they might conflict among themselves.

To better see how two kinds of conflicting values might arise, let us assume that the attacker already knows $x, y, z \in \mathbf{Z}_{256}$ such that $s(x, y) = z$. If, for some j, $b_{j,2} = x$ and $b_{j,1} = y$ but $b_{(j+r^*) \bmod t_3, 1} \ne z$, then we have the conflict of the first kind, i.e. the newly revealed point conflicts with the already known point.

On the other hand, if we have j_1 and j_2 such that $b_{j_1,2} = b_{(j_2+r^*) \bmod t_3, 1}$ and $b_{j_1,1} = b_{j_2,2}$ but $b_{(j_1+r^*) \bmod t_3, 1} \ne b_{(j_2+r^*) \bmod t_3, 2}$, then we have newly revealed points conflicting among themselves.

The attacker can test both conflicts together. For every set of newly revealed points, he tries to add these to the map. If he fails due to a conflict with the already known portion, he concludes that β was not obtained by a simple shuffling, but instead was obtained by the split of a larger orbit in π.

By going through all the orbits not used in the permutation attack, and testing if they were obtained by simple shuffles, the attacker enlarges the revealed portion of s.

Now, the attacker is left with sets of orbits which are certainly obtained by the split of larger orbits in π. Let $\beta^{(1)}$ and $\beta^{(2)}$ be two such orbits of the same length t_4. We cannot directly use Lemma 1 because it is still possible that they were obtained by the split of different orbits in π.

In order to better see how this can happen, assume π has two orbits of length 10 and 15 and that $r = 6$. Then, by Lemma 1, in π^5, the length 10 orbit will be split into two length 5 orbits and lenth 15 orbit will be split into three length 5 orbits. Hence, in π^5, we see length 5 orbits coming from the split of different orbits.

Even if $\beta^{(1)}$ and $\beta^{(2)}$ come from the split of the same orbit in π, we may not directly use Lemma 4, because we lack a sample point mapping from one orbit to another.

Hence, the test for the second case proceeds as follows. The attacker chooses two same length orbits $\beta^{(1)}$ and $\beta^{(2)}$ in π^r. Let n_4 be the common length of these two orbits. He assumes that $\beta^{(1)}$ and $\beta^{(2)}$ come from the split of the same larger orbit in π and that there exist two integers $0 \leq i, j < t_4$ such that $b_i^{(1)} \in \beta^{(1)}$, $b_j^{(2)} \in \beta^{(2)}$ and $\pi(b_i^{(1)}) = b_j^{(2)}$. Fixing $i = 0$, he tries every j, $0 \leq j < t_4$, each time assuming that $\pi(b_0^{(1)}) = b_j^{(2)}$. If $\beta^{(1)}$ and $\beta^{(2)}$ are consecutive splits of a larger orbit in π, then there is such a j. If the attacker hits upon the correct j, he uses Lemma 4 and possibly reveals $2t_4$ new points on the map s as

$$s(b_{0,2}^{(1)}, b_{0,1}^{(1)}) = b_{j,1}^{(2)}, \ 0 \leq j < t_4,$$
$$s(b_{j,1}^{(2)}, b_{0,2}^{(1)}) = b_{j,2}^{(2)}, \ 0 \leq j < t_4.$$

On the other hand, if the attacker encounters an inconsistency with the already revealed portion of the map s, he discards j. If all the j's in $0 \leq j < t_4$ are discarded as such, then either $\beta^{(1)}$ and $\beta^{(2)}$ do not come from the same orbit π, or they come from the same orbit but their ordering was wrong, i.e. they are not consecutive splits.

By trying all the ordered pairs of orbits of the same length, the attacker is highly likely to eliminate the wrong assumptions with inconsistencies and reveal whole of the map s.

Complexity of the attack

Once the attacker obtains the permutation π^r, it takes only 2^{16} lookups to construct the orbit structure of π^r. The computational complexity of the rest of the attack depends on the orbit structure of the permutation π^r.

For a random permutation over the set $\{1, 2, \ldots, n\}$, the expected number of orbits is approximately $\log n$, [Lovasz, 2007, p. 227]. In our case $n = 2^{16}$, so we expect to have about 11 orbits in π. In the worst case, all orbits are split into r smaller orbits in π^r and we expect to have about $11r$ split orbits in π^r. If we were to check all pairs of orbits in π^r for consistency, we would perform about $121r^2$ consistency checks. Consistency checks can be done by a fixed number of lookups and comparisons. Let C denote the fixed cost of one consistency check for an orbit pair. For each pairing of two orbits of length L, we have to perform the consistency check for L shift amounts. The average orbit length for the original permutation π is $n/\log n$. Hence, for the average case with $n = 2^{16}$, we can take L as 5960. The split orbits in π^r will have an average length of $5960/r$.

Thus, the average complexity of the attack is $2^{16} + 121 \times rC \times 5960$ lookup or comparison operations. For a particular case of $r = 5$, $C = 20$, the attack takes about 10^8 basic operations on average.

3.4   Cryptanalysis of Fridrich's Image Cipher

Fridrich's cipher proposed in [Fridrich, 1998] is one of the earliest chaotic image encryption algorithms. The following discussion is a summary of the cryptanalysis given in [Solak et al., 2010a].

The plaintext P is an $M \times N$ grayscale image, where each pixel is represented using a byte. The image is first vectorized using the usual row-scan. Let $p \in S^n$ represent this vectorized image, where $S = \{0, 1, \ldots 255\}$ and $n = NM$. Thus, the plaintext is the vector $p = [p_1 \; p_2 \; \cdots p_n]$.

Each round consists of two steps. In the first step, p is shuffled using a secret permutation. Let b denote this secret permutation defined on the set $\{1, 2, \ldots, n\}$. Let us denote the shuffled vector by f. The relation between the shuffled vector f and the vectorized plaintext p can be expressed as

$$f_i = p_{b(i)}, \; 1 \le i \le n. \tag{27}$$

Namely, the shuffled pixel at position i is obtained from the original pixel at position $b(i)$.

In the second step of the round, f is passed through a nonlinear function as

$$c_i = f_i + g(c_{i-1}) + h_i \bmod 256, \; 1 \le i \le n, \tag{28}$$

where $g : S \to S$ is a fixed nonlinear function and $h \in S^n$ is a fixed vector. In (28), c_0 is taken to be a fixed system parameter.

These two step are repeated for R rounds. In [Fridrich, 1998], $R = 10$ is suggested for good diffusion and confusion properties.

Combining (27) and (28), we obtain one round encryption as

$$c_i = p_{b(i)} + g(c_{i-1}) + h_i \bmod 256, \; 1 \le i \le n. \tag{29}$$

The decryption for a single round is defined as follows. Let u be the inverse of b, so that

$$j = b(i) \; \Leftrightarrow \; i = u(j). \tag{30}$$

Using (30) in (29), we obtain

$$p_j = c_{u(j)} - g(c_{u(j)-1}) - h_{u(j)} \bmod 256. \tag{31}$$

For $i = 1$, we have

$$c_1 = p_{b(1)} + g(c_0) + h_1 \bmod 256.$$

The secret component of the algorithm is the permutation p. A set of secret keys are used in a chaotic system to generate this permutation. It is desirable that the permutation shows good diffusion properties in order to hide local correlations in an image. For example, in one of the schemes proposed in [Fridrich, 1998], the original image P is partitioned and Baker map applied

to each partition to obtain the permutation. In this case, the set of keys are the boundaries where the image is partitioned. It is possible to use other schemes to generate a permutation. Our attack is general and applies to all of these cases.

A naive attack might try to reveal the keys that were used to generate the permutation b. However, anyone who knows the permutation p can decrypt the images. In our cryptanalysis, we develop methods to reveal the permutation b. Such an approach is more general as it easily covers cases where different chaotic maps are used to generate the permutation.

Inter-round dependencies in decryption

The function g in (29) forms a chain that relates consecutive ciphertext pixels. Hence, in encryption for a single round, a change in a plaintext pixel affects many ciphertext pixels. Indeed, if we change $p_{b(i)}$, by (29), c_i changes. Since we have

$$c_{i+1} = p_{b(i+1)} + g(c_i) + h_{i+1} \mod 256,$$

a change in c_i, in turn, changes c_{i+1}. Thus, for a single round, a change in $p_{b(i)}$ affects $c_i, c_{i+1}, \ldots, c_n$. As a result, a ciphertext pixel depends on many plaintext pixels.

However, the situation is quite different in decryption and there lies the weakest link in the algorithm. Using (31), we see that, for a single round, p_j is affected by only two ciphertext pixels, $c_{u(j)}$ and $c_{u(j)-1}$. Similarly, for two rounds, p_j is affected by at most four ciphertext pixels.

In order to see this more clearly, let us denote the output of the second round as $d_1 d_2 \cdots d_n$. Using (31) with c_k as the plaintext pixel that is input to second round, we obtain

$$c_k = d_{u(k)} - g(d_{u(k)-1}) - h_{u(k)} \mod 256, \quad 1 \le k \le n. \tag{32}$$

Substituting $k = u(j)$ in (32), we find

$$c_{u(j)} = d_{u^2(j)} - g(d_{u^2(j)-1}) - h_{u^2(j)}. \tag{33}$$

Here, we denote by u^s, the s times composition of u with itself.

Similarly, for $k = u(j) - 1$, we have

$$c_{u(j)-1} = d_{u(u(j)-1)} - g(d_{u(u(j)-1)-1}) - h_{u(u(j)-1)}. \tag{34}$$

Thus, we see from (31), (33) and (34) that, for two rounds of decryption, p_j is affected only by the ciphertext pixels

$$d_{u^2(j)}, d_{u^2(j)-1}, d_{u(u(j)-1)}, d_{u(u(j)-1)-1}.$$

Obviously, depending on the particular permutation u, some of these four pixels might coincide.

Note that the plaintext pixel $p_{b(1)}$ is affected by only c_1 because c_0 is a fixed system parameter. Hence, for two rounds, $p_{b(1)}$ is affected by the ciphertext pixels

$$d_{u(1)}, d_{u(1)-1}.$$

Example 3. We illustrate the dependencies in the decryption for two rounds. Here, $n = 6$ and the permutation u is given as

$$u = \begin{pmatrix} 1\ 2\ 3\ 4\ 5\ 6 \\ 2\ 4\ 1\ 5\ 6\ 3 \end{pmatrix}. \tag{35}$$
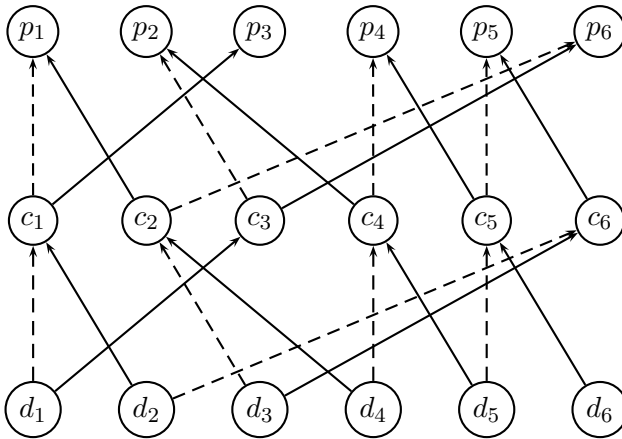

Fig. 5 The dependency paths for the permutation given in (35). A solid arrow indicates that the dependency is through u, while a dashed arrow indicates that the dependency is through $u - 1$.

The dependency paths are given in Fig. 5. In the figure, the directed arrows indicate which pixels affect the computation of the destination pixel. For example, two arrows going from c_5 and c_4 to p_4 means that p_4 is affected by c_5 and c_4.

The dependency chain from from the ciphertext d to the plaintext p is given as follows

$$\begin{aligned}
p_1 &\leftarrow c_1, c_2 \leftarrow d_1, d_2, d_3, d_4, \\
p_2 &\leftarrow c_3, c_4 \leftarrow d_1, d_4, d_5, \\
p_3 &\leftarrow c_1 \leftarrow d_1, d_2, \\
p_4 &\leftarrow c_4, c_5 \leftarrow d_4, d_5, d_6, \\
p_5 &\leftarrow c_5, c_6 \leftarrow d_5, d_6, d_2, d_3, \\
p_6 &\leftarrow c_2, c_3 \leftarrow d_3, d_4, d_1.
\end{aligned}$$

Note that p_3 is affected by only c_1 because $u(3) = 1$. c_1 is, in turn, affected by two ciphertext pixels d_1 and d_2. Also note that p_4 is affected by three ciphertext pixels rather than four because $u(u(4) - 1) = u^2(4) - 1 = 5$. This also means that there are two distinct dependency paths going from d_5 to p_4.

Detecting dependency using chosen ciphertext images

In general, for the decryption in an R round algorithm, a particular plaintext pixel p_j is affected by at most 2^R ciphertext pixels. For a 256×256 image encrypted in 10 rounds, we have $n = 65536$ and $2^R = 1024$. Hence, only about $\frac{1024}{65536} \approx 2\%$ of ciphertext pixels affect any given fixed plaintext pixel.

Let us denote by z, the ciphertext image after R rounds of encryption. The attacker wants to know if there is a dependency path from the ciphertext pixel z_i to the plaintext pixel p_j. Assume that the attacker knows a plaintext-ciphertext image pair (p, z). He changes the value of z_i and requests the plaintext for the changed ciphertext. If p_j changed in the new plaintext, then there is a dependency path from z_i to p_j so that z_i affects p_j.

Note that, for some changes to z_i, p_j might remain the same even when there are dependency paths from z_i to p_j. This is due the nonlinearity of encryption/decryption that operates in a finite domain. In order to detect all the dependency paths, the attacker needs to try more than one changes to z_i. It is highly unlikely that p_j remains fixed for all of these changes.

Detecting changes for all i, $1 \leq i \leq n$, the attacker constructs a binary matrix T showing the dependency relations between ciphertext and plaintext pixels in decryption. If $T_{ij} = 1$, then it means that z_i affects p_j. Since p_j is affected by at most 2^R pixels of z, each column of T contains at most 2^R 1's. All the other entries are zero.

Example 4. The matrix T for the permutation u used in Example 1 is given as

$$T = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}.$$

Finding $b(1)$

Writing (31) for $p_{b(1)}$, we have

$$p_{b(1)} = c_1 - g(c_0) - h_1 \bmod 256.$$

Hence, for one round, $p_{b(1)}$ is affected by only c_1, the first pixel of the output of the first round. The rest of the rounds generate at most 2^{R-1} distinct

dependency paths. Therefore the column $b(1)$ of T contains at most 2^{R-1} 1's. Thus, the column of the matrix T with the least number of 1's gives the attacker a starting point for the attack. Once an attacker constructs the matrix T, he can reveal $b(1)$ by choosing the column k with the least column sum. Then he knows that $b(1) = k$ or $u(k) = 1$.

For example, by inspecting the matrix T in Example 4, the attacker can see that the third column has the least sum. Thus, he concludes that $u(3) = 1$.

Tree of dependency

In order to generalize the attack to the rest of u, we define an operation to denote the dependency relations between the sets.

Given a permutation u on the set $\{1, 2, \ldots n\}$, define the operation L on a set A as follows.

$$L(A) = \{y \mid x \in A \text{ such that } y = u(x) \text{ or } y = u(x) - 1\}.$$

The set $L(A)$ has natural meaning in terms of decryption. Using (31), we see that the set $L(A)$ is the set of ciphertext pixels that affect the set A of plaintext pixels in one round of decryption. In particular, for an integer $k \in \{1, 2, \ldots, n\}$, $L(\{k\})$ is given as

$$L(\{k\}) = \begin{cases} \{u(k)\} & \text{if } u(k) = 1, \\ \{u(k), u(k) - 1\} & \text{otherwise} \end{cases} \tag{36}$$

When L operates on a set with a single element k, we drop the set notation in $L(\{k\})$ and use instead $L(k)$.

We can naturally compose L with itself to define its higher powers. Thus, for $L^2(k)$, we have

$$\begin{aligned} L^2(k) &= L(\{u(k), u(k) - 1\}) \\ &= \{u^2(k), u^2(k) - 1, u(u(k) - 1), u(u(k) - 1) - 1\}. \end{aligned}$$

Here, we implicitly assumed that $1 \notin \{u(k), u^2(k), u(u(k) - 1)\}$. If we have $u(k) = 1$ and $u^2(k) \neq 1$, then, by the definition of L, we have

$$\begin{aligned} L^2(k) &= L(u(k)) \\ &= \{u^2(k), u^2(k) - 1\}. \end{aligned}$$

Again, the powers of L has a natural interpretation in terms of multi-round encryption. For an integer k, $L^i(k)$ is the set of the indices of ciphertext pixels that affect the plaintext p_k in i round decryption. This set is also the set of row indices where the kth column of T has nonzero entries.

Example 5. For the permutation given in Example 1, we have

$$L(1) = \{1, 2\},$$
$$L^2(1) = \{1, 2, 3, 4\},$$
$$L^2(\{1, 6\}) = \{1, 2, 3, 4\}.$$

Overlapping sets of leaves

Using the chosen-plaintext attack given in the beginning of this section, the attacker constructs the matrix T. This is the same as attacker knowing the sets $L^R(k)$, $\forall k \in \{1, 2, \ldots, n\}$. The attacker uses this knowledge to reveal the secret permutation u. First, we need the following facts. For the proofs, see [Solak et al., 2010a].

Lemma 5. *Let x, y and z be integers in $\{1, 2, \ldots n\}$ such that they satisfy*

$$u(x) + 1 = u(y),$$
$$u(y) + 1 = u(z).$$

Then, for every positive integer R larger than 1,

$$L^R(y) \setminus L^R(x) \subset L^R(z).$$

Lemma 6. *Let x and y be integers such that $u(x) = 1$ and $u(y) = 2$. Then, $L^R(x) \subset L^R(y)$.*
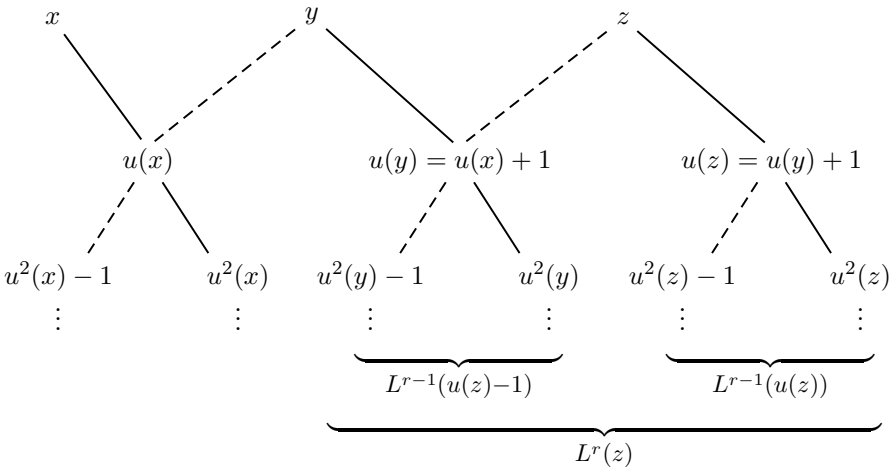

Fig. 6 The sets $L^R(a)$ and $L^{R-1}(a)$. Note that the sets are the leaves of overlapping dependency trees.

The attack

The attack starts with determining the integer x_1 that satisfy $u(x_1) = 1$. For this, the attacker chooses the set $L^R(x_1)$ that has the least number of elements. This also corresponds to choosing the column of the matrix T with the least column sum. It might happen that there are more than one candidate for x_1. For such cases, the attacker repeats the rest of the procedure for each candidate until he encounters a contradiction that he can use to eliminate the candidate.

Once the attacker knows x_1, he goes on to determine x_2 such that $u(x_2) = 2$. Define the set X_2 as

$$X_2 = \{x \mid L^R(x_1) \subset L^R(x)\}.$$

By Lemma 6, $x_2 \in X_2$. In the likely case that X_2 contains a single element, the attacker uniquely pins down x_2. If there are more then one candidate for x_2, the attacker again repeats the rest of the procedure until he can eliminate candidates.

Now, the attacker knows x_1 and x_2 such that $u(x_1) = 1$ and $u(x_2) = 2$. He then searches for x_3 such that $u(x_3) = 3$. In order to pin down x_3, the attacker finds the set defined by

$$X_3 = \{x \mid L^R(x_2) \setminus L^R(x_1) \subset L^R(x)\}.$$

By Lemma 5, $x_3 \in X_3$. If X_3 contains a single element, then the attacker has just found x_3 that satisfies $u(x_3) = 3$.

The attacker continues in this fashion and uses his knowledge of x_i and x_{i+1} to reveal x_{i+2} such that $u(x_i) = i$, $u(x_{i+1}) = i+1$ and $u(x_{i+1}) = i+2$. The attack concludes when all the entries of the secret permutation u are revealed.

In cases when X_{i+1} contains $z_1, z_2, \ldots z_v$, the attacker applies the procedure for each z_m, $1 \le m \le v$, each time assuming that $u(z_m) = i+1$.

For false candidates, we expect the iteration to yield an empty set at some point. Namely, if the set $L^R(z_m) \setminus L^R(x_{i+1})$ is not contained in any $L^R(w)$, then $u(z_m) \neq i+1$ and we eliminate the candidate z_m.

The iterations of the attack are expressed as a recursion in Algorithm 1. The recursive function is `FindNext()` which takes no arguments. The constant data of the algorithm are the sets $L^R(k)$, $\forall k \in \{1, 2, \ldots, n\}$. The algorithm manipulates the global variables b and i. The variable i shows the portion of b that is assumed to have been revealed. Namely, the function `FindNext()` assumes that $b(1), b(2), \ldots, b(i)$ have already been revealed. Note that we also assume that the values $b(1)$ and $b(2)$ are initially known.

In Algorithm 1, Line 1, we find the candidates for $b(i+1)$. In doing this, we exclude the set $\{b(1), b(2), \ldots, b(i)\}$ which is assumed to have been revealed so far. For each candidate z, Lines 6-10 recursively apply the algorithm assuming that $u(z) = i + 1$. The function `FindNext()` returns in Line 13 when no candidates are found. It means that the recursion can not go any deeper because a wrong assumption about the permutation value has been made. In this case, Line 11 backtracks once and another candidate is tried.

Algorithm 1. `FindNext()`

Data: $L^R(k)$, $\forall k \in \{1, 2, \ldots, n\}$, $b(1)$, $b(2)$.
Result: b
Global Variable: b and i. Initially $i \leftarrow 2$.

1 `FindNext()`;
2 **begin**
3    $Z \leftarrow \big\{x \mid L^R(b(i)) \setminus L^R(b(i-1)) \subset L^R(x)\big\} \setminus \{b(1), b(2), \ldots, b(i)\}$;
4    $i \leftarrow i + 1$;
5   **if** $Z \neq \emptyset$ **then**
6   **foreach** $z \in Z$ **do**
7          $b(i) \leftarrow z$;
8   **if** $i = n$ **then**
9   exit
10   ;
11   `FindNext()`;
12       $i \leftarrow i - 1$
13   **else**
14   return
15 **end**

Example 6. We illustrate the attack with an artificially small image size. We choose $R = 3$ and assume an image size of 4×4. Therefore, the secret permutation u maps within the set $\{1, 2, \ldots 16\}$. We generated the permutation randomly and it is given as

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

The other fixed functions g and h are chosen randomly. The attacker calculates the matrix T as

$$T = \begin{pmatrix}
0\,0\,0\,0\,0\,0\,1\,1\,0\,1\,0\,0\,1\,1\,1\,0 \\
0\,0\,1\,1\,0\,1\,0\,0\,1\,0\,0\,0\,1\,1\,1\,0 \\
1\,1\,1\,1\,0\,1\,1\,0\,1\,0\,0\,0\,0\,0\,0\,1 \\
1\,1\,0\,0\,0\,0\,1\,0\,1\,0\,0\,0\,0\,0\,0\,1 \\
1\,1\,0\,0\,0\,1\,0\,1\,1\,0\,1\,0\,1\,0\,0\,0 \\
0\,0\,0\,1\,1\,1\,0\,1\,0\,0\,1\,1\,1\,0\,0\,0 \\
0\,0\,0\,1\,1\,1\,0\,0\,0\,0\,1\,1\,0\,0\,0\,1 \\
0\,0\,1\,1\,0\,1\,0\,0\,0\,0\,1\,0\,0\,1\,0\,1 \\
0\,0\,1\,1\,0\,0\,0\,0\,1\,0\,0\,0\,0\,1\,1\,1 \\
1\,0\,1\,0\,0\,0\,0\,0\,1\,1\,1\,0\,0\,1\,1\,0 \\
1\,0\,0\,0\,0\,0\,1\,1\,0\,1\,1\,0\,1\,0\,1\,0 \\
0\,0\,0\,0\,0\,0\,1\,1\,0\,0\,0\,0\,1\,0\,0\,1 \\
1\,0\,0\,0\,0\,0\,1\,0\,0\,1\,1\,1\,0\,0\,0\,1 \\
1\,0\,0\,0\,1\,0\,0\,0\,0\,1\,1\,1\,0\,0\,0\,0 \\
0\,1\,0\,0\,1\,0\,0\,0\,1\,1\,0\,1\,1\,0\,0\,0 \\
0\,1\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,1\,0\,0\,0
\end{pmatrix}.$$

For the ith column of the matrix T, the row indices of the 1's give the set $L^3(i)$.

First, the attacker reveals $b(1)$. For this, he finds the minimum sum column which is the column 5. Thus, the attacker reveals that $u(5) = 1$, or equivalently that $b(1) = 5$. From the 5th column, the attacker sees that $L^3(5) = \{6, 7, 14, 15\}$. He then uses Lemma 6 and searches for the column that has 1's in its 6th, 7th, 14th and 15th rows. This column turns out to be the 12th one. Hence, he concludes $b(2) = 12$. Now that the attacker knows the values of $b(1)$ and $b(2)$. Next, he applies Algorithm 1. Using the matrix T, he calculates that $L^3(12) \setminus L^3(5) = \{13\}$. Searching through the columns of T, the attacker finds that columns 1, 7, 10, 11, 16 have 1 in their 13th rows. Thus, $Z = \{1, 7, 10, 11, 16\}$. Now, he tries those as candidates for $b(3)$. First, he assumes $b(3) = 1$. On this assumption, he calculates the set $L^3(1) \setminus L^3(12) = \{3, 4, 5, 10, 11\}$. But, there is no column that has 1's in its rows corresponding to this set. Hence, $b(3) \neq 1$. Next, he tries $b(3) = 7$. He calculates $L^3(7) \setminus L^3(12) = \{1, 3, 4, 11, 12\}$. Again, there is no column that has 1's in its rows corresponding to this set. The third candidate is 10, which happens to be the correct one. Assuming $b(3) = 10$, the attacker quickly reveals the rest of the secret permutation b.

4 Conclusion

The rich and complex behavior of chaotic systems attracted many researchers into designing chaotic ciphers using the inherent noise-like character of chaotic signals. During the last two decades, we have seen many proposals for chaotic ciphers. At the same time, many of these proposals have been shown to be very weak or in some cases even basically flawed.

We see that in many chaotic ciphers, it is possible to bypass the chaotic subsystems and attack the intermediate parameters instead. In such a case, it does not matter how rich and complex the chaotic behavior are. An attacker always tries to exploit the weakest link in the encryption chain.

In yet many other cases, algebraic structure of the chaotic cipher contains weaknesses that can be exploited by an attacker. Interestingly, in only rare cases, the chaos is the weak point. Rather, the break comes through the way that the chaotic signals are used in encryption.

A healthy co-development of analysis and design is crucial for the chaos cryptography to become a mature field. The designers should be well aware of the existing attacks and use strong and well-known structures in their designs. Also, chaos cryptography needs to incorporate rigorous tools and methods developed in mainstream cryptography.

References

Alvarez, G., Li, S.J.: Some basic cryptographic requirements for chaos-based cryptosystems. Int. J. Bifurcation Chaos 16(8), 2129–2151 (2006)

Amig, J.M., Kocarev, L., Szczepanski, J.: Theory and practice of chaotic cryptography. Physics Letters A 366(3), 211–216 (2007), ISSN 0375-9601

Anstett, F., Millerioux, G., Bloch, G.: Chaotic cryptosystems: Cryptanalysis and identifiability. IEEE Tran. Circuits and Systems I-Regular Papers 53(12), 2673–2680 (2006), ISSN 1057-7122

Çokal, C., Solak, E.: Cryptanalysis of a chaos-based image encryption algorithm. Physics Letters A 373(15), 1357–1360 (2009) ISSN 0375-9601

Dachselt, F., Schwarz, W.: Chaos and cryptography. IEEE Tran. Circuits and Systems I - Fundamental Theory and Applications 48(12), 1498–1509 (2001), ISSN 1057-7122

Fraleigh, J.B.: A First Course in Abstract Algebra. Addison Wesley, Reading (2002)

Fridrich, J.: Symmetric ciphers based on two-dimensional chaotic maps. International Journal of Bifurcation and Chaos 8(6), 1259–1284 (1998)

Guan, Z.-H., Huang, F., Guan, W.: Chaos-based image encryption algorithm. Physics Letters A 346(1-3), 153–157 (2005), ISSN 0375-9601, doi:10.1016/j.physleta.2005.08.006

Huang, C.K., Nien, H.H.: Multi chaotic systems based pixel shuffle for image encryption. Optics Communications 282(11), 2123–2127 (2009), ISSN 0030-4018, doi:10.1016/j.optcom.2009.02.044

Kocarev, L., Jakimoski, G.: Pseudorandom bits generated by chaotic maps. IEEE Tran. Circuits and Systems I - Fundamental Theory and Applications 50(1), 123–126 (2003), ISSN 1057-7122

Li, S., Li, C., Chen, G., Bourbakis, N.G., Lo, K.-T.: A general quantitative cryptanalysis of permutation-only multimedia ciphers against plaintext attacks. Signal Processing: Image Communication 23(3), 212–223 (2008), ISSN 0923-5965, doi:10.1016/j.image.2008.01.003

Lovasz, L.: Combinatorial Problems and Exercises. AMS, Providence (2007)

Masuda, N., Jakimoski, G., Aihara, K., Kocarev, L.: Chaotic block ciphers: From theory to practical algorithms. IEEE Tran. Circuits and Systems I-Regular Papers 53(6), 1341–1352 (2006), ISSN 1057-7122

Patidar, V., Pareek, N.K., Sud, K.K.: A new substitution-diffusion based image cipher using chaotic standard and logistic maps. Communications in Nonlinear Science and Numerical Simulation 14(7), 3056–3075 (2009), ISSN 1007-5704, doi:10.1016/j.cnsns.2008.11.005

Pisarchik, A.N., Flores-Carmona, N.J., Carpio-Valadez, M.: Encryption and decryption of images with chaotic map lattices. Chaos 16(3), 033118 (2006)

Rhouma, R., Solak, E., Belghith, S.: Cryptanalysis of a new substitution-diffusion based image cipher. Communications in Nonlinear Science and Numerical Simulation 15(7), 1887 (2010)

Solak, E., Cokal, C., Yildiz, O.T., Biyikoglu, T.: Cryptanalysis of fridrich's chaotic image encryption. Int. J. Bifurcation Chaos 20(5), 1405–1413 (2010a)

Solak, E., Çokal, C.: Cryptanalysis of a cryptosystem based on discretized two-dimensional chaotic maps. Physics Letters A 372(46), 6922–6924 (2008)

Solak, E., Çokal, C.: Algebraic break of a cryptosystem based on discretized two-dimensional chaotic maps. Physics Letters A 373(15), 1352–1356 (2009)

Solak, E., Çokal, C.: Algebraic break of image ciphers based on discretized chaotic map lattices. Information Sciences 181(1), 227–233 (2011)

Solak, E., Rhouma, R., Belghith, S.: Cryptanalysis of a multi-chaotic systems based image cryptosystem. Optics Communications 283(2), 232–236 (2010b)

Xiang, T., Wong, K.-W., Liao, X.: A novel symmetrical cryptosystem based on discretized two-dimensional chaotic map. Physics Letters A 364(3-4), 252–258 (2007)

Chapter 8
Lessons Learnt from the Cryptanalysis of Chaos-Based Ciphers

Gonzalo Alvarez[1], José María Amigó[2], David Arroyo[3], and Shujun Li[4]

[1] Instituto de Física Aplicada, Consejo Superior de Investigaciones Científicas
Serrano 144, 28006 Madrid, Spain
[2] Centro de Investigación Operativa, Universidad Miguel Hernández
Avda. de la Universidad s/n. 03202 Elche (Alicante) Spain
[3] Instituto de Acústica, Consejo Superior de Investigaciones Científicas
Serrano 144, 28006 Madrid, Spain
[4] Fachbereich Informatik und Informationswissenschaft, Universität Konstanz
Universitätsstrasse 10, 78457 Konstanz, Germany

1 Introduction

The idea of using chaotic transformations in cryptography is explicit in the foundational papers of Shannon on secrecy systems (e.g., [96]). Although the word "chaos" was not minted till the 1970s [71], Shannon clearly refers to this very concept when he proposes the construction of secure ciphers by means of measure-preserving, mixing maps which depend 'sensitively' on their parameters. The implementation of Shannon's intuitions had to wait till the development of Chaos Theory in the 1980s. Indeed, it was around 1990 when the first chaos-based ciphers were proposed (e.g., [78], [46]). Moreover, in 1990 chaos synchronization [91] entered the scene and shortly thereafter, the first applications to secure communications followed [56, 37]. The idea is remarkably simple: mask the message with a chaotic signal and use synchronization at the receiver to filter out the chaotic signal. The realization though had to overcome the desynchronization induced by the message itself. After this initial stage, the number of proposals which exploited the properties of chaotic maps for cryptographical purposes, grew in a spectacular way.

As any topic developed in a rush tempo, chaos-based (or 'chaotic') cryptography has suffered, and to a minor extent it is still suffering from annoying shortcomings. Some of them even breach basic principles of cryptography but, alas, persist in the publications after the many years and the many warnings [6, 15, 17]. In this chapter we are going through the most common errors and bad practices that have been marring chaos-based cryptography. Formulated in a positive way, we shall highlight a number of principles and practices that should be prior to any serious proposal in this field. These general principles may be classified in four groups.

- Design aspects (e.g., specification of the key space, existence of weak keys, etc.)
- Dynamic-theoretical aspects (e.g., use of maps with robust chaos, etc.)
- Computational aspects (e.g., computational efficiency, study of numerical degradation, etc.)
- Security-related aspects (i.e., whether a cipher is resistant to known attacks, etc.)

Needless to say, resistance of a new cipher against known attacks is no guarantee of security, but it is obviously a necessary condition; the same applies to the various statistical tests for pseudo-randomness. The security of each cipher has to be discussed on a case-by-case basis and it depends very much on its specifics. Contrarily to the commonplace in chaotic cryptography that the more "messy" the encryption process, the more secure the resulting cipher, the security of a cipher should rely on general principles, well-tested architectures and efficient implementations. This is the aim of this work, i.e., to emphasize the main problems of recent chaos-based cryptosystems in order to establish a methodology to avoid them. Although it is not possible to conclude the unconditional security of an encryption system, we can asses its practical security by means of common tools and practices in the cryptanalysis of chaotic cryptography. Furthermore, this analysis can be used to define the requirements of chaotic cryptography from a theoretical point of view. In particular, we shall advocate below for a chaos-based cryptography on integer numbers or, more generally, finite fields as a general framework where chaos-based cryptography can merge with conventional cryptography while preserving its identity.

This paper is organized as follows. The first section deals with the main problems of chaos-based cryptography, which are the conclusion of our work in the field of the cryptanalysis of chaotic cryptosystems. As a result of the study of those problems, in the second section we introduce a set of rules to avoid them and to design *secure* chaotic cryptosystems (by means of the critical contexts drawn through the discussion in the first section). The coherence between chaos-based cryptography and conventional cryptography requires to go into a common theoretical background, which is the core of the last two sections. Once the fundamental limitations of naive chaotic cryptography have been recognized, one can take a practical approach and try to make the most of it. After all, the properties of the numerical (actually, periodical) orbits of chaotic maps can be good enough in practice; as a matter of fact, this has been the general approach till now. This being the case, the last section concludes our paper by proposing the theoretical aspects of chaos-based cryptography. In particular, we go into the question of how chaos-based cryptography can be understood (and even defined) in the realm of finite-precision (hence, discrete) mathematics. We want to stress right away that a chaos-based (or rather 'chaos-inspired') cryptography on the integers, thus without numerical degradation, is possible and, of course, preferable to the

use of real-number approximations. In fact, some examples of chaos-based ciphers on integer numbers are mentioned in Sec. 2 and 3 —this is the approach we recommend.

2   Main Problems in Chaos-Based Cryptography

The number of chaos-based ciphers proposed in the literature is too large for us to attempt here a review of them. The interested reader is referred to [17] for a brief but sufficient overview. Rather than discussing this or that implementation of chaos-based cryptography, we will only delve into the basic ideas that unify all of them.

Roughly speaking, there are two classes of chaotic cryptosystems. The first one is based on chaotic systems implemented in digital (i.e., discrete-time and discrete-space) domain. This type of chaos-based ciphers are usually known as digital chaos-based cryptosystems or digital chaotic ciphers. One typical type of digital chaotic ciphers amounts to numerically computing a great number of iterations of a discrete chaotic system, using the message and/or the key as initial data (see [46, 43] and references therein). This is basically also the strategy in [18], [102], where periodic approximations of chaotic automorphisms are used to define substitutions (so-called S-boxes) resistant to linear and differential cryptanalysis. The ciphers which exploit the ergodicity of chaotic maps, like the one originally proposed in [31] and its posterior improvements, may be thought in this class as well. There are also some ciphers built on top of chaos-based pseudo-random number generators (PRNGs) like those proposed in [70, 19].

The second class amounts to scrambling a message via a chaotic system evolving in continuous-space (but maybe discrete-time) domain. Analog cryptosystems based on chaos synchronization belong to this second class. Various cryptosystems of this class, corresponding to distinct ways of hiding a message, have drawn the attention of researchers over the years. The most important schemes following such a principle are additive masking, chaotic switching, parameter modulation, and message-embedding (a.k.a. direct modulation). Additive masking was first suggested in [56], [37] and [109]. Chaotic switching is also referred to as chaos shift keying (CSK). A description with deep insights can be found in [61], even though the idea of CSK was proposed a couple of years before [38]. Essentially, chaotic switching is a special type of chaotic parameter modulation: binary modulation. As a generalization of chaotic switching, the parameter involved in a parameter modulation system can be both discrete [89, 38] or continuous [42, 51], rather than only 0 or 1 in chaotic switching. The message-embedding technique is given different names in the literature: embedding [72, 82], non autonomous modulation [112] or direct chaotic modulation [47]. A slight different method based on message-embedding is the hybrid message-embedding. It was first proposed in [113] but the terminology " hybrid" was actually introduced in [88]. Most analog

chaos-based cryptosystems are based on a single communication channel between the sender and the receiver, which is used to transmit the driving signal and the encrypted message to achieve synchronization between the slave and master systems. Some researchers also proposed to use two communication channels to enhance security, where one channel is used for synchronization and the other is for encryption [83, 54]. In Fig. 1, we show the basic structures of the three analog chaos-based cryptosystems: chaotic masking, chaotic switching and chaotic modulation.

(a) Chaotic masking

(b) Chaotic switching (CSK)

(c) Chaotic modulation
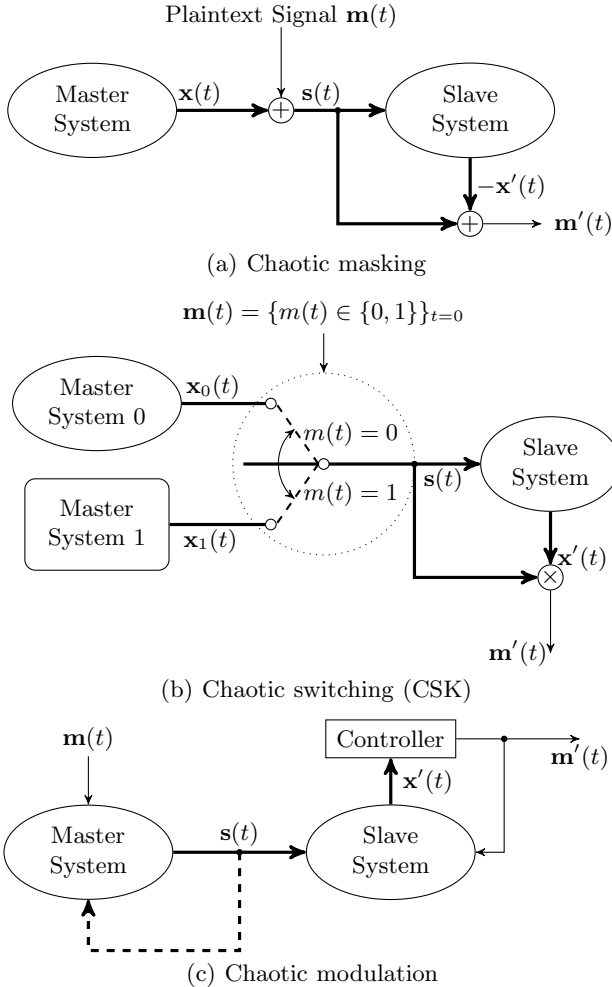
Fig. 1 Basic structures of the three analog cryptosystems based on chaos synchronization.

In any of the families of chaos-based cryptosystems the core of the design process is the selection of a *good* chaotic system for an encryption algorithm [55]. From a general point of view it is not possible to define chaotic cryptosystems satisfying the *chaotic-system-free property* [64]. This being the case, the selection of a certain encryption scheme demands the identification of a group of chaotic systems with a certain set of dynamical properties. Furthermore, the *hardware* implementation of the chaotic encryption algorithm must guarantee its security, but also its efficiency. According to our experience in the field of the cryptanalysis of chaos-based cryptosystems, the most critical problems in chaotic cryptography arise from three elements: the selection of a chaotic system, the choice of an encryption architecture, and the implementation of the encryption. Next those problems are listed and discussed, according to our previous works in [67, 24, 21].

2.1   Problems with the Selection of the Chaotic System

Problem 1. Definition of the key leading to non-chaotic behavior. In some chaos-based cryptosystems the control parameters (or part of it) of the underlying chaotic systems are determined by the secret key. If the link between the secret key and the control parameters is not established carefully, then it is possible that the underlying chaotic system evolves in an non-chaotic way, which further erodes the confusion and diffusion properties required by the resulting cryptosystem.

The chaotic systems used as base of cryptosystems are defined in a parametric way such that their dynamics depend on one or several control parameters. Moreover, those chaotic systems are dynamical systems which show a chaotic behavior for certain values of the associated control parameters. Therefore, the design of a cryptosystem based on any of those dynamical systems must be done by guaranteeing the use of the set of values of the control parameters leading to chaos. Otherwise, the underlying dynamical system associated with the cryptosystem will not be chaotic anymore, which implies a reduction of the level of entropy in the ciphertext (i.e., the output of the cryptosystem) and also a decreasing of the influence on the ciphertext of a change in the plaintext (i.e., the input of the cryptosystem). This problem is specially relevant when the design of the cryptosystem is based on a dynamical system with chaotic behavior only for a set of disjoint parts of the whole space of the control parameters. This is the case of the logistic map and the Hénon map, which have been used in [93, 73, 106] and in [34] respectively without a thorough analysis of their dynamics, as we have pinpointed in [29, 27, 25, 26]. As a conclusion, it is highly advisable to use dynamical systems with chaotic behavior for all the values of the control parameter(s). That is, *robust chaotic systems* [30] should be used instead of non-robust ones.

Problem 2. Nonuniform probability distribution function. In some chaos-based encryption architectures the confusion and/or diffusion proper-

ties depend on the probability distribution function of the orbits derived from the underlying chaotic systems. If that distribution is not uniform and dependent on the values of control parameters, then the quality of the diffusion process is reduced.

The iteration of a chaotic map can be used to generate pseudo-random sequences to encrypt the plaintext. The encryption procedure could be performed in different ways, but all of them demand the equiprobability of all the states contained in the pseudo-random sequences. If this requirement is not satisfied, then the conditional entropy of the ciphertext with respect to the plaintext may not be large enough so that some information will be leaked about relationships between the output and the input of the target cryptosystem (see the entropy attack in [11], and the cryptanalysis in [10]). This effect is specially significant for image encryption, as pointed out recently in [63]. As a remedy, chaotic maps with a uniform probability distribution function should be selected as base of this kind of cryptosystems, being the family of piecewise linear chaotic maps [68] a good option.

Problem 3. Return map reconstruction. The ciphertext of some cryptosystems makes it possible to reconstruct a return map of the underlying chaotic system. If such a return map is meaningful, then an attacker may be able to infer the values of the control parameters that govern the evolution of the chaotic system.

The most direct way to estimate the control parameters from a chaotic orbit is to plot x_{n+1} versus x_n, which is actually the chaotic map itself. If this representation shows a simple function between x_{n+1} and x_n, then it could be possible to infer the control parameter. In [97] a chosen-ciphertext attack is used to build a discretized version of the logistic map which further leads to the estimation of the control parameter. One solution against this kind of attack is to shuffle/truncate the chaotic orbit before using it for encryption, which randomizes the plot of the return map.

The reconstruction of the return map is specially meaningful in the context of analog chaos-based cryptography. Encryption techniques based on chaotic masking or chaotic switching can be circumvented by constructing some return maps of the master system of an analog chaotic cryptosystem [92], as it has been shown in [65, 66].

Problem 4. Low sensitivity to secret key. The most common problem (and one of the most serious ones) about analog chaos-based cryptosystems is the low sensitivity to the secret key. The low sensitivity is a necessary requirement for real implementations of any analog chaos-based cryptosystem because it is impossible to ensure exact matching of the master and slave systems. Unavoidable noise and manufactural component deviation involved in chaotic circuits are the two main factors causing this security problem [107, 116].

Problem 5. Erosion of computational efficiency due to the structural complexity of the underlying chaotic systems. The structural complexity of a chaotic system is a critical element when evaluating its suitability for cryptographic applications. With this bottom line in [21, Sec. 1.3.3] we emphasized that structural complexity can be minimized by selecting chaotic systems defined in discrete time. Indeed, in discrete time chaos can be achieved for phase space of Dimension 1, whereas it has to be at least of Dimension 3 when considering continuous time.

2.2   Problems with the Encryption Architecture

Problem 6. Part of the key should not leak the rest of the key. In some cryptosystems the secret key is composed of different subkeys. If the knowledge of some subkeys allows the recovery of the rest of the key, then a *partial key recovery attack* can be performed. Therefore, the design of a cryptosystem must guarantee that the different subkeys composing the secret key are uncorrelated.

In the context of a secure and robust encryption system it is assumed that partial knowledge of the key does not reveal information about the rest of the key and, as a result, the cryptosystem performance is not harmed [6, Rule 7]. This rule is not satisfied in the scenarios drawn by [31, 34, 20], partial knowledge of the key can be used to obtain the rest of the key [11, 26, 95].

Case study 2.1 ([26]). *Cryptanalysis of the cryptosystem proposed in [34]*

In [34] the Hénon map is used as the *heart* of a chaos-based cryptosystem, which entails a security problem that we have highlighted in [26]. The analytical definition of the Hénon map is:

$$x_{k+1} = \begin{bmatrix} u_{k+1} \\ v_{k+1} \end{bmatrix} = \begin{bmatrix} 1 - \delta \cdot u_k^2 + v_k \\ \beta \cdot v_k \end{bmatrix}, \tag{1}$$

with $\delta, \beta \in \mathbb{R}$. In the cryptosystem defined in [34] the plaintext is divided into blocks $\{p_k\}_{k=0}^{N-1}$, where each block has M bits. The encryption of the plain-blocks is carried out for $k = 0, \ldots, N-1$ in turn. For the k-th plain-block p_k, the corresponding cipher-block is x_{k+1}, which is calculated through Eq. (1) by setting

$$\delta = \psi(p_k) \cdot \mu_1(v_k), \tag{2}$$

$$\beta = \mu_2(v_k), \tag{3}$$

where $\psi(x)$ is a bijective function assuring that δ is a valid parameter of Eq. (1) and $\mu_i(x)$, $i \in \{1, 2\}$. Based on the general form of the proposed cryptosystem, the authors of [34] present a concrete configuration: $M = 48$, $\psi(x)$, $\mu_1(x)$ and $\mu_2(x)$ are set in Eqs. (4), (5), (6) respectively.

$$\psi(x) = 1.77 \cdot 10^{-2} + 1.39 \cdot 10^{-15} \cdot x, \tag{4}$$

$$\mu_1(x) = \begin{cases} 1.27 + \frac{x}{10.2}, & \text{if } |x| \le 0.1 + \frac{x}{1.3} \\ 1.28 + \frac{x}{10.2}, & \text{if } 0.1 + \frac{x}{1.3} < |x| \le 0.2 + \frac{x}{1.3} \\ 1.29 + \frac{x}{10.2}, & \text{if } 0.2 + \frac{x}{1.3} < |x| \le 0.3 + \frac{x}{1.3} \\ 1.30 + \frac{x}{10.2}, & \text{otherwise} \end{cases} \tag{5}$$

$$\mu_2(x) = \begin{cases} 0.29 + \frac{x}{10}, & \text{if } |x| \le 0.1 + \frac{x}{1.1} \\ 0.30 + \frac{x}{10}, & \text{if } 0.1 + \frac{x}{1.1} < |x| \le 0.2 + \frac{x}{1.1} \\ 0.31 + \frac{x}{10}, & \text{if } 0.2 + \frac{x}{1.1} < |x| \le 0.3 + \frac{x}{1.1} \\ 0.32 + \frac{x}{10}, & \text{otherwise} \end{cases} \tag{6}$$

Next we show how a known-plaintext attack can be employed to reconstruct Eq. (5), Eq. (6), and v_0 (initial condition of the underlying Hénon map defined in Eq. (1)) when Eq. (4) is known. Given two plaintexts $\{p_{1,k}\}_{k=0}^{N-1}$, $\{p_{2,k}\}_{k=0}^{N-1}$, then

$$u_{1,1} = 1 - \psi(p_{1,0}) \cdot \mu_1(v_0) \cdot u_0^2 + v_0, \tag{7}$$

$$u_{2,1} = 1 - \psi(p_{2,0}) \cdot \mu_1(v_0) \cdot u_0^2 + v_0, \tag{8}$$

and

$$u_{1,k+1} = 1 - \psi(p_{1,k}) \cdot \mu_1(v_k) \cdot u_{1,k}^2 + v_k, \tag{9}$$

$$u_{2,k+1} = 1 - \psi(p_{2,k}) \cdot \mu_1(v_k) \cdot u_{2,k}^2 + v_k, \tag{10}$$

$$v_k = \mu_2(v_{k-1}) \cdot u_{k-1}, \tag{11}$$

where $k \ge 1$. Subtracting Eq. (9) from Eq. (10):

$$\widetilde{\mu}_1(v_k) = \frac{u_{2,k+1} - u_{1,k+1}}{\psi(p_{1,k}) \cdot u_{1,k}^2 - \psi(p_{2,k}) \cdot u_{2,k}^2}. \tag{12}$$

From Eq. (11):

$$\widetilde{\mu}_2(v_{k-1}) = \frac{u_{1,k+1} - 1 + \psi(p_{1,k}) \cdot \widetilde{\mu}_1(v_k) \cdot u_{1,k}^2}{u_{k-1}}. \tag{13}$$

If the quantization error is ignored, we have $\widetilde{\mu}_1(v_k) = \mu_1(v_k)$ and $\widetilde{\mu}_2(v_{k-1}) = \mu_2(v_{k-1})$. As a consequence, it is possible to reconstruct $\mu_1(v_k)$ and $\mu_2(v_k)$ repeating this procedure for $k = 1, \ldots, N$. In order to prove the proposed known-plaintext attack, 10000 points for $\mu_1(v_k)$ and $\mu_2(v_k)$ were calculated for $u_0 = 0.4$, $v_0 = 0.9402036$ and $u_0 = 0.4$, $v_0 = -0.5123493$. In Fig. 2 it is shown how it was possible to get an estimation of $\mu_1(v_k)$, $\mu_2(v_k)$ shape. This is due to the fact that the first component of the Hénon map employed in the encryption process is sent through the communication channel without applying any masking transformation.
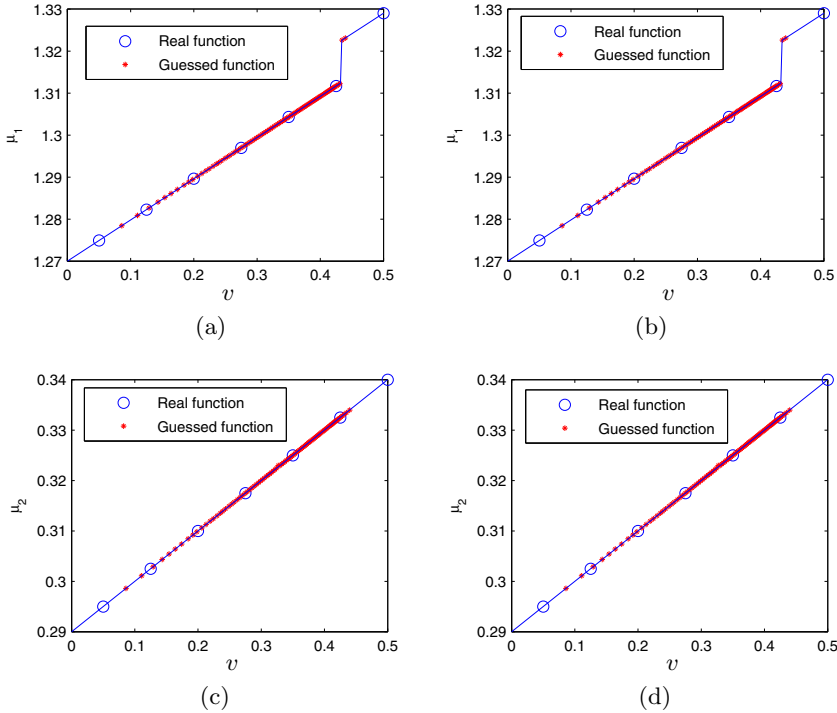
Fig. 2 Recovered and original functions for the PRSK mechanism when they are designed as in [34] (a) $\mu_1(v)$ for $v_0 = 0.9402036$; (b) $\mu_1(v)$ and $v_0 = -0.5123493$; (c) $\mu_2(v)$ for $v_0 = 0.9402036$; and (d) $\mu_2(v)$ and $v_0 = -0.5123493$.

Problem 7. Key estimation from the ciphertext. A bad definition of the ciphertext derived from a chaos-based cryptosystem could allow the estimation of the initial condition(s) and/or the control parameter(s) of the underlying chaotic system. This problem is present in some chaos-based cryptosystems whose ciphertext is given by fragments of orbits, sampled versions of the orbits, or discretized versions of the orbits of the underlying chaotic systems. Moreover, parameter estimation is a critical problem of cryptosystems based on chaotic parameter modulation, since adaptive synchronization techniques can be used to get an approximation of the control parameters of the underlying chaotic systems by minimizing the synchronization errors.

An m-dimensional discrete-time chaotic map is defined by the rule of evolution

$$x_{n+1} = f_\lambda(x_n),$$

and, as a result, the ciphertext cannot be the orbits of the map since it may allow the estimation of λ from $m + 1$ or a bit more consecutive units of

ciphertext. This is the case of the cryptosystem proposed in [73], and which we have cryptanalyzed in [27].

Furthermore, if the invariant set of the chaotic map has a size dependent on the control parameters, even sampled versions of the orbits may allow the estimation of the control parameters through a ciphertext-only attack [10]. We have analyzed this situation in [29] for the cryptosystem proposed in [93]. In addition, the theory of symbolic dynamics may also reveal the weakness of a cryptosystem if the ciphertext allows getting the symbolic sequences of the underlying chaotic system. In [25] we have shown through a chosen-ciphertext attack how to derive the symbolic sequence of the logistic map driving the encryption procedure defined in [106]. Once we have the symbolic sequence, we can infer the values of the control parameter and initial condition of the underlying logistic map according to the theory of applied symbolic dynamics described in [3]. Nevertheless, a constant size for the invariant set of a chaotic map is a necessary but not sufficient condition to avoid the estimation of the control parameter from the corresponding orbits. In this respect, and according to [22], chaotic orbits should be analyzed also by means of their associated order patterns. Finally, another critical context is depicted when measures of statistical distance can be applied to distinguish between keystreams, as we have shown in [23].

For most analog chaos-based secure communication systems, the ciphertext is not very sensitive to the secret key, which is caused by a simple relationship between synchronization error and key mismatch: the larger the key mismatch is, the larger the synchronization error will be, and vice versa. This means that an iterative algorithm can be used to determine the value of the secret parameters, which corresponds to the concept of adaptive synchronization. A lot of work has been reported about adaptive synchronization when the master systems' parameters are unknown to the receiver/attacker. Some of the work can directly be used or easily extended to break analog chaos-based secure communication systems [39, 103, 104]. In addition to methods based on adaptive synchronization, there are also other ways one can use to estimate the secret parameters (i.e., the key) of chaos-based cryptosystems. For instance, due to the nature of Lorenz and Chua Chaotic Systems, the secret parameters can be determined from the driving signal and its derivatives (mainly differentials of different orders) [32, 105, 74]. For some specific schemes, it is also possible to derive part of the secret parameters by analyzing the return maps of the master systems [65]. When chosen-ciphertext attacks are possible, i.e., when the attacker can access the decryption machine for some time, the attacker may set the driving signal to a fixed constant C in order to get the values of all secret parameters [50].

Case study 2.2 ([27]). *Cryptanalysis of the cryptosystem described in [73]*

In [73] the encryption procedure is carried out by decomposing the input plaintext signal into two different subbands and masking each of them with a

pseudo-random number sequence generated by iterating the chaotic logistic map. The decomposition of the input plaintext signal x_n is driven by

$$t_n = K \sum_{\forall j} x_j h_{2n-j}, \tag{14}$$

$$t'_n = K' \sum_{\forall j} x_j h'_{2n-j}. \tag{15}$$

Then, the masking stage generates the ciphertext signal (v_n, v'_n) according to the following equations:

$$v_n = t_n + \alpha(t'_n), \tag{16}$$

$$v'_n = t'_n - \alpha'(v_n), \tag{17}$$

where $\alpha(u) = u + s_n$ $(\alpha'(u) = u + s'_n)$ and s_n (s'_n) is the state variable of the logistic map.

The secret key of the cryptosystem is composed of the initial conditions and the control parameters of the two logistic maps involved, i.e., s_0, s'_1, λ and λ'.

In a known-plaintext attack the cryptanalyst possesses a plaintext signal $\{x_n\}$ and its corresponding encrypted subband signals $\{v_n\}$ and $\{v'_n\}$. Because $\{h_n\}$, $\{h'_n\}$, K and K' are public, we can get $\{t_n\}$ and $\{t'_n\}$ from $\{x_n\}$. Then we can get the values of $\{s_n\}$ and $\{s'_n\}$ as follows:

$$s_n = v_n - t_n - t'_n, \tag{18}$$

$$s'_n = t'_n - v_n - v'_n. \tag{19}$$

For $n = 0$, the values of the subkeys s_0 and s'_0 have been obtained. Furthermore, we can obtain the control parameters by just doing the following operations:

$$\lambda = \frac{s_{n+1}}{s_n(1 - s_n)},$$

$$\lambda' = \frac{s'_{n+1}}{s'_n(1 - s'_n)}.$$

Problem 8. Direct extraction of plaintext. In the context of analog chaos-based cryptography, in some cases it is feasible to infer the plaintext message signals from the driving signals without estimating the secret key or the carrier signals. Techniques such as power-spectral filtering (or power energy analysis) and return map analysis have been used for this purpose.

Regarding power-spectral filtering, even when the power spectra of some chaotic systems seem to be good, significant spectrum peaks can be found in the spectra by removing the symmetries of the chaotic attractors [62, 86, 85]. For instance, the spectrum of $x(t)$ in the Lorenz System is relatively good,

but that of $|x(t)|$ has a significant peak. When the plaintext message signal is hidden in the driving signal, the narrow-band spectrum means that the driving signal may be directly filtered to recover the message signal [114, 8]. On the other hand, for some parameter modulation systems the power energy of the driving signal varies according to the value of the transmitted signal. This makes it possible to obtain a smoother version of the message signal by observing the average power energy of the driving signal in a sliding time-window [12]. Exact recovery of the plain message signal is possible for chaotic switching systems, because each bit has to be held for a considerably long time to ensure that chaos synchronization is established.

In addition to power filtering, cryptanalysts have figured out some other techniques that can be used to extract the plaintext directly from the ciphertext, which include generalized synchronization, short-time period analysis and switching event detection. For chaotic switching systems and some parameter modulation systems, there is a simple relationship between the synchronization error and the value of the transmitted signal. This link can be interpreted as a way to extract the plaintext message signal directly [8, 115]. With respect to the study of short-time period, if the spectrum of the driving signal (or some modification of it) involved has a significant peak, generally there exists a simple relationship between the peak frequency and the values of the control parameters. In this case, one can try to extract the short-time period as a measurement of the peak frequency modulated by the plaintext message signal. According to the change of the extracted short-time periods, the plaintext message signal can be extracted exactly (for chaotic switching systems) or approximately (for some parameter modulation systems) [111, 4]. Finally, for chaotic switching and parameter modulation systems the dynamics of the master systems will change significantly when the value of the modulating signal (i.e., the plaintext message signal) changes. By detecting and tracking these switching events, it may be possible to recover the modulating signal [101].

Case study 2.3 ([4]). *Short-time period analysis based cryptanalysis of a cryptosystem proposed in [40]*

In [40], the author proposes a symmetric secure communication system based on parameter modulation of a chaotic oscillator acting as a transmitter. The receiver is a chaotic system synchronized by means of an adaptive observer. Two sample implementations are given: one with the Lorenz attractor and another with Chua's attractor. In this case study the latter will be broken, to illustrate how our method works with a different double-scroll attractor. It works equally well for Lorenz attractor, though.

Chua's circuit dynamics can be described by the following equations:

$$
\begin{aligned}
\dot{x}_1 &= \alpha(-x_1 + x_2) - f_1(x_1), \\
\dot{x}_2 &= x_1 - x_2 + x_3, \\
\dot{x}_3 &= -\beta x_2.
\end{aligned}
\tag{20}
$$

where $f_1(x) = bx + 0.5(a - b)(|x + 1| \quad x - 1|)$. In the example the system is implemented with the following parameter values, $(\alpha, \beta, a, b) = (10, 18, -4/3, -3/4)$. The encryption process is defined by modulating the parameter β with the binary encoded plaintext, so that it is $\beta + 1.25$ if the plaintext bit is "1" and $\beta - 1.25$ if the plaintext bit is "0". The duration of the plaintext bits must be much larger than the convergence time of the adaption law. The uncertain system can be rewritten in a compact form as:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} -10 & 10 & 0 \\ 1 & -1 & 1 \\ 0 & \beta & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{pmatrix} f_1(x_1) \\ 0 \\ 0 \end{pmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} x_2 \theta, \qquad (21)$$

$$y = C \cdot x = x_3, \qquad (22)$$

$$C = [0 \ 0 \ 1], \qquad (23)$$

$$\theta = \Delta\beta = \pm 1.25. \qquad (24)$$

To launch an attack based on short-time period analysis, we first transform $x_3(t)$ to $|x_3(t)|$, which has a significant peak in the frequency domain and thus a short-time periodicity linked to the plaintext bits. The short-time period of $|x_3(t)|$ can be measured as the distance between adjacent cross-zero points (or from the dominant frequency peak in the short-time FFT domain). The resultant signal is denoted by $p(t)$. Then, $p(t)$ is filtered by removing singular peaks and DC component to get $p^*(t)$. Next, an averaging filter is used to get a smoother signal $fp^*(t)$. Finally, this smoother signal is binarized to recover the plaintext signal. The signals involved in the process are shown in Fig. 3.

Problem 9. Efficiency of the cryptosystem depending on the value of the key. If the encryption and decryption times depend on the key or a subkey, then a timing attack can be performed to estimate the (sub)key.

Some encryption architectures perform the transformation of the plaintext into the ciphertext through several encryption rounds. Additionally, in each encryption round a chaotic map is iterated n times. Since the encryption and decryption times have to be constant and independent of the value of the key, it is not a good practice to select the number of encryption rounds and n as part of the key. Otherwise, a timing attack [60, 33] based on the analysis of the encryption and decryption time can be used for the partial estimation of the secret key, which is a serious security flaw.

Case study 2.4 ([29]). *Timing attack on a cryptosystem proposed in [93]*

Let us exemplified a timing attack by recalling the cryptosystem proposed in [93]. In every encryption round of the cryptosystem under consideration, the logistic map is iterated n times, where n is a subkey. This means that, for a
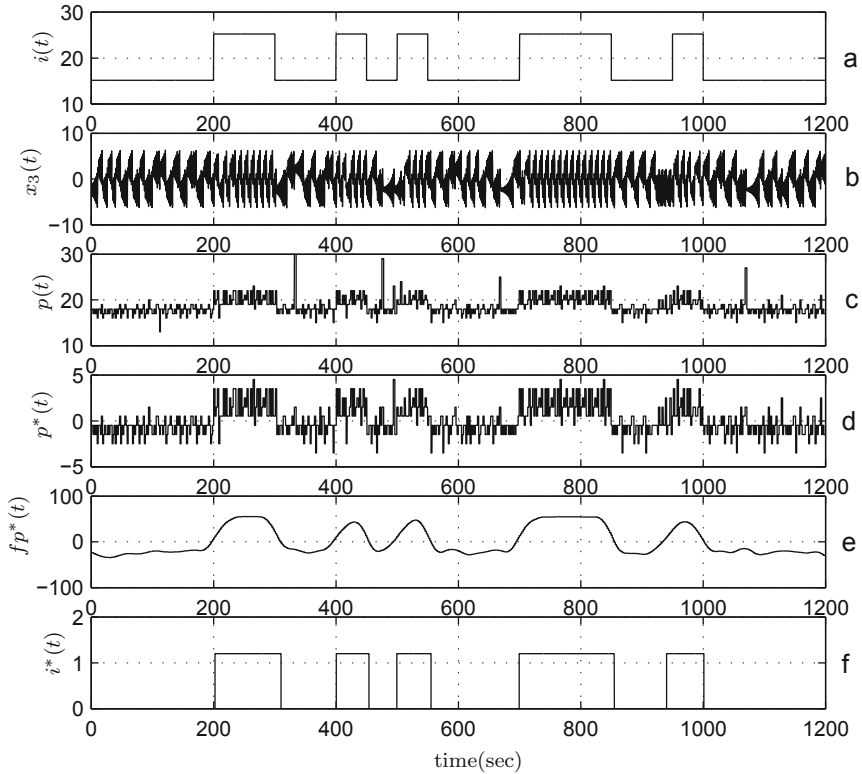
Fig. 3 Breaking a cryptosystem based on Chua's circuit: a) original binary information signal, $i(t)$; b) the transmitted state variable signal or ciphertext, $x_3(t)$; c) the short-time period signal, $p(t)$; d) the clipped signal, $p^*(t)$, after removing singular peaks and DC component; e) the low-pass filtered signal, $fp^*(t)$, revealing the modulation signal; f) recovered message signal, $i^*(t)$, after adequate detection.

certain number of encryption rounds (j) and a certain value of the control parameter λ, the encryption speed decreases as n increases. Similarly, because the encryption/decryption procedure is composed of j repeated cycles, the encryption speed will also become slower if the value of j increases. To be more precise, for a given plain-image, we can expect the existence of the following bi-linear relationship between the encryption/decryption time (EDT) and the values of n and j:

$$EDT(n, j) \approx (c \times n + d_0) \times j + d_1, \qquad (25)$$

where c corresponds to the common operations consumed on each chaotic iteration, d_0 to the operations performed in each cycle excluding those about chaotic iterations, and d_1 to those operations performed on the initialization process and the postprocessing after all the j cycles are completed. In

addition, because λ is just the control parameter of the chaotic map, it is expected that EDT will be independent of its value.

With the aim of verifying this hypothesis, some experiments have been made under the following scenario. An image with random pixel values of size 256×256 was encrypted for different values of λ, n and j. The encryption time corresponding to each key is shown in Fig. 4, from which one can see that Eq. (25) is verified.

The above experimental results ensure the feasibility of a timing attack to a subkey of the cryptosystem under study: by observing the encryption time, it is possible to estimate the value of n if j is known and vice versa. Without loss of generality, assuming an attacker Eve knows the value of n, but not that of j, let us demonstrate how the timing attack can be performed in practice. In this case, the relationship between EDT and the value of j can be simplified as $EDT(n, j) = c_n \times j + d_n$, where $c_n = c \times n$ and $d_n = d_0 \times j + d_1$. Then, if Eve gets a temporary access to the encryption (or decryption) machine, she can carry out a real timing attack in the following steps:

1. She observes the whole process of encryption (or decryption) to get the encryption (or decryption) time t_j and also the size of the ciphertext (i.e., the size of the plaintext).
2. By choosing two keys with different values of j, she encrypts[1] a plaintext (or decrypts a ciphertext) of the same size and gets t_1 and t_2.
3. She derives the values of c_n and d_n by substituting t_1 and t_2 into $EDT(n, j) = c_n \times j + d_n$.
4. She estimates the value of j to be $\hat{j} = \text{round}((t_j - d_n)/c_n)$.
5. She verifies the estimated value $\hat{j}$ by using it to decrypt the observed ciphertext. If the recovered plaintext is something meaningful, the attack stops; otherwise, she turns to search the correct value of j in a small neighborhood of $\hat{j}$ until a meaningful plaintext is obtained.

As a result of the previous analysis, the number of encryption rounds and the number of iterations of the map should be public parameters of the chaos-based cryptosystem instead of part of the key.

Problem 10. Faulty derivation of the parameters of the chaotic system from the key. In some chaos-based cryptosystems the key is used to derive the values of the parameters necessary to iterate a chaotic system and finally encrypt the information. If this mapping implies a reduction of the key space, i.e., that it is only used a subset of the possible values of those parameters, then a brute-force attack on the values of the parameter could be much less demanding than an attack on the secret key.

One important step in the design of a chaos-based cryptosystem is to decide what the key is. One possibility is to use the control parameters and the

---

[1] Please note that this can be done on her own computer, as long as she has the encryption/decryption software installed.
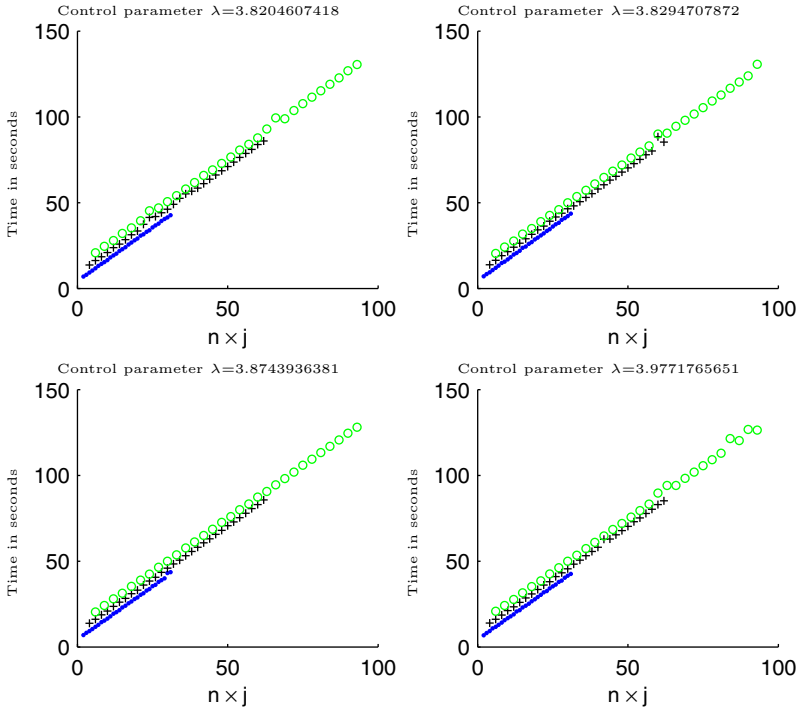
Fig. 4 Encryption time for images of size 256×256 and different values of the number of iterations n and the number of encryption rounds.

initial conditions of the underlying chaotic systems as the secret key or as part of the secret key. Another option is to establish the values of the control parameters and the initial conditions of the maps from the secret key through a certain function. In this sense, it must be assured that the image set of that function is the whole set of possible values of the control parameters and the initial conditions. Otherwise, a brute-force attack can be performed on the reduced space of control parameters and initial condition values with a lower computational cost than the one on the key space. A cryptosystem with this problem was introduced in [87] and was later cryptanalyzed in [9].

Problem 11. Encryption procedure equivalent to a map dependent only on the key. If the transformation of the plaintext into the ciphertext is determined by a procedure equivalent to a map only dependent on the key, then known/chosen-plaintext attacks may be performed to reconstruct the transformation procedure.

In some encryption schemes the transformation of the plaintext into the ciphertext is led either by a procedure derived using only the key, or by a

sampling process on a sequence of values generated using only the key. In those situations, it could be possible to estimate either the key or to make up some function somehow equivalent to the encryption procedure. For example, if the encryption procedure consists of searching plaintexts in pseudo-random sequences generated by iterating a chaotic map, since the pseudo-random sequence remains unchanged unless the key is modified, then it is possible to reconstruct the pseudo-random sequence through a chosen-plaintext attack (see [14, 13]). This problem also exists in those schemes where the encryption procedure consists of a permutation-only stage which is fixed unless the control parameters and initial conditions change, i.e., unless the the secret key is updated. In order to clarify this matter, let us consider again the cryptosystem defined in [44].

Case study 2.5 ([28]). *Cryptanalysis of the cryptosystem defined in [44]*

As mentioned above, the cryptosystem under consideration consists of two stages: a shuffling stage and a masking stage. Assuming that the size of the plain-image $\mathbf{I}$ is $M \times N$ and the cipher-image is $\mathbf{I}'$, the encryption scheme proposed in [44] can be described by the following two procedures.

- *Shuffling procedure*
  In this procedure, the plain-image $\mathbf{I}$ is permuted to form an intermediate image $\mathbf{I}^*$ according to a total shuffling matrix $\mathbf{P}^*$, which is derived by pseudo-randomly permuting the rows and columns of the original position matrix $\mathbf{P} = [(i,j)]$. The pseudo-random row and column permutations are generated by iterating the logistic map with $\lambda = 4$ from a given initial condition x_0.

- *Masking procedure*
  In this procedure, the intermediate image $\mathbf{I}^*$ is further masked by a keystream $\{B(i)\}_{i=1}^{MN}$ as follows: $\forall i = 1 \sim MN$, $I'(i) = I^*(i) \oplus B(i) \oplus I'(i-1)$, where $I(i)$, $I'(i)$ denote the i-th pixels of $\mathbf{I}^*$ and $\mathbf{I}'$ (counted from left to right and from top to bottom), respectively, and $I'(0) = 128$.

The keystream $\{B(i)\}_{i=1}^{MN}$ is generated by iterating Lorenz [75] and Chen [35] Systems and doing some postprocessing on all the variables of state. When a variation of stream cipher is created, as in the case under study, obtaining the keystream is totally equivalent to obtaining the key whenever different plain-images are encrypted using the same key. Upon this hint, in [28] we have carried out a chosen-plaintext attack to recover both the keystream and the shuffling matrix of the cryptosystem described in [44]. Let us choose a plain-image $\mathbf{I}_1$ such that $\forall i,j = 1 \sim MN$, $I_1(i) = I_1(j) = 0$. In this case, the shuffling part does not work, so we have $\mathbf{I}_1^* = \mathbf{I}_1$. Then, we can recover the keystream as follows: $\forall i = 1 \sim MN$, $B(i) = I_1(i) \oplus I_1'(i) \oplus I_1'(i-1)$. After removing the masking part, we can try to recover the shuffling matrix. According to the general cryptanalysis on permutation-only ciphers in [69], only $\lceil \log_{256}(MN) \rceil$ chosen plain-images are needed to recover the shuffling matrix $\mathbf{P}^*$. In total we need $\lceil \log_{256}(MN) \rceil + 1$ chosen plain-images to perform this chosen-plaintext attack.

As a conclusion, the encryption function that transforms a unit of plaintext into a unit of ciphertext should depend on the key and on the whole plaintext.
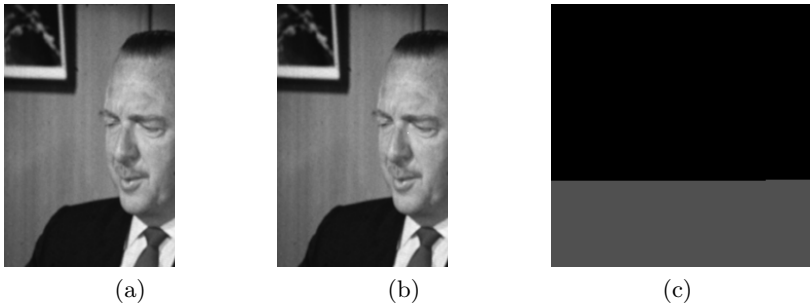

<div align="center">(a)   (b)   (c)</div>

Fig. 5 Illustration of the low sensitivity to the change of the plain-image: (a) the first plain-image $\mathbf{I}_0$; (b) the second plain-image $\mathbf{I}_1$ (only the center pixel is different from $\mathbf{I}_0$); (c) the differential cipher-image $\mathbf{I}_0' \oplus \mathbf{I}_1'$.

Problem 12. Low sensitivity to the change of plaintext. In some encryption architectures plaintexts with slightest differences are associated to very similar ciphertexts, which is a clear violation of the diffusion property.

This problem is specially relevant when considering the encryption of images. This being the case, the encryption scheme must guarantee that two images differing in just one pixel determine two totally different cipher-images. This requirement is not satisfied if encryption is performed through just one encryption round, as it occurs with the cryptosystem proposed in [44] (Case study 2.5). For that cryptosystem, given two plain-images $\mathbf{I}_0$ and $\mathbf{I}_1$ with only one pixel difference at the position (i, j), the difference will be permuted to a new position (i^*, j^*) according to the shuffling matrix $\mathbf{P}^*$. Then, because all plain-pixels before (i^*, j^*) are identical for the two plain-images, the ciphertexts will also be identical. This shows the low sensitivity of the image encryption scheme to changes in the plain-image. Figure 5 gives an example of this problem. It can be seen how the differential cipher-image is equal to zero for any pixel before (i^*, j^*) and equal to a constant value after that position.

2.3 Implementation Problems

Problem 13. Degradation of the efficiency of digital chaos-based cryptosystems based on chaotic systems. In digital chaos-based cryptography the encryption procedure is performed in discrete time and discrete space. Therefore, if the underlying chaotic system is defined in continuous domain, then it is necessary to apply some numerical methods to obtain the chaotic orbits. The application of these numerical methods increases the time to compute the orbits, and thus reduces the encryption efficiency.

The rule of evolution of continuous-time chaotic systems requires to solve a system of differential equations [49, p. 160]. From a general point of view, the solution of those differential equations cannot be accomplished analytically, and thus numerical methods must be applied. Numerical methods for the resolution of differential equations are highly time consuming. As a result, for a given encryption architecture with the underlying chaotic system defined in continuous-time, the encryption time is greater than the encryption time obtained when the chosen dynamical system is a chaotic map.

Case study 2.6 ([28]). *Evaluation of the encryption time of the chaos-based cryptosystem proposed in [44].*

For the sake of argument, let us recall our cryptanalytic work [28] on the chaos-based cryptosystem proposed in [44]. This cryptosystem is intended to encrypt images through the concatenation of a shuffling and masking procedures. The shuffling procedure is based on the iteration of the logistic map, whereas masking is built upon the iteration of two continuous-time chaotic systems: Lorenz [75] and Chen Systems [35]. The Lorenz System is given by

$$\frac{dx_1}{dt} = \alpha x_1 + \alpha x_2,$$
$$\frac{dx_2}{dt} = -x_1 x_3 + \beta x_1 - x_2,$$
$$\frac{dx_3}{dt} = x_1 x_2 - \rho x_3, \tag{26}$$

where α, β, and ρ are control parameters. The Lorenz system is chaotic for a set of parameters $\alpha = 10$, $\beta = 28$, and $\rho = 8/3$. On the other hand, the Chen System is defined as

$$\frac{dx_1}{dt} = \eta(x_2 - x_1),$$
$$\frac{dx_2}{dt} = (\sigma - \eta)x_1 - x_1 x_3 + \sigma x_2,$$
$$\frac{dx_3}{dt} = x_1 x_2 - \delta x_3, \tag{27}$$

being chaotic for a different set of parameters such as $\eta = 35$, $\sigma = 28$, and $\delta = 3$. Because the chaotic iterations of Lorenz and Chen Systems involve complicated numerical differential functions, the encryption speed is expected to be very slow compared with other traditional ciphers. To asses this fact, we derived a modified encryption scheme from the original one by replacing Lorenz and Chen Systems with the logistic map, and then compared the encryption speeds of the two cryptosystems. Both cryptosystems were implemented using MATLAB on a PC with a 1.6 GHz processor and 512 MB of RAM. For images of size 256×256, the typical encryption time for the original cryptosystem in [44] was around 5.8 s, while the modified cryptosystem based on the logistic map required on average around 1.2 s to

encrypt an image. The experiments have clearly shown that using continuous chaotic systems can drastically reduce the encryption speed. Since there are also no other obvious merits in using continuous chaotic systems rather than a simple discrete-time chaotic map, the use of Lorenz and Chen Systems in the image encryption scheme under study is unnecessary. Instead, these continuous-time chaotic systems can be replaced by a simpler discrete-time chaotic map without compromising the security. This statement is a general rule when designing encryption procedures working in discrete time.

Problem 14. Non-invertible encryption procedure. The iteration of the chaotic systems sustaining chaos-based cryptosystems implies working with real numbers. Since the implementation of chaos-based cryptosystems is done with finite precision arithmetic, round-off operations could lead to a non-invertible encryption procedure.

One critical point when working with dynamical systems and the analysis of their dynamics is the selection of a proper simulation framework. Indeed, the computer-based analysis of dynamical systems could lead to some conclusions different from those expected from theory. This divergence also influences and conditions chaos-based cryptosystems, as pointed out in [76] for the case of CSK. Thus, if the characteristics and problems of finite-precision are not handled properly, then it is possible that the orbits generated as base of the encryption procedure can not be regenerated exactly during the decryption stage and, consequently, the original plaintext can not be recovered even when the key is known. This problem is not only relevant for fixed-point arithmetic but also for floating-point one. Indeed, the round-off quantization errors could lead to the occurrence of a non-invertible function for encryption and, as a result, the decryption process will be impossible.

Case study 2.7 ([29, 26, 7]). *Non-invertible cryptosystems defined in [93, 34, 94].*

The cryptosystems introduced in [93, 34, 94] are examples of the consequences of not handling conveniently the limitations of finite precision arithmetics, as we have pinpointed in [29, 26, 7]. To clarify the problem under consideration, let us recall the scope depicted in [93], whose goal is to encrypt images. The cryptosystem described in [93] generates a ciphertext consisting of a number of real values. Encryption is performed through j encryption rounds, being $\{x_c^i(r)\}_{i=1}^J$ ($c = $ R, G and B, $r \in \{1, 2, \ldots, j-1\}$) the output in the r-th encryption round corresponding to color component c of the i-th pixel of the image of length $J = M \times N$. All the operations to encrypt an image in [93] are performed using floating-point arithmetic. The output of the r-th round is given as $x_c^i(r) = x_n + x_c^i(r-1)$, where x_n is the resulting value of iterating the logistic map n times from x_0. Hence, if during the decryption process we want to recover $x_c^i(r-1)$ (the original value of the i-th element in the last round), we have to iterate n times the logistic map from x_0 to get x_n and, after that, to substract this value from $x_c^i(r)$. However, the resulting value of this previous

operation might not match the actual value of $x_c^i(r-1)$, due to the *wobbling precision problem* that exists when dealing with floating-point operations [48, p. 39]. This wobbling precision problem also causes the resulting guessed value of $x_c^i(r-1)$ to depend on the cryptosystem implementation. Therefore, if an image is encrypted on one platform and decrypted on another, and the implementations of floating-point arithmetics on both platforms are not compatible with each other, then the decrypted image might not match the original one. In [93] the cryptosystem was implemented using Microsoft Visual C# .NET 2005 and no comment was given about the wobbling precision problem in the decryption process. However, we have experimentally verified that this problem indeed exists when the cryptosystem is implemented using MATLAB on a PC with a 3 GHz processor and 2 GB RAM. A very useful measure of the performance of the decryption procedure is the Mean Square Error or MSE. For P and P' being a plain image and the decrypted image respectively, the MSE for the color component c is defined as

$$MSE_c = \sum_{i=1}^{m}(P_c^i - P'^i_c)^2/J, \tag{28}$$

where $c \in \{R, G, B\}$, $J = M \times N$ is the number of pixels of the images considered and the sequences $\{P_c^i\}_{i=1}^{J}$ and $\{P'_c\}_{i=1}^{J}$ are the result of scanning P and P' in the raster order. Consequently, for a well designed encryption/decryption scheme the MSE should be 0 for each color component. Unfortunately, for the cryptosystem under study, the values of MSE for all three color components are generally not equal to 0 due to the wobbling precision problem associated to the floating-point arithmetic. In order to evaluate the underlying decryption error of the cryptosystem defined in [93], a 512×512 plain-image "Lena" was encrypted and decrypted using the same key $(n, j, \lambda) = (30, 1, 3.9)$. The results showed that the three MSEs obtained for the red, green and blue components of the decrypted image with respect to the original one were 6.49, 0.018, 0.057, respectively. For another key $(n, j, \lambda) = (30, 3, 3.9)$, the obtained MSEs were 206.96, 123.45, 58.65, respectively. Figure 6 shows the decrypted image and the error image when the cryptosystem was implemented in MATLAB using a third key $(n, j, \lambda) = (5, 2, 3.9)$.

Problem 15. Dynamical degradation. The implementation of chaotic systems in finite precision in digital computers leads often to dynamical properties completely different from the theoretical and expected ones. If this deviation is not considered during the design of chaos-based cryptosystems, it could imply a reduction of the performance and even a compromise of the security of the resulting cryptosystem.

This problem is closely related to the previous one, although the point of interest moves to degradation of dynamical properties of the implemented chaotic system with respect to the theoretical model. Consequently, the
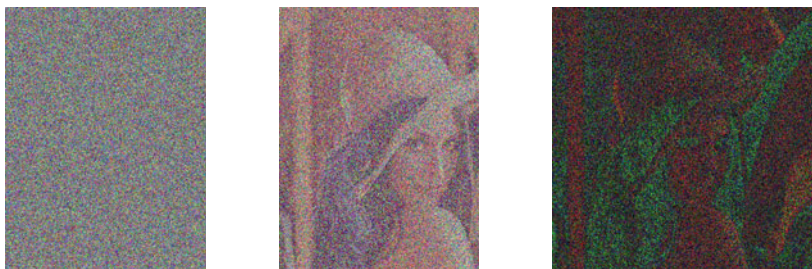
Fig. 6 Simulations with MATLAB (a) Ciphertext of the plain-image "Lena" (b) Recovered image of "Lena" using the same key (c) The error image between the original and the recovered "Lena".

design of an encryption scheme using a chaotic system must be done by considering its practical implementation (not only the theoretical model). In [5] some consequences of the dynamical degradation of a chaotic map are shown in the context of cryptography, whereas in [68] one can find a thorough analysis of the dynamical degradation of a specific chaotic map and some ways to overcome this problem.

Problem 16. Lack of details in the description. According to Kerckhoffs' principle [80, p. 14], the security of a cryptosystem can not be based on the secrecy of its encryption and decryption procedures. In other words, when dealing with the security of cryptosystems, everything is known except the key. Furthermore, the key of any cryptosystem has to be easy to establish and to exchange, and the key space must be defined in an explicit and clear way.

The consecution of security through obscurity is something to avoid when designing an encryption scheme. All the operations involved in the encryption/decryption procedures must be unmistakably explained, and the secret key must be clearly specified along with an exact estimation of the size of the key space. The security of the cryptosystem must be only related to the difficulty of recovering the key, and it can not depend on the lack of knowledge about the inner operation of the encryption and decryption procedures. Moreover, this lack of details implies a lack of security because without a careful investigation by the cryptography community, many security holes might not be able to be distinguished by the designers themselves.

Case study 2.8 ([26, 28]). *Non-exhaustive definition of the encryption and decryption scheme implying loss of chaoticity or randomness of keystreams [34, 44].*

In [21, Sec. 2.2] we have analyzed the loss of chaoticity in the cryptosystem defined in [34]. Indeed, in that cryptosystem part of the key is given by a set of functions changing the values of the control parameters of a Hénon map as the plaintext is encrypted. The authors of [34] do not define explicitly and rigorously those functions, which could result in a security flaw, as we have shown in [26] for the set of functions given by Eqs. (4)-(6). Another example of the kind of problem under consideration can be found in some encryption schemes built upon continuous-time chaotic systems. Certainly, when working with this type of chaotic systems it is necessary to use numerical methods to compute the chaotic orbits. The decryption procedure requires to generate the same chaotic orbits as in the encryption stage and, consequently, its computation must be done using the same numerical method and the same time step. Moreover, the influence of both the numerical method and the time step on the performance of the cryptosystem must be thoroughly evaluated. Let us take up again the cryptosystem defined in [44]. The masking stage of that cryptosystem is driven by a keystream derived from the orbits of Lorenz and Chen Systems. In [44], the authors did not say anything about the time step τ of iterating the Lorenz and Chen systems. However, the randomness of the keystream is tightly dependent on the value of the time step. As an extreme example, if $\tau = 10^{-20}$, we will get a keystream of identical elements (according to the algorithm described in Sec. 2.3 of [44]).

3   Design Rules for Chaos-Based Cryptography

According to the above problems, we proceed with the concretion and systematization of the guidelines to observe when designing a chaos-based digital cryptosystem. These guidelines, that can be interpreted as the extension of the set of rules provided in [6].

Rule 1. *Exhaustive and rigorous definition of the chaotic encryption and decryption algorithms.*

The design of any encryption system must be guided by Kerckhoffs' principles, and thus the consecution of security through obscurity must be totally discarded. The designed cryptosystem must be easily reproducible, in order to make its implementation, use and further analysis easy. Indeed, guaranteeing the security of an encryption procedure is a quite complex and elusive problem, so the more people participate in the analysis, the more complete the security assessment will be. Regarding specifically chaos-based cryptosystems, the design rule here referred implies that:

a) The encryption/decryption algorithms must assure control parameters determining the chaotic behavior of the selected maps.
b) The final cryptosystems must be evaluated by means of the classical cryptanalytic framework [21, Sec. 1.2.2].

c) It must be confirmed that an attacker cannot get enough information about the underlying chaotic orbits, and thus she cannot carry out an estimation of control parameters and/or initial condition.

Rule 2. *Exhaustive and rigorous definition of the key and the key space.*

In chaos-based cryptography it is mandatory to specify clearly and carefully the relationship between the secret key and the parameters determining the temporal evolution of the underlying chaotic maps, i.e., the control parameters and the initial conditions. In some cryptosystems either the control parameters or the initial conditions or both are part of the secret key, whereas in others they are just design parameters and, consequently, publicly known. Another possibility is that the secret key determines the values of the control parameter(s) and initial condition(s). In all situations it must be verified that the underlying dynamical systems involved in the considered chaos-based cryptosystem evolve as required, i.e., in a chaotic way. In other words, the values of the control parameters used during encryption and decryption must determine a positive value of the largest component of the Lyapunov Exponent (LE). In this respect, this rule is related to the diffusion property, since it is intended to make the relationship between the key (or the plaintext) and the ciphertext as complex as possible. The goal is to erase any possible pattern or redundancy in the ciphertext, and thus to avoid inference of the secret key from the ciphertext. In the context of chaos-based cryptography, diffusion is connected to the local rate of divergence of orbits. As a result, chaotic maps with high values of LE must be selected. It is also possible to use chaotic maps with small LE if the encryption of each unit of plaintext is performed iterating several times the chaotic map. Nevertheless, it implies a reduction of the efficiency of the cryptosystem, and thus it is preferable to discretize the key space to guarantee the *avalanche effect*, i.e., the result of encrypting a plaintext with two slightly different keys must produce totally different ciphertexts. The tools for verification of the avalanche effect are the same used in the assessment of parameter mismatch (next rule), i.e., the statistical distance and the MRE (MultiResolution Entropy) . The discretization of the key space implies a reduction of its size, which could result in a degradation of the protection against brute-force attacks. A possible solution to this problem is to discretise the orbits of the chaotic maps instead of the key space. As we have shown in [21, Sec. 2.2] for the skew tent map, this strategy determines an increasing of the LE and, consequently, of diffusion.

On the other hand, the determination of LE entails some inaccuracies [90] and, consequently, it is highly advisable to analyze the chaoticity of orbits using auxiliary tools as the entropy measures referred in [21, Secs. 2.4 and 2.5]. Finally, either LE or the different entropy measures can bring to light a somehow one-to-one relationship between the rate of divergence of orbits and the control parameter(s). In this case, if a chaos-based cryptosystem allows an estimation of the rate of divergence, then it could be possible to estimate

the control parameter(s), which represents a vulnerability of the cryptosytem being the control parameter(s) part or determined by the secret key.

Rule 3. *Selection of chaotic maps with high sensitivity to control parameter mismatch.*

The size of the key space of any cryptosystem must be large enough to avoid the feasibility of a brute-force attack. This is a common requirement of all encryption systems, and it has to be fulfilled in accordance with the computational capacity of any possible attacker. As it is pointed out in [6, Rule 15], today's computer speed requires a key space of size larger than 2^{100}. As indicated by the previous rule, in digital chaos-based cryptography the specification of the key space is mainly guided by the calculation of the LE. Consequently, the resolution in the computation of the LE is a measure of the maximum number of possible keys, and thus an approximation of the size of the key space. To get a number of keys larger than $2^{100} \approx 10^{30}$, the resolution must be 10^{-30}. However, with that resolution, thousands of keys would become equivalent, unless there is a strong sensitivity to parameter mismatch. It implies that the concretion of the key space must be accompanied of an exhaustive analysis of the orbits generated for each value of the control parameters. Indeed, it must be tested that the orbits are different enough to assure that the encryption procedure possesses confusion and diffusion properties. Useful tools in this regard are the statistical distance [21, Sec. 2.6] and the MRE [21, Sec 2.5.2].

Rule 4. *The selected chaotic map should not allow total characterization of its dynamics from partial knowledge of this dynamics.*

The total characterization of the dynamics of a chaotic map requires the knowledge of the initial condition and the control parameter(s). When considering a chaos-based cryptosystem, it could be possible for an attacker to guess either the initial condition or control parameter(s). Upon the guessed information, the attacker could use some of the general attack strategies [21, Sec. 1.2.2] to get some additional information about the orbits of the underlying chaotic map. For some chaotic maps, this additional information and the guessed information drive to the estimation of the rest of parameter(s) describing the dynamics of the map. For example, if the chaotic map selected for an encryption architecture is a unimodal map, and the encryption architecture allows to infer the symbolic sequences of the map through some attack, then the knowledge of the control parameter allows to recover the initial condition.

Rule 5. *Analysis of the performance of chaotic orbits as source of entropy.*

From the point of view of cryptography, the appealing of chaos is mainly motivated by its random-like behavior. Actually, "the battle" of any cryptographer is to look for sources of uncertainty that can be further used to

conceal the information. The design of a cryptosystem is the specification of a series of transformation procedures based on sources of indetermination and applied on the source of information. In chaos-based cryptography, all or some of the transformation procedures use chaos as source of indetermination. Since the security of the whole cryptosystem lies on the efficiency of each transformation procedure, the entropy must be evaluated. Again, the assessment can be done upon tools as those described in [21, Secs. 2.6 and 2.5.2]. Furthermore, this assessment can also be refined by considering every transformation procedure as a *Pseudo Random Number Generator* (PRNG). Upon this consideration, evaluation can be fulfilled using the battery of statistical tests of the National Institute of Standard & Technology (NIST) [84]. Nevertheless, if chaos is used as base of a stream cipher, then it is also necessary to analyze the possibility of reconstructing the symbolic dynamics in order to verify the feasibility of estimation for the control parameter(s) and initial condition(s) as it is done in [21, Chapters 3 and 4].

Rule 6. *Chaotic maps with uniform invariant density functions and measure of the invariant set independent of the control parameters should be used.*

If this requirement is satisfied, then the chaotic cryptosystem possesses the confusion property. If the underlying dynamical system evolves, as expected, chaotically, then it possesses the ergodic property and thus orbits are statistically independent of the control parameter(s) and initial condition(s). As a result, the ciphertext should be statistically undistinguishable from the output of a truly random function, and should be statistically the same for all the keys.

Rule 7. *The ciphertext space must be defined in such a way that the reconstruction of the dynamics of the underlying chaotic maps is not possible.*

Ciphertexts of chaos-based cryptosystems must not leak information about the symbolic dynamics, the return map or any other shortcut to reconstruct the dynamics of the underlying chaotic maps.

Rule 8. *The encryption/decryption time must not depend on the value of the secret key of a chaos-based cryptosystem.*

If it is necessary to perform encryption through several rounds and several iterations of the underlying chaotic maps, then the number of encryption rounds and the number of iterations must be publicly known. They can not be considered as part of the secret key, since a mere analysis of the encryption/decryption time allows an estimation of those values.

Rule 9. *Resistance to classical attacks.*

The cryptanalysis of chaos-based cryptosystems combined techniques from the theory of dynamical systems and from the cryptanalysis of conventional cryptography. In this concern, it must be verified the robustness of the cryptosystem against known-plaintext, chosen-plaintext, known-ciphertext (or ciphertext-only), and chosen-ciphertext attacks [100, p. 95]. Specialized attacks must also be evaluated [99]. For digital chaos-based block ciphers resistance to differential [102] and linear cryptanalysis [53] must be proved.

Rule 10. *Resistance to application-specific attacks.*

The encryption of information with special features must be defined carefully in order to avoid the leaking of such features in the ciphertext. This is the case of digital images and videos. In digital images (videos) there exists strong correlation between different pixels (transform coefficients), which can be used to develop some effective correlation-based attacks.

4   Chaos-Based Cryptography: A Conclusion But Not the End of the Road

Recalling the main conclusions of our works in the field of cryptanalysis, a theoretical framework is next proposed in order to fulfill the set of rules previously explained. This theoretical basis is conceived as a manner to achieve at least the same level of security and efficiency of conventional cryptography, but using the theory of chaotic dynamical systems as core instead of the theory of numbers.

Nowadays, the information (whether analog or digital) is processed by computers. This means that however chaos enters the encryption (key-stream generation, masking, etc.), the finite precision of the computer (or any other finite-state machine for this matter) will degrade chaotic orbits to periodic orbits — occasionally of a very long period. Although this may be acceptable from a practical point of view, it is not from a theoretical one. Put in other words, the concept and virtues of chaos are not exportable without change to the realm of discrete and finite mathematics.

If there is no chaos in a finite-state space, what is then a "chaotic" cipher? To answer this question in a satisfactory way, one has to go away from the real numbers (there are no real numbers in the real world) and use algorithms on integer numbers or finite fields, while preserving the spirit of chaos-based cryptography. A paradigmatic example was given by Pichler and Scharinger, and reproduced in [43], where several chaotic maps were discretized and used for image encryption. Let us remember it at this point.

Let $I^2 = [0,1] \times [0,1] \subset \mathbb{R}^2$ endowed with the Lebesgue measure, and let $B : I^2 \to I^2$ be the *baker map*,

$$B(x,y) = \begin{cases} (2x, \frac{1}{2}y), & 0 \leq x < \frac{1}{2}, \\ (2x - 1, \frac{1}{2}y + \frac{1}{2}), & \frac{1}{2} \leq x \leq 1. \end{cases}$$

The baker map is a chaotic bijection of the unit square I^2 onto itself. This map stretches the left rectangle $[0, 1/2) \times [0, 1)$ horizontally onto the "bottom" rectangle $[0, 1) \times [0, 1/2)$, while the right rectangle $[1/2, 1) \times [0, 1)$ is similarly mapped onto the "top" rectangle $[0, 1) \times [1/2, 1)$.

Pichler and Scharinger generalized this map in the following way [43]. The unit square I^2 is now divided into k vertical rectangles $[F_{i-1}, F_i) \times [0, 1)$, $i = 1, ..., k$, $F_i = p_1 + ... + p_i$, $F_0 = 0$, with $p_1 + ... + p_k = 1$. The *generalized baker map* stretches each rectangle horizontally by the factor of $1/p_i$, while it is contracted vertically by the factor of p_i. Finally, all transformed rectangles are stacked on top of each other. Formally,

$$B(x,y) = \left(\frac{1}{p_i}(x - F_i), p_i y + F_i \right)$$

for

$$(x, y) \in [F_i, F_i + p_i) \times [0, 1).$$

If we denote this map by $B_{(p_1,...,p_k)}$, then the standard baker map corresponds to $B_{(1/2,1/2)}$. The generalized baker map inherits all important properties of the baker map such as sensitivity to initial conditions and parameters, mixing and bijectiveness.

One possibility of discretizing $B_{(p_1,...,p_k)}$ on an integer lattice is the following. Let N be an integer, and let $n_1, ..., n_k$ be integers such that each n_i divides N, and $n_1 + ... + n_k = N$. Denoting $N_i = n_1 + ... + n_i$, the lattice point (r, s), with $N_{i-1} \leq r < N_i$, $N_0 = 0$, and $0 \leq s < N$, is mapped by the *discretized generalized baker map* $B_{(n_1,...,n_k)}$ to[2]

$$B_{(n_1,...,n_k)}(r, s) = \left(\frac{N}{n_i}(r - N_i) + s \bmod \frac{N}{n_i}, \frac{n_i}{N}\left(s - s \bmod \frac{N}{n_i} \right) + N_i \right).$$

Coming back to the question, what a chaotic cipher on a finite-state space should be, the answer (intentionally vague) proposed in [15] and repeated in the next section, was inspired by this and similar examples. Other technique, studied in [59] in the framework of *discrete chaos*, furnish permutations by truncation of chaotic orbits. Since the discretized map, whatever the discretization method used, is desired to somehow inherit the properties of the continuous chaotic map, the former should become increasingly close to the latter in the 'continuous' limit. In the case of the discretized baker map, the

---

[2] The cipher proposed in [43] based on the discretized generalized baker map, was cryptanalyzed in [98].

continuous limit refers to an ever finer coarse-graining of the unit square. In the framework of discrete chaos, the continuous limit refers to ever longer orbits segments, and the closeness refers to the convergence of the discrete LE(s) of the permutation to the LE(s) of the chaotic map.

4.1 Chaos-Based Cryptography on Integer Numbers and Finite Fields

The idea underlying the above construction of the discretized generalized baker map (namely, to define a permutation via discretization of a chaotic map) provides an approach to chaos-based cryptography that is compliant with the standards of conventional cryptography. This idea was captured with different degrees of generality in [16, 102] (definition of permutations via periodic approximations of automorphisms) and also in [15] (definition of chaotic cryptographic primitives). The minimal framework we need for this formulation is that of measure theory.

We say that $(X, \mathcal{A}, \mu)$ is a measure space if X is a non-empty set, $\mathcal{A}$ is a sigma-algebra of subsets of X and μ is a measure on $(X, \mathcal{A})$. If $\mu(X) < \infty$, $(X, \mathcal{A}, \mu)$ is called a finite-measure space. Typically, X will be a compact topological or even metric space (think of a finite interval of $\mathbf{R}^n$ or of an n-torus). We say that $\mathcal{P} = \{A_1, ..., A_N\} \subset \mathcal{A}$ is a partition of X if $\cup_{n=1}^{N} A_n = X$ and $A_i \cap A_j = \emptyset$ for all $i \neq j$. A norm of $\mathcal{P}$ is a measure of its 'coarseness' (e.g., maximal length, maximal diameter, etc. of the elements of $\mathcal{P}$). In order to streamline the notation, we will usually refer only to X, with the underlying $\mathcal{A}$ and μ being understood. Furthermore, chaos refers to dynamical systems and these call for measure-invariant self-maps on finite-measure spaces, i.e., maps $f : X \to X$ such that $f^{-1}A \in \mathcal{A}$ and $\mu(f^{-1}A) = \mu(A)$ for all $A \in \mathcal{A}$.

A generalization of the discretized generalized baker map is the following [36]. Suppose f is an automorphism of the finite-measure space $(X, \mathcal{A}, \mu)$, i.e., f is a one-to-one map of X onto itself such that both f and f^{-1} are μ-invariant. We consider sequences of finite partitions $\{\mathcal{P}_n\}$ of the space X, $\mathcal{P}_n = \{P_k^{(n)} : 1 \leq k \leq q_n\}$, such that $\lim_{n \to \infty} \mathcal{P}_n = \mathcal{E}$ (the partition of X into separate points) and sequences of automorphisms $\{f_n\}$ such that f_n preserves $\mathcal{P}_n$ (i.e., f_n sends every element of $\mathcal{P}_n$ into an element of the same partition). We say that an automorphism f of the space $(X, \mathcal{A}, \mu)$ possesses an approximation by periodic transformations with speed $\vartheta(n)$, if there exists a sequence of automorphisms f_n preserving $\mathcal{P}_n$ such that

$$\sum_{k=1}^{q_n} \mu\left(f(P_k^{(n)}) \triangle f_n(P_k^{(n)})\right) < \vartheta(q_n), \quad n = 1, 2, \ldots.$$

where $\triangle$ stands for symmetric set difference and ϑ is a function on the integers such that $\vartheta(n) \to 0$ monotonically. The sequence $(\mathcal{P}_n, f_n)$ is a discrete approximation of (X, f).

A further generalization brings us to the concept of *discrete approximation*. This time we leave deliberately open the way the 'discrete approximation' converges to the continuous map.

Definition 1. [15] *Let X be a finite-measure space and $f : X \to X$ a map. Let $X_\Delta = \{A_1, ..., A_{N(\Delta)}\}$ be a family of partitions of X, labelled by a parameter Δ, say, the partition norm, such that $\lim_{\Delta \to 0} X_\Delta = \mathcal{E}$, the partition of X into separate points. Furthermore, given a family of maps $f_\Delta : X_\Delta \to X$, define the extensions $\bar{f}_\Delta : X \to X$ as $\bar{f}_\Delta(x) = f_\Delta(A_n)$ if $x \in A_n \in X_\Delta$. We say that (X_Δ, f_Δ) is a discrete approximation of (X, f) if, moreover, $\lim_{\Delta \to 0} \bar{f}_\Delta = f$ in some relevant sense (depending on the structure we put on X).*

This definition of discrete approximation is an idealization of what actually happens when computing real functions with computers. Intuitively, discrete approximation of chaotic maps are expected to generate permutations with 'nice' mixing properties and, therefore, appropriate for cryptographic applications.

Definition 2. [15] *Discrete approximations of chaotic systems (X, f) in form of permutations $(\mathbf{Z}_M, F_M)$ are called chaotic cryptographic primitives. Furthermore, we say that a cryptographic algorithm is chaotic if some of its building blocks is a chaotic cryptographic primitive.*

Thus, a stream cipher whose keystream is generated by a chaotic primitive is a chaos-based cryptosystem. Let us present a few examples of chaotic primitives (see also [15, 17]).

Example 1. The Renyi map, $\phi_\beta : [0, 1) \to [0, 1)$, is defined as

$$\phi_\beta(x) = \beta x \bmod 1,$$

where $1 < \beta \in \mathbb{R}$. A discretized (or digitalized) Renyi map f can be defined on the set $\{0, 1, ..., 2^n - 1\}$ by

$$f(k) = \lfloor \beta \cdot k \rfloor \bmod 2^n.$$

The derivation proceeds via the approximation of real numbers in $[0, 1)$ by dyadic rationals. The *discretized Renyi map* was used in [1] to generate random numbers.

Example 2. The Chebyshev polynomial maps $T_n : \mathbf{R} \to \mathbf{R}$ of degree $n = 0, 1, ...$ are defined by the recursion

$$T_n(x) = 2x T_{n-1}(x) - T_{n-2}(x) \text{ for } n \geq 2,$$

with $T_0(x) = 1$, $T_1(x) = x$. The interval $[-1, 1]$ is invariant under the action of the map T_n: $T_n([-1, 1]) = [-1, 1]$. Alternatively, one can define

$$T_n(x) = \cos(n \arccos x), \quad -1 \le x \le 1.$$

The Chebyshev polynomial T_n restricted to $[-1, 1]$ is a well-known chaotic map for all $n \ge 2$: it has a unique absolutely continuous invariant measure,

$$\mu(x) = \frac{1}{\pi\sqrt{1-x^2}},$$

and LE $\ln n > 0$ with respect to μ. For $n = 2$, the Chebyshev map reduces to the logistic map.

It is straightforward to prove that Chebyshev polynomials have the semi-group property:

$$T_r(T_s(x)) = T_s(T_r(x)) = T_{rs}(x).$$

Let p be a prime number so as $\mathbb{Z}_p$ is a field. The *Chebyshev map* over the finite field $\mathbb{Z}_p$, $F_{n,p} : \{0, 1, \ldots, p-1\} \to \{0, 1, \ldots, p-1\}$, is defined as

$$F_{n,p}(\xi) = T_n(\xi) \pmod{p}.$$

The semi-group property of the 'discrete' (or finite state) Chebyshev maps $F_{n,p}$ can be used in key-exchange protocols or even in public-key algorithms [79, 58, 57]. However, this kind of maps (as any other) does not guarantee security itself, and a convenient key-exchange protocol along with an adequate encryption procedure should be provided to achieve not only security, but also computational efficiency (see the cryptanalysis of [110] in [2]).

Other examples of chaotic primitives used in ciphers published in the literature, include the discrete logistic map and toral automorphisms (see, e.g., [58, 77]).

As a last example, let us discuss a synchronization-based cipher.

Example 3. *Message-embedding* is an encryption algorithm in which the message m_t is directly injected (or "embedded") in a chaotic dynamic $f_\theta : J \to J$, where $J \subset \mathbb{R}^q$ and $\theta = (\theta_1, \ldots, \theta_D)$ is an, in general, multi-component parameter which acts as the secret key or is part of the secret key of the cipher. In the simplest versions, encryption takes place at the *sender* according to one of the following schemes:

$$\text{(I)} \begin{cases} x_{t+1} = f_\theta(x_t, m_t) \\ y_t = h_\theta(x_t, m_t) \end{cases} \text{ and (II)} \begin{cases} x_{t+1} = f_\theta(x_t, m_t) \\ y_t = h'_\theta(x_t) \end{cases}, \qquad (29)$$

the difference being the so-called *relative degree* r [81]; $r = 0$ in case (I) and $r > 0$ in case (II). h_θ is called the *output function* of the sender, since y_t is the message conveyed from the sender to the receiver through the communicaction channel. In general, $h_\theta : J \to \mathbb{R}^{q'}$ with $q' \le q$ (ideally $q' = 1$). The *receiver* is a kind of 'mirrored' dynamical system, generated by a family of maps $\tilde{f}_\theta$ and endowed with output functions $\tilde{h}_\theta$. The retrieval of the message (plaintext) at the receiver is achieved in two steps: (i) *synchronization* [45, 108], based on a

suitable choice of $\tilde{f}_\theta$, and (ii) *estimation* of m_t by means of a suitable function which depends on the internal state $\hat{x}_t$ of the receiver and the output y_t, the only information transmitted from the sender to the receiver.

Two mechanisms have been proposed in the literature to recover m_t: (i) the inverse system approach [41] and, in case of noisy channels, (ii) the unknown input observer approach (UIO) [52, 82]. In both cases, the equations governing the receiver are:

$$\begin{cases} \hat{x}_{t+r+1} = \tilde{f}_\theta(x_{t+r}, y_t, ..., y_{t+r}) \\ \hat{m}_{t+r} = g_\theta(\hat{x}_{t+r}, y_t, ..., y_{t+r}) \end{cases},$$

with $r \geq 0$ and g such that

$$\hat{m}_{t+r} = g_\theta(\hat{x}_{t+r}, y_t, ..., y_{t+r}) = m_t \text{ when } \hat{x}_{t+r} = x_t.$$

The existence of an inverse system or an UIO is guaranteed under the assumption that the system (29-I) or (29-II) is left invertible. In our setting, *left invertibility* means that there exists an integer $R \geq 0$ such that the input m_t is uniquely determined by the knowledge of the state x_t along with the output sequence $y_t, ..., y_{t+R}$.

Message-embedding is very attractive because synchronization is assured without any restriction on the variation rate of m_t. For cryptographic applications it is sometimes convenient to decompose the dynamic f_θ in two actions in order to eventually incorporate boolean and arithmetic operations. Such is the case of the *hybrid message-embedding* technique, where the chaotic dynamic is decomposed as follows:

$$(\text{I'}) \begin{cases} u_t = \nu_e(x_t, m_t) \\ x_{t+1} = q_\theta(x_t, u_t) \\ y_t = r_\theta(x_t, u_t) \end{cases} \text{ and } (\text{II'}) \begin{cases} u_t = \nu_e(x_t, m_t) \\ x_{t+1} = q_\theta(x_t, u_t) \\ y_t = r'_\theta(x_t) \end{cases}, \qquad (30)$$

for sender systems with relative degree $r = 0$ (I') and $r > 0$ (II'). u_t is sometimes called the *pre-ciphertext* in hybrid message-embedding.

Message-embedding and hybrid message-embedding can be used with letters taken from finite fields (and boolean operations in the hybrid scheme) [81]. Under the further assumption of *flatness* [81], they have been proved to be structurally equivalent to a self-synchronizing stream cipher —a well-tested architecture in conventional cryptography. We conclude that the security of these chaos-based ciphers relies on the security of the corresponding chaotic primitives (f_θ in (29), and q_θ and/or ν_e in (30)).

References

1. Addabbo, T., Alioto, M., Fort, A., Pasini, A., Rocchi, S., Vignoli, V.: A class of maximum-period nonlinear congruential generators derived from the reńyi chaotic map. IEEE Transactions on Circuits and Systems–I: Regular Papers 54(4), 816–828 (2007)

2. Alvarez, G.: Security problems with a chaos-based deniable authentication scheme. Chaos, Solitons & Fractals 26(1), 7–11 (2005)
3. Alvarez, G., Arroyo, D., Nunez, J.: Application of Gray code to the cryptanalysis of chaotic cryptosystems. In: 3rd International IEEE Scientific Conference on Physics and Control (PhysCon 2007), September 3-7, Potsdam, Germany (2007), http://lib.physcon.ru/?item=1358
4. Alvarez, G., Li, S.: Estimating short-time period to break different types of chaoitc modulation based secure communications. arxiv:nlin.CD/0406039 (2004), http://arxiv.org/abs/nlin/0406039
5. Alvarez, G., Li, S.: Breaking an encryption scheme based on chaotic baker map. Physics Letters A 352(1-2), 78–82 (2006)
6. Alvarez, G., Li, S.: Some basic cryptographic requirements for chaos-based cryptosystems. International Journal of Bifurcation and Chaos 16(8), 2129–2151 (2006)
7. Alvarez, G., Li, S., Hernandez, L.: Analysis of security problems in a medical image encryption system. Computers in Biology and Medicine 37(3), 424–427 (2007)
8. Alvarez, G., Li, S., Montoya, F., Romera, M., Pastor, G.: Breaking projective chaos synchronization secure communication using filtering and generalized synchronization. Chaos, Solitons & Fractals 24(3), 775–783 (2005)
9. Alvarez, G., Montoya, F., Pastor, G.: Cryptanalysis of a discrete chaotic cryptosystem using external key. Physics Letters A 319, 334–339 (2003)
10. Alvarez, G., Montoya, F., Romera, M., Pastor, G.: Cryptanalysis of a chaotic encryption system. Physics Letters A 276, 191–196 (2000)
11. Alvarez, G., Montoya, F., Romera, M., Pastor, G.: Cryptanalysis of an ergodic chaotic cipher. Physics Letters A 311, 172–179 (2003)
12. Alvarez, G., Montoya, F., Romera, M., Pastor, G.: Breaking parameter modulated chaotic secure communication system. Chaos, Solitons & Fractals 21(4), 793–797 (2004)
13. Alvarez, G., Montoya, F., Romera, M., Pastor, G.: Cryptanalysis of dynamic look-up table based chaotic cryptosystems. Physics Letters A 326, 211–218 (2004)
14. Alvarez, G., Montoya, F., Romera, M., Pastor, G.: Keystream cryptanalysis of a chaotic cryptographic method. Computer Physics Communications 156, 205–207 (2004)
15. Amigó, J., Kocarev, L., Szczepanski, J.: Theory and practice of chaotic cryptography. Physics Letters A 366(3), 211–216 (2007)
16. Amigó, J.M., Szczepanski, J.: Approximations of dynamical systems and their applications to cryptography. International Journal of Bifurcation and Chaos 13, 1937–1948 (2003)
17. Amigó, J.M.: Chaos-Based Cryptography. In: Kocarev, L., Galias, Z., Lian, S. (eds.) Intelligent Computing Based on Chaos. Studies in Computational Intelligence, vol. 184, pp. 291–313. Springer, Heidelberg (2009)
18. Amigó, J.M., Szczepanski, J., Kocarev, L.: A chaos-based approach to the design of cryptographically secure substitutions. Physics Letters A 343, 55–60 (2005)

19. Argenti, F., Benzi, S., Re, E.D., Genesio, R.: Stream cipher system based on chaotic maps. In: Proceedings of SPIE Mathematics and Applications of Data/Image Coding, Compression, and Encryption III, vol. 4122, pp. 10–17. SPIE (2001)
20. Ariffin, M., Noorani, M.: Modified Baptista type chaotic cryptosystem via matrix secret key. Physics Letters A 372, 5427–5430 (2008)
21. Arroyo, D.: Framework for the analysis and design of encryption strategies based on discrete-time chaotic dynamical systems. Ph.D. thesis, ETSIA of the Polytechnic University of Madrid, Madrid, Spain (2009),
http://digital.csic.es/handle/10261/15668
22. Arroyo, D., Alvarez, G., Amigó, J.M.: Estimation of the control parameter from symbolic sequences: Unimodal maps with variable critical point. Chaos: An Interdisciplinary Journal of Nonlinear Science 19 (2009), Art. no. 023125
23. Arroyo, D., Alvarez, G., Amigó, J.M., Li, S.: Cryptanalysis of a family of self-synchronizing chaotic stream ciphers. Communications in Nonlinear Science and Numerical Simulation 16(2), 805–813 (2011)
24. Arroyo, D., Alvarez, G., Li, S.: Some hints for the design of digital chaos-basedcryptosystems: lessons learned from cryptanalysis. In: Second IFAC Conference on Analysis and Control of Chaotic Systems, Queen Mary, University of London (2009)
25. Arroyo, D., Alvarez, G., Li, S., Li, C., Fernandez, V.: Cryptanalysis of a new chaotic cryptosystem based on ergodicity. International Journal of Modern Physics B 23(5), 651–659 (2009)
26. Arroyo, D., Alvarez, G., Li, S., Li, C., Nunez, J.: Cryptanalysis of a discrete-time synchronous chaotic encryption system. Physics Letter A 372(7), 1034–1039 (2008)
27. Arroyo, D., Li, C., Li, S., Alvarez, G.: Cryptanalysis of a computer cryptography scheme based on a filter bank. Chaos, Solitons & Fractals 41, 410–413 (2009)
28. Arroyo, D., Li, C., Li, S., Alvarez, G., Halang, W.A.: Cryptanalysis of an image encryption scheme based on a new total shuffling algorithm. Chaos, Solitons & Fractals 41(5), 2613–2616 (2009)
29. Arroyo, D., Rhouma, R., Alvarez, G., Li, S., Fernandez, V.: On the security of a new image encryption scheme based on chaotic map lattices. Chaos: An Interdisciplinary Journal of Nonlinear Science 18 (2008), Art. no. 033112
30. Banerjee, S., Yorke, J.A., Grebogi, C.: Robust chaos. Physical Review Letters 80, 14 (1998)
31. Baptista, M.S.: Cryptography with chaos. Physics Letters A 240(1-2), 50–54 (1998)
32. Beth, T., Lazic, D.E., Mathias, A.: Cryptanalysis of cryptosystems based on remote chaos replication. In: EUROCRYPT 1994. LNCS, vol. 950, pp. 318–331. Springer, Heidelberg (1994)
33. Brumley, D., Boneh, D.: Remote timing attacks are practical. In: Proceedings of the 12th USENIX Security Symposium, pp. 1–14. USENIX Association (2003)
34. Chee, C.Y., Xu, D.: Chaotic encryption using dicrete-time synchronous chaos. Physics Letters A 348(3-6), 284–292 (2006)
35. Chen, G., Ueta, T.: Yet another chaotic attactor. International Journal of Bifurcation and Chaos 9(7), 1465–1466 (1999)

36. Cornfeld, I.P., Fomin, S.V., Sinai, Y.G.: Ergodic theory. Springer, New York (1982)
37. Cuomo, K., Oppenheim, A.V., Strogatz, S.: Synchronization of Lorenz-based chaotic circuits with applications to communications. IEEE Transactions on Circuits and Systems–II: Analog and Digital Signal Processing 40(10), 626–633 (1993)
38. Dedieu, H., Kennedy, M., Hasler, M.: Chaos shift keying: modulation and demodulation of a chaotic carrier using self-synchronizing Chua's circuits. IEEE Transactions on Circuits and Systems–II: Analog and Digital Signal Processing 40, 634–641 (1993)
39. Dedieu, H., Ogorzalek, M.J.: Identifiability and identification of chaotic systems based on adaptive synchronization. IEEE Transactions on Circuits and Systems–I: Fundamental Theory and Applications 44(10), 948–962 (1997)
40. Feki, M.: An adaptive chaos synchronization scheme applied to secure communication. Chaos, Solitons & Fractals 18(1), 141–148 (2003)
41. Feldmann, U., Hasler, M., Schwarz, W.: Communication by chaotic signals: the inverse system approach. International Journal of Circuit Theory and Applications 24, 551–579 (1996)
42. Fradkov, A.L., Markov, A.Y.: Adaptive synchronization of chaotic systems based on speed gradient method and passification. IEEE Transactions on Circuits and Systems–I: Fundamental Theory and Applications 44, 905–912 (1997)
43. Fridrich, J.: Symmetric ciphers based on two-dimensional chaotic maps. International Journal of Bifurcation and Chaos 8, 1259–1284 (1998)
44. Gao, T., Chen, Z.: Image encryption based on a new total shuffling algorithm. Chaos, Solitons & Fractals 38(1), 213–220 (2008)
45. González-Miranda, J.: Synchronization and control of chaos. Imperial College Press, London (2004)
46. Habutsu, T., Nishio, Y., Sasase, I., Mori, S.: A secret key cryptosystem by iterating a chaotic map. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 127–140. Springer, Heidelberg (1991)
47. Hasler, M.: Synchronization of chaotic systems and transmission of information. International Journal of Bifurcation and Chaos 8(4), 647–659 (1998)
48. Higham, N.J.: Accuracy and Stability of Numerical Algorithms, 2nd edn. SIAM, Philadelphia (1961)
49. Hirsch, M.W., Smale, S.: Differential equations, dynamical systems, and linear algebra. Academic Press, Inc., San Diego (1974)
50. Hu, G., Feng, Z., Meng, R.: Chosen ciphertext attack on chaos communication based on chaos synchronization. IEEE Transactions on Circuits and Systems–I: Fundamental Theory and Applications 50(2), 275–279 (2003)
51. Huijberts, H., Nijmeijer, H., Willems, R.: System identification in communication with chaotic systems. IEEE Transactions on Circuits and Systems–I: Fundamental Theory and Applications 47, 800–808 (2000)
52. Inoue, E., Ushio, T.: Chaos communication using unknown input observers. Electronics and Communications in Japan Part III: Fundamental Electronic Science 84(12), 21–27 (2001)
53. Jakimoski, G., Kocarev, L.: Chaos and cryptography: Block encryption ciphers based on chaotic maps. IEEE Transactions on Circuits and Systems–I: Fundamental Theory and Applications 48(2), 163–169 (2001)

54. Jiang, Z.P.: A note on chaotic secure communication systems. IEEE Transactions on Circuits and Systems–I: Fundamental Theory and Applications 49(1), 92–96 (2002)
55. Kocarev, L.: Chaos-based cryptography: A brief overview. IEEE Circuits and Systems Magazine 1(2), 6–21 (2001)
56. Kocarev, L., Halle, K.S., Eckert, K., Chua, L.O., Parlitz, U.: Transimission of digital signals by chaotic synchronization. International Journal of Bifurcation and Chaos 2(4), 973–977 (1992)
57. Kocarev, L., Makraduli, J., Amato, P.: Public-key encryption based on Chebyshev polynomials. Circuits, Systems, and Signal Processing 24, 497–517 (2005)
58. Kocarev, L., Sterjev, M., Fekete, A., Vattay, G.: Public-key encryption with chaos. Chaos: An Interdisciplinary Journal of Nonlinear Science 14(4), 1078–1082 (2004)
59. Kocarev, L., Szczepanski, J., Amigo, J., Tomovski, I.: Discrete chaos–I: Theory. IEEE Transactions on Circuits and Systems–I: Regular Papers 53(6), 1300–1309 (2006)
60. Kocher, P.C.: Timing attacks on implementations of diffie-hellman, RSA, DSS, and other systems. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 104–113. Springer, Heidelberg (1996)
61. Kolumban, G., Kennedy, M., Chua, L.O.: The role of synchronization in digital communications using chaos - Part II: Chaotic modulation and chaotic synchronization. IEEE Transactions on Circuits and Systems–I: Fundamental Theory and Applications 45(11), 1129–1140 (1998)
62. Letellier, C., Gouesbet, G.: Topological characterization of reconstructed attractors modding out symmetries. Journal de Physique II 6(11), 1615–1638 (1996)
63. Li, C., Li, S., Alvarez, G., Chen, G., Lo, K.-T.: Cryptanalysis of two chaotic encryption schemes based on circular bit shift and XOR operations. Physics Letters A 369, 23–30 (2007)
64. Li, S.: Analyses and new designs of digital chaotic ciphers. Ph.D. thesis, School of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an, China (2003), http://www.hooklee.com/pub.html
65. Li, S., Alvarez, G., Chen, G.: Breaking a chaos-based secure communication scheme designed by an improved modulation method. Chaos, Solitons & Fractals 25(1), 109–120 (2005)
66. Li, S., Alvarez, G., Chen, G.: Return-map cryptanalysis revisited. International Journal of Bifurcation and Chaos 16(5), 1557–1568 (2006)
67. Li, S., Alvarez, G., Li, Z., Halang, W.: Analog chaos-based secure communications and cryptanalysis: a brief survey. In: Kurths, J., Fradkov, A., Chen, G. (eds.) 3rd Int. IEEE Scientific Conference on Physics and Control (PhysCon 2007), Potsdam, Germany, p. 92 (2007), Full edition available at http://www.hooklee.com/Papers/PhysCon2007.pdf
68. Li, S., Chen, G., Mou, X.: On the dynamical degradation of digital piecewise linear chaotic maps. International Journal of Bifurcation and Chaos 15(10), 3119–3151 (2005)
69. Li, S., Li, C., Chen, G., Bourbakis, N.G., Lo, K.T.: A general quantitative cryptanalysis of permutation-only multimedia ciphers against plaintext attacks. Signal Processing: Image Communication 23(3), 212–223 (2008)

70. Li, S., Mou, X., Cai, Y.: Pseudo-random bit generator based on couple chaotic systems and its applications in stream-ciphers cryptography. In: Pandu Rangan, C., Ding, C. (eds.) INDOCRYPT 2001. LNCS, vol. 2247, pp. 316–329. Springer, Heidelberg (2001)

71. Li, T.Y., Yorke, J.A.: Period three implies chaos. The American Mathematical Monthly 82, 985–992 (1975)

72. Lian, K.Y., Liu, P.: Synchronization with message embedded for generalized Lorenz chaotic circuits and its error analysis. IEEE Transactions on Circuits and Systems–I: Fundamental Theory and Applications 47(9), 1418–1424 (2000)

73. Ling, B.W.-K., Ho, C.Y.-F., Tam, P.K.-S.: Chaotic filter bank for computer cryptography. Chaos, Solitons & Fractals 34, 817–824 (2007)

74. Liu, L., Wu, X., Hu, H.: Estimating system parameters of Chua's circuit from synchronizing signal. Physics Letters A 324(1), 36–41 (2004)

75. Lorenz, E.: Deterministic nonperiodic flow. Journal of the Atmospheric Sciences 20, 130–141 (1963)

76. Manjunath, G., Fournier-Prunaret, D.: A qualitative analysis of deciphering errors in chaos shift keying. International Journal of Bifurcation and Chaos 19(6), 2085–2092 (2009)

77. Masuda, N., Jakimoski, G., Aihara, K., Kocarev, L.: Chaotic block ciphers: from theory to practical algorithms. IEEE Transactions on Circuits and Systems–I: Regular Papers 53(6), 1341–1352 (2006)

78. Matthews, R.: On the derivation of a chaotic encryption algorithm. Cryptologia 13, 29–42 (1989)

79. Maze, G.: Algebraic methods for constructing one-way trapdoor functions. Ph.D. thesis, University of Notre Dame (2003)

80. Menezes, A., van Oorschot, P., Vanstone, S.: Handbook of Applied Cryptography. CRC Press, Boca Raton (1997)

81. Millérioux, G., Amigó, J.M., Daafouz, J.: A connection between chaotic and conventional cryptography. IEEE Transactions on Circuits and Systems–I: Regular Papers 55(6), 1695–1703 (2008)

82. Millerioux, G., Daafouz, J.: Unknown input observers for message-embedded chaos synchronization of discrete-time systems. International Journal of Bifurcation and Chaos 14(4), 1357–1368 (2004)

83. Millerioux, G., Mira, C.: Coding scheme based on chaos synchronization from noninvertible maps. International Journal of Bifurcation and Chaos 8, 2019–2029 (1998)

84. NIST: A statistical test suite for random and pseudorandom number generators for cryptographic applications. NIST Special Publication 800-22 Revision 1A (2010), http://csrc.nist.gov/rng/rng2.html

85. Orúe, A., Alvarez, G., Pastor, G., Romera, M., Montoya, F., Li, S.: A new parameter determination method for some double-scroll chaotic systems and its applications to chaotic cryptanalysis. Communications in Nonlinear Science and Numerical Simulations 15(11), 3471–3483 (2010)

86. Orúe, A., Fernandez, V., Alvarez, G., Pastor, G., Romera, M., Montoya, F.: Determination of the parameters for a Lorenz system and application to break the security of two-channel chaotic cryptosystems. Physics Letters A 372(34), 5588–5592 (2008)

87. Pareek, N.K., Patidar, V., Sud, K.K.: Discrete chaotic cryptography using external key. Physics Letters A 309, 75–82 (2003)

88. Parker, A., Short, K.M.: Reconstructing the keystream form a chaotic encrypiton scheme. IEEE Transactions on Circuits and Systems–I: Fundamental Theory and Applications 48(5), 624–630 (2001)
89. Parlitz, U., Chua, L.O., Kocarev, L., Halle, K.S., Shang, A.: Transmission of digital signals by chaotic synchronization. International Journal of Bifurcation and Chaos 2(4), 973–977 (1992)
90. Pastor, G., Romera, M., Montoya, F.: A revision of the Lyapunov exponent in 1D quadratic maps. Physica D 107, 17–22 (1997)
91. Pecora, L.M., Carroll, T.L.: Synchronization in chaotic systems. Physical Review Letters 64(8), 821–824 (1990)
92. Pérez, G., Cerdeira, H.A.: Extracting messages masked by chaos. Physical Review Letters 74(11), 1970–1973 (1995)
93. Pisarchik, A.N., Flores-Carmona, N.J., Carpio-Valadez, M.: Encryption and decryption of images with chaotic map lattices. Chaos: An Interdisciplinary Journal of Nonlinear Science 16(3) (2006), Art. no. 033118
94. Rajendra, U., Bhat, S., Kumar, S., Min, L.: Transimission and storage of medical images with patient information. Comput. Biol. Med. 33, 303–310 (2003)
95. Rhouma, R., Solak, E., Arroyo, D., Li, S., Alvarez, G., Belghith, S.: Comment on "modified Baptista type chaotic cryptosystem via matrix secret key". Phys. Lett. A 372, 5427 (2008); Physics Letters A 373(37), 3398–3400 (2009)
96. Shannon, C.: Communication theory of secrecy systems. Bell System Technical Journal 28, 656–715 (1949)
97. Skrobek, A.: Approximation of a chaotic orbit as a cryptanalytical method on Baptista's cipher. Physics Letters A 372(6), 849–859 (2008)
98. Solak, E., Çokal, C., Yildiz, O.T., Biyikoğlu, T.: Cryptanalysis of Fridrich's chaotic image encryption. International Journal of Bifurcation and Chaos 20(5), 1405–1413 (2010)
99. Stamp, M., Low, R.M.: Applied cryptanalysis: breaking ciphers in the real world. John Wiley & Sons, Inc., Hoboken (2007)
100. Stinson, D.R.: Cryptography: Theory and Practice. CRC Press, Boca Raton (1995)
101. Storm, C., Freeman, W.J.: Detection and classification of nonlinear dynamic switching events. Physical Review E 58, 1159–1162 (2002)
102. Szczepanski, J., Amigó, J., Michalek, T., Kocarev, L.: Cryptographically secure substitutions based on the approximation of mixing maps. IEEE Transactions on Circuits and Systems–I: Regular Papers 52, 443–453 (2005)
103. Tao, C., Du, G.: A new approach to breaking down chaotic secure communication. International Journal of Bifurcation and Chaos 13(9), 2689–2698 (2003)
104. Tao, C., Du, G., Zhang, Y.: Decoding digital information from the cascaded heterogeneous chaotic systems. International Journal of Bifurcation and Chaos 13(6), 1599–1608 (2003)
105. Vaidya, P.G., Angadi, S.: Decoding chaotic cryptography without access to the superkey. Chaos, Solitons & Fractals 17(2-3), 379–386 (2003)
106. Wang, X., Duan, C., Gu, N.: A new chaotic cryptography based on ergodicity. International Journal of Modern Physics B 22(7), 901–908 (2008)
107. Wang, X., Zhan, M., Lai, C.H., Hu, G.: Error function attack of chaos synchronization based encrypiton schemes. Chaos: An Interdisciplinary Journal of Nonlinear Science 14(1), 128–137 (2004)

108. Wu, C.W.: Synchronization in coupled chaotic circuits and systems. World Scientific, New Jersey (2002)
109. Wu, C.W., Chua, L.O.: A simple way to synchronize chaotic systems with applications to secure communications systems. International Journal of Bifurcation and Chaos 3(6), 1619–1627 (1993)
110. Xiao, D., Liao, X., Wong, K.W.: An efficient entire chaos-based scheme for deniable authentication. Chaos, Solitons & Fractals 23(4), 1327–1331 (2005)
111. Yang, T.: Recovery of digital signals from chaotic switching. International Journal of Circuit Theory and Applications 23(6), 611–615 (1995)
112. Yang, T.: A survey of chaotic secure communication systems. Intenational Journal of Computational Cognition 2(2), 81–130 (2004)
113. Yang, T., Wu, C.W., Chua, L.O.: Cryptography based on chaotic systems. IEEE Transactions on Circuits and Systems–I: Fundamental Theory and Applications 44(5), 469–472 (1997)
114. Yang, T., Yang, L.–B., Yang, C.-M.: Breaking chaotic secure communications using spectrogram. Physics Letters A 247(1-2), 105–111 (1998)
115. Yang, T., Yang, L.-B., Yang, C.-M.: Breaking chaotic switching using generalized synchronization. IEEE Transactions on Circuits and Systems–I: Fundamental Theory and Applications 45(10), 1062–1067 (1998)
116. Zhang, Y., Tao, C., Jiang, J.J.: Theoretical and experimental studies of parameter estimation based on chaos feedback synchronization. Chaos: An Interdisciplinary Journal of Nonlinear Science 16(4) (2006), Art. no. 043122

Chapter 9
Hardware Implementation of Chaos Based Cipher: Design of Embedded Systems for Security Applications

Camel Tanougast

Laboratory of Sensors and Microelectronics,
Paul Verlaine University of Metz
7 rue Marconi, 57070 Metz Technopôle - France

Abstract. In the information and communication security fields, system designers are faced with many challenges in both the trade-off *cost/performance/power* and architecture design. This is especially true for embedded system designs, often operating in nonsecure environments, while at the same time being constrained by such factor as computational capacity, memory size, and in particular power consumption. One challenge is the design of hardware architecture in order to obtain the appropriate security and the best tradeoff between hardware resources and the best throughputs rate for embedded applications. This chapter broadly outlines a disciplined approach to design and implementation *3D* chaotic systems as *Lorenz, Lü, Colpitts*, *Chen* systems and so in embedded applications. The approach combines the numerical resolution method paradigm of *3D* differential equations characterizing some chaotic systems with the design hardware architecture paradigm. The model of *Runge-Kutta*'s numerical method to resolve *3D* chaotic system requirements used as key generator for data encryption applications is detailed. This chapter describes this approach and presents a case study where the *Lorenz's* chaotic system is implemented on a *FPGA* Chip.

1 Introduction

Since the discovery of deterministic chaos, the field of non-linear dynamical systems has found applications in areas as diverse as biophysics, meteorology, hydrodynamics, chemical engineering, optics, cryptology and communications cryptosystem, especially since the first description of chaos synchronization by *Pecora* and *Carroll* [1]. A large number of chaos-based secure communication schemes have been proposed in the last decades [2, 3]. Typically, chaotic systems are characterized by ergodicity, sensitive dependence on initial conditions and random-like behaviors, properties which seem pretty much the same as required by cryptographic primitive characteristics such as "diffusion" and "confusion".

Among the systems showing a chaotic behavior, we find *3D* continuous-time chaotic systems as systems of *Lorenz, Lü, Colpitts*, *Chen*, *Rössler*, *Sprott*, and so on. These chaotic structures seem to be abundant and complex dynamical behaviors which can be used for designing of chaotic hardware key generation for secure communication systems.

A digital implementation of chaotic generators presents some advantages and provides accuracy and large possibility of integration in embedded applications especially for data encryption and secures communications between embedded systems. Unlike, in analogue implementations which exhibit some practical difficulties to ensure information recovery and to deal with the problem of the chaotic synchronization since the component values vary with age, temperature, etc., a digital implementation avoids the parameter mismatch between the transmitter and the receiver. Actually, programmable hardware fabric like *FPGA* (*Field Programmable Logic Array*) is taking an increasingly significant place in the embedded digital system design paradigm, this is due to the excellent trade-off between computing power and flexibility of processing it provides.

This chapter broadly outlines a disciplined approach to design and implementation *3D* chaotic systems as *Colpitts*, *Chen*, *Lorenz*, *Lü* systems and so in embedded applications. The design approach combines the numerical resolution method paradigm of *3D* differential equations characterizing some chaotic systems with the design hardware architecture paradigm. The model of *Runge-Kutta*'s numerical method allows resolving *3D* chaotic system requirements used as key generator for data encryption applications. This chapter describes this approach and presents a case study where the *Lorenz's* chaotic system is implemented on a *FPGA* Chip.

The rest of the chapter is organized as follows: In Section 2 the related work is described. Section 3 gives an overview, a description and characterization of *3D* chaotic systems used for embedded encryption applications. Section 4 presents an overview of the analog hardware implementations and discuss of the synchronization problems. Section 5 discuss of the architecture exploration for the chaos behavioral digital hardware implementation. Section 6 gives the background of the digital design based on a numerical resolution method of the *3D* chaotic systems as well as the *FPGA* technology design. In this section, we discuss in detail the various step involved in the design of a chaotic system, and illustrate it with the *Lorenz's* chaotic system designing in *FPGA*. The performances and real time measurements are detailed and the chaotic synchronization of two chaotic generators using a *Feed-Back Synchronization* approach is also presented. The feasibility and the efficiency of the secure image encryption application based on the synchronized embedded chaotic generators are also given in the Section 7. The security of the encryption scheme is evaluated through both cryptanalysis and experiments. Finally, Section 8 summaries and concludes the chapter.

2 Related Works

In 1963, *Lorenz* found the first chaotic strange attractor in a continuous nonlinear three-dimensional autonomous system, which is classed as a double-scroll attractor

[4]. This discovery became the start point of extensive studies focused on chaos phenomena. Indeed, it has been found that a chaotic system exhibits a richness chaotic behavior according to the values of its bifurcation parameters. Therefore, the electrical engineering community realized that chaos could be useful in secure communication systems because chaos is extremely sensitive to the initial conditions and parameters [5, 6]. The chaotic behavior is described and characterized either by one, double, three, four- scrolls or more scrolls strange attractor. For example, the *Rössler's* system, *Linz's* and *Sprott's* systems exhibit one-scroll attractor [7, 8] while *Chua's* system exhibits one or more scroll-attractor [9]. Thus, chaos-based encryption has suggested a new and efficient way to deal with the problem of fast and highly secure data encryption [10, 11]. Indeed, these systems are very sensitive to small variations of their initial conditions and parameters and these systems can be synchronized between their and used for secure communications [12, 13].

To design chaotic generators, the main approach uses the analogue technique. During the last decades, many methods based on analogue circuits are used to implement chaotic generators. Generally, these generators are implemented using analogue nonlinear circuits (diodes, op-amps, transistor, etc.) associated with analogue components (resistor, inductance, capacitor, etc.). For instance, the Figure 1 presents analog circuit model of the *Lorenz's* chaotic system based on *op-amps* employed to provide look backing active integrator modeling the *Lorenz's* differential equation [13, 14].
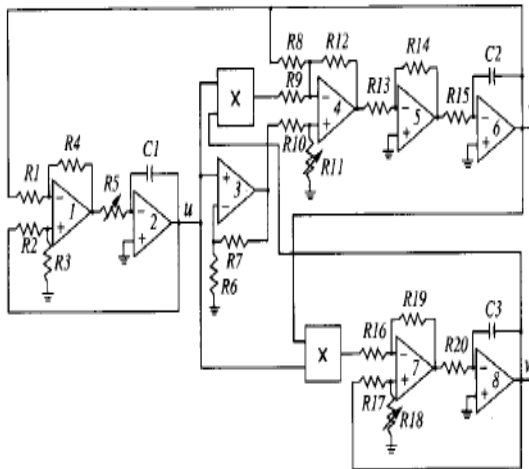
Fig. 1 Analog circuit model of the *Lorenz's* chaotic system [14]

Others methods based on analogue circuits are used to implement chaotic generators such as switched capacitor or analogue CMOS technology [14-17]. However, the physical parameters of these analogue devices need to be constant in order to ensure a permanent synchronization between both emitter and receiver. Indeed, these methods exhibit some practical difficulties since the component

values vary with age, temperature, etc. Therefore, one must implement both the transmitter and the receiver with very high accurate components to ensure information recovery, since the recovery characteristics are very sensitive to parameter mismatch between the transmitter and the receiver. In addition, it is very difficult to deal with the problem of the chaotic synchronization. Hence, analogue implementation is very difficult thought it is possible to overcome this difficulties to some extent. To overcome this problem, a second method based a digital hardware implementation of chaotic generators can be used since the problem of parameter mismatch between the transmitter and the receiver does not exist. Nevertheless, the difficulty of the information recovery depends only on the channel noise sensibility. In this case, the chaotic synchronization becomes less complex than the case of an analogue implementation counterpart. Moreover, with an analogue implementation, the circuitry responsible for chaos generation (at least the explicit form of the non linear differential equation system) should be given with specific details; while for a digital implementation, the following details should be provided: the finite computing precision, the adopted digital arithmetic (fixed-point or floating-point), the hardware/software configuration.

3 Chaotic Generators Based Encryption

Data security is a significant subject for which various encryption algorithmic solutions have been proposed [18]. Encryption methods using chaotic dynamics are used. In the case of chaotic encryption techniques, a sequence, which is generated by a discrete or continuous chaotic system, is used as a key for ciphering a plain text [19,20]. An overview of the secure communication approach consists to implement a cipher key generator based on a tri-dimensional data chaotic generator used for encryption a plaintext. This leads to the problem of synchronization between encryption and decryption processes. Figure 2 illustrates this secure communication approach. For embedded secure applications, the challenge based on this approach is to design synchronized key generators based on differential equation system characterized by a chaotic regime.
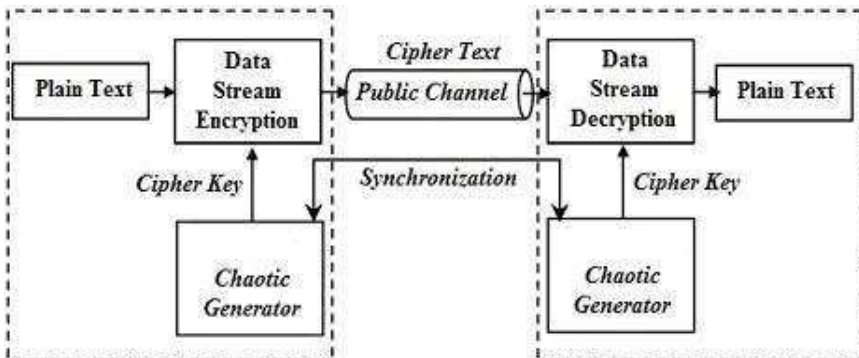
Fig. 2 Illustration of the embedded chaotic cryptosystems for communications

3.1   Tri-dimensional Chaotic Systems: Chaos Behavioral Modeling and Software Simulation

Currently, we distinguish two types of chaotic systems. First one we find the discrete chaotic systems such as the *Henon* and Logistic map, modeling by the equations 1 and 2, respectively.

$$X_{n+1} = 1 + Y_n - a.X_n^{\,2} \tag{1.a}$$

$$Y_{n+1} = b.X_n \tag{1.b}$$

$$X_{n+1} = A.X_n(1 - X_n) \tag{2}$$

And others hand, we have the continuous chaotic systems characterized by a system of differential equation systems [21, 22]. For example, we can cited the *Jerk's* dynamic system defined by the following third order differential equation:

$$\dddot{x} = -a\ddot{x} - \dot{x} + G(x) \tag{3}$$

where *a* is parameter of the system. $G(x)$ is a nonlinear function. Depending on the expression of G(x), there is a family of the *Jerk's* dynamic system. For example, consider us the *Jerk's* chaotic system defined by the nonlinear function $G(x) = bx^2 - 1$. The nonlinearity is expressed by the term x^2. Let us rewrite equation (3) in a equivalent form of a three dimensional system:

$$\begin{cases} \dfrac{dx}{dt} = y \\[2mm] \dfrac{dy}{dt} = z \\[2mm] \dfrac{dz}{dt} = -az - y + bx^2 - 1 \end{cases} \tag{4}$$

The solution of this nonlinear equation system depends mainly on the initial conditions specified by the initial values of x= x_0, y = y_0 and z = z_0. A numerical solution of this system using *Matlab* simulation tool [23] with parameters values a = 0.57, b = 0.56 and initial conditions ($x_0 = y_0 = z_0 = 0$), gives the corresponding chaotic signals x, y and z and the *3D* attractor of the chaotic system shown in Figure 3. The *Jerk's 3D* attractor is characterized by one spiral scroll.
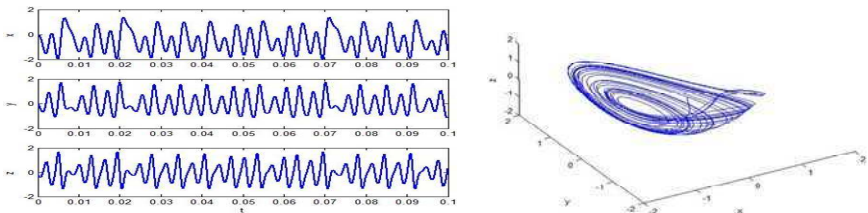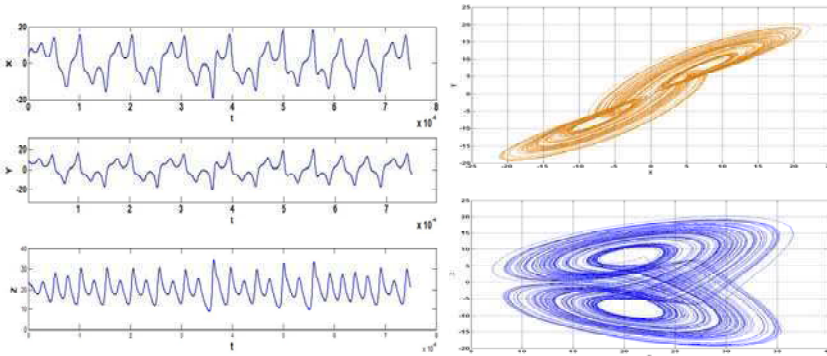

Fig. 3 *Jerk's* Chaotic Signals and *3D* attractor

Most of the continuous chaotic systems can be expressed by an equivalent form of a three dimensional system. Among then we can cited the tri-dimensional (*3D*) chaotic system such as *Lorenz's, Cheng's, Lü's, Colpitts, Chua's, Rössler's, Linz* and/or *Sprott's* systems. These *3D* systems provide chaotic behaviours depending on the initial condition and parameters values characterizing these systems. For example, the *Lü's* system knowing as a bridge between the *Lorenz's* and *Chen's* systems [21, 22, 24, 25], and represented by the simplified nonlinear equation system (5), can presented a chaotic behavior.

$$\begin{cases} \dfrac{dx}{dt} = a(y - x) \\ \dfrac{dy}{dt} = -xz + cy \\ \dfrac{dz}{dt} = xy - bz \end{cases} \tag{5}$$

A numerical solution of this system with *Lü's* parameters values a = 36, b = 3 and c = 20 and initial conditions ($x_0 = 0$, $y_0 = 5$, $z_0 = 25$), gives the corresponding chaotic signals x, y and z and the two different attractors of the chaotic system shown in Figure 4.

Fig. 4 Simulation results of *Lü's* chaotic signals and attractors in phase planes (x-y), (y-z).

4   Analog Hardware Implementation

Over the last several decades, many electronic circuits based passive or active analogue components exhibiting chaos have been proposed [26]. These circuits are considered as simplest nonautonomous (driven) or autonomous circuits constructed using the most common components. Thus, the *Linsay*'s driven inductor-varactor resonator is one of the first simple chaotic circuit [27]. One of the most recent claims to simplest autonomous chaos is *Piper and Sprott*'s *op-amps integration* or *gain block* based chaotic resonators [28]. These realizations implement

the *Jerk's* chaotic system (also see Section 3.1) defined by the nonlinear function $G(x) = b(sgn(x) - x)$ and characterized by the equation (6) where the signum function *sgn(x)* can be approximated by using an *op-amp* without negative feedback.

$$\dddot{x} = -a\,\ddot{x} - \dot{x} - b\,(\text{sgn}(x) - x) \tag{6}$$

The circuits (see Figure 5) use op-amps employ as comparators to provide signum nonlinearity and linear time-invariant passive components. The first (designated *CCC*, Figure 5.a) uses four resistors, three capacitors and two op-amps. In this circuit, R_c-R-C_1-U_{1a} forms the active integrator, while R_1-C_2-R_2-C_3 forms the passive second-order integrator. The second (designated *CLC*, Figure 5.b) uses two resistors, two capacitors, one inductor and two op-amps. Here, L-C_2 forms the second order passive integrator.
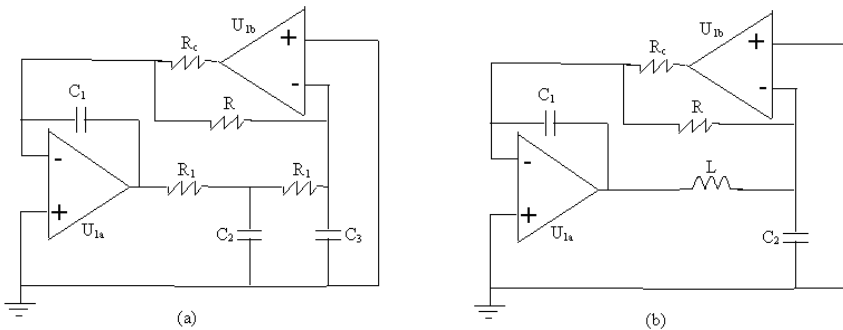
Fig. 5 *Piper and Sprott*'s autonomous chaotic circuits

4.1 Application to COLPITTS System

In this section, we consider the *Colpitts* system claim as a simple nonautonomous chaos based on single-transistor chaotic resonator. A *Colpitts* oscillator [17,29-31], is one of a number of electronic oscillator circuits which exhibits a rich dynamical behavior like many other third-order oscillator configurations available in the literature [29, 31]. In particular, for the *Colpitts* oscillator there is extensive numerical and experimental evidence of continuous chaotic behavior [31] that can be used in secure communication systems. Indeed, this system is non-symmetric and is therefore generic, and possesses an intrinsic nonlinearity given by the exponential characteristic of the active device [29, 30].

Generally, an oscillator is composed of an amplifier based transistor *FET* (*Field Effect Transistor*) or *BJT* (*Bipolar Junction Transistor*) associated with a resonant network based on self capacity (*LC*) mounting which is often used to operate at the limit of frequencies at some KHz to a few hundreds MHz. For this purpose, the *Colpitts* oscillator is the most widely used *LC* based oscillator. One of the key features of this type of oscillator is its simplicity and robustness since it is not difficult

to obtain satisfactory results with little effort. A *Colpitts* oscillator can be seen as an electrical dual of a *Hartley* oscillator. Thus, the *Colpitts* circuit comprises:

- An oscillating circuit (*LC*).
- A capacitive tension divider.
- A transistor amplifier.

The principle circuit of the *Colpitts* oscillator is shown in Figure 6. The amplifier consists of a bipolar transistor (*BJT*) assembled in a common-base configuration. The circuit resonator consists of one inductance (*L*) and two capacitances (*C1* and *C2*) which are connected between the collector and the base of the transistor. This is used to determine the frequency of oscillation. A fraction of the voltage of the circuit (*LC*) is turned over to the transmitter. This return is obtained by a capacitive voltage divider. The circuit bias is provided by the voltage *Vcc* and the current source I_0, the latter being characterized by a Norton-equivalent conductance G_0 [29]. The *Colpitts* oscillator is commonly designed to generate periodic oscillations. However with specific settings of the circuit parameters and appropriate circuitry modifications the bipolar junction transistor based oscillator delivers chaotic waveforms.

(a)   (b)
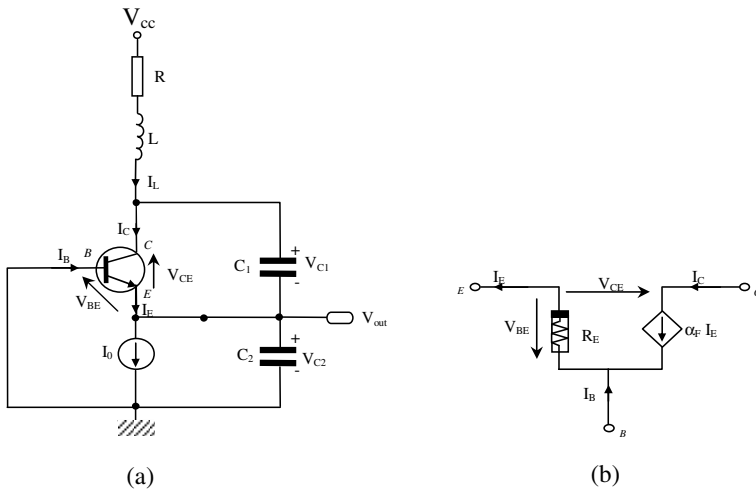
Fig. 6 Circuit model of the *Colpitts* oscillator: (a) circuit schematics; (b) transistor in a *BC* configuration.

The simulation of the *Colpitts* system only has bifurcations and chaotic phenomena but also diversified stability phenomena, such as multiple periods and stagnant dots. More precisely, by varying the value of the resistance *R*; we obtain a periodic solution. Thereafter, we passed by a periodic solution of p-finite order. Finally, the chaotic behavior is generated. Figure 7 presents the circuit simulation results showing a *Colpitts* chaotic behavior. It gives the output chaotic signal *Vcout* for a particular value of *R* (denoted as *x*) and one attractor in the phase plane (*x* - *z*),

where *z* corresponds to *Vcout* for a another value of *R*. The structure of this type of chaotic system is quite complex and depends on the variables and parameters of the system. These various stages usually lead to chaos by period-doubling in the *Colpitts* oscillator. From these simulation results and for certain parameter values (passive components), the *Colpitts* system can exhibit different steady-state behaviors depending on the initial conditions of the system. This is a typical situation in nonlinear dynamical systems where, in general, different stable solutions, equilibrium points, limit cycles, chaotic attractors, and so on may coexist.
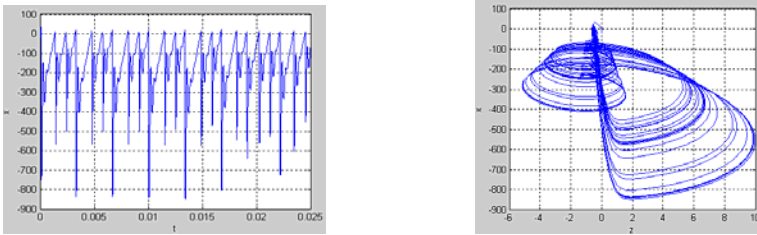
Fig. 7 *Colpitts* chaotic signal and one chaotic trajectory (attractor of chaotic phenomena).

4.2 Synchronization Problems

The synchronization becomes vital for accurate recovery of the transmitted signal in encrypted communications based on chaotic generators. Some researchers in the area have been then interested to investigate chaos synchronization in various fields including secure communications. Many methods and techniques for synchronizing chaos have been proposed in literature based on nonlinear observers, parameters estimation, control models [12, 32], etc. However, when implemented on analog circuit platform, it is impossible to preset the system without any errors. Indeed, the parameters of chaotic system may change with voltage and temperature (internal noise). Since chaotic systems are sensitive to parameters, most of the synchronization scheme will fail due to parameter mismatch. To overcome this problem, a numerical generation of chaos can be used since the problem of parameter mismatch does not exist. For instance, a real-time digital hardware implementation of the *Feed-Back chaotic synchronization* (*FCS*) between two chaotic signal generators allows one chaos synchronization. This approach can be used for synchronizing any three-dimensional continuous chaotic systems (*Chua's* system, *Lü's* system, *Colpitts* system, etc.) used as hardware key cipher generator for encrypted communications [33] (more details in the Section 6.9).

5 Digital Hardware Implementation

Digital implementation of chaotic generators provides accuracy and large possibility integration in embedded systems which control many of the common devices in use today, and allowing many possibilities for embedded applications especially

for data encryption and secures communications between embedded systems. In this context, advances in *VLSI* technology have been employed to the manufacturing of reconfigurable logic including *FPGA* chips and helped their rapid growth in logic capacity, performance and popularity. In this section we consider the architecture exploration for a digital hardware implementation of the *3D* chaotic generator based on numerical resolution of differential equation systems characterizing the most continuous chaotic systems.

5.1 Digital Implementation Based on Numerical Resolution of 3D Chaotic Systems

For resolve the *3D* differential equation, we can use the well-known fourth order *Runge-kutta* numerical resolution method (*RK-4*) which produces a more accurate estimate of the solution [34, 35]. Let us consider the following first order nonlinear differential equations system which can expressed one *3D* chaotic system:

$$\begin{cases} \dfrac{dx}{dt} = F(x, y, z) \\[2mm] \dfrac{dy}{dt} = G(x, y, z) \\[2mm] \dfrac{dz}{dt} = Q(x, y, z) \end{cases} \tag{7}$$

where $x(t_0) = x_0$, $y(t_0) = y_0$ et $z(t_0) = z_0$ and F, G, Q are nonlinear functions. The *RK-4* method uses several intermediate points to calculate the next value starting $(x_{n+1}, y_{n+1}, z_{n+1})$ from the initial value (x_0, y_0, z_0) and the step length h in t as it is specified by the following equations:

$$x_{n+1} = x_n + \frac{h}{6}(k_0 + 2k_1 + 2k_2 + k_3) \tag{8}$$

$$y_{n+1} = y_n + \frac{h}{6}(m_0 + 2m_1 + 2m_2 + m_3) \tag{9}$$

$$z_{n+1} = z_n + \frac{h}{6}(n_0 + 2n_1 + 2n_2 + n_3) \tag{10}$$

where at the initial t_0 instant:

$$k_0 = F(t_n, x_n) \tag{11}$$

$$m_0 = G(t_n, y_n) \tag{12}$$

$$n_0 = Q(t_n, z_n) \tag{13}$$

at $t_0 + h/2$ instant:

$$k_1 = F(t_n + \frac{h}{2}, x_n + \frac{h}{2}k_0) \tag{14}$$

$$m_1 = G(t_n + \frac{h}{2}, y_n + \frac{h}{2}m_0) \tag{15}$$

$$n_1 = Q(t_n + \frac{h}{2}, z_n + \frac{h}{2}n_0) \tag{16}$$

$$k_2 = F(t_n + \frac{h}{2}, x_n + \frac{h}{2}k_1) \tag{17}$$

$$m_2 = G(t_n + \frac{h}{2}, y_n + \frac{h}{2}m_1) \tag{18}$$

$$n_2 = Q(t_n + \frac{h}{2}, z_n + \frac{h}{2}n_1) \tag{19}$$

and at $t_0 + h$ instant:

$$k_3 = F(t_n + h, x_n + hk_2) \tag{20}$$

$$m_3 = G(t_n + h, y_n + hm_2) \tag{21}$$

$$n_3 = Q(t_n + h, z_n + hn_2) \tag{22}$$

5.2 Architecture Exploration

Security is a primary requirement of any communication application. In modern security problems, the need for secure cryptographic algorithms that are hardware implemental is increasingly important issue while allowing to satisfy the following two metrics in digital system design: the throughput rate and the area or the amount of hardware resources required to match this throughput rate. Efficient hardware design is essentially a resource allocation problem. The goal is, given the constraints, to find the optimal balance between required silicon area, operation throughput, energy consumption and design time to implement a system. The objective of an *architecture exploration* is to find an efficient matching between an algorithm and architecture. The aim is to realize an optimal implementation that satisfies the constraints (real time, logic area etc.).

The intrinsic parallelism of the cryptic algorithms is still well adapted to a hardware implementation. Digital hardware techniques can be used to implement efficiency chaotic generators by using digital devices such as *microcontrollers*, *Digital Signal Processors* (*DSPs*), *Application-Specific Integrated Circuits* (*ASICs*), *digital processors* or *Field Programmable Gate Array* (*FPGA*) technologies. The choice of the implementation in a digital system is driven by many criteria and heavily dependent on the application area.Thus, a chaotic generator for cipher encryption can be implementing both software (by coding in *C, C++, SystemC,* etc.) to be executed on processors and hardware (by designing and coding in hardware

description languages such as *VHDL* and *Verilog*). Table 1 gives the main contrast features of digital implementations based on digital software or hardware technologies.

Table 1 Features of implementations in *ASICs*, *FPGA*s and *processors*.

	Processors (Soft-ware)	FPGA (Hard-ware)	ASICs (Hard-ware)
Silicon area	Fixed	variable	Fixed and low
speed	moderate	Fast	Very fast
consump-tion	moderate	high	weak
Word size	Fixed	variable	variable
Flexibility	Yes	Yes	No
Design cy-cle	short	moderate	long
Performed at the user site	yes	yes	no

ASICs are designed all the way from the behavioural description to the physical layout and then sent for a fabrication in a semiconductor foundry. The technology of digital circuits as well as the tools available for the design, the use and the implementation of the algorithms has played a significant role to achieve a high throughput, but with a high cost in terms of resources used. The high operating frequency as well as the great amount of resources available in an *ASIC*, makes it possible to implement such algorithms. Nevertheless, their high cost of design and the long time to market, pushed the designers to find others technologies. *FPGA* technology then seemed to be an attractive alternative especially since this technology allows a good performances with a great design flexibility. *FPGA* is interesting in the way that it ensures a better computing performance in comparison with a *CPU* core and allows the flexibility and a lower cost compared to *Silicon IPs* or *ASICs* (see Table 1)). The most obvious is the possibility to frequently update the digital hardware functions. But we can also use the dynamic resources allocation feature in order to instantiate each operator only for the strict required time. This permits to enhance the silicon efficiency by reducing the reconfigurable array's area. In this context, the use of *FPGA* technology makes it possible to optimise the hardware resources required while allowing a real time computing.

In the next section, we will consider the *FPGA* based low-level design of the *3D* chaotic systems allowing optimized hardware implementations in terms of hardware resources (logic area, specific compute blocks, size memory, etc.) and performances (frequency and throughput).

6 Digital Programmable Hardware Implementation Using *FPGA*s

An *FPGA* can be exploited for the secure and efficient implementation of symmetric cryptographic algorithms and protocols. These reconfigurable devices intend

to fill the gap between hardware and software, achieving potentially much higher performance than software, while maintaining a higher level of flexibility than hardware. Several behavioral structures of chaotic systems have been implemented in *FPGA* technology. Among them, we find mainly *Lorenz*, *Chua* or *Chen* systems [36, 37] which can be used for designing chaotic hardware key generation for data encryption systems. These previous digital chaos implementations are not optimized since these works have used non optimal hardware description language code generation using *Simulink/Matlab* automatic code generation tools. The Figure 8 presents one hardware implementation model based automatic generator of the *Lorenz's* chaotic system from the *Simulink/Matlab* tool associated to *Xilinx's FPGA* design tool (*Xilinx System Generator*). The main drawback of this design approach is that the *high level* aspect of this method keeps the user far away from realities of the physical implementation (*low-level* architecture). Thus, the result in terms of performance and density of resources used remains out of the designer reach.
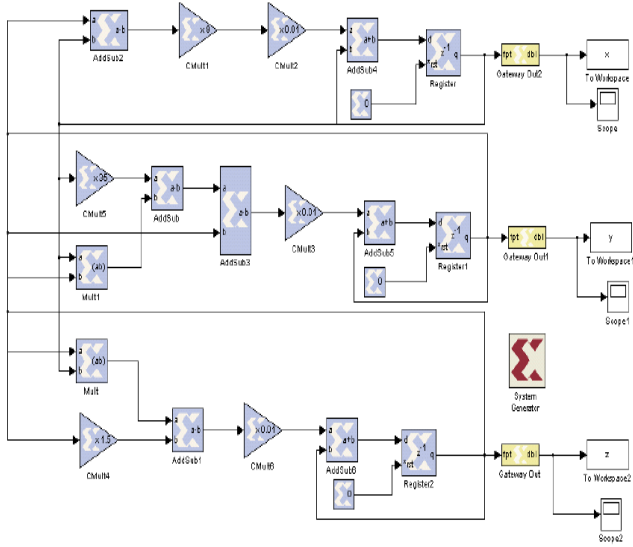
Fig. 8 Simulation model based automatic generator of the *Lorenz's* chaotic system [37].

6.1   FPGA Technology

An *Field programmable Gate Arrays*, introduced in 1985 by the company Xilinx, is a programmable device consisting:

- A set of *programmable logic cells* (called *Logic blocks or CLBs*) based on *n-inputs function generator* (usually denoted as *Look-Up Table*s (*LUTs*)) associated with registers through local select interconnects. Some *CLBs* can include complex arithmetic and logic units depending the type or family of the *FPGA* technology used (commonly referred to as the size granularity of the *Logic Cells*).

- A *programmable interconnection network* and *input and output cells* around the device (Figure 9).

The resulting structure is vendor-dependant (*Altera*, *Actel*, *Xilinx companies, etc.*). According to the arrangement of the *Logic Cells* and interconnection paradigm of the *Logic Cells* on the device, *FPGA*s can be classified in several categories such as *symmetrical array*, *hierarchy-based*, *row-based*, and so on. The Figure 9 presents an overview of a symmetrical array based *FPGA* currently used (*Xilinx's Virtex FPGA* technology). The *Logic blocks* are connected together using the *programmable interconnection* and through *Switch Matrix* which are used to connect vertical and horizontal lines, thus making routing possible on the *FPGA*. Locally around the periphery of the device, *input and output cells* (referred as *I/O components*) allow for the communication of the design inside the *FPGA* with off-chip modules. These *I/O components* can be configured as a input, output or bidirectional interface pin. All components of an *FPGA* can be programmed once or several times depending on the technology used (*SRAM*, *EPROM*, *ANTIFUSE*). Modern *FPGA*s contain embedded components such as memory blocks, multipliers and even processors cores (for example, two *PowerPC* processor cores in the *Xilinx's Virtex II Pro devices*) [38].
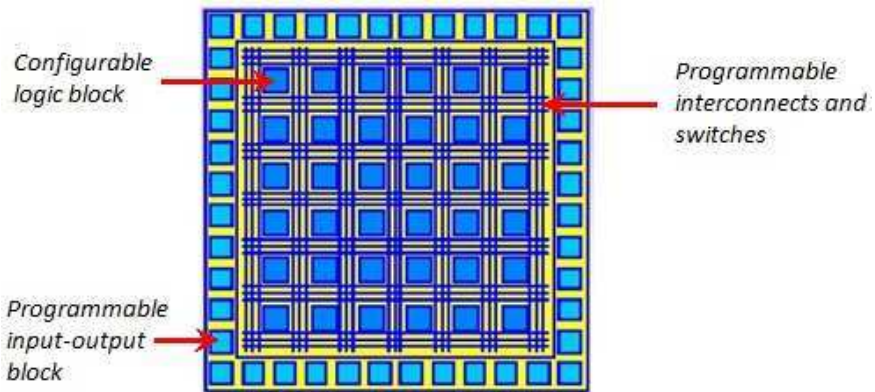
Fig. 9 General structure of an symmetrical array based *FPGA*

A digital function to be implemented in *FPGA* is partitioned in modules, each of which can be implemented in *Logic Blocks*. After the design description and functional simulation, the design can be logic synthesized at *Register transfer Level* (*RTL*). *LUT*-based technology mapping is then used to implement the functions as well as their interconnections in the *FPGA* by using the CAD tools. *LUTs* are the modules used in the *FPGA* to implement the logic and arithmetic operators of the *RTL* description. The *place* and *route* inside the *FPGA* as well as the generation of configuration data of the device is done also by the *FPGA* development tool. Consequently, *FPGA* designs can be performed by the user in the field.

In this remainder of this section, we assume a reader with basic knowledge in *FPGA* design and consider a Xilinx *Virtex-II FPGA*. Such devices embed *CLBs*,

RAM, multipliers and two *PowerPC* cores. The *CLBs* is composed of the *logic units* (denoted as *Slices*) that is generally used to evaluate an *FPGA* design's area requirements. Figure 10 depicts the structure of a *Slice*, mainly made up of two *4-inputs LUTs*, two *registers* and *feedback interconnections*. Depending on the logic synthesis, any of *Slice* can be configured as *RAM* storage, *linear shift register* or any *4-1 Boolean function* (logic or arithmetic operators). The *LUTs* with the registers in the *Slice* architecture facilitates the use of pipelining techniques to increase the clock frequency (and then throughput) of *FPGA* designs. Indeed, pipelining is an economic form of parallelism especially in *FPGAs* with *flip-flops* that are already present in each cell whether they are used or not. Where long circuit paths would preclude the use of a high-speed clock, *registers* can be placed along those paths thus decreasing the maximum distances between synchronizing *registers*. In other words, the number of levels of logic is reduced. However, the addition of such "pipeline stages" does increase the latency of the design, i.e., the number of clock cycles needed for a given set of input data to propagate through the entire design.

The hardware implementation of a stream cipher circuit using a *FPGA* follows the next steps. The hardware implementation is designed and coded in *VHSIC Hardware Description Language* (*VHDL*) with structural description logic way. With the aim of achieving high throughput rates with low hardware resource requirements. The description to implement is simulated for the correct operation with test vectors returned by the software implementation. The *VHDL* code given the structural architecture is synthesized in *Virtex*-II *Xilinx* [38] *FPGA* technology for demonstration purposes. The software tool used for these implementations is *ISE* of *Xilinx* [39].
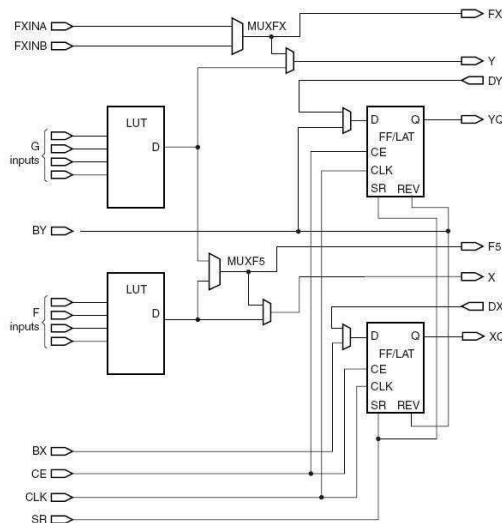
Fig. 10 *Virtex-II* Pro general *Slice*.

6.2   Case Study: Lorenz's System

In this of this sub-section, we will consider the *FPGA* design of the *Lorenz's* chaotic data generator. The system of *Lorenz* is a famous and widely used example of chaotic system. It is represented by the following nonlinear equation system [40]:

$$\frac{dx}{dt} = \sigma(y - x) \tag{23.a}$$

$$\frac{dy}{dt} = -xz + rx - y \tag{23.b}$$

$$\frac{dz}{dt} = xy - bz \tag{23.c}$$

where σ, r and b are parameters. The solution of this nonlinear equation system depends mainly on the initial conditions specified by the initial values of x= x_0, y = y_0 and z = z_0. A numerical solution of this system can be found with the fourth order *Runge-Kutta* method (*RK-4*) (as described in the Section 5.1) with the following value coefficients (h = 0.01, coef = 1/6) of the numerical resolution. By using *Matlab* simulation tool, this *3D* differential equation system has a chaotic behavior which can be used to control the generation of the cipher key (more details in the next Section 7) with *Lorenz*'s parameters values σ = 10, r = 28 and b = 8/3 and initial conditions (x0 = 0, y0 = 5, z0 = 20). Figure 11 gives the corresponding chaotic signals x, y and z and the two different attractors of the *Lorenz's* chaotic system obtained by software simulations. The second and the third parts relates to the phase plane (x-y) and the phase plane (y-z), respectively. These modeling and simulation are useful as references for an hardware implementation (see Section 6.8).
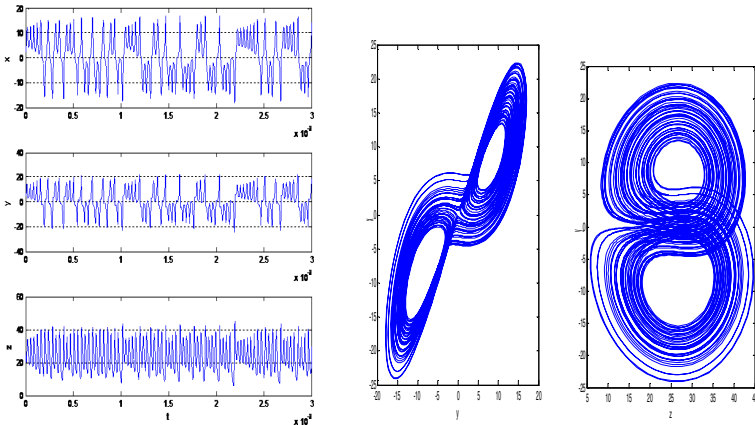
Fig. 11 *Matlab* simulation results of *Lorenz*'s Chaotic Signals and attractors in phase plane (x-y) and (y-z).

6.3   RTL Architecture

Generally, feature and available hardware resources are not always optimally exploited by the synthesis and implementation tools during an *high-level* system description. For an efficient hardware implementation, it is important to have them in mind and required during the *low-level* description. A *Register Transfer Level* (*RTL*) consists at a *low–level* hardware description required for an efficient implementation.

A *Register Transfer Level* (*RTL*) architecture of the *Lorenz's* data generator consists of the implementation of the *RK-4* method to resolve the *Lorenz's* differential equations system. An overview of the *RTL* architecture for a *Lorenz's* chaotic generator is given in Figure 12. This random key architecture is based on fixed parameters σ, r and b and consists of the structural feedback of the three main blocks: *F1, F2* and *F3*. These three functional units realize the equations (23.a), (23.b) and (24.c), respectively. These units are composed simply by an adder, a substractor and a multiplier logic arithmetic operators in accordance with the set *RK-4* solution of equation (23). The data-path processing architectures of these units are depicted in Figure 13.
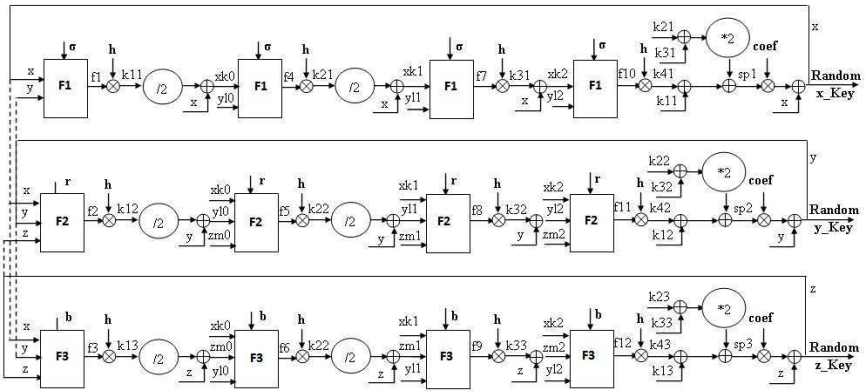
Fig. 12 *RTL* architecture of the *Lorenz's* chaotic generator.
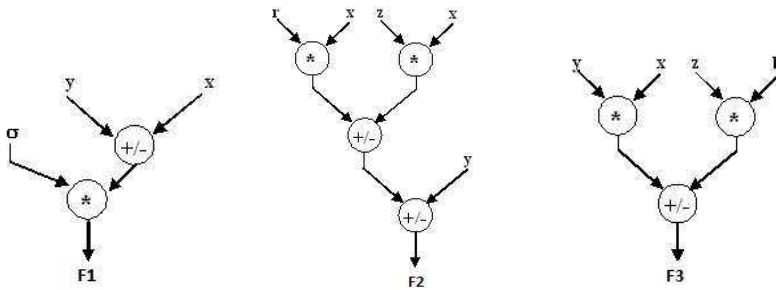
Fig. 13 *RTL* architecture of the *F1*, *F2* and *F3* functional units, respectively.

6.4 Logic Hardware Modeling and Simulation

The functional simulation of the *Lorenz's RTL* architecture could be simulated for the correct functional operation with test vectors returned by a software implementation. This validation consists to model and describe directly the *RK-4* method with the *VHDL*. It should be noted that the continuous chaotic signals are real. To get around this problem in the *VHDL* language, an implementation based on a finite solution numbers with a fixed point representation of real data on 32 bits (*16Q16*) can be adopted. i.e. all data are fixed point format with 16 bits integer and 16 bits fraction. Unlike the floating format approach, the fixed-point arithmetic allows a very useful and attractive trade off between high speed, low area cost and data transmission security.

A *RTL* description simulation with *ModelSim* tool [41] allows to test the effectiveness of the *Lorenz*'s chaotic generator architecture. Figure 14 presents the simulation results where the chaotic signals *x, y* and *z* are represented with 32 bits using the followings *RK-4* value coefficients (h = 0.01, coef = 1/6), with *Lorenz*'s parameters values $\sigma = 10$, $r = 28$ and $b = 8/3$ and ($x_0 = 0$, $y_0 = 5$, $z_0 = 20$) as the initial conditions. It can be seen that the functional hardware simulation results are very similar those of the *RK-4* numerical resolution by using *Matlab* software simulation tool (see Figure 11).
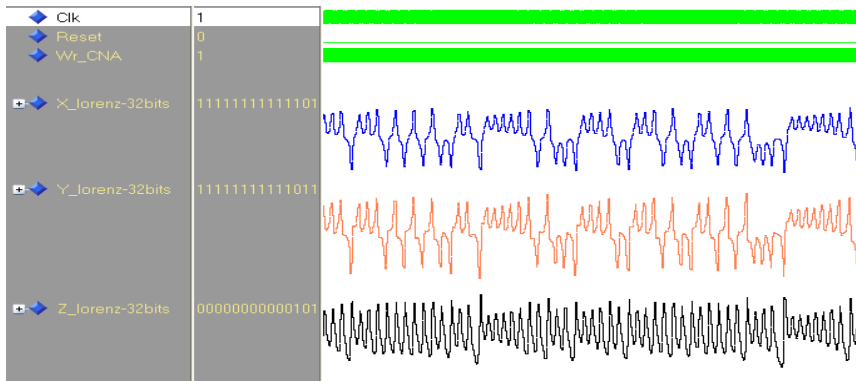
Fig. 14 *ModelSim* simulation results of *Lorenz's* chaotic generator.

6.5 Evaluation on the Effect of the Digital Error Types

To gauge the robustness and effect evaluation of the discretization errors several hardware implementation could be designed in *VHDL* with structural description logic and using a configurable *2n*-bit fixed-point format with *n* integer bit and *n* fraction bit (noted *nQn*) for real data representation of the continuous chaotic signals. Since a configurable technology solution like *FPGA* is not constrainted to a precision of a particular microprocessor (see Table 1), an evaluation on the effect

of the discretization errors allows to quantify the benefice to perform the trade-off between required resources and secure level for embedded applications.

A low precision or round-off error in the real data representation cannot preserve the dynamic of the generated chaotic signals. More precisely, the dynamic and chaos properties are not conserved by the effect of the discretization. *Autocorrelation* is the cross-correlation of a signal for finding repeating patterns, such as the presence of a periodic signal which has been buried under noise. The A*utocorrelation* test for different size fixed point data format allows to evaluate the randomness of the generated chaotic keys by the *Lorenz's* hardware architecture. In our case study, the *autocorrelation* of a random process describes the correlation between values of the process at different points in time by analyzing the series of values. The ideal result for a completely non correlated generated keys (denoting non-periodic system) is a centred single peak at the origin. Figure 15 gives the *autocorrelation* for the generated chaotic keys for two size fixed point format (*16Q16* and *8Q8*). The statistical quality of the chaotic keys *16Q16* is then considered as superior. As shown by the *autocorrelation* function for the *16Q16* and *8Q8* sizes of fixed-point format, one observe that increasing the size of the fixed-point format is decreasing over the peak behind begins to move up. Indeed, secondary peaks comes to a *8Q8* size begin to appear which implies a periodicity in the generated sequence. Therefore, the robustness increases proportionally of the *nQn* size value. This tend proves that a *16Q16* size format is sufficient for an efficiency robustness in term of generated key value while finding an optimized implementation. And other hand, for lower *nQn* format values, the robustness decreases while preserving the dynamic chaos range of the generated signals until one limit of *nQn* length.
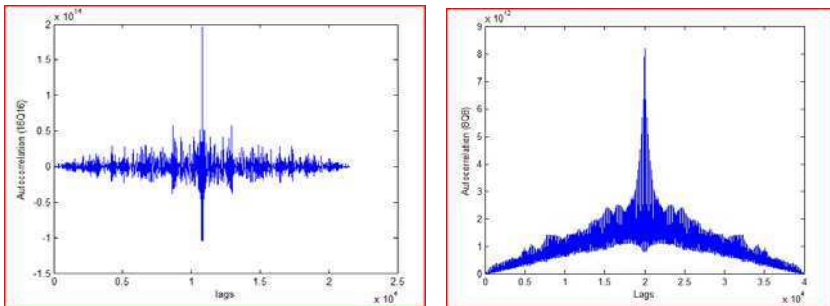

Fig. 15 *Autocorrelation* results for *16Q16* and *8Q8* format chaotic signals of *Lorenz's* architecture.

6.6 Logic Synthesis Results

Performances of an hardware implementation could be measured with the following metrics: hardware cost (in *LUTs*, *registers*, *slices*, *RAM*, specific computing blocks, etc.), operating frequency (in MHz), throughput (in Mbit/sec) and an efficiency measurement, e.g., throughput /hardware cost. *FPGA* synthesis and implementation tools allow low level description of a hardware design to be translated

into the programming file for an *FPGA*. At this step of implementation, one evaluation performance can be achieved.

The synthesis results after *place* and *route,* and performance analysis of the *Lorenz's RTL* implementation are shown in Table 2. This table specifies the hardware resources in terms of the *Slice* or the *Slice Flip-Flops* numbers and the speed performance.

Table 2 Implementation results with a *Virtex II FPGA* for the *Lorenz*'s chaotic System.

Device utilization summary - XC2v1000fg456-4 FPGA			
Number of Slices	1926	out of	5120
Number of Slice Flip Flops	791	out of	10240
Number of 4 input LUTs	2718	out of	10240
Number of bonded IOBs	11	out of	324
Number of MULT18X18s	40	out of	40
Number of GCLKs	1	out of	16
Maximum Frequency	15.598 MHz		

As can be seen, the hardware implementation exhibits good performances in terms of the frequency and resource costs required. The implementation on a *Xilinx* Virtex II device uses only 1926 *CLB-Slices*, 40 multipliers and no block RAMs. The results show that a real time *Lorenz*'s chaotic generator can be efficiency implemented with *FPGA* technology. Indeed, it can stated that an attractive trade off between high speed and low logic resources can be achieved. To evaluate the behaviour of the proposed system, it is necessary to use some evaluation metrics. The metrics used for the evaluation results for this system are the *Throughput rate* and the *Time latency*. The *Throughput* rate is defined as the number of bits key in a unit of time for a stream encrypted (or decrypted). More precisely, the number of bits key per unit time duration of the clock period.

$$Throughput = q * f_{clk} \tag{24}$$

where f_{clk} is the operating clock frequency and q is the bit representation of one random key. In the case study and from the performance results (see Table 2), a maximal *Throughput rate* of 124 Mbps is achieved. This rate is computed after the initialization phase. *Latency* is defined as the time necessary to generate a single random key after the start of the generator. *Time latency* is calculated as:

$$Latency = \frac{nu\_stage\_pipeline}{f_{clk}} \tag{25}$$

where *nu_stage_pipeline* corresponds to the number of the pipeline stages in the design. The presented implementation of the *Lorenz*'s chaotic system requires 6 clock cycles to generate one random key (more details, see the C*ontroller state machine* used and described in the Section 6.7) corresponding to a *Time latency* of 388 ns.

6.7 *Physical Implementation: Floorplanning, Placement and Routing*

In this of this section, we consider the *XUP Xilinx Virtex-II Pro* Development platform for physical hardware implementation. The *XUP* System consists of a high performance *Virtex*-II Pro *FPGA* (*XCV2PFF896-7*) surrounded by peripheral components that can be used to create a complex hardware system. Figure 16 shows a block diagram of the *XUP Virtex-II Pro Development* System, while Figure 17 gives a photo of the *XUP Xilinx Virtex-II Pro* platform [42]. Noted that an audio *CODEC* (*AC97*) and stereo power amplifier are included on the *XUP* platform provide all of the analog functionality [43].
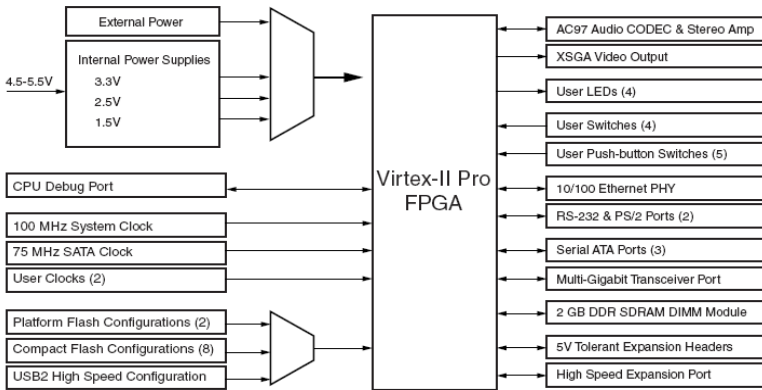
Fig. 16 *XUP Virtex-II Pro Development* System Block Diagram.
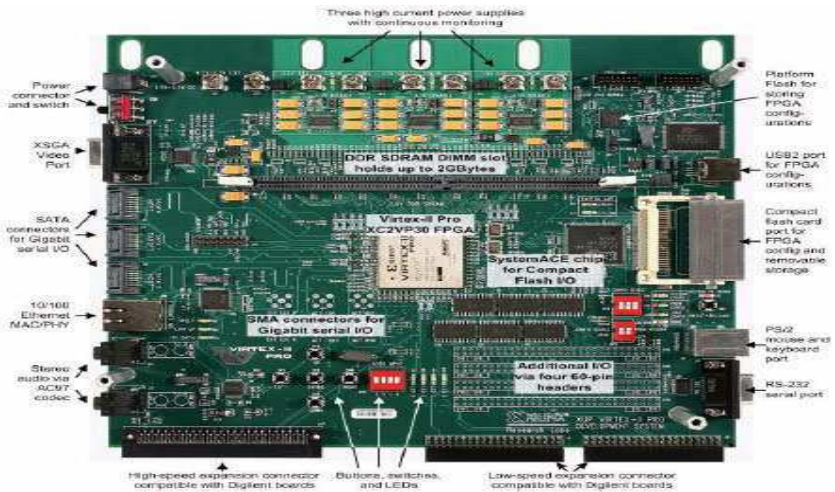
Fig. 17 Photo of the *XUP Xilinx Virtex-II Pro* platform.

An overview of the hardware implementation of the *RTL* architecture of the *Lorenz*'s system using *Virtex-II Xilinx FPGA* technology and implemented in the *XUP Virtex-II Pro Development* board is depicted in Figure 18.
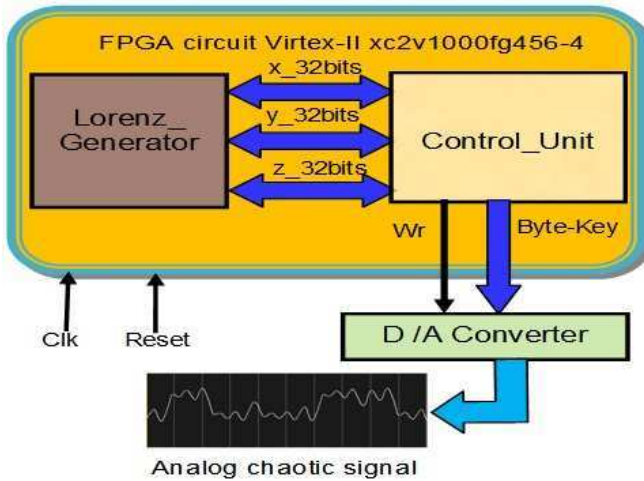
Fig. 18 Digital Hardware architecture of *Lorenz's* chaotic system.

The hardware chaotic generator has:

-Two inputs: one global clock system (denoted as *Clk*) and a reset input (denoted as *reset*).
- Three outputs denoted as (s_0, s_1, s_2) and corresponding to the generated *Lorenz's* chaotic system signals (x, y, z).

The architecture system consists of two main modules: *Control_Unit* and *Lorenz_Generator* sub-modules. The *Control_Unit* sub-module is a *Moore Finite State Machine* (or Moore *FSM*) which manages and schedules the different operations and functions of our proposed chaotic system. *Lorenz_Generator* sub-module generates the random Keys using the *RK-4* method as described in Section 5.1, which implements *Lorenz*'s nonlinear equation system defined par the set of equation (23) and designed by the *RTL* architecture presented in Figure 12. The implemented *FSM* of the *Controller* module, which schedules the *RK-4* numerical resolution, is described in the Figure 19. The *FSM* is composed of 6 states (denoted as *ST0* to *ST5*) and it driven by the global clock system without edge conditions. These states are described as follows:

ST0 : Initial state. The outputs are resetting. The *Lorenz_Generator* sub-module is initialized with the initial conditions such as ($x = x_0$, $y = y_0$ et $z = z_0$).
ST1 : *Runge-Kutta's* k_0, m_0 and l_0 parameters are computing in according of the equations (11), (12) and (13). The first intermediate results are x_1, y_1 et z_1 are then determined.
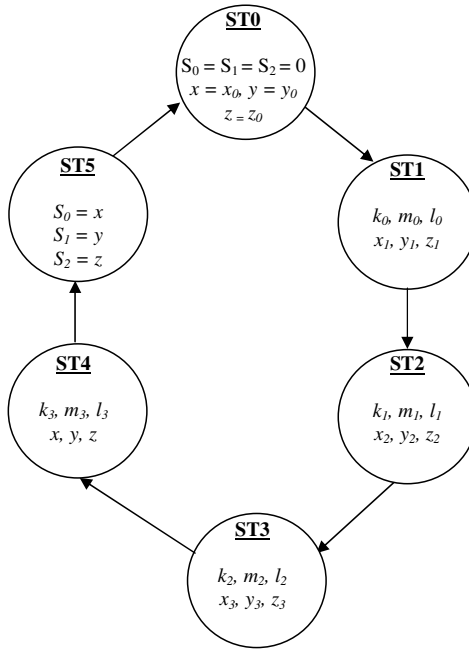
Fig. 19 *Finite State machine* of the *controller* sub-module.

ST2 **to** **ST4 :** Schedule the iterative computing. The intermediate numerical values of the fourth order *Runge-kutta's* resolution are executed. During these steps, the *Runge-Kutta's* resolution of *Lorenz's* nonlinear functions F, G and Q (defined by the equation 7) are computed by the *Lorenz_Generator* sub-module.

ST5 : Last state where the *Lorenz's* signals (x, y, z) are assigned to the outputs of the system.

All steps are repeated so that chaotic digital signals are obtained at the output. Once the chaotic signals (x, y and z) with 32 bit wordlenght are obtained, they are converted to analogue format as a sequence of 8 bits using a *Digital to Analog converter* (*DAC*) and this process is repeated so that real-time chaotic signals are obtained at the output of the *DAC* for visualization on an oscilloscope [44].

The performance of implemented architecture mainly depends on the accuracy of the size of fixed point data representation. The implementation on a *Xilinx Virtex-II Pro* device uses up to 1926 *CLB-Slices* (see Table 2) when the fixed-point data format is *16Q16* while using only 419 *CLB-Slices* for an *8Q8* format. Figure 20 gives the placement and routing on *Virtex II-Pro FPGA Chip* of the proposed *16Q16 - Lorenz's* architecture.
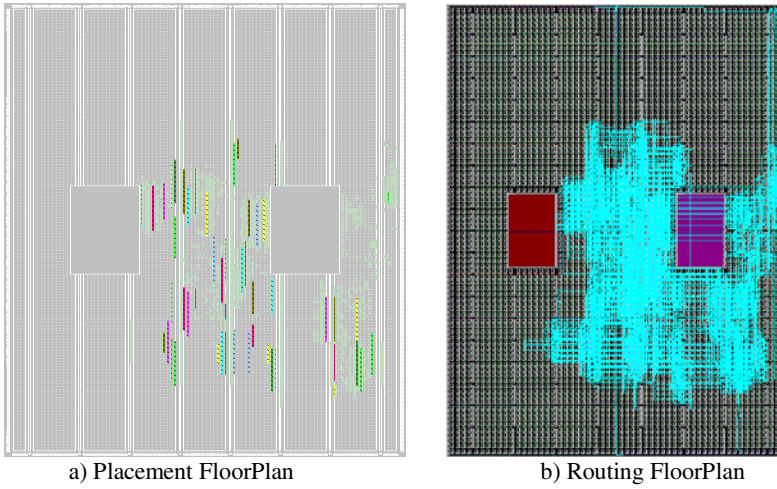
a) Placement FloorPlan   b) Routing FloorPlan

Fig. 20 Placement and routing on *Virtex II-Pro* Chip of the *Lorenz*'s *16Q16* architecture.

6.8 Real Time Measurements

This section considers the real time measurements of the chaotic signal generated by the hardware implementation. Figure 21 gives a view of the experimental hardware implementation and measurements of the *Lorenz's* chaotic signals. Real time measurements and digital acquisition can be made. The x, y and z real-time chaotic signal results of the *Lorenz*'s generator, obtained by a direct
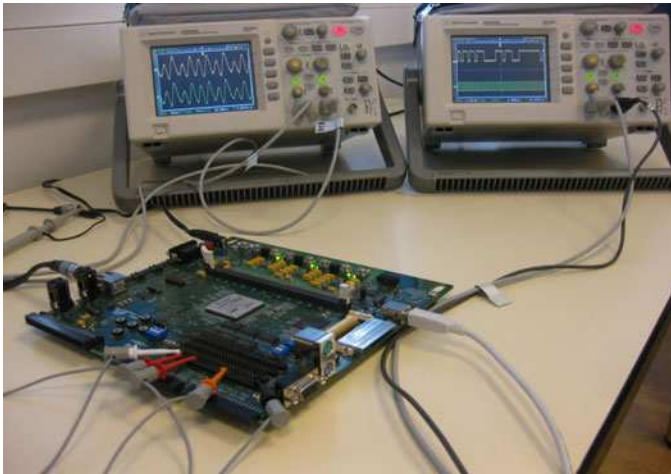
Fig. 21 Photo of the experimental hardware implementation and measurements of the *Lorenz's* chaotic signals.

implementation within *XUP Xilinx Virtex-II Pro* platform (after the download of the configuration *bitstream* file), are given in Figures (22.a), (22.b) and (22.c), respectively.

These snapshots are given by a Tektronix digital oscilloscope [44]. These results can be compared with those obtained using *Matlab* (see Figure 11) and *ModelSim* (see Figure 14) simulation tools to ascertain whether these results are similar. The measured real-time attractors (x-z) and (x-y) are presented in Figures (22.d) and (22.e), respectively. These results clearly confirm and validate that the implemented chaotic system work well in the chaotic mode.
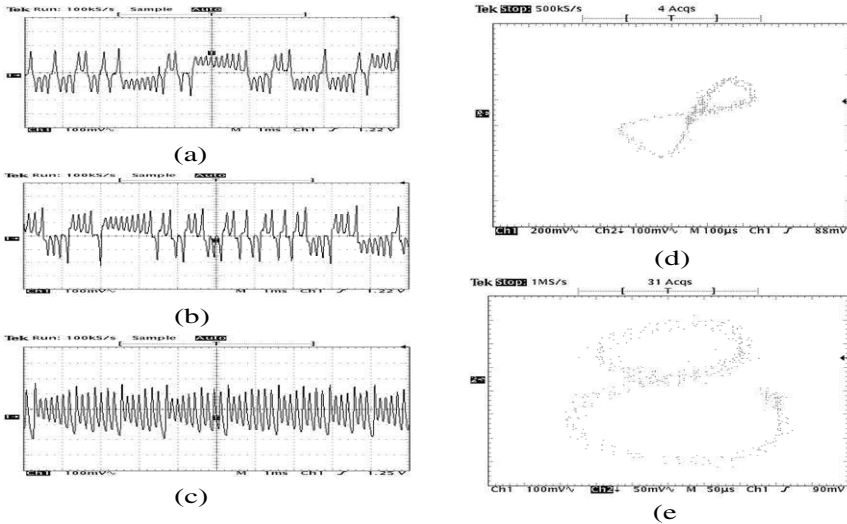
Fig. 22 Real-time results of *Lorenz's* chaotic generator: (a) x chaotic signal, (b) y chaotic signal and (c) z chaotic signal, (d) (x-y) attractor, (e) (y-z) attractor.

6.9 Feed-Back Chaotic Synchronization

The feasibility and efficiency of the encryption scheme could be demonstrated from the synchronization of two *Lorenz*'s chaotic systems (used as hardware key cipher generator for encrypted communications) where one correspond to a *Master* chaotic system embedded in one *FPGA* circuit (*Transmitter* side) and the second is the *Slave* chaotic system also embedded in an *FPGA* (*Receiver* side).

A real-time hardware implementation of the *Feed-Back chaotic synchronization* (*FCS*) between two chaotic signal generators using *FPGA* technology can be made. Figure 23 depicts the principle, structure and implementation of *FCS* allowing also to evaluate the synchronization error rate.
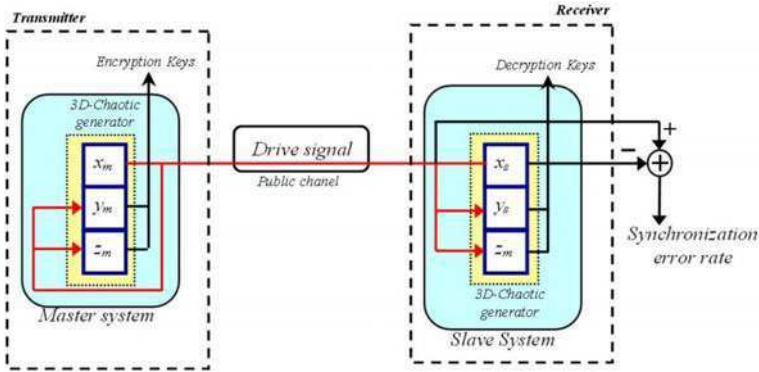
Fig. 23 Illustration of the FCS hardware implementation.

In this implementation, the *Master Lorenz* model of the set of equations (23), and the *Slave* system are nudged toward values obtained from the *Master* run as shown in the set of equations (26):

$$\frac{dx_s}{dt} = \sigma(y_s - x_s) \tag{26.a}$$

$$\frac{dy_s}{dt} = -x_m z_s + r x_m - y_s \tag{26.b}$$

$$\frac{dz_s}{dt} = x_m y_s - bz \tag{26.c}$$

where the subscript m represents the *Master* system (x_m, y_m, z_m) and s represents the *Slave* system (x_s, y_s, z_s). We consider that the two trajectories $x_m(t)$ and $x_s(t)$ are synchronized if :

$$\lim_{t \to \infty} |x_m(t) - x_s(t)| = 0 \tag{27}$$

The role of the *FCS* is to transmit the chaotic drive signal $x_m(t)$, which is then in-jected into the two subsystems (y_m, z_m) and (y_s, z_s) (as illustrated in Figure 23). At the receiver side, the *Slave* system regenerates the chaotic signal $x_s(t)$ and produces a synchronization error rate between the received drive and the regener-ated drive signals. An overview of the *Register Transfer Level* (*RTL*) architecture for the *Master* chaotic system is given in Figure 24. More precisely, this figure depicts our data-path processing architecture which is based on fixed parameters σ, r and b as specified in the previous Section 6.3.
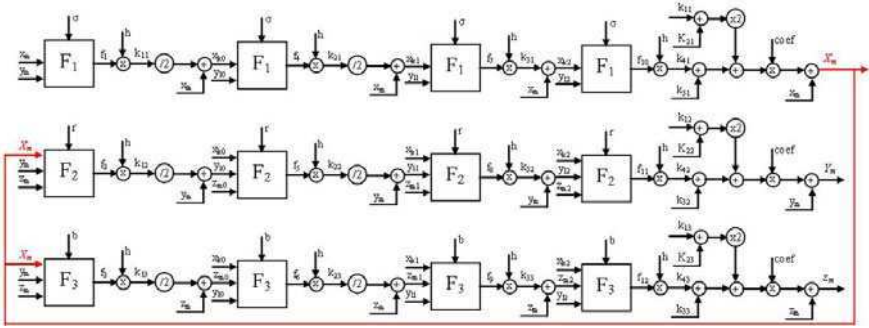
Fig. 24 *RTL* architecture of the *Master* chaotic system.

Similarly, an *RTL* architecture of the *slave* chaotic system is depicted in Figure 25. The parameters of the *Slave* chaotic system are similar to those of the *Master* one but the three main blocks F_1, F_2 and F_3. implement equations (26.a), (26.b) and (26.c), respectively.
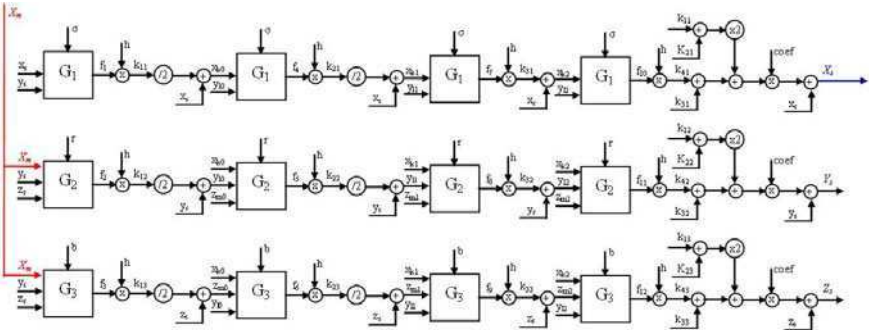

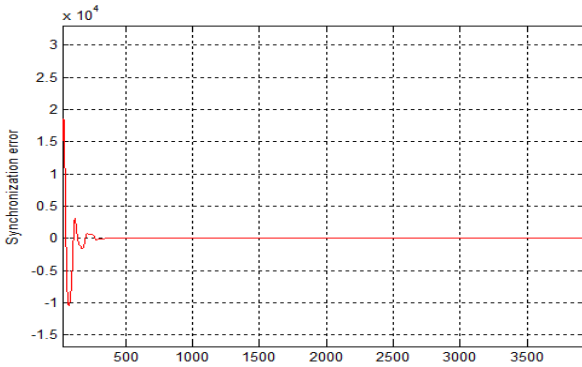**Fig. 25** *RTL* architecture of the *Slave* chaotic system.

Fig. 26 Measurement result of the *synchronization error rate*.

The synchronization error is depicted in Figure 26. The synchronization between the *Master* and *Slave* chaotic systems is achieved after 470 samples. The time of the simulation corresponding of the maximum frequency allowed by the hardware synthesis (see Table 2).

The transmitter and the receiver real-time attractors measurements obtained by a direct implementation are depicted in Figures (27.a) and (27.b), respectively.
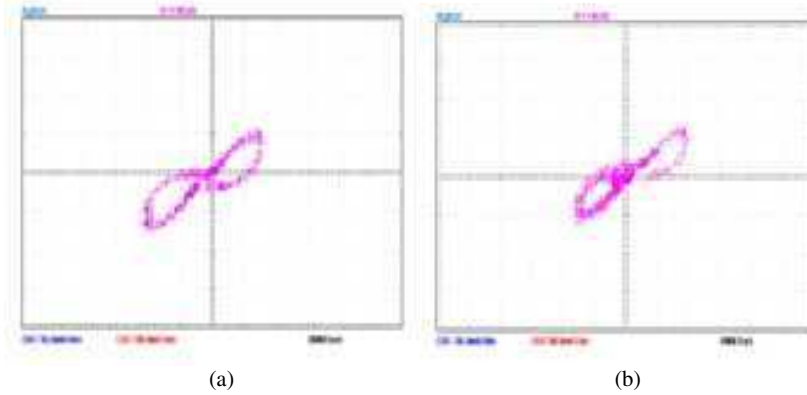

(a)   (b)

Fig. 27 Real-time (x-y) chaotic attractor of the *Lorenz's* model: (a) Transmitter attractor, (b) Receiver attractor.

These snapshots give the measured real-time (x-y) attractors from the drive signal X_m and the *regenerated X_s* at the transmitter and receiver, respectively.

7   Application Image Encryption Scheme

We consider in this section a cipher key scheming based on the *Lorenz's* chaotic system for designed an embedded c*ryposystem* for real-time image encryption. According to the basic principle of cryptology, a cryptosystem should be sensitive to the key, i.e., the cipher-text should have close correlation with the key. To accomplish this requirement, we must use an efficient (ideally, truly random) key generation mechanism and mix the key thoroughly into the plaintext through the encryption process. The *Lorenz*'s chaotic generator is employed in the key generation scheme in order to control the generation of the cipher key. The generated key used in this encryption scheme is a binary sequence of 32 bits. The complete image encryption/decryption scheme consists of two steps of operation as shown in Figure 28.

Step 1. Chaotic key selection. A key is generated from the previous key and one sequence of 32 bits as the key is selected in a chaotic way.

Step 2. Perform XOR or NXOR operation. According to the key binary sequence generated, each image pixels are then *XORed* with the selected key.

Therefore, the decipher procedure is similar to that of the encipher process illustrated above but with a reverse operation sequence to that described in Steps 1 and 2 above. Since both decipher and encipher procedures have similar structures, they have essentially the same algorithmic complexity and time of operation.
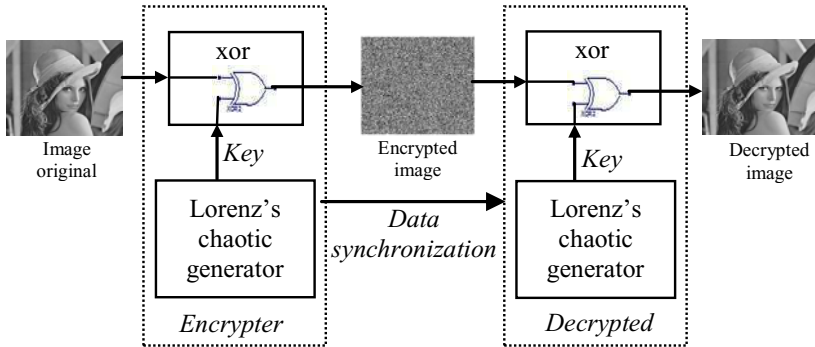

Fig. 28 Block diagram of the image encryption based chaotic key generator.

Figure 29 gives the encryption results applied on 256x256 image based on the *Lorenz*'s cipher key generator with the initial conditions [x_0=0, y_0= 5, z_0=25]. An efficient encryption scheme should resist all kinds of known attacks, such as known-plain-text, cipher-text only, statistical, brute-force attacks and so on. In the case study, it should be sensitive to the cipher keys. This means that encryption scheme must sensitive to initial conditions of the chaotic generator. Some security analysis can be performed on the image encryption scheme, including the most important ones like *key analysis*, *statistical analysis*; in order to assess the usefulness of the cipher technique in terms of security. A key space analysis and testing can be carefully performed and completely carried out with the results obtained summarized as follows. A typical initial conditions key sensitivity can be performed, according to the following steps:

1. First, a 256 x·256 image is encrypted by using the initial conditions of the *Lorenz*'s key generator [x_0=0, y_0= 5, z_0=25].
2. The least significant bit of the initial conditions of key generator is changed. The initial conditions modified ([x_0=0.125, y_0= 5, z_0=25] in the considered example) is used to encrypt the same image becomes.
3. The above two ciphered images, encrypted by the two slightly different initial conditions key, are compared.
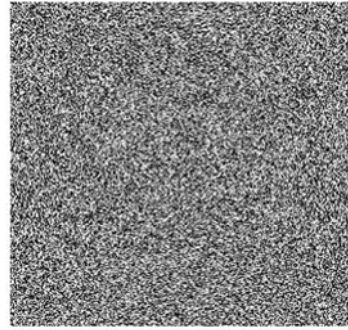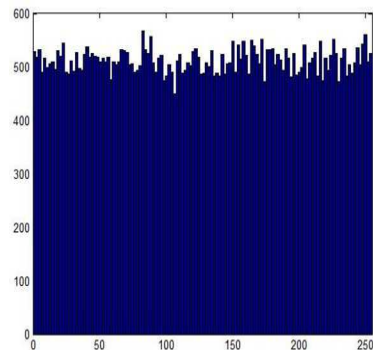
<table>
<tr><td>Original Image</td><td>Encrypted Image with initial conditions
[x0=0, y0= 5, z0=25]</td></tr>
</table>

Fig. 29 Encryption image base the *Lorenz's* chaotic Key cipher generator.

Figure 30 shows the test results. The result outcome has shown that the two en-crypted images with the two slightly different initial conditions are very different with more than 99 % of difference rate in terms of pixel grey-scale values, al-though there is only one bit difference in the two initial conditions of key chaotic generator. Thus, when a 32-bit key from an initial condition is used to encrypt an image while another trivially modified initial condition key is used to decrypt the ciphered image, the decryption also completely fails. A statistical analysis per-formed on the image encryption algorithm demonstrates its efficiency against sta-tistical attacks. This is shown by the histograms and on the correlation of adjacent pixels in the ciphered image. Figure 31 gives the histogram of the original and en-crypted image. One can see that the histogram of the ciphered image is fairly uni-form and is significantly different from that of the original image.
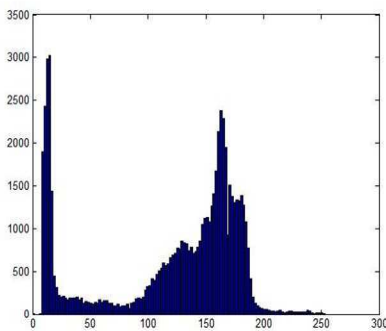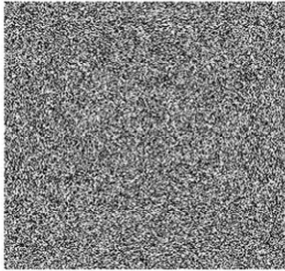

Histogram of Original Image   Histogram of Encrypted Image
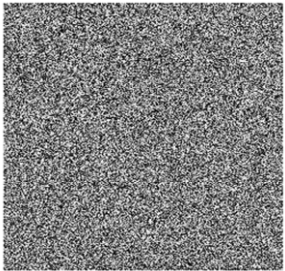
Fig. 31 Histograms of the plain-image and the cipher-image.

128	156	135	122	38	12
96	207	204	121	110	33
77	136	148	42	98	102
254	234	20	97	113	15
209	100	55	221	46	222
155	208	7	80	182	141
….					

Values of the Grey level pixels with
initial conditions [x0=0, y0= 5, z0=25]

Encrypted Image with initial conditions
[x0=0, y0= 5, z0=25]

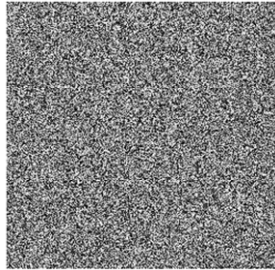| 128 | 157 | 244 | 114 | 241 | 182 |
30	33	78	136	212	247
171	111	174	17	181	144
84	79	111	212	2	104
201	118	150	121	112	165
66	46	70	16	35	222
….					

Values of the Grey level pixels with
initial conditions [x0=0.125, y0= 5, z0=25]

Encrypted Image with initial conditions
[x0=0.125, y0= 5, z0=25]

0	1	109	248	203	170
190	82	130	15	102	214
94	231	26	231	83	42
86	101	91	115	145	89
248	18	95	156	66	199
167	94	63	192	109	81
….					

Values of the Grey level pixels of the
difference image

Difference encrypted images

Fig. 30 Test of the initial conditions sensitive of the chaotic key generator.

8 Summary

This chapter presents hardware implementations of a random key generator based on a *3D* chaotic systems for embedded data stream encryption systems. The presented random key generator architecture based on the architecture modeling of the *Runge-Kutta* method (*RK-4*) is particularly attractive since it provides low-cost security communication solutions for embedded systems. This hardware design approach is validated by showing that real-time *Lorenz*'s chaotic signals obtained with the *RTL* architecture are similar to the software simulation counterparts. Moreover, cipher embedded systems can take several advantages of the use of *FPGA*s. Indeed, the experimental results using *Xilinx Virtex* technology have

demonstrated that the presented design approach can lead to designs with small logic area, satisfactory throughput rates and low latency for embedded applications. It has also shown that an implementation working with a reduction of the fixed-point *8Q8* integer precision setting, the hardware architecture continued to remain a chaotic system. The preservation of the chaotic dynamic property of the implemented hardware chaotic system can be explored by studying the trade-off allowing to obtain the appropriate secure for some embedded applications. This chapter concludes that the digital hardware implementation of synchronized chaos cipher (based on a *Feed-Back Chaotic* technique) for embedded security applications is simple, exhibits attractive, good performances and can be used for the implementation of others *3D* chaotic systems such as *Chen's* system, *Chua's* system, *Lü's* system, *Rössler's* system, *Colpitts'* system, and so on. Therefore, this work will permit to use in choice these chaotic generators in secure digital chaotic communication systems.

Finally, an image encryption application based on new random key generator scheme, which uses synchronized *Lorenz's* chaotic systems is also presented. The proposed scheme incorporates the *chaotic* key generation and its use to design a fast and secure symmetric image encryption, thereby increasing its resistance to various attacks such as the statistical and Key analysis attacks. Thorough experimental tests have been carried out with detailed numerical analysis, demonstrating the security and fast speed of the new image encryption scheme. This scheme is particularly suitable for real-time internet image encryption and transmission applications and is particularly attractive since it provides low-cost image encryption solutions for embedded systems.

References

1. Pecora, L.M., Carroll, T.L.: Synchronization in chaotic systems. Physical Review Letters 64(8), 821–824 (1990), doi:10.1103/PhysRevLett.64.821
2. Cuomo, K.M., Oppenheim, A.V.: Circuit implementation of synchronized chaos with application to communications. Physical Review Letters 71(1), 65–68 (1993)
3. Zhang, Y., Tao, C., Du, G., Jiang, J.J.: Physical Review Letters E 71, 016217 (2005)
4. Lorenz, T.E.N.: Deterministic nonperiodic flow. Journal of the Atmospheric Sciences 20(2), 130–141 (1963)
5. Yang, T.: A survey of chaotic secure communication systems. International Journal of Computational Cognition 2(2), 81–130 (2004)
6. Abel, A., Schwartz, W.: Chaos Communications- Principles, Schemes and Systems analysis. In: Proc. of the IEEE Inst. for Fundamentals of Electr. Eng. & Electron., Dresden Univ. of Technol, vol. 90, pp. 691–710 (2002)
7. Kvarda, P.: Investigating the Rössler attractor using Lorenz plot and Lyapunov exponents. Radioengineering 11(3), 22–23 (2002)
8. Indrusiak, L.S., Dutra e Silva Jr., E.C., Glesner, M.: Advantages of the Linz-Sprott weak nonlinearity on the FPGA implementation of chaotic systems: a comparative analysis. In: Proc. Int. Symp. Signals, Circuits and Sys., vol. 2, pp. 753–756 (2005)

9. Chua, L.0.: Chua's circuit: Ten years later. IEICE Trans. Fundamentals E77-A, 1811–1822 (1994)
10. Kocarev, L., Halle, K., Eckert, K., Chua, L.: Experimental demonstration of secure communication via chaotic synchronization. Int. J. Bifur. Chaos 2, 709–713 (1992)
11. Tao, Y.: Chaotic secure communication systems history and new results. Telecom. Rev. 9, 597 (1999)
12. Cuomo, K.M., Oppenheim, A.V., Strogatz, S.H.: Synchronization of Lorenz-Based Chaotic Circuits with Applications to Communications. IEEE Transactions on Circuits and Systems-11: Analog and Digital Signal Processing 40(10), 626–633 (1993)
13. Parlitz, U., Chua, L.O., Kocarev, L., et al.: Transmission of digital signals by chaotic synchronization. Int. J. Bifurcations Chaos 2, 973–977 (1992)
14. Matsumoto, T.: Chaos in electronic circuits. IEEE Inst. of Elec. and Elecs Eng. 75(8), 1033–1046 (1987)
15. Giannakopoulos, K., Souliotis, G., Fragoulis, N.: An integratable chaotic oscillator with Current Amplifiers. In: IEEE Int Symp. on Signals, Circuits and Systems, July 13-14, vol. 1, pp. 1–4 (2007)
16. Ozoguz, S., Ates, O., Elwakil, A.S.: An integrated circuit chaotic oscillator and its application for high speed random bit generation. In: ISCAS 2005, vol. 5, pp. 4345–4348 (2005)
17. Cha, C.Y., Lee, S.G.: Complementary Colpitts Oscillator in CMOS Technology. IEEE Transaction on Microwave Theory and Techniques 53(3) (March 2005)
18. Tanougast, C., Weber, S., Millerioux, G., Bouridane, A., Daafouz, J.: VLSI architecture and FPGA implementation of a hybrid message embedded self-synchronizing stream cipher. In: 4th IEEE Int. Symp. on Elec. Design, Test and Applications, pp. 386–389 (2008)
19. Sadoudi, S., Tanougast, C., Azzaz, M.S., Dandache, A., Bouridane, A.: Real-time FPGA Implementation of Lü's Chaotic Generator for Cipher Embedded System. In: ISSCS 2009, Iasi, Romania, July 9-10 (2009)
20. Azzaz, M., Tanougast, C., Sadoudi, S., Dandache, A., Monteiro, F.: Real Time Image Encryption Based Chaotic Synchronized Embedded Cryptosystems. In: 8th IEEE International NEWCAS Conference, IEEE Circuits and Systems Society, Montréal, Canada, June 20-23 (2010)
21. Linz, S.J., Sprott, J.C.: Elementary chaotic flow. Phys. Lett. A 259, 240 (1999)
22. Sprott, J.C., Linz, S.J.: Algebraically simple chaotic flows. Int. J. of Chaos Theory and Applications 5.3 (2000)
23. Mathworks, Matlab Software, Version 7.3, Mathworks (2006)
24. Chen, H.H., Chiang, J.S., Lin, Y.L., Lee, C.I.: Chaos synchronization of general Lorenz, Lü, and Chen systems. Hsiuping Journal 15, 159–166 (2007)
25. Lü, J., Chen, G.: A new chaotic attractor coined. Int. Journal of Bifurcation and Choas 12(3), 659–661 (2002)
26. Giannakopoulos, K., Souliotis, G., Fragoulis, N.: An integratable chaotic oscillator with Current Amplifiers. In: IEEE Int Symp. on Signals, Circuits and Systems, July 13-14, vol. 1, pp. 1–4 (2007)
27. Lindsay, P.S.: Period doubling and chaotic behavior in a driven anharmonic oscillator. Phys. Rev. Lett. 47(19), 1349–1352 (1981)
28. Piper, J.R., Sprott, J.C.: Simple autonomous chaotic circuits. IEEE Trans. on Circuits and Systems-II 57(9) (2010)

29. Maggio, G.M., De Feo, O., Kennedy, M.P.: Nonlinear analysis of the Colpitts oscillator and applications to design. IEEE Trans. on Circuits and Systems I: Fundamental Theory and Applications 46, 1118–1130 (1999)

30. Wegener, C., Maggio, G.M., Kennedy, M.P.: An approximate one-dimensional model for the chaotic Colpitts oscillator. In: Proc. Nonlinear Dynamics of Electronic Systems, pp. 441–446 (1996)

31. Kennedy, M.P.: Chaos in the Colpitts oscillator. IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications 41, 771–774 (1994)

32. Chen, H.H., Chiang, J.S., Lin, Y.L., Lee, C.I.: Chaos synchronization of general Lorenz, Lü, and Chen systems. Hsiuping Journal 15, 159–166 (2007)

33. Azzaz, M.S., Tanougast, C., Sadoudi, S., Bouridane, A., Dandache, A.: An FPGA implementation of a Feed-Back Chaotic Synchronization for secure communications. In: 7th International Symposium on Communication Systems Networks and Digital Signal Processing, pp. 239–243 (2010)

34. William Press, H., Flannery, B.P., Teukolsky, S.A., Vetterling, W.T.: Numerical Recipes in C, The Art of Scientific Computing. Cambridge University Press, Cambridge (1992)

35. Cartwright, J.H.E., Piro, O.: The Dynamics of Runge-Kutta Methods. Int. J. Bifurcation and Chaos 2, 427–449 (1992)

36. Sobhy, M.I., Aseeri, M.A., Shehata, A.E.R.: Real Time Implementation of Continuous (Chua And Lorenz) Chaotic Generator Models Using Digital Hardware. In: Proc. of the Third International Symposium on Communication Systems Networks and Digital Processing, pp. 38–41 (1999)

37. Aseeri, M.A., Sobhy, M.I., Lee, P.: Lorenz Chaotic Model Using Field Programmable Gate Array (FPGA). In: Midwest Symposium on Circuit and Systems, pp. 686–699 (2002)

38. Xilinx, VirtexII-pro complete Datasheet, Xilinx (2007)

39. Xilinx, Integrated Software Environment (ISE), Version 10.1, Xilinx (2008)

40. Azzaz, M.S., Tanougast, C., Sadoudi, S., Dandache, A.: Real-time FPGA Implementation of the Lorenz Chaotic Generator for Ciphering Telecommunications. In: Joint IEEE International Circuits and Systems and TAISA Conférence (2009)

41. Mentor Graphics, Modelsim SE User's Manuel, Sofware, Version 6. 4, Mentor Graphics (2008)

42. Xilinx, Xilinx University Program Virtex-II Pro Development System, Xilinx, UG069 (v1.1) (April 9, 2008)

43. Analog Devices, LC2MOS Complete, 8-Bit Analog I/0 Systems, AD7569/AD7669, Analog Devices (1996)

44. Tektronix, Digital Real-Time^{TM} Oscilloscopes TDS340A, TDS360 et TDS380, Tektronix (2006)

Chapter 10
Hardware Implementation of Chaos-Secured Optical Communication Systems

Apostolos Argyris

Department of Informatics and Telecommunications, National & Kapodistrian
University of Athens, Panepistimiopolis, 15784, Ilisia, Greece
`argiris@di.uoa.gr`

Abstract. In the present chapter, the implementation and performance of contemporary chaotic optical communication systems is presented, focusing on the physical layer encryption methods proposed so far. In communication systems that encrypt high-speed data within broadband chaotic carriers, authorized users share identical chaotic oscillators that are capable – after synchronization – of emitting exactly the same broadband chaotic optical signal. Several techniques - based on all-optical, electro-optical or photonic integrated circuits - that increase fiber communication security will be presented, while their drawbacks and limitations will be criticized. The efficiency of data encryption at the transmitter and the recovery performance from an authorized receiver are also presented through diverse fiber transmission experiments. In these experiments the security discrimination level between authorized and eavesdropping receivers are discussed. Finally, ultra-fast physical random number generators based on chaotic optical signals, as well as the potential of exploiting them in secure communication systems, are investigated.

1   Security in Optical Communications at the Physical Layer

In the present era of information technology and computer network communications, cryptography is a field of particular importance. Various cryptographic methods are routinely used to protect all types of data, from private via-web conversations to electronic fund transfers and classified communications, independently of the physical medium used for the communication. Current cryptographic techniques are based on number theoretic or algebraic concepts. Secret key cryptography uses a secret key, such as the DES and AES algorithms, in which both sender and receiver use the same key to encrypt and decrypt. This is a fast computational method; however getting the secret key to the recipient in the first place is a problem that is often handled by a public-key procedure. On the other hand, public-key cryptography is used to protect sensitive data during transmission over various channel types that support personalized communication [26,82] and includes tasks such as message encryption, key exchange, digital signatures, and digital certificates [67]. The above algorithmic types of cryptography secure the upper layers of

any type of communications, regardless the transmission medium. In the last decade strong research activity has been recorded in the securing data transmission by taking into account the properties of the transmission medium and incorporating specific advantages that might exist in order to strengthen the security of the utilized communication channel. The specific chapter focuses on the properties of fiber-optic communication systems that allow an *"upgrade"* in the protection of the link, by using optical chaos as a *"physically generated cipher"*. Despite fiber optic networks' reputation for being more secure than standard wiring or airwaves, the truth is that fiber cabling is just as vulnerable to eavesdropping as wired or wireless networks. Tapping into fiber optic cables with relatively inexpensive and appropriate equipment, an experienced hacker can perform a successful attack. Optical network attacks are accomplished by extracting light from the ultra-thin glass fibres by gaining access to the fiber optic cable. Although most of this cabling is difficult to access — it's underground, undersea, encased in concrete etc. — plenty of cables are readily accessible for eavesdroppers. Some cities, for example, have detailed maps of their fiber-optic infrastructure posted online in an effort to attract local organizations to include themselves into the network. After gaining access to the cable itself, the next step is to extract light and, eventually, data from the cable. Bending seems to be the easiest method, being practically undetectable since there is no interruption to the light signal. Such potential hacking attempts on the fiber infrastructure of optical networks have motivated the development of systems that provide transmission security: the component of this type of communications security results from the application of measures designed to protect transmissions from interception and exploitation by means other than cryptanalysis. Two main categories of this type of security have been established so far: *"quantum cryptography"* that exploits the quantum nature of light and *"chaos encryption"* that exploits the potential of the optical emitters to operate under chaotic conditions.

1.1 Quantum Cryptography

Quantum cryptography is a technique for two parties to form a key on an open optical network [17,34,105]. Such keys can subsequently be used for the encryption of data sent on the network between the two parties. An attraction of quantum cryptography is that fundamental laws of quantum mechanics guarantee its security. It allows the detection of unauthorized eavesdropping, as well as providing a guarantee of security when there is no eavesdropper present. This is not possible using any other form of key distribution, which relies either upon the difficulty of factorizing large numbers, or the assumed privacy of the network. In optical quantum cryptography the bits used to form the key are carried by single photons travelling either, along an optical fiber, or in an optical free space link. Information can be encoded on the photons in a variety of ways, such as by their polarization or phase. Because the information is carried by a single photon, it is not possible for a hacker to tap in and remove part of the signal. Since single photons don't split, if the hacker measures the photons on the fiber, they will not be received at the other end, alerting the intended recipient to the presence of the hacker. Furthermore, the technique is also secure from a slightly more sophisticated type of

eavesdropping where the hacker first measures the photons and then retransmits them. This is because the laws of quantum mechanics tells us quantum bits (or qubits) of information, such as encoded single photons, have the peculiar characteristic that they are disturbed by measurement. This fact allows the legitimate receiver of the message to test whether it has been intercepted or altered by a hacker on the channel.

Quantum cryptography belongs to the class of the hardware-key cryptography and thus can be used only to exchange a secret key and is not suitable for real time data encryption, at least up to now [99]. The reason is related to the low bit-rate (in the order of tens of KHz) and the incompatibility with some key components of the optical communication systems – the optical amplifiers – that are needed for long distance transmission links.

1.2 *Optical Chaos Encryption*

A different approach to strengthen the security of optical high-speed data transmitted in fiber networks has been investigated extensively the last decade. In this approach data are encoded at the physical layer (hardware encryption) using chaotic carriers generated by lasers operating in the non-linear regime. The objective of chaos hardware encryption is to encode the information signal within a chaotic carrier generated by components whose physical, structural and operating parameters form the secret key. Once the information encoding is carried out, the chaotic carrier is sent for transmission to the authorized receiver. Decoding is achieved directly in real-time through the *"chaos-synchronization"* process, within which the optical carrier with its chaotic signature is cancelled. The principle of operation of the chaos-based optical communications systems is schematically depicted in fig. 1.
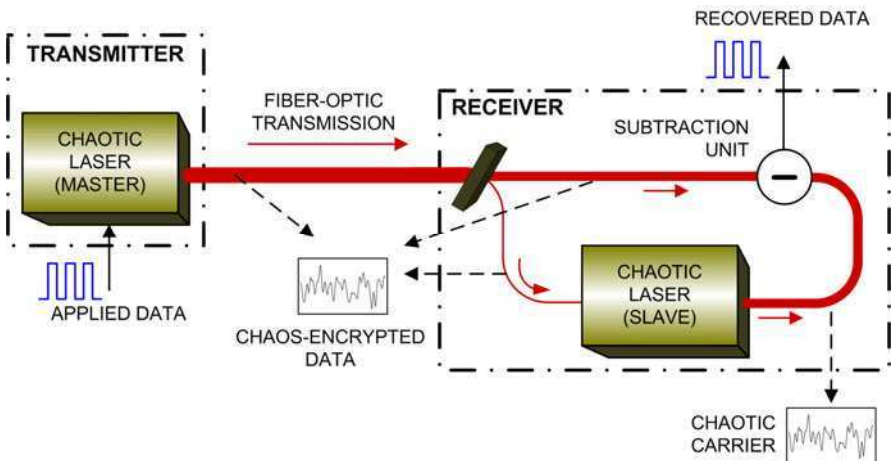

Fig. 1 Block diagram of an optical communication system based on optical chaos encryption.

A more analytical description for the above process follows below. In conventional communications systems an optical oscillator – usually a semiconductor laser – generates a coherent optical carrier on which the information is encoded using one of the many existing modulation schemes. On the contrary, in the proposed approach of the chaos based communications the transmitter consists of the same oscillator forced to operate in the chaotic regime – e.g. by applying external optical feedback – producing thus an optical carrier with extremely broadband spectrum (up to tens of GHz). The information – typically based on an on-off keying bit stream – is encoded on this chaotic carrier using different techniques (e.g. a simple yet efficient method is to use an external optical modulator electrically driven by the information bit stream while at its input is coupled the optical chaotic carrier). The amplitude of the encrypted message in all cases is kept small in respect to the amplitude fluctuations of the chaotic carrier, so that it would be practically impossible to extract this encoded information using conventional techniques like linear filtering, frequency domain analysis or phase-space reconstruction. Especially the latter assumes a high complexity of the chaotic carrier and is directly dependent on the method that the chaos dynamics are generated. At the receiver side of the system a second chaotic oscillator is used, as *identical* to that of the transmitter. This identity refers to the semiconductor laser structural, emission (emitting wavelength, slope efficiency, current threshold, etc.) and intrinsic (linewidth enhancement factor, non-linear gain, photon lifetime, etc.) parameters, as well as to the feedback loop characteristics (cavity length, cavity losses, possible non-linearity, etc.) and the operating parameters (bias currents, feedback strength, etc.). The above set of hardware-related parameters constitutes the key of the encryption procedure.

The message extraction procedure is based on the so called *"synchronization"* process. In the context of chaos communication terminology, synchronization expresses that the irregular time evolution of the chaotic emitter's output in the optical power can be perfectly reproduced by the receiver, provided that both transmitter and receiver chaotic oscillators are identical in terms of the above set of parameters. Even minor discrepancies between the two oscillators can result in degraded synchronization, which means deviation from a perfect reproduction of the emitter's chaotic carrier.

The key issue for efficient message decoding resides in the fact that the receiver synchronizes to the chaotic oscillations of the emitter's carrier without being affected by the encoded message, also referred in literature as *"chaos filtering effect"*. Based on the above considerations, the receiver's operation can be easily understood. Part of the incoming message with the encoded information is injected into the receiver. Assuming all those conditions that lead to a sufficiently good synchronization quality, the receiver generates at its output a chaotic carrier almost identical to the injected, without the encoded information. Therefore, by subtracting the chaotic carrier from the incoming chaotic signal with the encoded information, the transmitted information is revealed.

Chaos-based secure communications systems provide some major advantages. They support real-time high-bit rate message encoding since the data encoding process does not introduce any additional delay relative to that of the conventional

optical communication systems. The same holds for the receiver at least for bit rates up to 10 Gb/s since the synchronization process relies on the ultrafast dynamics of semiconductor lasers (in the all-optical case) or the time response of the fast photodiodes and other nonlinear elements (in the optoelectronic approach). This is a significant advancement relative to the conventional software based approaches, where real time encoding of the bit stream - exploiting fast processors and sufficiently long bit series key - would result in much lower effective bit rate, increased complexity and cost of the system. Moreover, chaos-encrypted optical communications can act complementarily to software encryption, providing a higher level of security. Compared to the quantum cryptography, the chaos-based approach provides the apparent advantage of significantly faster data transmission speed. Additionally, it provides enhanced security, since a potential eavesdropper has two main ways to attempt to extract the chaos encoded information. The first one is to reconstruct the chaotic attractor in the phase space using strongly correlated points densely sampled in time. In this case, the number of needed samples increases exponentially with the chaos dimension. Taking into account the attractor dimension of the generated chaotic optical carriers, which in some cases (optoelectronic approach) its Lyapunov dimension is of the order of a few hundreds, and considering the characteristics of today's recording electronics this solution seems to be impossible. The second one is to identify the key for reconstruction of the chaotic time series. In the case of the chaotic encryption, the key is the hardware used and the full set of operating parameters. This means that if a semiconductor laser coupled to an external cavity is, e.g., the chaotic oscillator in the emitter, the eavesdropper must have an identical laser diode, with identical external resonator providing the same amount of feedback and to know the complete set of operating parameters. Finally, chaos optical encryption allows compatibility with the installed network infrastructure since there is no fundamental reason to preclude its application on installed optical network infrastructure. With proper compensation of fiber transmission impairments the chaotic signal that arrives at the receiver triggers the synchronization process successfully. All feasibility experiments showed that the use of erbium-doped fiber amplifiers (EDFAs) might induce some power penalty in the decoded data due to noise addition, but in any case does not prevent from synchronizing and extracting the encrypted information.

The concept of chaos synchronization was firstly proposed theoretically by Pecora and Carroll in 1990 [78]. This pioneering work triggered a burst of activities covering in the early '90s mainly electronic chaotic oscillators [24]. The first theoretical work and preliminary reports and possibility of synchronization between optical chaotic systems came out in the nineties [5,23,68]. Since then the activities in the area of optical chaotic oscillators increased exponentially. Numerous research groups worldwide reported a large amount of theoretical and experimental work, covering mainly fundamental aspects related to synchronization of optical non-linear dynamical systems [9,59,74,96,102]. Special focus was given to semiconductor laser-based systems [38,88], but there was also work on fiber-ring laser systems [1] and optoelectronic schemes [2,36]. The applicability of the concept in optical communication systems was initially proved by encoding and recovery of

single frequency tones, starting from frequencies of a few KHz [53] up to several GHz [77]. However, it's worth mentioning that single frequency encoding is much less demanding in terms of chaos complexity than pseudorandom bit sequences used in conventional communication systems. In 2002, a 2.5 Gb/s NRZ pseudo-random bit sequence has been referred to be masked in a chaotic carrier, produced by a 1.3 μm DFB diode laser subjected to optoelectronic feedback, and recovered in a back-to-back configuration without including any fiber transmission [58]. The achieved bit-error rate values of that system were of the order of 10^{-4}. The above performance was improved in 2005 by an EU consortium [41], that announced a successful encryption of a 3 Gb/s pseudorandom message into a chaotic carrier, while the system's decoding efficiency was characterized by low BER values of the order of 10^{-9}[54]. The same consortium demonstrated also a 1,55μm all-optical communication system with chaotic carriers, characterized by low BER values for the decoding message at gigabit rates [7]. A transmission system based on the above configuration has been implemented in laboratory conditions [8], as well as in an installed optical fiber network with length over 100 km [10]. These works provided so far chaos-based methods appropriate for high bit-rate data encryption but not as an integrated, compact solution. The possibility of a realistic implemen-tation of networks with advanced security and privacy properties based on chaotic encryption depend strongly on the availability of either hybrid optoelectronic or photonic integrated components. Such systems have been very recently demon-strated, employing efficient photonic integrated circuits (PICs) [11], operating at 2.5Gb/s and including transmission links over 100 km [13,14]. The latter could be further integrated in communication cards compatible with conventional computer and communication systems. This is exactly the future need covered by new de-signs of development of the proper technology for the fabrication of components appropriate for robust and secure chaotic communication systems enabling crucial miniaturization and cost reduction as well. Finally, advanced optoelectronic confi-gurations that employed phase modulation techniques, operated efficiently up to 10 Gb/s, including fiber transmission links [56].

2   Optical Chaos Generators in Optical Communications

Since the technology of optical communications and networks is currently based on laser emitters fabricated by semiconductor materials, our study is focused on such compounds capable of emitting optical signals with complex dynamics. In the paragraphs that follow, a compendious description of these chaos generators is provided, highlighting those properties that lead to chaos dynamics.

2.1   Non-linear Dynamics in Semiconductor Lasers

In semiconductor edge emitting lasers simultaneous emission in several longitu-dinal modes is common. For this reason, many strategies have been devised in or-der to guaranty single-longitudinal mode operation which is useful for single mode transmission in fiber-optic links. Large side-mode suppression ratio can be

achieved using distributed feedback reflector (DFB) lasers, distributed Bragg reflector (DBR) lasers, and vertical-cavity surface-emitting lasers (VCSELs). The temporal evolution of the electric field's amplitude of this solitary longitudinal mode emitted by a semiconductor laser is described by means of a time-delayed rate equation. This field equation has to be complemented by specifying the evolution of the total carrier population $N(t)$. In the case of single-longitudinal mode operation the evolution of the field and carrier variables is governed by the following equations:

$$\frac{dE(t)}{dt} = \frac{1-ia}{2} \cdot \left[G(t) - t_p^{-1} \right] \cdot E(t) + F_E(t) \tag{1}$$

$$\frac{dN(t)}{dt} = \frac{I}{e} + \frac{N(t)}{t_n} - G(t) \cdot |E(t)| + F_N(t) \tag{2}$$

$$G(t) = \frac{g \cdot (N(t) - N_0)}{1 + s \cdot |E(t)|^2} \tag{3}$$

where $E(t)$ is the complex slowly varying amplitude of the electric field at the oscillation frequency ω_0, $N(t)$ is the carrier number within the cavity and t_p is the photon lifetime of the laser. The detailed derivation of these equations can be found in [4,85]. In eq. (2) I/e is the number of injected electron-hole pairs by current biasing the laser, t_n is the rate of spontaneous recombination (as also known as carrier lifetime), and $G(t)|E(t)|^2$ describes the processes of the stimulated recombination. The above set of equations take into account gain suppression effects through the non-linear gain coefficient s, and also Langevin noise sources $F_E(t)$, $F_N(t)$. These spontaneous emission processes are described by white Gaussian random numbers [94] with zero mean value:

$$<F_E(t)> = 0 \tag{4}$$

and delta- correlation in time:

$$\left\langle F_E(t) \cdot F_E^*(t') \right\rangle = 4 \cdot t_n^{-1} \cdot \beta_{sp} \cdot N \cdot \delta(t - t') \tag{5}$$

The spontaneous emission factor β_{sp}, represents the number of spontaneous emission events that couples with the lasing mode. The noise term in the carrier equation $F_N(t)$, coming from spontaneous emission as well as shot noise contribution, is generally small and thus usually neglected.

Semiconductor lasers are very sensitive to external optical light. Even small external reflections and perturbations may result in sufficient sources of unstable operating behavior [73,101]. For this reason almost all types of commercial semiconductor lasers that apply to the standard telecommunications systems are provided with an optical isolation stage that suppresses severely optical perturbations by the external environment. However, in applications – such as chaos communications – where the raise of instabilities plays a key role, the isolation stage is omitted and the semiconductor lasers are driven intentionally to unstable operation.
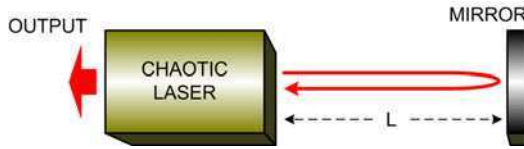
Fig. 2 Block diagram of a laser subjected to optical feedback.

Semiconductor lasers with applied optical feedback are very interesting configurations not only from the viewpoint of nonlinear dynamics they exhibit, but also for their potential for applications. Optical feedback is practically the process in which a small part of the laser's output field reflected by a mirror in distance L is re-injected into the laser's active region (fig. 2). The optical feedback system is a phase-sensitive delayed-feedback autonomous system for which all three known routes, namely, period-doubling, quasi-periodicity, and route to chaos through intermittency can be found. Many lasers exhibit the same or similar dynamics: edge-emitting semiconductor lasers such as Fabry–Perot, MQW, and DFB lasers exhibit similar chaotic dynamics, even though the parameter ranges for achieving the specific dynamics may differ. The measure of the feedback strength is usually expressed in literature by the C parameter [3]:

$$C = \frac{k_f T}{t_{in}} \sqrt{1 + a^2} \tag{6}$$

where k_f is the feedback fraction, $T = 2L/c_g$ is the round-trip time for light in the external cavity, c_g is the speed of light within the medium of the external cavity and L is the distance between the laser facet and the external mirror, α is the linewidth-enhancement factor that plays an important role in semiconductor lasers, and t_{in} is the round-trip time of light in the internal laser cavity. A semiconductor laser with optical feedback shows various dynamic behaviors depending on the system parameters and the instabilities of the laser [92]. For insignificant feedback fractions of the laser's field amplitude (up to 0.01%) the laser maintains its continuous wave operation. By increasing the feedback fraction to values up to 0.1% and at the same time $C>1$ generation of external cavity modes gives rise to mode hopping among internal and external modes. For a narrow region around 0.1% feedback (depending on laser dynamics) the mode-hopping noise becomes suppressed and the laser may oscillate with a narrowed linewidth. Increasing to moderate or strong feedback values (around 1% to 10%) the relaxation oscillation becomes undamped and the laser linewidth is greatly broadened to GHz bandwidth. It is then when the laser shows chaotic behavior and evolves into unstable oscillations in the so called *"coherence collapse"* regime. Finally, in extremely strong feedback regimes, usually defined for a feedback ratio higher than 10%, the internal and external cavities behave like a single cavity and the laser oscillates in a single mode. The linewidth of the laser in the case is narrowed greatly. The investigated dynamics were considered for a DFB laser with an emitting wavelength of 1.55 μm thus that above regions may be of consistency for other types of lasers for

slightly different values of the feedback fraction. However the dynamics for other lasers show always similar trends.

For chaos applications in communications, the coherence collapse regime is of great significance since it is where the laser generates chaotic dynamics. A semiconductor laser with optical feedback for this regime is modeled by the Lang–Kobayashi equations [3,51,57] that include the optical feedback effects in the laser rate equations model. When $C>1$, many modes for possible laser oscillations are generated, and the laser becomes unstable. The instabilities of semiconductor lasers depend on the number of excited modes or equivalently the value of C. The stability and instability of the laser oscillations have been theoretically studied in numerous works by the linear stability analysis around the stationary solutions for the laser variables [71,95].

The dynamics of semiconductor lasers with optical feedback depend on the system parameters; the key parameters which can be controlled are the feedback strength k_f, the length of the external cavity L formed between the front facet of the laser and the external mirror, as well as the bias injection current I. For variation of the external mirror reflectivity, the laser exhibits a typical chaotic bifurcation very similar to a Hopf bifurcation; however the route to chaos depends on the above crucial parameters [73]. Another type of instabilities produced by applying optical feedback is sudden power dropouts and gradual power recovery in the laser output power, the so-called *"low frequency fluctuations"* (LFFs) [28,47,70,75,84]. LFFs are typical phenomena observed in a low bias injection current condition, just above the threshold current of the laser. Usually this type of carrier consists of frequencies up to one GHz at maximum, thus message encryption could be applied only for such a limited bandwidth.

On the other hand, the spectral distribution of the chaotic carrier depends on the relaxation oscillation frequency of a semiconductor laser, which is directly determined by the biasing current of the laser. By increasing the optical feedback the chaotic carrier expands beyond the relaxation frequency of the laser, eventuating in a broadband fully developed chaotic carrier that may expand up to several tens of GHz. It has also been proved so far that the laser oscillates stably for a higher bias injection current. Thus, larger optical feedback strength is usually required to destabilize the laser at a higher bias injection current. The external cavity length also plays an important role in the chaotic dynamics of semiconductor lasers. There are several important scales for the length and change of the external mirror in the dynamics. Chaotic dynamics may be observed even for a small change of the external mirror position comparable to the optical wavelength λ [44]. For a small change, the laser output shows periodic undulations (with a period of $\lambda/2$) and exhibits a chaotic bifurcation within this period. When the external reflector is a phase-conjugate mirror, the phase is locked to a fixed value and the laser appears to be insensitive to small changes in the external cavity length and its dynamics are only defined by the absolute position of the external mirror [71]. This is observed for every external mirror position as far as the coupling between the external and internal optical field is coherent. When the external mirror is positioned within the distance corresponding to the relaxation oscillation frequency (on the

order of several centimeters) and the mirror moves within a range of millimeters, the coupling between the internal and external fields is strong and the laser shows a stable oscillation. When the external mirror is positioned over a distance equivalent to the relaxation oscillation frequency of a laser but it is within the coherence length of the laser (on the order of centimeter to several meters), the laser is greatly affected by the external optical feedback. In this region, the number of modes related to the *C* parameter is large and the laser shows various dynamical behaviors even at moderate feedback rate [44]. This region is also important for the study of fundamental dynamics and their applications, since external feedback length of many practical systems is on the order of several to tens of centimeters and providing chaos generating devices for various applications. Finally, when the external mirror is positioned at a distance beyond the coherence length of the semiconductor laser, it still exhibits chaotic oscillations, but the effects have a partially coherent or incoherent origin [89]. Instabilities and chaos generation are also induced by this type of incoherent feedback, which can originate not only from the laser itself but also from optical injection from another laser source.

Considering the case of a relatively weak optical feedback, the rate equation (1) that describes the semiconductor laser electric field can be altered appropriately in order to describe also the external cavity. In the case of single-longitudinal mode operation and application of a weak optical feedback condition the evolution of the field is governed now by the following equation:

$$\frac{dE(t)}{dt} = \frac{1-ia}{2} \cdot \left[G(t) - t_p^{-1}\right] \cdot E(t) + k_f \cdot E(t-T) \cdot e^{i\omega_0 T} + F_E(t) \qquad (7)$$

The carrier equation (2) does not need any modification with respect to the free-running case. This basic equation that includes the applied optical feedback was introduced by Lang and Kobayashi in 1980 [51]. From the mathematical point of view a delay term in a differential equation yield an infinite dimensional phase-space, since a function, defined over a continuous interval [0,*T*] has to be specified as initial condition. The understanding of delayed feedback systems has been boosted during the last years using semiconductor lasers. Fundamental nonlinear dynamical phenomena, such as, period doubling and quasi-periodic route to chaos have been characterized in these systems. Also high-dimensional chaotic attractors have been identified. Furthermore the analogy between delay differential equations and one-dimensional spatial extended systems have been established [33] and exploited for the characterization of the chaotic regimes [65].
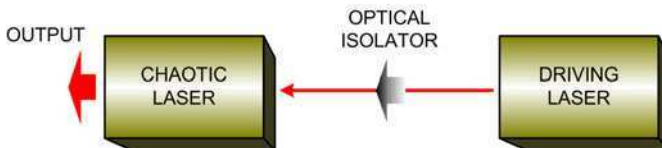

Fig. 3 Block diagram of a laser subjected to optical injection by a second driving laser.

However, there are alternative ways to cause instabilities and chaos dynamics in semiconductor lasers, besides optical feedback. In an optical injection system (fig. 3), the optical output of an independent driving laser is fed into the laser of importance in order to destabilize it and under specific conditions force it to oscillate in the chaotic regime [57,87]. Crucial parameters that determine the latter operation are the optical injection strength of the optical field – with values that are adequate to achieve injection locking condition – and the frequency detuning between the two lasers – which is usually below the region of ±10GHz. Compared to the rate equations for the solitary laser, an additional term representing the injection field from the driving laser is added to the field equation. This modification completely changes the dynamics of the system by adding one more dimension. In this case of a weak optical injection condition the evolution of the field is modified accordingly:

$$\frac{dE(t)}{dt} = \frac{1-ia}{2} \cdot \left[G(t) - t_p^{-1} \right] \cdot E(t) + k_{dr} \cdot E_{ext}(t) + F_E(t) \tag{8}$$

where k_{dr} is the coupling coefficient of the driving laser to the master laser and E_{ext} is the injected electrical field of the driving laser. In contradiction to the optical feedback case, in which the time-delayed differential equations provide infinite degrees of freedom, optical injection provides low-complexity attractors.
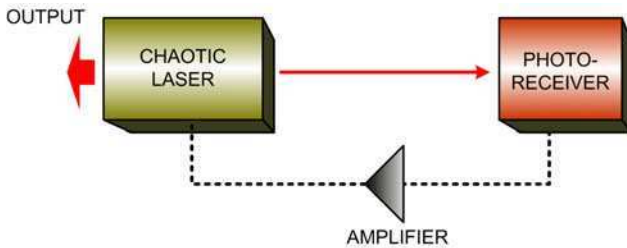

Fig. 4 Block diagram of a laser subjected to optoelectronic feedback.

A semiconductor laser with an applied delayed optoelectronic feedback loop is also an efficient technique of broadband chaos generation [90]. In such a configuration, a combination of photodetector and a broadband electrical amplifier is used to convert the optical output of the laser into an electrical signal that is fed back through an electrical loop to the laser by adding it to the injection current (fig.4). Since the photodetector responds only to the intensity of the laser output, the feedback signal contains the information on the variations of the laser intensity disregarding any phase information. Therefore, the phase of the laser field is not part of the feedback loop dynamics and consequently the dynamics of this system. The fact that part of the feedback loop is an electric path, the bandwidth response of this path may provide a filtered feedback. This can be justified by the limited bandwidth of the photoreceiver, the electric amplifier, as well as the electric cables. Additionally, an electrical filter may be also incorporated within this path,

with a pre-selected transfer function and bandwidth, enhancing thus the number of parameters that determine the final form of the generated chaotic output.

2.2 Non-linear Dynamics in Photonic Integrated Circuits

Based on the above chaos generation techniques various configurations of transmitters have been proposed and implemented based on standard optical components, providing high-dimensional chaotic carriers capable of message encryption.

The miniaturization of the above configurations through photonic integration appears very attractive, albeit scarce, considering the efficiency of specifically designed photonic integrated circuits (PICs) to generate non-linear dynamics. In [30] monolithic colliding pulse mode-locked lasers exhibited nonlinear behavior, from CW operation to self-pulsations and mode-locking, for the full range of the control parameters. In [16] a semiconductor laser, followed by a phase section and an active feedback element, form a very short complex photonic circuit that provides several types of dynamics and bifurcations under optical feedback strength and phase control. However, only multiple-mode beating operation may transit the dynamics beyond a quasi-periodic route to chaos with possible chaotic components. A simplified version of the aforementioned PIC, omitting though the active feedback element, was found to generate only distinct-frequency self-pulsations [100]. The implementation of an integrated colliding-pulse mode-locked semiconductor laser showed also nonlinear dynamics and low-frequency chaos, by controlling appropriately only the laser's injection current [108]. Finally, a photonic integrated circuit (fig. 5), capable to generate high-dimensional broadband chaos has also been recently proposed, designed and tested [11]. It consists of four successive sections: a DFB InGaAsP semiconductor laser, a Gain/Absorption section, a phase section and a 1cm long passive waveguide. The overall resonator length is defined by the internal laser facet and the chip facet of the waveguide which is highly reflective coated and provides an increased effective feedback round-trip time, therefore enhancing the probability to encounter fully chaotic behavior. The dynamics generated by this PIC could be easily controlled via phase conditions and feedback strength, establishing therefore this device as a compact integrated fully-controllable chaos emitter.
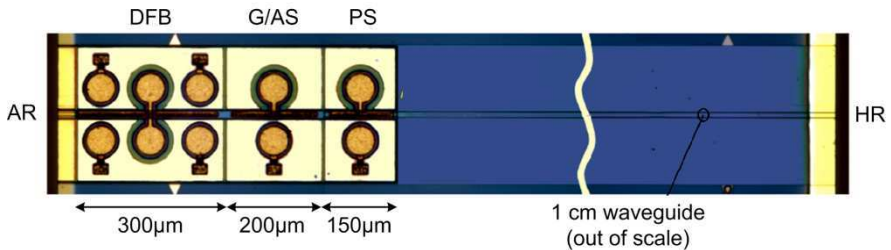

Fig. 5 Photonic integrated chaos-emitter for secure optical communications applications. DFB: Distributed feedback laser, G/AS: Gain – absorption section, PS: Phase section, AR: Anti-reflective coating, HR: Highly reflective coating.

3   Chaos Synchronization in Transmitter/Receiver Configurations

When exploiting chaotic carriers in communication systems, these carriers should be finally vanished after transmission, in order to retain the information. Thus, a crucial process is the carrier reproduction at the receiver. As discussed earlier, this process is achieved through *chaotic carrier synchronization*.

In chaos synchronization, when semiconductor lasers are employed as chaos generators, the dynamical variables used for the driving signal are not always separable from others and some are simply not extractable from a laser. When the output field of the master laser is transmitted and coupled to the slave laser, both its magnitude and phase contribute to the receiver's chaos generation. It is not possible to only transmit and couple the magnitude but not the phase, or only the phase but not the magnitude. Thus, for optical injection and optical feedback systems, the frequency, phase, and amplitude of the optical fields of both transmitter and receiver lasers are all locked in synchronism. Therefore, unless the phase is not part of the dynamics of the lasers, such as in the case of systems with optoelectronic feedback, the synchronization between two laser systems depends on the coupling of the two variables, the magnitude and phase of the laser field, at the same time. Furthermore, the carrier density is not directly accessible externally and therefore cannot be used as a driving signal to couple lasers. However, in laser systems that exhibit chaos dynamics, not only master–slave configurations but also mutually injected systems [31] can be used for chaos synchronization systems. The latter are not suited for chaos communications and thus are beyond the scope of the present analysis. Another issue that is of great interest but will be dealt in next paragraphs is the fact that for a synchronized chaotic communication system, the message encoding process – whatever this is – may have a significant impact on the quality of synchronization and thus on the message recoverability at the receiver end. It has been shown that high-quality synchronization can be maintained only when proper encoding schemes that maintain the symmetry between the transmitter and the receiver is employed. Additionally, in transmission systems, the transmission impairments should be minimized in order not to disturb the synchronization process.

3.1   *Chaos Synchronization of Semiconductor Lasers with Optical Feedback*

The condition that should always be satisfied for synchronizing chaotic waveforms – that are produced by two nonlinear systems – is that the deviations of the corresponding parameters that characterize each system must be insignificant. Practically, two categories of chaotic configurations of all-optical systems have been developed for efficient synchronization based on their robustness (fig. 6) [74,103]. The first one consists of two identical external-cavity semiconductor lasers for the transmitter and the receiver respectively (closed-loop scheme), while in the second approach, an external-cavity laser transmitter produces the chaotic car-

rier and a single laser diode similar to the transmitter is used as the receiver (open-loop scheme) [22,59,74,103]. The closed-loop scheme proves to be more robust in terms of synchronization; however it requires precise matching of the external cavity of the lasers to maintain a good synchronization quality [22,103]. On the contrary, the open-loop scheme is less robust with simpler receiver architecture [22,74,103]. It requires a large coupling strength between the transmitter and the receiver, however, there is no requirement of perfectly matched lasers' and there is no external-cavity receiver to be matched to that of the transmitter.
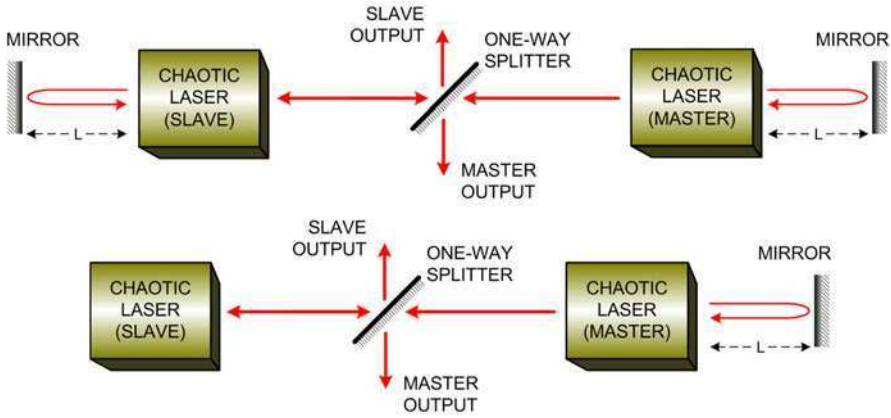

Fig. 6 Block diagram of: a closed-loop synchronization configuration between two semiconductor lasers both subjected to optical feedback (top), and an open-loop synchronization configuration between two semiconductor lasers, with only the master laser being subjected to optical feedback (bottom).

The rate equations that describe the coupled behavior between a transmitter and a receiver, based on the Lang-Kobayashi model are:

$$\frac{dE_i(t)}{dt} = \frac{1-ia}{2} \cdot \left[G_i(t) - t_{p,i}^{-1} \right] \cdot E_i(t) + k_{f,i} \cdot E_i(t-T) \cdot e^{i\omega_0 T} + k_{inj} \cdot E_{ext}(t) + F_E(t) \quad (9)$$

$$\frac{dN_i(t)}{dt} = \frac{I}{e} + \frac{N_i(t)}{t_{n,i}} - G_i(t) \cdot |E_i(t)| + F_N(t) \quad (10)$$

$$G_i(t) = \frac{g \cdot (N_i(t) - N_{0,i})}{1 + s \cdot |E_i(t)|^2} \quad (11)$$

where $i=\{t,r\}$ denotes the solution for the transmitter or the receiver, k_{inj} is the electrical field injection parameter applied to the receiver laser and E_{ext} is the amplitude of the injected electric field. The term $k_{inj} \cdot E_{ext}(t)$ is applicable only in the rate equation of the receiver. For the case of open loop, no optical feedback is applied on the receiver, thus $k_{f,r}=0$.

3.2 *Types of Synchronization*

Following the form of the Lang-Kobayashi rate equations that describe the dynamical operation of the transmitter and the receiver, two different types of synchronous responses of the receiver have been distinguished, referring to the weak and the strong injection condition, respectively [61,74,103].

The first one is the *"complete chaos synchronization"* in which the rate equations, both for the transmitter and the receiver, are written by the same or equivalent equations. In complete chaos synchronization, the frequency detuning between the transmitter and receiver lasers must be almost zero and the other parameters must also be nearly identical [72]. This type of synchronization in semiconductor laser systems is realized when the optical injection fraction is small (typically less than a few percent of the chaotic intensity variations) [74,103]. The synchronized solution emerges from the mathematical identity of the equivalent equations that describe the operation of the emitter and the receiver. Thus, these systems can be considered as very secure from eavesdroppers in communications, since the constraints on the parameter mismatches are very severe. The time lag that exists in this type of synchronization is defined by the propagation time between the transmitter and the receiver, as well as the roundtrip time of the transmitter's external cavity. The conditions under which the rate equations for the receiver laser are mathematically described by the equivalent delay differential equations as those for the transmitter laser are the following:

$$E_r(t) = E_t(t+T) \tag{12}$$

$$N_r(t) = N_t(t+T) \tag{13}$$

$$k_r = k_t - k_{inj} \tag{14}$$

Specifically, the receiver laser anticipates the chaotic output of the transmitter and it outputs the chaotic signal in advance with time as understood from (12), so that the scheme is also called *"anticipating chaos synchronization"* [66,88]. This type of synchronization is not easy to be implemented in real-world chaos communication systems that employ transmission. Various impairments such as amplification noise and chromatic dispersion are synchronization error sources, even if identical chaotic oscillators are chosen.

In the case of a much stronger injection (typically over 10% of the laser's electric field amplitude fluctuations), another type of synchronization is achieved, based on a driven response of the receiver to the transmitter's chaotic oscillations, called *"isochronous chaos synchronization"* [60,62,74,103]. An optically injected laser in the receiver system will synchronize with the transmitter laser based on the *optical injection locking or amplification effect*. The optical injection locking phenomenon in semiconductor lasers depends on the detuning between the frequencies of the master and slave lasers. In general, it is not easy to set the oscillation frequencies between the transmitter and receiver lasers exactly the same and a frequency detuning inevitably occurs. However, there exists a frequency pulling effect in the master-slave configuration as long as the detuning is small and the

receiver laser shows a synchronous oscillation with the transmitter laser. This has been recently observed for a wide range of frequency detuning between the transmitter and the receiver [60]. The time lag of the synchronization process is now equal to the propagation time only – which is, in most cases, considered to be zero in simulations for simplicity reasons – thus there is no need for a well-defined roundtrip time of the transmitter's external cavity. Generally, this type of synchronization is characterized by a tolerance to laser parameter mismatches and consequently it can be more easily observed in experimental conditions [60]. The relation between the electric fields of the two lasers in this type of synchronization is written as in [31]:

$$E_r(t) = A \cdot E_t(t) \tag{15}$$

The receiver laser responds immediately to the received chaotic signal from the transmitter, with amplitude multiplied by an amplification factor A. This scheme is sometimes also called *"generalized chaos synchronization"*. Most experimental results in laser systems including semiconductor lasers reported up to now were based on this type of chaos synchronization. However, in the final stage at the receiver where message recovery is the key, the cancellation of the above chaotic carriers in a communication configuration can be easily performed by attenuating the output of the receiver laser by the same amount of the amplification factor A.

3.3 Measuring Synchronization

The most common approaches to quantitively measure the synchronization quality of a chaotic system are the *synchronization error σ* and the *correlation coefficient C_{corr}*. The synchronization error between the transmitter and the receiver chaotic outputs is defined as [2,58,59]:

$$\sigma = \frac{\left\langle \left| P_t(t) - P_r(t) \right| \right\rangle}{\left\langle \left| P_t(t) \right| \right\rangle} \tag{16}$$

where $P_t(t)$ and $P_r(t)$ are the optical powers of the output waveforms of the transmitter and the receiver respectively, in the linear scale (eg. mW). The averaging is performed in the time domain. Small values of σ indicate low synchronization error and thus high synchronization quality.

The correlation coefficient [58,62,91,103], on the other hand, is defined as:

$$C_{corr} = \frac{\left\langle \left[P_t(t) - \left\langle P_t(t) \right\rangle \right] \cdot \left[P_r(t) - \left\langle P_r(t) \right\rangle \right] \right\rangle}{\sqrt{\left\langle \left| P_t(t) - \left\langle P_t(t) \right\rangle \right|^2 \right\rangle} \cdot \sqrt{\left\langle \left| P_r(t) - \left\langle P_r(t) \right\rangle \right|^2 \right\rangle}} \tag{17}$$

where the notation is the same as before. The correlation coefficient values lie between $-1 \leq C_{corr} \leq 1$, so large values of $|C_{corr}|$ indicate high synchronization quality. In both these definitions it is assumed that there is no time lag between the chaotic outputs of the transmitter and the receiver, which means that the time-traces must be temporally aligned before estimating the synchronization quality, as seen in
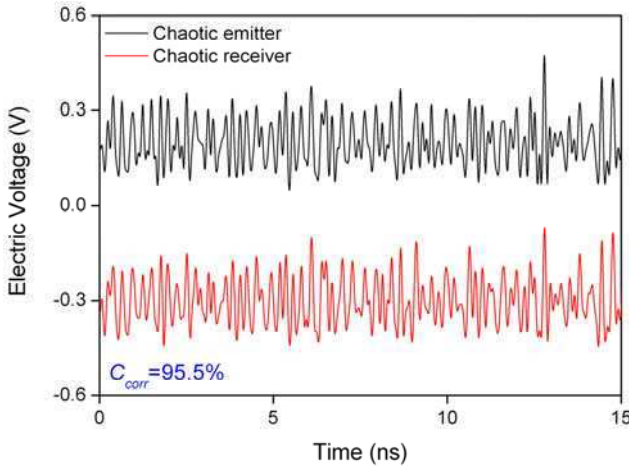
Fig. 7 Synchronized experimental output time-traces of a chaotic emitter and its matched receiver in an open-loop system based on the isochronous solution (time-series of receiver is vertically shifted for viewing purposes). The correlation coefficient is estimated to be equal to 95.5%, for the case of $k_{inj}=4 \cdot k_{f,t}$.

fig. 7. The latter is of great importance, since different types of synchronization (generalized and anticipating) correspond to different time lags between the chaotic carriers [50,59,63,74].

In all the theoretical works presented so far that use the Lang-Kobayashi approach are based on the time evolution of semiconductor laser non-linear dynamics, the time step of the numerical methods implemented to simulate this model is usually as small as 10^{-13} s, in order for the numerical methods to converge. Since the bandwidth of chaotic carriers usually may extend up to few tens of GHz, all the spectral content of the carriers is included in the time-series data by using such a time step.

However in the hardware implemented chaotic communication systems, the measurements and the recording of chaotic waveforms in the time domain are performed with oscilloscopes of limited bandwidth. Additionally, many of the components used in such systems (photoreceivers, filters, etc.) have a limited-bandwidth spectral response profile. Consequently, an alternative approach of measuring the synchronization error of a chaotic communication system is by transforming eq. (16) to the spectral domain. By subtracting the transmitter's and the receiver's chaotic spectra in a certain bandwidth Δf, we also get a quantitive estimation of the synchronization quality of the system, the spectral synchronization error $\sigma_{\Delta f}$ [9]:

$$\sigma_{\Delta f} = \frac{\left\langle \left| P_t(f) - P_r(f) \right| \right\rangle}{\left\langle \left| P_t(f) \right| \right\rangle} \Bigg|_{\Delta f} \tag{18}$$

where $P_t(f)$ and $P_r(f)$ are the optical power values of the chaotic carriers in the linear scale (mW), at frequency f and the averaging is performed in the frequency

domain. Eq. (18) provides additional information, since the synchronization error measured is associated with the spectral bandwidth Δf. In this case, one could constrain the synchronization study of the system only in the above spectral region of importance. For example, if 1Gb/s message bit sequences are to be encrypted in a baseband modulation format, the region of the first GHz is of great importance in terms of synchronization, since the rest of the spectral components of the carrier will be filtered in the final message recovery process. As emerges from different systems when studying the synchronization properties of chaotic carriers, the synchronization efficiency is different for the various frequencies of the carrier. For example, in a system that generates broadband chaos dynamics, there might be conditions that provide a very good synchronization in the low frequency region and beyond that only poor synchronization efficiency; however, using different conditions, one might achieve – in the same system – a moderate synchronization performance for the whole spectral bandwidth. Thus, a more suitable form of (18) when dealing with experimentally taken data in the spectral domain is the logarithmical transformation of the synchronization $\sigma_{\Delta f}$ that could also be defined as chaotic carrier *"optical cancellation $c_{\Delta f}$"*[9]:

$$c_{\Delta f}(dB) = -10 \log \sigma_{\Delta f} \qquad (19)$$

$$c_{\Delta f}(dB) = \left\langle \left| P_t(f) \right| \right\rangle \Big|_{\Delta f}(dBm) - \left\langle \left| P_t(f) - P_r(f) \right| \right\rangle \Big|_{\Delta f}(dBm) \qquad (20)$$

Eq. (18) gives practically the difference between the mean optical power of the transmitter and the subtraction signal, measured in a logarithmical scale (dBm) and in a specific spectral bandwidth Δf.

When dealing with electrical powers of the above signals eqs. (19) and (20) are transformed accordingly, following the square law dependence that describes the relationship in a photodetector's signal between its electrical and optical power, providing the chaotic carrier *"electrical cancellation $c_{\Delta f}^E$"* [9]:

$$c_{\Delta f}^E(dB) = -20 \log \sigma_{\Delta f} \qquad (21)$$

$$c_{\Delta f}^E(dB) = \left\langle \left| P_t^E(f) \right| \right\rangle \Big|_{\Delta f}(dBm) - \left\langle \left| P_t^E(f) - P_r^E(f) \right| \right\rangle \Big|_{\Delta f}(dBm) \qquad (22)$$

$P_t^E(f)$ and $P_r^E(f)$ are the electrical power values of the transmitter and the receiver output in a specific frequency f. The averaging in (1.20) and (1.22) is performed in the frequency domain and refers to the frequency bandwidth Δf.

4   Chaos-Secured Optical Communication Systems

Several hardware configurations have been presented so far that employ a satisfactory synchronization process between chaotic emitters and receivers capable of Gb/s data encryption and decryption. What is remarkable about the chaos optical communication systems deployed so far is that they use commercially available optical telecommunication components and technology, they can operate at data rates up to 10 Gbit/s and can be feasibly integrated into existing underground sys-

tems, as well as become upgraded at any time without altering the optical network infrastructure. Such systems, built in the last years, have been based on utilizing er-bium-doped fiber amplifiers (EDFAs), semiconductor lasers or electro-optic mod-ulators as the non-linear optical component for chaos optical generation. Addition-ally, some of these systems have been test in real-world environment in field trial tests. In this section, the encoding techniques that have been used in such systems are described, while the most important demonstrations are also analyzed.

4.1 Data Encoding Techniques

A very important issue that should be decided when building chaos-secured opti-cal communication systems is the data encryption method that will be adopted. The major concern in all encoding schemes is not to disturb the synchronization process, since the encrypted message is always an unwanted perturbation in the fragile synchronized system. All schemes used so far differ in the way the mes-sage is encoded within the chaotic carrier, although the decoding process is the same for all schemes, based on subtracting the output of the receiver's laser from the received signal. Some of these methods are described below.

4.1.1 Additive Chaos Modulation (ACM)

In the ACM encoding method, data $m(t)$ are applied by externally modulating the electric field E_{TR} of the chaotic carrier generated by the emitter laser, according to the expression:

$$E_{TR} = (1 + m(t)) \cdot E_M \cdot e^{i\phi_M} \qquad (23)$$

resembling the typical coherent amplitude modulation (AM) scheme [37,76,107]. The phase ϕ_M of the chaotic carrier with the message encoded on it, part of which is injected to the receiver for the synchronization process, is the same with that of the chaotic carrier without any information. Thus, the presence of the encrypted message on the chaotic carrier is only a small perturbation in amplitude and not in phase which turns to be crucial for the efficient synchronization process of a phase-dependent system. In such a case, the modulated signal does not contribute or alters the chaotic dynamics of the transmitter.

4.1.2 Multiplicative Chaos Modulation (MCM)

In MCM encoding method, the message is also applied by external modulation; however the modulation is applied within the external cavity of the transmitter, providing a message-dependent chaotic carrier generation process.

4.1.3 Chaos Masking (CMS)

In the CMS encoding method, the message is applied on an independent optical carrier which is masked by the chaotic optical carrier [48]. Both carriers should correspond to exactly the same wavelength, the same polarization state and the message carrier should be suppressed enough in respect to the chaotic carrier in

order to ensure an efficient message encryption. Since now the message is a totally independent electric field which is added to the chaotic carrier according to the expression:

$$E_M \cdot e^{i\phi_M} + E_{msg} \cdot e^{i\phi_{msg}} \qquad (24)$$

The phase of the total electric field injected now to the receiver consists of two independent components. The phase of the message ϕ_{msg} acts, in this case, as a perturbation in the phase-matching condition of a well-synchronized system. The above phase mismatch in the CMS method results in a significant perturbation in the synchronization process and for this reason this encoding method is not considered efficient.

4.1.4 Chaos Shift Keying (CSK)

In the CSK method, the bias current of the emitter's laser is modulated resulting two different states of the same attractor associated to the two levels of the biasing current [69,76]. The current of the emitter's laser is given by the equation:

$$I_M = I_B + m(t) \cdot I_{msg} \qquad (25)$$

with $m(t) = \frac{1}{2}$ (or $-\frac{1}{2}$) for a "1" (or a "0") bit and $I_B >> I_{msg}$, in order to keep the synchronization error in small values.

4.1.5 Phase Shift Keying (PSK)

The strong dependence of synchronization on the relative phase between the external cavities of ML and SL, may be also employed for message encoding [6]. Indeed, a phase variation of the ML external cavity, which is small enough to be undetectable by observation of the chaotic waveform or of its spectrum, can substantially affect the correlation between the two laser outputs. Thus, if the ML phase is modulated by a message, the latter can be extracted by transferring the induced variation of the correlation coefficient into amplitude modulation. This can be easily done by taking the difference between the phase-modulated chaotic waveform coming from the transmitter and the chaotic waveform from the receiver, as in the standard masking scheme.

4.1.6 Sub-Carrier Chaos Encryption

All the above encoding techniques referred to baseband encryption within the chaotic carriers. However, the power spectral distribution of a generated chaotic carrier is not always suitable for base-band message encryption. For example, in cases where short external optical cavities are employed the most powerful spectral components of the carrier arise on the external cavity mode (ECM) frequencies. By applying sub-carrier message encryption in those frequencies where the chaotic carrier has powerful spectral components, a higher signal-to-noise ratio of the encrypted signal may be applied without any compromises in the quality of encryption, providing a better message recovery performance in comparison to the base-band techniques [12,18].

4.2 *EDFA-Based Chaotic Optical Communication Systems*

The pioneering chaotic laser system, developed by Van Wiggeren and Roy, employed chaotic carriers with bandwidth around 100 MHz, which yields a data rate comparable with that used in radio-frequency communications [102]. In the proposed configuration the transmitter consisted of a fiber ring laser that included an EDFA (fig. 8). The optical signal that was generated by an EDFA was re-injected into the same EDFA after circling the fiber ring. In such a configuration, the fiber laser is driven by its own output but at some time delay leading finally to a chaotic behavior. This type of response is common to time-delayed dynamical systems of any kind. The message to be transmitted was another optical signal coupled into the fiber ring of the transmitter, and was injected into the laser together with the time-delayed laser signal. In such way, the information signal also drives the laser and thus it becomes mixed with the dynamics of the transmitter. As the combined information/laser signal travels around the transmitter ring, part of it is extracted and transmitted to the receiver. At the receiver the signal is split into two parts. One part is fed into an EDFA almost identical to the one used in the transmitter, which ensures that the signal is synchronized with the dynamics of the ring-fiber laser in the transmitter. Then it is converted into an electrical signal by a photodiode, providing a duplicate of the pure laser signal at some time delay. The other part was fed directly into another photodiode providing a duplicate of the laser-plus-information signal. After taking account of the time delays, the chaotic laser signal is subtracted from the signal containing the information, removing the chaos and leaving the initial message. In this work the information applied was a 10MHz square wave and finally the signal decoded by the receiver matched well the transmitted one.
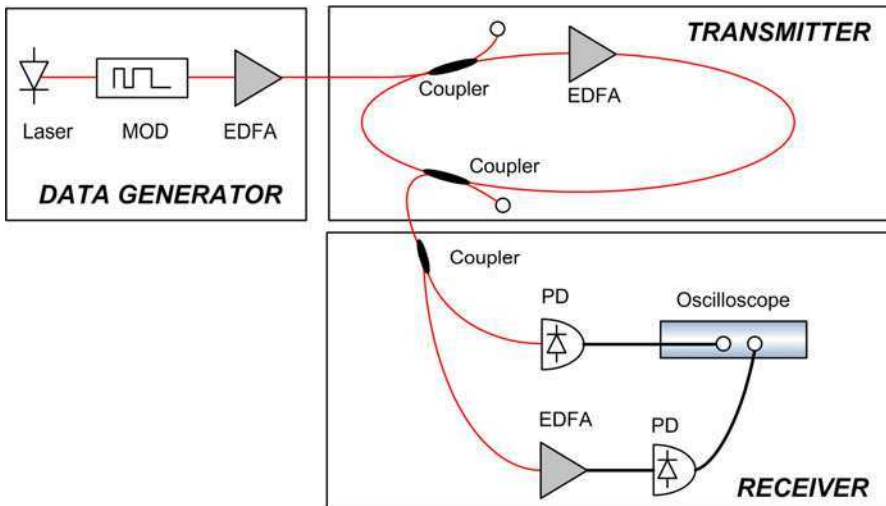

Fig. 8 The optical chaos communication setup proposed by Van Wiggeren and Roy.

4.3 All-Optical Chaotic Optical Communication Systems

In more recent works, researchers have increased the bandwidth of the chaotic carriers as well as the encrypted message bit rates. In [49] a sinusoidal message transmission up to 1.5 GHz was performed based on synchronization of chaos in experimental nonlinear systems of semiconductor lasers with optical feedback. The message is almost entirely suppressed in the receiver output, even if the message has non-negligible power in the transmitter. Also in [24], encoding, transmission and decoding a of a 3.5GHz sinusoidal message in an external-cavity chaotic optical communication scheme operating at 1550 nm has been demonstrated. Beyond these preliminary communication setups that employed only sinusoidal carriers to prove the principle of operation of the chaos communication fundamentals, contemporary optical chaotic systems tested with pseudorandom bit sequences have been recently demonstrated, also implying the feasibility of this encryption method to secure high-bit rate optical links. The latter are in principle more demanding in synchronization efficiency, since the chaotic carriers should be proficiently synchronized not only in a single frequency but in a wider spectral region – the one that covers the encrypted message. Such systems are presented in the following paragraphs.
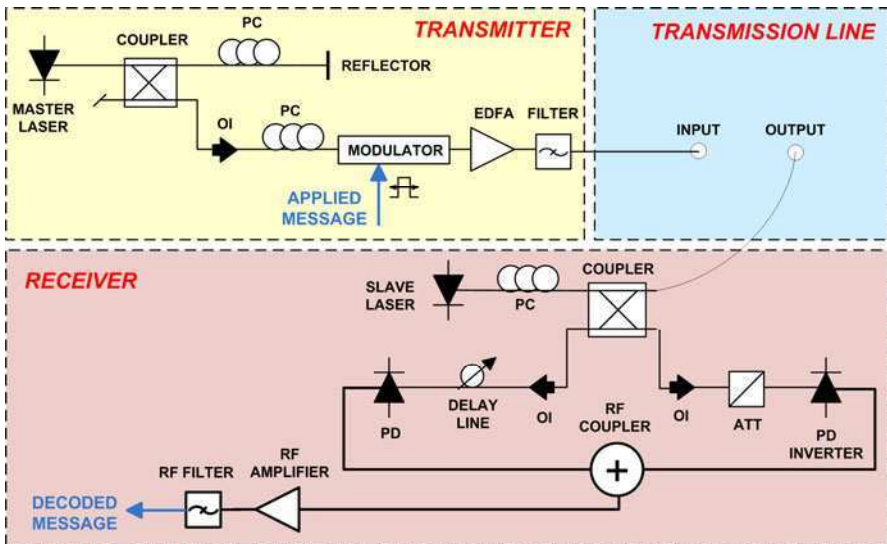

Fig. 9 Experimental setup of an all-optical communication transmission system based on chaotic carriers. PC: polarization controller, OI: optical isolator, PD: photoreceiver, ATT: attenuator.

An all-optical chaotic communication system built on the concept of an open-loop receiver is shown in fig. 9 and is the basis of the system presented in [10] by Argyris et al.. In this work, two distributed feedback semiconductor lasers (DFB) that were neighboring chips in the same fabrication wafer with almost identical characteristics have been selected as the transmitter and the receiver lasers. Both

lasers emitted at the same exactly wavelength of 1552.1nm. The biasing of the lasers close to their threshold value ensured a powerful chaotic carrier even at the low frequency regime, guarantying a sufficient encryption of the baseband message. The chaotic carrier was generated within a 6m long fiber-optic external cavity formed between the master laser and a digital variable reflector that determined the amount of optical feedback sent back. In the specific experiment the feedback ratio was set to moderate values (up to a few percent) in order to be efficient to cause broadband chaos dynamics. Polarization alignment was very significant through the whole system. The data encrypted in this configuration were non-return-to-zero pseudorandom sequences with small amplitudes and code lengths up to 2^{31}-1 and were applied by externally modulating the chaotic carrier using a Mach-Zehnder LiNbO$_3$ modulator (ACM encoding technique).

In this work, the chaotic optical carrier with the encrypted data was optically amplified and transmitted through a total length of 100km fiber span. The appropriate dispersion compensation fiber modules were used in order to eliminate the chromatic dispersion, while two EDFAs were used to compensate the transmission losses. Optical filtering was applied after EDFA modules in order to reject most of the amplified spontaneous emission (ASE) noise of the EDFA.

At the receiver's side, the synchronization process and the message extraction took place. The transmitted output was unidirectionally injected into the slave laser, in order to force the latter to synchronize and reproduce the emitter's chaotic waveform. The appropriate optical power of the injected signal into the receiver's laser was several times the optical feedback of the emitter. Generally, lower values of optical injection power prove to be insufficient to force the receiver to synchronize satisfactorily, while higher values of injection power lead to reproduction not only of the chaotic carrier but of the message too. The chaotic waveforms of the transmitter and the receiver were detected by two fast photodetectors that converted the optical input into electronic signal. The receiver's signal photodetector included a π-phase shift to the electrical output related to the optical one. Consequently, by combining with a microwave coupler the two electrical chaotic signals – the transmitter's output and the inverted receiver's output – an effective subtraction is actually carried out. In the transmitter's optical path an optical variable attenuator was used to achieve equal optical power between the two outputs, while a variable optical delay line in the receiver's optical path determined temporal alignment of both signal waveforms. The subtraction product from such a system is the amplified message, along with the residual high-frequency components of chaotic carrier which are eventually rejected by an electrical filter of the appropriate bandwidth.

The encryption quality is determined for a given bit-rate by the message amplitude. Its value is set so that the filtered encrypted message at any point of the link and at the receiver's input has a very high BER value (in this work no less than $6 \cdot 10^{-2}$). The BER value measured for the recovered message was 10^{-7}, for a message bit-rate of 0.8Gb/s and for the above encryption level. As the bit-rate is increased to a multi-Gb/s scale, the BER values are also increased monotonically. This is partially attributed to the filtering properties of the message at the receiver. The message filtering effect has been confirmed to be larger for lower frequencies and decreases as message spectral components approach the relaxation oscillation

frequency of the laser in the gigahertz regime, similar to the response of steady-state injection-locked lasers to small-signal modulation. Another important reason that justifies this limited performance is that the decoding process is based on signal subtraction and not on signal division, since only the former can be implemented with the traditional methods. The emitted signal is of the form of $[1+m(t)]\cdot E_t(t)$, while the receiver reproduces the chaotic carrier $E_r(t) = E_t(t)$. Thus, the output is not the encrypted message $m(t)$ but the product: $m(t)\cdot E_t(t)$. One should remember that the spectral components of the message are determined by the message bit-rate, while the chaotic carrier spectral components extend to tens of GHz. Thus, when applying low bit-rate messages, after the appropriate filtering, the received product contains the whole power of the message and only a small part of the carrier. By increasing the message bit-rate – and consequently the bandwidth of the received product – the proportion between the power of the chaotic carrier and the power of the message increases, deteriorating the final performance.
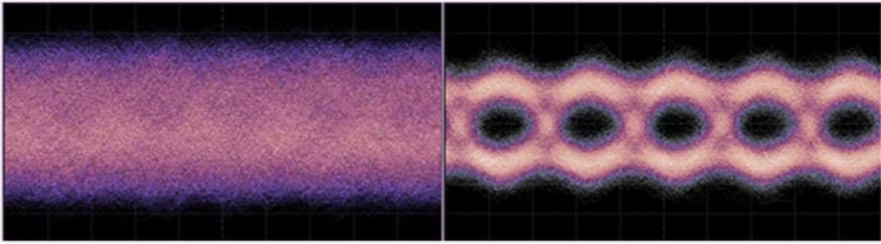

Fig. 10 Eye diagrams for a 2^{31}-1, 1Gb/s encrypted message with BER~$6\cdot10^{-2}$ (left) and decrypted message with BER~10^{-8} (right).

A follow-up of the above work was to use fully controllable broadband chaotic oscillators based on PICs. Such PICs were identified to be capable of exhibiting an outstanding performance in closed-loop synchronization architectures with extreme stability [13]. The philosophy of the presented communication system is twofold. On one hand, the emitter sends a completely hidden data stream within a chaotic carrier along the transmission link, and this is achieved – for a specific chaotic carrier – by restricting the message amplitude up to a maximum value. On the other hand, the receiver task is to cancel the chaotic carrier and recover data with a native bit-error rate as low as needed in order to acquire error rates below 10^{-12} using coding techniques. Practically, the BER improvement was achieved by an additional processing unit that employed a fast transceiver which enabled FEC methods. The reported results provided secure 2.5 Gb/s data exchange with bit-error-rates below 10^{-12}. Such low BER values were achieved only for authorized users, even for very small encrypted message amplitudes, through forward error correction (FEC) techniques [86,106]. The FEC method used in that system posed a digital bit-error-rate threshold (equal to $R=1.8\cdot10^{-3}$) in its operation, discriminating decisively the data recovery efficiency between authorized and unauthorized users. Thus unauthorized hardware receivers, either by direct detection of transmission line or by employing unmatched – compared to the PIC emitter –, will be able to recover data with BER values as low as this digital threshold R.

4.3.1 Single-Channel Transmission Field Demonstrators

Based on the above experimental configurations, the next step taken was to test such encryption systems in real world conditions, by sending chaos-encrypted data in a commercially available fiber network. Such an attempt was the transmission experiment published in [10]. The utilized infrastructure included an installed optical network of single mode fiber that covered the wider metropolitan area of Athens, Greece and had a total length of 120km. A dispersion compensation fiber (DCF) module, set at the beginning of the link (pre-compensation technique), canceled the chromatic dispersion induced by the single mode fiber transmission. Two amplification units that consist of erbium-doped fiber amplifiers and optical filters were used within the optical link for compensation of the optical losses and amplified spontaneous emission noise filtering, respectively.

The system's efficiency on the encryption and decryption performance was studied, as previously, by bit-error rate (BER) analysis of the encrypted/decoded message. The message amplitude was adjusted so that the BER values of the filtered encrypted message do not exceed in any case the value $6 \cdot 10^{-2}$, preventing any message extraction by linear filtering. In fig. 11 (right), spectra of the encrypted (upper trace) and the decrypted - after the transmission link - (lower trace), 1Gb/s message are shown. The good synchronization performance of the transmitter-receiver setup leads to an efficient chaotic carrier cancellation and hence to a satisfactory decoding process. The performance of the chaotic transmission system has been studied for different message bit rates up to 2.4Gb/s and for code lengths up to $2^{23}-1$ (fig. 11, left). All BER values have been measured after filtering the electric subtraction signal, by using low-pass filters with bandwidth adjusted each time to the message bit rate. For sub-gigahertz bit-rates the recovered message exhibits BER values always lower than 10^{-7}, while for higher bit-rates a relatively high increase was observed. This behavior characterized the back-to-back and the transmission setup, with relatively small differences in the BER values, revealing only a slight degradation of the system performance due to the transmission link.
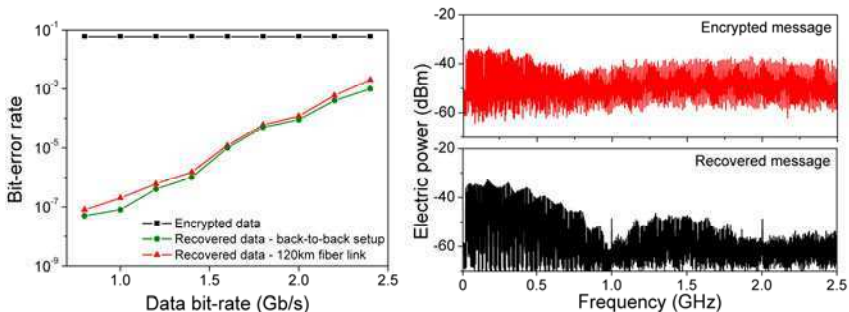

Fig. 11 Left: BER performance of encrypted (squares), back-to-back decoded (circles) and after transmission link decoded (triangles) message. Right: RF spectra of an encrypted and recovered 1 Gb/s pseudorandom message.

4.3.2 Multiple-Channel Transmission Field Demonstrators

The presented so far optical chaos communication systems have been demonstrated in single channel transmission experiments. However, contemporary optical networks consume the large bandwidth potential of fibers by supporting a large number of channels lying at distinct wavelengths densely spaced to each other. Preliminary studies have appeared to this direction [109] in order to numerically study the potential of operating in wavelength division multiplexed (WDM) transmission systems. This potential has been very recently deployed in chaos-encrypted installed systems with Gb/s data, including three neighboring channels according to the Telecommunication Standardization Sector (ITU-T) standards. The work presented in [14] was the first experimental evaluation of a dense-WDM (DWDM) communication system that includes chaotic optical channels with encrypted 1.25 Gb/s data streams, along with conventional channels that carry 10 Gb/s digital data sequences. The implementation of the chaos-secured communication channel was based on photonic monolithic integrated chaotic oscillators was on the basis of the system investigated in section 4.1, utilizing PICs [11,13]. The inclusion of additional channels in the installed transmission medium changes the performance of the chaotic channel's synchronization and decoding efficiency. Inter-channel interference is important for the decoding process of the chaos-encrypted channel. When the channel spacing was set to 0.8nm, the relative input polarization state of the neighboring channels affected significantly the decoding performance. When the polarization of each channel was set orthogonal to its neighbors, the interference effects were minimized. In the latter case, the recovery performance was identified to be almost equivalent to the back-to-back performance. On the contrary, if the polarization was set to be parallel to its neighbors, the decoding of the chaos-encrypted channel was downgraded significantly. In conclusion, chaotic communication channels that operated in a DWDM transmission environment were susceptible to transmission impairments. Cross-phase modulation played a significant role in the synchronization process of the chaos encrypted channel and was the factor that determined the final performance. Nevertheless, for wavelength spacing above 0.8nm, these impairments induce a tolerable penalty and allow the proficient operation of this channel, independently of the polarization states of the channels. If orthogonal polarization states of adjacent channels were chosen the above spacing tolerance was further reduced to 0.65nm.

4.4 *Optoelectronic Chaotic Optical Communication Systems*

An optoelectronic oscillator is a system in which the intensity of a continuous-wave laser is modulated by an electronically driven nonlinear optical device before being fed into a long length of optical fiber, which introduces delay. After propagation in the fiber, the delayed light is detected electronically, and this output is then used as an input to drive the modulation of the laser once again. The properties of the delayed output thus feed back into the system to modify the input, which in turn generates a new output and so on, in a periodic (oscillatory) manner. This is clearly not a simple system and, indeed, such oscillators exhibit a

wide range of rich dynamics. Several configurations have been implemented so far based on these optoelectronic oscillators, either encoding the message in amplitude or in phase. Such systems will be presented in the next paragraphs. [52]

4.4.1 Amplitude Data Encoding

In the system described by Gastaud et al. [32] the emitter was a laser diode whose output was modulated in a strongly nonlinear way by an electro-optic feedback loop through an integrated electro-optic Mach–Zehnder interferometer (MZ) (fig. 12). In this approach, the non-linear medium is not a semiconductor laser, but the MZ modulator. The system is known as a delay dynamical system because the delay in the optical fiber is long in comparison to the response time of the modulator [36,53]. The architecture of this type of chaotic systems is inspired by the pioneering work of Ikeda [43]. In order to mask the information within the chaotic waveform produced by the optoelectronic feedback loop, a binary message was encoded on the beam produced by a third laser operating in the same wavelength. Half of the message beam was coupled directly into the electro-optic feedback loop, while half of it was sent to the receiver, while the other half was combined with the message and circulated around the feedback loop. The receiver of this communication system consisted of an identical optoelectronic feedback loop that has been split apart. A fraction of the incoming signal was sent to a photoreceiver and converted into voltage. The rest of the signal propagated through an optical fiber, which delayed the signal by an amount identical to the delay produced by the long fiber in the transmitter, and the resulting signal was used to drive an identical MZ modulator. An auxiliary laser beam passed through this modulator and was converted to a voltage via a photoreceiver. This voltage was subtracted from the voltage proportional to the incoming signal. The resulting difference signal practically contains the original message; the chaos part of the signal has been removed from the waveform with high rejection efficiency.
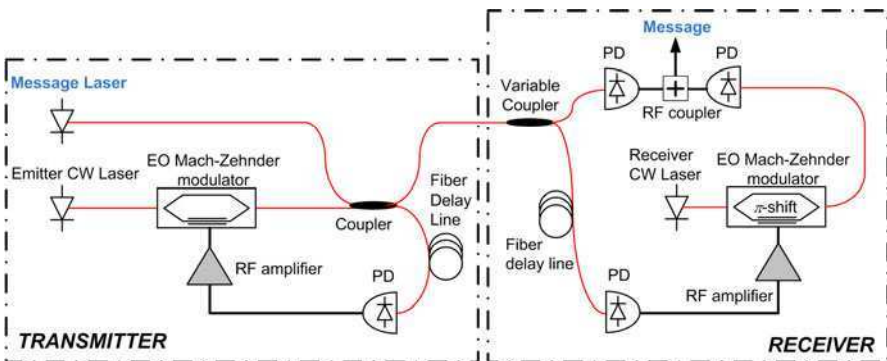

Fig. 12 Experimental setup of an optoelectronic communication transmission system based on chaotic carriers.

The message must reside in the frequency region of the chaotic carrier, in order to be indistinguishable in the frequency domain. Moreover, to avoid coherent interaction between message and chaos, the message and chaos polarization states should be orthogonal to each other. In order to prevent eavesdropping through polarization filtering, a fast polarization scrambler performing random polarization rotation should be used before transmitting the combined output. The system can be mathematically described by differential difference equations (DDEs) [43]. Specifically, the transmitter dynamics including the applied message, obey the following second-order DDE [19]:

$$x + \tau \frac{dx}{dt} + \frac{1}{\theta} \int_{t_0}^{t} x(s)ds = \beta \cdot \left[\cos^2(x(t-T) + \phi) + d \cdot m(t-T) \right] \qquad (26)$$

where $x(t) = \pi V(t)/(2V_\pi)$ is the normalized voltage applied to the RF electrode of the MZ and $\varphi = \pi V_B/(2V_{\pi,DC})$ corresponds to the operation point of the MZ determined by the voltage applied to the bias electrode. The message $d \cdot P \cdot m(t)$ has power equal to zero for "0" bits and power $d \cdot P$ for "1" bits. The parameter d is a measure of the message-to-chaos relative power and determines the masking efficiency of the system. The parameter β is the overall feedback parameter of the system and is usually called as the bifurcation parameter. β values in the range between 2.5 and 10 lead to an intense nonlinear dynamical operation of the MZ providing a hyper-chaotic optical signal at the output of the transmitter. The parameters τ, θ are the high and low cut-off characteristics times respectively of the electronic components of the feedback. The encrypted message is also a part of the signal entering into the optoelectronic feedback, meaning that the chaotic oscillations will depend on the message variations to such an extent determined by parameter d. The chaotic receiver is also governed by a similar second-order DDE, which would be identical to (26) provided that the channel effect is negligible and that the parameters of the components at the receiver side are identical to those of the transmitter module:

$$y + \tau \frac{dy}{dt} + \frac{1}{\theta} \int_{t_0}^{t} y(s)ds = \beta \cdot P_R(t-T) \qquad (27)$$

where P_R is the normalized to the received P optical power. In such a configuration, as long as the transmission effects become significant, P_R will differ from $cos^2(x(t-T)+\varphi)+d \cdot m(t-T)$ which is the originally transmitted normalized power, expecting synchronization degradation and poor performance in terms of the signal-to noise ratio of the decoded message.

An efficient temporal chaos replication between the transmitter and the receiver has been observed using this configuration, with an electrical cancellation of chaos equal to $c_{\Delta f}^{E} = 18$dB for the first 5 GHz. The message is obtained by direct modulating an external laser using an NRZ pseudorandom bit sequence (PRBS) of 2^7-1 bits up to 3 Gbit/s. This encryption scheme allowed for a BER of the decoded message equal to $7 \cdot 10^{-9}$ [32].

4.4.2 Phase Data Encoding

Chaos communications based on optoelectronic feedback has been studied and demonstrated as an alternative approach and indeed has been successfully used in

field experiments at comparable bit rates through optical intensity modulation [32]. Phase modulation systems have been studied numerically in the recent past [20] and have proved to be rather efficient in terms of high-bit rate data encryption; however, only recently Lavrov et al. reported on optical chaos encoding and decoding, involving a nonlinear delayed phase modulation architecture [55,56].

Lavrov's et al. [56] phase-chaos communication system is depicted in fig. 13. It describes a standard communication system based on differential phase shift keying (DPSK) modulation. In such a system, any wavelength division multiplexing (WDM) channel can be selected through the use of the proper external laser source, independently of the subsequent chaos communication processing. In this work, a message phase modulator *MΦM* performed the binary DPSK phase modulation ϕ_m corresponding to the message to be transmitted. *MΦM* could be equivalently placed before seeding the chaotic oscillator, or inside this oscillator. The chaotic masking consisted of the superposition of the message phase modulation and the chaos masking phase modulation, the latter being thus partly determined also by the message phase modulation. If the message phase modulation was performed outside the oscillation loop, the chaotic masking would be independent of the message phase modulation. The phase chaos generator (ΦCG) [13] thus performed the masking of the DPSK message. At the receiver side, phase chaos cancellation (ΦCC) was processed from the input light beam, after which a standard DPSK demodulation *MΦD* recovered the original binary message. Note that if *ΦCG* and *ΦCC* were omitted, the communication link becomes practically a standard optical DPSK transmission system.

4.4.3 Single-Channel Transmission Field Demonstrators

The above approach by Lavrov et al. provided the first experimental demonstration of 10 Gb/s chaos communication, and was tested also in field trials [56]. Two successive field experiments were conducted within 2009. For this reason the emitter and receiver of fig. 13 were fixed onto breadboards, so that they could be tightly fixed in handluggage-size transportation cases. The first field experiments were performed on the "Frères Lumière" all-optical fiber ring network installed in the city of Besançon, France. BER as low as 3×10^{-10} (2×10^{-7}) were obtained at 10 Gb/s for fully (partially) encrypted data and for transmission within a fiber ring over 22 km. Dispersion issues were critical, and successful data recovery was only possible with accurately tuned dispersion compensation modules. Additionally, polarization control had to be applied appropriately. Long-term stable operation of the encoding and decoding required the use of control systems for the operating point of the DPSK demodulators. The second experiment was performed on a more recent and advanced optical network in the metropolitan area of Athens, Greece. In this case the fiber loop path was close to 120 km and involved two EDFAs and two dispersion compensation units. The achieved BER value was of the order of 10^{-6} ($2 \cdot 10^{-4}$) at 10 Gb/s for fully (partially) encrypted data. By reducing the message bit rate to 3 Gb/s BER values less than 10^{-10} have been measured. These first 10 Gb/s field experiments were however not involving all the currently available optical signal processing techniques for 10 Gb/s link optimization.
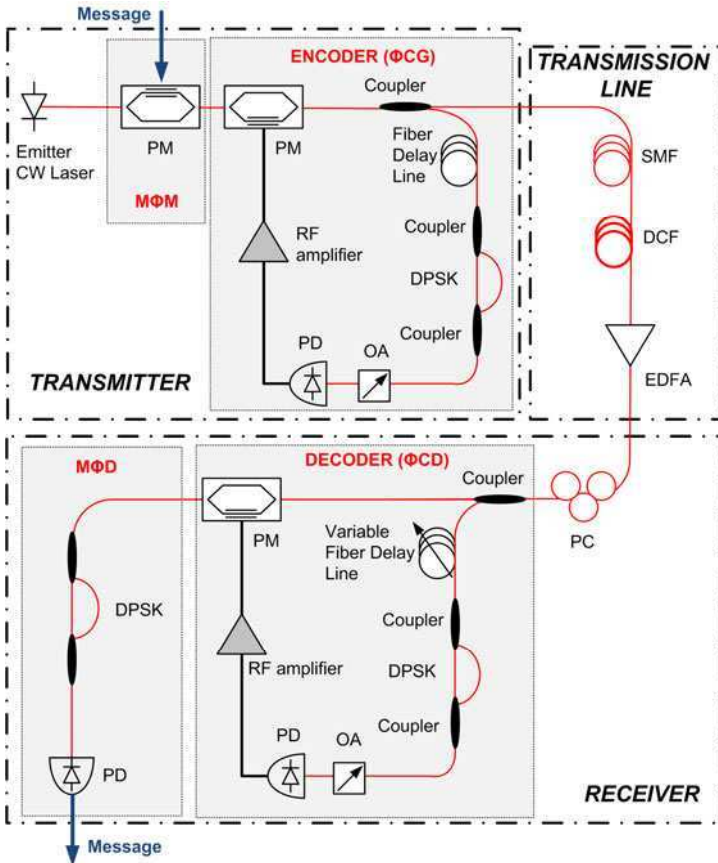
Fig. 13 Point-to-point transmission setup using optoelectronic phase chaos. Transmitter performs standard DPSK binary message modulation (MΦM) and chaotic phase masking (ΦCG). The phase modulated light beam is sent to the receiver through an installed network fiber link. The receiver performs first the chaos cancellation or demodulation (ΦCD) through phase chaos synchronization, and then standard DPSK demodulation retrieves the binary signal in the electrical domain (PM: phase modulator, DPSK: differential phase shift keying demodulator, OA: optical attenuator, PD: photodiode, SMF: single mode fiber channel, EDFA: erbium doped fiber amplifier, DCM: dispersion compensation module, PC: polarization controller).

5 Optical-Chaos-Based Ultra-Fast Physical Random Bit Generators for Secure Communications

A rather different than the previously reported applications of broadband chaotic optical signals is discussed in this section. Optical chaos generators can contribute to the implementation of secure communication systems not only via transmission of complex chaotic analog signals; they can provide the seed for generating ultra-fast physical random sequences for cryptographic applications.

Random number generators play a fundamental role in most algorithms and systems for cryptographic applications. Communications based on secret and public key cryptography, user authentication, as well as electronic lottery-based applications rely on the quality of the randomness and the generation speed provided by these generators. Two types of random number generators can be distinguished, namely true random number generators (TRNGs) and pseudorandom number generators (PRNGs) [42]. TRNGs produce random bits from random physical phenomena or noise sources [21,25,27,35,40,93]. Such non-deterministic generators have limited efficiency in number generation rates due to limitations of the mechanisms for extracting bits from the physical procedures. On the contrary, PRNGs are initiated by a relatively short key (seed) and their output is expanded into a long sequence of random bits using computational deterministic algorithms [29]. PRNGs were viewed until recently as a largely solved problem, but emerging technologies, such as cloud computing security and ultra-fast quantum key distribution, start to alter the scene. In cases where the "entropy pool" - from which the seed comes up - is inadequate, no matter how strong the encryption is, a malicious hacker could succeed in guessing correct. Thus, the question that rises is whether computers can produce or exploit truly random numbers that can't be guessed or replicated, and – at the same time – the bit rate generation can rise significantly in order to support these challenging modern applications.

Very recently, research in quantum noise TRNGs demonstrated configurations that could increase significantly the bit-rate generation efficiency, potentially over 100 Mbps [80,104]. However, even such bit-rates are much slower than the ones achieved by the recently proposed technique using broadband chaotic signals emitted from semiconductor lasers (SLs) with optical feedback. Optical chaos-based techniques provide truly random numbers as a result of partnership between quantum fluctuations and chaotic dynamics at the macroscopic level. Due to large amplitude signals the chaotic fluctuations can be easily detected, since they emerge from truly random photon quantum fluctuations enhanced by the nonlinear dynamical system of a semiconductor laser subjected to optical feedback. The specific implementations, which will be analyzed in the following sections, are thus non-deterministic although originating from chaos.

5.1 *Analog Signal Generation*

In order for a chaotic waveform to be considered as a potential seed for ultra-fast random number generation processes, it should have a bandwidth (in the electronic domain) of at least several GHz, high optical power, uniform spectral distribution and absence of residual periodicities. The statistical properties of the light emitted by semiconductor optical chaos generators are determined by the intrinsic characteristics of the semiconductor lasers (e.g. the relaxation frequency) and the characteristics of the external cavity (e.g. the strength of optical feedback, the roundtrip time of the external cavity). The adjustment of the above parameters may lead this oscillator to operate under completely diverse dynamic regimes, from stable operation and periodic solutions to complex non-linear dynamics and broadband chaos. The application of strong optical feedback makes this unit to act as a chaotic optical

oscillator, described by deterministic coupled time-delayed differential equations. However, the onset of the oscillator emission is not based on a predefined or predictable initial condition, but on the microscopic quantum noise of nonlinear amplification and mixing mechanisms of the semiconductor laser.

Since the described optical chaos generators exploit physical processes for emitting broadband optical signals, the characteristic frequencies of these processes may be evident in the output signal. Such periodicities emerge from at least two characteristic frequencies: one associated to the semiconductor laser (relaxation oscillation frequency) and the other one associated to the external cavity round-trip time. Such drawbacks of the analog chaotic signal are dealt in the relevant works so far either be optical de-correlation techniques, or by post-processing after sampling.

5.2 Single-Bit Sampling Systems

In the pioneering work of Uchida et al. [98] it was the first time that chaotic lasers have been used to demonstrate high-rate generation of random bit sequences with verified randomness. Specifically, a 1.7 Gb/s TRNG was presented based on the binary digitization of two independent chaotic SLs, finally combined under a XOR operation. The use of such chaotic optical signals in SLs led to achieve efficient and stable generation of random bits at high frequencies. High-bandwidth chaotic lasers have previously been used to demonstrate the transmission of messages hidden in complex optical waveforms. The scheme that authors used employed two semiconductor lasers with chaotic intensity oscillations. The output intensity of each laser was converted to an AC electrical signal by photodetectors, amplified and converted to a binary signal using a 1-bit analog-to-digital converter (ADC) driven by a fast clock. The ADC first converted the input analog signal into a binary signal by comparing with a threshold voltage, and then sampled the binary signal at the rising edge of the clock. The binary bit signals obtained from the two lasers were combined by a logical exclusive-OR (XOR) operation to generate a single random bit sequence. No other digital post-processing was applied. A single-laser chaotic signal could not be adopted as the random source since the periodicities of the external cavity modes (ECMs) of each chaotic waveform would cause bias in the final result [98].
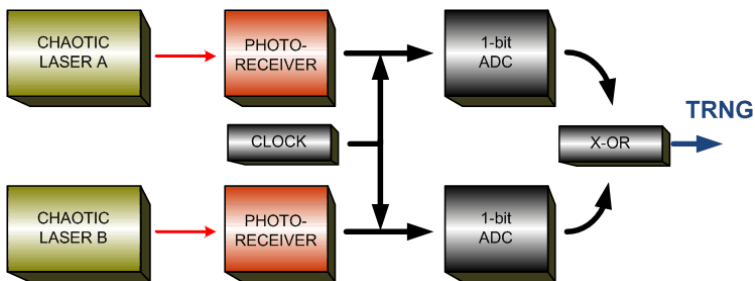

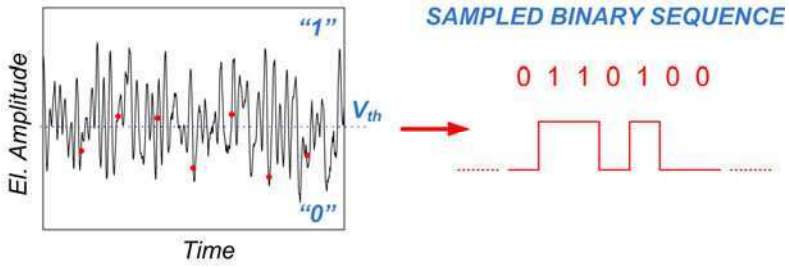**Fig. 14** A schematic diagram of Uchida's et al. TRNG.

Fig. 15 Single-bit analog-to-digital conversion, providing a sampled binary sequence. The threshold voltage V_{th} should be carefully determined in order not to cause any bias between the "0"s and "1"s.

5.3 Multi-Bit Sampling Systems

Initiated by the above work, several configurations followed targeting on increasing the bit-rate generation, as well as simplifying the used architectures. Kanter et al. in [81] increased the speed of the proposed TRNG to 12.5Gb/s. In this work authors used a single, off the shelf, multimode semiconductor laser, with absolutely no special requirements, operating in the coherence collapse regime due to feedback from an external. Due to feedback the laser is chaotic, with a broadened lasing frequency spectrum and intensity fluctuating in time. Only one incommensurate ratio between the external cavity length and an external clock rate was required. The detected laser output was sampled by an 8-bit ADC and was used to generate a Boolean sequence in the following way: the difference between consecutive sampled 8-bit values was obtained and the m least significant bits (LSBs) of the difference value served as the next m random bits of the sequence. This method was characterized as insensitive to variations of parameters such as the average laser power and did not require the tuning or determination of a decision threshold value.
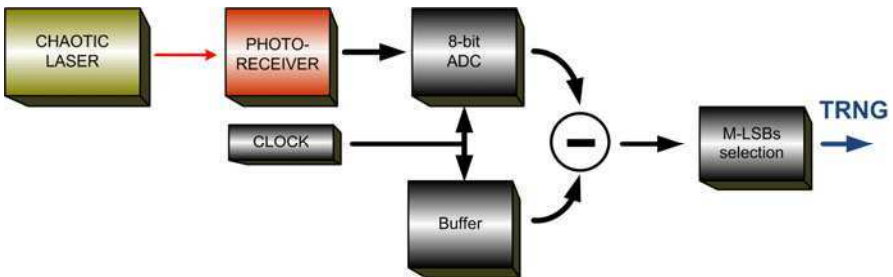

Fig. 16 A schematic diagram of Kanter's et al. TRNG.

Within the year 2010, Hirano et al. [39] adopted the multi-bit sampling at 12.5 GSa/s and, by using bandwidth-enhanced chaos in semiconductor lasers, demonstrated a 75 Gb/s TRNG. In this work, chaotic fluctuation of laser output was generated in a semiconductor laser with optical feedback and its chaotic output was injected into a second semiconductor laser in order to enhance its bandwidth up to 16 GHz, and random bit generation at rates.

In the same year, Kanter et al. have reported that bit sequences with verified randomness can be generated at higher bit rates up to 300 Gb/s using a single chaotic semiconductor laser by retaining a number of the LSBs of the value of a high derivative of the digitized chaotic laser intensity [45]. However, the bandwidth of the laser chaos is only a few GHz in their scheme. This suggests that the reported bit rate exceeds the capacity of the laser chaos to generate nondeterministic random bits. Moreover, extracting more bits from high derivatives could be more susceptible to the effects of physical noise in the AD converter - a potential additional source of randomness which is separate from the laser chaos. A strong motivation for using direct sampling of optical chaos for random number generation is to reduce the dependence on digital electronic operations, which may be difficult to implement at high frequencies, and which in principle cannot increase the rate for generation of nondeterministic bits.

Finally, compact photonic integrated circuits (PICs) that exploited broadband chaotic signals were used by Argyris et al. to generate TRBS at 140 Gb/s [15]. The proposed generator was a simple configuration, consisting of the PIC, a photodetector and a 40GSa/s oscilloscope, without including any optical de-correlation methods (fig. 17). Depending on the operating conditions of the PIC and by using MSB elimination post-processing, real time bit sequences were extracted from an oscilloscope's Ethernet output port. The proposed configuration provided a significant advance in terms of simplicity, performance and especially robustness. The PIC employed in this work is the same presented before in Section 2.2. The oscilloscope's 8-bit A/D converter, along with the internal 16-bit digital-to analog convertor (DAC) and the rest processing units, provided a noise-enhanced, 16-bit output binary sequence for each sample. An external down-sampling to 10GSa/s was applied – only one out of four samples was considered – in order to eliminate any effect of interpolation samples of the oscilloscope. In such a way the initial bandwidth of the chaotic signal was preserved.

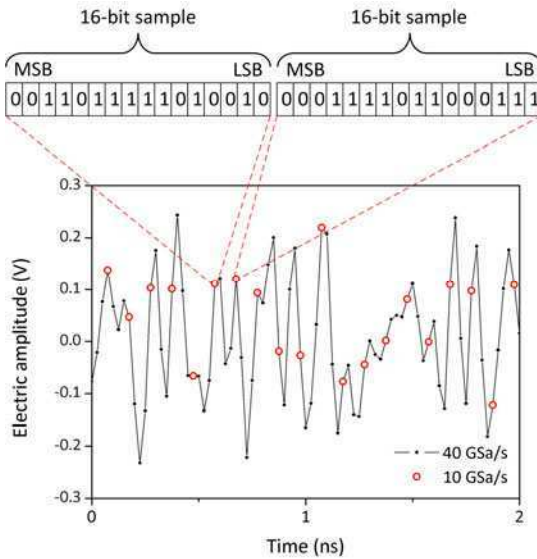**Fig. 17** A TRBG based on the emitted chaotic signal from the PIC.

Fig. 18 Multi-bit sampling of a chaotic analog signal. Only *k* LSBs out of the 16-bit digitized representation of each sample are exploited and included in the final output sequence.

5.4 *Random Bit Generation and Verification*

In order to claim and prove the randomness of a bit sequence strict conditions must be fulfilled. The most representative statistical tests performed for this reason are included in the NIST SP800-22 (last revision in April 2010) test suite [42]. In this suite at least 1000 samples of 1Mbit sequences – constructed by the appropriate number of the *k* LSBs adopted in each case – are evaluated. Pass criteria are determined by the sequence length and the significance level. A significance level $\alpha = 0.01$ is set for the p-values of each sequence test, with a desirable uniformity *P*-value larger than 0.0001. For the 1000 samples, the proportion of sequences that satisfy *p*-value > α, is estimated to be 0.99±0.0094. Another statistical test suite for random number generators that is commonly used is the so-called "Diehard" tests [64]. The Diehard test suite consists of 18 statistical tests and are performed using 74 Mbit sequences and significance level of $\alpha = 0.01$.

For example, a randomness mapping of the PIC-based TRBG of [15] in terms of operating conditions is illustrated in fig. 19. In this work, the characterization has been performed versus the biasing laser current and the number of LSBs of each sample included in the output sequence – from *k*=1 to *k*=16. The randomness is verified when all 15 statistical tests of the NIST test suite are passed. In the specific work, all NIST tests were passed, even when including 14 LSBs – for specific operating conditions – representing a final bit-rate generation of 140Gb/s.
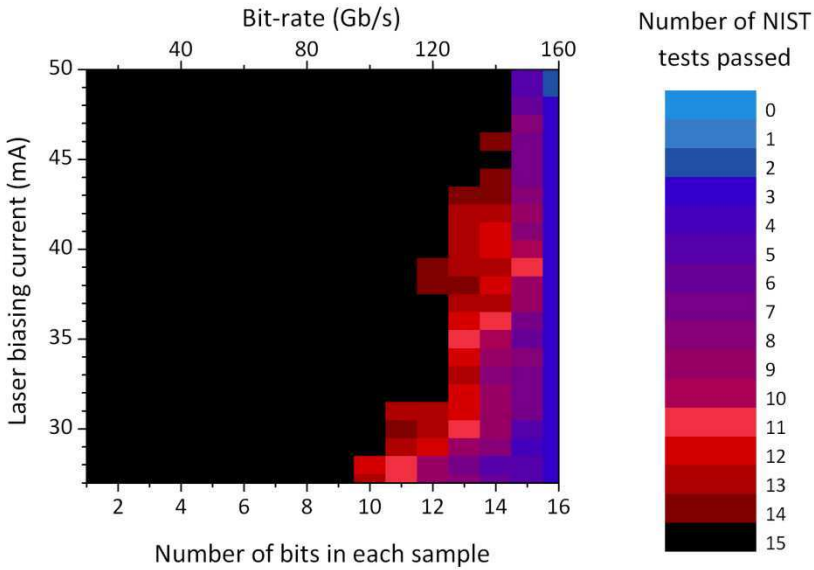
Fig. 19 Mapping of the TRBG performance in terms of bit-rate and number of NIST randomness tests passed. The characterization has been performed vs. the biasing laser current and the number of LSBs of each sample included in the output sequence. Black color grade regions designate operating conditions where all NIST randomness tests are successful.

A representative analysis of the NIST statistical tests results, for the worst value of the obtained p-values for each test, is presented in table 1, as submitted in the work of [15].

Table 1 Typical results of NIST SP800-22 (rev.1a) statistical test suite.

STATISTICAL TEST	P-VALUE (min)	PROPORTION	RESULT
Frequency	0.375313	0.986	Passed
Block frequency	0.674543	0.990	Passed
Runs	0.773405	0.988	Passed
Longest Run	0.087162	0.988	Passed
Rank	0.291091	0.991	Passed
Discrete Fourier transform	0.134355	0.984	Passed
Non-overlapping templates	0.002186	0.986	Passed
Overlapping templates	0.989425	0.990	Passed
Universal	0.705466	0.991	Passed
Linear complexity	0.883171	0.989	Passed
Serial	0.123038	0.995	Passed
Approximate entropy	0.607993	0.993	Passed
Cumulative sums	0.514124	0.991	Passed
Random excursions	0.023140	0.9886	Passed
Random excursions variant	0.015993	0.9869	Passed

An analogous analysis of the Diehart statistical tests [64] results, has been presented in the work [39], indicating the worst value of the obtained *p*-values for each test (Table 2).

Table 2 Typical results of the statistical test suite Diehard. "KS" indicates that a single P-value is obtained by the Kolmogorov-Smirnov (KS) test [39]. For the tests which produce multiple P-values without the KS test, the worst case is shown.

STATISTICAL TEST	P-VALUE (min)	RESULT
Birthday spacing	0.882291	Success (KS)
Overlapping 5-permutaion	0.483639	Success
binary rank for 31 × 31 matrices	0.658636	Success
binary rank for 32 × 32 matrices	0.391334	Success
binary rank for 8 × 8 matrices	0.367852	Success (KS)
Bit stream	0.056500	Success
Overlapping-Paris-Spares-Occupancy	0.000700	Success
Overlapping-Quadruples-Spares-Occupancy	0.015800	Success
DNA	0.015800	Success
Count-the-1's on a stream of bytes	0.049820	Success
Count-the-1's for specific bytes	0.353940	Success
Parking lot	0.260828	Success (KS)
Minimum distance	0.326556	Success (KS)
3D spheres	0.059882	Success (KS)
Speeze	0.458916	Success
Overlapping sums	0.965410	Success (KS)
Runs	0.181109	Success (KS)
Craps	0.812245	Success

5.5 Chaos-Based TRNGs Applied to Cryptography

In the typical scenarios of secured channels the communicating parties have to hold a common key in the form of a bit string which is known only to the two parties. Physical mechanisms based on quantum mechanics have been suggested recently for a secure key-exchange protocol with the important and unique ability of the two communicating parties to detect the presence of any third party trying to gain knowledge of the key. The first layer of the quantum protocol is based on quantum ingredients such as entangled pairs of photons and results in correlated keys for both partners. The second classical layer consists of information reconciliation and privacy amplification (error correcting code and source coding). These result in identical keys for the communicating pair while leakage of information to an eavesdropper is eliminated; however, such procedures lower the rate at which random bits can be generated. By exploiting the previously reported ultra-fast random number generators based on optical chaos, Kanter et al. proposed a secure synchronization method of two high bandwidth RBGs over a public channel using a classical mechanism [46]. The focus of this work was to propose a secure synchronization zero lag synchronization (ZLS) of two mutually coupled chaotic las-

ers. The ZLS mechanism is not sufficiently secure in its simple form to act as a key-exchange protocol, and it serves only as an information carrier to generate correlated random bit sequences. Identical random bit sequences could be constructed from these correlated sequences via information reconciliation and privacy amplification. Furthermore, the proposed mechanism allows the secure generation of a synchronized random bit string amongst a small network of communicating parties [46].

This application foretells that ultra-fast random number generators are expected to play a significant role in the future security of high-speed communications. Compact chaos-on-chip devices, which will be integrated in computer cards or motherboards with direct access to Ethernet ports, will provide each user an unlimited source of random codes for every potential application.

6 Conclusions

Chaos data encryption in optical communications proves to be an efficient alternative of securing fiber transmission lines in the physical layer. The first implementations of communication systems deployed so far provide strong evidence of the feasibility of this method to strengthen the security of fiber-based high-speed networks. Error-free data decoding in bit-rates up to 2.5 Gb/s in all-optical and 10 Gb/s in electro-optical systems has been a fulfilled target, not only by using fiber-based chaos generators but also completely photonic integrated devices that could be easily adaptive to emitter/receiver commercial network cards. The WDM transmission demonstrations showed the applicability of the method to support numerous users in multiple access networks. Chaos data encryption is a relatively recent method for securing optical communication networks in the hardware layer. Of course, this method is not foreseen to substitute the cryptographic methods developed so far in an algorithmic level that shield efficiently any type of communications nowadays. However, it could provide an additional level of transmission security when fiber-optic networks are the physical medium between the communicating parts. Finally, communication systems and networks could also benefit from the broadband chaotic signals that these systems support in a diverse way. By applying appropriate digital signal processing techniques on these signals ultra-fast physical random bit generators become available, at bit rates that no other physical mechanism can provide.

References

1. Abarbanel, H.D.I., Kennel, M.B., Buhl, M., et al.: Chaotic dynamics in erbium-doped fiber ring lasers. Phys. Rev. A 60, 2360–2374 (1999)
2. Abarbanel, H.D.I., Kennel, M.B., Illing, L., et al.: Synchronization and communication using semiconductor lasers with optoelectronic feedback. IEEE J. Quantum Electron 37, 1301–1311 (2001)
3. Agrawal, G.P., Dutta, N.K.: Semiconductor lasers. Van Nostrand Reinhold, New York (1993)
4. Agrawal, G.P., Dutta, N.K.: Semiconductor lasers, 2nd edn. Kluwer Academic Publishers, Massachusetts (2000)

5. Annovazzi-Lodi, V., Donati, S., Scire, A.: Synchronization of chaotic injected laser systems and its application to optical cryptography. IEEE J. Quantum Electron 32, 953–959 (1996)

6. Annovazzi-Lodi, V., Benedetti, M., Merlo, S., et al.: Message encryption by phase modulation of a chaotic optical carrier. IEEE Photon Technol. Lett. 19, 76–78 (2007)

7. Argyris, A., Kanakidis, D., Bogris, A., et al.: Experimental evaluation of an open-loop all-optical chaotic communication system. IEEE J. Sel. Topics Quantum Electron 10, 927–935 (2004)

8. Argyris, A., Kanakidis, D., Bogris, A., et al.: First experimental demonstration of an all-optical chaos encrypted transmission system. In: Proc. ECOC 2004, Tu4.5.1, pp. 256–257 (2004)

9. Argyris, A., Kanakidis, D., Bogris, A., et al.: Spectral synchronization in chaotic optical communication systems. IEEE J. Quantum Electron 41, 892–897 (2005)

10. Argyris, A., Syvridis, D., Larger, L., et al.: Chaos-based communications at high bit rates using commercial fiber-optic links. Nature 438, 343–346 (2005)

11. Argyris, A., Hamacher, M., Chlouverakis, K.E., et al.: A photonic integrated device for chaos applications in communications. Phys. Rev. Lett. 100, 194101 (2008)

12. Argyris, A., Bogris, A., Hamacher, M., et al.: Experimental evaluation of subcarrier modulation in all-optical chaotic communication systems. Opt. Lett. 35, 109–111 (2010)

13. Argyris, A., Grivas, E., Hamacher, M., et al.: Chaos-on-a-chip secures data transmission in optical fiber links. Opt. Express 18, 5188–5198 (2010)

14. Argyris, A., Grivas, E., Bogris, A., et al.: Transmission effects in wavelength division multiplexed chaotic optical communication systems. IEEE J. Lightwave Technol. 28, 3107–3114 (2010)

15. Argyris, A., Deligiannidis, S., Pikasis, E., et al.: Implementation of 140 Gb/s true random bit generator based on a chaotic photonic integrated circuit. Opt. Express 18, 18763–18768 (2010)

16. Bauer, S., Brox, O., Kreissl, J., et al.: Nonlinear dynamics of semiconductor lasers with active optical feedback. Phys. Rev. E 69, 016206 (2004)

17. Bennett, C.H., Brassard, G.: Quantum cryptography: public key distribution and coin tossing. In: Int. Conf. Comput. Systems & Signal Processing 1984, India, pp. 175–179 (1984)

18. Bogris, A., Chlouverakis, K.E., Argyris, A., et al.: Subcarrier modulation in all-optical chaotic communication systems. Opt. Lett. 32, 2134–2136 (2007)

19. Bogris, A., Argyris, A., Syvridis, D.: Analysis of the optical amplifier noise effect on electrooptically generated hyperchaos. IEEE J. Quantum Electron 47, 552–559 (2007)

20. Bogris, A., Rizomiliotis, P., Chlouverakis, K.E., et al.: Feedback phase in optically generated chaos: A secret key for cryptographic applications. IEEE J. Quantum Electron 44, 119–124 (2008)

21. Bucci, M., Germani, L., Luzzi, R., et al.: A high-speed oscillator-based truly random number source for cryptographic applications on a Smart Card IC. IEEE Trans. Comput. 52, 403–409 (2003)

22. Chen, H.F., Liu, J.M.: Open-loop chaotic synchronization of injection-locked semiconductor lasers with gigahertz range modulation. IEEE J. Quantum Electron 36, 27–34 (2000)

23. Colet, P., Roy, R.: Digital communication with synchronized chaotic lasers. Opt. Lett. 19, 2056–2058 (1994)

24. Cuomo, K.M., Oppenheim, A.V.: Circuit implementation of synchronized chaos with applications to communications. Phys. Rev. Lett. 71, 65–68 (1993)

25. Danger, J.L., Guilley, S., Hoogvorst, P.: High speed true random number generator based on open loop structures in FPGAs. Microelectron J. 40, 1650–1656 (2009)
26. Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Trans. Inf. Theory 22, 644–654 (1976)
27. Dynes, J.F., Yuan, Z.L., Sharpe, A.W., et al.: A high speed, post-processing free, quantum random number generator. Appl. Phys. Lett. 93, 031109 (2008)
28. Fischer, I., Van Tartwijk, G.H.M., Levine, A.M., et al.: Fast pulsing and chaotic itinerancy with a drift in the coherence collapse of semiconductor lasers. Phys. Rev. Lett. 76, 220–223 (1996)
29. Ferguson, N., Schneier, B.: Practical cryptography. John Wiley & Sons, Chichester (2003)
30. Franck, T., Brorson, S.D., Moller-Larsen, A., et al.: Synchronization phase diagrams of monolithic colliding pulse mode-locked lasers. IEEE Photon Technol. Lett. 8, 40–42 (1996)
31. Fujino, H., Ohtsubo, J.: Synchronization of chaotic oscillations in mutually coupled semiconductor lasers. Opt. Rev. 8, 351–357 (2001)
32. Gastaud, N., Poinsot, S., Larger, L., et al.: Electro-optical chaos for multi-10 Gbit/s optical transmissions. Electr. Lett. 40, 898–899 (2004)
33. Giacomelli, G., Politi, A.: Relationship between delayed and spatially extended dynamical systems. Phys. Rev. Lett. 76, 2686–2689 (1996)
34. Gisin, N., Ribordy, G., Tittel, W., et al.: Quantum cryptography. Rev. Mod. Phys. 74, 145–195 (2002)
35. Gleeson, J.T.: Truly random number generator based on turbulent electroconvection. Appl. Phys. Lett. 81, 1949 (2002)
36. Goedgebuer, J.P., Levy, P., Larger, L., et al.: Optical communication with synchronized hyperchaos generated electooptically. IEEE J. Quantum Electron 38, 1178–1183 (2002)
37. Halle, K.S., Wu, C.W., Itoh, M., et al.: Spread spectrum communication through modulation of chaos. Int. J. Bifurcation & Chaos 3, 469–477 (1993)
38. Heil, T., Mulet, J., Fischer, I., et al.: On/off phase shift-keying for chaos-encrypted communication using external-cavity semiconductor lasers. IEEE J. Quantum Electron 38, 1162–1170 (2002)
39. Hirano, K., et al.: Fast random bit generation with bandwidth-enhanced chaos in semiconductor lasers. Opt. Express 18, 5512–5524 (2010)
40. Holman, W.T., Connelly, J.A., Dowlatabadi, A.B.: An integrated analog/digital random noise source. IEEE Trans. Circuits Syst. I 44, 521–528 (1997)
41. http://nova.uib.es/project/occult
42. http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf (2001)
43. Ikeda, K.: Multiple-valued stationary state and its instability of the transmitted light by a ring cavity system. Opt. Commun. 30, 257–261 (1979)
44. Ikuma, Y., Ohtsubo, J.: Dynamics in compound cavity semiconductor lasers induced by small external cavity length change. IEEE J. Quantum Electron 34, 1240–1246 (1998)
45. Kanter, I., Aviad, Y., Reidler, I., et al.: An optical ultrafast random bit generator. Nature Photon 4, 58–61 (2010)
46. Kanter, I., Butkovski, M., Peleg, Y., et al.: Synchronization of random bit generators based on coupled chaotic lasers and application to cryptography. Opt. Express 18, 18292–18302 (2010)

47. Kao, Y.H., Wang, N.M., Chen, H.M.: Mode description of routes to chaos in external-cavity coupled semiconductor lasers. IEEE J. Quantum Electron 30, 1732–1739 (1994)
48. Kocarev, L., Halle, K.S., Eckert, K., et al.: Experimental demonstration of secure communications via chaotic synchronization. Int. J. Bifurcation & Chaos 2, 709–713 (1992)
49. Kusumoto, K., Ohtsubo, J.: 1.5-GHz message transmission based on synchronization of chaos in semiconductor lasers. Opt. Lett. 27, 989–991 (2002)
50. Kusumoto, K., Ohtsubo, J.: Anticipating synchronization based on optical injection-locking in chaotic semiconductor lasers. IEEE J. Quantum Electron 39, 1531–1536 (2003)
51. Lang, R., Kobayashi, K.: External optical feedback effects on semiconductor injection laser properties. IEEE J. Quantum Electron 16, 347–355 (1980)
52. Larger, L., Dudley, J.M.: Nonlinear dynamics: Optoelectronic chaos. Nature 465, 41–42 (2010)
53. Larger, L., Goedgebuer, J.P., Delorme, F.: Optical encryption system using hyperchaos generated by an optoelectronic wavelength oscillator. Phys. Rev. E 57, 6618–6624 (1998)
54. Larger, L., Goedgebuer, J.P., Udaltsov, V.: Ikeda-based nonlinear delayed dynamics for application to secure optical transmission systems using chaos. C. R. Physique 5, 669–681 (2004)
55. Lavrov, R., Peil, M., Jacquot, M., et al.: Electro-optic delay oscillator with nonlocal nonlinearity: Optical phase dynamics, chaos, and synchronization. Phys. Rev. E 80, 026207 (2009)
56. Lavrov, R., Jacquot, M., Larger, L.: Nonlocal nonlinear electro-optic phase dynamics demonstrating 10 Gb/s chaos communications. IEEE J. Quantum Electron 46, 1430–1435 (2010)
57. Liu, J.M., Simpson, T.B.: Four-wave mixing and optical modulation in a semiconductor laser. IEEE J. Quantum Electron 30, 957–965 (1994)
58. Liu, J.M., Chen, H.F., Tang, S.: Synchronized chaotic optical communications at high bit-rates. IEEE J. Quantum Electron 38, 1184–1196 (2002)
59. Liu, Y., Chen, H.F., Liu, J.M., et al.: Communication using synchronization of optical-feedback-induced chaos in semiconductor lasers. IEEE Trans. Circuits Syst. I 48, 1484–1490 (2001)
60. Liu, Y., Takiguchi, Y., Aida, T., et al.: Injection locking and synchronization of periodic and chaotic signals in semiconductor lasers. IEEE J. Quantum Electron 39, 269–278 (2003)
61. Locquet, A., Rogister, F., Sciamanna, M., et al.: Two types of synchronization in unidirectionally coupled chaotic external-cavity semiconductor lasers. Phys. Rev. E 64, 045203 (2001)
62. Locquet, A., Massoler, C., Mirasso, C.R.: Synchronization regimes of optical-feedback-induced chaos in unidirectionally coupled semiconductor lasers. Phys. Rev. E 65, 056205 (2002)
63. Locquet, A., Masoller, C., Megret, P., et al.: Comparison of two types of synchronization of external-cavity semiconductor lasers. Opt. Lett. 27, 31–33 (2002)
64. Marsaglia, G.: DIEHARD: A battery of tests of randomness (1996), http://stat.fsu.edu/
65. Masoller, C.: Spatio-temporal dynamics in the coherence collapsed regime of semiconductor lasers with optical feedback. Chaos 7, 455–462 (1997)

66. Masoller, C.: Anticipation in the synchronization of chaotic semiconductor lasers with optical feedback. Phys. Rev. Lett. 86, 2782–2785 (2001)

67. Menezes, A., van Oorschot, P., Vanstone, S.: Handbook of Applied Cryptography. CRC Press, Boca Raton (1996)

68. Mirasso, C.R., Colet, P., Garcia-Fernandez, P.: Synchronization of chaotic semiconductor lasers: Application to encoded communications. IEEE Photon Technol. Lett. 8, 299–301 (1996)

69. Mirasso, C.R., Mulet, J., Masoller, C.: Chaos shift keying encryption in chaotic external-cavity semiconductor lasers using a single-receiver scheme. IEEE Photon Technol. Lett. 14, 456–458 (2002)

70. Mork, J., Tromborg, B., Christiansen, P.L.: Bistability and low-frequency fluctuations in semiconductor lasers with optical feedback: a theoretical analysis. IEEE J. Quantum Electron 24, 123–133 (1998)

71. Murakami, A., Ohtsubo, J., Liu, Y.: Stability analysis of semiconductor laser with phase-conjugate feedback. IEEE J. Quantum Electron 33, 1825–1831 (1997)

72. Murakami, A., Ohtsubo, J.: Synchronization of feedback-induced chaos in semiconductor lasers by optical injection. Phys. Rev. A 65, 033826 (2002)

73. Ohtsubo, J.: Feedback induced instability and chaos in semiconductor lasers and their applications. Opt. Rev. 6, 1–15 (1999)

74. Ohtsubo, J.: Chaos synchronization and chaotic signal masking in semiconductor lasers with optical feedback. IEEE J. Quantum Electron 38, 1141–1154 (2002)

75. Pan, M.W., Shi, B.P., Gray, G.R.: Semiconductor laser dynamics subject to strong optical feedback. Opt. Lett. 22, 166–168 (1997)

76. Parlitz, U., Chua, L.O., Kocarev, L., et al.: Transmission of digital signals by chaotic synchronization. Int. J. Bifurcation & Chaos 2, 973–977 (1992)

77. Paul, J., Lee, M.W., Shore, K.A.: 3.5-GHz signal transmission in an All-optical chaotic communication scheme using 1550-nm diode lasers. IEEE Photon Technol. Lett. 17, 920–922 (2005)

78. Pecora, L.M., Carroll, T.L.: Synchronization in chaotic systems. Phys. Rev. Lett. 64, 821–824 (1990)

79. Petermann, K.: Laser diode modulation and noise. Kluwer, Dordrecht (1998)

80. Qi, B., Chi, Y.M., Lo, H.K., et al.: High-speed quantum random number generation by measuring phase noise of a single-mode laser. Opt. Lett. 35, 312–314 (2010)

81. Reidler, I., Aviad, Y., Rosenbluh, M., et al.: Ultrahigh-speed random number generation based on a chaotic semiconductor laser. Phys. Rev. Lett. 103, 24102 (2009)

82. Rivest, R.L., Shamir, A., Adleman, L.M.: A method for obtaining digital signatures and public-key cryptosystems. Commun. ACM 21, 120–126 (1978)

83. Rukhin, A., et al.: A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. NIST Special Publication 800-22, Revision 1a (2010)

84. Sano, T.: Antimode dynamics and chaotic itinerancy in the coherent collapse of semiconductor-lasers with optical feedback. Phys. Rev. A 50, 2719–2726 (1994)

85. Sargent III, M., Scully, M.O., Lamb, J.E.: Laser physics. Addison-Wesley, Massachusetts (1974)

86. Shu, L., Costello Jr., D.J.: Error control coding: fundamentals and applications. Prentice-Hall, New Jersey (1983)

87. Simpson, T.B., Liu, J.M.: Period-doubling cascades and chaos in a semiconductor laser with optical injection. Phys. Rev. A 51, 4185–4185 (1995)

88. Sivaprakasam, S., Shahverdiev, E.M., Spencer, P.S., et al.: Experimental demonstration of anticipating solution in chaotic semiconductor lasers with optical feedback. Phys. Rev. Lett. 87, 4101–4103 (2001)
89. Takiguchi, Y., Liu, Y., Ohtsubo, J.: Low-frequency fluctuation and frequency-locking in semiconductor lasers with long external cavity feedback. Opt. Rev. 6, 399–401 (1999)
90. Tang, S., Liu, J.M.: Message encoding-decoding at 2.5 Gbits/s through synchronization of chaotic pulsing semiconductor lasers. Opt. Lett. 26, 1843–1845 (2001)
91. Tang, S., Liu, J.M.: Synchronization of high-frequency chaotic optical pulses. Opt. Lett. 26, 596–598 (2001)
92. Tkach RW, Chraplyvy AR (1986) Regimes of feedback effects in 1.5 μm distributed feedback lasers. J Lightwave Technol LT-4:1655–1661
93. Tokunaga, C., Blaauw, D., Mudge, T.: True random number generator with a metastability-based quality control. IEEE J. Solid-State Circ. 43, 78–85 (2008)
94. Toral, R., Chakrabarti, A.: Generation of Gaussian distributed random numbers by using a numerical inversion method. Comp. Phys. Commun. 74, 327–334 (1993)
95. Tromborg, B., Osmundsen, J.H., Olesen, H.: Stability analysis for a semiconductor laser in an external cavity. IEEE J. Quantum Electron QE-20, 1023–1031 (1984)
96. Uchida, A., Liu, Y., Davis, P.: Characteristics of chaotic masking in synchronized semiconductor lasers. IEEE J. Quantum Electron 39, 963–970 (2003)
97. Uchida, A., Heil, T., Liu, Y., et al.: High-frequency broad-band signal generation using a semiconductor laser with a chaotic optical injection. IEEE J. Quantum Electron 39, 1462–1467 (2003)
98. Uchida, A., et al.: Fast physical random bit generation with chaotic semiconductor lasers. Nature Photon 2, 728–732 (2008)
99. Ursin, R., Tiefenbacher, F., Schmitt-Manderbach, T., et al.: Entanglement-based quantum communication over 144 km. Nature Phys. 3, 481–486 (2007)
100. Ushakov, O., Bauer, S., Brox, O., et al.: Self-organization in semiconductor lasers with ultrashort optical feedback. Phys. Rev. Lett. 92, 043902 (2004)
101. Van Tartwijk, G.H.M., Agrawal, G.P.: Laser instabilities: a modern perspective. Prog. Quantum Electron 22, 43–122 (1998)
102. Van Wiggeren, G.D., Roy, R.: Communications with chaotic lasers. Science 279, 1198–1200 (1998)
103. Vicente, R., Perez, T., Mirasso, C.R.: Open- versus close-loop performance of synchronized chaotic external-cavity semiconductor lasers. IEEE J. Quantum Electron 38, 1197–1204 (2002)
104. Wayne, M.A., Kwiat, P.G.: Low-bias high-speed quantum random number generator via shaped optical pulses. Opt. Express 18, 9351–9357 (2010)
105. Wiesner, S.: Conjugate coding. ACM Sigact News 15, 78–88 (1983)
106. Wilson, S.G.: Digital Modulation and Coding. Prentice-Hall, New Jersey (1996)
107. Wu, C.W., Chua, L.O.: A simple way to synchronize chaotic systems with applications to secure communication systems. Int. J. Bifurcation & Chaos 3, 1619–1627 (1993)
108. Yousefi, M., Barbarin, Y., Beri, S.: New role for nonlinear dynamics and chaos in integrated semiconductor laser technology. Phys. Rev. Lett. 98, 044101 (2007)
109. Zhang, J.Z., Wang, A.B., Wang, J.F., et al.: Wavelength division multiplexing of chaotic secure and fiber-optic communications. Opt. Express 17, 6357–6367 (2009)

Chapter 11
Performance Evaluation of Chaotic and Conventional Encryption on Portable and Mobile Platforms

Rogelio Hasimoto-Beltran[1], Fadi Al-Masalha[2], and Ashfaq Khokhar[2]

[1] Center for Research in Mathematics (CIMAT)
`hasimoto@cimat.mx`
[2] University of Illinois at Chicago (UIC)
`{falmas2,ashfaq}@uic.edu`

Abstract. Protection of private user information in computers and communication networks has been one of the major concerns during the last decade. It has become even more critical due to pervasive use of smart mobile devices, and is exacerbated due to their limited processing and battery power needed to manage complex encryption schemes, particularly for real-time multimedia applications (audio and video). Secure multimedia communication systems require processing of huge amounts of information at speeds ranging from Kilobits/sec (Kbs) to the order of Megabits/sec (Mbs). Provisioning of security for such large volumes of data in mobile devices may be simply infeasible when the complexity of related operations is beyond the processing limit of such devices. In this chapter we evaluate the performance of different encryption schemes, including AES implementations and non-conventional chaotic encryption on different architectures. Our experiments reveal that chaos-based schemes outperform the conventional AES implementation in terms of CPU usage, encryption speed, and energy consumption. Particularly they consume 300-400% less CPU power, and have over 250% faster encryption speed. However, the performance also depends on the floating point capability of the platform; a suitable scheme may be chosen depending on the CPU power of platform. The performance results reported in this chapter are based on experiments on contemporary desktops, laptops, netbooks, and cell phones (Nokia N800 and N900).

1 Introduction

Two decades of active learning in chaos-based cryptography have fructified in identifying some of the weaknesses and strengths of digital chaos theory application in secure communication. One of the main concerns is that when chaos is

represented digitally, it suffers degraded distribution and reduced cycle length, which severely affects the vulnerability of chaos based encryption systems [1-4]. On the positive side, however, digital chaos maintains the good properties required in cryptography, the most prominent being sensitivity to parameters, sensitivity to initial conditions, and unpredictable trajectories [5].

Different solutions have been investigated to deal, to some extent, with the digital degradation of chaos, including: periodic perturbations of the state variable and/or parameter [3, 14, 23-24], the use of higher dimensional maps or multiple one-dimensional maps [25-27], numerical and analytical cycle detection [28-31], etc. Perturbation based schemes are among the most popular because of their ability to generate a totally different chaotic trajectory without considerably affecting the performance of the encryption system.

With this broader understanding of digital chaos, the chaotic encryption community is undertaking the development of robust and high performance schemes suitable for more demanding applications, such as multimedia communications (audio, image and video transmission) [6-8], running under a wide range of platforms, including those with limited processing power, e.g., handheld or mobile devices [6]. Handheld devices using rechargeable battery units represent one of the most convenient ways to access private information wirelessly (bank accounts, bill paying, online browsing, video streaming, etc.) on anytime-anywhere basis. These advantages come with intrinsic security risks; data transfer is ubiquitous, and may be intercepted by malicious intruders. Protecting private user information has been one of the major concerns during the last decade, and more recently the efforts are being concentrated on mobile devices. This problem is further exacerbated by the growing prevalence of multimedia applications, where, at times, secure streaming of large video/audio streams is desired.

There are two particular issues in securing information on mobile devices: 1) lack of processor power to manage complex encryption schemes, particularly for live streaming, and 2) limited battery power to manage not only the encryption process, but also energy consuming computations such as video/audio encoding/decoding. The problem gets worse in a more constrained environment such as sensor networks [10], where memory storage is limited and battery power is difficult to replenish. Therefore, it is imperative that the encryption process must be computationally simple (without compromising security) and fast to satisfy current mobility or real time streaming demands.

Chaotic schemes for mobile or handheld devices can offer a good trade-off between security and performance according to users' needs and CPU power limitations. However, one of the main issues in chaotic encryption is the use of floating-point arithmetic, which may require additional hardware (and compatible arithmetic operations between different processors) in order to run efficiently. This concern is becoming less important because of the fact that current handheld device technology in most cases includes a floating-point unit. For example, the Texas Instrument OMAP3 3430 technology is used by the Motorola Droid, Milestone, Palm Pre, Samsung i8910, and Nokia N900 cellular phones. Note that

hardware efficiency concerns are also valid for conventional encryption schemes [11, 21]. For example, every implementation of AES (Advance Encryption Standard) is carefully targeted for a particular CPU architecture to speed up the most demanding operations such as integer multiplication, memory access, bit shifting and XOR [11].

We foresee chaos-based encryption as a viable alternative to traditional encryption schemes especially for real-time multimedia applications. In this chapter we study the performance of chaos-based encryption [9] and different AES implementations on different computing platforms, such as desktops, laptops, netbooks, and cell phones (Nokia N800 and N900). Several variables are analyzed in the experiments such as encryption speed, CPU usage, and energy consumption. These experiments provide a guideline to chaotic and conventional cryptography usability under different hardware and application scenarios.

2   Chaotic and Conventional Encryption Schemes

This section describes the theoretical aspects of conventional and chaotic encryption schemes employed in the performance evaluation. In the case of AES, we study the latest implementation proposed by Bernstein at al. [11], and other popular implementations of AES developed by Gladman [12], and PolarSSL [13]. In the case of chaotic encryption, there are several good candidates to be included in the evaluation process; however the scheme in [9] was selected for the following reasons: a) it is a general purpose scheme for all media types (text, image, audio, and video), b) authors report one of the best encryption speeds on different platforms and operating systems, and c) it provides a good balance between security and performance. Also, code availability was an important point in making the final decision.

For the sake of completeness, we provide a detail description of the chaos-based encryption proposed by Hasimoto [9]. For the case of AES, we only provide a brief description of AES [20, 21], since it is a widely known scheme.

2.1   Chaotic Encryption System

As stated in the previous section, the digital degradation of chaos is a major concern in developing robust chaotic encryption systems. Different solutions to this problem have been employed in [9] to increase the system cycle length, the most important are: a) an *N-array* of independently iterated maps, b) an intermittent 3-level hierarchical perturbation scheme to change the system dynamics, and c) a double feedback model that spreads out plaintext changes over the array of maps, producing a totally different system trajectory. All these solutions work together to provide system interdependencies, wherein the entire system (chaotic maps, ciphertext, feedback) reacts to any plaintext change in the case of attacks.
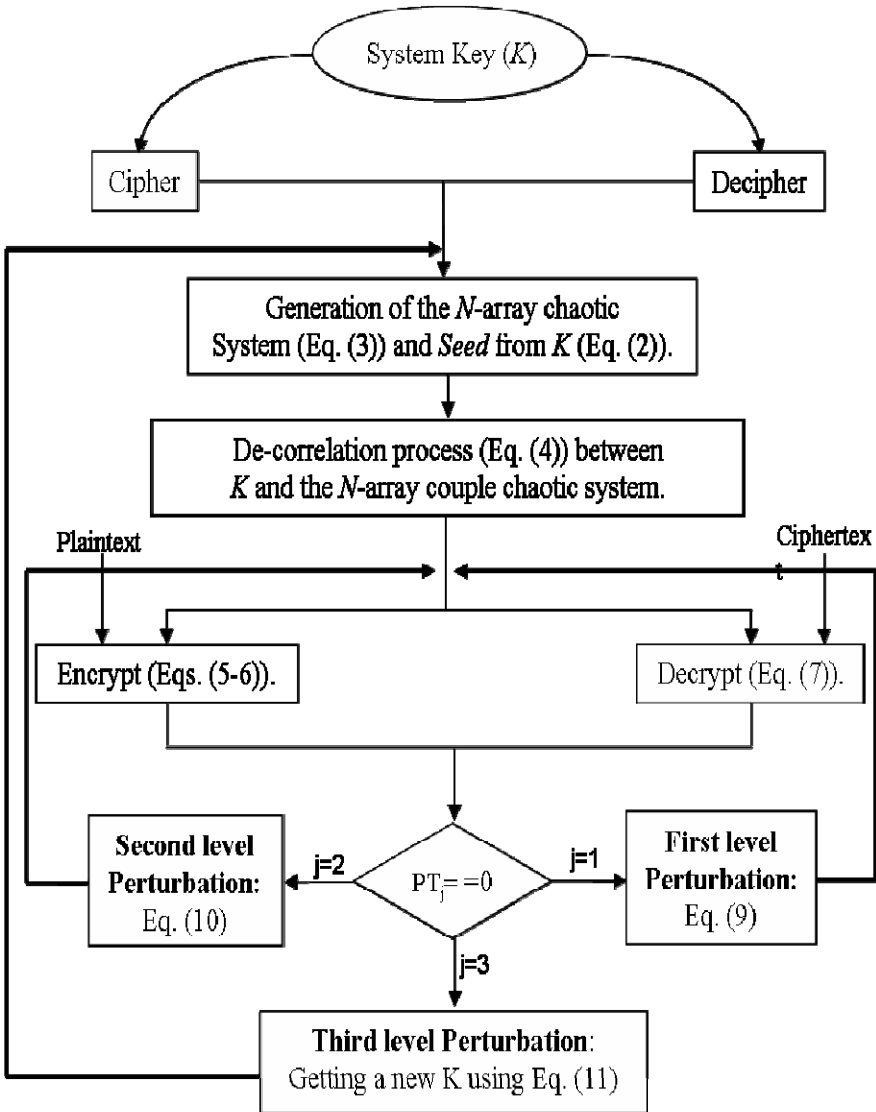
Fig. 1 Flow diagram of the Chaotic Map based Encryption/Decryption scheme.

The addition of system interdependencies is important for the overall security of the system. The general idea is as follows. Maps inter-dependency is created by taking a small fraction of the global feedback (total history of previous ciphertext values) during the encryption process to perturb the trajectories of the N maps. This implies that, any bit change in the plaintext will be propagated into every single ciphertext and chaotic map (with now different dynamics). This represents the first level of the hierarchical perturbation. The second level perturbation is

more drastic in the sense that it completely changes the system map variables (and parameter if specified). These two perturbations are necessary to immediately change the dynamics of the system when it gets trapped in a fixed point or short length periodic window. The third level perturbation represents a whole system perturbation, where the original system-key is updated. A general diagram of the scheme is presented in Fig.1. A detailed description of the scheme in [9] is presented in the following subsections.

2.1.1 Logistic Map

The scheme in [9] is based on the logistic map represented by:

$$X_n = \lambda X_{n-1}(1 - X_{n-1}), \qquad \lambda \in [1,4], \quad X \in [0,1] \tag{1}$$

with corresponding bifurcation diagram depicted in Fig.2. As the parameter λ increases from 1 to 4, the map experiences a period doubling to chaos [25]. In particular for $\lambda \geq 3.5699$ (known as accumulation point) it presents a chaotic behaviour, however there are many periodic windows (with all kind of periods) that appear abruptly. A very well known and prominent period-3 window appears at $\lambda = 1 + \sqrt{8} = 3.8284$. Short period windows of the logistic map are avoided during the encryption process, because they reveal statistical information useful for attackers to break into the system. This problem can be alleviated by perturbing the cycling chaotic signal with period T every Δ iterations, for $\Delta \leq T$ [14]. The perturbation drives the signal away from its cycle after I number of iterations, where I depends directly on the perturbation magnitude [1]. Therefore, the new period in the perturbed cycle becomes $\sigma\Delta(2^L -1) >> 2^L$, where σ is a positive integer and Δ is the perturbation period. Fixed points ($f(X) = X$) are also present at $X = 0$ and $X = (\lambda - 1)/\lambda$, which define a regular pattern in the logistic map [15].
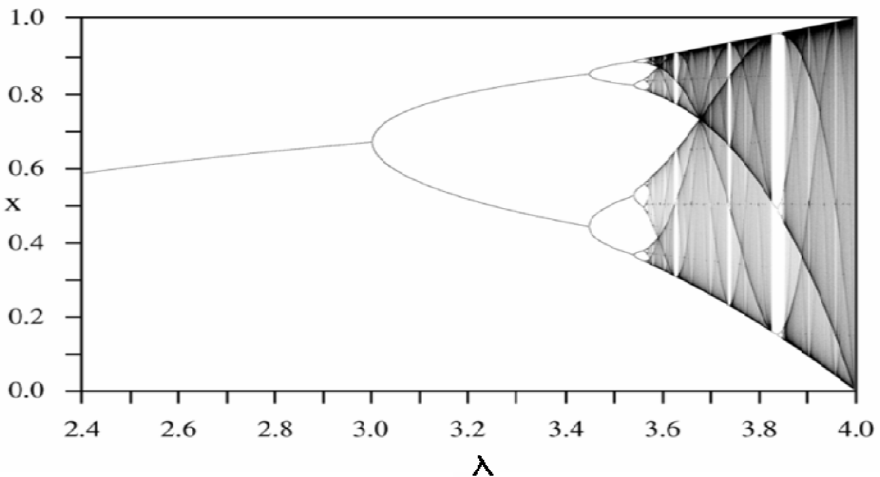

Fig. 2 Logistic map bifurcation diagram

2.1.2 System Initialization

Hasimoto's scheme [9] is symmetric, that is the system-key (K) of size B bits, for $B \geq 128$ and multiple of $n = 32$ is shared between cipher and decipher. K is employed for both the initialization of logistic maps (variables and parameters) and a system *Seed* for a Pseudo-Random Number Generator (PRNG). A PRNG is used to generate all random variables supporting the N-map system, such as current number of active maps, initial global and local feedbacks, and system perturbation frequencies. Note that any good PRNG can be used in the scheme.

The value of *Seed* is calculated as follows:

$$Seed = K_n(1) \oplus ... \oplus K_n(B/n) \qquad (2)$$

where $K_n(i)$ is the ith n-bit element of K (considered as an array of B/n elements) and $\oplus$ is the eXclusive-OR (XOR) operator. The *N-array* of chaotic maps for $N = B/n$ is defined as (see Fig.3):

$$X_{i,0} = K_{n/2}(2i-1)/2^{n/2},$$

$$\lambda_i = 3.68 \; + \; \frac{[K_{n/2}(2i)/2^{n/2} + K_{n/2}(2i)/10^{h_{n/2}} + (a \oplus b)/2^{n/4}]}{10} \cdot \frac{[0.3187]}{MAX}, \qquad (3)$$

$$i = 1, 2, ..., N$$

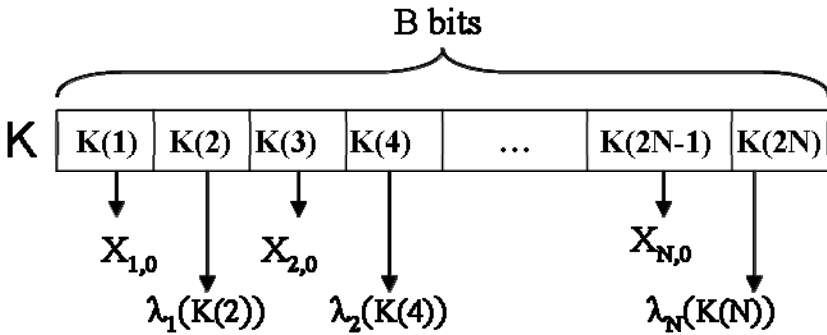**Fig. 3** System-key (K) partition for the creation of N variables ($X_{i,0}$) and corresponding parameters(λ_i).

where $X_{i,0}$ and λ_i are the ith map initial variable and parameter respectively, with $0.2 \leq X_{i,0} \leq 0.8$ (except $X_{i,0} \approx 0.5$) and $3.68 \leq \lambda_i \leq 3.9987$ for $\lambda_i \neq \lambda_j$ and $i \neq j$, $h_{n/2}$ is the number of digits in the largest decimal number represented by $n/2$ bits, $a \oplus b$ term is the XOR between the half-most and half-least significant bits

of $K_{n/2}(2i)$ respectively, yielding an *n/4* bits outcome, and *MAX* is the maximum value of $[K_{n/2}(2i)/2^{n/2} + K_{n/2}(2i)/10^{h_{n/2}} + (a \oplus b)/2^{n/4}]/10$. The valid interval for λ is set between the merge point of the two main bifurcation bands found in the interval $3.0 < \lambda < 3.68$ and (strictly speaking) the highest real number less than 4.0 represented by the precision of the corresponding machine.

In order to increase the sensitivity of the system to a magnitude change in the system-key, $X_{i,0}$ is iterated an *RT=PRNG(Seed)* random number of times over a Network of Coupled Chaotic Maps (NCM) governed by a coupling transformation over some defined neighborhood in the array [9]. New states represent the weighted interaction between each individual map (local term) and the coupling transformation (linear/nonlinear interaction term). When the weight of the coupling is weak, the system can be regarded as a local map perturbed by contributions from other sites, thus maintaining its main individual properties. On the other hand, when the weight of the coupling is large, the system reaches an asymptotic collective behavior characterized by intermittent periodic chaotic cycles (cycling chaos). Dellnitz, 1995 [16], found that when an individual map is active (presents chaotic behavior); the rest of the system elements remains quiescent.

The NCM is defined as:

$$X_{i,j} = (1-\varepsilon)[\lambda_i X_{i,j-1}(1 - X_{i,j-1})] + \varepsilon H(X_{1,j-1},...,X_{N,j-1}),$$

$$H(X_{1,j-1},...,X_{N,j-1}) = \frac{1}{N}\sum_{i=1}^{N} X_{i,j-1} \tag{4}$$

where *j* is the current map state iteration and *H* is the coupling function with coupling parameter ε . *H* takes the average of previous iteration map variables over all maps. Eq. (4) guarantees that a one-bit change in *K*, will affect all maps variables and therefore the system's output (ciphertext). The output of Eq. (4) after *RT* iterations becomes the initial state for each map in the encryption process, that is $X_{i,0}$, $1 \le i \le N$.

2.1.3 Cipher/Decipher Scheme

The *N-map* array is used as an *N* circular list of ciphers, but only a subset defined by a cipher window $W_{m,k} = \{((i-1) \mod N)+1 | k \le i \le k+m-1\}$, for $m = [PRNG(Seed) \mod N]+1$ and $k \in \{1,2,...,N\}$ is considered in the encryption process at a time (Fig.4). *k* represents the minimum index of $W_{m,k}$ at a given time and $1 \le m \le N$ is the size of the window. For simplicity, the elements of $W_{m,k}$ are renamed by *W(i)*, for $1 \le i \le m$ and some *N*. *W(i)* represents the ith element of $W_{m,k}$, i.e. if $W_{3,5}$ = *{5,1,2}* and *N=5*, then *W(1)=5*, *W(2)=1*, *W(3)=2*.
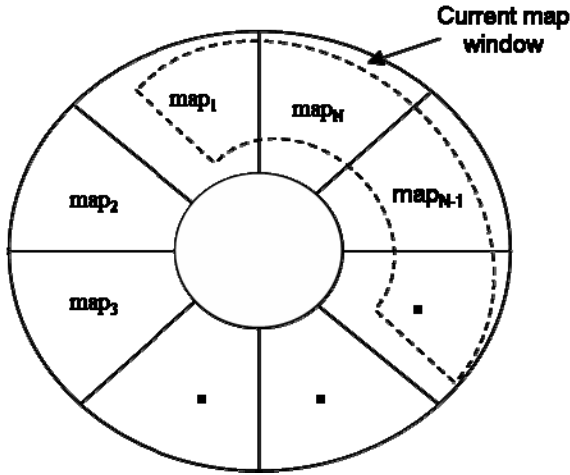
Fig. 4 Array of *N* chaotic maps viewed as a circular list. Current map window represents the active maps in the encryption process.

Let *j* and *l* represent the system state (iteration) and plaintext-ciphertext absolute indexes respectively $(l = (j-1)m+i)$. For a fixed state *j*, the *m* ciphers in $W_{m,k}$ are defined by the following equation:

$$C_l = C_{W(i),j} = ([P_l + X'_{W(i),j}] \mod 2^n) \oplus X'_{W(i),j}$$

$$\oplus ([X'_{W((i+1) \mod m),j} + X'_{W((i+2) \mod m),j}] \mod 2^n) \qquad (5)$$

$$\oplus ([C_{W(i-1),j} + C_{W(i),j-1}] \mod 2^n),$$

$$1 \le i \le m, \quad l = (j-1)m+i$$

where P_l is the lth plaintext input, $X'_{W(i),j}$ is the corresponding integer representation of $X_{W(i),j}$ using *n* bits, $C_{W(i-1),j}$ $(= C_{l-1})$ is the previous ciphertext output $(i-1)$ in current iteration (j^{th}), and $C_{W(i),j-1}$ is the previous ciphertext output of the same i^{th} map, but from the *j-1* iteration. $C_{W(i-1),j}$ and $C_{W(i),j-1}$ represent the global and local feedback respectively (Fig.5). The initial global feedback $C_{W(i-1),j}$ for the jth iteration takes in the last ciphertext output of the previous iteration $(C_{W(m),j-1})$ to spread the system changes on to future ciphertexts and current *m* logistic map variables (see next subsection). A total of *mn* bits are encrypted per iteration state *j* (*n* encrypted bits per map). $W_{m,k}$ is periodically rotated one map at a time by setting

$k = k + 1$ (see Fig.4); when $(N-k+1) < m$, the cipher index wraps around taking the corresponding first, second, up to the $m-(N-k+1)$ initial maps (when $k=N$ the current cipher window is $W_{m,N} = \{N, 1, 2, ..., m-1\}$).

To increase the encryption system security, ciphertext output $C_l = C_{W(i),j}$ is masked using two maps' variables:

$$C_l^M = (C_l + X'_T) \mod 2^n, \quad X'_T = X'_{W(i),j} \oplus X'_{W((i+1)\bmod m),j} \tag{6}$$

Therefore, decipher cannot use C_l^M directly to find its corresponding plaintext data, it needs to know X'_T.
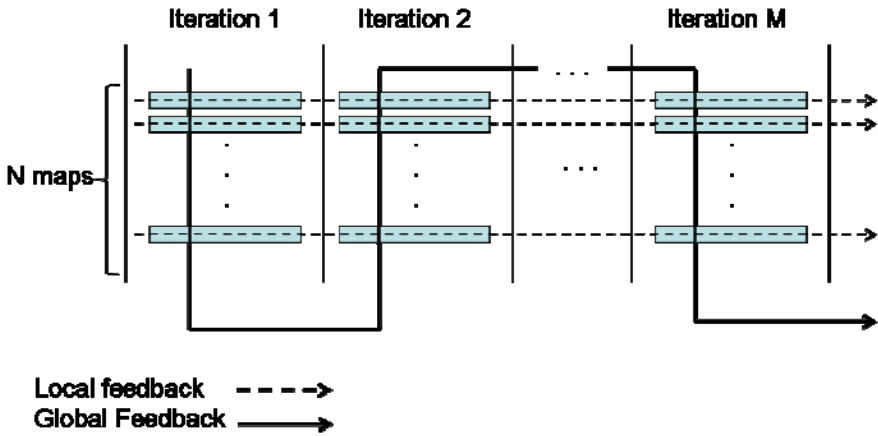
Fig. 5 Diagram representation of global and local feedback

The corresponding decryption system can be written as:

$$C_l = (C_l^M - X'_T) \mod 2^n;$$

$$P_l = [C_l \oplus X'_{W(i),j} \oplus ([C_{W(i-1),j} + C_{W(i),j-1}] \mod 2^n)$$

$$\oplus ([X'_{W((i+1)\bmod m),j} + X_{W((i+2)\bmod m),j}] \mod 2^n) - X'_{W(i),j}] \mod 2^n, \tag{7}$$

$$1 \le i \le m, \quad l = (j-1)m + i$$

Initial global and local feedbacks are calculated by:

$$global = C_{0,0} = PRNG(Seed);$$
$$for \quad i = 1, ..., N \tag{8}$$
$$local_i = C_{i,0} = PRNG(Seed);$$

2.1.4 Three-Level Perturbation Scheme

Under external perturbations (plaintext or system-key attacks), the global feedback
in Eq. (5) will drive the original system trajectory into a different chaotic state.
There are two problems though; the transition change is slow and the chaotic sys-
tem (*N-array*) does not participate in the trajectory change (chaotic parameters
and variables stay unchanged). In order to speed up the system reaction time under
external perturbations, a three-level periodic perturbation scheme is proposed. The
first two perturbation levels are related to the system variables and the third one is
related to the system-key. In the first perturbation level, the trajectory of every
map is slightly modified to increase its cycle length [14, 17]; in the second pertur-
bation level the current system variable is randomly changed creating a totally
new trajectory for the system; and, in the third perturbation level the system-key
value is renewed using current system map variables. Third-level perturbation
represents a reset operation, since the entire encryption/decryption system parame-
ters are completely modified.

The first-level perturbation for the ith logistic map is expressed as:

$$XP_{W(i),j-1} = X_{W(i),j-1} + \frac{\sum_{l=1}^{n/8} C_{W(m),j-1}(l)}{10^{h_8}}, \qquad 1 \le i \le m \qquad (9)$$

where $C_{W(m),j-1}(l)$ is the lth byte of the global feedback at the state *j-1*, and h_8 is the
number of digits in the largest decimal number represented by *8* bits. We post-
process $XP_{W(i),j-1}$ so that its first digit after the decimal point remains the same as
in $X_{W(i),j-1}$; therefore $abs(XP_{W(i),j-1} - X_{W(i),j-1}) < 10^{-1}$ (the perturbation signal must
be smaller than the chaotic signal to keep the good statistical properties of chaos
dynamics). Note that every single changed detected by the global feedback, it is
passed on to the array of chaotic maps. In the case of a differential attack, Eq. (9)
exacerbates every single plaintext change by disturbing not only future ciphertext
outputs through global and local feedback, but also the map variables in the cur-
rent cipher window. The combined effects (feedback and perturbation) generate
different trajectories for any pair of plaintexts when iterated by the system. An ad-
ditional benefit of involving $C_{W(m),j}$ in the perturbation process, is that in the long
run it has uniform distribution (see section 3), an important requirement for per-
turbation schemes [14].

Only one chaotic map may be enough if the chaotic signal reaction to Eq. (9)
were instantaneous. Unfortunately, it takes a certain number of iterations (depend-
ing on the perturbation magnitude) for a chaotic map to diverge from its original
signal trajectory. This reaction time may be dangerous under a differential or
plaintext attack if only one map is used in the encryption process (attacker may
find out current map's parameters). To avoid this problem in one-dimensional
chaotic encryption systems, Pareek, et.al, [18] iterated the logistic map a random
number of times. This solution affected considerably the system execution time

and did not solve the security problem since the scheme was broken by [19]. The use of *m* different chaotic maps producing *m* ciphertext values increases the number of variables to solve for the attacker during the reaction time without affecting the system's performance. Since we are not considering all possible chaotic flaws in the logistic map, in the low probable case where the first-level perturbation magnitude is very small for the entire *m-array* of chaotic maps (increasing the chaotic reaction time) or when the *m-array* is in a short cycle state, then a second-level perturbation comes in to play to complicate things up for the attacker by re-setting the system map variables. This is the same as resetting the maps' variables maintaining the same parameters.

The second-level perturbation adds $C_{W(m),j-1}$ to each map variable and cross-iterate the outcome throughout the maps. For the $W(i)^{th}$ map in state *j-1*, the new perturbed system variable $XP_{W(i),j-1}$ is obtained by:

$$
\begin{aligned}
&For \quad i = 1, m \\
&\quad \gamma = (C_{W(m),k} + X_{W(i),j-1}) - floor(C_{W(m),k} + X_{W(i),j-1}) \\
&\quad For \quad l = 1, m \\
&\qquad \gamma = \gamma.\lambda_{W(l)}.(1-\gamma) \\
&\quad XP_{W(i),j-1} = \gamma
\end{aligned} \tag{10}
$$

That is, new system variables are influenced by all maps in $W_{m,k}$ and their corresponding local feedback. For the next iteration $X_{W(i),j-1} = XP_{W(i),j-1}$.

Since we are working with very long multimedia sequences (from minutes to hours) and not checking for either bad chaotic points (periodic window, fixed points, etc.) or perturbation frequency values, the system may face low chaotic variation due to the low perturbation frequencies and/or small perturbation magnitudes in the first and second perturbation levels. To avoid this kind of vulnerability, the third-level perturbation enters in action by replacing the system-key using current map variables:

$$
K = K \oplus concatenate[X'_{1,j}, \cdots, X'_{N,j}] \tag{11}
$$

Immediately, the system (under a differential attack) turns into a chaotic behavior and the system becomes protected. By using current map variables in the new K, it is assured that any single change in the past be spread out into future ciphertext generations. The new system-key goes through the same process as in the original one (see Eq. (2)), including chaotic parameter and variable restrictions.

The perturbation cycles represented by PT_l, $1 \le l \le 3$, are selected randomly to increase the system-key space in the case of brute force attack (the opponent tries every possible system-key combination until the right one is found). We define the perturbation cycles as follows: $PT_1 = [PRNG(Seed)\ modulus\ 10] + 15$, $PT_2 = n_1 .PT_1$, and $PT_3 = n_2 .PT_2$, for $n_1 = [PRNG(Seed)\ modulus\ 64]+2$ and $n_2 = [PRNG(Seed)\ modulus\ 128]+3$. Less frequent perturbations have greater impact on the system's parameters. The value of PT_1 is related to the sensitivity of the

logistic map to a magnitude change of $1/2^8$ in the initial condition. For $B=128$ bits, the minimum magnitude change of two system variables is $\sim 10^{-3}$, which requires about 10-15 map iterations for their trajectories to diverge chaotically [18]. This result is important for the cipher in order to produce different trajectories when input values differ in the least significant bits.

2.1.5 Security Analysis

The scheme described above has the following characteristics:

a) Distribution of ciphertexts is uniform and independent of the input plaintext distribution (Fig.6).
b) Different systems keys and different plaintext produce totally different ciphertext, as shown in Figs. 7 and 8, respectively. On the average, 99.6% of the total bytes and 50% of the total bits are changed during the encryption process, fulfilling a basic requirement for secure cryptosystems.
c) Plaintext and corresponding ciphertext are statistically uncorrelated.

In the case of a brute force attack, a search of at least 2^B key possibilities is needed for $B > 128$ bits.
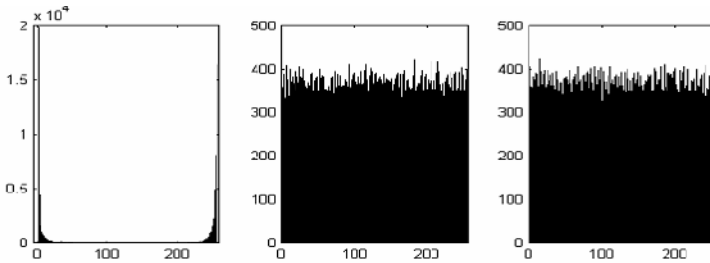

Fig. 6 Histogram of plaintext (left column) and corresponding ciphertext for two different system-keys (center and right columns).
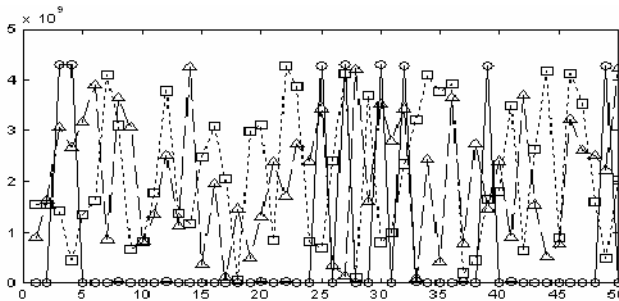

Fig. 7 Sensitivity to system-key changes. Plaintext (circled continuous line) encrypted with two slightly different system-keys (least significant bit changed).
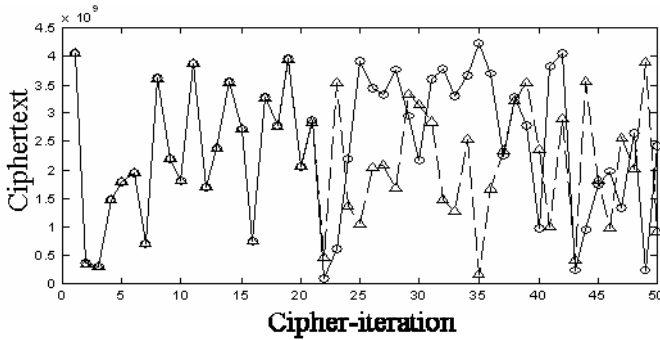
Fig. 8 Sensitivity to plaintext changes without perturbation scheme. Ciphertexts produced by chosen plaintexts with the least significant bit changed.

2.2 *Advanced Encryption Standard (AES)*

AES is the encryption standard adopted by the U.S. government in 2001, for the encryption of electronic data. It is based on the original proposal of Rijndael [20] with cryptographic keys of 128, 192, and 256 bits to encrypt and decrypt data in blocks of 128 bits. Following the description in [21], AES operates on an internal state of 128 bits, which is initially set to the plaintext block, and after transformations, becomes the output ciphertext block. The state is organized in a *4x4* array of 8-bit bytes, which is transformed according to a round function N_r times. The number of rounds is $N_r = 10$ for 128-bit keys, $N_r = 12$ for 192-bit keys, and $N_r = 14$ for *256-bit* keys. In order to encrypt, the state is first initialized, then the first 128-bits of the key are xored into the state, after which the state is modified $N_r - 1$ times according to the round function, followed by the slightly different final round.

The round function consists of four steps: SubBytes, ShiftRows, MixColumns and AddRoundKey␣(except for the final round which omits the MixColumns step). Each step operates on the state, at each round *r*, as follows:

1. SubBytes: substitutes every entry (byte) of the state with an S-box entry,
2. ShiftRows: cyclically left shifts every row i of the state matrix by i; $0 \leq i \leq 3$,
3. MixColumns: multiplies each column, taken as a polynomial of degree less than 4 with coefficients in F_{2^8}, by a fixed polynomial modulo $x^4 + 1$,

4. AddRoundKey: XORs the r-th round key into the state.

Each transformation has an inverse from which decryption follows in a straightforward way by reversing the steps in each round: AddRoundKey (inverse of itself), InvMixColumns, InvShiftRows, and InvSubBytes.

The key expansion into the Nr 128-bit round keys is accomplished using a key scheduling algorithm, the details of which can be found in [21]. The

design of the key schedule allows for the full expansion to precede the round transformations, which is advantageous if multiple blocks are encrypted using the same key, while also providing the option for on-the-y key generation. On-the-fly key generation proves useful in memory constrained environments such as microcontrollers.

For 32-bit (and greater word length) processors, in [20] Daemen and Rijmen detail a fast implementation method that combines the SubBytes, ShiftRows, and MixColumns transformations into four 256-entry (each entry is 4 bytes) lookup tables, Ti, $0 \leq i \leq 3$. Following [21], the "T-table" approach reduces the round transformations to updating the j-th column according to:

$$\left[S_{0,j}^{'}, S_{1,j}^{'}, S_{2,j}^{'}, S_{3,j}^{'} \right]^{T} = \oplus_{i=0}^{3} T_i \left[S_{i,j+C_i} \right], \quad for \quad 0 \leq j \leq 3 \tag{12}$$

Where $S_{j,k}$ is the byte in the jth row and k^{th} column of the state, and C_i is a constant equivalently doing the ShiftRows in-place. After the columns are updated, the remaining transformation is AddRoundKey (which is a 4-byte look-up and xor per column). Note, however, since the T_i's are simply rotations of each other, some implementations of (1) benefit from using a single table and performing the necessary rotations.

2.2.1 Berstein's Implementation of AES

It is important to point out that AES implementations are targeted to specific CPU architectures. In [11], authors describe a new AES implementation that takes advantage of the architecture-dependent reduction of instructions used to compute AES and the microarchitecture-dependent reduction of cycles used for those instructions. Authors report new software speed records than previous AES implementations for different CPUs: Motorola PowerPC G4 7410, ppc32 architecture, Intel Pentium 4 f12, x86 architecture, Sun UltraSPARC III, sparcv9 architecture, Intel Core 2 Quad Q6600 6fb, amd64 architecture, and AMD Athlon 64 X2 3800+ 15/75/2, amd64 architecture. Even though no new techniques were proposed, Berstein scheme combines previously known techniques to reduce the number of CPU integer instructions, load instructions, etc used for AES. In particular, they employed (whenever possible) a combined shift-and-mask instructions, combined second-byte extraction instructions, padded instructions, combined load-XOR, etc. to reduce the overall number of instructions required by AES.

Along with Berstein's implementation, two other AES implementations are employed in the performance evaluation, Gladman's [12] and PolarSSL [13] implementations. In particular, PolarSSL is a light-weight open source cryptographic and Secure Socket Layer/Transport Layer Security (SSL/TLS) library written in C language with embedded systems in mind. It has been ported to a large number of architectures, including ARM, Power-PC, MIPS and Motorola 68000 [13].

3 Performance Evaluation

3.1 Methodology

To evaluate the performance of different encryption schemes reviewed in the previous section, we have developed implementations of these schemes on the following computing platforms: Dell desktop, Lenovo laptop, Asus netbook, Nokia N800 and N900, all running a 32-bit architecture. Table 1 shows specifications of each platform, the asterisk means a testbed with floating-point processor unit. These platforms are chosen to reflect diversity in computation energy, battery capacity, and mobility characteristics. We evaluate and compare each scheme in terms of CPU usage and encryption speed.

Table 1 Specifications of Different Platforms used in Experiments.

Testbed	CPU Type	Clock Speed	Memory	Operating System
*Desktop	Intel DuoCore 2	2.2 Ghz	3 GB	Ubuntu 8.3
*Laptop	Intel DuoCore 2	2.2 Ghz	2 GB	Ubuntu 9.1
*NetBook	Intel Atom	1.6 Ghz	1 Gb	Ubuntu Netbook
Nokia N800	TI Omap 2420	333 Mhz	128 MB	Maemo
*Nokia N900	TI Omap3 3430	600 Mhz	256 MB	Maemo OS 5

We installed Ubuntu Linux distribution on all devices except Nokia N800 which runs Maemo [22] distribution. Maemo is a mobile operating system for Nokia PDAs based on Debian GNU/Linux operating system. Ubuntu implements battery management using the Advanced Configuration and Power Interface (ACPI), which exports battery data via the /proc/acpi/battery file system. The data values exported by ACPI expressed by millivolts and milliamps which can be converted to Watt. Given the current voltage and amps of battery using ACPI values, we compute energy consumed (in Watt Hour).

We conducted energy consumption tests on two devices only, i.e., laptop and netbook, as these are equipped with batteries, while we conducted encryption speed and CPU usage on all the four devices. We could not perform energy consumption tests on Nokia devices due to unavailability of power monitoring tools. Table 2 shows battery specification of laptop and netbook used in energy drain tests.

Table 2 Features of Rechargeable Batteries used in Experiments

	Laptop	**Netbook**
Battery Capacity	4752 mAh	4400 mAh
Voltage	11100 mV	11100 mV
Type	Lion	Lion

We adopted the following methodology to measure energy consumption. Each device is first fully charged, and each encryption schemes is modified so that it runs in an infinite loop. For each scheme, we periodically polled (every 60 seconds) the Linux ACPI values and computed energy usage. To measure the actual energy used by encryption operation, each device is first fully charged and then left on idle running. We periodically polled (every 60 seconds) the Linux ACPI values and computed the energy consumption. The difference of the idle energy and energy consumed during encryption operation is reported as the energy consumed by the encryption scheme.

3.2 Results

Berstein's implementation of AES accepts a key size of 128-bits. The initial set up for Hasimoto's chaotic schemes is a key size of *B=384* bits, generating a *12-map* array of logistic maps, where each map encrypts *n=32* bits (*N=384/32=12* maps) with double feedback for the dispersion of plaintext changes. The scheme only includes the first level perturbation scheme.

For the sake of clarity, speed and CPU usage on each platform are analyzed first, followed by battery dissipation and CPU usage. Fig.9 compares the CPU usage at a constant rate for different streaming rates; this means that all implementations are processing the same amount of data over the same period of time. The Chaotic encryption is ~4 times (400%) better in terms of CPU usage than Bernstein's and Gladman's AES implementations for the netbook and laptop platforms; on the desktop Bernstein's AES improved its performance, but still ~2 (200%) times worse than chaotic encryption. OpenSSL's performance was poor in all the tests. As shown in Fig.10, the Chaotic encryption is by far the fastest scheme on mid-level systems (desktop, laptop and netbook), with an average of 2.6 (260%) times better performance than the fastest AES implementation as of 2008 [11] (Bernstein). On Nokia N800 platform, Gladman's obtained the best performance. Among AES implementation, Bernstein's implementation is clearly the best choice for laptops and desktops, while Gladman's implementation for netbooks and both Nokia's architectures N800 and N900. This is in accordance to the platforms both implementations were targeted to.
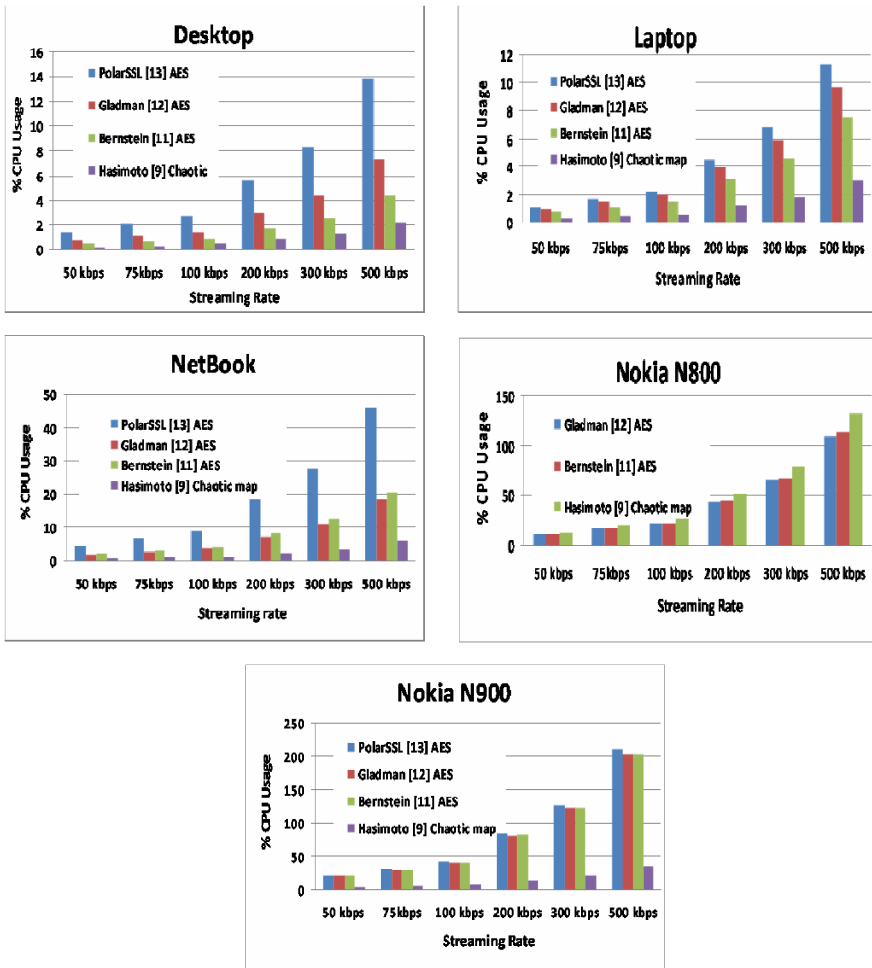
Fig. 9 Performance results on different platforms in terms of encryption CPU usage.

In the Nokia N800 device, Gladman's was the best, closely followed by Bernstein's and Hasimoto's chaotic scheme. We were expecting worse numbers for chaotic encryption since the core of the process consist of floating-point computation, and Nokia N800 lacks math coprocessor in its TI Omap2420 CPU. For Nokia N900, Hasimoto's scheme takes again the lead with more than 300% better performance than Berstein's and Gladman's implementations. The performance of the chaotic scheme increased ~21 times from the Nokia N800 to N900, versus ~5 times for Bersteins's and Gladman's implementations. This result reveals the dependency of chaotic schemes on hardware based floating-point computation.
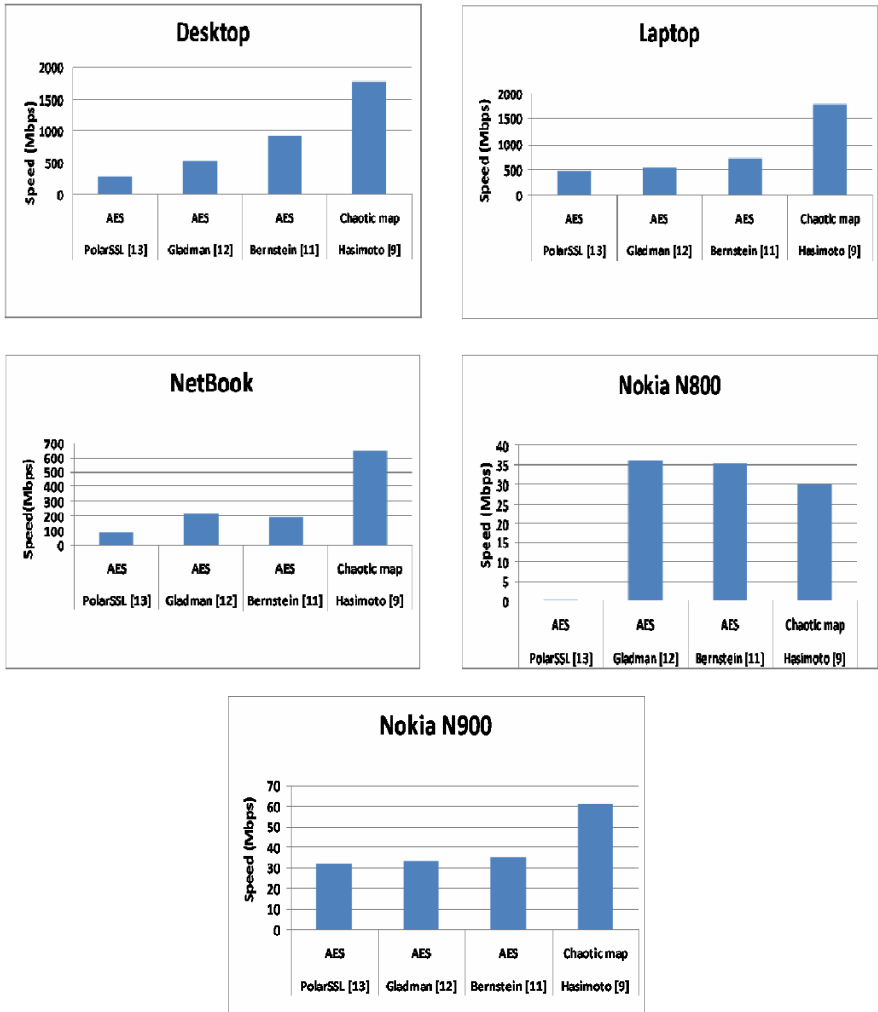
Fig. 10 Performance results on different platforms in terms of encryption speed.

Fig.11 shows the battery energy consumption on battery operated devices. Fig.11a shows the performance of encryption schemes in terms of energy usage and the amount of data encrypted. A linear behavior for all implementation is observed with AES implementation having the greatest slopes or energy consumption. For example, Bernstein's implementation encrypts 280 GBytes of data and consumes over 32 Watt Hour of battery energy. On the other hand, chaotic scheme encrypts 800 GBytes of data and consume in the same 32 Watt Hour of the energy. Fig. 11b shows energy consumed for different data sizes. Overall, Chaotic encryption can process 300%-400% more information with the same energy consumption than the AES implementations. While the trend is similar across

platforms, the differences are much sharper in the case of netbook. We attributed this fact to the energy characteristics of the Atom processor that is more optimized for the energy. These results show that choice of the encryption scheme for a desired rate should depend on the hardware and battery capabilities of the platform.
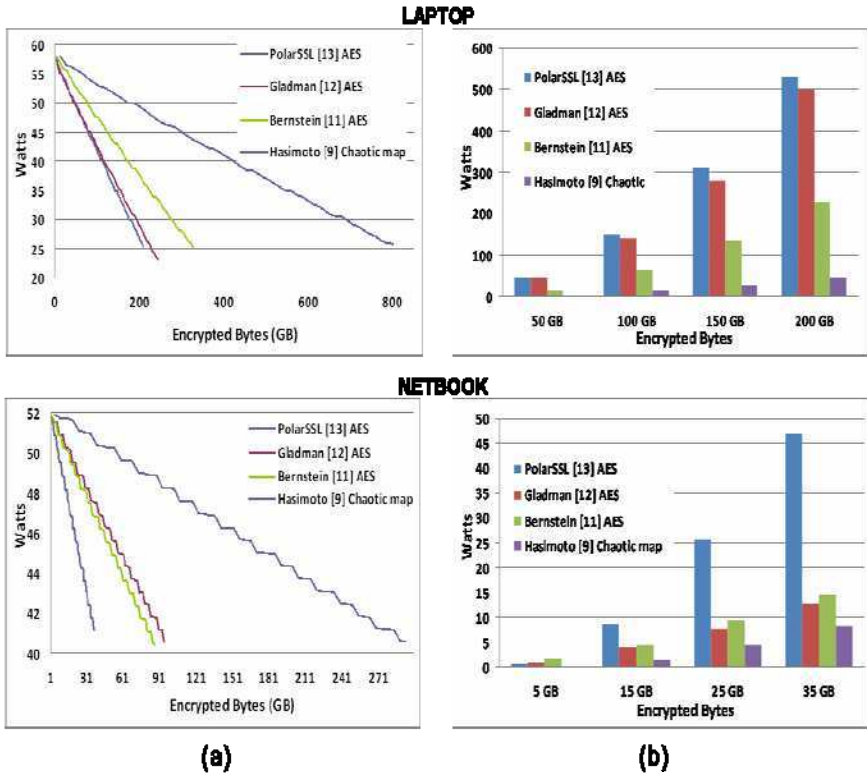

Fig. 11 Performance results in terms of power consumed: a) encryption running for a fixed time period (60 minutes) and corresponding bytes encrypted, b) encrypting different data sizes.

4   Final Remarks

We have evaluated conventional (AES) and chaotic encryption schemes in terms of encryption speed, CPU usage, and battery consumption for different platforms having different characteristics. These included from high-end desktops to handheld devices. Both encryption technologies can deal with real-time multimedia communications, but chaotic encryption performance was surprisingly good, outperforming latest implementations of AES. Even though chaotic encryption in [9] may not offer a robust security as provided by AES, it represents an excellent tradeoff between the security and performance needed in handheld devices and multimedia encryption servers (where many streams need to be encrypted). For AES

to get the reported performance, it needs to be optimized for a particular CPU architecture; chaotic encryption program in [9] on the other side was not optimized and the same program was used for all platforms.

For best results, chaotic encryption needs a hardware-based floating point unit. This requirement can be reduced if chaos based encryption process is combined with other useful techniques such as permutation as in [6].

References

1. Li, S., Mou, X., Ji, Z., Zhang, J., Cai, Y., Ji, Z., Zhang, J.: On the security of a chaotic encryption scheme: problems with computerized chaos. Comput. Phys. Commun. 153(1), 52–58 (2003)
2. Li, S., Chen, G., Mou, X.: On the dynamical degradation of digital piecewise linear chaotic maps. Int. Journ. Bifurcat. Chaos 15(10), 3119–3151 (2005)
3. Binder, P.M., Jensen, R.V.: Simulating chaotic behavior with finite-state machines. Physical Review A 34(5), 4460–4463 (1986)
4. Álvarez, G., Li, S.: Some basic cryptographic requirements for chaos-based cryptosystems. Int. J. Bifurcat. Chaos 16(8), 2129–2151 (2006)
5. Kocarev, L.: Chaos-based cryptography: a brief overview. IEEE Circ. Syst. Mag. 1(3), 6–21 (2001)
6. Al-masalha, F., Khokhar, A., Hasimoto-Beltran, R.: Scalable Encryption of Variable Length Coded Video Bit Streams. In: The 35th IEEE Conference on Local Computer Networks, LCN 2010 (2010)
7. Li, S., Zheng, X., Mou, X., Cai, Y.: Chaotic encryption scheme for real-time digital video. In: Real-Time Imaging VI. Proc. of the SPIE, vol. 4666, pp. 149–160 (2002)
8. Lian, S., Sun, J., Wang, Z., Dai, Y.: A fast video encryption scheme based-on chaos. In: 8th IEEE International Conference on Control, Automation, Robotics and Vision (2004)
9. Hasimoto-Beltran, R.: High-performance multimedia encryption based on chaos. Chaos 18, 023110 (2008)
10. Perrig, A., Szewczyk, R., Wen, V., Culler, D., Tygar, J.D., Eason, G.: SPINS: Security Protocols for sensor Networks. Wireless Networks 8, 521–534 (2002), doi:10.1023/A:1016598314198
11. Bernstein, D.J., Schwabe, P.: New AES Software Speed Records. In: Chowdhury, D.R., Rijmen, V., Das, A. (eds.) INDOCRYPT 2008. LNCS, vol. 5365, pp. 322–336. Springer, Heidelberg (2008)
12. Gladman, B.: AES and combined encryption/authentication modes (2006), http://fp.gladman.plus.com/AES/
13. PolarSSL, http://polarssl.org
14. Sang, T., Wang, R., Yan, Y.: Perturbance-based algorithm to expand cycle length chaotic key stream. Electron. Lett. 34, 873 (1998)
15. Clinton, J.S.: Chaos and Time-series Analysis. Oxford University Press, Oxford (2006)
16. Dellnitz, M., Field, M., Golubitsky, M., Hohmann, A., Ma, J.: Cycling chaos. IEEE Trans. Circuits Syst. I: Fundam. Theory Appl. 42, 821 (1995)
17. Cernak, J.: Digital generators of chaos. Phys. Lett. A 214, 151 (1996)

18. Pareek, N.K., Patidar, V., Sud, K.K.: Discrete chaotic cryptography using external key. Phys. Lett. A 309, 75 (2003)
19. Álvarez, G., Montoya, F., Romera, M., Pastor, G.: Cryptanalysis of a discrete chaotic system using external key. Phys. Lett. A 319, 334 (2003)
20. National Institute of Standards and Technology (NIST): FIPS-197: Advanced Encryption Standard (AES), `http://www.csrc.nist.gov/publications/fips/fips197/fips/-197.pdf`
21. Osvik, D.A., Bos, J.W., Stefan, D., Canright, D.: Fast software AES encryption. In: Hong, S., Iwata, T. (eds.) FSE 2010. LNCS, vol. 6147, pp. 75–93. Springer, Heidelberg (2010)
22. Maemo, `http://maemo.org/`
23. Hu, H., Xu, Y., Zhu, Z.: A method of improving the properties of digital chaotic system. Chaos, Solitons Fractals 38, 439–446 (2008)
24. Cernak, J.: Digital generators of chaos. Phys. Lett. A 214, 151–160 (1996)
25. Chen, G., Mao, Y., Chui, C.K.: A symmetric image encryption scheme based on 3D chaotic cat maps. Chaos, Solitons Fractals 21, 749 (2004)
26. Fridrich, J.: Symmetric ciphers based on Two-dimensional chaotic maps. Int. J. Bifurcation Chaos 8, 1259–1284 (1998)
27. Mao, Y., Chen, G., Lian, S.: A novel fast image encryption scheme based on the 3D chaotic baker map. Int. J. Bifurcat. Chaos 14(10), 3613–3624 (2004)
28. Xu, D., Li, Z., Bishop, S.R., Galvanetto, U.: Estimation of periodic-like motions of chaotic evolutions using detected unstable periodic patterns. Patt. Recog. Lett. 23, 245–252 (2002)
29. Davidchack, R.L., Lai, Y.-C.: Efficient algorithm for detecting unstable periodic orbits in chaotic systems. Phys. Rev. E 60, 6172–6175 (1999)
30. Saiki, Y.: Numerical detection of unstable periodic orbits in continuous-time dynamical systems with chaotic behavior. Nonlin. Processes Geophys. 14, 615–620 (2007)
31. Pei, X., Dolan, K., Moss, F.: Counting unstable periodic orbits in noisy systems: A scaling relation connecting experiment with theory. Chaos 8, 853–860 (1998)

Author Index