Aplicação do Algoritmo de Colônia de Formigas para Redução do Tamanho de Chaves Públicas em Criptografia Completamente Homomórfica

Joffre Gavinho Filho[1], Jonice Oliveira[1], Claudio Miceli[1], Gabriel Pereira da Silva[2]

[1]Programa de Pós-Graduação em Informática

{joffreufrj, jonice, cmicelifarias}@gmail.com

[2]Departamento de Ciência da Computação

gabriel.silva@ufrj.br

Universidade Federal do Rio de Janeiro

Abstract. *The fully homomorphic encryption, one of the areas post-quantum cryptography, is one of techniques for processing and manipulating encrypted data without deciphering them. Used on platforms where traditional encryption may not have the desired security, such as cloud computing case. Several methods have been proposed to facilitate the use of this technique. However, for its effective use, there is the need to generate parameters that result in keys with extremely large sizes. This article aims to reduce the size of public keys using the optimization algorithm ant colony to ratify the calibration of the variants of algorithms already present in the literature.*

Resumo. A *criptografia completamente homomórfica, uma das áreas da criptografia pós-quântica, é uma das técnicas para o processamento e manipulação de dados criptografados sem decifrá-los. Utilizada em plataformas onde a criptografia tradicional pode não ter a segurança desejada, como o caso da computação em nuvem. Diversos métodos têm sido propostos para viabilizar o uso dessa técnica. Porém, para a sua efetiva utilização, há a necessidade da geração de parâmetros que resultam em chaves com tamanhos extremamente grandes. Este artigo tem como objetivo reduzir o tamanho das chaves públicas usando o algoritmo de otimização por colônia de formigas para ratificação da calibração das variantes de algoritmos já presentes na literatura.*

1. Introdução

A Computação em Nuvem (*Cloud Computing*) – definida como uma plataforma para a utilização de uma estrutura com capacidades de armazenamento e processamento compartilhados e interligados por meio da Internet [Sousa 2009] – tem se tornado uma das soluções utilizadas na redução de custos para atender às crescentes demandas em informática nos tempos modernos.

Dentre todas as metas de projetos necessárias à viabilização da Computação em Nuvem, destacamos a provisão de segurança que, basicamente, tem a Criptografia como um dos métodos mais utilizados na determinação de confidencialidade e sigilo das informações manipuladas pela nuvem [NIST 1978]. Porém, os algoritmos de criptografia convencionais possuem uma característica em comum que inserem um

ponto de vulnerabilidade extremamente sensível no contexto de segurança na nuvem, a denominada "não-maleabilidade" [Coron 2011]. Este conceito requer que todos os dados devam ser descriptografados antes de serem processados, principalmente em ambiente compartilhado, isto é, em cada um dos servidores, componente da nuvem, onde haja necessidade de manipulação ou processamento dos dados, há a necessidade de se transformá-lo em texto claro para que se possa validar ou mesmo comparar os dados que estão codificados. Tendo-se então a necessidade de acesso direto aos dados originais. Em suma, o principal problema com ambiente compartilhado e a criptografia, reside no processamento dos dados, porque na criptografia tradicional, os dados não podem ser alterados enquanto criptografados.

Na Criptografia Homomórfica – que faz parte da chamada criptografia pós-quântica: ramo da criptografia que estuda as classes de algoritmos criptográficos resistentes à criptoanálise baseada na computação quântica [Daniel 2009] – os dados encriptados são manipulados em sua forma bruta, isto é, sem a necessidade de decifrá-los. Os modelos de encriptação homomórfica (EH) são baseados em somas e multiplicações de funções, isto é, além dos convencionais algoritmos de encriptação e desencriptação [Stalling 2007], há também um algoritmo de avaliação, que toma como entrada uma mensagem encriptada E(m) e retorna uma criptografia de f(E(m)) [Gentry 2009]. Os esquemas EH podem ser basicamente classificados em dois tipos: o primeiro são os esquemas, chamados de parcialmente homomórficos, que, além de operações de criptografia e decriptografia também executam operações de soma ou multiplicação, que tomam como entrada mensagens criptografadas m_1 e m_2 e regressam a criptografia de $m_1 + m_2$ ou $m_1 * m_2$, respectivamente. Se um esquema EH suporta tanto adição quanto multiplicação, ele também pode avaliar qualquer circuito aritmético em dados criptografados [Gentry 2009] e, portanto, podemos dizer que é um esquema de Criptografia Completamente Homomórfica (CCH), ou seja, se o $E(m)$ é a codificação de uma mensagem m, um modelo de criptografia é totalmente homomórfico se: $f(m) \rightarrow E(m_1 + m_2) = E(m_1) + E(m_2)$; e, $f(m) \rightarrow E(m_1 * m_2) = E(m_1) * E(m_2)$ [Coron 2011]. Usando esse tipo de regime, qualquer circuito pode receber uma avaliação homomórfica, permitindo a construção de programas que podem ser executados com as criptografias de suas entradas para produzir uma saída criptografada. Como esses programas não decodificam as informações, eles podem ser utilizados por terceiros não confiáveis sem revelar a sua entrada e estado interno. Por exemplo, podem-se somar dois números criptografados e, a menos que se possa decriptografar o resultado, não há como se descobrir o valor dos números originais individualmente [Gavinho 2015].

O grande problema dos métodos propostos para os sistemas completamente homomórficos é que os seus tempos de execução, bem como o tamanho dos parâmetros utilizados, especialmente os das chaves públicas, crescem ciclicamente a cada iteração em complexidade na ordem de $O(\lambda^{10})$ [Coron 2011]. Onde λ é o parâmetro de segurança de todo o regime, e que define o tamanho, em bits de comprimento, das chaves geradas. Melhorias dos processos estão sendo propostas. Como é o caso [Coron 2011] que propôs um esquema de chaves públicas DGHV [Dijk 2010] totalmente homomórfico, e que reduz o tamanho dessas chaves geradas para aproximadamente $O(\lambda^{7})$ [Coron 2011], e em [Coron 2012] onde tal redução chega a $O(\lambda^{5})$. Apesar de novas técnicas terem sido propostas [Mukherjee 2016], foi verificado que [Mukherjee 2016], os tamanhos de redução das chaves públicas têm sido pouco expressivos quando comparados a [Coron 2012], justificativa esta na qual tomamos como base para a

utilização especifica das primitivas de criptografia homomórfica de [Coron 2012] como fundamentos deste trabalho.

O trabalho que aqui apresentamos visa aplicar a meta-heurística de Otimização por Colônia de Formigas (*Ant Colony Optimization* - ACO), como forma de demonstrar as técnicas de redução de chaves públicas utilizadas por [Coron 2012], com os objetivos principais de: especificar, validar e ratificar a calibração dos parâmetros do regime das variantes de [Coron 2012] realizada por Gavinho em: [Gavinho 2015] e [Gavinho 2016], no qual o autor demonstra por meio da utilização de algoritmos genéticos (AG), a redução em cerca de 600 KB no tamanho da Chave Pública de 18 MB da compressão feita por [Coron 2012], mantendo-se, mesmo assim, os mesmos níveis de segurança alcançados por Coron [Coron 2012].

Este artigo está organizado da seguinte forma: na Seção 2 são descritos os mecanismos de Criptografia Completamente Homomórfica e os métodos de compactação de chaves públicas, bem como os trabalhos relacionados; na Seção 3, a proposta de avaliação e otimização é apresentada, bem como os experimentos realizados e as análises dos resultados obtidos; e finalmente, na Seção 4 são tecidas as conclusões finais e apresentadas as propostas de trabalhos futuros.

2. Trabalhos Relacionados

Nesta seção, descrevemos um breve histórico da criação e desenvolvimento das técnicas homomórficas, bem como os métodos e trabalhos propostos para a redução de chaves no processo de Criptografia Completamente Homomórfica.

2.1. Criptografia Completamente Homomórfica (CCH)

Os criptógrafos reconheceram há muitos anos a necessidade de um algoritmo de criptografia que permitisse a computação arbitrária em dados criptografados. A primeira proposta teórica prática foi feita em 1978 por Rivest, Adleman e Dertouzos [Rivest 1978], quando os autores sugeriram a construção de homomorfismos secretos - *privacy homomorphisms* [Rivest 1978] - como uma técnica para proporcionar um forma de proteção para a computação em dados sensíveis. O esquema proposto era parcialmente homomófico, i.e., usava apenas a propriedade multiplicativa do método e não a aditiva, além de não oferecer proteção contra ataques de texto simples escolhido (*Chosen Plaintext Attack - CPA*) [Morris 1993], não possuindo por isso segurança semântica. Caracterizamos, em criptografia, um esquema como semanticamente seguro se: dado um texto cifrado de qualquer mensagem, e o tamanho dessa mensagem, não há nenhum algoritmo probabilístico em tempo polinomial (*Probabilistic Polynomial-Time Algorithm - PPTA*), [Michael 1980], que possa determinar qualquer informação da mensagem com significado maior do que a probabilidade de uma escolha aleatória. Isto é: o conhecimento da mensagem cifrada e o tamanho da mensagem original, não revela nenhuma informação sobre esta mensagem, bem como, nenhum dado pode ser extraído com facilidade a partir do texto cifrado.

Após a bem sucedida construção de homomorfismos secretos - *privacy homomorphisms* [Rivest 1978], a comunidade científica começou a procurar implementações diversas para esta teoria – algoritmos capazes de executar a chamada criptografia completament homomórfica. Muitas tentativas sem sucesso foram realizadas, tanto que o problema permaneceu sem solução até pouco anos atrás – 2009 –

quando, Craig Gentry [Gentry 2009], demonstrou uma solução sugerindo a utilização de reticulados ideais na construção de um sistema de encriptação completamente homomórfico. Apesar do sucesso na solução, a proposta de Craig Gentry não foi suficientemente eficaz para ser usado na prática, devido à complexidade das avaliações de multiplicações e o tamanho excessivamente grande das chaves públicas geradas no processo.

O primeiro grande desafio em criptografia completamente homomórfica é torná-la eficientemente prática. Enquanto a construção original de [Gentry 2009] é vista como impraticável, as últimas construções e os esforços de implementação melhoraram sensivelmente a eficiência da criptografia completamente homomórfica. As propostas iníciais de implementações com foco na proposta original de Gentry e suas variações [Gentry 2010] melhoraram os gargalos de eficiência. Implementações posteriores fizeram uso de avanços recentes nos algoritmos [Dijk 2010] e das técnicas algébricas [Brakerski 2012] procurando melhorar a eficiência concreta dos resultados em tais esquemas Porém, o grande desafio está baseado no excessivo tamanho das chaves públicas criadas por esses métodos, bem como a necessidade de uma estrutura de armazenamento robusta. Em Dijk, Gentry, Halevi e Vaikuntanathan (DGHV) [Dijk 2010], por exemplo, o tamanho da chave pública se encontra na ordem de $O(\lambda^{10})$ bits e o tamanho da chave privada cerca de $O(\lambda^2)$ – sendo λ o parâmetro segurança de todo os esquemas. Coron, Mandal e Naccache [Coron 2011] propuseram uma modificação na geração de chaves, a fim de reduzir o tamanho da chave pública para $O(\lambda^7)$. A modificação consiste na utilização de formas quadráticas dos elementos da chave pública, em vez de formas lineares, como é feito no esquema DGHV. A ideia de Coron [Coron 2011] é armazenar apenas um pequeno subconjunto da chave pública e, quando necessário, gerar a chave pública completa multiplicativamente combinada aos elementos do subconjunto das chaves secretas. Esta proposta mantém a segurança semântica, e se baseia no problema do máximo divisor comum aproximado parcial (*Partial Approximate Greatest Common Divisor*) [Coron 2011], que consiste em retirar o erro dos primeiros conjuntos de chaves públicas criadas durante o processo. Uma segunda variante do método de otimização proposta em [Coron 2012], reduz ainda mais o tamanho das chaves prara $O(\lambda^5)$. Este esquema tem o mesmo problema básico de segurança do máximo divisor comum aproximado (*Approximate Greatest Common Divisor*) [DGHV 2010], isto é, o problema da Segurança Circular [Boneh 2008], que, dado um esquema de criptografia *E*, dizemos que *E* tem segurança circular se for seguro criptografar a chave privada com sua própria chave pública, esquema este que é a base do regime de segurança do DGHV, descrito a seguir.

2.1.1. O Esquema DGHV

Usando apenas álgebra modular ao longo de um anel de inteiros, Dijk, Gentry, Halevi e Vaikuntanathan propuseram um esquema totalmente homomórfico denominado DGHV [Dijk 2010]. Esquema que se provou ter uma menor complexidade matemática quando comparada com os regimes baseados em reticulados ideais de Gentry [Gentry 2009]. Coron, com base neste mesmo padrão, propôs suas duas variantes do método DGHV [Coron 2011] e [Coron 2012], conseguindo a otimização do custo computacional de certas primitivas, e a diminuição do tamanho do esquema de chaves públicas originais utilizando várias técnicas de redução e de compressão de chaves.

Basicamente, o esquema DGHV [Dijk 2010], é composto de cinco algoritmos – também chamados de primitivas: *E*(*KeyGen*, *Encrypt*, *Decrypt, Recrypt, Evaluate*), A primitiva *Keygen* é o esquema responsável por gerar o par de chaves (públicas e privadas); *Encrypt* é responsável por gerar o texto cifrado; *Decrypt* é responsável por decifrar o texto encriptado; *Recrypt* é utilizada para a criptografia da criptografia, isto é, a criptografia em níveis; e, *Evaluate* que realiza o processamento em tuplas de bits codificados em um circuito lógico, e que retorna o equivalente criptografado a este circuito como se fosse aplicado aos dados originais, isto é, faz a manipulação homomórfica nos bits codificados, sem a necessidade de decriptografá-los.

2.1.2. O Método de Coron.

O primeiro trabalho de Coron, denominado de sua primeira variante [Coron 2011], incluiu a adição de novos parâmetros quadráticos ao esquema de primitivas DGHV, armazenando assim apenas um pequeno conjunto de valores relacionados com a chave pública e, em seguida, gerando uma chave pública completa em tempo de execução. Usando esta técnica, Coron demonstrou a redução do tamanho da chave pública da ordem de $O(\lambda^{10})$ para a de $O(\lambda^{7})$, [Coron 2011], quando comparados ao DGHV.

Em sua segunda variante Coron [Coron 2012], demonstrou a redução do comprimento das chaves públicas produzidas para a ordem de $O(\lambda^{5})$. A principal inovação proposta por Coron neste esquema é que, em vez de armazenar todos os elementos-chave da criptografia, ele só armazena o valor de correção em relação a um gerador de números aleatórios. Assim, os dados a serem armazenados são menores, e, havendo necessidade dos dados, estes são recuperados "*on-the-fly*" pelas primitivas *Encrypt*, *Recrypt*, *Decrypt* e *Expand* (nova primitiva para expandir os blocos de dados reduzidos). Além disso, é descrita uma técnica de troca de módulos, que permite este regime, sem usar a estrutura proposta de *boostraping* descrito por Brakerski, Gentry e Vaikuntanathan, (BGV) [Brakerski 2012].

O regime inicial de inteiros usados por Coron como base para o seu trabalho, bem como para a criação de sua segunda variante, é fundamentada no trabalho de Gentry [Gentry 2009]. Este definiu o DGHV com base em um conjunto de inteiros, $x_i = p.q_i + r_i$, p $0 \leq i \leq \tau$, sendo λ o parâmetro de segurança em que se baseia todo o esquema. Os seguintes parâmetros devem ser utilizados para compor o esquema de Criptografia Homomórfica Reduzida (CHR) [Brakerski 2012], que, em seguida, deve ser reforçada para gerar o CCH de inteiros [Gavinho 2016]:

- γ é o comprimento em bits de x_i.
- η é o comprimento em bits da chave secreta p.
- ρ é o comprimento em bits do ruído r_i.
- τ é o número da chave pública de x_i.
- ρ' é um parâmetro de ruído secundário usado para criptografar, [Coron 2011].

O esquema é definido com as seguintes restrições:

- $\rho = \omega\ (log\lambda)$, necessário para a proteção contra ataques de força bruta.
- $\eta \geq \rho.\Theta\ (\lambda log 2\lambda)$, necessário para a execução de operações homomórficos de avaliação.
- $\gamma = \omega\ (\eta 2.log\lambda)$, necessário para a proteção a ataques com base no problema do MDC.
- $\tau \geq \gamma + \omega\ (log\lambda)$, necessário para reduzir a abordagem do MDC Aproximado.
- $\rho' = \rho + \omega\ (log\lambda)$, necessário para o parâmetro de ruído secundário.

2.1.3. A Otimização do Tamanho das Chaves Públicas de Gavinho.

O processo de otimização proposto por [Gavinho 2015] e [Gavinho 2016] na segunda variante de Coron, [Coron 2012], para a compactação e redução de chaves, consiste basicamente na calibragem dos valores dos parâmetros usados nas primitivas criptográficas através do Algoritmo Genético (AG), que, segundo [Lacerda 1999], são métodos de otimização e busca, inspirados nos mecanismos de evolução de populações de organismos.

Quanto à simulação, o esquema proposto por [Gavinho 2015] [Gavinho 2016], tanto para a criptografia parcialmente, como para a completamente homomórfica, foi implementado no software matemático Matlab/Simulink©, bem como todas as primitivas criptográficas descritas em [Coron 2012]. Coron implementou toda a sua proposta para o esquema DGHV utilizando o software matemático SAGE (Sistema para Experimentação de Álgebra e Geometria) [William 2012]. Como base comparativa, todas as métricas e primitivas de [Coron 2012], foram reimplementadas, e simuladas por [Gavinho 2016] no Matlab/Simulink©, com a efetiva calibração do simulador para que todas as métricas e tempos de execução repetissem fielmente os métodos em estudo. Os testes foram realizados, e os resultados analisados e comparados com os resultados obtidos pelos autores em seus trabalhos, tendo como resposta a igualdade de soluções que corroboraram com os resultados apresentados pelos artigos originais.

Como métricas para os valores dos parâmetros de segurança, foram utilizados os definidos por [Coron 2011]: *Toy* possuindo um lambda (λ) equivalente a 42 bits de segurança, *Small* equivalente a 52 bits, *Medium* equivalente a 62 bits e *Large* equivalente a 72 bits. Sendo λ o parâmetro que define o nível de segurança atribuído diretamente ao par de chaves gerados pela primitiva de geração de chaves *KeyGen,* além de definir todo o nível de segurança do método.

Como método comparativo, [Gavinho 2015] se baseou no proposto pela literatura especializada que, normalmente, utiliza a medida do tempo de execução de cada uma das primitivas criptográficas (os algoritmos) a fim de quantificar e avaliar o desempenho de cada uma delas [Gavinho 2016]. As primitivas são executadas de maneira repetitiva – todos os testes seguem uma distribuição normal e possuem intervalo de confiança de 95% – e tem o seu tempo de execução contabilizado por software. Em posse do tempo contabilizado e do número de vezes de execução das primitivas aplica-se uma média aritmética simples sobre os mesmos, obtendo assim o tempo médio de execução das respectivas primitivas que pode ser utilizado de maneira comparativa entre variadas implementações e variados esquemas homomórficos [Gavinho 2016].

Para otimização do método de [Coron 2012], [Gavinho 2015] utilizou uma Função Utilidade Aditiva (FUA) [Keeney 1976] com a finalidade de explorar e dar "pesos" ao Espaço de Soluções $[\lambda - 2, \lambda + 3]$, isto é, $39< \lambda<46$, $49< \lambda<56$, $59< \lambda<66$ e $69< \lambda<76$, sendo a FUA definida por: $S_{H\text{-}i} = T\lambda_{\text{-}1}P_1 + T\lambda_{\text{-}2}P_2 + T\lambda_{\text{-}3}P_3 + T\lambda_{\text{-}4}P_4 + T\lambda_{5}P_5 + T\lambda_{\text{-}6}P_6$, onde $S_{H\text{-}i}$ é a função Utilidade (Seleção); $T\lambda_{\text{-}i}$ é o peso do parâmetro λ_i sendo $\sum \lambda_{i\ =}\ 1$, e; P_i equivale ao elemento do espaço de soluções $[\lambda - 2, \lambda + 3]$ a ser determinado, por meio do seu peso, qual a sua influência real no processo de criptografia.

Os métodos de avaliações, calibrações e análises, propostas em [Gavinho 2015] [Gavinho 2016], iniciam-se com o processo de geração de valores ponderados para os pesos dos parâmetros λ, por meio do Algoritmo Genético (AG), e iniciados com a produção de uma massa de dados de 500 MB sem formatação. Essa massa de dados é gerada seguindo as métricas das primitivas de [Coron 2011] e reproduzem o estado original de testes de suas variantes [Coron 2011]. Esta massa é inicialmente usada para duas finalidades: *i*) a calibração dos módulos de forma análoga às primitivas de [Coron 2011], principalmente do algoritmo de avaliação do tempo de execução das primitivas; e, *ii*) ser usada como dados de treinamento para o agoritmo genético (AG). Foram executadas diversas simulações para cada um dos tamanhos originais dos parâmetros de segurança λ: *Toy* (42 bits), *Small* (52 bits), *Medium* (62 bits) e *Large* (72 bits), seguindo a proposta de [Coron 2011]. Após esta fase inicial dos módulos de calibração e formação dos algoritmos genéticos, a fase evolucionária do processo é iniciada. Sendo processada em um total de 100 gerações, onde as fases de seleção (método da roleta), cruzamento (em dois pontos) e mutação [Lacerda 1999] são executadas repetidamente, com a finalidade da convergência do algoritmo para um valor central do tamanho do parâmetro de segurança λ.

Tabela 1. Valores dos parâmetros obtidos com a variação dos tamanhos do parâmetro λ com os repectivos tempos de execução das primitivas criptográficas e os tamanhos finais das Chaves Públicas.

λ	ρ	η	μ	γ x 10^6	Θ	*KeyGen*	*Encryp*	*Decrypt*	*Expand*	*Recrypt*	*Evaluate*	*pk size*
40	12	318	54	0,03	131	00:00:05	00:00:04	00:00:00	00:00:01	00:00:29	00:00:17	346
41	**13**	**325**	**55**	**0,04**	**157**	**00:00:05**	**00:00:05**	**00:00:00**	**00:00:01**	**00:00:39**	**00:00:19**	**350**
42	14	336	56	0,06	195	00:00:06	00:00:05	00:00:00	00:00:01	00:00:41	00:00:20	354
43	15	341	56	0,10	229	00:00:06	00:00:05	00:00:00	00:00:01	00:00:41	00:00:20	358
44	16	348	57	0,14	263	00:00:17	00:00:06	00:00:00	00:00:03	00:01:00	00:00:35	362
45	17	355	58	0,22	330	00:00:21	00:00:16	00:00:00	00:00:05	00:01:30	00:00:43	688
50	18	368	63	0,54	600	00:00:49	00:00:53	00:00:00	00:00:14	00:04:15	00:01:50	1.356
51	**19**	**379**	**64**	**0,62**	**668**	**00:00:59**	**00:00:59**	**00:00:00**	**00:00:14**	**00:04:40**	**00:02:19**	**1.523**
52	20	390	65	0,70	735	00:01:00	00:01:00	00:00:00	00:00:15	00:04:50	00:02:30	1.690
53	21	395	65	64,40	872	00:03:00	00:03:50	00:00:00	00:00:19	00:05:51	00:02:40	680.451
54	22	400	66	128,10	1.009	00:07:00	00:07:10	00:00:00	00:00:22	00:06:55	00:03:15	1.698.593
55	23	404	67	255,50	1.283	00:08:00	00:08:00	00:00:00	00:00:30	00:13:10	00:04:30	2.716.734
60	24	416	71	765,20	2.378	00:27:00	00:20:00	00:00:01	00:02:30	00:48:00	00:11:50	6.419.067
61	**25**	**423**	72	**892,60**	**2.651**	**00:27:50**	**00:20:45**	**00:00:01**	**00:03:00**	**00:50:00**	**00:13:20**	**7.406.356**
62	26	438	73	1.020,00	2.925	00:28.00	00:21:00	00:00:01	00:03:10	00:51:00	00:15:10	7.900.000
63	27	443	73	1.092,00	3.099	00:31:00	00:22:12	00:00:01	00:05:20	00:01:00	00:30:15	8.151.398
64	28	449	74	1.166,00	3.272	00:58:00	00:24:10	00:00:02	00:07:30	00:02:10	00:01:00	8.528.496
65	29	456	75	1.315,00	3.619	00:02:00	00:28:00	00:00:03	00:11:10	00:03:23	00:03:12	8.905.594
70	30	464	79	1.906,00	5.006	00:09:10	00:03:35	00:00:04	00:50:00	00:10:10	00:10:30	16.342.969
71	**32**	**473**	**80**	**2.054,00**	**5.353**	**00:09:55**	**00:05:12**	**00:00:05**	**00:50:00**	**00:11:00**	**00:11:45**	**17.368.750**
72	34	492	82	2.200,00	5.700	00:10:00	00:07:15	00:00:05	00:51:00	00:11:34	00:12:00	18.000.000
73	36	503	84	2.260,00	6.104	00:12:00	00:11:10	00:00:05	00:51:00	00:12:20	00:13:30	18.631.250
74	39	522	87	2.510,00	6.412	00:17:00	00:15:00	00:00:06	00:55:00	00:13:00	00:14:05	19.657.031
75	41	541	89	3.000,00	6.921	00:21:00	00:21:00	00:00:07	00:58:00	00:15:05	00:15:00	21.708.594

Em cada uma das gerações do algoritmo, 1000 rodadas foram realizadas para cada valor de λ. Variando-se na ordem de números inteiros [λ - 2, λ + 3]. Totalizando para cada parâmetro: *Toy*, *Small*, *Medium* e *Large*, 6.000 rodadas. Um total de 24.000 rodadas para cada geração. Por fim todo o processo evolutivo do AG necessitou de $24x10^6$ rodadas. Podemos observar na Tabela 1, quando da execução dos parâmetros de segurança λ, os tamanhos obtidos para cada primitiva, bem como os seus tempos de execução. O resultado final da convergência do AG para o $T\lambda_{-i}$ foi: $S_{H\text{-}i} = \mathbf{10}P_1 + \mathbf{25}P_2 + \mathbf{23}P_3 + \mathbf{17}P_4 + \mathbf{14}P_5 + \mathbf{11}P_6$, sendo o maior peso dado a $\mathbf{P_2}$ (λ=41, λ=51, λ=61 e λ=71).

Durante todas as rodadas de simulações, testes idênticos aos de [Coron 2011] são realizados com valores calculados para parâmetros λ na verificação da segurança,

incluindo a segurança semântica do método. Foi verificado que os valores dos parâmetros λ convergiram para os observados na Tabela 1. Valores que, além de serem uma unidade de magnitude menores do que os parâmetros definidos por [Coron 2011], possuem uma redução substancial nos tempos de execução em cada umas das primitivas criptográficas do mecanismo. Mantendo mesmo assim a segurança semântica em todo o processo. Apesar de terem sido calculados outros valores menores, bem como menores tempos de execução das primitivas criptográficas, observado na Tabela 1, tais valores, como por exemplo: λ = 40, 50, 60 ou 70, quando utilizados no método não provêem segurança semântica, não sendo adequado o seu uso. Na Tabela 2 , observamos os valores comparativos das reduções dos tempos de execução e das diferenças dos tamanhos finais das chaves públicas.

Tabela 2. Valores comparativos das reduções dos tempos de execução e das diferenças dos tamanhos finais das chaves públicas.

λ	ρ	η	β	γ	Θ	*KeyGen*	*Encrypt*	*Decrypt*	*Expand*	*Recypt*	*Evaluate*	*#Pk*
42-41	1,00	11,00	1,00	0,02	38,00	00:00:00:133	00:00:00:015	00:00:00:000	00:00:00:007	00:00:00:984	00:00:00:004	4
52-51	1,00	7,00	1,00	0,08	67,50	00:00:00:600	00:00:00:070	00:00:00:000	00:00:00:054	00:00:18:987	00:00:00:069	167
62-61	1,00	6,00	1,00	157,00	273,75	00:00:05:900	00:00:00:020	00:00:00:000	00:00:00:944	00:06:34:000	00:00:01:064	1.288.289
72-71	1,00	7,00	1,00	22,00	346,88	00:00:39:000	00:00:00:319	00:00:00:000	00:00:27:945	00:12:26:000	00:00:03:988	631.250

2.1.4. A Otimização por Colônia de Formigas.

A Otimização por Colônia de Formigas (ACO, do inglês *Ant Colony Optimization*) é uma meta-heurística baseada em população e inspirada no comportamento de busca de alimento (comportamento forrageiro) das formigas. O Algoritmo formulado na década de 1990 por Marco Dorigo [Dorigo 1994], e que têm evoluído desde sua publicação [Dorigo e Stutzle 2004], relacionada às habilidades das formigas em encontrar o caminho mais curto entre o ninho e o alimento. Esta busca é efetuada através da exploração das trilhas de feromônio, substância química depositada pelas formigas durante seu percurso, e no comportamento denominado estimergia (em inglês *stigmergy*), que é o tipo de comunicação indireta entre as formigas, na qual elas rastreiam os melhores caminhos. Segundo [Castro 2006], estimergia é o método de comunicação em que os indivíduos de um sistema se comunicam entre si pela modificação do ambiente local. Devido a este comportamento cooperativo e eficaz de busca elas vão construindo melhores alternativas no caminho para encontrar o alimento.

Podemos observar na Figura 1 – baseada nos experimentos das pontes binárias realizados por Deneubourg [Dorigo 2004] – a varredura do terreno, bem como a formação da trilha de feromônio. Na letra (a), uma situação de demonstração do cenário inicial, composto do ninho (colônia) das formigas, da fonte de alimentos e de três caminhos alternativos até a fonte de alimento: [01], [02] e [03], sendo os caminhos [01] e [03] de mesmo comprimento e maiores que o caminho [02]. Em (b), as formigas podem ser observadas em sua busca aleatória por alimentos no terreno, movendo-se randomicamente, ou seja, realizam buscas exploratórias por possíveis soluções. Na letra (c), houve a detecção da fonte de alimento, e inicia-se então o transporte do mesmo para o ninho, como o caminho [02] é o menor entre os três, as formigas que escolheram esse percurso, chegam primeiro à colônia, tendo como consequência um maior acúmulo de feromônio em [02]; finalmente em (d), observamos a convergência da maior parte das formigas para o menor caminho, [02]; pode-se observar também, que, apesar da descoberta do menor percurso ninho-alimento, ainda há formigas procurando, aleatoriamente, caminho alternativos.

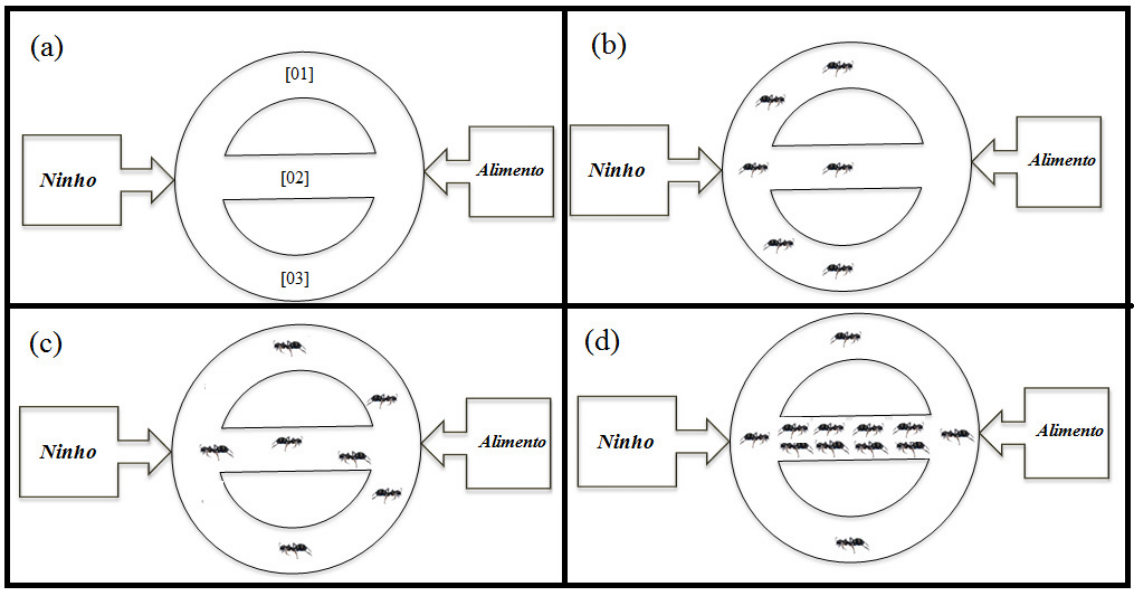


Figura 1. Busca por alimento: (a) condição inicial da colônia; (b) inicio da varredura no terreno; (c) descoberta do alimento; e, (d) formação da trilha de feromônio.

Este comportamento foi então simulado em algoritmos de otimização, conhecidos como ACO (do inglês *Ant Colony Optimization*-ACO). Estes algoritmos buscam melhores soluções nas trilhas em que se encontra a maior quantidade de feromônio, com o controle de seu depósito e evaporação. Realizam-se também atualizações locais e globais de feromônio, melhorando assim a busca de resultados e alternativas por caminhos não trilhados. As formigas artificiais constroem soluções de forma probabilística utilizando duas informações: (*i*) a trilha de feromônio (artificial) que muda dinamicamente durante a execução do programa de modo a refletir a experiência já adquirida durante a busca, e; (*ii*) a informação heurística específica do problema a ser resolvido. [Dorigo 2004] apresenta a correspondência entre o que acontece na natureza e o algoritmo ACO, correspondência representada na Tabela 2.

Tabela 2 - Correspondência entre a natureza e o ACO.

Natureza	ACO
Possíveis caminhos entre o ninho e o alimento	Conjunto de possíveis soluções
Caminho mais curto	Solução ótima
Ação via comunicação por feromônio	Procedimento de otimização

A construção da solução pela formiga artificial *k* começa de uma posição *i* , na Figura 2 → n_i, movendo-se e escolhendo probabilisticamente a posição vizinha *j* (entre os vizinhos factíveis), isto é uma das arestas A_i.

A probabilidade da formiga *k* que está no ninho de escolher a aresta $ij(A_i)$ para acessar o alimento f_i é dada pela regra [Dorigo 2004]:

$$p_{ij}^{k} = \frac{(\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}}{\sum_{i \in N_i^k}(\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}} \text{ , se } j \in N_i^k \text{ ; senão } p_{ij}^{k} = 0.$$

onde: τ_{ij}^{k} → é a quantidade de feromônio associada à aresta *ij*; α e β são parâmetros para determinar a influência do feromônio e da informação heurística, respectivamente; N_i^k

são os caminhos possíveis (arestas *ij*) associado à *k*-ésima formiga; η_{ij} é a informação heurística ou valor heurístico que é definida em função das características do problema.

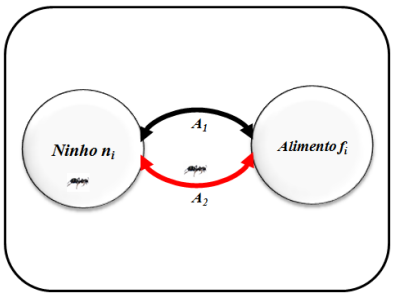


Figura 2. Diagrama formado pelo ninho n_i, fonte de alimento f_i e possíveis caminhos (arestas A_i) para a formiga *k* acessar a fonte de alimento f_i.

A atualização de feromônio, dada por: $\tau_{ij}^{k} \leftarrow \tau_{ij}^{k} + \Delta\tau^{k}$, ocorre no retorno do circuito da *k*-ésima formiga, depositando uma quantidade $\Delta\tau^{k}$ de feromônio nas arestas visitadas. O valor de $\Delta\tau^{k}$ é assumido constante para todas as formigas. Esta variação de feromônio faz com que os caminhos mais curtos sejam alcançados pela intensificação da quantidade de feromônio nos melhores trechos.

A evaporação do nível de feromônio nos caminhos tende a permitir uma convergência das formigas para um valor subótimo, ou seja, uma solução nas proximidades do ótimo. Como a quantidade de feromônio diminui, isto leva à exploração de caminhos alternativos durante o processo de busca. A evaporação limita o nível máximo de feromônio em cada trilha, evitando com isso, uma estagnação da solução em um valor ótimo local. A evaporação respeita a equação: $\tau_{ij}^{k} \leftarrow (1-\rho)\,\tau_{ij}^{k}$, $\forall(i,j) \in A_i; \rho \in [0,1]$, sendo ρ um parâmetro da taxa de evaporação de feromônio do ACO.

O algoritmo ACO resume-se em três procedimentos, [Dorigo 2004]: (*i*) construção das soluções com as formigas; (*ii*) atualização de feromônio, e; (*iii*) ações *daemon*. O primeiro procedimento é a construção das soluções pelas formigas com base em um grafo (estrutura formada por vértices e arestas) [Dorigo 2004]. As formigas movem-se entre os componentes do grafo, variáveis do problema, estabelecendo uma conexão entre eles. Este movimento é determinado estocasticamente e localmente pelas informações das trilhas de feromônio e pela informação heurística, η, a qual é utilizada para melhorar a eficiência do algoritmo sendo definida em função das características do problema. O segundo procedimento está relacionado ao depósito ou à evaporação do feromônio durante a construção na busca da solução. Quanto maior a quantidade de feromônio maior a probabilidade de uma mesma conexão ou componente ser usado, reforçando sua trilha. Por outro lado, a diminuição faz com que se busquem novas regiões ainda não consideradas, podendo ser regiões próximas do ótimo. O terceiro procedimento, ações *daemon*, diz respeito a rotinas que venham a melhorar a busca em determinado local, ações de busca local, ou um conjunto de ações globais que possibilitem tomar decisões positivas. O termo *daemon*, que é originário da linguagem computacional, significa uma rotina ou um programa criado para realizar determinada tarefa padrão, com fins específicos, a ser executado.

3. Proposta da Avaliação da Redução de Chaves.

Nossa proposta tem como diretriz a aplicação da ACO (*Ant Colony Optimization*), no método definido por [Coron 2012] como forma de verificação, ratificação e validação das otimizações dos métodos propostos por Gavinho em: [Gavinho 2015] e [Gavinho 2016]. Nesses trabalhos o autor demonstra, por meio da utilização de Algoritmos Genéticos (AG), a redução em cerca de 600 KB no tamanho da Chave Pública de 18 MB proposta e definida por [Coron 2012], mantendo-se, mesmo assim, os mesmos níveis de segurança dos alcançados por [Coron 2012] em seu método. Detalhamos nosso trabalho nesta seção.

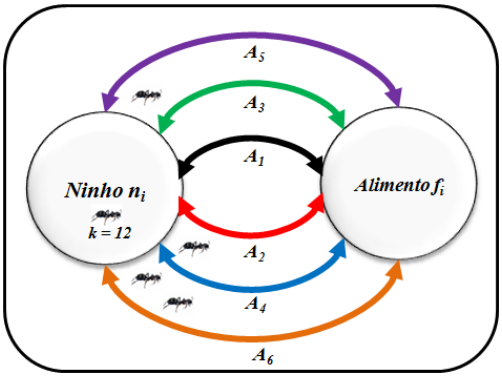


Figura 3. Diagrama formado pelo ninho n_i, fonte de alimento f_i e seis caminhos (A_i)

Em nossa proposta, os esquemas homomórficos de [Gentry 2009], [Dijk 2010], [Coron 2011], [Coron 2012], a otimização por AG de [Gavinho 2015] e [Gavinho 2016], bem como o método artificial de ACO (*Ant Colony Optimization*), foram implementados na ferramenta matemática Matlab/Simulink©. Bem como os três procedimentos de ACO:

```
1  Inicio
2    var i=0, C_i=0, t=0, nr_estagios=0, nr_ciclos=0,
3        τ=0; η_l=0; ρ =0, ω=0, λ=0, γ=0, α=0, τ_l=0, η=0, Θ=0, ρ' = 0,
4        nr_estagios=100, total_Ci=6,
5        Para i=1 ate nr_estagios Faça
6            Processar gerar_ACO(i);
7            Para i=1 ate total_C_i Faça
8                Processar inicializacao(ACO);
9                Processar solucao(p);
10
11                   p_ij^k = (τ_ij)^α (η_j)^β / Σ_{i∈N_i^k} (τ_ij)^α (η_j)^β
12
13               Processar atualizar_feromonio(τ);
14                   τ_ij^k ← τ_ij^k + Δτ^k
15               Processar evaporacao_feromonio(τ);
16                   τ_ij^k ← (1−ρ) τ_ij^k
17               C_i = C_i + 1;
18           FimPara
19       i = i + 1;
20 Fim
21 gerar_ACO ()
```

Figura 4. Pseudocódigo - ACO.

(*i*) Construção das soluções com as formigas → nosso esquema de solução, seguindo o definido por [Dorigo 2004] inicia-se com varredura aleatória de doze (12) (*m*) formigas artificiais (*k*), em um primeiro estágio (E_1), (num total de 30 execuções em 1000 estágios - E_i) partindo do ninho (n_i), com escolha, com mesma probabilidade → p_{ij}^{k} = 16,67%, entre uma das seis arestas (A_i), observados na Figura 3 e seguindo o pseudocódigo da Figura 4.

Cada aresta A_i é definida com o valor do parâmetro λ sendo $x - 2 < \lambda < x + 3$, para cada um dos quatro (04) níveis de segurança de λ (*Toy* → x= 42, *Small* → x= 52, *Medium* → x= 62 e *Large* → x= 72) – tamanhos definidos por [Coron 2011]. (*ii*) atualização de feromônio → Inicialmente todas as arestas A_i possuem a mesma quantidade de feromônio $\tau_{ij}=\tau_{Ai}=0{,}01$. A atualização de feromônio, é dada por: $\tau_{ij}^{k} \leftarrow \tau_{ij}^{k} + \Delta\tau^{k}$, sendo $\tau^{k} = 0{,}001$ [Dorigo 2004]. A evaporação respeita a equação $\tau_{ij}^{k} \leftarrow (1-\rho)\ \tau_{ij}^{k}$ com $\rho \in [0,1]$, que varia randomicamente adaptando-se as perdas "naturais" como "chuva" e "calor" (artificiais) [Castro 2004]. A valoração da taxa de depósito de feromônio α é a de evaporação são diretamente proporcionais aos níveis de segurança semântica e a da blindagem ao ataque do MDC aproximado [Coron 2011], verificações essas testadas a cada iteração do algoritmo.

(*iii*) Ações *daemon* [Dorigo 2004] → os valores da base heurística são associados à segurança ao ataque do aniversário [Gentry 2010], e da mesma forma que o processo para a atualização do feromônio, é realizado o processamento a cada iteração do método.

Tabela 3. Tabela dos resultados obtidos pelas heurísticas: (a) AG; (b) ACO.

$S_{H\text{-}1}$	$S_{H\text{-}2}$	$S_{H\text{-}3}$	$S_{H\text{-}4}$
$T\lambda_{-1} = 0{,}10$ $P_1 = 40$	$T\lambda_{-1} = 0{,}10$ $P_1 = 50$	$T\lambda_{-1} = 0{,}10$ $P_1 = 60$	$T\lambda_{-1} = 0{,}10$ $P_1 = 70$
$T\lambda_{-2} = 0{,}25$ $P_2 = 41$	$T\lambda_{-2} = 0{,}24$ $P_2 = 51$	$T\lambda_{-2} = 0{,}25$ $P_2 = 61$	$T\lambda_{-2} = 0{,}25$ $P_2 = 71$
$T\lambda_{-3} = 0{,}23$ $P_3 = 42$	$T\lambda_{-3} = 0{,}22$ $P_3 = 52$	$T\lambda_{-3} = 0{,}22$ $P_3 = 62$	$T\lambda_{-3} = 0{,}23$ $P_3 = 72$
$T\lambda_{-4} = 0{,}17$ $P_4 = 43$	$T\lambda_{-4} = 0{,}17$ $P_4 = 53$	$T\lambda_{-4} = 0{,}17$ $P_4 = 63$	$T\lambda_{-4} = 0{,}17$ $P_4 = 73$
$T\lambda_{-5} = 0{,}14$ $P_5 = 44$	$T\lambda_{-5} = 0{,}15$ $P_5 = 54$	$T\lambda_{-5} = 0{,}14$ $P_5 = 64$	$T\lambda_{-5} = 0{,}14$ $P_5 = 74$
$T\lambda_{-6} = 0{,}11$ $P_6 = 45$	$T\lambda_{-6} = 0{,}12$ $P_6 = 55$	$T\lambda_{-6} = 0{,}12$ $P_6 = 65$	$T\lambda_{-6} = 0{,}11$ $P_6 = 75$

(a)

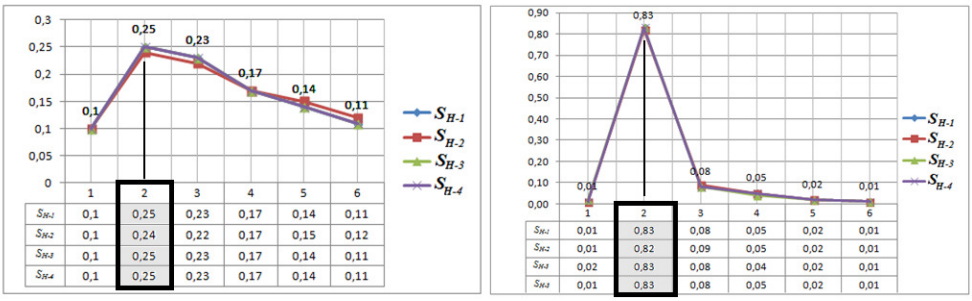
$S_{H\text{-}1}$	$S_{H\text{-}2}$	$S_{H\text{-}3}$	$S_{H\text{-}4}$
$T\lambda_{-1} = 0{,}01$ $P_1 = 40$	$T\lambda_{-1} = 0{,}01$ $P_1 = 50$	$T\lambda_{-1} = 0{,}02$ $P_1 = 60$	$T\lambda_{-1} = 0{,}01$ $P_1 = 70$
$T\lambda_{-2} = 0{,}83$ $P_2 = 41$	$T\lambda_{-2} = 0{,}82$ $P_2 = 51$	$T\lambda_{-2} = 0{,}83$ $P_2 = 61$	$T\lambda_{-2} = 0{,}83$ $P_2 = 71$
$T\lambda_{-3} = 0{,}08$ $P_3 = 42$	$T\lambda_{-3} = 0{,}09$ $P_3 = 52$	$T\lambda_{-3} = 0{,}08$ $P_3 = 62$	$T\lambda_{-3} = 0{,}08$ $P_3 = 72$
$T\lambda_{-4} = 0{,}05$ $P_4 = 43$	$T\lambda_{-4} = 0{,}05$ $P_4 = 53$	$T\lambda_{-4} = 0{,}04$ $P_4 = 63$	$T\lambda_{-4} = 0{,}05$ $P_4 = 73$
$T\lambda_{-5} = 0{,}02$ $P_5 = 44$	$T\lambda_{-5} = 0{,}02$ $P_5 = 54$	$T\lambda_{-5} = 0{,}02$ $P_5 = 64$	$T\lambda_{-5} = 0{,}02$ $P_5 = 74$
$T\lambda_{-6} = 0{,}01$ $P_6 = 45$	$T\lambda_{-6} = 0{,}01$ $P_6 = 55$	$T\lambda_{-6} = 0{,}01$ $P_6 = 65$	$T\lambda_{-6} = 0{,}01$ $P_6 = 75$

(b)

Podemos observar na Tabela 3 os resultados obtidos pelas heurística e meta-heurística em estudo: (a) Valores obtidos pelo algoritmo genético [Gavinho 2016]; (b) Valores obtidos pelo ACO em nossa proposta**.** Os valores dos pesos dados pelo AG, Tabela 3 (a) convergem, em seu valor máximo para $T\lambda_{-2} = 0.25$, e sendo $T\lambda_1 < T\lambda_6 < T\lambda_5 < T\lambda_{-4} < T\lambda_3$. Tendo a convergência do ACO, Tabela 3 (b), direcionando-se ao valor de $T\lambda_2 = 0{,}83$, como percurso de maior concentração de feromônio artificial.

Representamos a Tabela 3 em forma de gráficos com a finalidade de maior visualização dos resultados, observado na Figura 5. A Otimização por Colônia de Formigas selecionou como o "melhor caminho" (melhor solução), exatamente os tamanhos do parâmetro λ iguais a: *Toy* = 41, *Small* = 51, *Medium* = 61 e *Large* =71,

com valoração média de 0.83, Figura 5 (b). Valores esses verificados com as suas atribuições de peso pelo AG, Figura 5 (a).



(a) (b)

Figura 5. Gráficos de convergência: (a) do algoritmo genético; (b) ACO.

Ratificando e validando com isso o trabalho de [Gavinho 2015] onde foi verificado que, com a redução de um bit em cada nível de segurança do parâmetro λ, Tabela 4, (*Toy* = 41, *Small* = 51, *Medium* = 61 e *Large* =71) mantém-se os mesmos níveis de segurança quando comparados aos tamanhos definidos por [Coron 2012], bem como a redução dos tempos de execução do Algoritmo Criptográfico Totalmente Homomórfico.

Tabela 4. Tamanhos reduzidos dos Parâmetros *λ* com Segurança Semântica

Parâmetro	*Toy*	*Small*	*Medium*	*Large*
λ	41	51	61	71

4. Conclusão e Trabalhos Futuros.

Neste trabalho foi utilizada a Otimização por Colônia de Formigas para validar e ratificar as demonstrações das variantes de algoritmos presentes na literatura.Tais variantes utilizaram a heurística dos Algoritmos Genéticos para demonstrar, por meio de otimização e calibração, a redução dos tamanhos das chaves públicas e do tempo de execução das variantes das primitivas de Coron. Mantendo, mesmo assim, a segurança semântica do mecanismo e, alcançando, em consequência disso, a redução dos tempos de execução de todo o processo. Demonstramos que as trilhas de feromônio concentraram-se, em sua maior quantidade, exatamente nos tamanhos de chaves determinados pelo algorítimo em estudo. Como trabalho futuro, procuraremos melhorar a técnica utilizando a Otimização por Colônia de Formigas na proposta de Coron em sua segunda variante, por meio da: definição, calibração e fixação do valor da semente utilizada para a geração de chaves públicas, que no trabalho original de Coron é escolhida por meio de um gerador de números pseudo-aleatórios.

Referências

Boneh, D., Halevi, S., Hamburg, M., et al. “Circular-secure encryption from decision diffie-hellman”. In: Advances in Cryptology–CRYPTO 2008,Springer, 2008.

Buchmann, Johannes A. “Introdução a Criptografia”. Ed. Berkeley, São Paulo, 2002.

Brakerski, Z., Gentry, C., Vaikuntanathan, V. “Fully homomorphic encryption without bootstrapping”, ITCS 2012, 2012.

Coron, J., Naccache, D., Tibouchi, M. Optimization of Fully homomorphic Encryption. Cryptology ePrint Archive, Report 2011/440, 2012.

Castro, L. N. de. Fundamentals of Natural Computing: Basic Concepts, Algorithms, and Applications. Chapman & Hall/CRC, 2006.

Coron, J., Mandal, A., Naccache, D., et al. "Fully homomorphic encryption over the integers with shorter public keys", Advances in Cryptology–, pp. 487–504, 2011.

CSA Security Guidance for Critical Areas of Focus in Cloud Computing –v2.1. Cloud Security Alliance, 2009.

Daniel J. Bernstein. Introduction to post-quantum cryptography. In Post-Quantum Cryptography. 2009.

Dijk., M. Van, Gentry, C., Halevi, S. e Vaikuntanathan, V., Fully homomorphic encryption over the integers. *In H. Gilbert* (Ed.), EUROCRYPT 2010.

Dorigo, M.; Gambardella, L. M., Ant Colony System: A cooperative learning approach to the traveling salesman problem. IEEE TEC 1997.

Dorigo, M.; Stutzle, T. Ant Colony Optimization. Massachusetts Institute of Technology. Cambridge, 2004.

Gavinho, Joffre Filho; Micelli, C; Pereira, G. "Compressão e Otimização de Chaves Públicas usando Algoritmo Genético em Criptografia Completamente Homomórfica"- XV Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais - SBSEG 2015 - Florianópolis/SC, Brasil - Novembro 09-12, 2015.

Gavinho, Joffre Filho; Micelli, C; Pereira, G. " A Public Key Compression Method for Fully Homomorphic Encryption using Genetic Algorithms!" 19th International Conference on Information Fusion – Heildeberg - Alemanha - 2016.

Gentry, C. "Fully homomorphic encryption using ideal lattices". In: Proceedings of the 41st annual ACM Symposium on Theory of computing, pp. 169–178. ACM, 2009.

Keeney, R.L. & RAIFFA, H. Decisions with Multiple Objectives: Preferences and Value Tradeoffs , John Wiley & Sons, Nova Iorque, 1976.

Lacerda, E.G.M e Carvalho, A.C.P.L. "Introdução aos Algoritmos Genéticos", In: Sistemas inteligentes: aplicações a recursos hídricos e ciências ambientais. Editado por Galvão, C.O., Valença, M.J.S. Ed. Universidade/UFRGS: ABRH, 1999.

Michael O. Rabin. Probabilistic algorithm for testing primality. Journal of Number Theory , 12(1):128 – 138, 1980.

Morris, Christopher , "Navy Ultra's Poor Relations", in Hinsley, F.H.; Stripp, Alan, *Codebreakers: The inside story of Bletchley Park*, : Oxford University Press, 1993

NIST- National institute of standards and technology. Cyber security Framework Development Overview.NIST's Role in Implementing Executive Order 7213636, Improving Critical Infrastructure Cybersecurity, Presentation to ISPAB, 2013.

Mukherjee P, Wichs D. Two round MPC from LWE via multi-key FHE. IACR Cryptology ePrint Archive , 2015. To appear in Proceedings of Eurocrypt 2016.

Rivest R. L., L Adleman, and M L Dertouzos. On data banks and privacy homomorphisms, in r. a. demillo et al. In Eds.), FSC. Academic Press, 1978.

Smart, N.; Vercauteren, F. Fully homomorphic encryption with relatively small key and ciphertext sizes. Cryptology ePrint Archive, Report 2009/571, 2009.

Stalling, Willian, Criptografia e Segurança de Redes: Princípios e Práticas 4. Ed. Prentice Hall Brasil, pag 17-36, 2007.

Sousa, F. R. C.; Moreira, L. O.; Machado, J. C. Computação em Nuvem: Conceitos, Tecnologias, Aplicações e Desafios. Fortaleza, 2009.

William S. SAGE: A Computer System for Algebra and Geometry Experimentation, 2012.

