UNIVERSIDAD DE INVESTIGACIÓN DE TECNOLOGÍA EXPERIMENTAL YACHAY

Escuela de Ciencias Matemáticas y Computacionales

TÍTULO: Dynamical Networks with Global Interactions: Communication, Encryption, and Collective Behavior

Trabajo de integración curricular presentado como requisito para la obtención del título de Ingeniero en Tecnologías de la Información

Autor:

Flor Peñafiel Mario Francisco

Tutor:

Cosenza Miceli Mario Giuseppe, PhD

Co-Tutor:

Acosta Orellana Antonio Ramón, PhD

Urcuquí, Mayo 2021

Urcuquí, 28 de mayo de 2021

SECRETARÍA GENERAL
(Vicerrectorado Académico/Cancillería)
ESCUELA DE CIENCIAS MATEMÁTICAS Y COMPUTACIONALES
CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN
ACTA DE DEFENSA No. UITEY-ITE-2021-00011-AD

A los 28 días del mes de mayo de 2021, a las 15:00 horas, de manera virtual mediante videoconferencia, y ante el Tribunal Calificador, integrado por los docentes:

Presidente Tribunal de Defensa	Dr. ANTON CASTRO , FRANCESC , Ph.D.
Miembro No Tutor	Dr. FERNANDES CAMPOS, HUGO MIGUEL , Ph.D.
Tutor	Dr. COSENZA MICELI, MARIO GIUSEPPE , Ph.D.

El(la) señor(ita) estudiante **FLOR PEÑAFIEL, MARIO FRANCISCO**, con cédula de identidad No. **0105269435**, de la **ESCUELA DE CIENCIAS MATEMÁTICAS Y COMPUTACIONALES**, de la Carrera de **TECNOLOGÍAS DE LA INFORMACIÓN**, aprobada por el Consejo de Educación Superior (CES), mediante Resolución **RPC-SO-43-No.496-2014**, realiza a través de videoconferencia, la sustentación de su trabajo de titulación denominado: **DYNAMICAL NETWORKS WITH GLOBAL INTERACTIONS COMMUNICATIONS, ENCRYPTION AND COLLECTIVE BEHAVIOR**, previa a la obtención del título de **INGENIERO/A EN TECNOLOGÍAS DE LA INFORMACIÓN**.

El citado trabajo de titulación, fue debidamente aprobado por el(los) docente(s):

Tutor	Dr. COSENZA MICELI, MARIO GIUSEPPE , Ph.D.

Y recibió las observaciones de los otros miembros del Tribunal Calificador, las mismas que han sido incorporadas por el(la) estudiante.

Previamente cumplidos los requisitos legales y reglamentarios, el trabajo de titulación fue sustentado por el(la) estudiante y examinado por los miembros del Tribunal Calificador. Escuchada la sustentación del trabajo de titulación a través de videoconferencia, que integró la exposición de el(la) estudiante sobre el contenido de la misma y las preguntas formuladas por los miembros del Tribunal, se califica la sustentación del trabajo de titulación con las siguientes calificaciones:

Tipo	Docente	Calificación
Miembro Tribunal De Defensa	Dr. FERNANDES CAMPOS, HUGO MIGUEL , Ph.D.	10,0
Tutor	Dr. COSENZA MICELI, MARIO GIUSEPPE , Ph.D.	10,0
Presidente Tribunal De Defensa	Dr. ANTON CASTRO , FRANCESC , Ph.D.	10,0

Lo que da un promedio de: **10 (Diez punto Cero)**, sobre 10 (diez), equivalente a: **APROBADO**

Para constancia de lo actuado, firman los miembros del Tribunal Calificador, el/la estudiante y el/la secretario ad-hoc.

FLOR PEÑAFIEL, MARIO FRANCISCO
Estudiante

FRANCESC ANTON CASTRO

Signé électroniquement par FRANCESC ANTON CASTRO
cn=FRANCESC ANTON CASTRO, ou= ENTIDAD DE CERTIFICACION DE INFORMACION, o= SECURITY DATA S.A. 1, c= EC
Date: 2021.06.10 16:30:44 ECT

Dr. ANTON CASTRO , FRANCESC , Ph.D.
Presidente Tribunal de Defensa

Dr. COSENZA MICELI, MARIO GIUSEPPE , Ph.D.
Tutor

Firmado electrónicamente por:
MARIO GIUSEPPE COSENZA MICELI

HUGO MIGUEL FERNANDES CAMPOS

Assinado de forma digital por HUGO MIGUEL FERNANDES CAMPOS
Dados: 2021.06.10 22:32:39 -05'00'

Dr. FERNANDES CAMPOS, HUGO MIGUEL , Ph.D.
Miembro No Tutor

DAYSY MARGARITA MEDINA BRITO

Firmado digitalmente por DAYSY MARGARITA MEDINA BRITO
Fecha: 2021.06.10 15:24:18 -05'00'

MEDINA BRITO, DAYSY MARGARITA
Secretario Ad-hoc

AUTORÍA

Yo, **Mario Francisco Flor Peñafiel**, con cédula de identidad 0105269435,declaro que las ideas, juicios, valoraciones, interpretaciones, consultas bibliográficas, definiciones y conceptualizaciones expuestas en el presente trabajo; así cómo, los procedimientos y herramientas utilizadas en la investigación, son de absoluta responsabilidad de el/la autora (a) del trabajo de integración curricular. Así mismo, me acojo a los reglamentos internos de la Universidad de Investigación de Tecnología Experimental Yachay.

Urcuquí, Junio 2021.

Mario Francisco Flor Peñafiel
CI: 0105269435

AUTORIZACIÓN DE PUBLICACIÓN

Yo, **Mario Francisco Flor Peñafiel,** con cédula de identidad 0105269435, cedo a la Universidad de Investigación de Tecnología Experimental Yachay, los derechos de publicación de la presente obra, sin que deba haber un reconocimiento económico por este concepto. Declaro además que el texto del presente trabajo de titulación no podrá ser cedido a ninguna empresa editorial para su publicación u otros fines, sin contar previamente con la autorización escrita de la Universidad.

Asimismo, autorizo a la Universidad que realice la digitalización y publicación de este trabajo de integración curricular en el repositorio virtual, de conformidad a lo dispuesto en el Art. 144 de la Ley Orgánica de Educación Superior

Urcuquí, Junio 2021.

Mario Francisco Flor Peñafiel
CI: 0105269435

Dedication

I dedicate this work to my parents: Mario and Gabriela who have been my inspiration and unconditional support. This work is only the fruit of your efforts.

Acknowledgements

This work could not be possible without the help, guidance, mentoring, and friendship of my Advisor Mario Cosenzaand Co-Advisor Antonio Acosta whom I admire, respect and appreciate. Thanks to Yachay Tech University for bringing such great minds together.The completion of my higher education could not have been possible without the support and love of my parents during all these years. All the effort, dedication, and work required for the culmination of my studies were possible thanks to the support and help of the people I met at university that ended up becoming very dear friends. Thanks to patrimonial 14 people for being with me in the best and worst moments. Special thanks to Adri for all the support and love she gave me this past year.

Resumen

Proponemos una generalización de un método de cifrado introducido anteriormente basado en redes de mapas acopladas globalmente. Nuestro algoritmo emplea una red de mapas caóticos para cifrar directamente los bytes contenidos en cualquier archivo digital, en lugar de solo archivos de texto sin formato como método original. Nuestro método se puede aplicar para codificar y decodificar texto plano, imágenes en color, audio, video, archivos comprimidos, etc. Además, nuestro método abarca el almacenamiento y recuperación de la información encriptada. El método es conceptualmente simple y computacionalmente eficiente, y se adapta a altos niveles de seguridad. Mostramos varios ejemplos y aplicaciones del método. Un producto útil de esta tesis es la implementación del algoritmo de cifrado y almacenamiento generalizado en una interfaz gráfica o aplicación para el sistema operativo Linux-Ubuntu. Esta aplicación permite al usuario seleccionar archivos de diferentes tipos para codificarlos y decodificarlos.

Palabras Clave: Criptografía, Sistemas Dinámicos, Caos, Redes de Mapas Acoplados.

Abstract

We propose a generalization of previously introduced encryption method based on globally coupled map networks. Our algorithm employs a network of chaotic maps to directly encrypt bytes contained in any digital file, instead of just plain text files as the original method. Our method can be applied to encode and decode plain text, color images, audio, video, compressed files, etc. Additionally, our method encompasses the storage and recovery of the encrypted information. The method is conceptually simple and computationally efficient, and adaptable to high levels of security. We show several examples and applications of the method. A useful product of this thesis is the implementation of the generalized encryption and storage algorithm in a graphical interface or app for the Linux-Ubuntu operating system. This app allows the user to select files of different types to be coded and decoded.

Keywords: Cryptography, Dynamical Systems, Chaos, Coupled Map Networks.

Contents

List of Figures

Chapter 1

Introduction

1.1 Chaotic dynamical systems, Cryptography, and this thesis

After the pioneering work of Edward Lorenz[1], Chaos Theory was established by the end of the XX century with developments from many different research areas such as Mathematics[2–5], Physics[6–8], Biology[9], Chemistry[10,11], and Engineering[12]. Illustrative and documented accounts of those advances and their discoverers are presented in Refs.[13–15]. The phenomenon of chaos consists of the extreme sensitivity of the evolution of a deterministic nonlinear dynamical system under small changes in its initial conditions[1,16,17]. Very close initial conditions will cause completely different results as time elapses. The sensitivity to initial conditions makes the trajectories generated by deterministic equations eventually unpredictable. This property of chaotic systems is known as the “Butterfly Effect”. Chaos has had a profound impact in all sciences and human culture.

Since early works, researchers have pointed out that there exists a close relationship between chaos and cryptography[18–24]. The properties of chaotic dynamics, such as extreme sensitivity to initial conditions, ergodicity, and mixing, relate with the “confusion" and "diffusion" that are characteristic attributes of several methods in cryptography. It was natural to think that Chaos Theory can be used as a tool in the development of new techniques for ciphers. The approach of using dynamical systems as cryptosystems was anticipated by Shanon in “Communication Theory of Secrecy Systems” in 1949[25]. In recent times, cellular automata are considered among the first applications of dynamical systems as ciphers[26,27]. On the other hand, it was shown that that some conventional stream ciphers can exhibit chaotic behavior[23]. Then, from an algorithmic point of view, any good cipher could be considered as a pseudo-chaotic or chaotic system[28]. Currently, it is accepted that the investigation about chaotic cryptosystems has opened roads to the design of new ciphers and has enriched conventional cryptography[29,30].

There are two general ways for designing chaotic-based ciphers: stream ciphers and block ciphers. Stream ciphers[31–40] are used to mask text with no format; it consists of using chaotic systems to generate pseudo-random strings of symbols. Block ciphers[41–44] use plain text and secret keys as initial conditions, iterating and/or counter-iterating chaotic systems multiple times to obtain a ciphertext. Most of the first models were stream ciphers that employed only one chaotic dynamical element, either for masking the message to be sent or for transmitting a controlled signal.

In particular, iterated functions or discrete-time maps are naturally suited for applications as string generators and stream ciphers[21,31,38,45]. However, it has been shown that cryptosystems based on one-dimensional chaotic maps, such as the logistic map, may result in poor security when used as communication systems[46–48].

Garcia et al.[37] proposed an alternative approach that increases the security and the scope of applications of chaotic cryptosystems, where a multidimensional dynamic system, specifically a network of coupled chaotic maps, is considered as a generator of strings of symbols and as an encryption and communication system. This approach constitutes the theoretical framework for the present thesis.

Coupled map lattices or coupled map networks (CMN) are spatiotemporal dynamical systems where space and time are discrete, but the dynamical states are continuous. They consist of a set of maps or iterative functions considered as nodes interacting on a lattice or on a general network[49–53]. Coupled map networks have provided useful models for the study of diverse spatiotemporal processes in spatially extended systems, with the advantage of being computationally efficient and, in many cases, mathematically tractable[54]. In this context, globally coupled map networks, where each element interacts with each other in the system, constitute paradigmatic models for the current research of complex systems[55].

In particular, the encryption capability of coupled map networks can be seen as an emerging functionality of an autonomous dynamical system of interacting elements, without any external influences. This is a distinctive property of complex systems, where nontrivial collective behaviors arise from the interactions among the constitutive elements of the system[56].

In this thesis, we propose a generalization of the encryption method based on globally coupled map networks introduced by García et al.[37]. Our algorithm employs a network of coupled chaotic maps to directly encrypt bytes contained in any digital file, instead of just plain text files as the original method of García et al. Thus, our method can be applied to encode and decode plain text, color images, audio, video, compressed files, etc. Additionally, our proposed method encompasses the storage and recovery of the encrypted information. We show that the method is conceptually simple, computationally efficient, and, in terms of security, scalable and comparable with current standard methods.

Chapter 2 of this thesis presents a brief review of the article by García et al.[37] that lies the theoretical framework for our model. Our generalized encryption method based on a coupled chaotic map network is described in Chapter 3. The processes of coding, storing, and decoding are explained and illustrated with examples. The security of the method is also discussed in Chapter 3.

We present our main results and show several applications of the generalized encryption method in Chapter 4. A useful product of this thesis is the implementation of the generalized encryption and storage algorithm in an app for the Linux operating system. This app facilitates the user to select files of different types to be coded and decoded. It also offers the option of encrypting plain text that is typed directly in a displayed window. The cipher text can be decoded to yield the original plain text.

The Conclusions and future extensions of the present thesis are contained in Chapter 5.

1.2 Problem statement

Modern communication media through the Internet or the distribution of any digital content involves the exchange of enormous amounts of information. Without security, these media are vulnerable to third-party members. In recent years, not only companies and governments have been affected by deficiencies in digital security, but also common users. The need for the development of new and alternative forms of encrypting information when sharing data is an important problem in our digital world. This thesis presents a general method to encrypt and store any type of digital file taking into account a compromise between encryption speed and security.

1.3 General and Specific Objectives

General Objective

Our main objective is to propose and investigate a conceptually simple, computationally efficient, and sufficiently secure method based on the properties of chaotic dynamical systems for encrypting digital files.

Specific Objectives

1. To develop an algorithm that generalizes the encryption scheme proposed in Ref.[37].
2. To extend the scope of the scheme to include digital files of any type.
3. To formulate a process for storing the encrypted information in optimal form, allowing its distribution through any digital medium with sufficient security.
4. To create an app with a graphical interface that facilitates the application of the generalized encryption method to users.

Chapter 2

Theoretical framework: Coupled map networks as cryptosystems

In this chapter, we review the article that provides the theoretical basis for our proposed encryption method. Here, we adapt the main ideas and the notation for our purpose.

In the article *Coupled Map Networks as Communication Schemes*[37], García et al. investigate networks of coupled chaotic maps as generators of strings of symbols, and propose to use them as an encrypting system for ASCII symbols.

Consider a coupled map network (CMN) defined as

$$x_{t+1}^i = f(x_t^i) + \sum_{j=1}^{N} \epsilon_{ij} x_t^j, \tag{2.1}$$

where x_t^i gives the state variables of the local element i ($i = 1, \ldots, N$) at discrete time t; $f(x_t^i)$ is a real valued function that describes the dynamics of element i; ϵ_{ij} are the coupling strengths among elements in the system; and N is the size of the network.

The CMN system Eqs. (2.1) can be expressed in vector form as

$$\mathbf{x}_{t+1} = \mathbf{f}(\mathbf{x}_t) + \mathbf{E}\,\mathbf{x}_t, \tag{2.2}$$

where $\mathbf{x}_t = (x_t^1, x_t^2, \ldots, x_t^N)$ is the state vector of the system at discrete time t, $\mathbf{f}(\mathbf{x}_t) = ((f(x_t^1), f(x_t^2), \ldots, f(x_t^N))$, and the ϵ_{ij} are components of the $N \times N$ coupling matrix $\mathbf{E}$. The coupling strengths ϵ_{ij} can be different from each other; i. e., the coupling can be heterogeneous. If $\epsilon_{ij} \neq 0$, $\forall i, j$, the CMN system is called *globally* coupled.

Since the maps are chaotic, the time evolution of the system Eqs. (2.1) can exhibit a great variability, and therefore, can be used as a generator of multiple strings that can be interpreted as a sequence of different symbols. The authors of Ref.[37] were thinking in ASCII symbols, and thus, they fixed the number of maps in the network at $N = 7$. Then, to each state of the network $\mathbf{x}_t = (x_t^1, x_t^2, x_t^3, x_t^4, x_t^5, x_t^6, x_t^7)$, one can assign a binary sequence $b_t = (b_t^1, b_t^2, b_t^3, b_t^4, b_t^5, b_t^6, b_t^7)$ according to the following rule

$$b_t^i = 0, \quad \text{if } x_t^i < x^*, \tag{2.3}$$

$$b_t^i = 1, \quad \text{if } x_t^i \geq x^*, \tag{2.4}$$

where x^* is some threshold value. With a prefixed correspondence rule, each of the $2^7 = 128$ possible seven-digit binary states $(b_t^1, b_t^2, b_t^3, b_t^4, b_t^5, b_t^6, b_t^7)$ can be uniquely assigned an ASCII symbol that we denote by z^k, belonging the set $\{z^1, z^2, \ldots, z^{128}\}$.

Assume that, starting from an initial condition $\mathbf{x}_0$, the evolution of CMN Eq. (2.2) eventually generates all ASCII symbols in the set $\{z_1, z_2, \ldots, z_{128}\}$, provided that there are no symbols forbidden by the dynamics of the CMN. Denote by

$$\alpha_T(\mathbf{x}_0) = (z_1^k, z_2^k, \ldots, z_t^k, \ldots, z_T^k) \tag{2.5}$$

the chain of symbols generated in a time T. Similarly, denote by $|\alpha| = T$ the length of the chain, or the number of iterations performed on the CMN system up to time T.

Consider an ordered sequence of ASCII symbols $P = (p^1, p^2, \ldots, p^n)$. For a sufficiently large number of iterations T, a chain $\alpha_T(\mathbf{x}_0)$ can be expressed as a succession of substrings $\beta^l p^l$, $l = 1, \ldots, n$, in the form

$$\alpha_T(\mathbf{x}_0) = (\beta^1 p^1, \beta^2 p^2, \ldots, \beta^{l-1} p^{l-1}, \beta^l p^l, \ldots, \beta^n p^n), \tag{2.6}$$

where $\beta^1 p^1$ is the substring beginning at z_1^k and ending at the first occurrence of symbol p^1, the substring $\beta^2 p^2$ begins after p^1 and ends at the first occurrence of symbol p^2, and so on. Because the CMN Eq. (2.2) is deterministic, the substring $\beta^l p^l$ depends only on the state of the system $\mathbf{x}_{l-1}$ at time $l-1$ and on the symbol p^l.

As an example, consider the sequence of ASCII symbols for the word "Rival" as $P = (R, i, v, a, l)$. The string

$$\alpha = (d, 4, \$, R, m, e, >, i, \&, H, +, t, 5, v, ?, u, K, g, a, i, a, 6, l) \tag{2.7}$$

is segmented in 5 substrings by $P = (R, i, v, a, l)$ as

$$\alpha = (\underbrace{d, 4, \$}_{\beta^1}, \overbrace{R}^{p^1}, \underbrace{m, e, >}_{\beta^2}, \overbrace{i}^{p^2}, \underbrace{\&, H, +, t, 5}_{\beta^3}, \overbrace{v}^{p^3}, \underbrace{?, u, K, g}_{\beta^4}, \overbrace{a}^{p^4}, \underbrace{i, a, 6}_{\beta^5}, \overbrace{l}^{p^5}). \tag{2.8}$$

The segmentation of the chain α by a finite sequence $P = (p^1, p^2, \ldots, p^n)$ can be represented by an n-dimensional vector $C(P)$ whose components are the natural numbers giving the lengths $|\beta^l p^l|$, $l = 1, \ldots, n$. That is,

$$C(P) = (|\beta^1 p^1|, |\beta^2 p^2|, \ldots, |\beta^l p^l|, \ldots, |\beta^n p^n|). \tag{2.9}$$

Since the CMN can be iterated indefinitely, Eq. (2.9) just expresses the segmentation $C(P)$ for the first $T = \sum_{l=1}^{n} |\beta^l p^l|$ symbols of the string α. For the example considered, the word $P = (R, i, v, a, l)$ will be represented by the vector $C(P) = (4, 4, 6, 5, 4)$, as illustrated in Fig. 2.1.

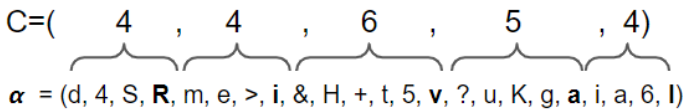

Figure 2.1: Word $P = (R, i, v, a, l)$ contained in string α is represented by the vector of natural numbers C.

García et al.[37] propose to use Eq. (2.9) as an encryption procedure for the plain ASCII text $P = (p^1, p^2, \ldots, p^n)$ in terms of the vector $C(P)$. If the local function $f(x)$ is public, the secret key may consist of the couplings e_{ij} and the initial condition $\mathbf{x}_0$. Once the coupling matrix $\mathbf{E}$ is specified, the autonomous chaotic evolution of the CMN will generate a string α that depends only on $\mathbf{x}_0$.

In other words, if the matrix $\mathbf{E}$ and $\mathbf{x}_0$ are used as a secret key, the string $\alpha_T(\mathbf{x}_0)$ will always be the same for a fixed key and a sufficiently large T. Then, given $P = (p^1, p^2, \ldots, p^n)$, the CMN system Eq. (2.2) uniquely generates the string $\alpha(\mathbf{x}_0)$ that can be represented and ciphered by $C(P)$, according to Eq. (2.9). Conversely, knowing the cipher $C(P)$, the public key $f(x)$, and the private keys $\mathbf{E}$ and $\mathbf{x}_0$, allows us to obtain the original plain text $P = (p^1, p^2, \ldots, p^n)$, thanks to the deterministic evolution of the CMN, Eq. (2.2).

Note that a number $n = |P|$ symbols of string α can be known if the plain text $P = (p^1, p^2, \ldots, p^n)$ and its corresponding cipher text $C(P)$ are known. Unknown elements between the n known symbols $(p^1, p^2, \ldots, p^n)$ can be inferred by using new messages encrypted with the same key, even when the new plain texts are unavailable. In the example, the word "Rival" is encrypted as $C(R, i, v, a, l) = (4, 4, 6, 5, 4)$; therefore, after 19 iterations and after 23 iterations the CMN generates the symbols "a" and "l", respectively. If another word has the encryption $C(Q) = (4, 4, 4, 4, 3, 4)$, it can be inferred that $Q = (R, i, t, u, a, l)$, and two additional symbols of the corresponding string α can be inferred.

In principle, any chaotic function $f(x_t)$ can be used as a local map in the CMN system, Eq. (2.1). As an application, García et al. considered the logarithmic map[57] $f(x_t) = b + \ln|x_t|$. This map is unbounded and chaotic, with no periodic windows in the parameter interval $b \in (-1, 1)$, as shown in Fig. 2.2.

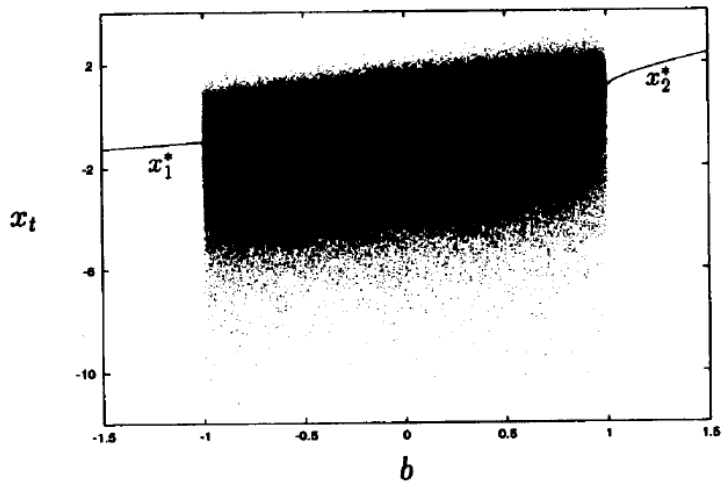

Figure 2.2: Bifurcation diagram x_t as a function of the parameter b for the logarithmic map $x_{t+1} = b + \ln|x_t|$. For each value of b, 500 iterates x_t are plotted, after discarding 500 transients. Complete chaos occurs in the range $b \in (-1, 1)$. The labels x_1^* and x_2^* indicate fixed points orbits for $b < 1$ and $b > 1$, respectively. Reprinted from Ref.[57]

The unbounded character of the local logarithmic functions has the advantage that there are no restrictions on the range of parameter values of the CMN system that can be explored. The threshold value chosen is $x^* = 0$. García et al. verified that, for this threshold, local parameter values about $b \approx 0.5$, and couplings randomly selected in the interval $|\epsilon_{ij}| < 0.1$ $\forall i, j$, all the 128 ASCII symbols in the set $\{z^1, z^2, \ldots, z^{128}\}$ are generated by the seven-dimensional CMN with approximately the same probability of $1/128$.

The useful property of chaotic CMNs as encrypting schemes is their sensitivity to initial conditions and/or couplings. For a fixed value of the parameter b and $N = 7$, the maximum encrypting key consists of 7×7 coupling strengths and seven initial conditions $x_i(0)$. A change of $\delta = 10^{-10}$ in one of the couplings will produce more than $10^{\delta^{-1} \times 7 \times (7+1)} \sim 10^{560}$ possible keys. In general, such a large number of possible keys is unnecessary, and in practice the key can be reduced by using a set of random number seeds to generate the 7×7 coupling strengths and the seven initial conditions $x_i(0)$.

At the end of the article, P. García et al.[37] also suggest that this method may be used to encrypt black and white images. By setting $N = 8$, the CMN Eqs. (2.1) can be employed to generate strings with elements in a scale of $2^8 = 256$ gray tones and therefore to encrypt images pixel by pixel. Figure 2.3 shows a black-and-white image and its decoding using this method with a sightly wrong key.

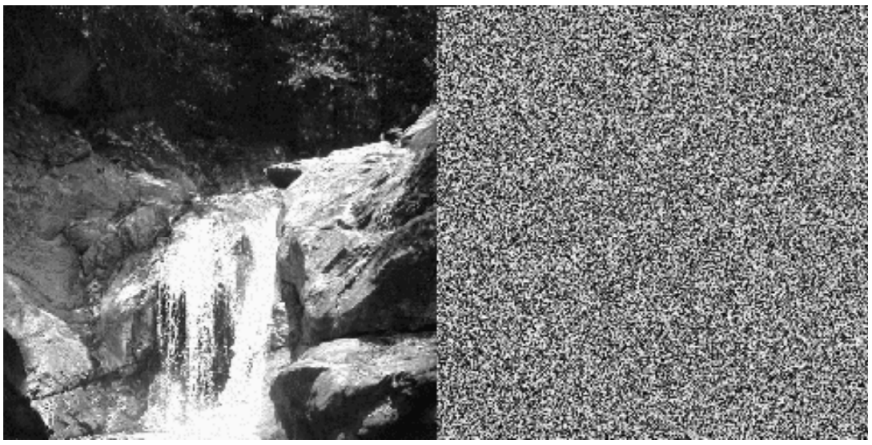

Figure 2.3: The bitmap image on the left has been encrypted assuming $b = 0.5$, $x_i(0) = 1.0 + 0.1i$, and $e_{ij} = 0.01(i - j/2)$ $(i, j = 1, 2, \ldots, 8)$. On the right, the corresponding decoded image shown when a slightly wrong key is used by adding 10^{-10} to ϵ_{35}. Reprinted from Ref.[37]

Chapter 3

Generalized CMN encryption method

3.1 A general cryptosystem based on chaotic CMN

In this Chapter, we propose a generalization of the encryption method based on coupled map networks (CMN) that was presented in Chapter 2. We are motivated by two observations (i) the system size N of the CMN can be reduced in order to increase the encrypting speed and to decrease the number of possible keys; and (ii) consequently, it is not necessary to generate 128 ASCII symbols or 256 gray tones for black-and-white images.

Instead, of ASCII symbols, we propose a general method to directly encrypt bytes contained in a digital file of any type. Thus, we define a two-dimensional ($N = 2$) coupled map network (CMN) as

$$\mathbf{x}_{t+1} = \mathbf{f}(\mathbf{x}_t) + \mathbf{E}\,\mathbf{x}_t, \tag{3.1}$$

where

$$\mathbf{x}_t = \begin{pmatrix} x_t^1 \\ x_t^2 \end{pmatrix} \quad \text{and} \quad \mathbf{f}(\mathbf{x}_t) = \begin{pmatrix} f(x_t^1) \\ f(x_t^2), \end{pmatrix} \tag{3.2}$$

describe the state variables and the local chaotic dynamics of the system at discrete time t, respectively; both defined in some real interval. The real matrix

$$\mathbf{E} = \begin{pmatrix} e_{11} & e_{12} \\ e_{21} & e_{22} \end{pmatrix} \tag{3.3}$$

expresses the coupling strengths between the variables of the system.

We assign binary values to the states x_t^1 and x_t^2 of the system at a time t according to the rule

$$\text{if } x_t^i \geq x^*, \quad b_t^i = 1 \tag{3.4}$$

$$\text{if } x_t^i < x^*, \quad b_t^i = 0, \quad i = 1, 2, \tag{3.5}$$

where x^* is some appropriately chosen threshold value. Then, the state of the system at time t can be expressed as a binary pair in the form $\mathbf{x}_t = (x_t^1, x_t^2) \to (b_t^1, b_t^2)$.

On the other hand, a binary pair can be interpreted as a number in a quaternary numeral system. Therefore, a quaternary representation can be assigned to the state $\mathbf{x}_t$ as follows

$$(x_t^1, x_t^2) \to (b_t^1, b_t^2) \in \{(0,0), (0,1), (1,0), (1,1)\} = \{z_t^0, z_t^1, z_t^2, z_t^3\} = \{0, 1, 2, 3\}. \tag{3.6}$$

Given an initial condition $\mathbf{x}_0 = (x_0^1, x_0^2)$, the CMN system Eq. 3.1 generates the following sequence of iterates or orbit in a time T,

$$(x_0^1, x_0^2) \to (x_1^1, x_1^2), (x_2^1, x_2^2), (x_3^1, x_3^2), \ldots, (x_T^1, x_T^2). \tag{3.7}$$

The sequence Eq. 3.7 can be expressed as a chain of quaternary symbols

$$\alpha(\mathbf{x}_0) \equiv (z_1^k, z_2^k, \ldots, z_T^k), \tag{3.8}$$

where $(x_t^1, x_t^2) \to z_t^k$ according to the assignment Eq. 3.6. We denote the length of the sequence α by $|\alpha| = T$.

Consider a byte $B = (b^1, b^2, b^3, b^4, b^5, b^6, b^7, b^8)$, where $b^i \in \{0, 1\}$, $i = 1, 2, \ldots, 8$. We can write byte B as an ordered sequence of four quaternary symbols: $B = (w^1, w^2, w^3, w^4)$, where $w^1 = (b^1, b^2)$, $w^2 = (b^3, b^4)$, $w^3 = (b^5, b^6)$, and $w^4 = (b^7, b^8)$; such that $w^i \in \{z^1, z^2, z^3, z^4\}$.

Then, a sufficiently long orbit of the CMN Eq. 3.1 that starts at $\mathbf{x}_0$ can be represented by a chain of quaternary symbols $\alpha(\mathbf{x}_0)$ that will eventually contain byte B:

$$\alpha_T(\mathbf{x}_0) = (z_1^k, z_2^k, \ldots, z_{t_1}^k = w^1, \ldots, z_{t_2}^k = w^2, \ldots, z_{t_3}^k = w^3, \ldots, z_{t_4}^k = w^4). \tag{3.9}$$

Define the set of integers $c_j \equiv t_j - t_{j-1}$, $j = 1, 2, 3, 4$. That is, $c_j < 256$ is the number of iterations between the appearances of the quaternary symbols w^{j-1} and w^j contained in byte B. If $w^j = w^{j-1}$, the CMN does not need to iterate from the symbol w^{j-1} to find w^j, and we set $c_j = 0$. Then, the ordered sequence of integers

$$(c_1, c_2, c_3, c_4) = (t_1, t_2 - t_1, t_3 - t_2, t_4 - t_2), \tag{3.10}$$

is employed to encode byte B. Since the CMN system is deterministic, given an initial condition $\mathbf{x}_0$, there is a unique sequence (c_1, c_2, c_3, c_4) generated by the CMN system Eq. 3.1 that encodes byte B.

Furthermore, the coding sequence (c_1, c_2, c_3, c_4) can be written into a cipher file file.ds as an ordered sequence of four bytes $(\tilde{B}_1, \tilde{B}_2, \tilde{B}_3, \tilde{B}_4)$, where the integer c_j is assigned its corresponding binary representation as a byte $\tilde{B}_j$, $j = 1, 2, 3, 4$. Thus, if the original byte B belongs to a file file.ext, it is converted into four bytes $(\tilde{B}_1, \tilde{B}_2, \tilde{B}_3, \tilde{B}_4)$ in the cipher file with extension "*.ds", as file.ds. Figure 3.1 illustrates our encryption method.

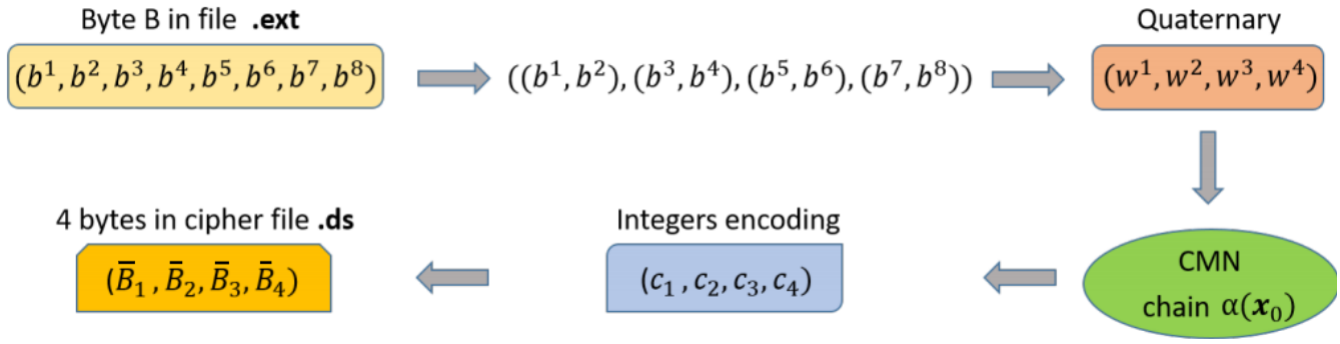

Figure 3.1: Schematic representation of the proposed encryption method.

By applying this procedure byte by byte, a file file.ext consisting of N bytes B_i, $i = 1, \ldots, N$, can be encoded into a cipher file file.ds that will consists of $4N$ bytes $(\tilde{B}_1, \tilde{B}_2, \tilde{B}_3, \tilde{B}_4)_i$. Then, an encryption system for any file file.ext of the form $(B_1, B_2, \ldots, B_N)$ can be defined, where the encrypted code is $[(c_1, c_2, c_3, c_4)_1, (c_1, c_2, c_3, c_4)_2, \ldots, (c_1, c_2, c_3, c_4)_N]$ and the encrypted file file.ds has the form $\left[(\tilde{B}_1, \tilde{B}_2, \tilde{B}_3, \tilde{B}_4)_1, (\tilde{B}_1, \tilde{B}_2, \tilde{B}_3, \tilde{B}_4)_2, \ldots, (\tilde{B}_1, \tilde{B}_2, \tilde{B}_3, \tilde{B}_4)_N\right]$. Similarly to in Ref.[37], the function $\mathbf{f}(\mathbf{x}_t)$ can be used as a public key, while the matrix $\mathbf{E}$ and the initial condition $\mathbf{x}_0$ can be taken as the private keys. Thus, since the chaotic system Eq. 3.1 is deterministic, knowing $\mathbf{x}_0$ and $\mathbf{E}$ allows to obtain the original file file.ext from the encrypted code or from the cipher file file.ds.

The processes of coding and decoding a file is illustrated in Fig. 3.2. The algorithm receives an input file file.ext and returns an encrypted file file.ds. The cipher file file.ds contains all the information necessary for recovering the original input file, including the extension or type of file. For the decoding process, the extension "*.ext" of the input file is found in the first bytes of the file file.ds. Once the extension is known, a new file is created with the corresponding name and extension file.ext. The algorithm will read byte by byte into file.ds in order to know how many times the CMN system should be iterated to get the information of the original input file.

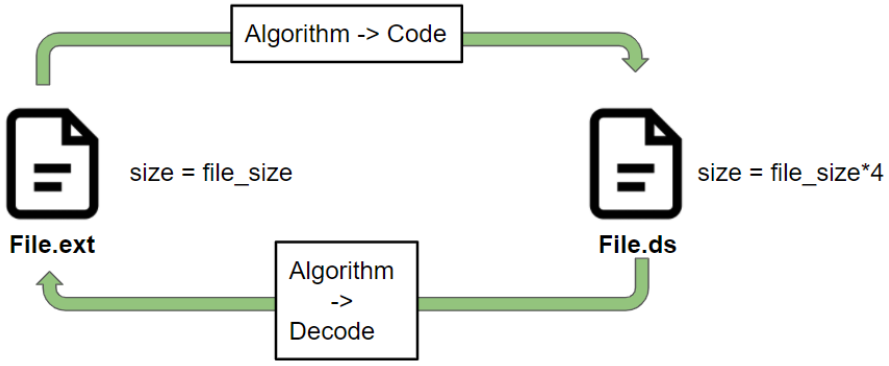

Figure 3.2: Scheme of the process for general encryption and decryption of a file with our proposed method.

3.2 Coding and Storing

Any digital file consists of bytes, and each byte is a sequence of 8 bits. There are $2^8 = 256$ distinct bytes. This can be compared with the 256 gray tones proposed in Ref.[37] for encrypting black-and-white images. Therefore, we can extend this logic to encrypt any type of file byte by byte instead of pixel by pixel as for black-and-white images. Our method proposes to take each byte in a file, ordered from the first to the last, independently of its extension or type. Thus, we consider any file as an ordered sequence of bytes, in analogy to the ordered sequence of ASCII symbols for words or pixels for images considered in Ref.[37].

A byte can be divided into four groups of two bits each, each of these can be associated with quaternary symbols, as explained above. The number of iterations of the CMN system Eq. 3.1 required to encrypt each quaternary symbol does not exceed 255. Therefore, the number of iterations necessary to encrypt each of these groups or quaternary symbols can be stored in one byte. Then, each byte of information can be encoded and stored in four bytes in a file "*.ds". As a consequence, the storing file "*.ds" will have a size four times greater than the original file.

In Figure 3.3, we show an example of the coding and storing process of our method, where 4 bytes from the original file are transformed into 16 bytes in the storing file. We assume that the coding and decoding devices have the same endianness.

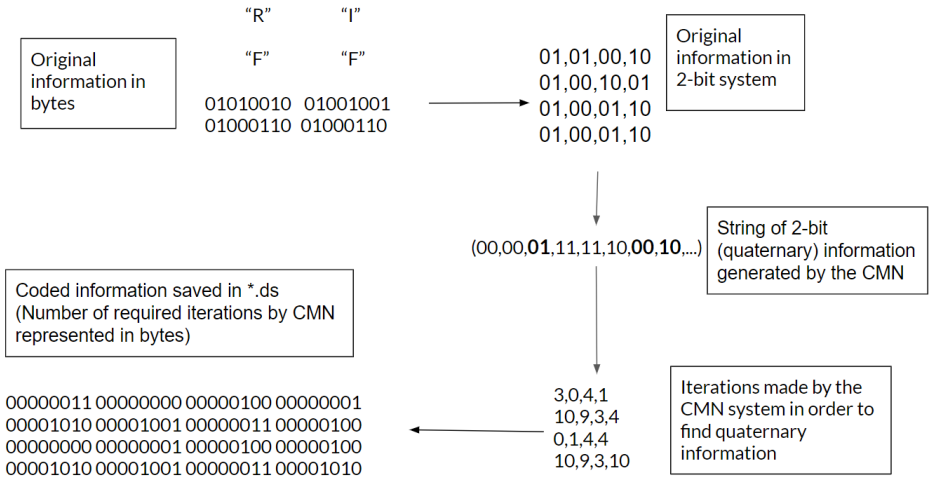

Figure 3.3: Process of coding and storing the 4 bytes R,I,F,F, into a file "*.ds".

Figure 3.4 shows the resulting coding and storing of bytes from a file file.wav in the cipher file file.ds.

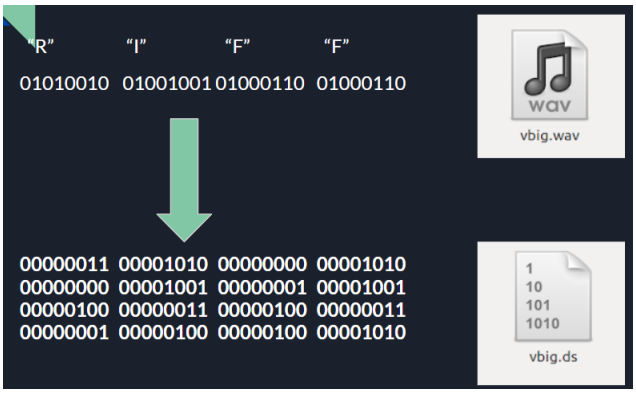

Figure 3.4: Coding and storing of bytes "R,I,F,F" from the file vbig.wav into the file vbig.ds.

3.2.1 How the file extension is saved

An input file with extension type "*.ext" is encoded and stored into a file with the same name but with the extension "*.ds". In general, for encoding a file extension, the encrypting program proceeds as follows:

1. The program reads the extension type and the number of bytes or length of the extension, in the form: (3,jpg); (3,exe); (1,c); (3,wav); (4,jpeg); etc.
2. In the file ".ds", the length of the extension is written as the first byte.
3. The next bytes in the file "*.ds" correspond to the ASCII symbols for each letter of the file extension.

For example, if the input file type is "*.jpg", the information of the file extension (3,j,p,g) is written in the file "*.ds" in bytes as: 00000011, 01101010, 01110000, 01100111. Figure 3.5 shows how different file types are stored in files "*.ds".

Figure 3.5: Examples of input files and stored encrypted files "*.ds".

3.2.2 What is saved in a file *.ds

In summary, the information contained inside a cipher file file.ds consists of the following three consecutive parts:

1. First byte comprising the length of the extension of the input file file.ext.
2. ASCII symbols that express the file extension of the input file.
3. Number of iterations that should be taken on the CMN system Eq. 3.1 to recover the information of the input file.

3.3 Decoding

To decode a file “*.ds", the program reads the first byte to know the number of next bytes required for recovering the file extension. Then, each of the remaining bytes is read and converted into an integer number. This integer corresponds to the number of iterations that should be performed on the CMN system Eq. 3.1, starting from the initial condition, to generate the two-bits group or quaternary symbol that constitutes a fourth of the corresponding decoded byte. Each 4 bytes of the cipher file “*.ds" allows the recovery of 4 two-bits groups. These 4 groups are concatenated into one byte that corresponds to the original file. This process is repeated until all the bytes in the file “*.ds" are converted. Therefore, the original file will have a fourth of the size of the cipher file “*.ds".

Figure 3.6 shows the process of decoding information from a “*.ds" file. In the example exhibited, 16 bytes from the “*.ds" file are transformed into 4 bytes in the original file.

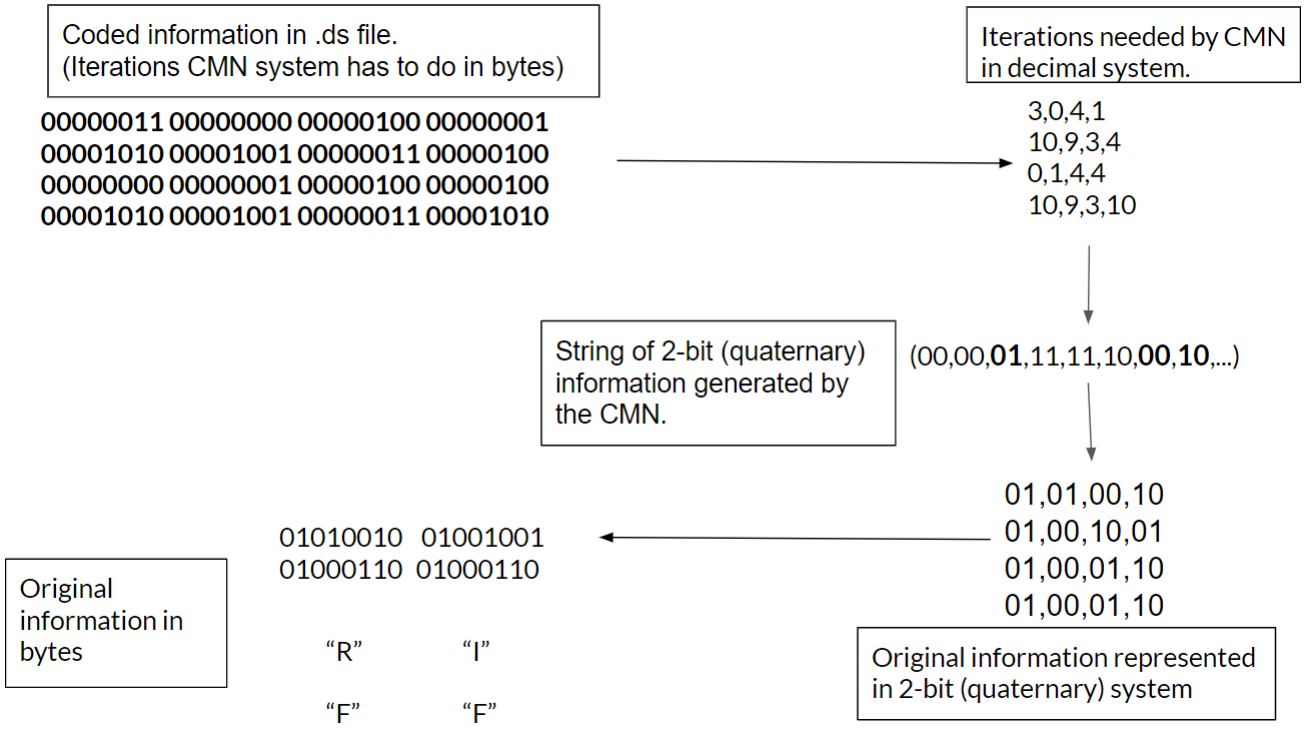

Figure 3.6: Decoding process for a “*.ds" file. In this example, 16 bytes from the “*.ds" file are converted into 4 bytes in the original file.

Figure 3.7 shows the resulting decoded information, where 16 bytes from the cipher file file.ds are converted into the 4 bytes "R,I,F,F" in the original file file.wav.

Figure 3.7: Decoding of 16 bytes from the file vbig.ds into the 4 bytes "R,I,F,F" in the file vbig.wav.

3.4 Security

Let us assume that the only secret key is the initial condition $\mathbf{x}_0 = (x_0^1, x_0^2)$ for the CMN system Eq. 3.1. Then, the key comprises two double-precision numbers. Each of these numbers consists of 8 bytes or 64 bits, which implies that there are $(2^{64})^2 = 2^{128}$ possible keys. We assume that the coding and decoding devices has the same endianness and architecture. Each key will have 128 bits. The property of chaos means that the time evolution of the CMN system is extremely sensitive to small changes in the initial conditions. Then, even a change in one bit of the key represents a change of the initial condition that will lead to another trajectory and therefore to a different encoding or decoding. Thus, the security of our generalized CMN cryptosystem is quite robust since the 128-bit key is comparable to other methods, such as Data Encryption Standard (DES) that uses a 56-bit key, and Advanced Encryption Standard (AES) that employs keys having between 128 and 256 bits.

In addition, we can take as a secret key the double-precision values of the 2×2 components ϵ_{ij} of the coupling matrix Eq. 3.3. By combining initial conditions and coupling components, the security of our cryptosystem can be further enhanced.

Chapter 4

Results and Applications

4.1 Local dynamics

In order to implement the generalized encryption method presented in Chapter 3, we shall use a two-dimensional coupled map network (CMN) of the form

$$x_{t+1}^i = f(x_t^i) + \sum_{j=1}^{2} \epsilon_{ij} x_t^j, \qquad i, j = 1, 2. \tag{4.1}$$

For simplicity, we shall consider that the coupling strengths are equal, i.e.; $\epsilon_{ij} = \epsilon$. Then, Eqs. 4.1 become

$$\begin{aligned} x_{t+1}^1 &= f(x_t^1) + \epsilon\,(x_t^1 + x_t^2), \\ x_{t+1}^2 &= f(x_t^2) + \epsilon\,(x_t^1 + x_t^2). \end{aligned} \tag{4.2}$$

As sources of local chaos, we choose the following family of maps,

$$x_{t+1} = f(x_t) = \mu - |x_t|^z. \tag{4.3}$$

These maps are singular for $|z| < 1$ and have been shown to possess robust chaos, with no periodic windows, on a single interval of the parameter μ whose boundaries depend on z[58]. Figure 4.1 shows the bifurcation diagram of the singular map Eq. 4.3 as a function of the parameter μ, corresponding to a fixed exponent $z = -0.5$

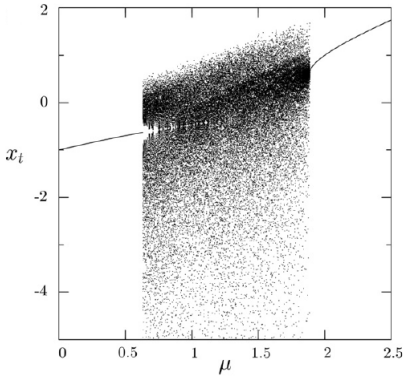

Figure 4.1: Bifurcation diagram of x_t as a function of the parameter μ for the singular map $x_{t+1} = \mu + |x_t|^{-0.5}$. For each value of μ, 500 iterates x_t are plotted, after discarding 500 transients. Robust chaos occurs on the interval $\mu \in (0.63, 1.89)$, while stable fixed points appear outside this interval. Reprinted from Ref.[58]

Figure 4.2 shows the dynamical behavior of the local singular5 map Eq. 4.3 on the space of parameters (μ, z). The boundaries shown separate the region on this plane where robust chaos occurs from the region where the map reaches a stable fixed or stationary point[58].

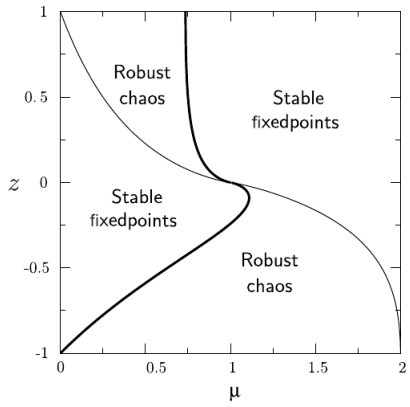

Figure 4.2: Dynamical behavior the singular map Eq. 4.3 on the space of parameters (μ, z). Regions where robust chaos and stable fixed points appear are indicated. Reprinted from Ref.[58]

For applications of the encryption method, we choose values of μ and z of the local map Eq. 4.3 in the region corresponding to robust chaos in Fig. 4.2. Robustness provides an advantage in applications that require reliable operation under chaos, such as secure communications, since the chaotic behavior cannot be modified by arbitrarily small perturbations of the parameters of the system. As local map parameters for the CMN system Eqs.4.2, we fix $z = -0.25$ and $\mu = 1.4$.

4.2 Threshold and coupling parameter values for the CMN cryptosystem

García et al.[37] chose a value $x^* = 0.0$ for their CMN system with size $N = 7$ that used a local logarithmic map. In our case, the threshold value x^* defines how binary pairs or quaternary symbols are assigned to the state (x_t^1, x_t^2). The threshold value should guarantee that the four binary pairs appear with approximately the same probability in the time evolution of the CMN system Eqs.4.2.

Figure 4.3 compares the frequency distribution of the quaternary symbols or binary pairs for two different threshold values. For the fixed local parameters $z = -0.25$ and $\mu = 1.4$, and a range of the coupling parameter ϵ in the CMN system Eqs.4.2, we have found that the choice $x^* = -0.03$ yields an approximate equitable distribution for the quaternary symbols for long times.

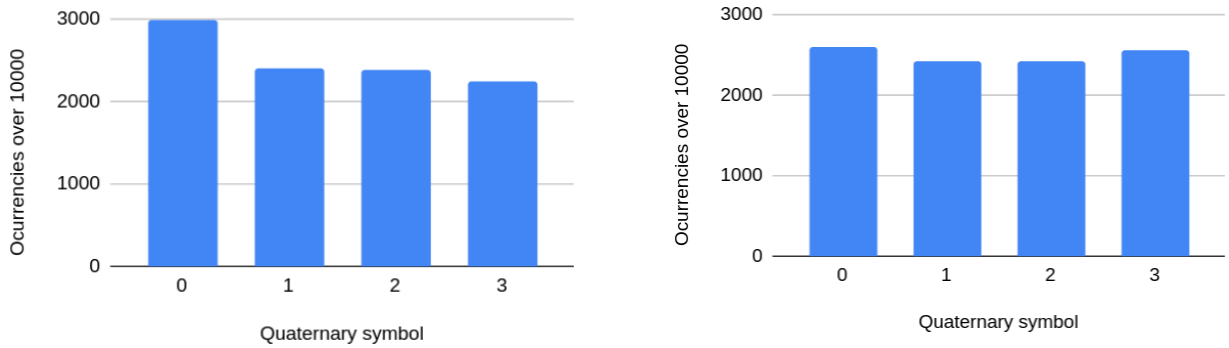

Figure 4.3: Frequency distribution of quaternary symbols for different threshold values x^*. Left: $x^* = 0.0$. Right: $x^* = -0.03$. Fixed parameters are $z = -0.25$, $\mu = 1.4$, $\epsilon =?$.

The performance of the CMN system may also depend on the value of the coupling parameter. To find optimal values of the coupling parameter, we have calculated the time employed by our algorithm with the CMN system Eqs.4.2 for encrypting files of different sizes. Figure 4.4 shows these times for a range of values of ϵ. The results indicate that the encrypting time is shorter on the range $\epsilon \in [-0.2, 0.1]$.

	Time (seconds)	
ϵ	1101219 bits	22601950 bits
-0.4	inf	inf
-0.3	0.936967	inf
-0.2	1.24834	24.2
-0.1	0.92	18.2
-0.05	0.91	18.1
-0.025	0.91	17.9
0.05	0.91	18.642
0.1	0.94	18.7525
0.2	1.27	25.51
0.3	inf	inf

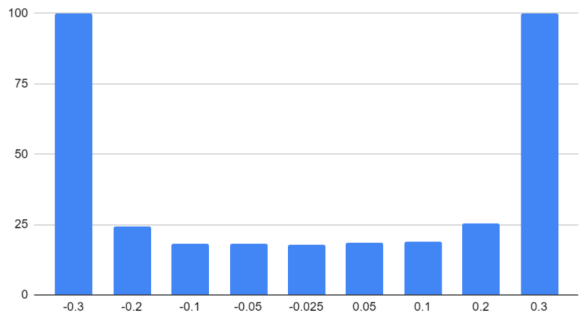

Figure 4.4: Left: Time for encrypting files of different sizes on a range of values of the coupling parameter ϵ. Right: Graphic representation of the encrypting times (vertical bars) on a range of ϵ for a file of 22601950 bits.Fixed parameters: $z = -0.25$, $\mu = 1.4$.

4.3 Different CMN system sizes

The proposed generalized method of Chapter 3 for encrypting bytes directly from any digital file is defined for two-dimensional ($N = 2$) CMN systems. This provides a binary pair or quaternary representation of the CMN dynamics. In principle, this encryption method can be extended for different CMN system sizes N. However, differences in the performances arise when considering this method for different CMN sizes.

The main distinction is that the size of the cipher file “*.ds" increases when decreasing the number of maps N in the CMN system. For example, if $N = 8$, each iteration of the CMN system generates one byte. The number of iterations between two successive byte symbols generated by a chain α can be represented as a complete byte that will be converted and stored as one byte in the cipher file. Therefore, the size of the file “*.ds" will be the same as the number of bytes in the input file. For a CMN system consisting of $N = 4$ maps, a byte B should be divided into 2 groups of 4 bits each. A chain α will correspond to a succession of 4-bits groups. Then, the size of the information to be processed and stored in the file “*.ds" will be twice that of the input file. For a CMN system with $N = 2$ maps, as we have seen, the size of information stored will be four times that of the input file.

Another aspect to take into account when changing the size N is the number of keys or the security of the cryptosystem. The size N is the number of initial conditions that are taken as keys. Thus, $N = 2$ implies 2 double-precision numbers for the initial state and for the key; $N = 4$ and $N = 8$ correspond to keys with 4 and 8 double-precision numbers, respectively. In this sense, increasing N enhances security.

However, the crucial factor for any cryptosystem is the speed of the encryption process. In this regard, we have computed the encryption time for files of different sizes when using our method with different numbers of maps. The results are shown in Figure 4.5. Although it does not constitute a CMN system, we have included the encryption times for one map using our method, for reference. As mentioned in the Introduction, cryptography based on one chaotic map has been used before[31].

The encryption time increases non-linearly when N is increased. The encryption time is more than triple when employing $N = 4$ maps instead of $N = 2$; while using $N = 8$ maps instead of $N = 2$ increases this time by a factor of 40.

		Time (seconds)			
File.extension	File size (bits)	1 Map	2 Maps	4 Maps	8 Maps
Small.jpg	1101219	0,73	1	3,5	44
Mid.mp4	22601950	15,15	20,1	58,3	846
vbig.wav	42085234	27	38	114,7	1500

Figure 4.5: Encryption times using a CMN system with different number of maps. File types and sizes are indicated on the first and second columns. Fixed parameters are $z = -0.25$, $\mu = 1.4$, $\epsilon = -0.025$.

The corresponding encryption speeds (i.e., the number of encrypted bits per second), are shown in Fig. 4.6. The comparison of the average speeds is also shown in Fig. 4.6. We notice that the increment in the encryption speed when using $N = 2$ maps is enormous (3 to 40 times faster) compared to the speeds corresponding to $N = 4$ and $N = 8$ maps.

	bits/second			
Bits processed	**1 map**	**2 maps**	**4 maps**	**8 maps**
1101219	1.508.519	1.101.219	314.634	25.028
22601950	1.491.878	1.124.475	387.684	26.716
42085234	1.558.712	1.107.506	366.916	28.057
Average	1.519.703	1.111.067	356.411	26.600

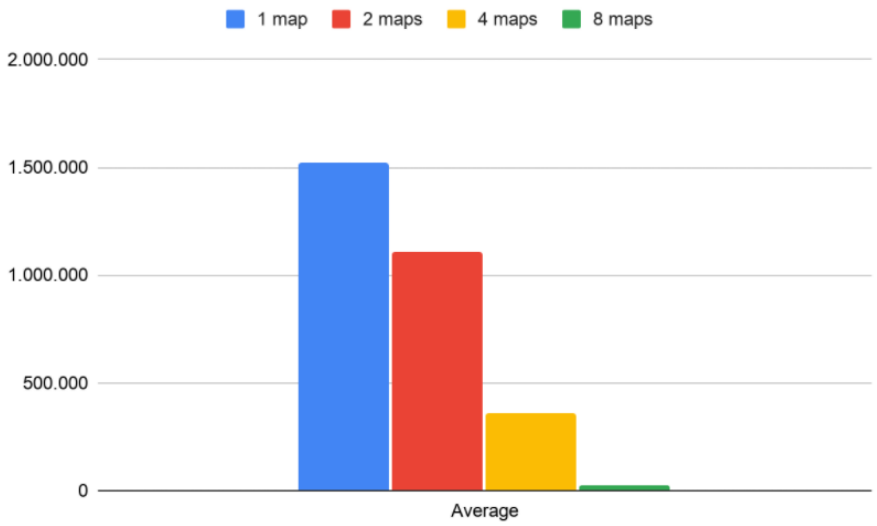

Figure 4.6: Top: Encryption speeds (bits/second) for CMN systems with different number of maps. Bottom: Graphic comparison of the average encryption speeds. Fixed parameters: $z = -0.25$, $\mu = 1.4$, $\epsilon = -0.025$.

In comparison, the relative speed increment when using one map instead of $N = 2$ maps is about 37%. Then, one may think that a single map would work better as a cryptosystem. However, the memory needed for storing the processed information for one map will be bigger, since the size of the cipher file "*.ds" for one map will be twice the corresponding size for an $N = 2$ map system (that is, 8 times larger than the original input file). Then, the average writing speed for a single map will be about half of that for an $N = 2$ CMN map system. In total, the encrypting time plus the writing time for a single map does not improve the total time for encoding and writing of a two-dimensional CMN system. Additionally, as we have mentioned, the security of a CMN system with $N = 2$ maps is twice that for a single map. Overall, a two-dimensional CMN map system exhibits the best performance for our generalized encryption method.

4.4 Applications

4.4.1 Text

As an application of the generalized encryption method presented in Chapter 3 for encrypting text files, consider the following statement from Edward Lorenz[17], the discoverer of the butterfly effect and pioneer of Chaos Theory:

> I shall probably never know to what extent my paper was responsible for setting off the outburst of activity that followed, and to what extent I was simply lucky that it became known when a scientific revolution was due to occur in any case.

The corresponding ciphertext and the decoded text with a slightly different initial condition are shown in Fig 4.7.

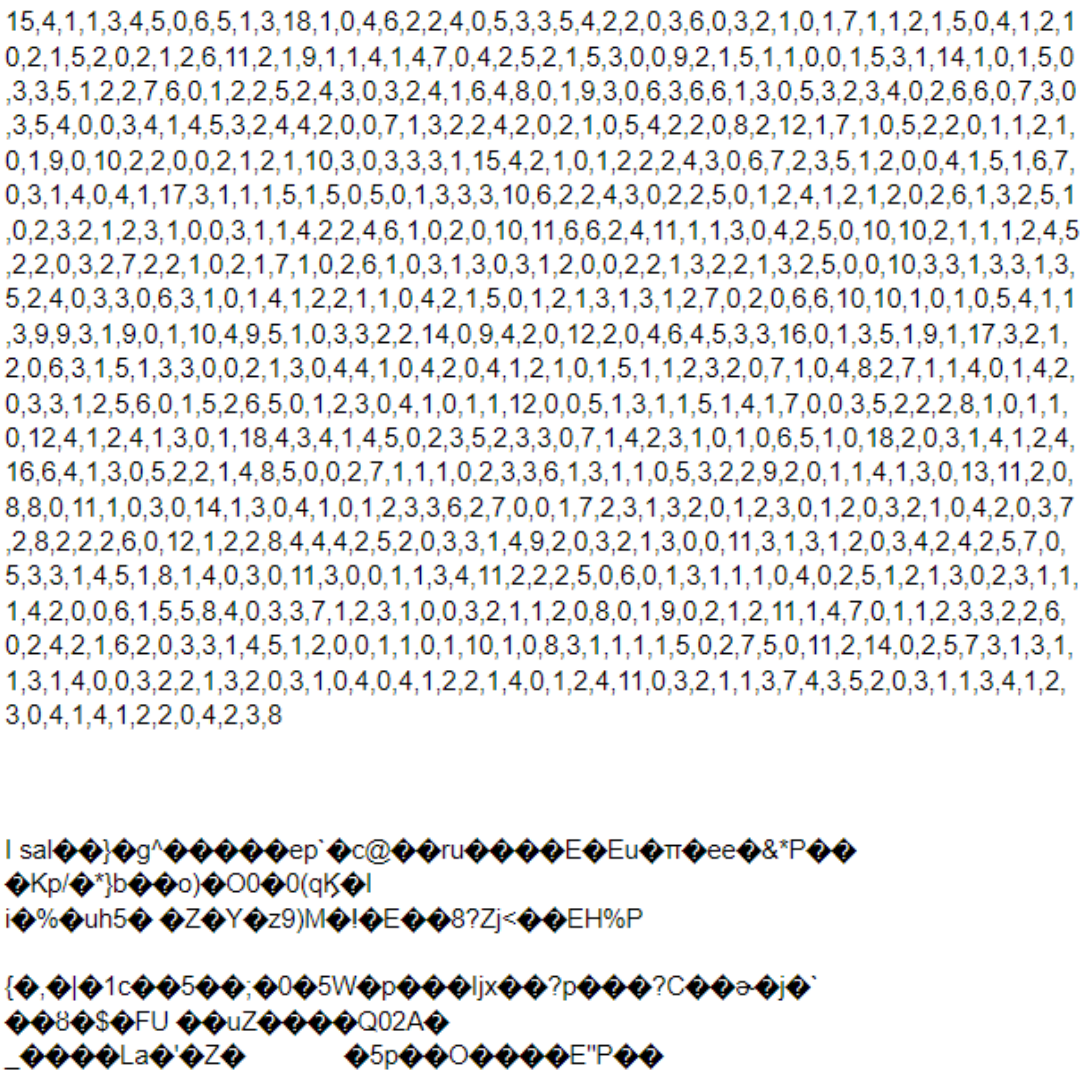

Figure 4.7: Top: Coding of the text by Lorenz using the generalized encryption method with initial condition $\mathbf{x}_0 = (10.0, 10.1)$. Each integer tells how many times the CMN system must be iterated to decode the text. Bottom: Decoded text with initial condition $\mathbf{x}_0 = (10.0000000000001, 10.1)$. The rhomboidal symbols correspond to non-printable ASCII characters. Fixed parameters for both processes: $\mu = 1.4$, $z = -0.25$, $\epsilon = -0.025$.

4.4.2 Color Images

As an example of the applications of the generalized encryption method for diverse file types, we present a .jpg color image of size of 101.231 bytes in Fig. 4.8 (Left). The resulting decoded image using slightly different initial conditions is shown in Fig. 4.8 (Right). We see that an error of 10^{-13} on just one initial condition returns a non-recognizable image.

Figure 4.8: Left: Original .jpg color image. Right: Decoded image with a small difference in initial conditions. The original image was codified with initial conditions $\mathbf{x}_0 = (10.0, 10.1)$ and decoded with $\mathbf{x}_0 = (10.0000000000001, 10.1)$. CMN fixed parameters are: $\mu = 1.4$, $z = -0.25$, $\epsilon = -0.025$.

4.4.3 App

We have created a software that implements the generalized encryption and storage method in an app, so that it can be used by anyone with a computer with Linux operating system. In order to create a friendly and simple GUI, we make use of Tkinter package. Note that the code for the GUI is made in python while the code for cypher is made in C. The Tkinter package ("Tk interface") is the standard Python interface to the Tk GUI toolkit. Both Tk and Tkinter are available on most Unix platforms, as well as on Windows systems. The app allows the user to choose between 4 options from an entrance window, as seen in Fig. 4.9 (Left):

- Code.
- Decode.
- Code Text.
- Decode Text.

The Code option allows to select among several files with different extensions, as seen in Fig. 4.9 (Right). A selected file of the form file.ext becomes the input for the generalized encryption algorithm when it is open in the box dialog. The corresponding encrypted file is stored in file.ds. The Decode option allows to select among cipher files of the type file.ds and to recover the associated original file.

The Code Text option is specifically dedicated to encrypt plain text. This option opens a window where any text to be coded can be directly typed by the user, as shown in Fig. 4.10 (Left). Selecting 'Code´ at the bottom will display the corresponding encrypted text as a sequence of integers. Similarly, the Decode Text option takes as input the ciphertext and returns the original plain text at the bottom, as seen in Fig. 4.10 (Right).

By employing this app, a user can safely share information by exchanging encrypted files (Fig. 4.9) or through encrypted text messages (Fig. 4.10).

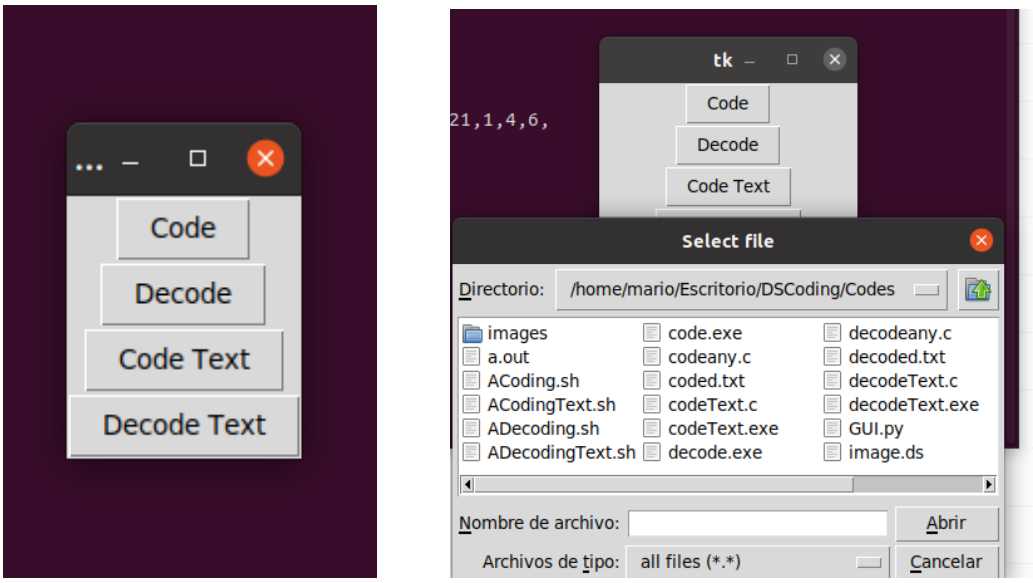

Figure 4.9: Left: App entrance window displaying the options: Code, Decode, Code Text, Decode Text. Right: File selector for the option Code. The selected file will be encrypted by the program.

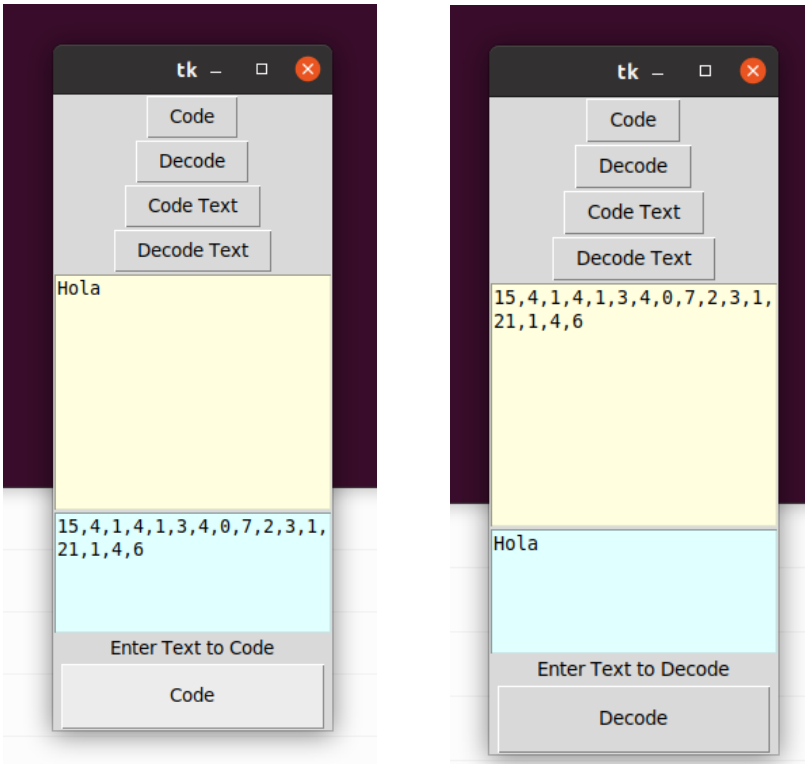

Figure 4.10: Left: Code Text option opens a window where a plain text to be encrypted can be directly typed by the user, in this case the word “Hola". The encrypted text appears in a window below. Right: Decode Text option recovers the original plain text.

4.5 Improving Security

The efficiency of any cryptosystem is given by a comprise between its speed and security. Encryption methods based on chaotic dynamics, such as ours, generally rely on the initial conditions as the secret key. As we have discussed, the initial condition for the generalized encryption method with the two-dimensional CMN system that gives the best performance provides a key size of 128 bis. This lies in the lower bounds of what is currently considered secure. In this section, we propose a variant of the method that greatly improves its security without substantially decreasing its speed.

Let us consider a CMN system consisting of N maps. The state of the system at a time t is given by the vector

$$\mathbf{x}_t \equiv (x_t^1, x_t^2, \ldots, x_t^N). \tag{4.4}$$

The components x_t^i, $i = 1, 2, \ldots, N$, are coupled to each other and evolve simultaneously. Suppose that we take only two state components to be expressed as binary pairs for the encryption process as before; for example $(x_1, x_2) \rightarrow (b_1, b_2)$. Then the generalized encryption method can be carried out as defined for 2 maps. However, since the other $N - 2$ maps are interacting with x_t^1 and x_t^2, their initial conditions and states affect the evolution of x_t^1 and x_t^2 and the corresponding sequence of binary pairs or quaternary symbols in a chain $\alpha(\mathbf{x}_0)$. Then, the initial condition $\mathbf{x}_0 \equiv (x_0^1, x_0^2, \ldots, x_0^N)$ can be employed as the key, without changing the size of storing cipher file “*.ds" that will still be four times the size of the input file. We denote this variation of the generalized encryption method as CMN-N-E2, where the number 2 after E indicates how many maps are being taken into account for the generation of symbols. The generalized encryption method introduced in Chapter 3 and employed in the applications corresponds to CMN 2.

As we saw, the number of possible keys provided by the initial condition of a CMN with 2 maps is $2^{64\times2}$, yielding a key size of 128 bits. For a CMN system with N maps, the number of possible keys will be $2^{64\times N}$, and the corresponding key size will be $64 \times N$.

Figure 4.11 shows the encryption times and speeds for different CMN system sizes and variants.

		Time (seconds)				
File	File size (bits)	CMN 2	CMN 4 E2	CMN 8 E2	CMN 16 E2	CMN 4
Small.jpg	1101219	1	1,42	2,14	3,8	3,5
Mid.mp4	22601950	20,1	28,06	42,6	73,12	58,3

	bits/second				
Bits processed	CMN 2	CMN 4 E2	CMN 8 E2	CMN 16 E2	CMN 4
1101219	1.101.219	775.506	514.588	289.794	314.634
22601950	1.124.475	805.486	530.562	309.108	387.684
Average	1.112.847	790.496	522.575	299.451	351.159

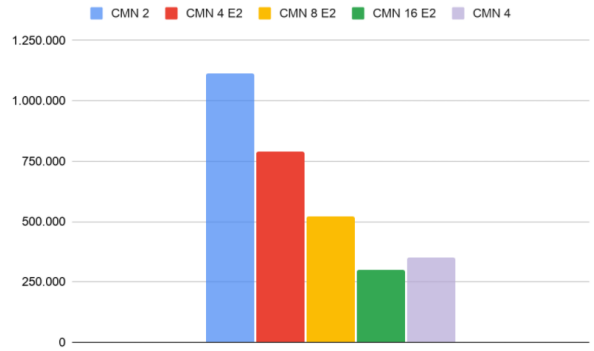

Figure 4.11: Left: Encryption times (top) and speeds (bottom) for different CMN systems and variants. Right: Graphical comparison of the average speeds (bits/second) for the different systems. CMN fixed parameters are: $\mu = 1.4$, $z = -0.25$, $\epsilon = -0.025$

The encryption speed decreases as the size N increases. However, this loss of speed is small when compared with the security that is being gained. For example, the variant CMN 4 E2 yields an encryption speed 25% less than that of the original CMN 2 system. Correspondingly, the number of keys for the CMN 4 E2 is $2^{(64\times4)}$ and the key size is 256, similar to the AES 256 cryptosystem currently regarded among the most common and secure on the Internet. The number of possible keys grows exponentially with N but the encryption speed decreases almost linearly with the variants of the method. For example, the encryption speed of the CMN8 E2 system is half of that of the original CMN2 system, but its key size is 512, offering much greater security than the AES cryptosystem.

Chapter 5

Conclusions and Outlook

The application of chaotic systems in cryptography has been a research topic of much interest since the rise of Chaos Theory at the end of the XX Century. In this thesis, we have proposed a general encryption method based on the phenomenon of chaos exhibited by nonlinear deterministic dynamical systems. Our method is a generalization of the scheme that uses a coupled map network (CMN) as a text encryption system, initially proposed in Ref.[37]. The consideration of a network of coupled chaotic elements instead of a single unit substantially increases the security and the range of applications of chaotic dynamics as cryptosystems.

The encryption capability of coupled map networks can be seen as an emerging functionality or property of an autonomous dynamical system of interacting elements, without any external influence. This is one of the main characteristics of complex systems, where nontrivial collective behaviors arise from the interactions among the constitutive parts of the system[56]. In particular, the study of complex systems is a very active field of interdisciplinary research in contemporary Sciences.

Our generalized encryption algorithm uses a coupled map network to directly encrypt bytes contained in any digital file, regardless of its type. Thus, our method extends the scope of the original scheme to include plain text, color images, audio, video, compressed files, etc. In addition, our method is integral, in the sense that it encompasses the optimal storage of the encrypted information. The method is conceptually simple, computationally efficient, and sufficiently secure.

We have employed a two-dimensional CMN system as a generator of strings of binary pairs or quaternary symbols that can be used to encode bytes. In this form, an input byte is expressed as four binary pairs that are codified by the algorithm as a sequence of four integer numbers. This results in a storing cipher file whose size is four times that of the input file. The splitting of bytes into binary pairs to be encrypted conforms to the idea of “divide-and-conquer" common in many problem-solving algorithms.

By comparing the encryption speeds for CMN systems of different sizes, we have shown that a two-coupled map system provides the best overall performance for the application of the method when considering encryption speed and security. The chaotic nature of the local dynamics allows the use of the initial condition as the secret key, providing a level of security within current standards. The security can be enhanced by considering the coupling parameters as an additional part of the key. On the other hand, the local robust chaos dynamics of our CMN system contains the parameters μ and z that can be varied and incorporated into the secret key while keeping the system functioning in chaos, to further increase the security of the cryptosystem. Furthermore, we have shown that the security can be greatly improved by considering a simple variant of the method, where the number of maps in the CMN is increased while keeping two maps for the generation of binary pairs or quaternary symbols.

We have shown applications of our method to encrypt text and image files, but any digital file can be encoded. We have illustrated the security of the method by decoding the corresponding cipher files with tiny differences in the initial conditions used for the coding process. By applying this method, any encrypted file could be distributed through diverse digital media with extra security besides the commonly employed encrypting methods on Internet, such as AES.

As a useful product of this thesis, we have created a software that implements the generalized encryption and storage method into a simple app for the Linux-Ubuntu operating system. This app facilitates the user to select files to be coded and decoded. It also offers the option of encrypting plain text that is typed directly in a displayed window. The resulting ciphertext consisting of a sequence of integers can also be seen in a window. The ciphertext can be decoded to yield the original plain text. The code for this app can be obtained in this Github link.

Several extensions of the generalized encryption method proposed here can be investigated in the future. A straightforward step would be the use of a password instead of the initial condition as the key. In this way, the recollection of the secret key would be more practical. This can be achieved by transforming the initial condition consisting of 128 bits into a password with 16 ASCII characters.

The simple app that we have implemented can be expanded to include more features and to make it more attractive and user-friendly. The app can be developed for other platforms and operating systems, such as Windows or Android.

Since the parameters of the CMN system determine the probability of occurrence of symbols and the transition probability of generating symbol w^{j-1} followed by symbol w^j in the chain, it is possible in principle to select the CMN parameters to enhance or to inhibit some symbols and transitions. This is analogous to selecting grammatical rules. Thus, the CMN system as a generator of symbols could be of interest in the investigation of language models.

Bibliography

[1] Lorenz, E. N. Deterministic nonperiodic flow. *Journal of Atmospheric Sciences* **1963**, *20*, 130–141.

[2] Sharkovskii, A. N. Coexistence of cycles of a continuous map of the line into itself. *Urain. Mat. Zh.* **1964**, *16*, 61–71.

[3] Li, T.-Y.; Yorke, J. A. Period three implies chaos. *The American Mathematical Monthly* **1975**, *82*, 985–992.

[4] Ruelle, D. The Mathematical Intelligencer. *Springer-Verlag, New York* **1980**, *2*, 126.

[5] Sharkovskii, A. N. Coexistence of cycles of a continuous map of the line into itself. *International Journal of Bifurcation and Chaos* **1995**, *5*, 1263–1273.

[6] Rössler, O. E. An equation for continuous chaos. *Physics Letters A* **1976**, *57*, 397–398.

[7] Hénon, M. *A two-dimensional mapping with a strange attractor*; Springer, 1976; pp 94–102.

[8] Feigenbaum, M. J. Quantitative universality for a class of nonlinear transformations. *Journal of Statistical Physics* **1978**, *19*, 25–52.

[9] May, R. M. Simple mathematical models with very complicated dynamics. *Nature* **1976**, *261*, 459–467.

[10] Zhabotinsky, A. M. Periodic liquid phase reactions. *Proc. Acad. Sci. USSR* **1964**, *157*, 392–395.

[11] Zaikin, A. N.; Zhabotinsky, A. M. Concentration wave propagation in two-dimensional liquid-phase self-oscillating system. *Nature* **1970**, *225*, 535–537.

[12] Madan, R. N. *Chua's circuit: a paradigm for chaos*; World Scientific, 1993; Vol. 1.

[13] Gleick, J. *Chaos: Making a New Science*; Open Road Media, 2011.

[14] Stewart, I. *Does God play dice?: The New Mathematics of Chaos*; Penguin UK, 1997.

[15] Abraham, R. H.; Ueda, Y. *The Chaos Avant-Garde. Memories of the Early Days of Chaos Theory*; World Scientific, 2001; Vol. 39.

[16] Lorenz, E. N. Section of planetary sciences: The predictability of hydrodynamic flow. *Transactions of the New York Academy of Sciences* **1963**, *25*, 409–432.

[17] Lorenz, E. N. The Essence of Chaos. *Pure and Applied Geophysics* **1996**, *147*, 598–599.

[18] Hayes, S.; Grebogi, C.; Ott, E. Communicating with chaos. *Physical Review Letters* **1993**, *70*, 3031–3034.

[19] Brown, R.; Chua, L. O. Clarifying chaos: Examples and Counterexamples. *International Journal of Bifurcation and Chaos* **1996**, *6*, 219–249.

[20] Fridrich, J. Symmetric ciphers based on two-dimensional chaotic maps. *International Journal of Bifurcation and chaos* **1998**, *8*, 1259–1284.

[21] Kocarev, L.; Jakimoski, G.; Stojanovski, T.; Parlitz, U. From chaotic maps to encryption schemes. *Proceedings of the 1998 IEEE International Symposium on Circuits and Systems* **1998**, 514–517.

[22] Alvarez, G.; Montoya, P.; Pastor, G.; Romera, M. Chaotic cryptosystems. *Proceedings IEEE 33rd Annual 1999 International Carnahan Conference on Security Technology* **1999**, 332–338.

[23] Gotz, M.; Kelber, K.; Schwarz, W. Discrete-time chaotic encryption systems. I. Statistical design approach. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications* **1997**, *44*, 963–970.

[24] Shujun, L.; Xuanqin, M.; Yuanlong, C. Pseudo-random bit generator based on couple chaotic systems and its applications in stream-cipher cryptography. **2001**, *2247*, 316–329.

[25] Shannon, C. E. Communication theory of secrecy systems. *The Bell system technical journal* **1949**, *28*, 656–715.

[26] Wolfram, S. Cryptography with cellular automata. *Conference on the Theory and Application of Cryptographic Techniques* **1985**, *218*, 429–432.

[27] Guan, P. Cellular automaton public-key cryptosystem. *Complex Systems* **1987**, *1*, 51–57.

[28] Chirikov, B. V.; Vivaldi, F. An algorithmic view of pseudochaos. *Physica D: Nonlinear Phenomena* **1999**, *129*, 223–235.

[29] Kocarev, L. Chaos-Based Cryptography: A Brief Overview. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications* **2000**, 1531–636X.

[30] Li, S.-J. Analysis and new designs of digital chaotic ciphers. Ph.D. thesis, Xi'an Jiaotong University, 2003.

[31] Baptista, M. S. Cryptography with chaos. *Physics Letters A* **1998**, *240*, 50–54.

[32] Alvarez, E.; Fernández, A.; Garcıa, P.; Jiménez, J.; Marcano, A. New approach to chaotic encryption. *Physics Letters A* **1999**, *263*, 373–375.

[33] Wong, W.-K.; Lee, L.-P.; Wong, K.-W. A modified chaotic cryptographic method. *Computer Physics Communications* **2001**, *138*, 234–236.

[34] Li, S.; Mou, X.; Cai, Y. Improving security of a chaotic encryption approach. *Physics Letters A* **2001**, *290*, 127–133.

[35] Palacios, A.; Juarez, H. Cryptography with cycling chaos. *Physics Letters A* **2002**, *303*, 345–351.

[36] Wong, K.-W. A fast chaotic cryptographic scheme with dynamic look-up table. *Physics Letters A* **2002**, *298*, 238–242.

[37] Garcia, P.; Parravano, A.; Cosenza, M. G.; Jiménez, J.; Marcano, A. Coupled map networks as communication schemes. *Physical Review E* **2002**, *65*, 045201.

[38] García, P.; Jiménez, J. Communication through chaotic map systems. *Physics Letters A* **2002**, *298*, 35–40.

[39] Wong, K.-W. A combined chaotic cryptographic and hashing scheme. *Physics Letters A* **2003**, *307*, 292–298.

[40] Wong, K.-W.; Ho, S.-W.; Yung, C.-K. A chaotic cryptography scheme for generating short ciphertext. *Physics Letters A* **2003**, *310*, 67–73.

[41] Urıas, J.; Ugalde, E.; Salazar, G. A cryptosystem based on cellular automata. *Chaos: An Interdisciplinary Journal of Nonlinear Science* **1998**, *8*, 819–822.

[42] Kocarev, L.; Jakimoski, G. Logistic map as a block encryption algorithm. *Physics Letters A* **2001**, *289*, 199–206.

[43] 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* **2001**, *48*, 163–169.

[44] Li, S.; Zheng, X.; Mou, X.; Cai, Y. Chaotic encryption scheme for real-time digital video. **2002**, *4666*, 149–160.

[45] Matthews, R. On the derivation of a "chaotic" encryption algorithm. *Cryptologia* **1989**, *13*, 29–42.

[46] Perez, G.; Cerdeira, H. Extracting Messages Masked by Chaos. *Physical Review Letters* **1995**, *74*, 1970.

[47] Yang, T.; Yang, L.; M., Y. C. Cryptanalyzing chaotic secure communications using return maps. *Physics Letters A* **1998**, *245*, 495–510.

[48] Alvarez, G.; Montoya, F.; Romera, M.; Pastor, G. Cryptanalysis of a chaotic encryption system. *Physics Letters A* **2000**, *276*, 191–196.

[49] Kaneko, K. Period-doubling of kink-antikink patterns, quasiperiodicity in antiferro-like structures and spatial intermittency in coupled logistic lattice: Towards a prelude of a "field theory of chaos". *Progress of Theoretical Physics* **1984**, *72*, 480–486.

[50] Waller, I.; Kapral, R. Spatial and temporal structure in systems of coupled nonlinear oscillators. *Physical Review A* **1984**, *30*, 2047.

[51] Crutchfield, J. P. Space-time dynamics in video feedback. *Physica D: Nonlinear Phenomena* **1984**, *10*, 229–245.

[52] Kuznetsov, S.; Pikovskii, A. Universality of period-doubling bifurcations in one-dimensional dissipative media. *Radiofizika* **1985**, *28*, 308–319.

[53] Cosenza, M. G.; Kapral, R. Coupled maps on fractal lattices. *Physical Review A* **1992**, *46*, 1850.

[54] Kaneko, K. Theory and Applications of Coupled Map Lattices. *Chaos, Focus issue in Coupled Map Lattices* **1992**, *2*.

[55] Kaneko, K. From globally coupled maps to complex-systems biology. *Chaos: An Interdisciplinary Journal of Nonlinear Science* **2015**, *25*, 097608.

[56] Kaneko, K.; Tsuda, I. *Complex Systems: Chaos and Beyond*; Springer, 2001.

[57] Cosenza, M. G.; González, J. Synchronization and collective behavior in globally coupled logarithmc maps. *Progress of Theoretical Physics* **1998**, *100*, 21–38.

[58] Cosenza, M. G.; Alvarez-Llamoza, O.; Ponce, G. A. Critical behavior of the Lyapunov exponent in type-III intermittency. *Chaos, Solitons and Fractals* **2008**, *36*, 150–156.

Appendices

Appendix A

Code for Encryption any file

The following code is made in C. The input file will be coded into a .ds file with the same name. It must be compiled with -lm flag.

```
#include <sys/stat.h>
#include <sys/types.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <math.h>
#include <time.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <string.h>
#include <stdio.h>
#include <errno.h>

#define w 1
#define ww w*4
#define N 2


off_t fsize(const char *filename) {
    struct stat st;
    if (stat(filename, &st) == 0)
        return st.st_size;
    return -1;
}

int g(int c1, int c2, double x[],double epsilon)
{
    int iter=0;
    double tmp;
    int x2,x1;
```

```
    double treshold=-0.03;
while(1)
{
        x1= x[0]<treshold ? 0:1;
        x2= x[1]<treshold ? 0:1;
if(x1==c1 && x2==c2)return iter;
else {
            //Coupled Map
            tmp= (x[0]+x[1])*epsilon;
            x[0]=(1.4-pow(fabs(x[0]),-0.25))+tmp;
            x[1]=(1.4-pow(fabs(x[1]),-0.25))+tmp;
iter++;
}
}
}

void CodeHeader(double epsilon,int lenExt,char Ext[], int output[], double x[]){
    int i,j;
    j=0;
    unsigned char buhherHeader[lenExt+1];
    printf("LEN EXT:%d \n",lenExt);
    printf("EXT: %s asd\n\n",Ext);
    printf("\n");
}

void CodeLine(double epsilon,char input[], int output[], double x[]){
    int i,j;
    j=0;
    for(i=0; i<8; i+=2){
        output[j]=g(input[i]-48,input[i+1]-48,x,epsilon);
        j++;
    }
}

int main(int argc, char **argv){
    //
    clock_t start = clock();
    double epsilon=-0.025;
    //DECLARATIONS
    FILE *pFile, *pData;
    int i,j;
    unsigned char buffer1[1],buffer4[4];
    char hexa[3],binary[9];
    int coded[ww];
    double x0[N];
    //x0 is the vector of initial conditions. When decoding, x0 must be exactly the same as when coding
```

```
for(i=0; i<N; i++) x0[i]=10.0+(0.1*i);
x0[0]=10.0;
x0[1]=10.1;
//GET FILE DATA
const char* filename=argv[1];
int size = fsize(filename);
pFile=fopen(filename,"rb");
int lenName=strlen(filename);
char filenameChar[lenName+1];
for(i=0; i<lenName+1; i++) filenameChar[i]=filename[i];
//DS FILE
char *token=strtok(filenameChar,".");
char dataName[lenName];
sprintf(dataName,"%s%s", token,".ds");
pData=fopen(dataName,"wb");
token = strtok(NULL, ".");
char bufferH[lenName+1];
sprintf(bufferH,"%s", token);
//PRINTS
printf("NameLen: %ld\n",strlen(token));
printf("FORMAT: %s\n", bufferH);
printf("Filename: %s\n",filename);
printf("Size of File: %d\n",size);
int h=size/w;
//**********************WRITTE HEADER*********************************
//WRITE LEN EXT
//Get Line Len
    for(i=0; i<lenName; i++)if(bufferH[i]=='\0')break;
    int lenExt=i;
    printf("Len de extension:%d\n",lenExt);
    //Writte len in file
        buffer1[0]=lenExt& 0xFF;
        fwrite(buffer1, 1, sizeof(buffer1), pData);
    //Writte EXT in file
        fwrite(bufferH, 1, lenExt, pData);
    CodeHeader(epsilon,lenExt,bufferH,coded,x0);
//***************READ AND WRITE FROM FILE TO DS*****************************
int percentage= h/10;
for(i=0; i<size; i++){
    if(i%percentage==0)printf("Progress: %.0f/100\n", ((float)i/h)*100);
    fread(buffer1,sizeof(buffer1),1,pFile);
    sprintf(hexa,"%d",buffer1[0]);
    for (j = 0; j < 8; j++) sprintf((char*)(binary+j),"%d", !!((buffer1[0] << j) & 0x80));
    //AQUI DEBEMOS CIFRAR
    CodeLine(epsilon,binary,coded,x0);
    //WRITE TO DS FILE
```

```
        for(j=0; j<ww; j++) buffer4[j]=coded[j]& 0xFF;
        fwrite(buffer4, 1, sizeof(buffer4), pData);
    }
    fclose(pFile);
    fclose(pData);
    printf("Tiempo transcurrido: %f\n", ((double)clock() - start) / CLOCKS_PER_SEC);
    return 0;
}
```

Appendix B

Code for Decryption any file

The following code is made in C. The input file will be a .ds file and the output will be a decoded file with its corresponding extension and the same name. It must be compiled with -lm flag.

```
#include <sys/stat.h>
#include <sys/types.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <math.h>
#include <time.h>

#include <sys/stat.h>
#include <sys/types.h>
#include <string.h>
#include <stdio.h>
#include <errno.h>

#define w 1
#define ww w*4
#define N 2

off_t fsize(const char *filename) {
    struct stat st;

    if (stat(filename, &st) == 0)
        return st.st_size;

    return -1;
}
void CoupledMap(double x[],double epsilon)
{
    int i;
double tmp=0;                       //e
```

```
    for(i=0; i<N; i++) tmp=tmp+x[i];    //e
    tmp=tmp*epsilon;                    //e
    for(i=0;i<N;i++) x[i]=(1.4-pow(fabs(x[i]),-0.25))+tmp;  //e
//compile with -lm
    }
int Bin2Int(int aBin[]) {
  if (aBin[0]==0){
      if (aBin[1]==0) return 0;
      else return 1;
  }else
  {
      if (aBin[1]==0) return 2;
      else return 3;
  }

}
int ReadCuat(int a, int b,int c, int d){
    char QUAD[5]={a,b,c,d,'\0'};
    return(int)strtol(QUAD, NULL, 4);
}

int Decode(int hexas[],double epsilon,double x0[]){
    int i,k,st_bin[N],quad_linea[ww];
    for(i=0; i<4; i++){
        for(k=0; k<hexas[i]; k++)CoupledMap(x0,epsilon);
        st_bin[0]= x0[0]<0 ? 0:1;
        st_bin[1]= x0[1]<0 ? 0:1;
        quad_linea[i]=(Bin2Int(st_bin))+48; //sum 48 to read as char on ReadCuat()
    }
    return ReadCuat(quad_linea[0],quad_linea[1],quad_linea[2],quad_linea[3]);
}

int main(int argc, char **argv){
    clock_t start = clock();
    double epsilon=-0.025;
    int i,j,read;
    //DECLARATIONS
    FILE *pInput, *pOutput;
    unsigned char buffer1[1],buffer4[4];
    char buf[4];
    double x0[N];
    //x0 is the vector of initial conditions. When decoding, x0 must be the same as when coding.
    for(i=0; i<N; i++) x0[i]=10.0+(0.1*i); //Reiniciamos x0
    x0[0]=10.0;
    x0[1]=10.1;
    //FILE DATA
```

```
    const char* filename=argv[1];
    int size = fsize(filename);
    pInput=fopen(filename,"rb");
    int h=size/w;
    int res= size-(w*h);
    char hexa[4];
    int hexas[4],outH;
    //GET LEN OF EXT AND EXT
    read=fread(buffer1,sizeof(buffer1),1,pInput);
    sprintf(buf,"%d",buffer1[0]);
    int lenExt=atoi(buf);
    printf("lenExt: %d\n",lenExt);
    unsigned char bufferH[lenExt+1];
    read=fread(bufferH,lenExt,1,pInput);
    bufferH[lenExt]='\0';
    printf("Ext: %s\n",bufferH);
    //OUTPUT
    int lenName= strlen(filename);
    char out[lenName+lenExt];
    char filenameChar[lenName+1];
    for(i=0; i<lenName+1; i++) filenameChar[i]=filename[i];
    char *token=strtok(filenameChar,".");
    sprintf(out,"%s.%s", token,bufferH);
    pOutput=fopen(out,"wb");
    //READ AND WRITE FROM FILE TO DS
    int percentage= h/10;
    printf("size: %d\n",size);
    for(i=0; i<size-(lenExt+1); i+=4){
        if(i%percentage==0)printf("Progress: %.0f/100\n", ((float)i/h)*100);
        for (j = 0; j < 4; j++)
        {
            read=fread(buffer1,sizeof(buffer1),1,pInput);
            sprintf(hexa,"%d",buffer1[0]);
            hexas[j]=atoi(hexa);
        }
        //DECODE
        outH=Decode(hexas,epsilon,x0);
        buffer1[0]=outH & 0xFF;
        fwrite(buffer1, 1, sizeof(buffer1), pOutput);
    }
    //CLOSE FILES
    fclose(pInput);
    fclose(pOutput);
}
```