Ciencias Matemáticas, Vol. 33, No. 1, Pag. 71-74, 2019

Ataques a cifrados en bloques mediante búsquedas en grupos cocientes de las claves
Attacks to block ciphers by searching in quotient groups of the keys

Osmani Tito Corrioso[1]*, Miguel A. Borges Trenard[2], Mijail Borges Quintana[3]

Resumen El Algoritmo Genético (AG) ha sido aplicado en los últimos años con el propósito de realizar ataques a los cifrados en bloques. Buscar la clave en todo el espacio de las claves puede llegar a ser imposible en la práctica, en este sentido, el principal objetivo de este trabajo es realizar una partición del espacio de las claves en clases de equivalencia, usando Teoría de Grupos, y así concentrar el ataque sobre estas clases. Los resultados se aplican a un criptoanálisis al cifrado AES(t).

Abstract The Genetic Algorithm (GA) has been applied in the last years with the purpose of making attacks on block ciphers. Searching for the key in the whole key space can be impossible in practice, in this sense, the main objective of this work is to make a partition of the key space in equivalence classes, by using Group Theory, and thus to concentrate the attack on these classes. Results are applied to a cryptanalysis on the cipher AES(t).

Palabras Clave
Algoritmo Genético, grupo cociente, criptoanálisis, AES(t)

Keywords
Genetic Algorithm, quotient group, cryptanalysis, AES(t)

[1] *Departamento de Matemática, Facultad de Ciencias de la Educación, Universidad de Guantánamo, Cuba, osmanitc@cug.co.cu*
[2,3] *Departamento de Matemática, Facultad de Ciencias Naturales y Exactas, Universidad de Oriente, Cuba, [2]mborges@uo.edu.cu, [3]mijail@uo.edu.cu*
***Autor para Correspondencia, Corresponding Author**

Introducción

El Algoritmo Genético (AG) es un método de optimización utilizado en los últimos años en la Criptografía con diversos propósitos, en particular con el de realizar ataques a varios tipos de cifrados. Algunas de las investigaciones realizadas en esta dirección se mencionan abajo.

En [2] se describe un método para descifrar mensajes con máquinas de rotor utilizando el AG para buscar en el espacio de las claves. El autor describe una función de aptitud basada en la prueba phi. En [12] el autor prueba el AG con un cifrado de sustitución mono-alfabética con resultados aceptables. Los experimentos fueron realizados con una muestra de 4 claves y diferentes variantes de los operadores de mutación y cruzamiento. Los resultados de este artículo muestran que haciendo una selección aleatoria de los individuos y el cruzamiento por un punto, el algoritmo es más rápido que con otras combinaciones (como selección por torneos). [9] describe un AG llamado DUREHA, con el objetivo de automatizar el ataque a cifrados clásicos y de esta forma ahorrar tiempo y recursos disponibles en el criptoanálisis.

En [14] los autores presentan una combinación del Algoritmo Genético con la Optimización de Enjambres de Partículas (otro método heurístico basado en técnicas evolutivas), llamaron a su método "Optimización de Enjambre Genético" y lo aplicaron para atacar el DES. Sus resultados experimentales muestran que se obtienen mejores resultados aplicando su método combinado que utilizando ambos métodos por separado. [7] proporciona una exploración preliminar del uso del AG sobre un cifrado del tipo Red de Sustitución Permutación (SPN por sus siglas en inglés). El propósito de la exploración es determinar cómo encontrar claves débiles. Ambos trabajos ([14] y [7]) usan un ataque a texto claro conocido, es decir, dado un texto claro T y el correspondiente texto cifrado C, se está interesado en encontrar la clave K. En [7], la función de aptitud evalúa la diferencia bit a bit (distancia de Hamming) entre C y el texto cifrado de T, usando un candidato para la clave, mientras que, por el contrario, en [14] se mide la distancia de Hamming entre T y el descifrado del texto cifrado de C. En [10] se muestra un ataque sólo a texto cifrado al SDES, obteniendo mejores resultados que por fuerza bruta. Los autores usan una función de aptitud que es una combinación de la frecuencia relativa de monogramas, digramas, y trigramas (para un idioma particular). Como la longitud de clave es muy pequeña, pudieron usar este tipo de función. [1] es similar a [10], se utiliza en esencia la misma función de aptitud, pero con diferentes parámetros, también es más detallado sobre los

experimentos y los comparan no sólo con respecto por fuerza bruta, sino también con la búsqueda aleatoria.

En [4] se propone una nueva función de aptitud que compara no solo la cercanía entre los textos cifrados, sino también entre la clave inicial y algunos bits de las posibles claves. Por otra parte, en [5] se propone un procedimiento para particionar el espacio de las claves basándose, en cierto sentido, en una cierta congruencia aritmética, dividiendo el espacio en intervalos para luego orientar el ataque en uno de estos intervalos. Los autores adaptan el AG con este propósito y realizan un ataque al cifrado AES(t).

Una descripción de lo que se ha hecho en los últimos años en el área del criptoanálisis mediante el Algoritmo Genético se puede consultar en [8], [3] y [11].

En este trabajo se sigue la idea de [5], de dividir el espacio de las claves. En este caso se propone realizar la partición usando Teoría de Grupos. El propósito es particionar el espacio de las claves en clases de equivalencias contenidas en cierto grupo cociente y centrar la búsqueda de claves consistentes en estas clases. El procedimiento y los experimentos realizados se comparan con los obtenidos en [5].

1. Desarrollo

1.1 Particionando el espacio de las claves

En esta sección se describe el procedimiento para realizar la partición y calcular el grupo cociente de las claves, así como la metodología para moverse por cada elemento de cada clase de equivalencia.

Sea $\mathbb{F}_2^{k_1}$ el espacio de las claves de longitud $k_1 \in \mathbb{Z}_{>0}$. Es conocido que, como grupo aditivo, es isomorfo a $\mathbb{Z}_{2^{k_1}}$. Sea h el homomorfismo definido del modo siguiente:

$$\begin{array}{rcl} h : \mathbb{Z}_{2^{k_1}} & \longrightarrow & \mathbb{Z}_{2^{k_2}} \\ n & \longrightarrow & n \pmod{2^{k_2}}, \end{array} \tag{1}$$

donde $k_2 \in \mathbb{Z}_{>0}$ y $0 < k_2 < k_1$. Denotemos por N al núcleo de h, es decir,

$$N = \{x \in \mathbb{Z}_{2^{k_1}} | h(x) = 0 \in \mathbb{Z}_{2^{k_2}}\}.$$

Luego, por la definición de h se tiene que N está formado por los elementos de $\mathbb{Z}_{2^{k_1}}$ que son múltiplos de 2^{k_2}. Se sabe que N es un subgrupo invariante (o normal), por tanto, el principal objetivo es calcular el grupo cociente de $\mathbb{Z}_{2^{k_1}}$ por N y de esta forma el espacio de las claves quedará dividido en 2^{k_2} clases de equivalencia.

1.1.1 Calculando el grupo cociente de las claves

Denotemos por G_K al grupo cociente de $\mathbb{Z}_{2^{k_1}}$ por N, o sea,

$$G_K = \mathbb{Z}_{2^{k_1}}/N$$

Por el Teorema de Lagrange, se tiene que,

$$o(G_K) = o(\mathbb{Z}_{2^{k_1}})/o(N)$$

pero,

$$o(G_K) = o(\mathbb{Z}_{2^{k_2}}) = 2^{k_2}$$

luego,

$$o(N) = o(\mathbb{Z}_{2^{k_1}})/o(\mathbb{Z}_{2^{k_2}}) = 2^{k_1-k_2}$$

Ahora se puede describir a N, teniendo en cuenta que sus elementos son los múltiplos de 2^{k_2}. Para ello tomemos $Q = \{0,1,2,\ldots,2^{k_1-k_2}-1\}$, entonces:

$$\begin{array}{rcl} N & = & <2^{k_2}> = \{x \in \mathbb{Z}_{2^{k_1}} | \exists\, q \in Q, x = q2^{k_2}\} = \\ & = & \{0, 2^{k_2}, 2*2^{k_2}, 3*2^{k_2}, \ldots, (2^{k_1-k_2}-1)*2^{k_2}\}. \end{array}$$

Por otra parte,

$$G_K = \{N, 1+N, 2+N, \ldots, (2^{k_2}-2)+N, (2^{k_2}-1)+N\}$$

De esta forma $\mathbb{Z}_{2^{k_1}}$ queda dividido en una partición de 2^{k_2} clases dadas por N. G_K se denominará *grupo cociente de las claves*. Semejante a [5], con G_K también se busca recorrer, del espacio total, los elementos que se encuentran en una partición, para luego encontrar una, o varias, claves consistentes en esas particiones (en este caso son clases de equivalencia). El grupo cociente de las claves le brinda propiedades algebraicas a la partición, lo cual permite que luego se pueda seguir trabajando en ella.

1.1.2 El problema de recorrer cada elemento de cada clase de equivalencia

Teniendo en cuenta que $\mathbb{Z}_{2^{k_2}}$ es isomorfo con G_K y el isomorfismo hace corresponder a cada $r \in \mathbb{Z}_{2^{k_2}}$ su clase de equivalencia $r+N$ en G_K, se tiene que, seleccionar una clase es fijar un elemento $r \in \mathbb{Z}_{2^{k_2}}$. Por otro lado, los elementos de N tienen la forma $q2^{k_2}$ ($q \in Q$), por tanto, los elementos de la clase $r+N$ tienen la forma

$$q2^{k_2} + r,\ q \in Q. \tag{2}$$

Luego, el problema de recorrer cada elemento de cada clase de equivalencia se reduce a fijar primero un elemento de $\mathbb{Z}_{2^{k_2}}$ y luego recorrer cada elemento del conjunto Q, para buscar una clave consistente en G_K mediante (2). Los elementos del conjunto Q tendrán longitud de bloque $k_d = k_1 - k_2$ y cada clase tendrá 2^{k_d} elementos (como se explicó antes). La metodología propuesta en [5] se basa en la partición del espacio de llaves en intervalos de igual longitud. La misma parte del hecho de que $\mathbb{F}_{2^{k_1}}$ tiene cardinal 2^{k_1} y por tanto hay una correspondencia uno a uno entre $\mathbb{F}_{2^{k_1}}$ y el intervalo entero $[0, 2^{k_1}-1]$. Si se fija un entero k_2, $(1 < k_2 \leq k_1)$, entonces el espacio de las claves puede ser representado por los números,

$$q2^{k_2} + r \tag{3}$$

donde $q \in [0, 2^{k_1-k_2}-1]$ y $r \in [0, 2^{k_2}-1]$. De esta forma el espacio queda dividido en $2^{k_1-k_2}$ bloques (determinados por el algoritmo de la división dividiendo por 2^{k_2}) y, dentro de cada bloque, la clave correspondiente está determinada por su posición dada por el resto r. Dado un valor para k_2 y la

correspondiente partición, la idea clave es moverse con el AG únicamente dentro de un bloque (dado por q) en la partición, y luego recorrer los elementos r en dicho bloque. Por lo que, las operaciones de mutación y cruzamiento se hacen a los elementos r de forma cerrada en cada bloque, pero la función de aptitud se le calcula a la clave que le corresponde a r en el espacio completo mediante (3). De forma análoga se aplica el AG en G_K, sustiyendo en este trabajo el papel de r en [5] por q.

2. Experimentación y resultados

Los experimentos se realizaron con el cifrado AES(t), para $t = 3$. AES(t) es una versión paramétrica del cifrado AES, en particular AES(8) = AES. Una descripción de AES(t) puede consultarse en [6]. Para realizar el ataque a este cifrado se utilizó el criptoanálisis mediante el AG. Una explicación detallada del funcionamiento del AG y su implementación se puede ver en los trabajos [13], [4] y [5].

Es bastante semejante lo expuesto aquí y lo que se propone en [5], es decir:

- Se toma la misma cantidad de parámetros de entrada: k_2, q, y r.
- La fórmula (2) sigue siendo la misma.

En contraste:

- En [5], q indica la partición y r recorre cada elemento de esa partición. En este trabajo es al contrario de lo anterior, se fija r (que indica la partición o clase de equivalencia en este caso) y q varía.
- En [5], la longitud de bloque para el movimiento es k_2, mientras que aquí es $k_d = k_1 - k_2$.
- Por lo anterior, para la implementación del AG mediante búsquedas en las particiones de G_K, es posible utilizar los mismos códigos utilizados en [5]; sólo es necesario intercambiar las funciones de r y q, así como sustituir k_2 por k_d.

Se usó una computadora con procesador Intel(R) Core (TM) i3-4170 CPU @3.70GHz (4CPUs) $\sim$ 3.7GHz, y la versión 17 de Maple. La longitud de clave es $k_1 = 48$, la longitud de clave para el movimiento es $k_d = 32$ y $k_2 = 16$ (o sea, son 2^{32} clases de equivalencia). Las probabilidades para el cruzamiento y la mutación son 0.6 y 0.2 respectivamente. El número de generaciones es 655 y la población se inicializa con 100 individuos elegidos de forma aleatoria. El experimento se repitió 20 veces y los resultados obtenidos son los siguientes:

1. Se encontró la clave en un 65 % de los casos.
2. Se llegó a la solución en un promedio de 427 generaciones.
3. El tiempo promedio en que se encontró la clave fue de 1227.6 segundos (aproximadamente 20.46 minutos) y en los casos en que no se encontró la clave, el programa terminó en un promedio de 28.5 minutos, o sea, este es el tiempo que se tardó en recorrer las 655 generaciones.

3. Conclusiones

El cálculo del grupo cociente de las claves abre la cantidad de posibilidades y maneras de realizar la partición y la búsqueda de claves consistentes mediante el AG. En este sentido, no se trata de una alternativa mejor que lo propuesto en [5], sino, de una alternativa más. Sin embargo, esta partición permite seguir trabajando en ella con la aplicación de otras técnicas del Álgebra Abstracta en futuros trabajos. Los resultados del ataque al AES(3) con el Algoritmo Genético son bastante parecidos en ambos artículos. Por el momento se mantiene el problema de elegir el mejor valor para k_2 y por tanto para k_d: un valor óptimo entre la cantidad de clases y la cantidad de elementos de cada clase.

Se puede utilizar la ecuación (2) para resumir las diferencias entre la metodología propuesta en [5] y la de este trabajo. En ambos casos consiste en mantener el funcionamiento del AG sobre un subconjunto del espacio de claves en lugar del espacio completo. En el caso de [5] el subconjunto está asociado a la clase determinada por la fórmula en (2) de las claves que corresponden a un mismo cociente (q), note que éste representa un subconjunto de claves consecutivas dentro del espacio completo. Por otra parte, la metodología expuesta en este trabajo consiste en trabajar con el subconjunto dado por las claves asociadas en la fórmula en (2) con el mismo resto (r). En este caso el subconjunto se encuentra esparcido a lo largo del conjunto de claves.

Resulta de interés continuar comparando los enfoques de [5] y el presentado aquí, en tal sentido, es recomendable destacar el siguiente hecho (implícito en las consideraciones anteriores): En [5], mientras mayor es la longitud de clave k_2, el espacio a recorrer en cada partición es superior, por lo que habrán menos particiones. En el presente trabajo, a mayor k_2, menor es el espacio a recorrer en cada clase de equivalencia, pero habrán más clases.

Referencias

[1] Adwan, Al, M. Al Shraideh y M.R.S. Al Saidat: *A Genetic Algorithm Approach for Breaking of Simplified Data Encryption Standard*. International Journal of Security and Its Applications, 9(9):295–304, 2015. http://www.sersc.org/journals/IJSIA/vol9no92015/26.pdf.

[2] Bagnall, A. J.: *The applications of Genetic Algorithms in Cryptanalysis*. Thesis submitted for the degree of Master of Science, School of Information Systems, University of East Anglia, 1996. http://www2.cmp.uea.ac.uk/ ajb/Download/MSc_thesis.pdf.

[3] Baragada, SR. y P.S. Reddy: *A Survey of Cryptanalytic Works Based on Genetic Algorithms*. International Journal of Emerging Trends & Technology in Computer Science (IJETTCS). ISSN 2278-6856, 2(5), September-October 2013. http://www.ijettcs.org/Volume2Issue5/IJETTCS-2013-08...

[4] Borges-Trenard, M.A., M. Borges-Quintana, A. Donatien-Charón y L. Monier-Columbié: *Nueva función de aptitud en el criptoanálisis genético a cifrados en bloques*. Congreso Internacional COMPUMAT. La Habana, Cuba, 2017.

[5] Borges-Trenard, M.A., M. Borges-Quintana y L. Monier-Columbié: *An application of genetic algorithm to cryptanalysis of block ciphers by partitioning the key space*. Mathematics Department, Faculty of Exact and Natural Sciences, University of Oriente, Santiago de Cuba, Cuba, 2018. Enviado a J. Discrete Mathematical Sciences and Criptografy.

[6] Borges-Trenard, M.A. y L. Monier-Columbié: *AES(t): Una versión parametrizada del AES*. Congreso Internacional COMPUMAT. La Habana, Cuba, 2015.

[7] Brown, J.A., S.K. Houghten y B. Ombuki-Berman: *Genetic Algorithm Cryptanalysis of a Substitution Permutation Network*. IEEE Symposium on Computational Intelligence in Cyber Security, páginas 115–121, 2009.

[8] Delman, Bethany: *Genetic algorithms in cryptography*. Thesis. Rochester Institute of Technology, RIT Scholar Works, 2004. http://scholarworks.rit.edu/cgi/viewcontent.cgi?article=6...

[9] Dureha, Anukriti y Arashdeep Kaur: *A Generic Genetic Algorithm to Automate an Attack on Classical Ciphers*. International Journal of Computer Applications (0975 - 8887), 64(12), 2013.

[10] Garg, P., S. Varshney y M. Bhardwaj: *Cryptanalysis of Simplified Data Encryption Standard Using Genetic Algorithm*. American Journal of Networks and Communications, 4(3):32–36, 2015. http://article.sciencepublishinggroup.com/pdf/10.11648.j...

[11] Khan, A.H., A.H. Lone y F.A. Badroo: *The Applicability of Genetic Algorithm in Cryptanalysis: A Survey*. International Journal of Computer Applications, 130(9), 2015. http://www.ijcaonline.org/research/volume130/number9/...

[12] N, P. Shreeraj: *Application of Genetic Algorithm in Cryptanalysis of Mono-alphabetic Substitution Cipher*. International Journal of Trend in Scientific Research and Development, 1(4), 2017. http://www.ijtsrd.com/papers/ijtsrd2191.pdf.

[13] Trenard, M.A. Borges, M. Borges Quintana y A. Donatien Charón: *Algoritmo Genético en cifradores modernos*. Congreso Internacional COMPUMAT. La Habana, Cuba, 2015.

[14] Vimalathithan, R. y M.L. Valarmathi: *Cryptanalysis of DES using Computational Intelligence*. European Journal of Scientific Research, ISSN 1450-216X, 55(2):237–244, 2011.