**UNIVERSIDADE DE SÃO PAULO
INSTITUTO DE FÍSICA DE SÃO CARLOS**

MARINA JEANETH MACHICAO JUSTO

**Autômatos celulares caóticos aplicados na
Criptografia e Criptoanálise**

São Carlos
2013

MARINA JEANETH MACHICAO JUSTO

Autômatos celulares caóticos aplicados na Criptografia e Criptoanálise

Dissertação apresentada ao Programa de
Pós-graduação em Física do Instituto de Física
de São Carlos da Universidade de São Paulo,
para a obtenção do título de Mestre em Ciências.

Área de Concentração: Física Aplicada
Opção: Física Computacional
Orientador: Prof. Dr. Odemir Martinez Bruno

Versão Corrigida

(Versão original disponível na Unidade que aloja o Programa)

São Carlos
2013

AUTORIZO A REPRODUÇÃO E DIVULGAÇÃO TOTAL OU PARCIAL DESTE
TRABALHO, POR QUALQUER MEIO CONVENCIONAL OU ELETRÔNICO,
PARA FINS DE ESTUDO E PESQUISA, DESDE QUE CITADA A FONTE.

Ficha catalográfica elaborada pelo Serviço de Biblioteca e Informação do IFSC,
com os dados fornecidos pelo(a) autor(a)

Machicao, Jeaneth
Autômatos celulares caóticos aplicados na
Criptografia e Criptoanálise / Jeaneth Machicao;
orientador Odemir Martinez Bruno - versão corrigida -
- São Carlos, 2013.
121 p.

Dissertação (Mestrado - Programa de Pós-Graduação em
Física Aplicada Computacional) -- Instituto de Física
de São Carlos, Universidade de São Paulo, 2013.

1. Autômato celular caótico. 2. Criptografia. 3.
Criptoanálise. 4. Reconhecimento de padrões. 5.
Redes complexas. I. Martinez Bruno, Odemir, orient.
II. Título.

Dedico este trabalho aos meus pais Gonzalo e Alicia

Agradecimentos

A minha mãe Alicia pelas orações e pelas tantas mostras de carinho. Ao meu pai Gonzalo pelas lições de vida e por sempre ter acreditado em mim. Aos meus irmãos César, Carlos, Jonathan e Fredy; pelos momentos gratos em família.

Ao meu amado João pelo carinho, paciência, e apoio constante durante este empreendimento tão árduo.

Ao meu orientador Prof. Odemir Martinez Bruno, pela orientação, motivação e confiança em mim, durante o desenvolvimento deste trabalho.

Aos meus professores da graduação Prof. César Beltrán Castañón e Juan Carlos Gutierrez, pela apoio e incentivo ao realizar este sonho.

A todos os colegas do grupo, pelo companheirismo e ajuda em tantas dúvidas tiradas. Especialmente ao Anderson pelas tantas dicas de programação.

A todos meus amigos, pelos momentos de alegria e descontração.

Ao Instituto de Física de São Carlos (IFSC-USP), aos professores e funcionários. Em especial à Cristina da biblioteca, pela simpatia e ajuda nos últimos estágios de desenvolvimento deste trabalho. E a todos que, direta ou indiretamente, colaboraram neste trabalho.

A Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), pela concessão da bolsa de Mestrado.

Por último mas não menos importante, meu eterno agradecimento a todos os funcionários, médicos e enfermeiros do Hospital fornecedores de cana de Piracicaba.

Resumo

MACHICAO, J. *Autômatos celulares caóticos aplicados na Criptografia e Criptoanálise*. 2013. 121 p. Dissertação (Mestrado em Ciências) - Instituto de Física de São Carlos, Universidade de São Paulo, 2013.

A teoria do caos estuda o tipo de comportamento, aparentemente aleatório, que apresentam alguns sistemas complexos sensíveis à perturbação dos seus parâmetros, como por exemplo sistemas dinâmicos, fractais, autômatos celulares, entre outros. Os autômatos celulares (ACs) são sistemas dinâmicos discretos que podem apresentar comportamentos caóticos a partir de regras simples. Os ACs tem sido empregados em diversas aplicações principalmente em simulações, mas também tem contribuído no reconhecimento de padrões, processamento de imagens e na Criptografia. A necessidade em transmitir informação de forma mais segura vem crescendo com a necessidade por novos algoritmos criptográficos. Paralelamente, os criptoanalistas vem progredindo constantemente na quebra e na procura de vulnerabilidades destes algoritmos, sendo necessaria a incursão de novas abordagens para atender estes desafios. Neste trabalho é proposto o desenvolvimento e avaliação de algoritmos criptográficos, assim como um novo método de criptoanálise, motivados pela adequação dos ACs caóticos com os princípios de confusão e difusão da Criptografia, seguindo critérios apropriados para a boa construção destes algoritmos, que são sintetizados em três partes: (i) Na proposta do algoritmo de cifra criptográfico baseado no AC caótico, foi sugerida uma estratégia de seleção de ACs em base a combinação de vários critérios como o expoente de Lyapunov, a entropia e a distância de Hamming; visando selecionar um AC apropriado para a geração de números pseudo-aleatórios usados no processo de encriptação/decriptação do algoritmo, o qual é validado por diversos testes de aleatoriedade. (ii) Foi proposto o algoritmo de *hash* criptográfico baseado numa abordagem híbrida dos ACs e as redes complexas, visando a construção de um algoritmo flexível e de bom desempenho. Os resultados alcançados por ambos os algoritmos criptográficos mostraram-se relevantes quando comparados com o estado da arte, com boas qualidades de segurança e um grande potencial para ser aplicados em problemas reais. (iii) Na proposta do método de criptoanálise foi sugerido traçar equivalências entre os sistemas criptográficos e os ACs caóticos visando explorar e analisar seu comportamento dinâmico, por meio da adaptação do algoritmo do expoente de Lyapunov dos ACs, cujos resultados permitiram encontrar padrões característicos nos modos de operação criptográficos. Os resultados obtidos mostraram que a abordagem dos ACs caóticos para desenvolver os algoritmos pode ser bastante útil em aplicações de Criptografia e Criptoanálise.

Palavras-chave: Autômato celular caótico. Criptografia. Criptoanálise. Reconhecimento de padrões. Redes complexas.

Abstract

MACHICAO, J. *Chaotic cellular automata applied to Cryptography and Cryptanalysis*. 2013. 121 p. Dissertação (Mestrado em Ciências) - Instituto de Física de São Carlos, Universidade de São Paulo, 2013.

Chaos theory studies the apparently random behaviour from some complex systems with highly sensitive to the initial conditions, such as dynamical systems, fractals, cellular automata, among others. Cellular automata (CA) are discrete dynamical systems that may exhibit chaotic behaviour from simple rules. CA have been employed in many multidisciplinary applications, most of them in simulations systems, including pattern recognition, image processing and Cryptography. Nowadays, the development of new cryptographic algorithms is required in order to fulfil the increasing demand for secure transmission of confidential information. These algorithms are intensively analyzed, most of them broken by the cryptanalyst community. We proposed to develop two cryptographic algorithms: a block cipher and a hash function based on chaotic CA and its corresponding evaluation. We also proposed a new cryptanalysis methodology motivated by the strong relationship between the chaotic properties of CA and the cryptographic principles of confusion and diffusion, by following appropriate criteria to the proper design of these algorithms, which are summarized into three parts: (i) To proposed the block cipher proposed it was suggested a methodology to select a suitable CA to Cryptography by means of compounded measures such as the Lyapunov exponent, entropy and Hamming distance. Moreover, this selected CA is employed to generate pseudo-random numbers, which are further used in the encryption/decryption of the proposed block cipher and validated under several randomness tests. The results obtained by this cryptographic algorithm achieved similar and even higher performance when compared to others found in literature. (ii) The cryptographic hash function was developed using an hybrid approach of CA and complex networks, in order to build a flexible algorithm with acceptable performance when compared to conventional hash functions. In general, the results obtained from both cryptographic algorithms showed good security qualities and great potential to be applied in real problems. (iii) To proposed the cryptanalysis methodology it was suggested to draw parallels between cryptographic systems and CA, in order to explore and analyse their dynamic behaviour. Hence, upon drawing such parallels, we have a means to adapt the Lyapunov exponent algorithm conceived in the framework of CA. Unexpectedly, the results obtained allow to discriminate among cryptographic modes of operation, which provides significant contributions to the field. Finally, we proved that the chaotic cellular automata approach can be quite useful in applications cryptography and cryptanalysis.

Keywords: Chaotic cellular automata. Cryptography. Cryptanalysis. Pattern recognition. Complex network.

Lista de Figuras

- Figura 1.0.1 - Na esquerda, um AC elementar usando a regra 30 do Wolfram, a partir de um estado inicial “desordenado”, em que cada célula tem valor 0 ou 1. Na direita, o molusco *Conus textile* com um padrão de pigmentação que lembra o padrão gerado pelo AC. Adaptado de (4, 88). 27
- Figura 2.1.1 - Evolução temporal de duas trajetórias da Equação (2.1.1) com condições iniciais $X_0 = 0.9$ e $X'_0 = 0.90001$, para os parâmetros (a) $\mu = 3.57$, Periódico; (b) $\mu = 3.9$, Caótico e (c) $\mu = 2.85$, Estável. 32
- Figura 2.3.1 - Exemplos de 1D- e 2D-AC com diferentes tipos de vizinhança. Em (a) e (b) ACE com célula central em preto e a vizinhança em cinza de raio $r = 1$ e $r = 2$ respectivamente, da mesma maneira, em (c) e (d) vizinhança de Von Neumann, e em (e) e (f) vizinhança de Moore. Adaptado de (71). 35
- Figura 2.4.1 - Exemplo de ACE de raio $r = 1$ e com $k = 2$ estados cinza (1) e branco (0), evoluindo com a regra 30 do Wolfram. 37
- Figura 2.4.2 - Classificação qualitativa dos ACE do Wolfram. As figuras mostram o diagrama espaço-tempo do ACE de 600 células com uma configuração inicial aleatória durante 400 iterações (91). 38
- Figura 2.5.1 - Exemplo de uso da regra de transição do “Jogo da vida” do Conway, dada na Equação (2.5.1). A cor cinza representa uma célula viva, e a cor branca representa uma célula morta. 39
- Figura 2.5.2 - Evolução do “Jogo da vida”. A cor preta representa uma célula viva, e a cor branca representa uma célula morta. As Figuras mostram a evolução do AC simulado numa malha de 200×200 e condição inicial aleatória durante 800 iterações. 39

Figura 2.5.3 - Evolução de diferentes regras da família “ <i>life-like</i> ”. A cor preta representa uma célula viva, e a cor branca representa uma célula morta. As Figuras mostram a evolução do AC simulado numa malha de 128×128 e condição inicial aleatória após 1000 iterações.	41
Figura 2.6.1 - Evolução de três ACs usando a regra B1357/S2468. Na esquerda o AC com configuração inicial aleatória e, no centro e na direita a mesma configuração porém com uma célula perturbada. A cor preta representa uma célula viva, e a cor branca representa uma célula morta. As figuras mostram a evolução do AC simulado numa malha de 16×16 para $t = 100$ e $t = 101$ iterações.	42
Figura 2.7.1 - (a) Exemplo de tesselação irregular. (b) Rede complexa: grafo não orientado modelando o exemplo de tesselação dado em (a).	43
Figura 2.7.2 - Modelamento de AC empregando uma rede complexa como tesselação do AC (34).	43
Figura 2.7.3 - Exemplo de evolução do modelo 2D-AC caótico híbrido como uma rede complexa de 128 vértices (células) e $\langle \bar{k} \rangle = 8$ (vizinhança) com a regra B1357/S2468. A cor cinza representa uma célula viva, e a cor azul representa uma célula morta.	44
Figura 3.1.1 - Taxonomia da Criptografia. Adaptado de (60).	48
Figura 3.1.2 - Modo de operação ECB (79).	50
Figura 3.1.3 - Modo de operação CBC (79).	51
Figura 3.1.4 - Modo de operação OFB (79).	52
Figura 3.1.5 - Modo de operação CFB (79).	53
Figura 3.1.6 - Modo de operação CTR (79).	54
Figura 3.1.7 - Modo de operação PCBC.	55
Figura 3.1.8 - Princípios de construção de uma função <i>hash</i> baseado no modelo Merkle-Damgård (41).	56
Figura 4.3.1 - Evolução do 2D-AC B1357/S2468 com diferentes configurações iniciais (56).	69

Figura 5.1.1 - Proposta de um algoritmo de criptografia baseado em AC caótico, dividido em quatro componentes principais: (a) A transformação da chave na condição inicial do AC usando o mapa logístico, (b) O AC é definido com a regra B1357/S2468 e é iterada sucessivas vezes. (c) O AC atua como um PRNG para gerar blocos de números pseudo-aleatórios. (d) O processo de encriptação aplica operação de XOR entre os textos claros e os blocos gerados. Nota-se que o processo de deciptação emprega o <i>design</i> inverso.	72
Figura 5.1.2 - Encriptação de uma imagem e suas respectivas imagens cifradas a) e b) com o algoritmo proposto, usando duas chaves com um bit de diferença (esquerda) e a distribuição de frequências da imagem original e das respectivas imagens cifradas (direita).	75
Figura 5.2.1 - Proposta do algoritmo de <i>Hash</i> caótico baseado em AC híbrido, de acordo com o modelo Merkle-Damgård.	80
Figura 5.2.2 - Estrutura de dados da matriz de arestas do AC híbrido.	81
Figura 5.2.3 - <i>Hashcodes</i> das diferentes condições.	84
Figura 5.2.4 - Comparação da velocidade por número de blocos calculados entre o Whirlpool e o algoritmo de <i>hash</i> caótico proposto.	86
Figura 6.3.1 - (a) Distribuição das frequências do logaritmo base 10 do desvio padrão σ_λ para MLE λ_{200} calculado para as 60 combinações de sistemas criptográficos. (b) Distribuição de frequências $\Delta = \max(\{e\mathbf{P} \mid e = 1, \dots, 200\}) - \min(\{e\mathbf{P} \mid e = 1, \dots, 200\})$	94
Figura 6.3.2 - Gráfico do expoente de Lyapunov em função do número de iterações para os diferentes modos de operação: ECB, CBC, OFB, CFB, CTR e PCBC, usando as cifras de blocos (a) IDEA de 64 bits e (b) AES de 128 bits. As curvas representam a média e desvio padrão do LE λ estimado para 200 textos claros $e\mathbf{P}$ durante $t = 200$ iterações, as quais foram normalizadas com respeito do limite superior do LE λ_m obtido com a Equação. (6.3.1). Nota-se que a linha traçada que corta ambos os gráficos é apenas referencial.	95

Figura 6.3.3 - Gráfico do expoente de Lyapunov em função do número de iterações para os diferentes modos de operação: ECB, CBC, OFB, CFB, CTR e PCBC, usando as cifras de blocos da FAMILIA-64: DES, TEA, XTEA e da FAMILIA-128: RC6, Twofish, Seed, Serpent e Camellia. As curvas representam a média e desvio padrão do LE λ estimado para 200 textos claros ${}_e\mathbf{P}$ durante $t = 200$ iterações, as quais foram normalizadas com respeito do limite superior do LE λ_m obtido com a Equação. (6.3.1).	96
Figura 6.3.4 - Tempo de execução em segundos para estimar o LE dos modos de operação com diferentes cifras de bloco da FAMILIA-64 (acima) e FAMILIA-128 (abaixo). O LE λ foi estimado para 200 textos claros ${}_e\mathbf{P}$ durante $t = 200$ iterações.	98
Figura 6.3.5 - Comparação do LE para diferentes modos de operação, usando diferentes números de blocos $b = 2, 4, 8, \dots, 20$ com a cifra de bloco IDEA. Cada gráfico mostra as curvas do λ durante $t = 200$ iterações.	99
Figura A.1.1 -Exemplo de modo de operação CBC, iterando no tempo com $b = 2$ blocos, $n = 3$ bits e cifra de bloco E_k	116
Figura A.3.1 -Ilustração de uma árvore de defeitos, para indicar o número de defeitos em cada iteração.	117
Figura A.3.2 -Ilustração do cálculo do LE para um modo de operação genérico com uma cifra de bloco de $b = 2$ blocos e $n = 3$ bits.	118

Lista de Tabelas

Tabela 3.2.1 - Comparação entre as propriedades dos sistemas caóticos e as propriedades da Criptografia (25).	58
Tabela 4.1.1 - Algoritmo para calcular o expoente de Lyapunov dos ACs proposto em (31).	64
Tabela 4.3.1 - Análise comparativa de regras de autômatos celulares da família <i>life-like</i> , baseada nas medidas de entropia, expoente de Lyapunov e distância de Hamming (D_H). As regras estão em ordem decrescente (da esquerda para direita) de acordo com seus valores de Max (56).	67
Tabela 4.3.2 - (Continuação) Análise comparativa de regras de autômatos celulares da família <i>life-like</i> , baseada nas medidas de entropia, expoente de Lyapunov e distância de Hamming (D_H) (56).	68
Tabela 5.1.1 - Resultados teste ENT (média e desvio padrão) aplicados ao PRNG proposto para diferentes números de blocos ρ (56).	78
Tabela 5.1.2 - Comparação dos resultados obtidos na proposta com outros algoritmos na literatura (56).	78
Tabela 5.1.3 - Comparação dos resultados dos testes DIEHARD com $p = 90\%$. .	79
Tabela 5.2.1 - Análise estatística do número de diferenças E_e para diferente número de blocos.	85
Tabela 5.2.2 - Comparação dos testes de segurança entre o algoritmo <i>hash</i> proposto e Whirlpool.	85
Tabela 6.2.1 - Algoritmo do expoente de Lyapunov do modo de operação	92
Tabela 6.3.1 - Especificações das duas famílias de cifras de blocos consideradas. .	93
Tabela 6.3.2 - MLE λ_{200} e desvio padrão ($\times 10^{-3}$) dos modos de operação das FAMILIA-64 e 128.	94

Lista de abreviaturas e siglas

AC	Autômato celular
ACE	Autômato celular elemental
RC	Redes complexas
LE	<i>Lyapunov exponent</i> (Expoente de Lyapunov)
NIST	<i>National Institute of Standards and Technology</i>
ISO	<i>International Organization for Standardization</i>
ECB	<i>Electronic codebook</i>
CBC	<i>Cipher block chaining</i>
OFB	<i>Output feedback</i>
CFB	<i>Cipher feedback</i>
CTR	<i>Counter mode</i>
PCBC	<i>Propagating cipher-block chaining</i>
IDEA	<i>International data encryption algorithm</i>
DES	<i>Data encryption Standard</i>
TEA	<i>Tiny encryption algorithm</i>
XTEA	<i>Extended tiny encryption algorithm</i>
AES	<i>Advanced encryption Standard</i>
RC6	<i>Rivest cipher 6</i>
MD5	<i>Message digest algorithm 5</i>
SHA-1	<i>Secure hash algorithm 1</i>
PRNG	Gerador de números pseudo-aleatórios
LCG	Gerador linear congruencial
ENT	Conjunto de testes de pseudoaleatoriedade ENT
DIEHARD	Conjunto de testes de pseudoaleatoriedade DIEHARD
ASCII	<i>American Standard Code for Information Interchange</i>

Lista de Símbolos

c_i	i -ésima célula do autômato celular
s_0	Função que atribui a configuração inicial do autômato celular
$\mathbf{N}(c_i)$	Função de vizinhança de tamanho $ \mathbf{N}(c_i) $
$s(c_i, t)$	Estado da i -ésima célula no tempo t
$G(V, E)$	Grafo: conjunto de vértices V conectados pelas arestas do conjunto E
$\langle \bar{k}_i \rangle$	Grau do vértice i
$\langle \bar{k} \rangle$	Grau médio
A	Matriz de arestas
$\mathbf{P}$	Texto claro de comprimento N bits
$\mathbf{C}$	Texto cifrado de comprimento N bits
K	Chave ou senha para encriptar de comprimento k
E_K	Função de encriptação
E_K^{-1}	Função de decifração
b	Número de blocos de texto claro
n	Comprimento de bits de cada bloco de texto claro
γ	Vetor de inicialização
M_{E_K}	Modo de operação com cifra de bloco E_K
$\mathbf{C}_j^t$	j -ésimo bloco de texto cifrado no tempo t
$s(c_i^j, 0)$	Estado do i -ésimo bit do j -ésimo bloco do texto claro
$\mathcal{M}$	Mensagem de comprimento N
$H(\mathcal{M})$	<i>Hashcode</i> de comprimento h
λ	Expoente de Lyapunov médio
$\lambda(t)$	Máximo expoente de Lyapunov
λ_m	Limite superior do expoente de Lyapunov
H	Entropia
D_H	Distância de Hamming
$h(c_i^j, t)$	vetor de defeitos
ϵ	Número de bits diferentes (defeitos)

Sumário

1	Introdução	27
1.1	Justificativa e motivação	28
1.2	Objetivos	29
1.3	Organização	30
2	Teoria do caos e autômatos celulares	31
2.1	Introdução à teoria do caos	31
2.2	Autômato celular	33
2.3	Definição matemática	33
2.3.1	Vizinhança	34
2.3.2	Regras de transição	35
2.4	Autômato celular unidimensional	36
2.4.1	Classes de comportamento de Wolfram	36
2.5	Autômato celular bidimensional	38
2.5.1	Nomenclatura da família de AC <i>life-like</i>	40
2.6	Autômatos celulares caóticos	40
2.7	AC irregular como rede complexa	42
2.7.1	Modelo híbrido de redes complexas e ACs	44
2.8	Considerações finais	44
3	Criptografia e Criptoanálise	47
3.1	Criptografia	47
3.1.1	Cifras de bloco	49

3.1.2	Modos de operação	49
3.1.3	Algoritmos de <i>hash</i> criptográficos	55
3.1.4	Gerador de números pseudo-aleatórios	57
3.2	Criptografia baseada em caos	57
3.2.1	Criptografia baseada em AC	59
3.3	Criptoanálise	59
3.3.1	Criptoanálise, um novo enfoque	60
3.4	Considerações finais	61
4	Medidas e análise dos ACs caóticos	63
4.1	Expoente de Lyapunov λ nos ACs	63
4.2	Medidas da teoria da informação	65
4.2.1	Entropia	65
4.2.2	Distância de Hamming	65
4.2.3	Combinação de medidas	65
4.3	Seleção do AC caótico	66
4.3.1	Análise da dinâmica do AC <i>life-like</i> B1357/S2468	69
4.4	Considerações finais	70
5	Proposta de algoritmos de Criptografia caótica	71
5.1	Algoritmo de cifra de bloco baseado em autômato celular caótico	71
5.1.1	Transformação da chave usando o mapa logístico	73
5.1.2	Gerador de números pseudo-aleatórios baseado no AC	73
5.1.3	Encriptação e decriptação	74
5.1.4	Resultados	74
5.1.5	Comparação com outros algoritmos	76
5.1.6	Considerações gerais do algoritmo de cifra de bloco proposto	79
5.2	Algoritmo de <i>hash</i> caótico proposto	80

5.2.1	Matriz de arestas A	80
5.2.2	O algoritmo de <i>Hash</i>	81
5.2.3	Resultados	83
5.2.4	Comparação com outros algoritmos <i>hash</i>	85
5.2.5	Considerações gerais do algoritmo de <i>hash</i> proposto	86
5.3	Considerações finais	87
6	Proposta de método de Criptoanálise	89
6.1	Abordagem de autômato celular nos sistemas criptográficos	90
6.2	Expoente de Lyapunov nos sistemas criptográficos	90
6.3	Resultados	91
6.3.1	Expoente de Lyapunov para os modos de operação	93
6.3.2	Influência do número de blocos	98
6.3.3	Análises dos resultados	99
6.4	Considerações finais	100
7	Conclusões	101
7.1	Questões para trabalhos futuros	102
7.2	Publicações	103
	Referências	105
A	Apêndice A: Complemento do Capítulo 6	115
A.1	Exemplificação de um modo de operação como um sistema dinâmico	115
A.2	Limite superior do expoente de Lyapunov	116
A.3	Otimização do algoritmo proposto	117
B	Apêndice B: Resumo das publicações	119

1 *Introdução*

Nas últimas décadas, cientistas de diversas áreas da ciência estudaram o comportamento complexo de certos fenômenos da natureza, como a formação das nuvens, das tempestades, dos furacões, das turbulências do mar, dos terremotos, das variações populacionais, da dinâmica das sociedades, do mercado financeiro, entre outros. Estes estudos teceram uma nova área da ciência: a teoria do caos. A teoria do caos estuda fenômenos caracterizados por uma dinâmica instável e imprevisível, que pode ter seu comportamento consideravelmente afetado por uma perturbação mínima no sistema (1).

A teoria do caos está intimamente relacionada com os sistemas complexos: sistemas dinâmicos, fractais, autômatos celulares, entre outros. Os autômatos celulares (ACs) são sistemas dinâmicos discretos em tempo e espaço que permitem, em muitos casos, reduzir os fenômenos naturais a partir de regras simples, chegando a produzir comportamentos complexos e incluso caóticos. Um exemplo de AC pode ser observado na Figura 1.0.1, a qual lembra o padrão de pigmentação encontrada nas conchas de moluscos.

Figura 1.0.1 – Na esquerda, um AC elementar usando a regra 30 do Wolfram, a partir de um estado inicial “desordenado”, em que cada célula tem valor 0 ou 1. Na direita, o molusco *Conus textile* com um padrão de pigmentação que lembra o padrão gerado pelo AC. Adaptado de (4,88).

Inicialmente, os ACs estudavam os sistemas de crescimento e auto-reprodução na procura de modelagens matemáticas que pudessem sintetizá-lo (2,5,92). Porém, os ACs também têm contribuído na modelagem de sistemas biológicos (3–5) e químicos (7), mo-

delagem de fluxo de tráfego (8), do mercado financeiro (9), no processamento de imagens (10), no reconhecimento de padrões (11), na Criptografia (12,15), entre outros.

A Criptologia é a disciplina científica que abrange o estudo da Criptografia e a Criptoanálise, ambas áreas antagonistas e coexistentes. Enquanto a Criptografia visa manter em segredo e proteger informações sigilosas, evitando a decifração não autorizada, a Criptoanálise tem como objetivo a quebra desta proteção (60). A história da Criptologia está repleta de fatos marcantes desde sua origem no antigo Egito, passando pelas conquistas do Júlio César, e com maior destaque pela equipe de criptoanalistas de Alan Turing (48) durante a II guerra mundial. É notável que os tempos de guerra foram um dos principais gatilhos para o desenvolvimento da Criptologia. Porém, em tempos de paz, com a introdução da matemática do Shannon (76) formou-se a base do que hoje é a Criptografia moderna.

A propagação e fluxo contínuo de informação são de extrema importância para o desenvolvimento de economias estáveis em todo o mundo, já que estes são um pré-requisito para as transações financeiras, comunicação, etc. (60). O principal objetivo da Criptografia é de “ocultar” essas informações de tal modo que possam ser facilmente transferidas entre o emissor e o receptor, sem permitir que outras pessoas possam compreender as informações que estão presentes na mensagem (13). Por outro lado, os criptoanalistas trabalham continuamente para quebrar as cifras e/ou explorar as informações das mensagens encriptadas (13).

1.1 Justificativa e motivação

Hoje em dia, todos os níveis de comunicação digital, desde do uso pessoal, empresas, corporações, até mesmo governos, necessitam trocar informações por meio de redes locais ou públicas. Muitas destas informações são sigilosas e podem conter dados confidenciais como por exemplo dados de sistemas financeiros, estratégicos, bancários, senhas de contas, entre outros. A única forma de garantir o sigilo destas informações em veículos abertos como a Internet, é o uso de algoritmos criptográficos.

A Criptografia é o estudo do conjunto de princípios e técnicas pelas quais a informação pode ser transformada por meio da encriptação/decriptação para torná-la incompreensível por alguém não autorizado. O principal objetivo da Criptografia é de “ocultar” a informação, visando uma transmissão segura, assim como a proteção da informação, sem permitir a compreensão do conteúdo nestas mensagens (13). A Criptoanálise é a disciplina

oposta à Criptografia que procura encontrar fraquezas ou vulnerabilidades nos sistemas de criptografia, afim de quebrá-las e conseguir informações secretas.

A necessidade por algoritmos criptográficos mais seguros tem crescido em razão da necessidade da transmissão massiva de informação. Diversas organizações tem promovido a continuidade no desenvolvimento de novos algoritmos criptográficos para torná-los mais seguros, e novas tendências para abordar estes desafios tem surgido. Entretanto, a Criptoanálise tem progredido paralelamente, procurando vulnerabilidades e tentando “quebrar” os algoritmos criptográficos para expor o conteúdo das mensagens encriptadas.

Em 1985, Wolfram (12) introduziu o primeiro arcabouço de criptografia baseado em AC caótico que atualmente tem recebido crescente interesse (14–22). Muitas das propriedades do caos estão relacionadas com as propriedades criptográficas, as quais são apontadas como requisitos para garantir uma encriptação adequada (25, 29). Além da Criptografia, os ACs também poderiam ser aplicados à Criptoanálise, devido a suas características que permitem explorar e analisar o comportamento de padrões (94, 96).

1.2 Objetivos

Neste trabalho é proposta uma abordagem que combina a Criptografia convencional com modelos matemáticos que apresentam comportamentos caóticos, cujas propriedades são úteis à Criptografia e à Criptoanálise. Este trabalho visa desenvolver e testar três aplicações baseados nos autômatos celulares caóticos. Assim, o trabalho está dividido em duas partes. Na primeira parte, são desenvolvidos dois algoritmos de criptografia: um algoritmo de cifra de bloco e um algoritmo de *hash* capazes de oferecer qualidade melhor que ou semelhante a outros algoritmos da literatura, avaliados por meio de diferentes testes de aleatoriedade, segurança e velocidade.

Na segunda parte, é abordada a Criptoanálise como um problema de detecção de padrões ou “assinaturas” em um dos componentes de um sistema criptográfico. Esta abordagem é pouco explorada na literatura da Criptoanálise devido à dificuldade inerente das características de um sistema de criptografia que, “idealmente” deveriam ser isentos de padrões. Recentemente, foi proposta uma abordagem promissora (24, 26), na qual é utilizado o expoente de Lyapunov—usada na teoria do caos para medir a estabilidade de um sistema—para explorar algoritmos de criptografia convencional.

Neste trabalho, propomos um novo método de criptoanálise baseada numa adaptação

do expoente de Lyapunov dos ACs (31), visando estudar o comportamento dos modos de operação criptográficos tratando-os como autômatos celulares. Esta proposta é justificada por três motivos: (i) pela importância dos modos de operação durante a encriptação de dados para garantir a confidencialidade e segurança dos dados (46); (ii) pela escassez de métodos na literatura para discriminar e/ou classificar os diferentes modos de operação; (iii) pelo interesse de abordar a criptoanálise e o reconhecimento de padrões como estudos análogos.

Em resumo, os objetivos e contribuições desse trabalho são:

- Avaliar critérios de seleção de ACs caóticos apropriados à Criptografia;
- Desenvolver um gerador de números pseudo-aleatórios baseado em um AC caótico;
- Desenvolver um algoritmo de cifra de bloco, valendo-se dos números pseudo-aleatórios gerados;
- Desenvolver um algoritmo de *hash* baseado em AC caótico;
- Propor um método de criptoanálise baseada no expoente de Lyapunov dos ACs;
- Avaliar os algoritmos;
- Comparar os resultados com outros métodos da literatura.

1.3 Organização

Esta dissertação está dividida em sete capítulos, incluindo esta introdução. O Capítulo 2 aborda os conceitos relacionados à teoria do caos e autômatos celulares, já no Capítulo 3 são introduzidos os conceitos básicos de Criptografia e Criptoanálise, incluindo a respectiva revisão bibliográfica. No Capítulo 4 são descritas várias medidas para a seleção de um AC caótico para ser usado na proposta do algoritmo de cifra descrito no Capítulo 5, assim como da proposta do algoritmo *hash* e seus respectivos testes, análises comparativos com outras abordagens, atende discussão sobre os resultados alcançados. No Capítulo 6 é apresentado o método de Criptoanálise baseada no expoente de Lyapunov e resultados de experimentos. No Capítulo 7 as conclusões e as contribuições são discutidas, assim como são sugeridas algumas questões para trabalhos futuros.

2 Teoria do caos e autômatos celulares

2.1 Introdução à teoria do caos

A teoria do caos nasceu em 1963, quando o físico e meteorologista Edward Lorenz fez uma importante descoberta, enquanto estudava um modelo que pudesse prever o clima. Lorenz descobriu um comportamento imprevisível no modelo matemático, quando introduziu pequenas variações (erros de arredondamento) nas **condições iniciais**, cujos resultados seguiram trajetórias muito divergentes, tornando virtualmente impossível a previsão do clima por modelagem matemática.

O **caos determinístico**, ou caos simplesmente, é o estudo dos sistemas dinâmicos não lineares cuja característica mais importante é sua **sensibilidade** às condições iniciais (1). Em particular, o comportamento caótico apresenta características como: ergodicidade^a, dinâmica determinística, e sensibilidade às condições iniciais. O interesse na teoria do caos é que mesmo as equações simples podem gerar comportamentos complexos. Por exemplo, o mapa logístico (Equação (2.1.1)) é um dos sistemas dinâmicos mais conhecidos, usado para descrever o crescimento das populações de insetos. Observa-se que sua equação é muito simples, porém certos valores da constante μ podem gerar caos.

$$X_{h+1} = \mu X_h(1 - X_h). \quad (2.1.1)$$

Para medir a sensibilidade às condições iniciais em sistemas dinâmicos contínuos, discretos e séries numéricas é usado o **expoente de Lyapunov** (LE, pelas siglas em inglês) representado por λ (47,87) (ver Capítulo 4). O cálculo do LE pode distinguir três tipos de comportamentos nos sistemas dinâmicos, quando iterados ao longo do tempo

^a A ergodicidade implica a estatística da distribuição do sistema ao longo do tempo. Sendo que o estado seguinte X_{h+1} pode ser calculado em termos de probabilidades.

(convergência ao atrator^b) (47):

$\lambda = 0$	Periódico (atrator periódico)
$\lambda > 0$	Caótico (atrator estranho)
$\lambda < 0$	Estável (atrator fixo)

Estes comportamentos são ilustrados na Figura 2.1.1, cada subfigura amostra a evolução do mapa logístico para duas condições iniciais X_0 e X'_0 (muito próximas), para diferentes valores de μ , que quando iterado ao longo do tempo geram diferentes comportamentos (a) periódico, (b) caótico e (c) estável. Havendo especial interesse pelo regime caótico ilustrado em (b). Nota-se que inicialmente ambas trajetórias seguem o mesmo padrão, porém a partir de certo tempo ambas divergem drasticamente, em razão de uma taxa exponencial (47).

Figura 2.1.1 – Evolução temporal de duas trajetórias da Equação (2.1.1) com condições iniciais $X_0 = 0.9$ e $X'_0 = 0.90001$, para os parâmetros (a) $\mu = 3.57$, Periódico; (b) $\mu = 3.9$, Caótico e (c) $\mu = 2.85$, Estável.

^b Um atrator é uma região do espaço de fase que atrai os pontos de regiões vizinhas ao longo do tempo (62)

2.2 Autômato celular

A literatura define o autômato celular (AC) como um sistema dinâmico discreto no espaço e o tempo, constituído por componentes individuais (**células**) que possuem estados que se alteram no tempo (**evolução** ou **iteração**), de acordo com um conjunto de **regras de transição**, e baseando-se no estado da célula e/ou sua **vizinhança**.

Parte do fascínio pelos ACs é que estes permitem, em muitos casos, reduzir os fenômenos naturais a partir de regras simples, a qual chega produzir comportamentos complexos. A pesquisa inicial do John Von Neumann (2) em 1947, para desenvolver um modelo abstrato de crescimento e auto-reprodução em Biologia, marcou os primeiros traços dos ACs. Já em 1951 Neumann, aconselhado por Stanislaw Ulam, introduziu formalmente o conceito de autômatos celulares formalmente (2).

Em 1970 o matemático John Conway contribuiu significativamente na área, modelando o famoso “Jogo da vida” (ou *Game of life*) a partir de uma malha bidimensional de células com estados vivos ou mortos e um conjunto de regras simples, na qual estruturas complexas podem emergir (5).

No ano de 1981 o cientista Stephen Wolfram começou um estudo massivo sobre o comportamento dos ACs (89,92), que é sem dúvida um dos trabalhos mais influentes na área. Dentre as contribuições do Wolfram encontra-se o sistema de nomenclatura para uma classe especial de AC unidimensional conhecidos como AC elementares (ACE), outra importante contribuição é o esquema de classificação que dividiu regras dos ACs em quatro classes qualitativas que, de acordo com os padrões gerados por todos os ACE, evoluem a partir de uma **configuração** inicial “desordenada”.

2.3 Definição matemática

De acordo com uma generalização matemática de AC de Baetens e De Baets (31), um autômato celular $\mathcal{C}$ pode ser representado como um sêxtuplo $\langle \mathcal{T}, S, s, s_0, \mathbf{N}, \phi \rangle$ onde:

1. $\mathcal{T}$ é uma tesselação n-dimensional no espaço euclidiano $\mathbb{R}^n$, composto de células c_i , $i \in \mathbb{N}$;
2. S é um conjunto finito de z estados, $S \subset \mathbb{N}$;

3. A função de saída $s : \mathcal{T} \times \mathbb{N} \rightarrow S$ mapeia os estados da célula c_i no tempo discreto t , ou seja, $s(c_i, t)$;
4. A função $s_0 : \mathcal{T} \rightarrow S$ atribui a configuração inicial de cada célula c_i , assim $s(c_i, 0) = s_0(c_i)$.
5. A função de vizinhança $N : \mathcal{T} \rightarrow \bigcup_{p=1}^{\infty} \mathcal{T}^p$, retorna a cada célula c_i uma sequência finita $\mathbf{N}(c_i) = (c_{i_j})_{j=1}^{|\mathbf{N}(c_i)|}$ composto de $|\mathbf{N}(c_i)|$ células c_{i_j} distintas;
6. A função de transição $\phi : S^{|\mathbf{N}(c_i)|} \rightarrow S$, define:

$$s(c_i, t + 1) = \phi \left((s(c_i, t))_{j=1}^{|\mathbf{N}(c_i)|} \right) \quad (2.3.1)$$

Nas seções seguintes são detalhados cada um dos itens mencionados durante a definição anterior.

2.3.1 Vizinhança

Normalmente os ACs empregam estruturas **regulares** de vizinhança, esses ACs têm dimensão espacial (1D, 2D, 3D) e, portanto, apresentam formatos (linear, triangular, quadrangular, hexagonal, etc.) que em conjunto são chamadas da geometria do AC (50). Mas na literatura encontra-se também os ACs **irregulares** (31, 69), quando o número de vizinhos varia, as quais geram estruturas abstratas.

Cada célula c_i da vizinhança está conectada com seus $|\mathbf{N}(c_i)|$ vizinhos que cobrem um raio r de ação da regra (53). A Figura 2.3.1 apresenta exemplos de diferentes configurações para 1D- e 2D-AC com raios diferentes e tipos de vizinhanças de Neumann e Moore.

2.3.1.1 Condições de fronteira

Por definição, um AC consiste de uma tesselação infinita, porém computacionalmente os ACs são simulados numa malha finita ao invés da infinita, logo, certas condições nas fronteiras devem ser consideradas para completar o conjunto de vizinhos das células que se encontram nas bordas do AC. Existem três tipos de fronteiras: periódica, reflexiva e fixa (85).

A fronteira periódica procura conectar as células dos extremos e dar continuidade, por exemplo no caso de um 1D-AC será considerado um espaço circular, no caso de um

Figura 2.3.1 – Exemplos de 1D- e 2D-AC com diferentes tipos de vizinhança. Em (a) e (b) ACE com célula central em preto e a vizinhança em cinza de raio $r = 1$ e $r = 2$ respectivamente, da mesma maneira, em (c) e (d) vizinhança de Von Neumann, e em (e) e (f) vizinhança de Moore. Adaptado de (71).

2D-AC um toróide. A fronteira reflexiva é a condição em que cada célula dos extremos é auto-refletida. A fronteira fixa define desde o começo os estados de cada extremo mantendo-os fixos durante sua evolução. Estas três condições de fronteira também podem ser combinadas. Weimar (85) acrescenta uma outra solução ao problema, a qual consiste em estender o espaço do AC a cada passo de tempo, dando a impressão de um AC ilimitado, porém desvantajoso computacionalmente.

2.3.2 Regras de transição

Tipicamente a regra de transição ϕ descreve as regras que regem a dinâmica de evolução de cada célula c_i no tempo $t + 1$ dependendo da configuração do AC no tempo t . Na literatura se encontram diferentes propriedades de uma regra de transição, ou função de transição, dentre essas:

- A regra de transição pode ser **determinística** ou **não determinística**. É determinística quando dada uma configuração no tempo t o estado seguinte é sempre o mesmo no tempo $t + 1$. Por outro lado, é não determinística quando define o estado

seguinte com base numa função de probabilidade.

- O AC se diz homogêneo, ou uniforme, quando as regras são em comum para todas as células e não mudam ao longo do tempo; senão híbrido, ou não uniforme.
- Como descrito na Equação (2.3.1), ϕ atualiza os estados de todas as células do AC de forma simultânea e sincronizada (89), o qual significa que os novos estados $s(c_i, t + 1)$ não substituem os antigos $s(c_i, t)$ até que todos sejam calculados.
- Quando a regra de transição depende apenas do número total de células vivas na vizinhança chama-se de AC **totalista** (91), como por exemplo o ACE. Por outro lado, quando a regra de transição depende tanto do estado atual da célula como da sua vizinhança, chama-se de AC **não-totalista**; um exemplo deste AC é o “Jogo da vida” do Conway.

2.4 Autômato celular unidimensional

O Autômato celular elementar (ACE) proposto por Wolfram (88, 91), é um caso especial de um AC unidimensional, já que está composto por uma linha de células que ao evoluir adiciona os novos estados embaixo dos anteriores, ao qual Wolfram chamou de diagrama de espaço-tempo. Formalmente, o ACE é determinístico, totalista, de raio $r = 1$ e com $k = 2$ estados, ou seja $\langle \mathcal{Z}^1, S = \{0, 1\}, s, s_0, \mathbf{N} = (c_{i-r}, c_i, c_{i+r}), \phi : S^3 \rightarrow S \rangle$. O AC elementar é chamado assim porque a configuração inicial contém apenas uma célula ativa posicionada no centro, porém Wolfram também estudou os ACE com configurações iniciais aleatórias. Wolfram introduziu uma nomenclatura padrão para as 256 regras do ACE, baseados nas $k^{|\mathbf{N}(c_i)|} = 2^3 = 8$ possíveis configurações que uma célula pode evoluir, assim cada regra é especificada por uma sequência de oito bits que formam o nome de regra. Um exemplo de evolução de um ACE de regra 30 é dado na Figura 2.4.1, observa-se que a vizinhança da célula c_i é composta pelas células da esquerda c_{i-r} e da direita c_{i+r} no instante t e evolui à $t + 1$.

2.4.1 Classes de comportamento de Wolfram

Wolfram (89) propôs quatro classes qualitativas de padrões de comportamento de ACs com base na observação dos padrões gerados na evolução a partir de uma configuração

Figura 2.4.1 – Exemplo de ACE de raio $r = 1$ e com $k = 2$ estados cinza (1) e branco (0), evoluindo com a regra 30 do Wolfram.

inicial aleatória das 255 regras do ACE. Ele defende que qualquer AC corresponde a uma destas quatro classes básicas:

- Classe I (Comportamento fixo): Quase todas as configurações iniciais do AC evoluem (convergem) rapidamente para um estado estável e homogêneo. Exemplos: regras 0, 16, 48, 54, 60, 172 (ilustrada na Figura 2.4.2), etc.
- Classe II (Comportamento periódico): A evolução dos ACs cria estruturas periódicas com estados estáveis. Exemplos: regras 7 (ilustrada na Figura 2.4.2), 8, 24, 40, 56, 58, etc.
- Classe III (Comportamento caótico): Exibem comportamentos caóticos, caracterizados pela forte dependência nas condições iniciais. Exemplos: regras 2, 6, 14, 18, 22, 30 (ilustrada na Figura 2.4.2), etc.
- Classe IV (Comportamento complexo): ACs que formam estruturas localizadas de estados repetitivos ou estáveis capazes de sobreviver durante longos períodos de tempo, mas também formam estruturas que interagem uns com os outros de forma complexa (propagar, criar e/ou destruir). Exemplos: regras 20, 52, 110 (ilustrada na Figura 2.4.2), etc.

A Figura 2.4.2 mostra os exemplos de regras 172, 7, 30 e 110 com configuração inicial aleatória sendo classificados 4 grupos respectivamente.

Wolfram (90) também apresentou algumas analogias entre a classificação dos ACs com a dos sistemas dinâmicos, em que há ponto fixo estável (Classe I, $\lambda < 0$), periódico (Classe II, $\lambda = 0$), e comportamento caótico (Classe III, $\lambda > 0$), porém não existe uma analogia direta com os ACs da Classe IV. Entretanto, a classificação do Wolfram apresenta algumas desvantagens, já que existem traços de subjetividade nesta classificação com respeito às

Figura 2.4.2 – Classificação qualitativa dos ACE do Wolfram. As figuras mostram o diagrama espaço-tempo do ACE de 600 células com uma configuração inicial aleatória durante 400 iterações (91).

Classes III e IV. Ante essa problemática, diversos cientistas abordaram algumas medidas para quantificar, analisar e compreender a dinâmica dos ACs (31, 33), tema que será tratado na Seção 4.1.

2.5 Autômato celular bidimensional

O popular “Jogo da vida” é um dos exemplos de autômato celular bidimensional (2D-AC) mais conhecidos na literatura. O qual foi proposto por John Conway (5) e inicialmente baseado em um modelo biológico. O “Jogo da vida” é um 2D-AC determinístico, *não-totalista*, regular de geometria quadrangular, baseado na vizinhança de Moore com raio $r = 1$ de estados $k = 2$ (vivo ou morto) e com condição de fronteira periódica. Formalmente, é descrito como $\langle \mathbb{Z}^2, S = \{0, 1\}, s, s_0, \mathbf{N} = (c_1, \dots, c_9), \phi : S^9 \rightarrow S \rangle$, cuja regra de transição é da seguinte forma (23):

$$\phi \left(\begin{pmatrix} c_1 & c_2 & c_3 \\ c_4 & c_5 & c_6 \\ c_7 & c_8 & c_9 \end{pmatrix} \right) = \begin{cases} 1, & \text{se } \sum_{i=1}^9 s(c_i, t) = 3 \\ 1, & \text{se } \sum_{i=1}^9 s(c_i, t) = 2, i \neq 5 \\ 0, & \text{caso contrário.} \end{cases} \quad (2.5.1)$$

A regra de transição dada na Equação (2.5.1) é ilustrada na Figura 2.5.1: qualquer célula viva com dois ou três vizinhos vivos, sobrevive (ver célula c_5), e qualquer célula com exatamente três vizinhos vivos, nasce (ver célula c_2), de outra forma morre.

Por outro lado, a Figura 2.5.2 apresenta a simulação do modelo do Conway representado por uma malha de 200×200 , a partir de uma configuração inicial aleatória iterado

Figura 2.5.1 – Exemplo de uso da regra de transição do “Jogo da vida” do Conway, dada na Equação (2.5.1). A cor cinza representa uma célula viva, e a cor branca representa uma célula morta.

em intervalos de tempo.

Figura 2.5.2 – Evolução do “Jogo da vida”. A cor preta representa uma célula viva, e a cor branca representa uma célula morta. As Figuras mostram a evolução do AC simulado numa malha de 200×200 e condição inicial aleatória durante 800 iterações.

2.5.1 Nomenclatura da família de AC *life-like*

A partir da proposta do AC do Conway, muitas outras regras de transição podem ser derivadas, é dizer um “Jogo da vida” generalizado chamado também como família *life-like*. Golly (82) usou uma notação para a nomeação desta família. Uma regra de transição é descrita da forma Bx/Sy , onde x e y é dado como uma sequência de nove dígitos distintos de 0 a 8, em ordem numérica. Assim, o “Jogo da vida” é denotado como B3/S23, onde uma célula “nasce” (B) quando o total de vizinhos vivos é exatamente três, ou “sobrevive” (S) se possui dois ou três vizinhos vivos. A família de AC *life-like* está composta pelas $2^{18} = 262144$ regras, pois existem 18 diferentes configurações que uma célula do 2D-AC *não-totalista* pode formar^c.

Na Figura 2.5.3 são dados vários exemplos de regras da família *life-like*, muitos destes possuem nomes dos padrões que formam ou nomes das pessoas que as estudaram. Cada uma das subfiguras amostra a mesma configuração inicial aleatória contendo aproximadamente 50% de população de células vivas. Nas regras B357/S238, B3678/S34678, B36/S125, pode ser observada a diminuição da população de células vivas quando são iteradas por um tempo considerável ($t = 1000$ iterações), dependendo da regra, este fenômeno é chamado de “tendência à morte” (75). O caso oposto ocorre com as regras B378/S235678, B3/S45678 que parecem ter superpopulação. Também observa-se que há formação de padrões com regras B45678/S2345, B3/S1234. Porém, a regra B1357/S2468 não possui nenhum padrão visível.

2.6 Autômatos celulares caóticos

Como indicado na Seção 2.4.1, Wolfram (90) já tinha classificado o comportamento dos ACs em 4 classes e concluiu que certos ACs possuem comportamento caótico. Nota-se que o significado de caos não é o oposto de ordem, e também não é sinônimo de aleatoriedade, porém suas características os assemelham.

A Figura 2.6.1 mostra três ACs da família “*life-like*”, a coluna da esquerda mostra a configuração inicial numa malha de 16×16 . A coluna do centro e da direita mostra o mesmo AC porém com uma célula perturbada (uma viva e a outra morta) nas posições $c(8, 8)$ e $c(10, 2)$. A sensibilidade às condições iniciais da regra B1357/S2468 torna-se evi-

^c Nota-se que o número de regras de um 2D-AC *totalista* é de $k^{k^{|\mathbf{N}(c_i)|}} = 2^{2^9} = 2^{512} \approx 10^{152}$.

Figura 2.5.3 – Evolução de diferentes regras da família “*life-like*”. A cor preta representa uma célula viva, e a cor branca representa uma célula morta. As Figuras mostram a evolução do AC simulado numa malha de 128×128 e condição inicial aleatória após 1000 iterações.

dente ao compararmos as diferenças entre as configurações dos três ACs a cada iteração. Visualmente parece não existir nenhum padrão; sendo sensível às condições iniciais, é portanto caótica. A caoticidade desta regra será analisada quantitativamente no Capítulo 4.

Figura 2.6.1 – Evolução de três ACs usando a regra B1357/S2468. Na esquerda o AC com configuração inicial aleatória e, no centro e na direita a mesma configuração porém com uma célula perturbada. A cor preta representa uma célula viva, e a cor branca representa uma célula morta. As figuras mostram a evolução do AC simulado numa malha de 16×16 para $t = 100$ e $t = 101$ iterações.

2.7 AC irregular como rede complexa

Nesta seção são discutidos brevemente os ACs irregulares (69). Os ACs irregulares, cujo espaço é uma tesselação irregular, são uma generalização dos AC regulares cuja vizinhança e geometria são conhecidas. Muitos cientistas apresentaram os grafos como uma estrutura espacial e dinâmica para modelar qualquer AC (34,53). Um exemplo dessa modelagem pode ser observado na Figura 2.7.1.

A rede complexa é representada por um grafo $G(V, E)$, um conjunto de vértices $V = \{v_1, v_2, \dots, v_Q\}$ interligados por um conjunto de arestas $E = \{(v_i, v_j) | (v_i, v_j) \in V \times V\}$ (40). Uma rede complexa pode ser implementada a partir de uma matriz de adjacência. Assim um AC pode ser modelado da seguinte forma: Os vértices v_i representam as células c_i , cujas arestas representam a vizinhança $E = N(c_i)$, cada elemento na matriz armazena o estado das células $S = \{0, 1\}$. A rede complexa pode ser percorrida em base ao conjunto

Figura 2.7.1 – (a) Exemplo de tesselação irregular. (b) Rede complexa: grafo não orientado modelando o exemplo de tesselação dado em (a).

de regras de transição (caminhadas), a fim de modelar a dinâmica do AC. O tamanho da vizinhança $|\mathbf{N}(c_i)|$ é representado pelo grau $\langle \bar{k}_i \rangle$ que corresponde ao número de vértices que possua, o grau meio é dado pela Equação (2.7.1).

$$\langle \bar{k} \rangle = \frac{1}{Q} \sum_{i=1}^{|\mathbf{N}(c_i)|} \langle \bar{k}_i \rangle. \quad (2.7.1)$$

A Figura 2.7.2 mostra um exemplo de um AC irregular modelado como uma rede complexa, pode se observar que cada célula tem diferente número de vizinhos que são adjacentes uns aos outros, formando um espaço toroidal.

Figura 2.7.2 – Modelamento de AC empregando uma rede complexa como tesselação do AC (34).

2.7.1 Modelo híbrido de redes complexas e ACs

Na seção anterior foi apresentado uma modelagem de uma rede complexa como tesselação do AC, nesse sentido as células do AC representam os vértices, a vizinhança representa as arestas e a sua dinâmica é determinada pela regra do AC. Porém, o espaço do AC é uma tesselação em que uma célula é vizinha de um conjunto de células que são adjacentes no espaço. Nesta seção, sugere-se uma modelagem um pouco diferente que à anterior, a qual nomeamos como AC “híbrido”, no sentido que a vizinhança não é, necessariamente, adjacente no espaço—lembrando um toroide—e sim uma vizinhança por conexão de vértices.

Estas diferenças são mais evidentes na Figura 2.7.3, a qual mostra um exemplo de 2D-AC híbrido com regra B1357/S2468 evoluindo no tempo a partir de uma condição inicial aleatória. Esta rede complexa lembra mais a uma rede aleatória porém com regras de evolução dos ACs.

Figura 2.7.3 – Exemplo de evolução do modelo 2D-AC caótico híbrido como uma rede complexa de 128 vértices (células) e $\langle k \rangle = 8$ (vizinhança) com a regra B1357/S2468. A cor cinza representa uma célula viva, e a cor azul representa uma célula morta.

2.8 Considerações finais

Neste capítulo foram apresentados os principais aspectos teóricos sobre a teoria do caos, dado enfoque nos sistemas dinâmicos caóticos. O capítulo foi aprofundado nos autômatos celulares dos tipos uni- e bidimensional de AC de Wolfram e Conway, assim como

foi abordada uma modelagem generalizada dos ACs usando redes complexas e sugerido um modelo híbrido entre ambas. Foram apresentadas as quatro classes básicas de comportamento dos ACs, em especial dos ACs caóticos. No Capítulo 3 serão apresentadas as bases teóricas da Criptografia e Criptoanálise e, posteriormente no Capítulo 4 serão abordadas algumas medidas usadas nos ACs que serão aplicadas às propostas de Criptografia e Criptoanálise.

3 *Criptografia e Criptoanálise*

O principal objetivo deste capítulo é de introduzir os conceitos fundamentais da Criptografia, dando ênfase nas primitivas criptográficas: cifras de blocos, modos de operação, algoritmos *hash* e geradores de números aleatórios. Entretanto, também são apresentados os conceitos básicos da Criptoanálise convencional, assim como desde o enfoque do reconhecimento de padrões como uma extensão da Criptoanálise.

3.1 Criptografia

A Figura 3.1.1 mostra a taxonomia das primitivas criptográficas. Em base à chave, a Criptografia pode ser classificada em chave simétrica, chave assimétrica e sem chave. A criptografia de chave simétrica (ou chave privada) precisa que as partes da comunicação (emissor e o receptor) entrem em acordo prévio quanto a uma chave comum. Por outro lado, na chave assimétrica (ou chave pública) cada parte deve utilizar uma chave diferente, uma para o processo de encriptação (encriptar ou criptografar) e outra para o processo de decriptação (decriptar ou descriptografar) (60).

Terminologia:

- **Sistema de criptografia:** Em geral, um sistema de criptografia consiste de um ou mais algoritmos que encriptam/decriptam um texto claro, por meio de uma chave, transformando-o no texto cifrado e vice-versa;
- **Texto claro:** Considerado como uma sequência de N bits, de tal forma que ele pode ser representado como um vetor de bits $\mathbf{P}$ de comprimento N ;
- **Texto cifrado:** Representado por um vetor de bits $\mathbf{C}$ de comprimento N ;
- **Chave ou senha:** A chave K é representada por uma sequência de k bits.

Figura 3.1.1 – Taxonomia da Criptografia. Adaptado de (60).

- **Função de encriptação:** Recebe como entrada uma chave K e um texto claro $\mathbf{P}$, produzindo um texto cifrado $\mathbf{C}$, denotada por

$$\mathbf{C} = E_K(\mathbf{P}). \quad (3.1.1)$$

Portanto E é função de $\{0, 1\}^k \times \{0, 1\}^N$ para $\{0, 1\}^N$.

- **Função de deciptação:** Recebe como entrada uma chave K e um texto cifrado $\mathbf{C}$, e produz a mensagem original $\mathbf{P}$.

$$\mathbf{P} = E_K^{-1}(\mathbf{C}). \quad (3.1.2)$$

Portanto E^{-1} é função de $\{0, 1\}^k \times \{0, 1\}^N$ para $\{0, 1\}^N$.

- **Operador xor $\oplus$:** Operador lógico **XOR**, ou disjunção exclusiva. O uso deste operador abeliano^a na Criptografia foi proposto por Vernam (83), assim pode encriptar $\mathbf{P} \oplus K = \mathbf{C}$ e deciptar $K \oplus \mathbf{C} = \mathbf{P}$.

Shannon (77) ressalta duas características que qualquer sistema de criptografia, bem projetado, deve possuir: **confusão** e **difusão**. A primeira propriedade de confusão de um sistema de criptografia assegura que o texto cifrado é estatisticamente independente do texto claro e da chave. A segunda propriedade de difusão indica que uma pequena perturbação no texto claro, ou na chave, provocaria uma grande mudança no texto cifrado.

^a O grupo abeliano refere-se às operações com propriedade comutativa, em que o resultado da aplicação da operação a dois elementos do grupo não depende de sua ordem, *e.g.*, a aritmética de adição e multiplicação.

3.1.1 Cifras de bloco

Uma cifra de bloco divide o texto claro $\mathbf{P}$ em b **blocos** $(P_1, P_2, \dots, P_b)$, em que cada bloco P_i possui n bits (geralmente de 64 ou 128 bits) e, em seguida, cada bloco passa por um mesmo processo de encriptação/decriptação produzindo o correspondente bloco de texto cifrado e vice-versa. Nota-se que, como cada bloco possui n bits, o tamanho do texto claro N pode variar e, muitas vezes, ser diferente do comprimento dos blocos para o tamanho fixo que foi concebido. Consequentemente, se o comprimento do texto claro N não é um múltiplo inteiro de b , os bits faltantes do último bloco do texto claro P_b são preenchidos por algum esquema de preenchimento até completar um múltiplo de N .

As cifras de blocos ganharam grande popularidade desde a introdução do DES (*Data Encryption Standard*) (65), em meados dos anos 1970, embora o DES já foi propenso a ataques de força bruta^b. Para superar essa deficiência, Massey e Lai criaram o IDEA (*International Data Encryption Algorithm*) (30) que foi projetado em 1991 para substituir ao DES. Após esses acontecimentos, incrementou-se o desenvolvimento de novos algoritmos de criptografia que atingissem melhores expectativas de segurança. Em 1997, o NIST (*National Institute of Standards and Technology*) (67) iniciou um processo seletivo para anunciar ao futuro AES (*Advanced Encryption Standard*) para ser adotado internacionalmente, dentre muitos concorrentes: Serpent (27), Twofish (74), RC6 (6), Rijndael (66), entre outros, sendo este último o atual vencedor.

3.1.2 Modos de operação

Em geral, uma cifra de bloco permite a encriptação de um bloco por vez, mas quando blocos consecutivos são encriptados usando a mesma chave, os riscos de segurança aumentam (79). O NIST recomenda que toda cifra de bloco utilize um **modo de operação** (ou modo) em conjunto, para garantir a confidencialidade^c das mensagens (46). O modo de operação especifica os mecanismos para “melhorar” os processos de encriptação/decriptação, enquanto a cifra do bloco encripta/decripta todos os blocos (um de cada vez) conforme vai operando.

No ano 2001, o NIST (46) recompilou cinco modos de operação para garantir a

^b A força bruta é um ataque criptoanalítico usado para quebrar uma cifra. O ataque consiste em verificar sistematicamente todas as chaves possíveis até que a chave correta seja encontrada. No pior dos casos, isso implicaria percorrer o espaço de busca inteiro.

^c O princípio de confidencialidade estabelece que só o destinatário autorizado deve ser capaz de extrair o conteúdo da mensagem da sua forma cifrada.

confidencialidade— a maioria destes foram desenvolvidos há 30 anos— para serem usados conjuntamente com uma cifra de bloco. Esses modos de operação são: o *Electronic Codebook* (ECB), *Cipher Block Chaining* (CBC), *Cipher Feedback* (CFB), *Output Feedback* (OFB), e *Counter* (CTR) (46). Além desses modos do NIST, existem muitos outros, entre eles o *Propagating Cipher-Block Chaining* (PCBC) (60).

3.1.2.1 *Electronic codebook* ECB

O ECB (*Electronic Codebook*) é um modo de operação muito simples. O texto claro $\mathbf{P}$ é dividido em blocos $P_1, P_2, \dots, P_j$ de n -bits que são encriptados, independente dos outros blocos, sob o mesmo processo de encriptação E . O texto cifrado resultante $\mathbf{C}$ será a concatenação, mantendo a mesma ordem, de cada um dos blocos decriptados C_j previamente (Equação (3.1.3)).

$$C_j = E_K(P_j), \quad j = 1, 2, \dots, b. \quad (3.1.3)$$

A Figura 3.1.2 mostra o esquema do modo de operação ECB para o processo de encriptação e decriptação.

Figura 3.1.2 – Modo de operação ECB (79).

A principal desvantagem deste modo é evidente, dado que a cifra de bloco utiliza uma única chave K para encriptar de cada vez. Ao encriptar blocos do texto claro que fossem idênticos, ou com presença de padrões, implicaria que os textos cifrados sejam idênticos também ou sem conseguir esconder esses padrões de repetição. Um cenário muito comum deste problema é, por exemplo, ao encriptar arquivos de HTML (abreviação da expressão inglesa *HyperText Markup Language*) cujo conteúdo é altamente estruturado e com esses marcadores que se repetem constantemente, tornando da cifra fácil de quebrar e explorar estas regularidades.

O modo de operação ECB é indicado apenas para mensagens de comprimento não maior do que um bloco. Não sendo apropriado para mensagens de comprimento maior, já que este modo não favoreceria à confidencialidade desejada.

3.1.2.2 Cipher block chaining CBC

O modo de operação CBC (*Cipher Block Chaining*), foi criado para lidar com as deficiências mencionadas do ECB, no sentido que, se o texto claro apresentasse padrões no conteúdo, repetindo-se várias vezes, este seria encriptado de maneira diferente do texto cifrado a cada repetição.

O formalismo de modo CBC é dada pela Equação (3.1.5).

$$\begin{aligned} C_j &= E_K(C_{j-1} \oplus P_j), \quad j = 1, 2, \dots, b, \\ C_0 &= \gamma. \end{aligned} \quad (3.1.4)$$

O CBC é um modo que opera sequencialmente, como o nome indica, baseia-se no encadeamento do texto cifrado anterior C_{j-1} mediante uma operação XOR com o seguinte texto claro P_j . Para produzir o primeiro bloco cifrado C_1 , é preciso de um **vetor de inicialização** denotado por γ , cuja função é iniciar o processo de encriptação/decriptação e, deve ser um valor selecionado aleatoriamente no espaço $\{0, 1\}^n$ (60). De acordo com Stallings (79), o vetor de inicialização do modo CBC deve ser conhecido pelo emissor e receptor, mas deve ser imprevisível para terceiros. Assim, deve ser garantida a sua integridade durante a transmissão, ou seja γ deve ser secreto.

Figura 3.1.3 – Modo de operação CBC (79).

O modo CBC é fortemente dependente dos blocos cifrados precedentes, o qual dificultaria sua programação paralela, tanto da sua encriptação como a decriptação. Porém, esta dependência também representa a maior vantagem do CBC, já que uma alteração nos

blocos de texto claros provocariam uma alteração no bloco cifrado e, conseqüentemente em todos os blocos seguintes (efeito dominó).

3.1.2.3 *Output feedback* OFB

O modo OFB (*Output Feedback*) baseia-se no emprego das saídas produzidas pela cifra de bloco (antes do XOR) para realimentar as entradas dos estágios seguintes. A Equação (3.1.5) descreve o processo de encriptação do modo OFB:

$$C_j = P_j \oplus O_{j-1} \quad j = 1, 2, \dots, b, \quad (3.1.5)$$

$$O_j = E_K(O_{j-1}),$$

$$O_0 = E_K(\gamma).$$

Pode ser visto na Figura 3.1.4 que a primeira saída produzida é totalmente determinada pela chave K e pelo vetor de inicialização e não pelo texto claro em si. Sendo que, o primeiro texto cifrado C_1 é produzido após fazer uma operação XOR com o texto claro P_1 . De acordo com Stallings (79), a maioria dos modos de operação usam um vetor de inicialização para dar início ao processo. Porém, o vetor de inicialização empregado no OFB deve ser além de aleatório, único e diferente a cada vez que for executada a cifra de bloco.

O processo de encriptação e decifração do OFB não pode ser paralelizado, já que cada operação de realimentação é dependente do resultado de todos os anteriores.

Figura 3.1.4 – Modo de operação OFB (79).

Deve ser ressaltado que, a função de encriptação E dos modos de operação OFB, CFB

e CTR é usada tanto para a encriptação como para a deciptação.

3.1.2.4 Cipher feedback CFB

O modo CFB (*Cipher Feedback*) é análogo ao modo OFB. Porém, o CFB pode funcionar com “unidades” de blocos de qualquer tamanho de bits (8, 16, 64, *etc.*), enquanto, o OFB funciona com blocos completos de cada vez. A Equação (3.1.7) descreve o processo de encriptação do modo CFB:

$$\begin{aligned} C_j &= P_j \oplus E_K(C_{j-1}), & j = 1, 2, \dots, b, \\ C_1 &= P_1 \oplus E_K(\gamma). \end{aligned} \quad (3.1.6)$$

Da mesma forma que o CBC, o CFB depende do texto cifrado anterior C_{j-1} para realimentar as seguintes etapas. Isto pode ser observado no esquema de encriptação da Figura 3.1.5, onde as unidades de texto cifrado (depois do XOR) do estágio anterior são usadas para realimentar as entradas dos estágios seguintes, dificultando da mesma forma a sua programação paralela que, entretanto, seria possível na deciptação.

Figura 3.1.5 – Modo de operação CFB (79).

3.1.2.5 Counter mode CTR

O modo de operação CTR (*Counter Mode*) foi introduzido por Diffie e Hellman (44) em 1979, devido a algumas falhas no modo OFB, e foi o último em ser padronizado pelo NIST (46). No modo CTR não há formalmente um vetor de inicialização, no lugar deste, há um “contador”. A função do contador é a parte fundamental do CTR, já que ele garante a unicidade da sequência produzida. O algoritmo de contador mais simples é de

incrementar γ numa unidade a cada etapa do processo, ou da concatenação de um valor inteiro do contador a γ e, ser incrementado da mesma forma.

A Equação (3.1.7) apresenta formalmente o processo de encriptação do CTR. Nota-se que a função “rand” implica um gerador de números aleatórios, a qual toma como semente ao valor do vetor de inicialização.

$$\begin{aligned} C_j &= P_j \oplus O_j, & j = 1, 2, \dots, b, \\ O_j &= E_K(\alpha_j), \\ \alpha_j &= \alpha_{j-1} + 1, \\ \alpha_1 &= \text{rand}(\gamma). \end{aligned} \tag{3.1.7}$$

A Figura 3.1.6 mostra o esquema de encriptação/decriptação do modo de operação CTR.

Figura 3.1.6 – Modo de operação CTR (79).

Uma das vantagens que destacam ao CTR é a independência com respeito aos blocos cifrados predecessores, que torna o CTR totalmente paralelizável tanto no processo de encriptação como na decriptação. Além disso, a função de encriptação E é aplicável para ambos os processos.

3.1.2.6 *Propagating cipher-block chaining* PCBC

O modo de operação PCBC (*Propagating Cipher-Block Chaining*) é uma variação do CBC. O PCBC foi projetado para propagar pequenas alterações no texto cifrado, visando melhorar o modo CBC. Embora não esteja formalmente publicado pelo NIST existem diversas versões dele na comunidade científica. Considera-se a Equação (3.1.8)

para definir o processo de encriptação do PCBC.

$$\begin{aligned} C_j &= E_K(P_j \oplus P_{j-1} \oplus C_{j-1}), & j = 1, 2, \dots, b, \\ C_1 &= E_K(P_1 \oplus \gamma). \end{aligned} \quad (3.1.8)$$

No PCBC também é necessário o emprego do vetor de inicialização, de maneira similar à maioria dos modos de operação estudados, com exceção do ECB. O esquema de encriptação e decrptação do PCBC são mostrados na Figura 3.1.7.

Figura 3.1.7 – Modo de operação PCBC.

3.1.3 Algoritmos de *hash* criptográficos

Os algoritmos *hash* ou função de mão-única (*one-way function*) tem larga aplicação na Criptologia, como por exemplo, na assinatura digital (35), compressão de documentos, autenticação de documentos, etc., sendo de fundamental importância na realização de transações financeiras e outras operações. Um algoritmo *hash* é um função matemática que mapeia uma mensagem de tamanho arbitrário e gera uma “mensagem digesto” (resultado *hash* ou *hashcode*) de tamanho fixo.

Terminologia:

A seguir é dada a notação padrão na literatura de acordo com Mironov (61):

- $\mathcal{M} \in \{0, 1\}^*$ — A mensagem de comprimento arbitrário, $N = |\mathcal{M}|$;
- $H()$ — Função *hash* com $H : \{0, 1\}^* \mapsto \{0, 1\}^h$;
- $H(\mathcal{M}) \in \{0, 1\}^h$ — resultado *hash*, ou *hashcode* de aplicar a função *hash* à mensagem $\mathcal{M}$, de comprimento h .

Definição: Uma função *hash* $H()$ é dita de mão-única, se satisfaz as três propriedades (35) descritas a seguir. Primeiro, que a função $H()$ é uma função de conhecimento público e não necessita de alguma informação secreta para realizar suas operações. Segundo, dado $\mathcal{M}$, é fácil de calcular $H(\mathcal{M})$. Terceiro, que deve ser computacionalmente improvável de encontrar duas mensagens diferentes $\mathcal{M}, \mathcal{M}'$ tal que $H(\mathcal{M}) = H(\mathcal{M}')$ (resistência a colisões). A propriedade da resistência a colisões estabelece que dada qualquer alteração na mensagem original, por mínima que for, acabará por gerar uma alteração significativa no resultado *hash* final.

Construção da função *hash*

De acordo com Mironov (61), uma função *hash* criptográfica consistiria de dois componentes. O primeiro componente é a **função de compressão** que mapeia uma entrada de comprimento fixo e devolve uma saída de comprimento fixo também. O segundo componente é o algoritmo de **extensão de domínio**, que é uma construção genérica que transforma a função de compressão à uma função *hash* com entrada arbitrária. Um dos algoritmos mais simples e conhecido é a construção Merkle-Damgård, e funciona da seguinte forma (Ver Figura 3.1.8):

Figura 3.1.8 – Princípios de construção de uma função *hash* baseado no modelo Merkle-Damgård (41).

Dentre os algoritmos *hash* mais conhecidos na Criptografia, e que são baseados no algoritmo de extensão de domínio Merkle-Damgård, são o MD5 (*Message Digest Algorithm 5*), o SHA-1 (*Secure hash algorithm 1*) e o Whirlpool. O MD5 foi projetado no ano 1991 por Ronald Rivest (72), com um comprimento do *hashcode* de 128 bits. O MD5 chegou a ser amplamente usado na Internet, porém foi quebrado no ano 1993 (38). O algoritmo SHA-1 foi proposto pelo NIST no ano 1995 (64), e baseia-se nos mesmos princípios que os utilizados pelo MD5, exceto que o SHA-1 tem comprimento *hashcode* de 160 bits, porém já foi quebrado teoricamente (80). O Whirlpool foi criado no ano 2000 pelo coautor

do AES (Advanced encryption Standard) e o cientista brasileiro Paulo Barreto (37). O Whirlpool com comprimento *hashcode* de 512 bits é um dos algoritmos *hash* mais utilizados no mercado e atualmente adotado pela Organização Internacional para Padronização (ISO).

3.1.4 Gerador de números pseudo-aleatórios

Sequências de números aleatórios desempenham um papel importante em diversos ramos da ciência, como por exemplo, na simulação estocástica para a modelagem de sistemas, na teoria da decisão, no cálculo numérico, e essencialmente na Criptografia. Um gerador de números pseudo-aleatórios (PRNG), é um algoritmo determinístico que produz sequências de números que aparentam ser “aleatórias”, com propriedades estatísticas ideais para produzir números independentes e uniformemente distribuídos sobre o intervalo $[0, 1)$, e com uma periodicidade de repetição extremadamente longa (49).

Um dos geradores mais conhecido é o gerador linear congruencial (LCG), o qual foi introduzido por Lehmer em 1951 (70). O LCG possui a seguinte equação de recorrência:

$$x_{n+1} = (Ax_n + B) \bmod C. \quad (3.1.9)$$

Sendo A (multiplicador), B (aditivo) e C (módulo) constantes inteiras. Um número aleatório x_{n+1} é gerado empregando o número anterior x_n , e x_0 é chamado de semente inicial. De acordo com Press *et al.* (70), existem diversos princípios para a escolha de A, B e C afim de obter a maior periodicidade possível. Porém, de acordo com Hellekalek (52), este tipo de geradores possui diversas desvantagens, *e.g.*, o LCG pode ser facilmente quebrado para certos parâmetros.

3.2 Criptografia baseada em caos

Até o momento, foi abordada a Criptografia na sua forma convencional baseada na matemática discreta e na teoria dos números. Mas, há quinze anos foi proposta a teoria do caos como uma nova abordagem para criar sistemas de criptografia (36). A partir desta, muitos pesquisadores têm se interessado na relação entre o caos e a Criptografia, aproveitando-se de muitas das propriedades dos sistemas caóticos (*e.g.*, imprevisibilidade, instabilidade, sensibilidade às condições iniciais, dinâmica determinística, ergodicidade) com a finalidade de empregar estas vantagens para criar sistemas de criptografia.

Tabela 3.2.1 – Comparação entre as propriedades dos sistemas caóticos e as propriedades da Criptografia (25).

Propriedade Caótica	Propriedade Criptográfica	Descrição
Ergodicidade	Confusão	A saída tem a mesma distribuição de qualquer entrada
Sensibilidade às condições iniciais	Difusão	Uma pequena variação na entrada pode causar grande alteração na saída
Transitividade topológica (implica ergodicidade)	Difusão, uma pequena mudança em um dos blocos muda completamente todo o texto claro	Uma pequena variação local pode causar uma grande variação em todo o espaço
Dinâmica determinista	Pseudo-aleatório (determinista)	Um processo determinista pode causar um comportamento pseudo-aleatório.

De acordo com diversas pesquisas (25, 29, 54), muitas das propriedades caóticas são relacionadas com as propriedades criptográficas homólogas, detalhado na Tabela 3.2.1. Estas propriedades também são apontadas como requisitos que devem ser cumpridos para garantir uma “boa” encriptação.

De acordo com Arroyo (29), a construção de um sistema criptográfico baseado em caos, baseia-se na oportuna escolha de um sistema dinâmico caótico a ser usado pelo sistema. Arroyo recomenda seguir alguns critérios básicos para a concepção de um algoritmo criptográfico, afim de evitar uma iminente quebra do algoritmo (29), entre eles:

1. Assegurar o comportamento caótico do sistema dinâmico;
2. Garantir o efeito avalanche;
3. Alto nível de entropia evitando fugas de informação por causa dos parâmetros de controle;
4. Processo de encriptação evitando a reconstrução da dinâmica caótica, *i.e.*, evitar as possíveis fontes de sincronização do sistema dinâmico;
5. Mapas caóticos com histogramas uniformes independente dos parâmetros de controle;
6. Seleção de mapas caóticos com alta sensibilidade dos parâmetros de controle;
7. A chave ou senha deve ser independente do número de iterações do sistema dinâmico caótico;

Na literatura, encontram-se diversos algoritmos de encriptação baseados na teoria do caos, entre estes usando mapas caóticos como o mapa logístico (36), o atrator de Lorenz (58), mapa de Tent (51), além de algoritmos *hash* baseado em caos (42, 93). Também encontram-se na literatura o emprego de autômatos celulares caóticos para criar algoritmos de criptografia que serão relatados a seguir.

3.2.1 Criptografia baseada em AC

No ano 1985, Wolfram (12) introduziu a primeira cifra de fluxo baseado no ACE. A principal ideia desta cifra foi de empregar as características caóticas da regra 30 do Wolfram, e assim aplicar o operador XOR bit a bit. Embora a introdução dos ACs na Criptografia não seja relativamente nova, devemos notar que as primeiras tentativas precedem às tentativas com a teoria do caos, como é caso do algoritmo do Wolfram (12) comparado com o primeiro algoritmo baseado na teoria do caos do Baptista (36) no ano 1998. Porém, muitos destes trabalhos iniciais não contavam com bases sólidas da Criptografia moderna e eram destinadas apenas a casos de estudo. Mas, a partir da relação entre a teoria do caos e os autômatos celulares (12), vieram propostas mais elaboradas.

No 1994 Nandi *et al.*, (14) empregou um ACE híbrido (não uniforme), nesse sentido eram alternadas as regras 90 e 150, por meio de um controlador PCA (autômato celular programável) que selecionava as regras. Mais tarde no trabalho de Tomassini *et al.*, (18), empregou-se um ACE híbrido com as regras 90, 105, 150 e 165, com vantagens para ser usadas como geradores de números pseudo-aleatórios. Seguindo a mesma linha, também foi proposto o trabalho do Seredynski *et al.*, (19), assim como Guan e Tan (16), para criar uma cifra de fluxo. Outras propostas de sistemas de criptografia baseado AC bidimensionais, encontra-se em (15, 22). Outro exemplo seria a abordagem de PRNGs baseados em AC (15–17, 20–22), cujos algoritmos de criptografia baseiam-se no AC na obtenção de sequências pseudo-aleatórias.

3.3 Criptoanálise

Nesta seção será brevemente apresentada a teoria da Criptoanálise, a ciência complementar da Criptografia. O principal objetivo da Criptoanálise é de estudar as diferentes formas de obter-se a mensagem original (quando teoricamente quebrado) sem conhecimento da chave, a partir do texto cifrado, ou tentar de encontrar a correspondente chave.

A criptoanálise procura por algoritmos que sejam mais eficazes que um ataque de força bruta, isto é, usando menos recursos computacionais que uma busca exaustiva. Na criptoanálise convencional (enfoque criptográfico), existem diversos modelos de ataques aos sistemas de criptografia (81). Estes modelos de ataque especificam certos conhecimentos (parcial ou total) sobre o sistema, para que o atacante formule seu ataque. Os mais conhecidos são:

- **Ataque do texto cifrado puro:** O atacante tem acesso apenas a textos cifrados;
- **Ataque de texto claro conhecido:** O atacante tem conhecimento de um trecho de texto claro e seu correspondente texto cifrado;
- **Ataque de texto (claro ou cifrado) escolhido:** O atacante pode escolher o texto claro (ou cifrado) obtendo seu correspondente texto cifrado (ou claro).

3.3.1 Criptoanálise, um novo enfoque

Por outro lado, a criptoanálise também procura encontrar fraquezas, ou vulnerabilidades, nos sistemas de criptografia, afim de quebrar estes e/ou conseguir informações secretas. Encontrar estas vulnerabilidades é comparada com noções semelhantes as do reconhecimento de padrões, *i.e.*, a detecção de padrões ou “assinaturas” em alguma das componentes de um sistema criptográfico.

Na literatura encontram-se diferentes abordagens para encontrar vulnerabilidades nos sistemas de criptografia. A maioria destes trabalhos tenta classificar as cifras de blocos por meio de metodologias como: os algoritmos de aprendizado de máquinas (algoritmos genéticos, redes neurais), técnicas de recuperação de informação, técnicas de reconhecimento de padrões, teoria do caos, etc.

Um dos primeiros arcabouços para à identificação de cifras de bloco foi proposto por Maheshwari (57), baseado em técnicas estatísticas (distribuição de frequências) e testes de aleatoriedade. Considerando o sistema criptográfico como uma caixa preta, o primeiro passo para uma ataque de criptoanálise é conhecer de qual cifra se trata. Assim a proposta de Maheshwari encaixa-se como um ataque de texto cifrado puro, a partir de um conjunto de textos cifrados com IDEA e DES. Porém, este trabalho possui poucas bases teóricas e resultados limitados.

Carvalho (39) aplicou técnicas de recuperação de informação e inteligência artificial para à classificação de cifras de blocos DES e AES, empregando técnicas de agrupamento,

que permitem a identificação de padrões e separando textos cifrados de acordo com a chave. Na mesma linha, Souza (78) desenvolveu uma rede neural artificial, baseada no mapa de Kohonen, para o agrupamento e classificação de textos cifrados, o método foi aplicado nas cifras de blocos DES, AES e RSA, cujos experimentos revelaram correlacionamento entre as chaves e os textos cifrados. Porém, as cifras de bloco DES, AES e RSA consideradas em (39,78) pertencem a diferentes famílias, cujo tamanho de chave difere e que “aparentemente” não influenciariam.

No trabalho de Dileep e Sekhar (45) foram empregadas máquinas de vetores de suporte (SVM) para a identificação das cifras de bloco: DES, Triple DES, Blowfish, AES e RC5, em base a um conjunto de textos cifrados correspondente a cada uma das cifras. Nagireddy (63) explorou vários métodos como histogramas de casamento, SVM e modelo de mistura Gaussiano, para o problema de identificação e detecção de padrões nas cifras de bloco DES, AES, Blowfish, TDES e RC5.

Muitos dos trabalhos mencionados possuem propriedades similares, no sentido que servem para a identificação de cifras de blocos, com porcentagens de precisão promissoras. Outra característica é que os testes realizados foram feitos com cifras de bloco e modo de operação ECB, sendo que a precisão decresce com outros modos, por exemplo CBC têm sido resistente a muitas tentativas de classificação. Além disso, a maioria dessas propostas não consideram o tamanho das chaves ou dos blocos, sendo que estes parâmetros poderiam influenciar em cada família de cifra de blocos.

3.4 Considerações finais

Este capítulo foi dividido em duas partes: Criptografia e Criptoanálise. Na primeira parte foi apresentada uma revisão teórica das primitivas criptográficas: cifras de blocos e os modos de operação, algoritmos de *Hash*, gerador de números pseudo-aleatórios e finalmente uma revisão bibliográfica sobre a Criptografia baseado na teoria do Caos assim como nos ACs caóticos. Na segunda parte foi abordada uma visão geral da Criptoanálise, assim como a identificação de vulnerabilidades nos sistemas de criptografia a partir do reconhecimento de padrões, que vem sendo pesquisado pelos criptoanalistas das diversas áreas afines à Criptografia com métodos e propostas promissoras. No próximo capítulo serão introduzidas diversas medidas, tanto da teoria do Caos para medir a sensibilidade às condições iniciais dos ACs, assim como as medidas da teoria da Informação, visando selecionar ACs com características que possam ser aproveitadas na Criptografia e na

Criptoanálise.

4 *Medidas e análise dos ACs caóticos*

Certos autômatos celulares caóticos são de grande interesse pois, cumprem com os princípios fundamentais de confusão e difusão estipulados por Shannon (77) para a construção apropriada de um sistema de criptografia. Mas, é necessário selecionar ACs cujas características sejam propícias as da Criptografia. Assim, foram consideradas várias medidas da teoria do Caos e da teoria da Informação para a procura de um candidato apropriado aos objetivos deste trabalho, e o enfoque limita-se a analisar o comportamento dinâmico dos ACs da família *life-like* quanto aos seguintes critérios:

- O AC deve possuir regime caótico, *i.e.*, sensibilidade às condições iniciais ($\lambda > 0$),
- Estados da vizinhança com alta entropia;
- Que cada configuração futura do AC seja diferente da anterior;
- Não possuir padrões reconhecíveis.

Neste capítulo será apresentado o método para o cálculo do expoente de Lyapunov, entropia e distância de Hamming e sua aplicação às regras dos ACs da família *life-like*.

4.1 **Expoente de Lyapunov λ nos ACs**

A estabilidade dos sistemas dinâmicos é avaliada calculando o LE que quantifica a taxa de divergência do sistema, quando evolvido a partir de duas condições iniciais muito próximas (47). Diversos métodos foram desenvolvidos para estimar o LE em ACs (31–33). Neste trabalho é considerado o algoritmo proposto por Baetens e De Baets (31) baseado em derivadas booleanas (ver Tabela 4.1.1). Nele é definido o **vetor de defeitos** $h(c_i, t)$,

que indica quais são as células perturbadas c_i a cada iteração t . Por exemplo, o vetor de defeitos da célula c_i na primeira iteração é dada da seguinte forma:

$$h(c_i, 1) = \begin{cases} 1 & \text{se } s^*(c_i, 1) \neq s(c_i, 1) \\ 0, & \text{caso contrário.} \end{cases} \quad (4.1.1)$$

A idéia principal do algoritmo é calcular o número total de células perturbadas que foram originadas a partir de duas configurações iniciais s_0 e s_0^* muito próximas. Onde a distância mais próxima que podem ter é $\sum_{c_i} h(c_i, 0) = 1$.

Tabela 4.1.1 – Algoritmo para calcular o expoente de Lyapunov dos ACs proposto em (31).

$t = 0$	<ul style="list-style-type: none"> • Considerar s_0 e s_0^*, tal que $\sum_{c_i} h(c_i, 0) = 1$. • Armazenar s_0^* no conjunto A_0.
$t = 1$	<ul style="list-style-type: none"> • Iterar e calcular $s(c_i, 1)$ e $s^*(c_i, 1)$. • Calcular o vetor de defeitos $h(c_i, 1)$. • Para todo c_i, tal que $h(c_i, 1) = 1$. Criar uma réplica R^i tal que $R^i(c_i, 1) = \bar{s}(c_i, 1)$ (complemento booleano), de tal forma que $R^i(c_q, 1) = s(c_q, 1)$, para todo $c_q \neq c_i$. • Armazenar as réplicas no conjunto A_1.
$t = 2$	<ul style="list-style-type: none"> • Iterar e calcular $s(c_i, 2)$ e $R(c_i, 2)$, para todo $R \in A_1$. • Calcular o vetor de defeitos da seguinte forma: $h(c_i, 2) = \#\{R \in A_1 R(c_i, 2) \neq s(c_i, 2)\} \quad (4.1.2)$ • Para todo $c_i \wedge R \in A_1$, tal que $R(c_i, 2) \neq s(c_i, 2)$. Criar uma réplica R^i tal que $R^i(c_i, 2) = \bar{s}(c_i, 2)$, de tal forma que $R^i(c_q, 2) = s(c_q, 2)$, para todo $c_q \neq c_i$. • Armazenar as réplicas no multiconjunto $A_2 = \{(R^i, h(c_i, 2)) R \in A_1 \wedge R(c_i, 2) \neq s(c_i, 2)\}$
$t + 1$	<ul style="list-style-type: none"> • Repetir o passo anterior para cada $t + 1$. Nota-se que cada novo multiconjunto A_{t+1} é construído com base nas réplicas em A_t.

Finalmente o Máximo Expoente de Lyapunov (MLE) $\lambda(t)$ é definido como o número total de células perturbadas a cada iteração:

$$\lambda(t) = \frac{1}{t} \log \left(\frac{\sum_{c_i} h(c_i, t)}{\sum_{c_i} h(c_i, 0)} \right). \quad (4.1.3)$$

4.2 Medidas da teoria da informação

4.2.1 Entropia

A medida de entropia proposta por Shannon (76) é utilizada na Criptografia para medir a quantidade de informação contida num texto cifrado. Deste outro ponto de vista, valores altos de entropia indicariam um sistema imprevisível. Da mesma forma, pode ser medida a entropia dos ACs obtida a cada iteração conforme a Equação 4.2.1.

$$H = - \sum_{i=0}^z P_i \log_2(P_i). \quad (4.2.1)$$

Em que P_i é a probabilidade de células vivas ou mortas ($z = 2$ possíveis estados) a cada configuração do AC, sendo que $0 \leq H \leq 1$. O valor máximo de entropia que pode ser atingido é $H=1$, o qual implicaria um alto nível de imprevisibilidade.

4.2.2 Distância de Hamming

A distância de Hamming, representado por D_H , é definida por:

$$D_H = \frac{\sum s(c_i, t) \oplus s(c_i, t + 1)}{|\mathcal{T}|}. \quad (4.2.2)$$

Esta distância é empregada nos ACs como uma medida para estimar o número de diferenças entre os estados das células de duas configurações $s(c_i, t)$ e $s(c_i, t + 1)$. Neste sentido, o maior número de perturbações implicará no menor número de repetição de padrões, e assim cada configuração futura do AC seja diferente da anterior.

O valor máximo normalizado que a distância de Hamming pode ter é $D_H = 1$, o que representaria o melhor dos cenários, isto é, quando todas as células de uma configuração são opostas aos da configuração futura.

4.2.3 Combinação de medidas

Como pode ser observado, as três medidas dadas: o expoente de Lyapunov, a entropia e a distância de Hamming, abordam diferentes aspectos dos ACs, na procura de valores altos afim de atingir a maior “aleatoriedade” para um sistema de criptografia. Os valores

isolados da entropia, o expoente de Lyapunov e a distância de Hamming podem não ser suficientes para distinguir quem é o melhor candidato.

Para tanto, é utilizada uma combinação destas medidas, multiplicando H , λ e D_H , para maximizar os resultados, *i.e.*, $\text{Max} = (H\lambda D_H)$.

4.3 Seleção do AC caótico

A família de AC *life-like* é composta por 262144 regras de transição da forma By/Sx . Portanto, fazer uma busca exaustiva requereria um alto custo computacional, por esse motivo, o espaço de busca foi reduzido a apenas um conjunto de regras mais conhecidas, com o objetivo de descobrir algumas regras que apresentem um comportamento mais caótico por ter alta entropia, expoente de Lyapunov e distância de Hamming.

Para a realização desta seleção, utilizaram-se ACs de 128×128 com condições iniciais aleatórias. Para cada uma das medidas estudadas foi calculada a média respectiva dependendo do número de iterações, assim para estimar a entropia H (Equação (4.2.1)) foi calculada a média de após $t = 2 \cdot 10^7$ iterações. Para estimar o LE foram empregadas várias configurações de células perturbadas e foi calculada a média de λ_{200} empregando a Equação (4.1.3). Finalmente, foi calculada a distância média de Hamming durante $t = 1000$ iterações usando a Equação (4.2.2).

As Tabelas 4.3.1–4.3.2 mostram os resultados após aplicação das 3 métricas estudadas nas regras sugeridas. Os resultados obtidos estão em ordem decrescente de acordo com seus valores de Max, também é mostrado o AC correspondente numa malha de 128×128 após 1000 iterações. As três primeiras regras com as medidas mais altas são: B1357/S2468, B357/S1358, e B23/S36, o qual pode ser examinado nas subfiguras da evolução dos ACs. A maior parte das regras são caóticas $\lambda > 0$, com exceção das regras B35678/S5678, B3678/S235678, que são dinamicamente estáveis (atrator fixo) com $\lambda = -\infty$. Na seção seguinte, será analisado o comportamento dinâmico da regra selecionada B1357/S2468.

Tabela 4.3.1 – Análise comparativa de regras de autômatos celulares da família *life-like*, baseada nas medidas de entropia, expoente de Lyapunov e distância de Hamming (D_H). As regras estão em ordem decrescente (da esquerda para direita) de acordo com seus valores de Max (56).

<i>B1357/S2468</i>			<i>Amoeba</i> <i>B357/S1358</i>
Entropia	0.9999		Entropia	0.9830	
LE	2.1716		LE	1.9811	
D_H	0.5019		D_H	0.4393	
Max	1.0897		Max	0.8556	
<i>B23/S36</i>			<i>Serviettes</i> <i>B234/S</i>
Entropia	0.9686		Entropia	0.9513	
LE	1.4244		LE	0.8087	
D_H	0.5461		D_H	0.7417	
Max	0.7534		Max	0.5706	
<i>34 Life</i> <i>B34/S34</i>			<i>Long life</i> <i>B345/S5</i>
Entropia	0.9843		Entropia	0.9040	
LE	1.1673		LE	0.8099	
D_H	0.4603		D_H	0.6796	
Max	0.5289		Max	0.4976	
<i>Seeds</i> <i>B2/S</i>			<i>WalledCities</i> <i>B45678/S2345</i>
Entropia	0.7430		Entropia	0.9973	
LE	1.3430		LE	0.9470	
D_H	0.4196		D_H	0.4216	
Max	0.4187		Max	0.3981	
<i>Gnarl</i> <i>B1/S1</i>			<i>Coagulations</i> <i>B378/S235678</i>
Entropia	0.7755		Entropia	0.3777	
LE	1.1803		LE	1.5474	
D_H	0.3811		D_H	0.1634	
Max	0.3488		Max	0.0955	
<i>2x2</i> <i>B36/S125</i>			<i>Pseudo life</i> <i>B357/S238</i>
Entropia	0.1800		Entropia	0.0954	
LE	1.6239		LE	1.5369	
D_H	0.1323		D_H	0.0932	
Max	0.0387		Max	0.0137	

Tabela 4.3.2 – (Continuação) Análise comparativa de regras de autômatos celulares da família *life-like*, baseada nas medidas de entropia, expoente de Lyapunov e distância de Hamming (D_H) (56).

<i>Conway's Life</i> <i>B3/S23</i>			<i>Coral</i> <i>B3/S45678</i>
Entropia	0.2120		Entropia	0.6949	
LE	1.2730		LE	1.0673	
D_H	0.0477		D_H	0.0160	
Max	0.0129		Max	0.0118	
<i>HighLife</i> <i>B36/S23</i>			<i>Day & Night</i> <i>B3678/S34678</i>
Entropia	0.1270		Entropia	0.0714	
LE	1.4238		LE	1.5480	
D_H	0.0397		D_H	0.0509	
Max	0.0072		Max	0.0056	
<i>Mazectric</i> <i>B3/S1234</i>			<i>Move</i> <i>B368/S245</i>
Entropia	1.0000		Entropia	0.0330	
LE	0.7011		LE	1.7938	
D_H	0.0039		D_H	0.0265	
Max	0.0027		Max	0.0016	
<i>Assimilation</i> <i>B345/S4567</i>			<i>Maze</i> <i>B3/S12345</i>
Entropia	0.7934		Entropia	1.0000	
LE	0.0582		LE	0.0460	
D_H	0.0030		D_H	0.0011	
Max	0.0001		Max	0.0000	
<i>Diamoeba</i> <i>B35678/S5678</i>			<i>Stains</i> <i>B3678/S235678</i>
Entropia	0.0000		Entropia	0.0875	
LE	-Infinito		LE	-Infinito	
D_H	0.0194		D_H	0.0024	
Max	0.0000		Max	0.0000

Figura 4.3.1 – Evolução do 2D-AC B1357/S2468 com diferentes configurações iniciais (56).

4.3.1 Análise da dinâmica do AC *life-like* B1357/S2468

A seguir será analisada, mais profundamente, o comportamento caótico da regra B1357/S2468. Posteriormente, a sensibilidade às condições iniciais desta regra será introduzida na Seção 2.6. Na Figura 4.3.1, observa-se a natureza caótica da regra B1357/S2468, as imagens representam um AC como uma malha de 64×64 , com população inicial de células vivas de 10%, 50%, e 90%, sendo iteradas no tempo. Em (C) podemos observar que a regra B1357/S2468 se estabiliza numa população aproximadamente de 50% de células vivas, após poucas iterações (tempo transiente). Porém, esta mantém $D_H \approx 50\%$ de células diferentes a cada iteração, uma característica muito conveniente para o propósito

deste trabalho.

4.4 Considerações finais

Neste capítulo foram introduzidos os critérios para selecionar um sistema dinâmico caótico com características aproveitáveis na Criptografia. Foram estudados especificamente autômatos bidimensionais da família *life-Like*, uma vez que essa família apresenta certas regras de AC caóticos compatíveis com a Criptografia como fonte de aleatoriedade de alta qualidade. Para isto, realizou-se experiências para selecionar uma regra com base nas medidas da teoria do caos e da teoria da informação, e com base nestes resultados se optou pela regra B1357/S2468, a qual será empregada na primeira proposta desta dissertação introduzida no Capítulo 5.

5 *Proposta de algoritmos de Criptografia caótica*

Neste capítulo são propostos dois algoritmos baseados em autômatos celulares caóticos. A primeira proposta é de um algoritmo de cifra de bloco e a segunda é um algoritmo de *hash* empregando uma rede complexa como tesselação do AC. Cada proposta e método é descrito e detalhado e, em seguida, são usados vários experimentos para avaliá-las e/ou compará-las com outros métodos da literatura. Ambos os algoritmos propostos seguem com as recomendações para uma apropriada construção de um sistema criptográfico, de acordo com diversas pesquisas (25, 29) como foi mencionado durante a Seção 3.2.

5.1 Algoritmo de cifra de bloco baseado em autômato celular caótico

É proposto um algoritmo de cifra de bloco baseado em AC caótico, especificamente um AC bidimensional da família *life-like*, exposto nos Capítulos 2 e 4. A proposta é separada em quatro componentes principais ilustrados na Figura 5.1.1. A idéia principal da proposta é que o AC caótico da regra B1357/S2468 atue como um gerador de números pseudo-aleatórios. Os números pseudo-aleatórios são gerados constantemente enquanto o AC é iterado, estes números são agrupados em blocos de 8 bits que são usados durante o processo de encriptação/decriptação.

Também é empregada uma função baseada no mapa logístico que encarrega-se de converter a chave K (usada como semente) para criar as condições iniciais do AC, visando que cada célula esteja composta pelos valores binários gerados pelo mapa. Desta forma a chave é dispersada na configuração inicial do AC.

As seções seguintes detalham cada um destes componentes, com exceção do componente do AC caótico e a seleção da regra B1357/S2468 que foram descritas no Capítulo 4.

Figura 5.1.1 – Proposta de um algoritmo de criptografia baseado em AC caótico, dividido em quatro componentes principais: (a) A transformação da chave na condição inicial do AC usando o mapa logístico, (b) O AC é definido com a regra B1357/S2468 e é iterada sucessivas vezes. (c) O AC atua como um PRNG para gerar blocos de números pseudo-aleatórios. (d) O processo de encriptação aplica operação de XOR entre os textos claros e os blocos gerados. Nota-se que o processo de deciptação emprega o *design* inverso.

5.1.1 Transformação da chave usando o mapa logístico

O mapa logístico $X_{h+1} = \mu X_h(1 - X_h)$ introduzido na Equação (2.1.1), possui espaço de fase no intervalo $[0, 1]$. Este recebe a chave K como uma sequência de $k = 128$ bits, que é previamente normalizada como condição inicial (semente), para ser transformada em um valor $\Omega \in [0, 1[$ da seguinte forma:

$$\Omega = \frac{\sum_{i=1}^k 2^{8(i-1)} K_i}{2^{8k}}. \quad (5.1.1)$$

A condição inicial do mapa logístico é dada por $X_0 = \Omega + \omega$, com uma constante $\omega = 10^{-5}$, pois $X_0 = 0$ seria trivial. Além disso é usado $\mu = 4.0$, já que a região mais caótica do mapa logístico se encontra entre $3.9 \leq \mu \leq 4.0$. Assim, o mapa logístico se encarrega de preencher a configuração inicial s_0 do AC, enquanto o mapa é iterado após do tempo transiente $\alpha = 1000$.

Considerando à tesselação $\mathcal{T}$ de um AC como uma matriz de ordem $m \times n$, com $1 \leq i < m$; $1 \leq j < n$, considerando-se neste trabalho $|\mathcal{T}| = 128 \times 128$. Assim, X_h é discretizado em base à seguinte equação:

$$s(c_{n(i-1)+j-1}, 0) = \begin{cases} 1 & \text{se } X_{((n(i-1)+j-1)+\alpha) < 0.5} \\ 0 & \text{se } X_{((n(i-1)+j-1)+\alpha) \geq 0.5} \end{cases} \quad (5.1.2)$$

5.1.2 Gerador de números pseudo-aleatórios baseado no AC

Aqui é descrito um algoritmo PRNG que emprega o AC (ver Seção 3.1.3 e 3.2.1). A idéia é simples: concatenar os estados das células c_i (bits) para compor blocos de números pseudo-aleatórios conforme o AC é iterado. Assim, todos os estados das células $s(c_i, t)$ são armazenadas no vetor de bits $\vec{\beta}$, tal que $|\vec{\beta}|$ dependerá do número de iterações t necessárias para preencher o tamanho do texto claro a ser encriptado, o qual será discutido na segunda parte da proposta.

$$\beta_{n(i-1)+(j-1)+(tmn)} = s(c_{n(i-1)+(j-1)}, t). \quad (5.1.3)$$

Considere $\vec{B}$ um vetor de blocos de bytes, tal que $|\vec{B}| = \lfloor \frac{|\beta|}{8} \rfloor$

$$B_i = \sum_{j=1}^9 2^{8-j} \beta_{8(i-1)+j}. \quad (5.1.4)$$

Considere $\vec{Y}$ um vetor composto de ρ blocos, para um inteiro $\rho > 0$, tal que $|\vec{Y}| \leq \lfloor \frac{|\beta|}{\rho} \rfloor$. Em seguida, é aplicada a operação XOR dos blocos consecutivos em $\vec{B}$, da seguinte forma:

$$Y_i = B_{(i-1)\rho} \oplus B_{(i-1)\rho+1} \oplus \dots \oplus B_{(i-1)\rho+\rho}. \quad (5.1.5)$$

5.1.3 Encriptação e decriptação

Considere o texto claro $\mathbf{P}$ um vetor que contem N bits distribuídos em b blocos após o processo de preenchimento se for o caso, de tal forma que $|\vec{Y}| = N$, para o qual é necessário que $|\vec{\beta}| = \lceil \frac{8N}{mn} \rceil$.

O processo de encriptação proposto é da forma:

$$\begin{aligned} C_{i+1} &= P_{i+1} \oplus C_i \oplus Y_{i+1} \\ C_1 &= Y_1 \oplus P_1 \quad \text{para } i = 1, 2, \dots, b. \end{aligned} \quad (5.1.6)$$

O processo de decriptação proposto é da forma:

$$\begin{aligned} P_{i+1} &= C_{i+1} \oplus Y_i \oplus C_i \\ C_1 &= Y_1 \oplus P_1 \quad \text{para } i = 1, 2, \dots, b. \end{aligned} \quad (5.1.7)$$

5.1.4 Resultados

Esta seção apresenta resultados experimentais da execução do algoritmo de cifra de bloco proposto e respectiva análise. Também são apresentados alguns testes de aleatoriedade visando comparar os resultados obtidos com outros encontrados na literatura.

Foi cifrada uma imagem de 511×516 pixels empregando duas chaves com um bit de diferença (bit menos significativo). Na esquerda da Figura 5.1.2 é apresentada a imagem original (texto claro), e as respectivas imagens cifradas. Observa-se que as imagens cifradas são muito mais uniformes e que estas não guardam semelhança com a imagem original.

Da mesma forma, foi analisada a distribuição estatística do teste anterior. Foi gerado um histograma das imagens obtidas, mostradas na mesma figura. Como pode ser observado os histogramas das imagens cifradas possuem uma distribuição uniforme e diferente, demonstrando a alta sensibilidade à chave. Além disso, a frequência que todos os 256 números (código ASCII) gerados pelo AC se encontram mais distribuídas, pois parece esconder toda informação da imagem original.

Figura 5.1.2 – Encriptação de uma imagem e suas respectivas imagens cifradas a) e b) com o algoritmo proposto, usando duas chaves com um bit de diferença (esquerda) e a distribuição de frequências da imagem original e das respectivas imagens cifradas (direita).

5.1.4.1 Teste de aleatoriedade

Os testes de aleatoriedade, a maioria deles estatísticos, são usados para analisar os padrões de distribuição das sequências de números gerados por um PRNG. Estes testes **tentam** verificar a aleatoriedade dos dados, ou mesmo revelar a presença de distribuições não óbvias. Portanto é apenas usado como um indicador de pseudo-aleatoriedade, pois nenhum destes testes estatísticos pode certificar um PRNG como “apropriado” para alguma aplicação em particular (52, 73).

Foram estudados vários conjuntos de testes, afim de contrastar os resultados obtidos com outros algoritmos baseados em ACs encontrados na literatura. Aqui são considerados dois testes:

- ENT: composto por cinco medidas: Entropia, chi-quadrado, coeficiente de correlação serial (SCC), média aritmética e Monte Carlo (84).
- DIEHARD: Composto por 18 testes estatísticos (59). O resultado de cada teste é uma probabilidade p entre $[0, 1)$. É considerado como “falhou” quando o valor de p é próximo de 1 ou 0, caso contrário, pode ser considerado como “passou”.

Para avaliar a pseudo-aleatoriedade do PRNG baseado no AC proposto, é necessário isolá-lo para que possa gerar sequências de números usando sementes aleatórias. Nessa condição, foram geradas 100 amostras de sequências de 10MB, cada uma com 10 valores diferentes de números de blocos ρ . A seguir são realizados os testes usando ENT, e em seguida o DIEHARD.

A Tabela 5.1.1 exibe os resultados da média e desvio padrão das cinco medidas do ENT, para diferentes valores de ρ . Segundo o teste de ENT, todos os resultados são considerados de qualidade aceitável. Nota-se que conforme o valor ρ é incrementado a qualidade do PRNG também aumenta, porém o custo computacional é elevado.

5.1.5 Comparação com outros algoritmos

Na Tabela 5.1.2 é mostrada a comparação entre os resultados obtidos para $\rho = 10$ blocos e outros algoritmos encontrados na literatura (15–17, 22). Como pode ser visto o PRNG usado na proposta supera os outros algoritmos em termos da entropia e SCC, embora os resultados do chi-quadrado obtidos não superem a os valores atingidos por Shin *et al.*, (15).

Da mesma forma que nas experiências anteriores, foi aplicado o conjunto de testes DIEHARD sobre o mesmo conjunto de dados. Neste conjunto de testes apenas existe os termos “passou” e “falhou”, e as probabilidades p obtidas não devem ser considerados como indicadores de qualidade (59). Considera-se que um PRNG bem projetado deve passar em todos os testes, para não ser descartado. Na Tabela 5.1.3 são apresentados os resultados obtidos do algoritmo de cifra de bloco caótico proposto com o DIEHARD, os quais são comparados com os outros algoritmos de PRNG baseados no AC.

Tabela 5.1.1 – Resultados teste ENT (média e desvio padrão) aplicados ao PRNG proposto para diferentes números de blocos ρ (56).

21

ENT											
N°	Nome do Teste	Número de blocos ρ									
		1	2	3	4	5	6	7	8	9	10
1	Entropia (≈ 8.0)	7.9999821	7.9999817	7.9999818	7.9999819	7.9999822	7.9999822	7.9999822	7.9999825	7.9999819	7.9999819
	<i>Desv. padrão</i>	12.10^{-7}	14.10^{-7}	13.10^{-7}	12.10^{-7}	12.10^{-7}	14.10^{-7}	13.10^{-7}	13.10^{-7}	15.10^{-7}	15.10^{-7}
2	Chi-quadrado (≈ 1.0)	0.997893	0.984747	0.988358	0.995630	0.988156	0.990519	0.990106	0.968390	0.998606	0.998784
	<i>Desv. padrão</i>	0.068852	0.073852	0.070944	0.068508	0.065252	0.076053	0.066655	0.071157	0.080108	0.080108
3	SCC (≈ 0.0)	0.00401	0.00004	0.00008	-0.00002	0.00000	0.00005	-0.00004	0.00000	-0.00003	0.00002
	<i>Desv. padrão</i>	0.00028	0.00023	0.00025	0.00027	0.00025	0.00023	0.00024	0.00027	0.00021	0.00021
4	Média aritmét. ≈ 0.5	0.499998	0.499956	0.499991	0.499989	0.499978	0.499998	0.499991	0.499989	0.499988	0.499961
	<i>Desv. padrão</i>	0.000069	0.000062	0.000069	0.000067	0.000068	0.000076	0.000081	0.000075	0.000070	0.000070
5	Monte Carlo ($\approx \pi$)	3.14173	3.14182	3.14137	3.14154	3.14172	3.14170	3.14163	3.14138	3.14156	3.14127
	<i>Desv. padrão</i>	0.00092	0.00102	0.00105	0.00095	0.00097	0.00102	0.00105	0.00110	0.00103	0.00103

Tabela 5.1.2 – Comparação dos resultados obtidos na proposta com outros algoritmos na literatura (56).

ENT						
N°	Nome do teste	Tomassini (17)	Guan (16)	Shin (2008) (22)	Shin (2009) (15)	Proposta $\rho = 10$
1	Entropia (≈ 8.0)	7.999710	7.999830	7.999880	7.999910	7.999982
2	Chi-quadrado (≈ 1.0)	0.989412	0.992002	0.999281	0.999283	0.998784
3	SCC (≈ 0.0)	0.000227	0.000171	0.000051	0.000062	0.000016

Tabela 5.1.3 – Comparação dos resultados dos testes DIEHARD com $p = 90\%$.

DIEHARD							
Teste N°	Nome do teste	Chowdhury <i>et al.</i>	Tomassini <i>et al.</i>	Guan <i>et al.</i>	Shin <i>et al.</i> (2008)	Shin <i>et al.</i> (2009)	Proposta $\rho = 10$
1	Birthday Spacing	Passou	Passou	Passou	Passou	Passou	Passou
2	Over. 5-Per.	Falhou	Falhou	Falhou	Passou	Passou	Passou
3	Binary Rank 31×31	Passou	Passou	Passou	Passou	Passou	Passou
4	Binary Rank 32×32	Passou	Passou	Passou	Passou	Passou	Passou
5	Binary Rank 6×8	Falhou	Passou	Passou	Passou	Passou	Passou
6	Bitstream	Passou	Passou	Passou	Passou	Passou	Passou
7	OPSO	Falhou	Passou	Passou	Passou	Passou	Passou
8	OQSO	Falhou	Falhou	Falhou	Passou	Passou	Passou
9	DNA	Falhou	Passou	Passou	Passou	Passou	Passou
10	Count-The-1's 01	Falhou	Falhou	Passou	Passou	Passou	Passou
11	Count-The-1's 02	Falhou	Passou	Falhou	Passou	Passou	Passou
12	Parking Lot	Passou	Passou	Passou	Passou	Passou	Passou
13	Minimum Distance	Passou	Passou	Passou	Passou	Passou	Passou
14	3DS Spheres	Passou	Passou	Passou	Passou	Passou	Passou
15	Squeeze	Falhou	Passou	Passou	Passou	Passou	Passou
16	Overlapping Sums	Passou	Passou	Passou	Passou	Passou	Passou
17	Runs	Passou	Falhou	Passou	Passou	Passou	Passou
18	Craps	Passou	Passou	Passou	Passou	Passou	Passou

5.1.6 Considerações gerais do algoritmo de cifra de bloco proposto

Em primeiro lugar, as medidas do expoente de Lyapunov, entropia, distância de Hamming, e a combinação deles, destacam a regra B1357/S2468 como uma excelente escolha para gerar números pseudo-aleatórios.

Em segundo lugar, o PRNG é um componente importante do algoritmo proposto e por isto é importante garantir sua qualidade. Os resultados dos testes de DIEHARD e ENT aplicados ao PRNG, atingiram resultados similares e até superiores se comparados com os principais métodos encontrados na literatura (15–17,22). Porém, é necessário fazer uma comparação com outros algoritmos de criptografia tradicionais, cuja análise está fora do escopo deste trabalho.

Em geral, os resultados mostraram que o algoritmo de criptografia baseado no AC *life-like* possui boas qualidades de segurança e um grande potencial para ser aplicado em problemas reais.

5.2 Algoritmo de *hash* caótico proposto

Nesta segunda parte do capítulo, propomos um algoritmo de *hash*, seguindo o modelo de construção Merkle-Damgård, cujo objetivo é ser uma aplicação da combinação dos ACs caóticos e das redes complexas, a qual chamamos de AC caótico híbrido na Seção 2.7.1.

A idéia principal é usar cada bloco da mensagem $\mathcal{M}$ para perturbar a evolução do modelo AC caótico híbrido, assim todos os blocos influenciam no processo, até que finalmente é obtido o *hashcode*. Supondo que a mensagem $\mathcal{M} \in \{0, 1\}^N$ é dada como uma sequência de blocos $\mathcal{M} = \{\mathcal{M}_1, \mathcal{M}_2, \dots, \mathcal{M}_b\}$, com $n = 512$ bits cada. A função *Hash* proposta $H(\mathcal{M}) \in \{0, 1\}^h$ retorna o *hashcode* de comprimento $h = 512$, já que atualmente este é o comprimento padrão de acordo com a maioria dos algoritmos de *hash* tradicionais.

O algoritmo proposto é ilustrado na Figura 5.2.1. Para entender o funcionamento deste, empregaram-se valores prefixados em base ao gerador linear congruencial LCG (Seção 3.1.4) usando números primos e co-primos. Nota-se que em um cenário real, é sugerido o uso de diferentes parâmetros que ficam à livre escolha.

Figura 5.2.1 – Proposta do algoritmo de *Hash* caótico baseado em AC híbrido, de acordo com o modelo Merkle-Damgård.

5.2.1 Matriz de arestas A

Este algoritmo utiliza principalmente uma matriz de arestas A de ordem 512×8 ($h \times \langle \bar{k} \rangle$), onde cada linha i representa a vizinhança de c_i , e cujo tamanho é $|\mathbf{N}(c_i)| = \langle \bar{k}_i \rangle = 8$, como ilustrado na Figura 5.2.2. Esta matriz de arestas é a estrutura de dados que armazena o AC híbrido. E os estados das células $s(c_i, t+1)$ do AC são atualizados a cada iteração.

Figura 5.2.2 – Estrutura de dados da matriz de arestas do AC híbrido.

A função de transição do AC caótico híbrido é também B1357/S2468, formalmente dada na Equação (5.2.1).

$$s(c_i, t + 1) = \begin{cases} 1, & \text{se } \sum_{j=0}^{\langle \bar{k} \rangle} s(A_{i,j}, t) = 1 \wedge 3 \wedge 5 \wedge 7 \wedge s(c_i, t) = 0 \\ 1, & \text{se } \sum_{j=0}^{\langle \bar{k} \rangle} s(A_{i,j}, t) = 2 \wedge 4 \wedge 6 \wedge 8 \wedge s(c_i, t) = 1 \\ 0, & \text{caso contrário.} \end{cases} \quad (5.2.1)$$

5.2.2 O algoritmo de *Hash*

O cálculo do *hashcode* é formado a partir da última configuração do AC híbrido após iterada T_b vezes (tendo um limite de iterações). Essa é empregada para extrair 512 células, dentre as 512×8 células na matriz de arestas, e finalmente os estados destas células escolhidas, quando concatenadas, formarão o *hashcode*. É sugerido usar valores aleatórios fixados pelo LCG em $\vec{L}in$ e $\vec{C}ol$ (parâmetros do Algoritmo 5.2.1). Esta escolha de células é uma alternativa rápida, porém poderia ser usado diferentes heurísticas para a escolha destes. Por outro lado, este algoritmo contém certas restrições e limitações que não são abordadas nesta dissertação, entre estas o comprimento da mensagem deve ser maior de 512 bits, e cada bloco deve ser passado pelo esquema de preenchimento, *i.e.*, que cada bloco de mensagem contenha exatamente 512 bits.

Algoritmo 5.2.1 Algoritmo *Hash* caótico proposto $\vec{H} = H(\mathcal{M})$.

Declaração e inicialização de vetores $\vec{W}$, $\vec{T}$, $\vec{\text{Lin}}$, $\vec{\text{Col}}$
 Criar AC híbrido, na matriz de arestas A
 Iniciar estados do AC $s(c_i, 0) = M_{W_i}$, com $\vec{M} = \mathcal{M}_1$
para $i = 2 \rightarrow b$ **faça**
 $\vec{M} = \mathcal{M}_i$
 $\vec{u} = s(c_{i-1}, T_{i-1})$
 $\vec{v} = M_{W_i}$
 $\vec{z} = \vec{u} \oplus \vec{v}$
fim-para
 iterar $s(c_i, T_b)$
para $i = 0 \rightarrow h$ **faça**
 $H_i = A_{\text{Lin}_i, \text{Col}_i}$
fim-para
retornar $\vec{H}$

Algoritmo 5.2.2 Declaração e inicialização de vetores $\vec{W}$, $\vec{T}$, $\vec{\text{Lin}}$, $\vec{\text{Col}}$.

$W_0 = 31319$, $\text{Lin}_0 = 859433$, $\text{Col}_0 = 756839$, $T_0 = 132049$, $\text{maxIterações} = 107$, $i = 1$
enquanto $h \times \text{LCG}(W_{i-1}) \notin \vec{W}$ e, $i \leq h$ **faça**
 $W_i = h \times \text{LCG}(W_{i-1})$
 $i = i + 1$
fim-enquanto
para $i = 1 \rightarrow b$ **faça**
 $T_i = \text{LCG}(T_{i-1}) \times \text{maxIterações}$
fim-para
para $i = 1 \rightarrow h$ **faça**
 $\text{Lin}_i = h \times \text{LCG}(\text{Lin}_{i-1})$
 $\text{Col}_i = \langle \bar{k} \rangle \times \text{LCG}(\text{Col}_{i-1})$
fim-para

Algoritmo 5.2.3 Algoritmo para criar AC híbrido.

$X_0 = 788549$
para $i = 0 \rightarrow h$ **faça**
 para $j = 0 \rightarrow \langle \bar{k} \rangle$ **faça**
 $X_{i\langle \bar{k} \rangle + j} = \text{LCG}(X_{i\langle \bar{k} \rangle + j})$
 $A_{i,j} = X_{i\langle \bar{k} \rangle + j} \times h$
 fim-para
fim-para

Algoritmo 5.2.4 Algoritmo $\text{LCG}(X_0)$. Equação (3.1.9) normalizada.

$a = 17413869$, $b = 1000033$, $c = 9997954933$
return $((aX_0 + b) \bmod c)/c$

5.2.3 Resultados

O algoritmo de *Hash* proposto foi aplicado em um método de verificação comum na literatura. Neste teste são empregados dois tipos de entrada ASCII com uma frase e a mesma frase com uma letra variada, e é feito a comparação das duas *hashcodes*. A seguir são mostrados os resultados correspondentes, representados como números hexadecimais de 128 dígitos:

- $\mathcal{M}^1$: “Instituto de Física de São Carlos, Universidade de São Paulo-USP”
- $\mathcal{M}^2$: “Instituto de Física de São Carlos, Universidade de Sao Pulo-USP”
- $\mathcal{M}^3$: “Instituto de Física de São Carlos, Universidade de São Paulo-USP”
- $\mathcal{M}^4$: “Instituto de Física de São Carlos Universidade de São Paulo-USP”

$\text{Hash}(\mathcal{M}^1) =$	EB5604BE19C39BC8884CF2C6EEF5F3907C670EF67B668FE703A666F6F7158A45C CD1A9CC0577182BB2A3E3131AB5E606DCD5AD7EF6DEEED898453698ABCA776E
$\text{Hash}(\mathcal{M}^2) =$	819BC1781FC37576754BE267191897B1818AD583042B4C716A6DB90139B722C74 92116E1FEE62191EA94FBF4D3781D4123D5A41A28300DF7E81000E250D9141E
$\text{Hash}(\mathcal{M}^3) =$	16E71D76FD00159806B9732B93337B13C43837FE9CAA0C93688ECB3A720E38264C A90B65B35490FF5B0B137F7AEAB1AA63B3E8D8DC91138ADD5FA41C2E3C17CE
$\text{Hash}(\mathcal{M}^4) =$	EAFCFD6E1425483790BC703DA5E1F3064B4B2019048E15F6973DF038F81C3031F 9E9EB4FFAE318DD33D4CF525E8CB3838410C7FCEAF208EFD7DD53FCD542416

A diferencia entre os quatro *hashcodes* é notável, ao observar a mensagem original $\mathcal{M}^1$ e suas perturbações $\mathcal{M}^2$, $\mathcal{M}^3$, $\mathcal{M}^4$. Da mesma maneira, a Figura 5.2.3 mostra as sequências binárias geradas pelos quatro *hashcodes* da experiência anterior. O resultado desta simulação mostra uma boa qualidade do algoritmo proposto: uma perturbação mínima da mensagem pode causar grandes mudanças no valor *hash* final.

5.2.3.1 Teste de segurança

Um teste de segurança bastante aplicado nos algoritmos *hash* tradicionais encontrados na literatura (43,95), consiste em realizar uma série de análises estatísticos sobre um conjunto de experiências denotado por $\mathcal{E} = \{R^e \mid e = 1, \dots, N\}$, sendo R uma replica da mensagem original $\mathcal{M}$ com apenas um bit perturbado. De tal forma que $\mathcal{E}_e = d(H(\mathcal{M}), H(R^i))$

Figura 5.2.3 – *Hashcodes* das diferentes condições.

indique o número de bits perturbados, *i.e.*, a Equação (5.2.2) mede a distância entre dois *hashcodes*.

$$d(\vec{H}, \vec{H}') = \sum_{i=1}^N |H_i - H'_i| \quad (5.2.2)$$

As réplicas R^e são cópias idênticas da mensagem original $\mathcal{M}$, exceto o e -ésimo bit perturbado, tal que $R^e = \overline{\mathcal{M}_e}$ (complemento booleano) e, portanto, o número de experiências é correspondente ao comprimento da mensagem.

Estes testes foram executados para diversos comprimentos dependendo do número de blocos $b = 1, \dots, 10$. Foram calculados os *hashcodes* para todas as réplicas contidas em E e foi calculada a distância entre cada uma destas com o *hashcode* da mensagem original escolhida aleatoriamente. A partir destes resultados, foram calculados a distância média $\bar{\mathcal{E}}$ e desvio padrão σ , os valores mínimos e máximos de bits diferentes encontrados, e finalmente a probabilidade da variação de bits $p = \frac{\bar{\mathcal{E}} * 100}{N}$.

Em geral, um algoritmo de criptografia ideal (77) frente a qualquer perturbação mínima da condição inicial deve produzir uma saída significativamente diferente, com uma probabilidade de 50% de variação para cada bit de saída (efeito avalanche). Nos algoritmos de *hash*, o pior dos casos frente à mesma situação seria não existir nenhuma diferença, já que significaria uma colisão entre os *hashcodes* de duas mensagens com apenas um bit de diferença, o qual desqualificaria o algoritmo.

Na Tabela 5.2.1 são resumidos os parâmetros do teste de segurança ao aplicar o algoritmo de *Hash* caótico proposto para diferentes número de blocos. Por exemplo, no

caso de $b = 10$ blocos, a média indica que a distância entre dois *hashcodes* a partir de uma perturbação mínima inicial (1 bit de diferença) tem aproximadamente (255.223 ± 3.710) bits diferentes, com 199 e 318 bits de diferença no pior e melhor dos casos respectivamente, e que este mantém aproximadamente 49.8% de chances de bits diferentes entre as duas *hashcodes*. Nota-se também que neste conjunto de testes não existiu nenhum caso de colisão o qual é um ótimo indicador.

Tabela 5.2.1 – Análise estatística do número de diferenças E_e para diferente número de blocos.

blocos	Média	Dsv pad.	Mín.	Máx.	Probabilidade
$b = 1$	255.1	3.7	206	305	49.8%
$b = 2$	255.6	3.8	198	323	49.9%
$b = 3$	256.0	3.7	197	311	50.0%
$b = 4$	255.8	3.7	189	315	50.0%
$b = 5$	255.8	3.7	195	312	50.0%
$b = 6$	255.9	3.6	197	313	50.0%
$b = 7$	255.2	3.6	199	312	49.8%
$b = 8$	255.2	3.7	190	319	49.9%
$b = 9$	255.1	3.7	196	322	49.8%
$b = 10$	255.2	3.7	199	318	49.8%

5.2.4 Comparação com outros algoritmos *hash*

A Tabela 5.2.2 mostra uma comparação entre o método proposto e o algoritmo de *Hash* Whirlpool (37) (Seção 3.1.3). Ambos foram sujeito ao mesmo teste de segurança e usando $b = 10$ blocos nas mensagens.

Comparando ambos os algoritmos, percebe-se que o algoritmo de *Hash* caótico proposto possui resultados próximos ao Whirlpool, é dizer, possui resultados equivalentes. Esta aproximação reforça o potencial do algoritmo proposto e seu bom nível de segurança.

Tabela 5.2.2 – Comparação dos testes de segurança entre o algoritmo *hash* proposto e Whirlpool.

Análise estatístico sobre $\mathcal{E}$	Whirlpool	<i>Hash</i> caótico
Média	255.950	255.223
Desvio Padrão	2.970	3.710
Mínimo	216	199
Máximo	298	318
Probabilidade	49.990%	49.848%

5.2.4.1 Teste de velocidade

O desempenho do algoritmo é medido em termos de velocidade do processamento, ou seja, o tempo médio de execução do algoritmo em segundos dependendo do tamanho da mensagem de entrada.

Neste experimento foi calculada a velocidade média para 10 mensagens aleatórias com diferentes números de blocos de mensagem $b = \{1 \times 10^5, \dots, 10 \times 10^5\}$. A Figura 5.2.4 apresenta os resultados da velocidade média para os dois algoritmos Whirlpool e o *Hash* caótico proposto, os quais foram executados em um Intel(R) Core(TM) i7-2600 CPU @ 3.40GHz 8GB RAM. Pode ser observado que a velocidade do Whirlpool é até 10^3 vezes maior que a proposta deste trabalho.

Figura 5.2.4 – Comparação da velocidade por número de blocos calculados entre o Whirlpool e o algoritmo de *hash* caótico proposto.

5.2.5 Considerações gerais do algoritmo de *hash* proposto

Primeiramente, nota-se que o algoritmo de *Hash* caótico proposto possui um grande potencial. Isto se deve principalmente ao AC híbrido, a estratégia de combinar ACs caóticos e as redes complexas, que proporcionaram inúmeras vantagens na projeção e construção do algoritmo. Nota-se que foi projetado um algoritmo bastante flexível e com muitas opções de modificação de parâmetros que podem melhorar o seu desempenho, como por exemplo pode ser usado como um algoritmo *Hash* de chave secreta, e que a chave seja alterada dinamicamente.

Embora o algoritmo apresente um desempenho menor (relacionado ao de tempo de execução) quando comparado com algoritmos de *Hash* tradicionais (*e.g.*, Whirlpool) isto

não lhe tira importância uma vez que o mesmo possui uma segurança equivalente em outras qualidades como a segurança, resistência às colisões, flexibilidade e relativa simplicidade conceitual para a construção do algoritmo proposto. Por outro lado, o algoritmo também obedece aos estipulados de difusão e confusão do Shannon (77), uma vez que uma alteração na mensagem provoca uma grande mudança no *hashcode* resultante.

5.3 Considerações finais

Neste capítulo foram propostos dois algoritmos de criptografia baseados nos ACs caóticos. Na primeira parte foi proposto um algoritmo de cifra de bloco, os resultados obtidos indicam que o AC caótico da família *life-like* de regra B1357/S2468 possui alta sensibilidade às condições iniciais, alta complexidade e ótimas qualidades para ser utilizado como gerador de números aleatórios, e conseqüentemente um bom algoritmo de cifra de bloco. O qual foi avaliado pelos diversos testes de aleatoriedade.

Na segunda parte, foi proposto um algoritmo de *Hash* criptográfico, o qual foi submetido a diversos testes de segurança, resistência às colisões, e teste velocidade; os resultados demonstraram que a utilização dos AC híbrido (AC caótico e redes complexas) na construção do algoritmo possui um grande potencial em aplicações criptográficas reais.

6 *Proposta de método de Criptoanálise*

Na revisão bibliográfica do Capítulo 3, foram relatadas algumas tentativas para classificar as cifras de blocos por meio de: algoritmos de aprendizado de máquinas (algoritmos genéticos, redes neurais), técnicas de recuperação de informação, técnicas de reconhecimento de padrões, entre outros (39, 45, 57, 63, 78). Muitos destes testes foram realizadas sobre cifras de bloco com modos de operação ECB e CBC, sendo que a taxa de acerto decresce com outros modos, e além disso, foi reportado que o CBC têm sido resistente às tentativas de classificação. Deste modo surge a necessidade de métodos para classificar e analisar os modos de operação.

Esta proposta de exploração dos modos de operação é justificada por três motivos. Primeiro, pela importante atuação dos modos de operação durante a encriptação de dados em garantir a confidencialidade e segurança dos dados (46). Segundo, pela escassez de métodos para discriminar e/ou classificar dentre os diferentes modos de operação. E terceiro, pelo interesse de abordar a Criptoanálise e o reconhecimento de padrões como estudos análogos.

Neste capítulo são dados argumentos e propomos um método para a caracterização dos modos de operação ECB, CBC, OFB, CFB, CTR e PCBC. Este método é baseado no expoente de Lyapunov dos autômatos celulares (Capítulo 4) na avaliação da estabilidade dos modos de operação tratando-os como ACs unidimensionais, e partindo desta premissa, medir a taxa de divergência ou convergência a partir de duas condições iniciais próximas. Neste contexto, pode-se explorar as trajetórias de evolução e caracterizar os sistemas como instável, assintoticamente estável, ou conservador quando a separação inicial permanece (47).

6.1 Abordagem de autômato celular nos sistemas criptográficos

Seguindo os conceitos de Criptografia do Capítulo 3, o modo de operação pode ser visto como uma função $M_{E_K} : \{0, 1\}^N \rightarrow \{0, 1\}^N$ que mapeia o texto claro $\mathbf{P}$ de comprimento N , fragmentado em b blocos de n bits, no texto cifrado $\mathbf{C}$ de igual comprimento, tal que $\mathbf{C} = M_{E_K}(\mathbf{P})$.

É traçada uma equivalência entre o AC unidimensional $\langle \mathcal{T}, S, \phi \rangle$ e o sistema de criptografia $\langle \mathbf{C}, \mathbf{P}, M_{E_K} \rangle$. Dize-se que um sistema de criptografia pode ser emulado pelo AC com a equivalência de $\phi \equiv M_{E_K}$, assim o modo de operação M_{E_K} , é submetido como a função de transição. Então, a função de transição M_{E_K} pode ser aplicada de forma iterativa, fazendo que os novos texto cifrados, de cada iteração, $\mathbf{C}^t$ sejam recursivamente encriptados (evoluídos) no tempo t .

Logo, uma iteração na evolução do AC corresponde a uma iteração do modo de operação M_{E_K} , desde que essa iteração seja submetida a várias sub-iterações, uma para cada bloco encriptado, *i.e.*, uma iteração do modo de operação M_{E_K} implica b operações de E_K . Definidos na seguinte formulação, onde o texto claro $\mathbf{P}$ é a condição inicial:

$$\begin{aligned} \mathbf{C}^{t+1} &= M_{E_K}(\mathbf{C}^t), \\ \mathbf{C}^1 &= M_{E_K}(\mathbf{P}). \end{aligned} \tag{6.1.1}$$

6.2 Expoente de Lyapunov nos sistemas criptográficos

Para explorar a dinâmica e estabilidade do AC e seu equivalente sistema criptográfico, emprega-se o expoente de Lyapunov exposto no Capítulo 4. Nesse contexto, será avaliada a taxa de divergência, a longo prazo, dos modos de operação quando evoluídos a partir de dois textos claros $\mathbf{P}$ e $\mathbf{P}^*$ de distância muito próxima.

Nessa situação, considera-se que a perturbação de um bit no primeiro bloco de texto claro P_1 é a distância mais próxima possível, de tal forma que o **vetor de defeitos** $h(c_i^j, t)$, o vetor que indica quais são os bits c_i^j (células) perturbados a cada iteração t ,

onde $s(c_i^j, 0)$ é o estado do i -ésimo bit do j -ésimo bloco do texto claro e que satisfaz:

$$s(c_i^j, t + 1) = M_{E_K}(s(c_i^j, t)). \quad (6.2.1)$$

O estado inicial do texto claro $\mathbf{P}$ é representado por s_0 e sua versão perturbada $s_0^* = \mathbf{P}^*$ satisfazem que $\sum_{c_i^j} h(c_i^j, 0) = 1$, o que é equivalente a que o número de bits diferentes (defeitos) $\epsilon_0 = 1$. Logo, na primeira iteração $\epsilon_1 = \sum_{j=1}^b \sum_{i=1}^n h(c_i^j, 1)$, e em que o vetor de defeitos é calculado assim:

$$h(c_i^j, 1) = \begin{cases} 1 & \text{se } s^*(c_i^j, 1) \neq s(c_i^j, 1), \\ 0, & \text{caso contrário.} \end{cases} \quad (6.2.2)$$

O algoritmo do expoente de Lyapunov (usado nos ACs) foi adaptado para estimar o número total de bits perturbados ϵ_t que foram originadas a partir de duas configurações iniciais s_0 e s_0^* , e deve ser calculado de acordo com os passos dados na Tabela 6.2.1.

Após o cálculo do número de defeitos ϵ_t em cada passo de tempo consecutivo t , o máximo expoente de Lyapunov MLE $\lambda(t)$ é definido como o número total de bits perturbados a cada iteração:

$$\lambda(t) = \frac{1}{t} \log \left(\frac{\epsilon_t}{\epsilon_0} \right). \quad (6.2.4)$$

6.3 Resultados

6.3.0.1 Cifras de bloco consideradas

Para realizar os testes empregando o método proposto, foram considerados duas famílias de cifras de bloco agrupadas pelo comprimento de bloco n . As cifras de blocos empregadas são as mais conhecidas e disponíveis na literatura são mostradas na Tabela 6.3.1. Estas cifras de blocos foram implementadas com o número de *rounds*^a dependendo do algoritmo (60).

Nota-se que, as especificações dos algoritmos das cifras de blocos e os modos de operação mencionados são públicas, para que possam ser implementadas (46). Além disso, estes 10 algoritmos foram adaptados de tal forma que recebem como entrada o texto claro $\mathbf{P}$, a

^a O *round* consiste de várias etapas internas que ocorrem durante o processo de encriptação/decriptação.

Tabela 6.2.1 – Algoritmo do expoente de Lyapunov do modo de operação

$t = 0$	<ul style="list-style-type: none"> Considerar s_0 e s^*0, tal que $\epsilon_1 = 1$. Armazenar s_0^* no conjunto A_0.
$t = 1$	<ul style="list-style-type: none"> Iterar por primeira vez o sistema criptográfico, <i>i.e.</i>, $\mathbf{C}^1 = M_{E_K}(\mathbf{P})$ e analogamente a versão perturbada $\mathbf{C}^{1*} = M_{E_K}(\mathbf{P}^*)$, ambos usando o mesmo vetor de inicialização γ, e calcular $s(c_i, 1)$ e $s^*(c_i^j, 1)$. Calcular o vetor de defeitos h_1 (Equação (6.2.2)) e atribuir ao vetor de inicialização com o último bloco calculado para ambas versões, <i>i.e.</i>, $\gamma = C_b^1$ e $\gamma^* = C_b^{1*}$. Para todo c_i^j, tal que $h(c_i^j, 1) = 1$. Criar uma replica R_i^j tal que $R_i^j(c_i^j, 1) = \bar{s}(c_i^j, 1)$ (complemento booleano), de tal forma que $R_i^j(c_q^j, 1) = s(c_q^j, 1)$, para todo $c_q^j \neq c_i^j$. Armazenar estas réplicas no conjunto A_1.
$t = 2$	<ul style="list-style-type: none"> Iterar o sistema criptográfico, <i>i.e.</i>, calcular $s(c_i^j, 2)$ e $R(c_i^j, 2)$ que resume-se a avaliar $M_{E_K}(R_i^j)$ com o mesmo γ^* para todas as réplicas $R \in A_1$. Calcular o vetor de defeitos e calcular o total de número de defeitos em t : $\epsilon_t = \sum_{R_i^j \in A_1} \sum_{j=1}^b \sum_{i=1}^n h(c_i^j, t) \quad (6.2.3)$ e atribuir ao vetor de inicialização o último bloco calculado de ambas versões, <i>i.e.</i>, $\gamma = C_b^2$ e $\gamma^* = R_{i_b}^j$. Para todo $c_i^j \wedge R \in A_1$, para os quais $R(c_i^j, 2) \neq s(c_i^j, 2)$. Criar uma replica R_i^j tal que $R_i^j(c_i^j, 2) = \bar{s}(c_i^j, 2)$, de tal forma que $R_i^j(c_q^j, 2) = s(c_q^j, 2)$, para todo $c_q^j \neq c_i^j$. Armazenar estas réplicas no multiconjunto $A_2 = \{(R_i^j, h(c_i^j, 2)) R \in A_1 \wedge R_i^j(c_i^j, 2) \neq s(c_i^j, 2)\}$
$t + 1$	<ul style="list-style-type: none"> Repetir o passo anterior para cada $t+1$. Nota-se que cada novo multiconjunto A_{t+1} é construído em base as réplicas em A_t.

chave K e o vetor de inicialização γ . Por outro lado, os 6 modos de operação: ECB, CBC, OFB, CFB, CTR e PCBC foram implementados com base nas Equações (3.1.3)-(3.1.8).

6.3.0.2 Conjunto de dados

Foi criado um conjunto de dados $\mathcal{E} = \{e\mathbf{P} \mid e = 1, \dots, 200\}$ contendo 200 textos claros $e\mathbf{P}$ e suas correspondentes versões perturbadas $e\mathbf{P}^*$ obtidas alternando apenas um bit do primeiro bloco do correspondente $e\mathbf{P}$, assim como 200 chaves K e vetores de inicialização γ , todos estes gerados aleatoriamente. Assim, no total são 60 conjuntos de dados $\mathcal{E}$, um por cada um dos sistemas criptográficos formados pelas combinações entre as 10 cifras de bloco e os 6 modos de operação. Além disso, todos os textos claros de $\mathcal{E}$ são de comprimento $N = b \times 64$ para a FAMILIA-64 e $N = b \times 128$ para a FAMILIA-128 e as chaves e vetor de inicialização são do tamanho especificado por cada cifra de bloco.

Tabela 6.3.1 – Especificações das duas famílias de cifras de blocos consideradas.

FAMILIA-64 $n = 64$			FAMILIA-128 $n = 128$		
Cifra	<i>rounds</i>	chave k	Cifra	<i>rounds</i>	chave k
DES (65)	16	56	AES (66)	10	128
IDEA (30)	8	64	Twofish (74)	16	128
TEA (86)	32	64	RC6 (6)	20	128
XTEA (68)	32	64	Seed (55)	16	128
			Camellia (28)	18	128
			Serpent (27)	32	128

6.3.1 Expoente de Lyapunov para os modos de operação

Nota-se que os LEs descritos no decorrer deste capítulo, representam as médias calculadas sobre cada conjunto de dados, e portanto λ refere-se à média calculada. Além disso, estes valores foram normalizados em base ao limite superior do LE da Equação (6.3.1) (detalhado no Apêndice A) $\lambda_m = \log(64 \times 5) \approx 5.77$ para a FAMILIA-64, e $\lambda_m = \log(128 \times 5) \approx 6.46$ para a FAMILIA-128, a fim de comparar os LEs de ambas famílias.

$$\lambda_m = \frac{1}{t} \log(\lambda^t) = \log(N) = \log(b \times n). \quad (6.3.1)$$

Neste experimento, foi executado o algoritmo para estimar os LEs para os 60 sistemas criptográficos em questão. Empregaram-se $t = 200$ iterações, sendo o tempo suficiente para que os $\lambda(t)$ obtidos convergissem, tal que $|\lambda(t) - \lambda(t+1)| < 1.19 \times 10^{-4}$ foi a máxima separação atingida entre $\lambda(t)$ e $\lambda(t+1)$ no final da execução. Além disso, para demonstrar a consistência dos valores de λ para todos os membros da família 64 e 128, foi calculado o desvio padrão σ_λ do MLE λ_{200} . Na Tabela 6.3.2 encontram-se resumidos os MLE normalizados e seus correspondentes desvios padrão calculados ao final da execução.

A Figura 6.3.1 (a) mostra a distribuição de frequências do desvio padrão σ_λ do MLE λ_{200} dos 60 sistemas criptográficos. Claramente, este histograma mostra a alta concentração dos valores de λ em torno da média, o que é validado pelo pequeno desvio padrão em cada caso. Por outro lado, a Figura 6.3.1 (b) mostra o histograma das diferenças entre o máximo e o mínimo MLE observado entre os membros de $\mathcal{E}$, denotado por $\Delta = \max(\{e\mathbf{P} \mid e = 1, \dots, 200\}) - \min(\{e\mathbf{P} \mid e = 1, \dots, 200\})$. Todos os resultados obtidos atingem $3.99 \times 10^{-5} < \Delta < 4.79 \times 10^{-3}$, em que o máximo e o mínimo Δ observados correspondem, respectivamente, aos sistemas criptográficos IDEA-OFB e Serpent-CFB.

Tabela 6.3.2 – MLE λ_{200} e desvio padrão ($\times 10^{-3}$) dos modos de operação das FAMILIA-64 e 128.

λ_{200}	IDEA	DES	TEA	XTEA	AES
ECB	0.6008 ± 0.2	0.6008 ± 0.2	0.6008 ± 0.2	0.6008 ± 0.2	0.6436 ± 0.1
CBC	0.6165 ± 0.2	0.6165 ± 0.2	0.6165 ± 0.1	0.6165 ± 0.2	0.6576 ± 0.1
CFB	0.0276 ± 0.0	0.0276 ± 0.0	0.0276 ± 0.0	0.0276 ± 0.0	0.0268 ± 0.0
OFB	0.8751 ± 0.7	0.8751 ± 0.7	0.8752 ± 0.7	0.8752 ± 0.6	0.8882 ± 0.4
CTR	0.8752 ± 0.6	0.8752 ± 0.7	0.8752 ± 0.7	0.8751 ± 0.7	0.8882 ± 0.4
PCBC	0.8798 ± 0.3	0.8798 ± 0.4	0.8798 ± 0.3	0.8797 ± 0.3	0.8927 ± 0.2

λ_{200}	RC6	Twofish	Seed	Serpent	Camellia
ECB	0.6436 ± 0.1	0.6436 ± 0.1	0.6436 ± 0.1	0.6436 ± 0.1	0.6436 ± 0.1
CBC	0.6576 ± 0.1	0.6576 ± 0.1	0.6576 ± 0.1	0.6576 ± 0.1	0.6576 ± 0.1
CFB	0.0268 ± 0.0	0.0268 ± 0.0	0.0268 ± 0.0	0.0268 ± 0.0	0.0268 ± 0.0
OFB	0.8882 ± 0.5	0.8881 ± 0.4	0.8881 ± 0.4	0.8881 ± 0.4	0.8882 ± 0.5
CTR	0.8882 ± 0.4	0.8882 ± 0.4	0.8882 ± 0.4	0.8881 ± 0.4	0.8881 ± 0.4
PCBC	0.8927 ± 0.2	0.8927 ± 0.2	0.8927 ± 0.2	0.8927 ± 0.2	0.8927 ± 0.2

Estas pequenas diferenças demonstram que os valores calculados de λ para os diferentes textos claros do conjunto de dados $\mathcal{E}$ são consistentes.

Figura 6.3.1 – (a) Distribuição das frequências do logaritmo base 10 do desvio padrão σ_λ para MLE λ_{200} calculado para as 60 combinações de sistemas criptográficos. (b) Distribuição de frequências $\Delta = \max(\{e\mathbf{P} \mid e = 1, \dots, 200\}) - \min(\{e\mathbf{P} \mid e = 1, \dots, 200\})$.

Para simplificar, limita-se a atenção nas cifras de bloco IDEA e AES como membros representantes das famílias de 64 e 128, respectivamente, já que foram obtidos resultados semelhantes para o restante de cifras de bloco consideradas. Os resultados são mostrados na Figura 6.3.2 e na Figura 6.3.3. Os gráficos mostram as curvas normalizadas dos LE obtidos durante cada iteração de tempo t por cada um dos modos de operação. Assim, cada curva representa a média do LE e desvio padrão, este último é muito pequeno e é apenas visível para as curvas ECB e PCBC.

Figura 6.3.2 – Gráfico do expoente de Lyapunov em função do número de iterações para os diferentes modos de operação: ECB, CBC, OFB, CFB, CTR e PCBC, usando as cifras de blocos (a) IDEA de 64 bits e (b) AES de 128 bits. As curvas representam a média e desvio padrão do LE λ estimado para 200 textos claros $\mathbf{P}$ durante $t = 200$ iterações, as quais foram normalizadas com respeito do limite superior do LE λ_m obtido com a Equação. (6.3.1). Nota-se que a linha traçada que corta ambos os gráficos é apenas referencial.

Figura 6.3.3 – Gráfico do expoente de Lyapunov em função do número de iterações para os diferentes modos de operação: ECB, CBC, OFB, CFB, CTR e PCBC, usando as cifras de blocos da FAMILIA-64: DES, TEA, XTEA e da FAMILIA-128: RC6, Twofish, Seed, Serpent e Camellia. As curvas representam a média e desvio padrão do LE λ estimado para 200 textos claros $\mathbf{P}$ durante $t = 200$ iterações, as quais foram normalizadas com respeito ao limite superior do LE λ_m obtido com a Equação. (6.3.1).

A partir destes gráficos, visualmente se pode discriminar entre os modos de operação, exceto para as curvas de OFB e CTR que se sobrepõem. Além disso, pode ser observado que os correspondentes LE para OFB e CTR aumentam seguindo um padrão exponencial durante as primeiras iterações. Este comportamento pode ser explicado ao relembrar as Equações (3.1.5) e (3.1.7), em que o CTR utiliza um gerador de números pseudo-aleatórios interno, enquanto que o OFB na primeira iteração re-usa o resultado da encriptação do vetor de inicialização para as seguintes iterações, de tal forma que o número de bits diferentes (defeitos) aumenta não somente pelas diferenças emergentes entre os textos claros, mas também devido aos defeitos adicionais introduzidos por meio desses processos internos aleatórios. Em contraste, o CFB e CBC tem o efeito oposto, ou seja, a curva do CFB decai seguindo um padrão exponencial.

Os expoentes Lyapunov do ECB e PCBC exibem duas curvas que mantêm-se mais ou menos constante durante o tempo, de fato, o número de defeitos ϵ cresce proporcionalmente ao longo do tempo. Isto pode ser explicado considerando que, por exemplo, o IDEA-ECB atinge $\lambda_{200} = 3.46565 \pm 1.35 \times 10^{-3}$, o que significa que o número de defeitos $\epsilon_t = e^{(\lambda_{200})} = 31.99728 \pm 4.32 \times 10^{-2}$ é aproximadamente $n/2$, o que representa 50% do comprimento do bloco. Este mesmo comportamento ocorre para a cifra de bloco AES-ECB, onde $e^{(\lambda_{200})} = 63.994055 \pm 4.29 \times 10^{-2}$.

Além disso, foi realizado o teste t de *student* a um nível de significância de 5%, cujos resultados revelam que existe diferenças estatisticamente significantes entre ECB, CBC, CFB, CTR e PCBC independentemente da cifra de bloco, o qual indica que também pode-se discriminar entre as duas famílias de 64 e 128 bits, por exemplo, pode-se distinguir entre CBC-IDEA, CBC-AES, CFB-TEA, CFB-Camellia. Por outro lado, não foi encontrada diferença estatisticamente significativa entre CTR e OFB. Além disso, não há diferença estatisticamente significativa para distinguir amostras emparelhadas de sistemas criptográficos de 64 e 128 bits, por exemplo, o método não distingue entre CBC-IDEA de CBC-DES, ou entre CFB-TEA de CFB-XTEA e assim por diante.

Os testes realizados acima foram executados em um Intel(R) Core(TM) i7-2600 CPU @ 3.40GHz 8GB RAM. Assim, foi medido o tempo de execução do método proposto para analisar seu desempenho. A Figura 6.3.4 mostra os tempos em segundos para a execução do LE para os 200 textos claros do conjunto de dados $\mathcal{E}$, para cada modo de operação com respeito à cifra de bloco empregada. Observa-se que na média, o modo CTR demora 12.905s para estimar o LE de um texto claro de $N = 640$ bits, a comparação do menor tempo de execução do ECB 1.071s.

Figura 6.3.4 – Tempo de execução em segundos para estimar o LE dos modos de operação com diferentes cifras de bloco da FAMILIA-64 (acima) e FAMILIA-128 (abaixo). O LE λ foi estimado para 200 textos claros $\epsilon \mathbf{P}$ durante $t = 200$ iterações.

6.3.2 Influência do número de blocos

Com o propósito de avaliar a influência do número de blocos b no cálculo do LE, foi realizado um experimento para estimar o LE dos sistemas de criptografia de acordo ao número de blocos b , organizados da mesma forma que nos resultados anteriores. Na Figura 6.3.5 são mostradas as curvas de LE para cada modo de operação usando apenas a cifra de bloco IDEA aplicando o método para o mesmo conjunto de dados $\mathcal{E}$ para diferentes parâmetros $b = 2, 4, 8, \dots, 20$.

Ao observar os gráficos de cada modo de operação fica evidente a existência de um padrão de crescimento em quase todas as curvas com respeito a b . No caso excepcional do ECB, diz-se que o LE é independente do número de blocos b do texto claro. Isto é devido a que, ao calcular o LE, o número de defeitos ϵ do ECB atinge aproximadamente $n/2$, o que demonstra que o primeiro bloco do texto claro das condições iniciais é o único

que influência nos subsequentes defeitos a serem calculados, e que o resto de blocos não afetam este cálculo, portanto não espalhariam defeitos no texto cifrado. Assim, nota-se que o número de blocos é influente no cálculo do LE para os 5 modos de operação: CBC, OFB, CFB, CTR e PCBC. Além disso, este comportamento reforça ao método proposto, pois foi observado para ambas FAMILIA-64 e FAMILIA-128.

Figura 6.3.5 – Comparação do LE para diferentes modos de operação, usando diferentes números de blocos $b = 2, 4, 8, \dots, 20$ com a cifra de bloco IDEA. Cada gráfico mostra as curvas do λ durante $t = 200$ iterações.

6.3.3 Análises dos resultados

Condições iniciais $\mathbf{P}$ e $\mathbf{P}^*$: Lembrando que o LE é medido a partir de dois textos claros $\mathbf{P}$ e $\mathbf{P}^*$ de distância muito próxima. Nesse contexto, a menor perturbação possível implicaria perturbar o bit menos significativo (o bit mais à direita) do texto claro. No entanto, em base aos resultados, observou-se que a posição do bit perturbado não afeta o valor numérico do LE. Portanto, não importa a posição do bit perturbado na condição inicial, assim que for apenas um bit.

Padrões na Criptografia: A partir das Figuras 6.3.2–6.3.3 pode ressaltar-se que nenhum dos λ atingiram o melhor dos casos, *i.e.*, $\lambda = 1$, o qual representa um cenário ideal para os sistemas de criptografia, e é notável que as cifras de blocos consideradas neste estudo possuem limitações. Além disso, inesperadamente, o método proposto consegue

diferenciar entre cifras de blocos da FAMILIA-64 e FAMILIA 128, o que implica a presença de características e padrões nestes sistemas de criptografia.

Limitações no tempo de execução: Embora o algoritmo foi projetado com diversas estratégias computacionais para evitar duplicar réplicas que são similares durante o cálculo do LE, detalhadas no Apêndice A, o custo computacional do método proposto é ainda muito elevado. Uma vez que o tempo de processamento aumenta drasticamente conforme o comprimento N do texto claro $\mathbf{P}$ aumenta. Neste sentido, acredita-se que a otimização do algoritmo é importante para trabalhos futuros.

6.4 Considerações finais

No presente capítulo foi proposto um método baseado no expoente de Lyapunov dos autômatos celulares, para avaliar a estabilidade dos modos de operação. Assim, foi descrita uma abordagem que considera um sistema criptográfico como um equivalente AC unidimensional. O método proposto foi capaz de distinguir dentre seis modos de operação: ECB, CBC, OFB, CFB, CTR e PCBC; usando duas famílias de blocos de cifras DES, IDEA, TEA e XTEA de 64 bits, bem como AES, RC6, Twofish, Seed, Serpent e Camellia de 128 bits. Os resultados mostraram que o padrão de evolução do expoente de Lyapunov é mantido para cada um dos seis modos de operação e que este é independente da cifra de bloco, o qual foi corroborado pelo teste t de *Student*.

Além disso, neste trabalho foi fornecido a base matemática para obter o limite superior teórico do expoente de Lyapunov λ_m para os sistemas criptográficos, no Apêndice A. Embora, deve ser salientado que, ainda existem muitas limitações em uma análise do limite superior do LE para cada um dos modos de operação. Assim, este trabalho significa um dos primeiros arcabouços de uma alternativa para a Criptoanálise a partir da existência de padrões nos sistemas criptográficos, pelas quais podem ser exploradas fraquezas e até conseguir classificar entre cifras de blocos.

7 Conclusões

Ao longo deste documento foi apresentada a alta aplicabilidade dos autômatos celulares, devido à sua natureza multidisciplinar e à simplicidade com que conseguem modelar sistemas complexos. Este trabalho focou principalmente no estudo do comportamento caótico dos ACs, com características intrínsecas (*e.g.*, ergodicidade, sensibilidade às condições iniciais, entropia, dinâmica determinista) que são homólogas as da Criptografia (difusão e confusão), comprovando sua utilização não somente apropriada na Criptografia, mas também na Criptoanálise.

As principais contribuições desta dissertação de mestrado são sintetizadas a seguir:

- **Algoritmo de cifra de bloco caótico:** Foi projetado e desenvolvido um algoritmo de cifra de bloco caótico baseado na apropriada seleção dos ACs com alto regime caótico. Para tanto, foram propostos vários critérios para a seleção de ACs, que são utilizados para examinar as diversas regras da família de 2D-AC *life-like*, por meio de medidas como a entropia, a distância de Hamming e o expoente de Lyapunov, sendo a regra B1357/S2468 uma excelente escolha para a geração de números pseudo-aleatórios. Assim, durante a construção do sistema criptográfico proposto, foi necessário garantir a boa qualidade do PRNG por meio das baterias de testes ENT e DIEHARD, que ofereceram resultados vantajosos quando comparados com outros algoritmos da literatura. Descrito no Capítulo 5.
- **Algoritmo de *hash* criptográfico:** foi elaborado a partir de um modelo híbrido que emprega ao AC caótico (selecionado anteriormente) e as redes complexas como tesselação do AC. Esta estratégia permitiu projetar uma função *hash* com bom desempenho, segurança e resistência às colisões. Embora este algoritmo de *hash* mostrou um desempenho menor (relacionado ao tempo de execução) em comparação ao algoritmo padrão internacional Whirlpool, mas o algoritmo proposto se destaca pela flexibilidade em termos de implementação. Descrito no Capítulo 5.
- **Método de criptoanálise:** foi proposto um método de criptoanálise baseada na

adaptação do expoente de Lyapunov dos ACs. Foram explorados os comportamentos dinâmicos dos modos de operação criptográficos, por meio do qual, encontraram-se padrões característicos que permitiram uma comparação quantitativa. Esses resultados representam ainda um avanço para a Criptoanálise, porém evidenciando a relevância desta, pois não existem referências na literatura que tenham conseguido discriminar dentre os modos: ECB, CBC, OFB, CFB, CTR e PCBC. Mais em particular, poder-se-ia detectar possíveis vulnerabilidades das cifras, já que a qualidade depende de valores e classificações numéricas. Finalmente os resultados obtidos mostram ao expoente de Lyapunov como uma ferramenta potencial de criptoanálise. Descrito no Capítulo 6.

Em conclusão, foi corroborada a forte relação entre a teoria do caos e a Criptografia e consequentemente a Criptoanálise. Este trabalho de mestrado aproveitou as virtudes dos ACs caóticos, contribuindo com um método inovador à crescente demanda por algoritmos criptográficos que possam oferecer maior segurança da informação. Além disso, foi reforçada a abordagem de reconhecimento de padrões como estudo análogo à Criptoanálise, visando um primeiro arcabouço para posteriores estudos.

7.1 Questões para trabalhos futuros

Uma das principais dificuldades ao trabalhar com sistemas de criptografia é a falta de testes de segurança rigorosos que possam ser empregados como indicadores qualitativos de um sistema com uma análise definitiva. A maioria dos testes amplamente utilizados e aceitos na literatura são de natureza empírica (59,84), significando um enorme vazio para a Criptografia, pois existem muitos algoritmos criptográficos que foram quebrados e, que passaram nestes testes.

Outro ponto diz respeito ao tamanho das sequências de números para realizar um teste de aleatoriedade. Qual é a diferença entre avaliar 10MB ou 100TB de uma sequência de números pseudo-aleatórios? Qual seria um tamanho ideal? Este é, aparentemente, um problema da teoria da complexidade que aguarda solução.

Na teoria do caos, uma questão pouco abordada é o erro de arredondamento na aritmética de ponto fixo (vírgula fixa). A precisão é uma questão muito importante ao implementar um algoritmo para alguma arquitetura de computador específica, pois, um sistema criptográfico baseado em caos é fortemente dependente às condições iniciais do sistema e, por consequência, uma mínima variação poderia mudar todos os resultados.

Neste trabalho foi usado uma aritmética de ponto fixo de aproximadamente 308 casas decimais, o que permite ao algoritmo ser utilizado em diferentes plataformas computacionais. Na literatura não se observa esta preocupação ao analisar os algoritmos de criptografia caóticos propostos.

O alto custo computacional foi uma das grandes limitações durante a fase experimental do método de criptoanálise. Pois, mesmo usando otimizações no cálculo do expoente de Lyapunov, o custo computacional aumenta em razão do tamanho do texto claro. Estes algoritmos podem ser otimizados ou paralelizados, sendo uma sugestão para futuros trabalhos.

7.2 Publicações

Este mestrado gerou três artigos, uma delas numa revista internacional e as outras em conferencia internacional conforme o Apêndice B. Outros dois trabalhos (cujos títulos estão sujeitos a alterações) devem ainda ser publicados como fruto desta dissertação, sendo que um esta em fase de redação, enquanto o outro aguardando submissão em periódico de excelência da área.

- Publicados:

1. MACHICAO, J.; MARCO, A.; BRUNO, O. M. Chaotic encryption method based on life-like cellular automata. *Expert Systems with Applications*, v. 39, p. 12626–12635, 2012.
2. MACHICAO, J.; MARCO, A. G.; BRUNO, O. M. Lyapunov exponent: A qualitative ranking of block cipher modes of operation. In: GILBERT T.; KIRKILIONIS M.; NICOLIS G. (Ed.) *EUROPEAN CONFERENCE ON COMPLEX SYSTEMS, Springer Proceedings in Complexity, Springer International Publishing, ECCS'12*, 2012, Brussels. *Proceedings...* Brussels, Belgium, 2012. p. 979–985.
3. MACHICAO, J.; MARCO, A. G.; BRUNO, O. M. Lyapunov exponent: towards randomness measure. In: *Complex behavior in discrete dynamical systems (Satellite), ECCS'12*, 2012, Brussels. *Proceedings...* Brussels, Belgium, 2012. p. 15–18.

- Aguardando submissão:

1. MACHICAO, J.; BAETENS, J. M.; MARCO, A. G.; DE BAETS, B.; BRUNO, O. M. A dynamical systems approach to the discrimination of the modes of operation of cryptographic systems.

- Em Redação:

1. MACHICAO, J.; BRUNO, O. M. Cryptographic hash function based on hybrid chaotic cellular automata.

Referências

- 1 DEVANEY, R. *An introduction to chaotic dynamical systems*. Boulder, USA: Addison-Wesley, 1989. 50 p. (Advanced book program).

- 2 NEUMANN, J. V.; BURKS, A. W. *Theory of self-reproducing automata*. Champaign, USA: University of Illinois Press, 1966.

- 3 RIETMAN, E. *Exploring the geometry of nature: computer modeling of chaos, fractals, cellular automata, and neural networks*. Mountain, USA: Windcrest, 1989.

- 4 COOMBS, S. *The geometry and pigmentation of seashells*. Nottingham, UK: Department of Mathematical Sciences, University of Nottingham, 2009. p. 3–4.

- 5 GARDNER, M. The fantastic combinations of John Conway’s new solitaire game “life”. *Scientific American*, v. 223, p. 120–123, 1970.

- 6 RIVEST, R.; ROBSHAW, M.; SIDNEY, R.; YIN, Y. L. The RC6 block cipher. *RSA Laboratories and M.I.T. Laboratory for Computer Science*, Version 1.1, p. 1–3, 1998.

- 7 KYOOMARSI, F.; TORKESTANI, J.; DEHKORDY, P.; PEYRAVI, M. Using chemical cellular automata in simulation of chemical materials. In: *INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING RESEARCH, MANAGEMENT & APPLICATIONS*. 5, ACIS, 2007. Washington. *Proceedings...* Washington, USA: IEEE Computer Society, 2007. p. 769–773. (SERA '07).

- 8 MAERIVOET, S.; MOOR, B. D. Cellular automata models of road traffic. *Physics Reports*, v. 419, n. 1, p. 1–64, 2005.

- 9 WEI, Y.-M.; YING, S.-J.; FAN, Y.; WANG, B.-H. The cellular automaton model of investment behavior in the stock market. *Physica A: statistical mechanics and its applications*, v. 325, n. 3–4, p. 507–516, 2003.

- 10 ROSIN, P. L. Training cellular automata for image processing. *IEEE Transactions on Image Processing*, v. 15, n. 7, p. 2076–2087, 2006.

- 11 GANGULY, N.; MAJI, P.; SIKDAR, B.; CHAUDHURI, P. C. P. Design and characterization of cellular automata based associative memory for pattern recognition. *IEEE Transactions on Systems, Man, and Cybernetics Part B: cybernetics*, v. 34, n. 1, p. 672–678, 2004.
- 12 WOLFRAM, S. Cryptography with cellular automata. In: PICHLER, F. (Ed.) *Advances in cryptology – CRYPTO '85*. Santa Barbara, USA: Springer-Verlag, 1986. p. 429–432. (Lecture notes in computer sciences, v. 218).
- 13 SINGH, S. *The code book: the evolution of secrecy from Mary, queen of Scots, to quantum cryptography*. New York, USA: Doubleday, 1999.
- 14 NANDI, S.; KAR, B. K.; CHAUDHURI, P. P. Theory and applications of cellular automata in cryptography. *IEEE Transactions on Computers*, v. 43, p. 1346–1357, 1994.
- 15 SHIN, S.-H.; YOO, K.-Y. An efficient PRNG based on the hybrid between one- and two-dimensional cellular automata. In: *INTERNATIONAL CONFERENCE ON INFORMATION TECHNOLOGY: NEW GENERATIONS, ITNG '09*, 2009, Las Vegas. *Proceedings...* Nevada, USA: IEEE Computer Society, 2009. p. 498–503.
- 16 GUAN, S. U.; TAN, S. K. Pseudorandom number generation with self-programmable cellular automata. *IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems*, v. 23, n. 7, p. 1095–1101, 2004.
- 17 TOMASSINI, M.; SIPPER, M.; PERRENOUD, M. On the generation of high-quality random numbers by two-dimensional cellular automata. *IEEE Transactions on Computers*, v. 49, n. 10, p. 1146–1151, 2000.
- 18 TOMASSINI, M.; PERRENOUD, M. Stream cyphers with one and two-dimensional cellular automata. In: *INTERNATIONAL CONFERENCE ON PARALLEL PROBLEM SOLVING FROM NATURE, 6*, 2000, Paris. *Proceedings...* Paris: Springer-Verlag, 2000. p. 722–731.
- 19 SEREDYNSKI, F.; BOUVRY, P.; ZOMAYA, A. Y. Cellular automata computations and secret key cryptography. *Parallel Computing*, v. 30, n. 5–6, p. 753–766, 2004.
- 20 MARTIN, B.; SOLÉ, P. *Pseudorandom sequences generated by cellular automata*. 2008. Disponível em: <<http://arxiv.org/pdf/0807.3865.pdf>>. Acesso em: 15.06.2013.
- 21 SETH, A.; BANDYOPADHYAY, S.; MAULIK, U. Pseudorandom pattern generation by a 4-neighborhood cellular automata based on a probabilistic analysis. In: *INTERNATIONAL MULTICONFERENCE OF ENGINEERS AND COMPUTER*

SCIENTISTS, IMECS, 2008, Hong Kong. *Proceedings...* Hong Kong, China: Newswood Limited, 2008. v. 2.

22 SHIN, S.-H.; PARK, G.-D.; YOO, K.-Y. A virtual three-dimension cellular automata pseudorandom number generator based on the Moore neighborhood method. In: *INTERNATIONAL CONFERENCE ON INTELLIGENT COMPUTING*. 2008, Shanghai. *Proceedings...* China: Springer-Verlag, 2008. p. 174–181.

23 ADAMATZKY, A. *Game of life cellular automata*. New York, USA: Springer Publishing Company, 2010.

24 AMIGÓ, J.; KOCAREV, L.; SZCZEPANSKI, J. Discrete Lyapunov exponent and resistance to differential cryptanalysis. *IEEE Transactions on Circuits and Systems II*, v. 54, n. 10, p. 882–886, 2007.

25 ÁLVAREZ, G.; LI, S. Some basic cryptographic requirements for chaos-based cryptosystems. *International Journal of Bifurcation and Chaos*, v. 16, n. 8, p. 2129–2151, 2006.

26 AMIGÓ, J.; KOCAREV, L.; SZCZEPANSKI, J. On some properties of the discrete Lyapunov exponent. *Physics Letters. A*, v. 372, n. 41, p. 6265–6268, 2008.

27 ANDERSON, R.; BIHAM, E.; KNUDSEN, L. *A candidate block cipher for the advanced encryption standard*. 1998. Disponível em: <<http://www.cl.cam.ac.uk/~rja14/serpent.html>>. Acesso em: 07.03.2011.

28 AOKI, K.; ICHIKAWA, T.; KANDA, M.; MATSUI, M.; MORIAI, S.; NAKAJIMA, J.; TOKITA, T. Camellia: a 128-bit block cipher suitable for multiple platforms - design and analysis. In: *INTERNATIONAL WORKSHOP ON SELECTED AREAS IN CRYPTOGRAPHY*, 7, 2000. London. *Proceedings...* London, UK: Springer-Verlag, 2000. p. 39–56. (SAC '00).

29 ARROYO, D. *Framework for the analysis and design of encryption strategies based on discrete-time chaotic dynamical systems*. 164 p. Tese (Doutorado) — Universidad Politécnica de Madrid, Madrid, 2009.

30 ASCON TECH AG.; LAI X.; MASSEY, J. L. *Device for the conversion of a digital block and use of same*. US5214703 A, 25 May 1993.

31 BAETENS, J. M.; DE BAETS, B. Phenomenological study of irregular cellular automata based on Lyapunov exponents and Jacobians. *Chaos*, v. 20, n. 3, p. 033112, 2010.

- 32 COURBAGE, M.; KAMINSKI, B. Space-time directional Lyapunov exponents for cellular automata. *Journal of Statistical Physics*, v. 124, n. 6, p. 1499–1509, 2006.
- 33 LANGTON, C. Computation at the edge of chaos. *Physica D*, v. 42, p. 12–37, 1990.
- 34 BAETENS, J. M.; DE BAETS, B. On the topological sensitivity of cellular automata. *Chaos*, v. 21, n. 2, p. 023108, 2011.
- 35 BAKHTIARI, S.; SAFAVI-NAINI, R.; PIEPRZYK, J. *Cryptographic hash functions: A Survey*. New South Wales, Australia: University of Wollongong, 1995.
- 36 BAPTISTA, M. Cryptography with chaos. *Physics Letters A*, v. 240, n. 1–2, p. 50–54, 1998.
- 37 BARRETO, P. S. L. M.; RIJMEN, V. The WHIRLPOOL hashing function. *First open NESSIE Workshop*, Leuven, Belgium, 2000. Disponível em: <<http://www.larc.usp.br/~pbarreto/WhirlpoolPage.html>>. Acesso em: 13.06.2013.
- 38 BOER, B. den; BOSSELAERS, A. Collisions for the compression function of MD5. In: HELLESETH, T. (Ed.) *Advances in cryptology – EUROCRYPT '93: workshop on the theory and application of cryptographic techniques*. Lofthus, Norway: Springer, 1993. p. 293–304. (Lecture notes in computer science, v. 765).
- 39 CARVALHO, C. *O uso de técnicas de recuperação de informações em criptoanálise*. 2006. 79 p. Dissertação (Mestrado) — Instituto Militar de Engenharia, Rio de Janeiro, 2006.
- 40 COSTA, L. D. F.; RODRIGUES, F. A.; TRAVIESO, G.; BOAS, P. V. Characterization of complex networks: a survey of measurements. *Advances In Physics*, v. 56, n. 1, p. 167, 2007.
- 41 DAMGÅRD, I. Collision free hash functions and public key signature schemes. In: CHAUM, D.; PRINCE, W. (Ed.) *Advances in cryptology – EUROCRYPT '87: workshop on the theory and application of cryptographic techniques*. Amsterdam, The Netherlands: Springer, 1987. p. 203–216. (Lecture Notes in Computer Science, v. 304).
- 42 DENG, S.; XIAO, D.; LI, Y.; PENG, W. A novel combined cryptographic and hash algorithm based on chaotic control character. *Communications in Nonlinear Science and Numerical Simulation*, v. 14, n. 11, p. 3889–3900, 2009. ISSN 1007–5704.

- 43 DENG, S.; LI, Y.; XIAO, D. Analysis and improvement of a chaos-based hash function construction. *Communications in Nonlinear Science and Numerical Simulation*, v. 15, n. 5, p. 1338–1347, 2010.
- 44 DIFFIE, W.; HELLMAN, M. Privacy and authentication: an introduction to cryptography. *Proceedings of the IEEE*, v. 67, n. 3, 1979.
- 45 DILEEP, A.; SEKHAR, C. Identification of block ciphers using support vector machines. In: *INTERNATIONAL JOINT CONFERENCE ON NEURAL NETWORKS*, 2006, Vancouver. *Proceedings...* Vancouver, Canada: IEEE, 2006. p. 2696–2701.
- 46 DWORKIN, M. *Recommendation for block cipher modes of operation*. Gaithersburg, USA: NIST, 2001.
- 47 ECKMANN, J.-P.; RUELLE, D. Ergodic theory of chaos and strange attractors. *Review Modern Physics*, v. 57, n. 3, part 1, p. 617–656, 1985.
- 48 FALEIROS, A. C. *Criptografia*. São Carlos: SBMAC, 2011. 138 p. (Notas em matemática aplicada, v. 52).
- 49 GOLDREICH, O. *A primer on pseudorandom generators*. Providence, USA: American Mathematical Society, 2010. (University lecture series).
- 50 GREMONINI, L.; VICENTINI, E. *Autômatos celulares: revisão bibliográfica e exemplos de implementações*. *Revista Eletrônica Lato Sensu - UNICENTRO*, p. 1–19, 2008.
- 51 HABUTSU, T.; NISHIO, Y.; SASASE, I.; MORI, S. A secret key cryptosystem by iterating a chaotic map. In: DAVIES, D. W. (Ed.) *Advances in cryptology – EUROCRYPT '91, workshop on the theory and application of cryptographic techniques*. Brighton, UK: Springer, 1991. p. 127–140. (Lecture notes in computer science, v. 547).
- 52 HELLEKALEK, P. Good random number generators are (not so) easy to find. *Mathematics and Computers in Simulation*, v. 46, n. 5–6, p. 485–505, 1998.
- 53 KAYAMA, Y. *Complex networks derived from cellular automata*. Disponível em: <<http://arxiv.org/pdf/1009.4509v1.pdf>>. Acesso em: 15.06.2013.
- 54 KOCAREV, L. Chaos-based cryptography: a brief overview. *IEEE Circuits and Systems Magazine*, v. 1, n. 3, p. 6–21, 2001.

- 55 KOREA INFORMATION SECURITY AGENCY. *A design and analysis of 128-bit symmetric block cipher (SEED)*. 1999. Disponível em: <<http://tools.ietf.org/pdf/rfc4009.pdf>> Acesso em: 15.06.2013.
- 56 MACHICAO, J.; MARCO, A.; BRUNO, O. M. Chaotic encryption method based on life-like cellular automata. *Expert Systems with Applications*, v. 39, n. 16, p. 12626–12635, 2012.
- 57 MAHESHWARI, P. *Classification of ciphers*. 2001. (Master thesis) — Department of Computer Science & Engineering, Indian Institute of Technology, Kanpur, 2001.
- 58 MARCO, A.; MARTINEZ, A.; BRUNO, O. M. Fast, parallel and secure cryptography algorithm using Loren's attractor. *International Journal of Modern Physics C*, v. 21, n. 3, p. 12626–12635, 2010.
- 59 MARSAGLIA, G. *The Marsaglia random number CDROM, with the DIEHARD battery of tests of randomness*. Florida, USA: The National Science Foundation, 1998. Disponível em: <<http://www.stat.fsu.edu/pub/diehard>>. Acesso em: 20.04.2013.
- 60 MENEZES, A. J.; VANSTONE, S. A.; OORSCHOT, P. C. V. *Handbook of applied cryptography*. Boca Raton, USA: CRC Press, Inc., 1996.
- 61 MIRONOV, I. *Hash functions: theory, attacks, and applications*. 2005. Disponível em: <http://research.microsoft.com/pubs/64588/hash_survey.pdf>. Acesso em: 15.06.2013.
- 62 MOREIRA, I. de C. Sistemas caóticos em física - uma introdução. *Revista Brasileira de Ensino de Física*, v. 15, n. 1–4, p. 163–181, 1993.
- 63 NAGIREDDY, S. *A pattern recognition approach to block cipher identification*. 2008. 52 p. (Master thesis) — Indian Institute of Technology Madras, Chennai, 2008.
- 64 NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY. *FIPS PUB 180-1: secure hash standard*. 1995. Disponível em: <<http://www.itl.nist.gov/fipspubs/fip180-1.htm>>. Acesso em: 20.04.2013.
- 65 NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY. *FIPS PUB 46-3: data encryption standard (DES)*. Gaithersburg, USA: Pub-NIST, 1999.

- 66 NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY. *FIPS 197: advanced encryption standard*. 2001. Disponível em: <<http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf>>. Acesso em: 15.06.2013.
- 67 NECHVATAL, J.; BARKER, E.; BASSHAM, L.; BURR, W.; DWORKIN, M.; FOTI, J.; ROBACK, E. *Report on the development of the advanced encryption standard (AES)*. Gaithersburg, USA: NIST, 2000.
- 68 NEEDHAM, R.; WHEELER, D. *Extended tiny encryption algorithm*. 1997. Disponível em: <<http://www.cix.co.uk/~klockstone/xtea.pdf>>. Acesso em: 13.06.2013.
- 69 O'SULLIVAN, D. Exploring spatial process dynamics using irregular cellular automaton models. *Geographical Analysis*, v. 33, n. 1, p. 1–18, 2001.
- 70 PRESS, W. H.; FLANNERY, B. P.; TEUKOLSKY, S. A.; VETTERLING, W. T. *Numerical recipes in C: the art of scientific computing*. 2nd. ed. Cambridge, UK: Cambridge University Press, 1992. 269–277 p. (Fortran numerical recipes).
- 71 REIA, S. *Transições de fase e processos de nucleação no espaço de regras de autômatos celulares*. Dissertação (Mestrado) — Universidade de São Paulo, Faculdade de Filosofia, Ciências e Letras de Ribeirão Preto, Ribeirão Preto, 2011.
- 72 RIVEST, R. *The MD5 message-digest algorithm*. 1992. Disponível em: <<http://www.ietf.org/rfc/rfc1321.txt>>. Acesso em: 20.04.2013.
- 73 RUKHIN, A.; SOTO, J.; NECHVATAL, J.; SMID, M.; BARKER, E.; LEIGH, S.; LEVENSON, M.; VANGEL, M.; BANKS, D.; HECKERT, A.; *NIST special publication 800-22: a statistical test suite for random number generator for cryptographic applications*. Gaithersburg, USA: NIST, 2001.
- 74 SCHNEIER, B. *Twofish*. 1998. Disponível em: <<http://www.schneier.com/twofish.html>>. Acesso em: 07.03.2011.
- 75 SCHROEDER, M. *Fractals, chaos, power laws: minutes from an infinite paradise*. New York, USA: Freeman, 1991.
- 76 SHANNON, C. E. A mathematical theory of communication. *Bell System Technical Journal*, v. 27, p. 379–423, 1948.
- 77 SHANNON, C. E. Communication theory of secrecy systems. *Bell System Technical Journal*, v. 28, n. 4, p. 656–715, Oct.1949.

- 78 SOUZA, W. de. *Identificação de padrões em criptogramas usando técnicas de classificação de textos*. 2007. 252 p. Dissertação (Mestrado) — Instituto Militar de Engenharia, Rio de Janeiro, 2007.
- 79 STALLINGS, W. NIST block cipher modes of operation for confidentiality. *Cryptologia*, v. 34, n. 2, p. 163–175, 2010.
- 80 STÉPHANE, M. Classification and generation of disturbance vectors for collision attacks against SHA-1. *Designs, Codes and Cryptography*, v. 59, n. 1–3, p. 247–263, 2011. ISSN 0925-1022.
- 81 STINSON, D. R. *Cryptography: theory and practice*. 3rd. ed. New York: Chapman & Hall/CRC, 2006. Disponível em: <<http://www.worldcat.org/isbn/1584885084>>. Acesso em: 15.06.2013.
- 82 TREVORROW, A.; ROKICKI, T. *Golly An open source, cross-platform application for exploring Conway's game of life and other cellular automata*. Disponível em: <<http://golly.sourceforge.net/>>. Acesso em: 20.04.2013.
- 83 VERNAM, G. S. Cipher printing telegraph systems for secret wire and radio telegraphic communications. *Journal of the IEEE*, v. 55, p. 109–115, 1926.
- 84 WALKER, J. *A pseudorandom number sequence test program*. 1998. Disponível em: <<http://www.fourmilab.ch/random/>>. Acesso em: 20.04.2013.
- 85 WEIMAR, J. *Simulation with cellular automata*. Berlin: Logos, 1998.
- 86 WHEELER, D.; NEEDHAM, R. TEA, a tiny encryption algorithm. In: PRENEEL, B. (Ed.) *Fast software encryption: second international workshop*. Leuven, Belgium: Springer-Verlag, 1994. p. 363–366. (Lecture Notes in Computer Science, v. 1008)
- 87 WOLF, A.; SWIFT, J.; SWINNEY, H.; VASTANO, J. Determining Lyapunov exponents from a time series. *Physica*, v.16, p. 285–317, 1985.
- 88 WOLFRAM, S. Cellular automata. *Los Alamos Science*, v. 9, p. 2–27, 1983.
- 89 WOLFRAM, S. Universality and complexity in cellular automata. *Physica D*, v. 10, n. 1-2, p. 1–35, 1984.

- 90 WOLFRAM, S. Twenty problems in the theory of cellular automata. *Physica Scripta*, T, v. 9, p. 170–183, 1985.
- 91 WOLFRAM, S. Statistical mechanics of cellular automata. *Reviews of Modern Physics*, v. 55, n. 3, p. 601–644, 1983.
- 92 WOLFRAM, S. Cellular automata as models of complexity. *Nature*, v. 311, n. 5985, p. 419–424, 1984.
- 93 XIANG, T.; LIAO, X.; TANG, G.; CHEN, Y.; WONG, K. A novel block cryptosystem based on iterating a chaotic map. *Physics Letters A*, v. 349, n. 1–4, p. 109–115, 2006.
- 94 BACKES, A. R.; GONÇALVES, W. N.; MARTINEZ, A. S.; BRUNO, O. M. Texture analysis and classification using deterministic tourist walk. *Pattern Recognition*, v. 43, n. 3, p. 685–694, 2010.
- 95 XIAO, D.; LIAO, X.; DENG, S. One-way hash function construction based on the chaotic map with changeable-parameter. *Chaos, Solitons & Fractals*, v. 24, n. 1, p. 65–71, 2005.
- 96 GONÇALVES, W. N. *Caminhadas determinísticas em redes complexas aplicadas em visão computacional*. 2010. 84 p. Dissertação (Mestrado) — Universidade de São Paulo, Instituto de Ciência Matemática e Computação, São Paulo, 2010.

A Apêndice A: Complemento do Capítulo 6

Este apêndice foi elaborado com a finalidade de aprofundar, detalhar e exemplificar o algoritmo tratado no Capítulo 6.

A.1 Exemplificação de um modo de operação como um sistema dinâmico

Com o propósito de esclarecer o funcionamento de um sistema criptográfico seguindo as formulações dadas na Equação (6.1.1), é ilustrado na Figura A.1.1 um sistema criptográfico com $b = 2$ blocos de $n = 3$ bits, com uma cifra de bloco E_K usando o modo de operação CBC durante duas iterações, para o caso hipotético de blocos em binário $P_1 = 010$, $P_2 = 001$ e vetor de inicialização $\gamma = 101$, e por exemplo, que $C_1 = E_K(\gamma \oplus P_1) = E_K(111) = 100$.

Nota-se que o foi adicionado texto em **negrito e sublinhado** para rastrear os passos dos blocos em cada iteração, e além disso, enquanto o modo de operação é aplicado, γ recebe o texto cifrado anterior de C_b^t . Isto devido a que, para estimar o LE do modo de operação, deve selecionar-se uma réplica que esteja mais próxima da condição inicial, é dizer, orientá-la à configuração seguinte C_0^{t+1} na direção de C_b^t , e que por razões de uniformidade γ é o único valor que pode receber esta configuração inicial.

No algoritmo proposto, empregaram-se várias estratégias para calcular o expoente de Lyapunov, porém esta última estratégia foi a única que consegue distinguir dentre os modos de operação, portanto foi usada no algoritmo proposto.

$t = 0$	$t = 1$	$t = 2$...
$C_0 = \gamma = [\underline{101}]$	$C_0^1 = \gamma = [\underline{101}]$	$\gamma = C_0^2 = C_2^1 = [\underline{000}]$
$\mathbf{P} = [010, 001]$	$C_1^1 = E_K(\underline{101} \oplus 010) = \underline{100}$	$C_1^2 = E_K(\underline{000} \oplus 100) = \underline{011}$
	$C_2^1 = E_K(\underline{100} \oplus 001) = \underline{000}$	$C_2^2 = E_K(\underline{011} \oplus 000) = 111$
	$\mathbf{C}^1 = M_{E_K}(\mathbf{P}) = [100, 000]$	$\mathbf{C}^2 = M_{E_K}(\mathbf{C}^1) = [011, 111]$

Figura A.1.1 – Exemplo de modo de operação CBC, iterando no tempo com $b = 2$ blocos, $n = 3$ bits e cifra de bloco E_K .

A.2 Limite superior do expoente de Lyapunov

De acordo com Baetens e De Baets (34), o limite superior (*upper bound*) do expoente de Lyapunov nos ACs pode ser aproximado teoricamente. Este limite superior constitui a máxima taxa de divergência como consequência de uma perturbação topológica e não apenas de uma única célula isolada.

Assim, no contexto dos sistemas criptográficos, numa situação ideal, procuramos pelo número máximo de bits que podem tornar-se em bits defeituosos na seguinte iteração. Por exemplo, consideramos o texto claro $\mathbf{P} = 10101101$ e sua versão perturbada $\mathbf{P}^* = 10101100$, bem como uma cifra de bloco hipotética que encripta da seguinte forma $E_K(\mathbf{P}) = 10101000$ e $E_K(\mathbf{P}^*) = 01010111$, respectivamente. O pior cenário possível, o máximo número de diferenças pode ser 8, que é o comprimento do texto claro $N = 8$. Isto significa que, uma perturbação do bit c_i^j no tempo t , pode propagar-se no máximo à N bits defeituosos no tempo subsequente. Portanto, a aproximação do limite superior λ_m para os sistemas de criptografia (que foi dada na Equação (6.3.1) é:

$$\lambda_m = \frac{1}{t} \log(\lambda^t) = \log(N) = \log(b \times n). \quad (\text{A.2.1})$$

Para o caso de uma cifra de bloco com texto claro de $b = 5$ blocos da FAMILIA-64 o $\lambda_m = \log(64 \times 5) \approx 5.77$ e para o caso da FAMILIA-128 seria $\lambda_m = \log(128 \times 5) \approx 6.46$. Nota-se que estes valores foram empregados para a normalização dos expoente de Lyapunov calculados no Capítulo 6.

A.3 Otimização do algoritmo proposto

Para compreender melhor o algoritmo proposto, podemos visualizá-lo como uma árvore com ramificações em vários níveis, cujos nós representam as posições dos bits do texto claro (ver Figura A.3.1) e os níveis representam o tempo de iteração do algoritmo. A raiz da árvore representa o bit de diferença da configuração inicial. Assim, o bit da raiz pode afetar x bits da seguinte iteração, abrindo x ramificações, e cada um dos bits das ramificações pode afetar y bits subsequentes, e assim por diante. Tendo isto em mente, existem situações em que um bit em particular pode ser afetado por vários bits de níveis superiores. Se o algoritmo fosse implementado tal qual, atingiríamos um alto custo computacional. Porém, o algoritmo pode ser facilmente otimizado, pois não é necessário recalcular os bits diferentes ϵ de cada vez, senão apenas considerar a quantidade de ramificações de cada nível. Assim, evitamos percorrer ramificações repetidas (flechas sobrepostas) que viessem de níveis superiores.

Figura A.3.1 – Ilustração de uma árvore de defeitos, para indicar o número de defeitos em cada iteração.

Para otimizar então, apenas contamos cada réplica uma única vez e mantemos o controle da sua multiplicidade. Por exemplo na Figura A.3.2, durante o $t = 2$ a réplica R_2^1 seria repetida duas vezes e, em seguida, na iteração subsequente, nasceriam dois bits defeituosos, o que significa um total de quatro ramificações no $t = 3$. Assim, seguindo a multiplicidade de todos os defeitos separadamente, em vez de todos os defeitos, a eficiência do algoritmo aumenta substancialmente, pois não é necessário manter o controle de cada ramificação que emerge durante as iterações do sistema criptográfico.

Tempo	Texto claro original $\mathbf{P}$	Texto claro perturbado $\mathbf{P}^*$	# defeitos
$t = 0$	$C_0 = \gamma = [\underline{101}]$ $\mathbf{P} = [010, 001]$	$C_0 = \gamma^* = [\underline{101}]$ $\mathbf{P}^* = [01\underline{1}, 001]$	$\epsilon_0 = 1$
$t = 1$	$\mathbf{C}^1 = M_{E_K}(\mathbf{P}) = [\underline{100}, 000]$	$\mathbf{C}^{1*} = M_{E_K}(\mathbf{P}^*) = [\underline{111}, 010]$ <div style="display: flex; justify-content: space-around; margin-top: 10px;"> <div style="text-align: center;">$R_1^1 = [\underline{110}, 000]$</div> <div style="text-align: center;">$R_2^1 = [\underline{101}, 000]$</div> <div style="text-align: center;">$R_3^1 = [\underline{100}, 010]$</div> </div>	$\epsilon_1 = 3$
$t = 2$	$\mathbf{C}^2 = M_{E_K}(\mathbf{C}^1) = [\underline{011}, 111]$	<div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> $M_{E_K}(R_1^1) = [\underline{001}, 1\underline{01}]$ $M_{E_K}(R_2^1) = [\underline{001}, 111]$ $M_{E_K}(R_3^1) = [\underline{010}, 11\underline{0}]$ </div> <div style="width: 45%;"> $R_1^2 = [\underline{001}, 111] \times 2$ $R_2^2 = [\underline{011}, 1\underline{01}]$ $R_3^2 = [\underline{010}, 111]$ $R_4^2 = [\underline{011}, 11\underline{0}]$ </div> </div>	$\epsilon_2 = 5$
$t = 3$ $\vdots$	$\mathbf{C}^3 = M_{E_K}(\mathbf{C}^2) = [\underline{010}, 011]$	<div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> $M_{E_K}(R_1^2) = [\underline{110}, 01\underline{0}]$ $M_{E_K}(R_2^2) = [\underline{011}, 011]$ $M_{E_K}(R_3^2) = [\underline{010}, \underline{101}]$ $M_{E_K}(R_4^2) = [\underline{010}, 0\underline{11}]$ </div> <div style="width: 45%;"> $R_1^3 = [\underline{110}, 011] \times 2$ $R_2^3 = [\underline{010}, 01\underline{0}] \times 2$ $R_3^3 = [\underline{011}, 011]$ $R_4^3 = [\underline{010}, \underline{111}]$ $R_5^3 = [\underline{010}, 0\underline{01}]$ </div> </div>	$\epsilon_3 = 2 \times 2 + 1 + 2 = 7$

Figura A.3.2 – Ilustração do cálculo do LE para um modo de operação genérico com uma cifra de bloco de $b = 2$ blocos e $n = 3$ bits.

B Apêndice B: Resumo das publicações

A seguir são apresentados os resumos dos artigos publicados durante este mestrado:

- MACHICAO, J.; MARCO, A. G.; BRUNO, O. M. **Chaotic encryption method based on life-like cellular automata.** *Expert Systems with Applications*, vol. 39, p. 12626–12635, 2012.

Abstract: A chaotic encryption algorithm is proposed based on the “life-like” Cellular Automata (CA), which acts as a pseudo-random generator (PRNG). The paper main focus is to use chaos theory to cryptography. Thus, CA was explored to look for this “chaos” property. This way, the manuscript is more concerning on tests like: Lyapunov exponent, Entropy and Hamming distance to measure the chaos in CA, as well as statistic analysis like DIEHARD and ENT suites. Our results achieved higher randomness quality than others ciphers in literature. These results reinforce the supposition of a strong relationship between chaos and the randomness quality. Thus, the “chaos” property of CA is a good reason to be employed in cryptography, furthermore, for its simplicity, low cost of implementation and respectable encryption power.

- MACHICAO, J.; MARCO, A. G.; BRUNO, O. M. **Lyapunov exponent: a qualitative ranking of block cipher modes of operation.** In: GILBERT T.; KIRKILIONIS M.; NICOLIS G. (Ed.) *EUROPEAN CONFERENCE ON COMPLEX SYSTEMS, Springer Proceedings in Complexity, Springer International Publishing, ECCS'12*, 2012, Brusels. *Proceedings...* Brusels, Belgium, 2012. p. 979–985

Abstract: In Cryptography, a mode of operation is a technique to improve a block cipher effect. A challenge question of how to “qualitatively compare” among block cipher modes of operation, remains poorly studied. To overcome this lack we propose a methodology through some analogies to discrete dynamical systems (DDS). The

method consists in the measure of chaos on the modes of operation by estimations of the Lyapunov exponent (LE). We opted to exploit the LE measures to compare among: ECB, CBC, OFB, CFB and CTR modes of operation. Results showed an effectively tool to qualitatively rank modes of operation. These novel findings represent an important advance to cryptography.

- MACHICAO, J.; MARCO, A. G.; BRUNO, O. M. **Lyapunov exponent: towards randomness measure.** In: *COMPLEX BEHAVIOR IN DISCRETE DYNAMICAL SYSTEMS (SATELLITE)*, ECCS'12, 2012, Brussels. *Proceedings...* Brussels, Belgium, 2012. p. 15–18.

Abstract: We propose a method based on Lyapunov Exponent (LE) capable of measuring randomness of PRNG (Pseudo-random number generator). Motivated by the strong relationship between chaos and randomness. We study PRNG as a discrete dynamical system. Results shows good discrimination among PRNG parameters.

- MACHICAO, J.; BAETENS, J.; MARCO A. G.; DE BAETS, B.; BRUNO, O. M. **A dynamical systems approach to the discrimination of the modes of operation of cryptographic systems.** (Aguardando submissão)

Abstract: Evidence of signatures associated with cryptographic modes of operation is established. Motivated by some analogies between cryptographic and dynamical systems, in particular with chaos theory, we propose an algorithm based on Lyapunov exponents of discrete dynamical systems to estimate the divergence among ciphertexts as the encryption algorithm is applied iteratively. The results allow to distinguish among six modes of operation, namely ECB, CBC, OFB, CFB, CTR and PCBC using DES, IDEA, TEA and XTEA block ciphers of 64 bits, as well as AES, RC6, Twofish, Seed, Serpent and Camellia block ciphers of 128 bits. Furthermore, the proposed methodology enables a classification of modes of operation of cryptographic systems according to their strength.

- **Cryptographic hash function based on hybrid chaotic cellular automata.** MACHICAO, J.; BRUNO, O. M. (Em redação)

Esse artigo recompila os resultados obtidos da proposta do algoritmo de *Hash* criptográfico desenvolvido durante este mestrado. Neste artigo será introduzida a abordagem híbrida entre os ACs caóticos e as redes complexas, como uma estratégia

para proporcionar vantagens na projeção e construção do algoritmo.