Computational Intelligence Applied on Cryptology: a Brief Review

M. Danziger, Member IEEE, and M. A. A. Henriques

***Abstract*— Many cryptographic techniques have been developed and several were broken. Recently, new models have arisen with different and more complex approaches to cryptography and cryptanalysis, like those based on the Computational Intelligence (CI). Different bio-inspired techniques can be found in the literature showing their effectiveness in handling hard problems in the area of cryptology. However, some authors recognize that the advances have been slow and that more efforts are needed to take full advantage of CI techniques. In this work, we present a brief review of some of the relevant works in this area. The main objective is to better understand the advantages of applying CI on cryptology in the search for new ways of improving computer security.**

***Keywords*— Artificial Neural Network, Evolutionary Computation, Artificial Immune Systems, Cellular Automata, DNA Computing, Cryptology, Cryptography, Computational Intelligence.**

I. INTRODUÇÃO

A CRIPTOLOGIA é uma importante área da ciência. Ela é baseada na teoria dos números e na teoria da informação sendo composta por duas frentes interdependentes: a criptografia e a criptoanálise. Assim, um cifrador é um algoritmo criptográfico que usa funções específicas para cifrar e decifrar mensagens. Tais algoritmos são categorizados por dois tipos principais de chaves: simétricas e assimétricas [1]. Nos últimos anos muitos esquemas criptográficos foram criados e vários sofreram ataques. Um fator preponderante nesses ataques é o crescimento do potencial computacional e das técnicas de criptoanálise. Por esse motivo, há a necessidade de desenvolver sistemas criptográficos cada vez mais robustos, assim como estudar novas técnicas de criptoanálise para que seja possível conhecer melhor a robustez dos criptossistemas.

A inteligência computacional (CI – *Computational Intelligence*) tem como característica principal a habilidade em resolver problemas complexos, fato que facilita sua aplicação em criptologia. Na literatura, é possível encontrar várias técnicas de CI sendo aplicadas para resolver os mais variados problemas. Na área de criptologia destacam-se as técnicas de computação evolutiva (EC – *Evolutionary Computation*), redes neurais artificiais (ANN – *Artificial Neural Networks*), autômatos celulares (CA – *Cellular Automata*) e computação baseada em DNA (*DNA Computing*). Nosso objetivo ao fazer essa revisão é a investigação das vantagens e desvantagens da aplicação de CI à criptologia buscando novas linhas de pesquisa. Para melhor compreensão, dividimos o trabalho da seguinte forma: na seção 1 é apresentado um breve resumo sobre criptologia, seguido de um breve resumo das técnicas de CI na seção 2. Ainda na seção 2, nós apresentamos alguns estudos que tratam do potencial de CI à criptologia. Na seção 3 apresentamos algumas aplicações de CI à criptologia. Na seção 4 discutimos alguns pontos dessas aplicações que consideramos importantes e que foram levantados durante o estudo e, por fim, na seção 5 tecemos algumas conclusões.

M. Danziger, Universidade Estadual de Campinas (UNICAMP), Campinas, SP, Brasil, danziger@ dca.fee.unicamp.br

M. A. A. Henriques, Universidade Estadual de Campinas (UNICAMP), Campinas, SP, Brasil, marco@dca.fee.unicamp.br

II. CRIPTOLOGIA

Na Criptologia, a função básica da criptografia é a codificação de uma mensagem (texto em claro) em outra mensagem (texto cifrado) de difícil compreensão, caso seja interceptada por entidades não autorizadas [1]. Existem muitos tipos de algoritmos de codificação/decodificação. Porém, os mais utilizados são baseados em blocos como, por exemplo, o DES – *Data Encryption Standard* e o AES – *Advanced Encryption Standard* [2]. Ambos fazem parte do modelo criptográfico baseado em chaves simétricas. Existem ainda os algoritmos baseados em chaves assimétricas (isto é, existem duas chaves diferentes sendo uma pública e outra privada). Atualmente, as técnicas de CI têm sido adotadas para ambos os modelos de algoritmos.

A análise de um sistema criptográfico é algo essencial para a criptologia e a criptoanálise não existiria sem a criptografia [1]. A criptoanálise se utiliza de vários meios (por exemplo, estatísticos) para investigar possíveis vulnerabilidades nos algoritmos criptográficos. De acordo com Joux [1], um bom algoritmo de criptografia precisa passar pelo crivo da criptoanálise. Todos os algoritmos criptográficos são vulneráveis a pelo menos um tipo de ataque: o ataque de força bruta no qual todas as possíveis chaves são testadas. Mas, este ataque esbarra no tamanho do espaço de busca da chave, o qual geralmente é tão grande que não é possível testar todas as possíveis chaves em um tempo aceitável. Abordagens bio-inspiradas têm sido usadas para analisar criptossistemas. Como princípio básico, o uso de CI nesse caso objetiva encontrar soluções que possam diminuir o tempo e a complexidade da busca da chave ou parte dela (por exemplo, ajudando uma busca baseada em força-bruta).

III. INTELIGÊNCIA COMPUTACIONAL E CRIPTOLOGIA

A inteligência computacional inclui várias áreas que são geralmente inspiradas por processos encontrados na natureza. Para este trabalho, nós abordaremos algumas das mais importantes técnicas, parametrizadas pela quantidade de

trabalhos encontrados na literatura. Entendemos que não se devem organizar as técnicas por importância ou capacidade, afinal, cada uma tem sua aplicação, podendo ser bastante distinta uma da outra para o mesmo problema. Assim, a ordem em que apresentaremos as técnicas não tem relação com estas premissas.

Existem várias técnicas bio-inspiradas, porém, neste trabalho não apresentaremos suas definições completas e formais, mas apenas uma breve introdução. Recomendamos o livro de Norvig e Russel [3] para uma visão mais completa das técnicas de inteligência artificial. Para aqueles que já têm algum conhecimento, indicamos os seguintes trabalhos: em [4] são apresentadas as ANNs, em [5] é mostrado o potencial dos algoritmos genéticos, em [6] é apresentado o modelo computacional baseado em DNA (*Deoxyribonucleic acid*), a teoria dos autômatos celulares é vista em [7], em [8] é abordada a teoria dos enxames de partículas (PSO – *Particle Swarm Optimization*), em [9] é mostrada a inspiração das colônias de formigas para problemas de busca e otimização (ACO – *Ant Colony Optmization*) e, finalmente, em [10] é detalhado o novo paradigma da inteligência computacional conhecido como sistemas imunológicos artificiais (AIS – *Artificial Immune System*). Existem outras técnicas de CI além das investigadas nesse trabalho, as quais poderão ser objetivo de uma versão futura.

Nas subseções seguintes apresentaremos os trabalhos que consideramos mais relevantes para cada técnica.

A. Redes Neurais Artificiais

Algumas das características mais importantes das ANNs derivam da sua estrutura paralela e sua inerente capacidade de se adaptar a problemas específicos [11]. Assim, guiados por estas habilidades, muitos pesquisadores têm aplicado ANN para criptologia. Por exemplo, Laskari et. al. [11] estudaram o desempenho das ANNs para resolver o problema do logaritmo discreto (DLP – *Discrete Logarithm Problem*) e o problema da fatoração de inteiros grandes (ambos os problemas sustentam a criptografia de chaves assimétricas pela dificuldade matemática de encontrar a chave privada a partir dos dados disponíveis). Os autores utilizaram uma rede neural recorrente (FNN – *Feedforward Neural Network*) e, de acordo com as simulações realizadas, as FNNs alcançaram resultados satisfatórios apenas para números pequenos. Tanto no trato do DLP como no problema da fatoração de inteiros. Sobre o uso de ANN os autores apresentaram um fato importante: a necessidade de alterar as configurações (por exemplo, treinamento da FNN) conforme cresce a dificuldade do problema. Não é possível dizer que este fato desqualifica as ANNs para criptoanálise, porém, dificulta o seu uso.

Liu e Guo [12] usaram redes neurais de Hopfield (HNN) com uma camada caótica (isto é, uma camada baseada na teoria do caos) para quebrar a linearidade, indesejável na criptografia, e desenvolver um criptossistema de chaves públicas. O modelo desenvolvido apresentou características que facilitam o seu uso no contexto atual de criptografia, já que tem uma codificação 50 vezes mais rápida que a do RSA. Ao fazer a criptoanálise do modelo, os autores apontaram que, pela dificuldade da decomposição de matrizes (bastante usada na teoria do caos) e das propriedades dos classificadores caóticos, o modelo apresenta índices satisfatórios de robustez. Há outra característica importante no modelo de Liu e Guo: o uso de processos diferentes para cifrar e decifrar. O resultado prático dessa técnica é o aumento da dificuldade para que atacantes possam encontrar a chave privada usando ataques do tipo *chosen-plaintext* e *known-plaintext*.

Lian [13] também usou uma camada caótica para desenvolver um cifrador de bloco. No modelo proposto, existem duas camadas neurais: uma para o processo de difusão dos dados e outra para o processo de confusão de dados. A camada neural caótica foi implementada para o primeiro caso, enquanto que uma camada neural linear foi implementada para o segundo caso. As camadas são repetidas t vezes a fim de garantir maior segurança ao criptossistema. Os parâmetros das camadas neurais são parametrizados pela chave (inclusive os parâmetros da parte caótica). O desafio neste caso é garantir o processo de decodificação pela sensibilidade existente no processo de reversibilidade do mapeamento da matriz de confusão (isto é, conversão de texto cifrado para texto claro novamente). Segundo Lian, ao acrescentar a camada neural caótica o sistema se tornou mais robusto. Como foram usadas matrizes de pesos como chave (somente as matrizes não resistem aos ataques do tipo *known-plaintext* ou *chosen-plaintext*) foi necessário inserir uma relação não-linear entre o texto claro e o texto cifrado. Esse problema foi mitigado através da inserção de uma função caótica e de um vetor com informações caóticas dentro do sistema. Portanto, para recuperar a chave, é necessário quebrar primeiro a função caótica que requer S^n, e a matriz de peso, que requer $n!$ (S é um número inteiro e n é a quantidade de bits). Lian mostrou que o espaço de busca para a chave por ataque de força-bruta é menor que o espaço de busca para os ataques à camada caótica.

ANNs têm sido propostas também para uso em funções *hash*, assunto que abordaremos na subseção H.

B. Computação Evolutiva

John Clark [14], aproximadamente dez anos após os primeiros trabalhos sobre a convergência de CE e criptologia, foi convidado a escrever um artigo no qual fez uma discussão sobre o passado, presente e futuro da criptografia bio-inspirada. Este trabalho talvez tenha sido o primeiro a fazer um levantamento sobre a aplicação de técnicas de CI à criptologia. Dentre as técnicas, o autor aponta como candidata promissora a dos algoritmos genéticos (GA – *Genetic Algorithms*).

A história dos GAs para criptologia iniciou-se com o trabalho de Spillman [15] ao utilizar um modelo de GA para criptoanálise de algoritmos clássicos de criptografia. Spillman aplicou GA sobre cifradores que usavam substituições simples e também sobre cifradores baseados no problema da mochila (um criptossistema de chave pública que é baseado no conhecido problema NP-completo de maximizar a quantidade

de objetos dado o tamanho da mochila). Dessa forma, Spillman demonstrou que era possível aplicar GA para criptoanálise de algoritmos simétricos e assimétricos.

No mesmo ano, Mathews [16] investigou o uso de GA para criptoanálise de cifradores que usavam transposição. Clark [14] apontou que o poder real dos GAs, neste caso, surge quando a permutação verdadeira é procurada. O método foi considerado sofisticado e original sendo bastante citado em outros trabalhos. Mathews e Spillman usaram funções de aptidão (*fitness* – que mostra o quanto uma solução candidata é apta perante a função objetivo) sobre o tamanho da mensagem e a distribuição da frequência dos digramas e trigramas.

Andrew Clark [17] propôs o uso de GA para cifradores usando substituição simples e permutações. Apesar de ser um trabalho considerado curto e com algumas falhas (por exemplo, é difícil a re-implementação por não conter explicações detalhadas do GA utilizado) é a primeira referência de uso para o algoritmo TEA (*Tiny Encryption Algorithm*). Clark, Dawson e Bergen [18] propuseram uma extensão do trabalho de Spillman. Porém, a função de aptidão apresentou uma séria dificuldade: nem sempre indicava a distância correta entre a distância de *Hamming* da solução proposta e a solução esperada. Em [19], Clark, Dawson e Nieuwlan usaram pela primeira vez um GA paralelo. No trabalho, eles atacaram um cifrador que fazia substituição polialfabética usando um número de GAs seriais trabalhando em partes diferentes e separadas do problema. Assim, um número X de processadores é utilizado. Cada processador trabalha em uma das K chaves e troca informações após certo número de iterações. Um ponto fraco no modelo é a aplicação apenas para ambientes executando em paralelo, o que impede comparações com outros modelos.

Em 1997, três trabalhos foram publicados, dois deles novamente atacando algoritmos clássicos (*Vigenère e Merkle-Hellman Knapsack*) [20] e [21]. Porém, Bagnall, McKeown e Rayward-Smith [22], aplicaram pela primeira vez GA sobre uma técnica considerada moderna (trata-se apenas de uma técnica mais elaborada, diferente dos algoritmos modernos de criptografia). Os autores realizaram ataque sobre uma versão simplificada de uma máquina de três rotores conhecida como ENIGMA (dispositivo eletromecânico para codificar e decodificar mensagens, usado pelos alemães na segunda guerra mundial). Porém, GA foi usado para encontrar o último dos três rotores. Para os outros dois rotores foi usada uma técnica iterativa. A função de aptidão é baseada no "*phi-test*" (técnica estatística) para avaliar a aleatoriedade do texto. Os resultados mostram que uma máquina de três rotores desconhecida pode ser criptoanalisada com aproximadamente 4000 letras do texto cifrado. Clark e Dawson [23] apresentaram, em 1998, um trabalho no qual fizeram comparações entre três algoritmos aplicados a cifradores usando substituição simples. Os algoritmos foram: GA, Busca Tabu (TS – *Tabu Search* é uma meta-heurística que guia um algoritmo de busca local na exploração contínua dentro de um espaço de busca) [24] e *Simulated Annealing* (SA é inspirado no modelo de cozimento e resfriamento de metais). Os critérios de comparação foram: a quantidade de texto cifrado usado para o ataque, o número de chaves consideradas antes de a chave correta ser encontrada, e o tempo necessário para o ataque determinar a solução correta. O desempenho dos três algoritmos foi semelhante. Porém, quando considerada a complexidade, o algoritmo TS foi o mais eficiente, enquanto que GA foi o menos eficiente.

Yaseen e Sahasrabuddhe [25] apresentaram um trabalho atacando o cifrador de chave pública Chor-Rivest. Neste trabalho, o ponto mais importante foi o uso da técnica de distância de Hamming, no qual quando o tamanho da mochila aumentou, a porcentagem do espaço de busca visitado diminuiu. Sem a técnica, o percentual visitado não tinha relação com o tamanho da mochila. Se o espaço de busca diminui, a busca tem alta probabilidade de encontrar mais rapidamente a solução (porém, é preciso ter cuidado para não cair em mínimos locais que podem ocultar o mínimo global).

Hernández et. al. [26] e [27] propuseram uma nova técnica de criptoanálise para o TEA e, posteriormente em [28], novos resultados para o TEA e para sua versão estendida XTEA (*Extended Tiny Encryption Algorithm*) com número reduzido de rodadas (a redução do tamanho do problema para instâncias menores é uma técnica bastante comum para abordagens iniciais do problema). Eles provaram que o uso de GA para distinguir um cifrador de bloco de uma permutação aleatória é possível. Posteriormente, Garret, Hamilton e Dozier [29] estenderam o trabalho de Hernández mostrando que, se cuidadosamente adaptados, GA são capazes de determinar com mais rapidez se um dado cifrador produz saída aleatória.

John Clark comparou, em sua tese de doutorado, o algoritmo TS com várias técnicas, incluindo GA, para quebra de criptossistemas clássicos [30]. John também apresentou algumas limitações, afirmando que é preciso definir com muito cuidado os parâmetros dos algoritmos para que seja possível encontrar boas soluções. Em [14], John apresentou a ideia de que a natureza pode inspirar a criptologia. Em suas considerações, ele também propôs o uso de CE, principalmente para criptoanálise. O autor chegou a propor que não há provas de que os cifradores modernos sejam imunes a ataques por técnicas evolucionárias ou outras técnicas computacionais não padronizadas. O autor ainda conclamou a comunidade de CE a para acreditar na possibilidade da aplicação bem sucedida desta área à criptoanálise dos algoritmos modernos de criptografia.

Recentemente, trabalhos usando GA para criptoanálise de versões simples do DES com menos rodadas vêm sendo apresentadas. É o caso do trabalho de Yang, Song e Zhang [31] que propuseram um modelo de criptoanálise aplicado ao DES com seis rodadas. Para analisar quantitativamente a validade da abordagem evolucionária de criptoanálise em cifradores de blocos, eles desenvolveram uma função de aptidão através da ideia da estimação de máxima vizinhança (foi usada uma representação binária para o problema) e aproximação linear. As chaves para as seis rodadas foram divididos em dois grupos de experimentos: com 49 e 42 bits.

Segundo os autores, o fato da introdução de técnicas não-linear na criptografia de cifradores de blocos modernos prejudica a criptoanálise usando abordagens evolucionárias. Mesmo assim, eles são categóricos em afirmar que existe um campo aberto para exploração.

Em [32], Husein et. al. propuseram o uso de GA como técnica para criptoanálise do DES (*Data Encryption Standard*) de oito rodadas. Segundo os autores, a abordagem usada obteve resultados melhores que força-bruta e criptoanálise diferencial (DC – *Differential Cryptanalysis*) [33]. No trabalho, GA foi usado para encontrar os bits de saída de cada "*S-box*". Assim, em cada iteração (oito no total) os bits de saída da "*S-box*" constituem o cromossomo atual do GA (cada cromossomo é constituído de 6 bits). Certamente, esta é uma técnica baseada em DC (*Differential Cryptanalysis* é uma técnica que avalia de forma estatística as diferenças de entrada e saída de dois textos escolhidos a fim de encontrar a subchave da última rodada). Um conceito que se destaca neste trabalho é a geração de pares direitos geneticamente, isto é, a geração mais eficiente de pares que fornecem indicadores que podem ajudar na determinação da chave correta para a iteração analisada, a fim de mitigar o problema do uso de grande número de pares direitos. Pares direitos são os pares que possuem alguma correlação entre a entrada do texto em claro e a saída do texto cifrado (os textos são analisados em pares). Normalmente, há necessidade de se usar muitos pares direitos até encontrar a chave correta, o que provoca sobrecarga na execução do ataque. Ao diminuir o número de pares direitos, o método diminui o custo e o tempo para o ataque.

Algoritmos evolucionários também podem ser usados para o desenvolvimento de criptossistemas ou parte deles. Millan, Clark e Dawson [34], [35] demonstraram essa capacidade propondo um modelo para a geração de funções *Booleanas* com excelentes aplicações criptográficas. Através desse trabalho, inaugurou-se um caminho para algoritmos evolucionários serem empregados à criptografia. É o caso do trabalho de Nedjah e Mourelle [36]. O trabalho focou o desenvolvimento de uma "*S-box*" regular que apresentasse alta não-linearidade e baixa propriedade de auto-correlação usando CE. Para alcançar bons resultados com as "S-boxes", foi necessário otimizar três propriedades: regularidade, não-linearidade e auto correlação. Para o trabalho foi utilizado um algoritmo evolucionário multiobjetivo baseado no equilíbrio de Nash (aplicado na teoria dos jogos, o equilíbrio de Nash representa uma situação em que nenhum jogador pode melhorar a sua situação, dada a estratégia seguida pelo jogador adversário) para construir caixas de substituição resilientes. Nedjah e Mourelle enfatizaram que, segundo os resultados, sua abordagem para construção de S-boxes era melhor que as de Millan et. al. [37] e Clark et. al. [38] que usaram SA para construir *S-boxes*.

Nos últimos anos, vários trabalhos usando CE têm sido propostos, porém, a grande maioria tem realizado pequenas modificações (principalmente na forma de representação) que geralmente apontam para resultados melhores que os anteriores. Nesta linha estão os trabalhos de Bergman, Scheidler e Jacob [39] que apresentaram um GA baseado em permutações para atacar com melhor qualidade os cifradores de transposição. Eles também não apontaram nenhum sucesso do uso do modelo contra o DES ou o AES (completos). Vimalathithan e Valarmathi [40] descreveram um bom desempenho de um ataque (baseado em texto cifrado) usando GA contra uma versão simplificada do DES. Usando a abordagem do trabalho é possível encontrar a chave mais rapidamente. Garg [41] comparou as técnicas GA, algoritmo memético (MA – *Memetic algorithm* é um algoritmo evolutivo que se utiliza dos operadores genéticos em combinação com um procedimento de busca local) e SA sobre uma versão simplificada do DES. Segundo o autor, MA obteve desempenho um pouco melhor que GA e SA. O fato ocorre por causa da busca local executada pelo MA que explora de forma mais efetiva o espaço de busca. Garg e Shastri [42] mostraram que variações nos parâmetros de entrada iniciais podem melhorar o desempenho do GA para criptoanálise do cifrador baseado no problema da mochila. Venkateswaran e Sundaram [43] demonstraram que é possível explorar as características dos GAs com métodos de poli substituição em um caminho linear para gerar valores ASCII de um dado texto e então aplicar conversão, transposição com as características de criptografia. Esse é um modelo em desenvolvimento, porém, trabalha com chave simétrica e assimétrica. Em [44] é encontrado um modelo semelhante.

C. Particle Swarm Optimization (PSO)

Em 1995, Kennedy e Eberhart [8] propuseram a nova técnica conhecida como "*Particle Swarm*". Como inspiração, os autores tiveram: movimento (voo) das aves, cardume de peixes e a teoria de enxames. PSO possui muitas similaridades com GA. Ou seja, alguns conceitos chaves de GA estão presentes em PSO, por exemplo: população, função de aptidão, indivíduos representando possíveis soluções ótimas. Porém, existem diferenças, e elas são importantes: em PSO não existe o conceito de *crossover* (recombinação de dados usando dois indivíduos para gerar um terceiro, conhecido como filho) e mutação. Dessa forma, esta técnica é mais simples de implementar que GA. Certamente, existem menos parâmetros para ajustar quando comparada com GA.

Inicialmente, PSO foi aplicado para resolver alguns problemas difíceis que são usados na criptologia: caso da tarefa de fatoração de inteiros e da tarefa de encontrar a cadeia de adição com comprimento mínimo (importante para o cálculo da exponenciação de corpos finitos) [45] e [46].

Porém, Laskari et. al. [47] fizeram um primeiro estudo usando PSO para criptoanálise do DES com quatro de 16 rodadas. Mais especificamente, eles investigaram o problema de encontrar alguns bits faltantes da chave usada em um simplificado cifrador Feistel simplificado. Segundo os resultados encontrados, foram necessárias 1500 funções de avaliação em oposição aos 2^{14} requeridos quando usada força bruta. Porém, os autores utilizaram juntamente PSO e força bruta para encontrar os 14 bits faltantes. Nesse caso, o PSO

foi usado para acelerar o processo de busca. É importante relatar que o ataque não busca alcançar a chave aleatoriamente (isto é, sem um parâmetro inicial). Toda a base do ataque é guiada pela criptoanálise diferencial.

Nalini e Rao [48] também aplicaram PSO para versões simplificadas e modificadas do DES (especificamente para 6 rodadas). Segundo as autoras, pela sua simplicidade, o PSO tem desempenho melhor que outras técnicas (TS, GA) quando comparadas em relação a evolução em longos períodos (nesse caso os valores da função de avaliação tendem a piorar bastante). Para usar o PSO, foi dividido o espaço de busca em 100 enxames sendo garantido que no início eles começam em posições diferentes. Porém, é muito difícil garantir que as trajetórias deles não se cruzam durante a execução. Os resultados encontrados mostraram que PSO possui melhor desempenho que outras estratégias evolucionárias.

Laskari et. al. [49] apresentaram um dos trabalhos mais elaborados e extensos sobre a aplicação de CI à criptologia. No trabalho, os autores enfatizaram o PSO para criptoanálise do DES simplificado para 4 e 6 rodadas apenas. Nesse trabalho eles melhoraram a média de funções de avaliações, que ficou bem inferior às 2^{14} avaliações necessárias por força brutas. Em contrapartida, eles apontaram como pontos críticos para o desempenho das técnicas de CI aplicadas à criptologia a formulação do problema e a representação.

Estudos mais recentes foram encontrados na literatura. Como exemplo, citamos o trabalho de Léon-Javier et. al. [50] que aplicaram PSO para encontrar a cadeia de adição de menor comprimento. As cadeias de adição são representadas diretamente como partículas e os valores de velocidade representam as regras indicando quais elementos devem ser adicionados para obter cada elemento das partículas. Segundo os autores, o resultado foi bastante competitivo, embora necessite de mais simulações para expoentes maiores. Uddin e Youssef [51] usaram PSO para criptoanálise de cifradores de substituição simples e apontaram uma desvantagem do uso da técnica: a alta sensibilidade às variações dos parâmetros.

D. Ant Colony Optimization (ACO)

Inspirada em colônia de formigas na busca por comida, ACO é uma heurística que se baseia na probabilidade de uma formiga escolher um determinado caminho entre a comida e seu formigueiro de acordo com a quantidade de feromônio depositado naquele caminho [9]. Por simular um comportamento social, a ação coletiva de muitas formigas (através de comunicação indireta) pode resultar na localização do menor caminho entre a comida e o formigueiro. A técnica foi originalmente criada para resolver problemas computacionais que envolvem difíceis problemas de otimização discreta (problema NP-difícil do caixeiro viajante) [9].

Na criptologia, ACO tem sido aplicado para criptoanálise. Russel, Clark e Stepney [52] apresentaram o primeiro trabalho com aplicação de ACO para cifradores clássicos que usam transposição. Segundo o trabalho, as formigas constroem caminhos no grafo de anagramas de um criptograma e cada caminho completo é uma permutação de nós que corresponde a uma chave. Não há necessidade de podar arcos nessa abordagem. Isso ocorre porque as formigas fazem uso da heurística $n_{i,j}$, na qual representa algum conhecimento a priori sobre a possibilidade de escolher um determinado nó j ao invés do nó i. Em cada iteração do algoritmo, após as formigas terem construído seus caminhos, feromônios são depositados em arcos no grafo correspondente ao melhor caminho encontrado, desde o início do algoritmo. Neste caso, um caminho completo, que é uma chave de decodificação, é avaliado usando a heurística *Dict* (uma heurística proposta pelos autores usando um dicionário de 40 mil palavras) sobre o texto em claro produzido. Assim, eles descobriram que *Dict(M)*, sendo *M* o texto claro, é o máximo sempre que *M* é o texto claro correto, considerado sobre todas as *M* decodificações possíveis de um criptograma. Portanto, uma forma de recuperar o texto original é considerar o problema de maximização com *Dict* como função objetivo. Todos os caminhos no grafo de anagramas são avaliados usando *Dict* sendo o máximo identificado como o texto claro.

Segundo os autores, ACO foi mais eficiente na busca pela chave completa que os algoritmos GA, SA e TS. A eficiência nesse caso está relacionada à quantidade de texto cifrado necessário para a execução do ataque.

Bafghi e Sadeghiyan [53] apresentaram um trabalho usando ACO para melhorar o ataque diferencial DC sobre o algoritmo SERPENT, um dos finalistas do concurso realizado pelo NIST (*National Institute of Standards and Technology* – órgão americano) para escolha do novo algoritmo de criptografia simétrica que substituiria o DES. Para executar o ataque eles desenvolveram um modelo para representar um algoritmo de criptografia com um grafo ponderado direcionado. Desta forma, encontrar a melhor característica diferencial de um algoritmo de criptografia corresponde ao problema de encontrar o caminho mais curto em um grafo orientado. Usando este modelo juntamente com a técnica ACO, os autores encontraram melhores características diferenciais do que as publicadas em trabalhos originais aplicados para o SERPENT.

Nedjah e Mourelle mostraram que ACO pode ser utilizado para resolver alguns problemas (matemáticos) difíceis que são fundamentais para alguns criptossistemas (chaves públicas). Por exemplo, os autores aplicaram ACO para a busca pela cadeia de adição de menor comprimento nos seguintes trabalhos [54], [55], [56] e [57]. Ao encontrarmos a cadeia de adição de menor tamanho, estamos reduzindo o número de multiplicações modulares. Como consequência, a exponenciação modular pode ser implementada mais eficientemente.

Nos últimos anos, alguns trabalhos foram propostos nesta linha. Uddin e Youssef [58] aplicaram ACO para criptoanálise do esquema de identificação de *Pointcheval*, esquema baseado no problema de *perceptrons* permutados, que é um problema NP-completo. Esta técnica parece ser bem adaptada para dispositivos com poucos recursos como *smart cards* [59]. Embora já houvesse ataques demonstrando fragilidade no

esquema de *Pointcheval*, os autores mostraram com esse trabalho que mais uma meta-heurística pode ser aplicada para problemas de criptologia. A desvantagem para as outras abordagens anteriormente aplicadas a este problema está na sensibilidade dos parâmetros do ACO.

Fidanova [60] também propôs um modelo de ACO para tratar do MKP (*Multiple Knapsack Problem*). Este é um subconjunto do problema da mochila e pode ser visto como um modelo genérico para qualquer problema binário com coeficiente positivo. No quesito criptologia, resolver o problema MKP significa quebrar a segurança do criptossistema de chave pública baseado no problema da mochila. Os resultados encontrados pela autora demonstram que o uso de modelos probabilísticos (caso do ACO) permite aumentar o potencial de solução para o problema MKP.

Recentemente, Khan, Shahzad e Khan [61] propuseram pela primeira vez o uso de ACO binário para criptoanálise do DES com quatro rodadas apenas. O espaço de busca no caso é um grafo dirigido que é construído para busca eficiente da chave secreta. A função de aptidão é o número do mesmo bit em posições idênticas entre o texto cifrado original C_s (gerado usando a chave original) e o texto cifrado candidato C_t (gerado pelo uso de chaves geradas a partir do caminho completo de uma formiga). Assim, a função de aptidão é $fitness = \bar{o}/64$, onde $\bar{o}$ é a quantidade dos mesmos bits nas posições idênticas em C_s e C_t. Os autores também utilizaram um valor de uma função heurística para ajudar na eficiência dos resultados do ACO. Essa função ajuda na decisão da formiga sobre qual direção se mover no caminho (probabilisticamente). Os resultados alcançados neste trabalho apontam para uma técnica eficiente para criptoanálise de cifradores em bloco. Certamente, o resultado ainda é imaturo, ou seja, não é suficiente para embasar afirmações mais abrangentes (por exemplo, para todos os cifradores de bloco, como o AES).

E. Autômatos Celulares

Autômato celular (CA) nada mais é do que um modelo discreto que contempla células simples arranjadas de forma regular (conhecidas também como grelha). Quatro propriedades ajudam a categorizar os CAs: geometria celular, especificação de vizinhança, número de estados por célula e as regras para alcançar o próximo estado. As regras são determinísticas. Para as grelhas é possível existir um número finito de dimensões. O próximo estado de um CA depende do estado corrente e a regra a ele aplicada. Cada vez que as regras são aplicadas sobre a grelha completa, uma nova geração é produzida.

Wolfram continua sendo um dos mais influentes pesquisadores de CA. Suas ideias impulsionaram as pesquisas com CA para várias áreas e as primeiras ideias sobre possíveis aplicações de CA em criptologia foram propostas em um de seus trabalhos [62]. Na mesma época, Guan [63] apresentou o primeiro modelo de criptossistema de chave pública usando CA. Para evitar que as mensagens pudessem ser manipuladas e fossem falsificadas, foi proposto um modelo no qual o remetente envia a mensagem concatenada com seu nome codificado com a chave privada. Dessa forma, ao receber a mensagem, o remetente tem condições de saber se a mensagem é autêntica ao descobrir o nome do autor e decifrar a mensagem com a sua chave pública. Guan credita a segurança do modelo à dificuldade implícita no problema de resolver um sistema de equações não lineares (um problema NP-completo). Porém, a complexidade envolvida no desenvolvimento de um criptossistema baseado em CA é muita alta, sendo a questão da irreversibilidade seu maior desafio.

Acompanhando a ideia dada por Wolfram em [64] sobre o uso de CA para geradores de números aleatórios (PRNG – *Pseudo-Random Generator*), alguns trabalhos foram propostos mostrando resultados expressivos. Hortensius et. al. [65] apresentaram uma variação na técnica BIST *(built-in self-test)*, a qual é baseada em um gerador pseudoaleatório distribuído derivado de um CA unidimensional (vetor). O modelo é uma técnica BIST alternativa na qual pode encontrar aplicação como uma alternativa para BILBO *(built-in logic block observation)* ou esquemas similares.

Bardell [66] explorou a similaridade de transformação entre a matriz de transição de um CA linear e aquela de um LFSR (*Linear Feedback Shift Register*), que é um registrador de deslocamento no qual o bit de entrada é determinado pelo valor do ou-exclusivo de alguns dos seus bits). No trabalho também é mostrado que a sequência de bits de um estágio de um CA é idêntica ao do LFSR. Para calcular a rotação de fase entre as sequências de bits, emitidos através dos diferentes estágios de um CA linear, foi usado o conceito de logaritmos discreto de um polinômio binário. Segundo o autor, o fluxo de bits gerado através de cada estágio de um CA linear é idêntico ao fluxo de bits gerado através do LFSR. Ao mostrar que é possível calcular o deslocamento de fase (por ser constante) em algumas implementações de CA, este trabalho desestimulou os esforços para usar CA como PRNG em aplicações BIST. O problema estava na aparente aleatoriedade do deslocamento de fases entre a sequência de bits de saída e os vários estágios.

Meier e Staffelbach [67] também analisaram sequências pseudoaleatórias geradas através de CA. Eles propuseram um método para reconstrução do estado inicial de um CA baseado na sequência de bits gerada pelo CA. A consequência deste trabalho é a mesma do trabalho de Bardell.

Nandi, Kar e Chaudhuri [68] propuseram um elegante esquema de baixo custo usando autômatos celulares programáveis (PCA – *Programmable Cellular Automata*) construídos ao redor das regras 90 e 150. Chamados de PCA de dois estágios e PCA com ROM (*Read Only Memory*), eles foram usados para gerar a chave (segura) em cifradores de fluxo e também de blocos. O baixo custo está na adaptação do modelo para implementação em VLSI. Porém, em [69] foi provado que o modelo era inseguro por causa da linearidade presente no CA. Assim, eles mostraram a importância da não-linearidade para cifradores usando CA. Mas, inserir não-linearidade deve ser um processo cuidadoso para não alterar o

ciclo natural (estrutura) do CA.

Mihaljevic [69] apresentou um modelo melhorado para a proposta apresentada em [68]. Porém, em [71] foi apresentado um modelo de cifrador de blocos usando CA no qual os autores embutiram uma transformação não-linear. Os autores propuseram um modelo de CA não-linear (CA não-linear é composto de operação ou-exclusivo mais outras operações) reversível que pode operar com 128 bits de entrada. Existe uma transformação que é chamada de transformação fundamental que é executada para todos os bits (ou 16 bytes que é o tamanho do estado). Assim, para fazer a codificação e a decodificação de forma idêntica nos blocos de bits, a regra 153 é repetida 8 vezes. O resultado positivo para esta abordagem está na agilidade do cifrador.

Tomassini e Perrenoud [73] propuseram um criptossistema de chave única baseado em CA não uniforme de uma e duas dimensões obtidos pela evolução de um GA específico para uso com tabelas de regras (esse GA é conhecido como programação celular). O esquema de codificação é baseado na geração de sequências de bit pseudoaleatórias através de CA. O objetivo do GA é evoluir boas regras para um CA não-uniforme (isto é, regras que dão origem a sequências de alta qualidade de números aleatórios). Segundo os autores, após criteriosa criptoanálise, o modelo apresentou boa qualidade para sequências de bits pseudoaleatórias (principalmente usando duas dimensões) e, portanto, é um bom candidato para gerar chaves seguras. Eles também destacaram para o baixo custo de implementação do modelo em hardware, sendo uma boa abordagem para muitas aplicações (por exemplo, aplicações sobre VLSI). Em [74], os autores apresentaram uma extensão do modelo de Tomassini e Perrenoud no qual, através de programação celular, são encontrados novos conjuntos de regras para os CAs. Isso tornou o modelo de criptografia mais robusto e resistente à tentativa de quebra da chave.

Bao [75] fez uma análise da segurança dos modelos criptográficos baseados em CA e criticou veemente a comunidade científica por não apresentar a criptoanálise feita sobre os modelos criptográficos apresentados. Ele mostrou que a maioria dos modelos é vulnerável ao ataque *choosen-plaintext*. Tal deficiência é decorrente da fragilidade do processo de geração das regras para o modelo CA criptográfico. Apesar de ter mostrado as deficiências, Bao também mostrou que, quando o conjunto de regras é bem analisado e criteriosamente escolhido, é possível mitigar o problema. Assim, é possível aproveitar as vantagens dos CAs para uso em criptologia tais como, a facilidade em gerar padrões pseudoaleatórios e a eficiência quando implementado em hardware.

Fúster-Sabater e Caballero-Gil [76] modelaram uma classe de geradores de sequência não-linear em termos de CA linear. O trabalho considerou a linearização de geradores de sequência pseudoaleatórias com base em domínios finitos. O objetivo básico do algoritmo é que geradores concebidos como complexos modelos não-lineares podem ser escritos em termos de simples modelos lineares.

Certamente existem vários outros trabalhos que foram apresentados recentemente usando CA para desenvolver criptossistemas, para fazer parte deles através de geradores de sequências pseudoaleatórias ou construção de S-boxes. Para exemplificar, citamos o trabalho de Szaban e Seredynski [77] no qual foram construídas *S-boxes* (o principal elemento de cifradores de blocos) baseadas na aplicação de CA. Eles encontraram bons resultados e confirmaram que CA é hábil para executar eficientemente funções *Booleanas* correspondentes às clássicas *S-boxes*.

F. Computação Baseada em DNA

A computação baseada em DNA foi originalmente proposta por Leonard Adleman em [78] e fez nascerem novas perspectivas para os vários problemas difíceis de computação, como mostrou Lipton em [79] e Bone, Dunworth e Lipton em [80]. O potencial desse paradigma está no maciço paralelismo do DNA.

Por ter questões de alta complexidade (principalmente para criptoanálise), aplicações em criptologia surgiram rapidamente tendo Boneh et. al. [81] como precursores. O primeiro modelo (matemático) usando computação molecular (outro nome para computação baseada em DNA) foi idealizado para quebrar o DES. Porém, o modelo era genérico o suficiente para ser usado contra qualquer tipo de criptossistema que usasse chaves de até 64 bits. Por ser um trabalho pioneiro, ele evidenciou o potencial da computação DNA para criptoanálise de criptografia moderna (na época, o DES era o algoritmo padrão, adotado no mundo todo). Adleman et. al. [82] deu continuidade ao trabalho de Bone et. al. usando o modelo conhecido como *"sticker"*. Eles aplicaram força-bruta utilizando o ataque *plaintext-ciphertext*. Neste tipo de ataque o criptoanalista obtém um texto em claro e seu texto cifrado correspondente e tenta determinar a chave usada para a codificação. O algoritmo proposto tem três passos principais:

- passo de entrada: inicialize os filamentos de memória para formar complexos de memória representando todas as 2^{56} chaves (a chave do DES);
- passo de codificação: em cada complexo de memória, calcule o texto cifrado correspondente à codificação do texto em claro sobre a chave do complexo;
- passo de saída: selecione o complexo de memória cujo texto cifrado corresponde ao texto em claro e leia a chave correspondente.

Apesar de serem teóricos, esses dois primeiros trabalhos mostraram à criptologia que é preciso levar em consideração que o aumento do potencial computacional usando super-paralelismo é um problema para os algoritmos criptográficos. Mesmo que não seja possível construir um computador molecular, o super-paralelismo apresenta forte indícios que podem prejudicar os algoritmos de criptografia existentes (até o presente momento nenhum trabalho mostrou vulnerabilidades no AES, mas, se existir, elas poderão ser atacadas usando super-parelelismo). De fato, as 2^{56} chaves do DES são testadas simultaneamente, além disso, soluções determinísticas não são necessárias, basta apenas uma baixa

taxa de erros.

Porém, computação molecular não tem sido proposta apenas para criptoanálise. Suas qualidades demonstram potencial para novos sistemas criptográficos, como mostrou o trabalho de Gehani, LaBean e Reif [82]. Eles apresentaram uma investigação inicial sobre o uso de DNA para segurança através de dois métodos: (i) um modelo de criptografia DNA baseada no OTP (*one-time pad*) e (ii) em métodos de esteganografia DNA. No caso "i" o uso de criptossistemas baseados nos conceitos do OTP torna o sistema inquebrável. Porém, na prática, os computadores atuais não tem capacidade para usufruir dessa segurança por causa do tamanho da OTP. Entretanto, é conhecido que o DNA possui alta capacidade de armazenamento de informações. Assim, mesmo uma extremamente pequena quantidade de DNA, é suficiente até mesmo para grandes OTPs. Assim, novamente é demonstrado o poder da computação molecular. No caso "ii", a ideia de esconder informação usando DNA já tinha sido proposta em [83] através do uso de um microponto (*microdot* – uma técnica desenvolvida para encobrir mensagens que foi muito utilizada por espiões alemães na segunda guerra mundial para transmitir informações secretas).

Em [82] também foi utilizada uma implementação em codificação binária usando DNA artificial através do uso de um chip baseado na tecnologia de *microarray* de DNA.

Shimanovsky et. al. [83] desenvolveram duas técnicas para esconder dados em DNA e RNA (*Ribonucleic acid*). A primeira técnica esconde dados em DNA não codificado tal como em regiões não transcritas e não traduzidas e também em DNA não genético (computação baseada em DNA é vista como material não genético). A segunda técnica pode ser usada para embutir informação diretamente nos segmentos genéticos. Essas técnicas podem ajudar na proteção de dados usando DNA como meio de armazenagem.

Nos últimos anos vários trabalhos têm sido apresentados nesta linha tendo a criptografia como foco principal. Em [85], os autores usaram biotecnologia para codificação e decodificação através de sondas DNA e embutiram o texto cifrado em um modelo de chip DNA especialmente desenvolvido para este trabalho, cujo alvo foi a criptografia simétrica.

Tornea e Borda [86] apresentaram dois algoritmos criptográficos originais baseados nas ideias dos algoritmos de chave pública e no princípio do OTP. No primeiro modelo é realizada a codificação de dados binários seguida da transformação em sequência digital DNA. No segundo modelo, os dados binários são transformados em estruturas de DNA geradas quimicamente e conhecidas como tiles (por causa do formato que lembra telhas). Segundo os autores, os dois algoritmos podem ser embutidos em *microarrays*, porém, o custo da tecnologia ainda é muito alto, inviabilizando testes reais. Mesmo assim, eles mostraram que é possível construir algoritmos criptográficos com alta segurança indicando que esta é uma área bastante promissora.

Hirabayashi, Kojima e Oiwa [87] também propuseram um modelo elegante de criptossistema no qual fizeram uso da computação molecular para gerar chaves verdadeiramente aleatórias. Eles apresentaram um novo algoritmo multicamadas implementável a fim de usar hibridização aleatória (relacionada à hibridização de quatro filamentos oligonucleotídeos) de tiles de DNA para a geração da chave aleatória. A hibridização de eventos DNA é vista como um útil processo físico para geração de números aleatórios. O modelo apresentou significativa tolerância a erros.

Um dos últimos trabalhos na literatura é uma aplicação de DNA para o desenvolvimento de criptossistema de chave pública [88]. Assim como na criptografia de chave pública tradicional, a chave de codificar é diferente da chave de decodificação. Existem várias partes da chave pública para codificação e apenas uma parte da chave privada para decodificação. Ao contrário dos sistemas convencionais, as chaves DNA-PKC são sondas de DNA, em vez de uma cópia dos dados. Assim, a chave privada é entregue fisicamente e (por exemplo, através de um *biochip*), e não por via eletrônica.

G. Sistemas Imunológicos Artificiais

Baseados no sistema imunológico humano (HIS), essa abordagem da CI cresce rapidamente como um novo campo de pesquisa em inteligência artificial (AI). O HIS contém aspectos importantes, principalmente relacionados à sua capacidade de tratar problemas complexos e de forma distribuída. Entretanto, elas têm sido pouco empregadas em criptografia.

Por ter algumas características que se assemelham àquelas encontradas em CE, algumas aplicações áreas parecem ser mais promissoras, por exemplo, a de criptoanálise. Neste sentido, um dos primeiros trabalhos apresentados usa sistemas imunológicos artificiais (AIS) para encontrar a cadeia de adição ótima para o problema da exponenciação de corpos finitos [89]. Os resultados mostraram que o esquema obteve cadeias de tamanho mais curto para expoentes tipicamente usados para calcular os inversos multiplicativos no corpo para aplicações de correção de erros e criptografia de curvas elípticas. Os mesmos autores estenderam o trabalho para expoentes de tamanhos moderados (até 20 bits) e para valores altos, como aqueles usados em aplicações criptográficas (entre 128 e 1024 bits). Nesse trabalho, o algoritmo imuno-inspirado foi capaz de encontrar quase todas as cadeias de adição ótimas para qualquer expoente fixo *e* menor que *4096* bits. Assim, a taxa de sucesso encontrada foi considerada de alto sucesso, pois alcançou 99.6%.

Jacob et. al. [91] apresentaram um modelo de classificador imunológico com capacidade para detectar informações escondidas em imagens. Os resultados alcançaram índices de até 91% de acerto usando sensores que nunca tinham sido expostos ao problema.

Recentemente, Ali et. al. [92] aplicaram AIS para atacar o DES, porém com apenas quatro rodadas. Os resultados mostram que sistemas imuno-inspirados possuem qualidades que precisam ser estudadas e desenvolvidas a fim de alcançar maior utilidade perante a classe de problemas da criptologia.

H. Funções Hash

O mundo aguarda a definição do NIST (*National Institute of Standards and Technology*) sobre o novo padrão de funções *hash*. A motivação foi a descoberta de vulnerabilidades nos modelos vigentes (por exemplo, MD5 e SHA-1). Apesar de nenhum trabalho dos cinco finalistas contemplarem técnicas de CI, muitos trabalhos têm sido apresentado na literatura mostrando que é possível desenvolver funções *hash* com resultado satisfatório. Inserimos esta subseção para funções *hash* por termos encontrado várias abordagens usando técnicas diferentes de CI e pela importância atual da discussão sobre a qualidade das funções *hash*.

Estébanez, Hernández e Ribagorda [93] usaram programação genética para evoluir uma função *hash*. Eles deram ênfase ao efeito avalanche para mitigar o problema de alteração de informações no texto original. Assim, um único bit alterado gerará uma saída totalmente diferente. A função de aptidão evolui através do cálculo da distância de *Hamming* para duas saídas geradas a partir de duas entradas iguais que têm apenas um bit invertido. O processo é repetido *t* vezes e a cada vez que uma distância de *Hamming* entre 0 e 32 é obtida ela é armazenada. O objetivo da função de aptidão é um efeito avalanche perfeito. Para isso, a distância de *Hamming* deve se ajustar à probabilidade da distribuição teórica de Bernoulli (1/2,32). Assim, a aptidão de cada indivíduo é calculada pela soma de dois fatores: o primeiro mede o quão perto está de 16 (16/32 = 1/2) a média das distâncias de *Hamming* calculada e o segundo, é usado o teste estatístico do chi-quadrado X^2 que mede a distância da distribuição observada das distâncias de *Hamming* da distribuição de probabilidade teórica de *Bernoulli* (1/2,32). De acordo com os experimentos, os resultados encontrados foram satisfatórios quanto ao número de colisões por *hash* e o tempo de execução da função quando comparados a outros modelos de função *hash*.

Yun-qiang e Ai-lan [94] apresentaram criptoanálise usando GA para aumentar a possibilidade de colisões próximas para um dos algoritmos (*Hamsi-256*) que ficaram entre os quatorze escolhidos para a segunda rodada da chamada pública do NIST. Os autores usaram o ataque diferencial obtendo algumas propriedades não aleatórias do *Hamsi-256*. A base da avaliação da aptidão dos indivíduos é: se uma diferença de entrada tem alta probabilidade de obter uma quase colisão, usando pares de entradas aleatórios sobre a diferença de entrada, então é fácil encontrar a menor distância de *Hamming* entre os pares de valores para a função de compressão. O resultado encontrado após a execução do GA para a busca da quase colisão foram os melhores dentre os trabalhos que foram apresentados anteriormente para a mesma função *hash*. Assim, o trabalho alcançou 2^{-23} de probabilidade com quatro números de diferenças no valor de entrada.

ANNs também têm sido usadas para desenvolver funções *hash*. É possível encontrar vários trabalhos nesta direção. Um fato observado é que a maioria dos trabalhos usa algum tipo de ANN combinada com a teoria do caos. Ou seja, nesses casos a ANN é chamada rede neural caótica (CNN – *Chaotic Neural Network*). O uso da camada caótica para desenvolver funções *hash* é estimulado por algumas das propriedades implícitas na teoria do caos: sensibilidade às condições iniciais e aos parâmetros, comportamento aleatório, ergodicidade e direção-única. Assim, Xiao e Liao [95] propuseram um modelo combinando *hash* e esquema de codificação através de uma CNN. A rede neural é constituída de 4 camadas na qual a primeira camada é a entrada de um texto em claro de 640 bits. A segunda camada possui 64 neurônios, a terceira consiste de 128 neurônios e gera o resultado final do *hash.* A quarta camada, com 640 neurônios, é a responsável pela saída do texto cifrado. O sistema caótico é usado para distribuir os pesos e permutar a relação correspondente entre o texto em claro e o texto cifrado. O resultado prático do modelo demonstra que a função atende a propriedade de mão-única, assim como também é mostrado no trabalho que apenas 1 bit modificado pode gerar grande diferença no final. Esse efeito é importante para que a função apresente a menor chance possível de colisão e seja resistente ao ataque do aniversário.

Em [96], os autores apresentaram um algoritmo para construção de uma função *hash* com chave usando uma RBF (*Radial Basis Function* – modelo de rede neural no qual os neurônios da camada escondida contêm funções de transferência gaussiana cujas saídas são inversamente proporcionais à distância do centro do neurônio). Um mapeamento caótico foi adicionado tornando o modelo bem parecido com o modelo de Xiao e Liao.

Em [97], Lian, Sun e Wang construíram uma função *hash* baseada em uma ANN de três camadas. As três camadas foram usadas para realizar confusão, difusão e compressão dos dados. Para melhorar a sensibilidade do valor inicial e dos parâmetros iniciais foi adicionado um mapeamento caótico.

Xiao, Liao e Wang [98] propuseram um modelo de função *hash* parametrizado por uma chave. O modelo também foi baseado em CNN, porém, o que distingue esse modelo para os outros apresentados usando CNN é a abordagem paralela. As análises de resistência e robustez obtiveram os mesmos expressivos resultados dos modelos anteriores.

Recentemente, novos trabalhos têm sido propostos, porém, ao analisá-los, nós verificamos que os mesmos são modificações dos já apresentados ou então, são os mesmos algoritmos usando abordagens diferentes de ANNs. Por exemplo, em [99], os autores construíram uma função *hash* baseada em uma FNN de duas camadas que facilita a inserção de mais neurônios para aumentar o tamanho da chave. Uma nova abordagem para o mapeamento caótico foi usado neste trabalho conhecido como PWLCM (*piecewise linear chaotic map*).

Por fim, o trabalho de Li, Deng e Xiao [100] mostra um novo algoritmo de *hash* baseado novamente em CNN. A melhoria encontrada no trabalho está na abordagem de quatro dimensões para o gerador de chaves da CNN, que recebe o nome de 4D OWCML (*4-dimensional and one-way coupled map lattices*), juntamente com o CTM (*chaotic tent map*). Os autores melhoraram significativamente a complexidade espaço-temporal da dinâmica não linear e da mistura dos dados. Para a CNN foram usadas apenas duas camadas, uma de entrada e outra de saída. Os resultados das análises realizadas mostraram uma forte sensibilidade à mensagem e à chave secreta, forte difusão e capacidade de confusão, além da resistência a colisão.

Autômatos celulares também são usados para funções *hash*. Em [101] foram introduzidas as primeiras ideias sobre o uso

de CA para funções *hash*. Porém, este modelo foi criptoanalisado em [102]. O resultado mostrou vulnerabilidades no primeiro modelo e, por isso, os autores propuseram um novo modelo conhecido como *Celhash*. Hirose e Yoshida [103] propuseram uma função de mão-única baseada em um CA de duas dimensões. Em [104] foi proposto um algoritmo baseado em CA para uma função *hash* de mão-única com maior agilidade o que o tornou útil para implementação em hardware.

Del Rey [105] estudou o desenvolvimento de protocolos de autenticação de mensagens usando CA. O foco do trabalho foram os CAs com memória nos quais as funções de transição são consideradas sobre o conjunto definido $GF(2^n)$, onde $n = 1,2,4,8,16,32$. A chave secreta tem 160 bits de tamanho e o protocolo envolve três tipos de CA: o CA elementar, com a regra 150, e dois CA de memória. O tamanho do *hash* calculado é de 160 bits. Os resultados dos testes mostraram que o modelo é resistente a ataques de força bruta e criptoanálise diferencial.

IV. Discussão

Observando os trabalhos existentes na literatura, percebe-se que, apesar do potencial para aplicação de CI à criptologia, esta abordagem está caminhando a passos lentos. Embora existam dificuldades em trabalhar com as técnicas de CI aplicadas à criptologia (a dificuldade no mapeamento computacional é uma delas), acreditamos que exista espaço para novas abordagens criptográficas ou de criptoanálise bio-inspiradas e fazemos coro ao chamado de John Clark [14] à comunidade de CI para avaliar os problemas complexos da criptologia que podem servir de alvo para as técnicas de CI.

Certamente, a representação dos problemas criptográficos usando as técnicas de CI é um problema importante e precisa ser levado em consideração. Geralmente, as abordagens criptográficas são processos discretos e aleatórios. A maioria absoluta das representações encontradas é binária. É conhecido que representação binária não é a melhor representação para GA. O problema está na aplicação dos operadores de crossover e mutação. Por exemplo, suponha que temos um vetor binário de 8 posições que represente um cromossomo (cada cromossomo é uma solução candidata). Em determinada iteração, aplicamos o operador de mutação que faz a troca de 0 por 1, e vice-versa, em uma posição apenas. A função de aptidão avalia a distância de *Hamming* de uma solução comparando-a com a função objetivo. Suponha ainda que, cada posição do vetor mapeia uma posição no espaço de busca bidimensional. Assim, ao aplicar a mutação nos bits, podemos alterar totalmente a direção da busca diminuindo a qualidade da função de aptidão da solução. Esse caso é indesejável, pois pode dificultar a busca e até mesmo evitar o resultado esperado.

Além das dificuldades em representar, há ainda o problema da configuração dos parâmetros. Todas as técnicas de CI possuem muitos parâmetros que precisam ser ajustados de acordo com cada problema. Para dificultar, existem situações em que o ajuste dos parâmetros precisa ser feito dinamicamente e continuadamente. Mas, qual a melhor configuração? Não existem valores mágicos, é preciso avaliar. É possível usar algumas ferramentas propostas na literatura que ajudam na busca pelos valores mais apropriados para uso com algumas técnicas [108]. Porém, nem sempre há condições de avaliação teórica (matemática). Assim, alguns valores que são apresentados na literatura foram definidos de forma experimental. É possível ainda que, para os problemas criptográficos, esses valores pré-estabelecidos não levem ao sucesso na aplicação da técnica. Portanto, a definição de alguns parâmetros é um desafio a mais.

É preciso lembrar que as técnicas de CI são técnicas meta-heurísticas e não-determinísticas. Assim, quando aplicadas à criptoanálise, é possível que a solução (a chave) nunca seja encontrada. Porém, pode ser que a melhor solução seja uma aproximação do resultado esperado e, assim, podemos usá-la como ponto inicial de algum outro processo de criptoanálise (um processo estatístico, por exemplo). Desta forma, é possível alcançar o resultado mais rapidamente que com o uso de um processo de força bruta. Entretanto, para aplicações em criptografia, há necessidade de alguma abordagem que possibilite a inserção de determinismo para que sejam viáveis os processos de codificação e decodificação. Uma alternativa para estas aplicações é a adoção de autômatos celulares. A maioria das regras existentes para formação de autômatos celulares acarreta em perda de informações durante a sucessão de gerações. Assim, para que seja possível a reversibilidade do processo de codificação, é necessário o uso de regras específicas (reversíveis) como aquelas encontradas em [107].

Outro ponto que visualizamos com potencial para exploração é a hibridização de técnicas de CI para um mesmo problema. Certamente, este tema é aplicável para vários tipos de problemas que estão além do escopo criptográfico. Mas, pela complexidade implícita em determinados problemas de criptologia, acreditamos que, o uso de mais de uma técnica para um mesmo problema é passível de estudo, principalmente pelo fato de que, na literatura, essa é uma abordagem raramente utilizada. Para sermos mais claros, algumas técnicas de CI podem ser usadas para ajudar na determinação dos melhores valores dos parâmetros de entrada de outra técnica. Por exemplo, na literatura encontramos trabalhos que combinam usando GA e ANN, mas existem outros casos.

Assim como a computação quântica tem despertado interesse pelo potencial computacional, a computação baseada em DNA (ou molecular) também surge como uma candidata a um novo modelo com elevado potencial para resolver problemas de tratamento inviável com a computação tradicional. Certamente, o potencial computacional da abordagem quântica já é suficiente para novas e promissoras perspectivas. Porém, em concordância com as considerações apontadas por Ezziane [106], cremos que o uso de CI juntamente a este novo paradigma de computação possui grande potencial a ser explorado. Por exemplo, os trabalhos que apontamos durante esta revisão (Subseção F) são baseados apenas nas características pertinentes à abordagem molecular (paralelismo e capacidade de armazenamento de informações). Entretanto, a inserção de alguma inteligência poderia aproveitar melhor as qualidades e proporcionar resultados mais interessantes do que aqueles já apresentados, mesmo que de forma teórica (matemática) ou na forma de chips moleculares.

Visualizamos ainda a necessidade de comparações entre as técnicas aplicadas aos problemas criptográficos. Ao compará-las, seria possível identificar as qualidades de cada uma, assim como, construir um mapeamento das aplicações de CI para cada problema. Certamente, a escolha da técnica ideal para cada tipo de problema seria facilitada e, dessa forma, os recursos de CI seriam mais atraentes para a comunidade de segurança da informação.

V. Conclusão

Fazendo uma breve comparação sobre o uso das técnicas nós podemos observar que: (i) ANN e CA são usados principalmente para construir sistemas criptográficos, (ii) computação evolutiva e AIS são usados principalmente para criptoanálise e (III) computação molecular tem sido aplicada para ambos os casos. Porém, para computação molecular os últimos trabalhos têm sido sobre criptografia, algo que pode ter relação com o potencial de paralelismo desta técnica. Outra observação é a grande quantidade propostas usando ANN, GA e CA em relação às outras abordagens.

Acreditamos que há grande potencial de aplicação de AIS em criptologia. Em nossas pesquisas, ficou evidente que AIS ainda não é uma técnica muito utilizada pela comunidade científica nas aplicações em criptologia. Entendemos ser relevante comparar esta técnica com outras e avaliar seu potencial de maneira mais profunda, bem como avaliar o potencial da aplicação de CI a problemas de criptologia de um modo geral, observando os pontos discutidos na Seção IV.

É notável o aumento de publicações de trabalhos sobre CI aplicada à criptologia recentemente. Porém, a convergência das duas áreas ainda padece de alguns problemas como a distância entre os grupos de CI e de criptologia. Acreditamos que, se houver maior integração entre os dois grupos, certamente surgirão respostas mais contundentes a respeito do potencial da CI aplicado à criptologia.

Referências

[1] A. Joux, *Algorithmic cryptanalysis.* CRC Press series on cryptography and network security, 2009.

[2] R. A. Mollin, *An Introduction to Cryptography* – Second Edition. Taylor and Francis Group, 2007.

[3] S. Rusel and P. Norvig, *Artificial Intelligence*, Person Education Inc., 1995.

[4] S. Haykin, *Neural Networks, A Comprehensive Foundation*, Prentice Hall, 1999.

[5] D. E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Publishing Company Inc. 1989.

[6] G. Paun, G. Rozenberg and A. Salomaa, *DNA Computing: New Computing Paradigm.* Springer, 1998.

[7] H. Gutowitz, *Cellular Automata: theory and experiment.* The Mit Press, 1991.

[8] J. E. Kennedy, R. Eberhart; Y. Shi, *Swarm Intelligence.* Morgan Kaufmann, 2001.

[9] M. Dorigo and T. Stützle, *Ant Colony Optimization.* The Mit Press, 2004.

[10] L. N. de Castro and J. Timmis, *Artificial Immune Systems: A New Computational Intelligence Approach*. Springer, 2002.

[11] E.C. Laskari, G.C. Meletiou, D.K. Tasoulis, and M.N. Vrahatis, "Studying the performance of artificial neural networks on problems related to cryptography," *Neural Networks*, vol. 7, 2006, pp. 937 - 942.

[12] N. Liu and D. Guo, "Security Analysis of Public-Key Encryption Scheme Based on Neural Networks and Its Implementing," *Computational Intelligence and Security*, p. 443–450, 2007.

[13] S. Lian, "A block cipher based on chaotic neural networks," *Neurocomputing*, vol. 72, Jan. pp. 1296-1301, 2009.

[14] J. A. Clark, "Invited Paper. Nature-Inspired Cryptography: Past, Present and Future," *Citeseer*, pp. 1647-1654, 2003.

[15] R. Spillman, "Cryptanalysis of Knapsack Ciphers Using Genetic Algorithms", *Cryptologia*, vol. 17, issue 4, SN 0161-1194, pp. 367-377, 1993.

[16] R. A. J. Matthews, "The use of genetic algorithms in cryptanalysis", Cryptologia, vol. 17, issue 2, pp. 187-201, 1993.

[17] A. Clark, "Modern optimisation algorithms for cryptanalysis," *Intelligent Information Systems, 1994. Proceedings of the 1994 Second Australian and New Zealand Conference on*, IEEE, 1994, p. 258–262.

[18] A. Clark, Ed. Dawson, and H. Bergen, "Combinatorial Optimisation and the Knapsack Cipher." *Cryptology*, 20(1), 85-93, 1996.

[19] A. Clark, Ed Dawson, and H. Nieuwland, Cryptanalysis of Polyalphabetic Substitution Ciphers Using a Parallel Genetic algorithm. *In Proceedings of IEEE International Symposium on Information and its Applications*, September 17-20.

[20] J. Kolodziejczyk, J. Miller, and P. Phillips, The application of genetic algorithm in cryptanalysis of knapsack cipher. In V. Krasnoproshin, J. Soldek, J., S. Ablameyko, and V. Shmerko, (Eds.), *Proceedings of Fourth International Conference PRIP '97 Pattern Recognition and Information Processing*, May 20-22, (pp. 394-401).Poland: Wydawnictwo Uczelniane Politechniki Szczecinskiej, 1997.

[21] A. Clark and Ed. Dawson, "A Parallel Genetic Algorithm for Cryptanalysis of the Polyalphabetic Substitution Cipher." *Cryptologia*, 21 (2), 129-138, 1997.

[22] T. Bagnall, G. P. McKeown, and V. J. Rayward-Smith, "The cryptanalysis of a three rotor machine using a genetic algorithm," *Proceedings of the Seventh International Conference on Genetic Algorithms (ICGA97), San Francisco, CA*, 1997.

[23] A. Clark and Ed Dawson, Optimization Heuristics for the Automated Cryptanalysis of Classical Ciphers. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 28, 63-86, 1998.

[24] F. Glover, "Tabu Search: A Tutorial", *Interfaces*, 20(4):74-94, 1990.

[25] I. F. T. Yaseen, and H. V. Sahasrabuddhe, A genetic algorithm for the cryptanalysis of Chor-Rivest knapsack public key cryptosystem (PKC). *In Proceedings of Third International Conference on Computational Intelligence and Multimedia Applications*, (pp. 81-85), 1999.

[26] J. Hernàndez, J. M. Sierra, P. Isasi, and A. Ribagorda, "Genetic cryptanalysis of two rounds TEA." *Lectures Notes in Computer Science,* 2331:1024-1031, 2002.

[27] J. Hernàndez, P. Isasi, and A. Ribagorda, "Easing collision finding in cryptographic primitives with genetic algorithms," *wcci*, IEEE, 2002, p. 535–539, 2002.

[28] J. C. Hernández and P. Isasi, "New results on the genetic cryptanalysis of TEA and reduced-round versions of XTEA," *Evolutionary Computation, 2004. CEC2004. Congress on*, IEEE, p. 2124–2129, 2004.

[29] A. Garrett, J. Hamilton, and G. Dozier, "A Comparison of Genetic Algorithm Techniques for the Cryptanalysis of TEA," *International journal of intelligent control and systems*, vol. 12, p. 325–330, 2007.

[30] J. A. Clark, "Metaheuristic Search as a Cryptological Tool," U*niversity of York department of computer science-publications-ycst*, pp. 01-191, 2002.

[31] F. Yang, J. Song, and H. Zhang, "Quantitative Cryptanalysis of Six-Round DES Using Evolutionary Algorithms," *ISICA 2008, LNCS 5370*, pp. 134–141, 2008.

[32] H. M. H. Husein, B. I. Bayoumi, F. S. Holail, B. E. M. Hasan, and M. Z. A. El-Mageed, "A Genetic Algorithm for Cryptanalysis of DES-8," *International Journal of Network Security, pp213-219*, vol. 9, pp. 213-219, 2006.

[33] E. Biham and A. Shamir, "Differential cryptanalysis of data encryption standard," pp. 2-21, Springer-Verlag, New York, 1993.

[34] W. Millan, A. Clark, and E. Dawson, "Smart Hill Climbing Finds Better Boolean Functions." *In Workshop on Selected Areas in Cryptology (SAC),* pp. 50-63, Ottawa, Canada, August, 1997.

[35] W. Millan, A. Clark, and E. Dawson, "An Effective Genetic Algorithm for Finding Boolean Functions," *International Conference on*

Information and Communications Security (ICICS), Beijing, China, November, 1997.
[36] N. Nedjah and L. D. M. Mourelle, "Evolutionary Regular Substitution Boxes," *Evolutionary Computation*, vol. 88, pp. 79-88, 2007.
[37] W. Millan, L. Burnett, G. Carter, A. Clark, and E. Dawson, "Evolutionary heuristics for finding cryptographically strong S-boxes," *Information and Communication Security*, p. 263–274, 2004.
[38] J. A. Clark, J. L. Jacob, and S. Stepney "The Design of S-Boxes by Simulated Annealing," *Elsevier*, pp. 1533-1537, 2004.
[39] R. Muthuregunathan, D. Venkataraman, and P. Rajasekaran, "Cryptanalysis of Knapsack Cipher using Parallel Evolutionary Computing," *International Journal of Recent Trends in Engineering,* vol. 1, n° 1, pp. 3-6, 2009.
[40] V. R and M. L. Valarmathi, "Cryptanalysis of S-DES using Genetic Algorithm," *International Journal of Recent Trends in Engineering*, vol. 2, pp. 2-5, 2009.
[41] P. Garg, "Cryptanalysis of SDES via Evolutionary Computation Techniques," *Journal of Computer Science and Information Security*, vol. 1, pp. 117-123, 2009.
[42] P. Garg and A. Shastri, "An Improved Cryptanalytic Attack on Knapsack Cipher using Genetic Algorithm," *International Journal of Information and Communication Engineering*, vol. 3, pp. 449-456, 2006.
[43] R. Venkateswaran and D. V. Sundaram, "Information Security: Text Encryption and Decryption with poly substitution method and combining the features of Cryptography," *International Journal of Computer Applications*, vol. 3, Jun. pp. 28-31, 2010.
[44] J.K. Ambulkar, "Poly Substitution Method for Encryption and Decryption," *International Journal on Computer Science and Engineering*, vol. 02, pp. 1810-1812, 2010.
[45] E. C. Laskari, K. E. Parsopoulos, and M. N. Vrahatis, Particle swarm optimization for integer programming. In Proceedings of the IEEE Congress on Evolutionary Computation, pp. 1576-1581, IEEE Press, 2002.
[46] E. C. Laskari, K. E. Parsopoulos, and M. N. Vrahatis, Particle swarm optimization for minimax problems. In Proceedings of the IEEE Congress on Evolutionary Computation pp. 1582–1587. IEEE Press, 2002.
[47] E. C. Laskari, G. C. Meletiou, Y. C. Stamatiou, and M. N. Vrahatis, "Evolutionary computation based cryptanalysis: A first study," *Nonlinear Analysis*, vol. 63, pp. 823-830, 2005.
[48] N. Nalini and G.R. Rao, "Experiments on cryptanalysing block ciphers via evolutionary computation paradigms," *Proceedings of the 7th WSEAS International Conference on Evolutionary Computing*, World Scientific and Engineering Academy and Society (WSEAS), p. 20–27, 2006.
[49] E. Laskari, G. Meletiou, Y. Stamatiou, and M. Vrahatis, "Cryptography and Cryptanalysis through Computational Intelligence," *Computational Intelligence in Information Assurance and Security*, vol. 49, p. 1–49, 2007.
[50] A. León-Javier, N. Cruz-Cortés, M. Moreno-Armendáriz, and S. Orantes-Jiménez, "Finding Minimal Addition Chains with a Particle Swarm Optimization Algorithm," *MICAI 2009: Advances in Artificial Intelligence*, p. 680–691, 2009.
[51] M. F. Uddin and A. M. Youssef, "Cryptanalysis of simple substitution ciphers using particle swarm optimization," *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, IEEE, p. 677–680, 2006.
[52] M. Russell, J. A. Clark, S. Stepney, "Using Ants to Attack a Classical Cipher Cryptanalysis of Transposition Ciphers," *GECCO 2003, LNCS 2723*, pp. 146-147, 2003.
[53] A. G. Bafghi and B. Sadeghiyan, "Differential model of block ciphers with ant colony technique," *Proceedings of the Second International Symposium on Telecommunications, Iran*, p. 556–560, 2003.
[54] N. Nedjah and L. de M. Mourelle, "Finding minimal addition chains using ant colony," *Intelligent Data Engineering and Automated Learning–IDEAL 2004*, p. 642–647, 2004.
[55] N. Nedjah and L. de M. Mourelle, "Towards Minimal Addition Chains Using Ant Colony Optimisation," *Journal of Mathematical Modelling and Algorithms*, pp. 525-543, 2006.
[56] N. Nedjah and L. de M. Mourelle, "Efficient pre-processing for large window-based modular exponentiation using ant colony," *Knowledge-Based Intelligent Information and Engineering Systems*, Springer, p. 640–646, 2005.
[57] N. Nedjah and L. de M. Mourelle, "Ant Colony Optimisation for Fast Modular Exponentiation using the Sliding Window Method," *Swarm Intelligent Systems*, vol. 147, p. 133–147, 2006.
[58] M. F. Uddin and A. M. Youssef, "Cryptanalysis of Pointcheval's identification scheme using ant colony optimization," *Evolutionary Computation, CEC 2007. IEEE Congress on*, IEEE, 2007, p. 2942–2947, *2007.*
[59] D. Pointcheval, A new identification scheme based on the perceptrons problem, In L. C. Guillou and J. J. Quisquater editors. Advances in Cryptology – EUROCRYPT'95, LNCS 921, pp. 319-328, Springer Verlag, 1995.
[60] S. Fidanova, "Probabilistic Model of Ant Colony Optimization," *LSSC 2007, LNCS 4818*, 2008, pp. 545-552, 2008.
[61] S. Khan, W. Shahzad, and F. A. Khan, "Cryptanalysis of Four-Rounded DES Using Ant Colony Optimization," *Information Science and Applications (ICISA), 2010 International Conference on*, IEEE, 2010, p. 1–7, 2010.
[62] S. Wolfram (Editor), Theory and Applications of Cellular Automata, World Scientific, 1986.
[63] P. Guan, "Cellular Automaton Public-Key Cryptosystem," *Complex Systems*, vol. 1, 1987, p. 51–56, 1987.
[64] S. Wolfram, Random sequence generation by cellular automata, Advances in applied mathematics, vol. 7, issue 2, pp. 123-169 June, 1986.
[65] P. D. Hortensius, R. D. McLeod, W. Pries, D. M. Miller, and H. C. Card, "Cellular Automata-Based Pseudorandom Number Generators for Built-In Self-Test," *IEEE Transactions on ComputerAided Design*, vol. 8, pp. 842-859, 1989.
[66] P. H. Bardell, "Analysis of cellular automata used as pseudorandom pattern generators," *Proceedings. International Test Conference 1990*, 1990, pp. 762-768, 1990.
[67] W. Meier and O. Staffelbach, "Analysis of pseudo random sequences generated by cellular automata", Advances in Cryptology - EUROCRYPT '91, Lecture Notes in Computer Science, vol. 547, pp. 186-199, 1992.
[68] S. Nandi, B. K. Kar, and P. Chaudhuri, "Theory and applications of cellular automata in cryptography," *Computers, IEEE Transactions on*, vol. 43, p. 1346–1357, 1994.
[69] S. Murphy, S. R. Blackburn, and K. G. Paterson, "Comments on theory and applications of cellular automata in cryptography," IEEE Trans. Comp., vol. 46, no. 5, pp. 637–638, 1997.
[70] M. Mihaljević, "An improved key stream generator based on the programmable cellular automata," *Information and Communications Security*, p. 181–191, 1997.
[71] M. Mihaljevic, "Security examination of certain cellular automata based key stream generator", ISITA '96 - 1996 IEEE International Symposium on Informa- tion Theory and Its Applications, Canada, Victoria, B.C., September 1996, Pro- ceedings, pp. 246-249, 1996.
[72] D. Mukhopadhyay and D. Roychowdhury, "Cellular Automata: An Ideal Candidate for a Block Cipher," *ICDCIT*, LNCS 3347, pp. 452-457, 2004.
[73] M. Tomassini, "Cryptography with cellular automata," *Applied Soft Computing*, vol. 1, Aug. 2001, pp. 151-160, 2001.
[74] F. Seredynski, "Cellular automata computations and secret key cryptography," *Parallel Computing*, vol. 30, May. 2004, pp. 753-766, 2004.
[75] F. Bao, "Cryptanalysis of a new cellular automata cryptosystem," *Information Security and Privacy*, Springer, 2003, p. 216–217, 2003.
[76] A. Fúster-Sabater and P. Caballero-Gil, "On the Use of Cellular Automata in Symmetric Cryptography," *Acta Applicandae Mathematicae*, vol. 93, Aug. 2006, pp. 215-236, 2006.
[77] M. Szaban and F. Seredynski, "Cryptographically Strong S-Boxes Based on Cellular Automata," *Cellular Automata*, 2010, p. 478–485, 2010.
[78] L. Adleman, "Molecular computation of solutions to combinatorial problems," *Science*, vol. 266, 1994, pp. 1021-1024, 1994.
[79] R. J. Lipton, "Using DNA to solve NP-complete problems," *Science*, vol. 268, 1995, p. 542–545, 1995.
[80] [1] D. Boneh, C. Dunworth, R.J. Lipton, and J. Sgall, "On the computational power of DNA," *Discrete Applied Mathematics*, vol. 71, 1996, p. 79–94, 1996.

[81] D. Boneh, C. Dunworth, and R. J. Lipton, "Breaking DES using a molecular computer," *Technical Report CS-TR-489-95,* Princeton University, 1995.
[82] A. Gehani, T. LaBean, and J. Reif, "DNA-based cryptography," *5th DIMACS workshop on DNA Based Computers, MIT*, Citeseer, 1999, pp. 167-188, 1999.
[83] C.T. Clelland, V. Risca, and C. Bancroft, "Hiding messages in DNA microdots," *Nature*, vol. 399, Jun. 1999, pp. 533-4, 1999.
[84] B. Shimanovsky, J. Feng, and M. Potkonjak, "Hiding data in DNA," *Information Hiding*, Springer, 2003, p. 373–386, 2003.
[85] M. Lu, X. Lai, and G. Xiao, "Symmetric-key cryptosystem with DNA technology," *Science in China Series F: Information*, vol. 50, 2007.
[86] O. Tornea, M. E. Borda, "DNA Cryptographic Algorithms," *Conference on Advancements of Medicine and TECHNOLOGY*, vol. 26, 2009, pp. 223-226, 2009.
[87] M. Hirabayashi and H. Kojima, "Design of True Random One-Time Pads in DNA XOR Cryptosystem," *Natural Computing*, 2010, pp. 174-183.
[88] L. A. I. Xuejia, L. U. Mingxin, Q. I. N. Lei, H. A. N. Junsong, and F. Xiwen, "Asymmetric encryption and signature method with DNA technology," *Science*, vol. 53, 2010, pp. 506-514, 2010.
[89] Cruz-Cortés and F. Rodríguez-Henríquez, "On the optimal computation of finite field exponentiation," *Advances in Artificial*, 2004, pp. 1-10, 2004.
[90] J.T. Jackson, E. Air, F. Base, G.H. Gunsch, R.L. Claypoole, G.B. Lamont, and H. Way, "Novel Steganography Detection Using an Artificial Immune System Approach," *Notes*, 2000.
[91] S. Ali, A. Hamdani, S. Shafiq, and F.A. Khan, "Cryptanalysis of Four-Rounded DES Using Binary," *System*, 2010, pp. 338-346.
[92] F. Glover and M. Laguna, *Tabu Search*. Kluwer Academic Publishers, Boston, 1997.
[93] C. Estébanez, J. Hernández-Castro, A. Ribagorda, and P. Isasi, "Finding state-of-the-art non-cryptographic hashes with genetic programming," *Parallel Problem Solving from Nature-PPSN IX*, p. 818–827, 2006.
[94] L.I. Yun-qiang and W. Ai-lan, "Near Collisions for the Compress Function of Hamsi-256 Found by Genetic Algorithm," *Evaluation*, pp. 4-7, 2010.
[95] D. Xiao and X. Liao, "A combined hash and encryption scheme by chaotic neural network," *Advances in Neural Networks-ISNN 2004*, p. 13–28, 2004.
[96] D. Xiao, X. Liao, and Y. Wang, "Neurocomputing Parallel keyed hash function construction based on chaotic neural network," *Neurocomputing*, vol. 72, pp. 2288-2296, 2009.
[97] S.L. Lian, J. Sun, and Z. Wang, "Secure hash function based on neural network," *Neurocomputing*, vol. 69, pp. 2346-2350, 2006.
[98] D. Xiao, X. Liao, and Y. Wang, "Parallel keyed hash function construction based on chaotic neural network," *Neurocomputing*, vol. 72, 2009, pp. 2288-2296, 2009.
[99] V.R. Kulkarni, S. Mujawar, and S. Apte, "Hash function implementation using artificial neural network," *Soft Computing*, vol. 1, 2010, pp. 1-8, 2010.
[100] Y. Li, S. Deng, and D. Xiao, "A novel Hash algorithm construction based on chaotic neural network," *Neural Computation*, pp. 133-141, 2011.
[101] I.B. Damgard, "A design principle for hash functions", Advances in Cryptology - CRYPTO 89, Lecture Notes in Computer Science, vol. 435, pp. 416-427, 1990.
[102] J. Daemen, R. Govaerts and J. Vandewalle, "A framework for the design of one- way hash functions including cryptanalysis of Damgard's one-way function based on cellular automaton", Advances in cryptology - ASIACRYPT '91, Lecture Notes in Computer Science, vol. 739, 1993.
[103] S. Hirose and S. Yoshida, "A one-way hash function based on a two-dimensional cellular automaton", The 20th Symposium on Information Theory and Its Applications (SITA97), Matsuyama, Japan, Proc. vol. 1, pp. 213-216, 1997.
[104] M. J. Mihaljevic, Y. Zheng, and H. Imai, A Cellular Automaton Based Fast One-Way Hash Function Suitable for Hardware Implementation. In *Proceedings of the First International Workshop on Practice and Theory in Public Key Cryptography: Public Key Cryptography* (PKC '98), Hideki Imai and Yuliang Zheng (Eds.). Springer-Verlag, London, UK, 217-233, 1998.
[105] A. del Rey, "Message Authentication Protocol Based on Cellular Automata," *Applications of Evolutionary Computing*, p. 52–60, 2007.
[106] Z. Ezziane, "Artificial Intelligence and DNA Computing," *Intelligent Computing Everywhere*, pp. 196-209, 2007.
[107] T. Toffoli, N. Margolus, "Cellular Automata Machines: A New Environment for Modelling," *The MIT Press*, 1987.
[108] A. E. Eiben, S. K.Smit, "Parameter Tuning for Configuring and Analyzing Evolutionary Algorithms", Swarm and Evolutionary Computation, Vol. 1, N° 1, pp.19-31, 2001.

Moisés Danziger é graduado em Ciência da Computação pela Universidade Paulista (UNIP), Campinas, São Paulo, Brasil, em 1999. Possui título de pós-graduação em comércio eletrônico pela Faculdade Frassineti do Recife (FAFIRE) e também em redes de computadores pela Universidade Católica de Pernambuco, ambas em Recife, Pernambuco, Brasil. Obteve o título de mestre em Engenharia da Computação pela Universidade Estadual de Pernambuco (UPE), Recife, Pernambuco, Brasil, em 2010. Atualmente, é aluno de doutorado na Faculdade de Engenharia Elétrica e de Computação (FEEC) da Universidade Estadual de Campinas (UNICAMP), Campinas, São Paulo, Brasil. Suas pesquisas se concentram nas áreas de inteligência e segurança computacional.

Marco Aurélio Amaral Henriques Possui graduação em Engenharia Elétrica pela Universidade Federal de Juiz de Fora (1986), mestrado em Engenharia Elétrica pela Chiba University - Japão (1990) e doutorado em Computer Science, também pela Chiba University - Japão (1993). Atuou como professor universitário no Japão (Shinshu University) de 1993 a 1996 e é professor associado da Faculdade de Engenharia Elétrica e de Computação (FEEC) da Universidade Estadual de Campinas (UNICAMP), onde exerce também as funções de Superintendente do Centro de Computação e Coordenador Geral de Tecnologia de Informação e Comunicação. Tem experiência na área de Ciência da Computação, com ênfase em processamento de alto desempenho e segurança da informação, atuando principalmente nas áreas de processamento maciçamente paralelo e criptografia aplicada.