De la búsqueda de funciones booleanas con buenas propiedades criptográficas

Francisco Rodríguez Henríquez
Departamento de Computación, CINVESTAV-IPN
Av. IPN 2508, Col. Zacatenco
07300, México D.F.
correo-e: francisco@cs.cinvestav.mx

Resumen

Se presenta una descripción general de las técnicas computacionales y principios matemáticos modernos que se utilizan en el problema de búsqueda de funciones booleanas con muy alta no linealidad y otras propiedades criptográficas necesarias para su aplicación en la criptografía de llave simétrica.

1. Introducción

Las cajas de substitución (cajas *S*) constituyen la piedra angular en criptografía para lograr que los cifradores por bloque exhiban la ineludible propiedad de no linealidad. [1] En efecto, si la o las cajas S de un determinado cifrador por bloque no alcanzan una alta no linealidad, entonces se considera que tal algoritmo no podrá ofrecer una seguridad adecuada para impedir que información confidencial pueda ser develada por entidades no autorizadas [27,31].

Formalmente, una caja *S* es una función o correspondencia de *n* bits de entrada a *m* bits de salida, $S : Z_2^n \rightarrow Z_2^m$, esto es, una caja S puede ser vista como una función booleana de *n* bits de entrada y *m* bits de salida. Cuando $n = m$ la función es reversible y por lo tanto biyectiva. Sin embargo en muchas ocasiones, las cajas *S* de los cifradores por bloque no son biyectivas. Por ejemplo, como se describe en la siguiente sección, el estándar de cifrado de datos (DES por sus siglas en inglés) emplea cajas *S* en las cuales el número de bits de entrada (seis) es mayor que el número de bits de salida (cuatro).

Dada su definición, es claro que el número de funciones booleanas elegibles para diseñar una caja *S* de *n* bits de entrada y *m* bits de salida está dado por 2^{m2^n}, de tal manera que aun para valores moderados de *n* y *m* el tamaño del espacio de búsqueda de este problema tiene un tamaño desmesurado (por ejemplo para el algoritmo DES el total de funciones booleanas candidatas es un número con 78 dígitos decimales).

[1] En la sección 2 se describen de manera general los cifradores por bloque; y en la sección 3 se define la propiedad de *no linealidad* (en el contexto de cifradores por bloque) de manera formal.

Sin embargo, no todas las funciones booleanas son apropiadas para construir buenas cajas *S*. Además de la ya mencionada propiedad de no linealidad, algunas de las principales propiedades criptográficas requeridas para dichas funciones booleanas incluyen: balance, alto grado algebraico, criterio de avalancha estricto, orden de inmunidad, etc.[2]

En general, los métodos para diseñar y construir funciones booleanas y cajas *S* pueden ser categorizados en tres tipos de técnicas: generación aleatoria, construcción algebraica y diseños heurísticos [12]. El método de generación aleatoria evita con facilidad una variedad de propiedades combinatorias que son consideradas debilidades criptográficas. Sin embargo, las funciones booleanas generadas por este método no suelen exhibir buenas propiedades de no linealidad. En contraste, las construcciones algebraicas pueden brindar propiedades combinatorias específicas y una muy alta no linealidad, no obstante, tienden a tener pobre calidad en aquellas características que no fueron específicamente consideradas durante su diseño [1,9,11,30,34-35].

Una tercera estrategia para diseñar funciones booleanas y cajas *S* se basa en diseños heurísticos [2-5,10,16-17]. En este apartado, las técnicas evolutivas han sido particularmente útiles debido, especialmente, a su muy alto poder exploratorio, que les permite evaluar a partir de una población de soluciones potenciales vastas regiones del espacio de diseño sin necesidad de agotar exhaustivamente todo el universo de posibilidades [12]. Entre los principales logros obtenidos en el problema del diseño eficiente de cajas S por parte de las heurísticas evolutivas se cuentan: hallazgo de funciones booleanas con hasta nueve entradas de máxima no linealidad, confirmación/refutación de conjeturas sobre la máxima no linealidad alcanzable con funciones no lineales de siete, ocho, nueve y diez entradas, etc. [2-5,10,16-17,22-23,28].

En este artículo se explican las aplicaciones de las cajas S en la llamada criptografía de llave secreta o simétrica, se describen los principios matemáticos básicos que están detrás del diseño de funciones booleanas con buenas propiedades criptográficas, y se explican varios métodos de búsqueda de dichas funciones basados en técnicas heurísticas. Finalmente, se presentan algunos de los retos y conjeturas relacionados a este ilustre problema combinatorio que, a pesar de décadas de intenso estudio, continúan abiertos.

El resto de este manuscrito está organizado como sigue. En la sección 2, se describen las aplicaciones, modos de uso y criterios de diseño que se utilizan en las cajas S de cifradores por bloque. En la sección 3 se definen las distintas representaciones de las funciones booleanas junto con el espectro de Walsh-Hadamard, que es una herramienta crucial para hacer la clasificación de tales funciones. Asimismo, se enlistan las principales propiedades matemáticas que deben exhibir las funciones booleanas a ser utilizadas como constructores de

[2] En la Sección 3 se presentan las definiciones formales de estas propiedades.

cajas S. Enseguida, en la sección 4, se esboza una metodología que mediante el uso de motores de búsqueda de heurística evolutiva permite hallar funciones booleanas con buenas propiedades criptográficas. En la sección 5, se hace un resumen y se dan algunas conclusiones y perspectivas del material cubierto en este manuscrito y se señalan algunos de los retos y problemas abiertos que han sido estudiados recientemente en la literatura abierta.

2. Uso de las cajas *S* en la criptografía de llave simétrica

Las técnicas científicas para la implementación de la seguridad computacional son desarrolladas por la criptografía, la cual puede sucintamente ser definida como el estudio del problema de cómo establecer un intercambio de información seguro a través del uso de un canal de comunicación que no lo es.

De manera general, los métodos de cifrado/descifrado pueden clasificarse en dos categorías: criptografía de llave simétrica y criptografía de llave pública. En el resto de esta sección se describen los aspectos de diseño más destacados de la primera clase.

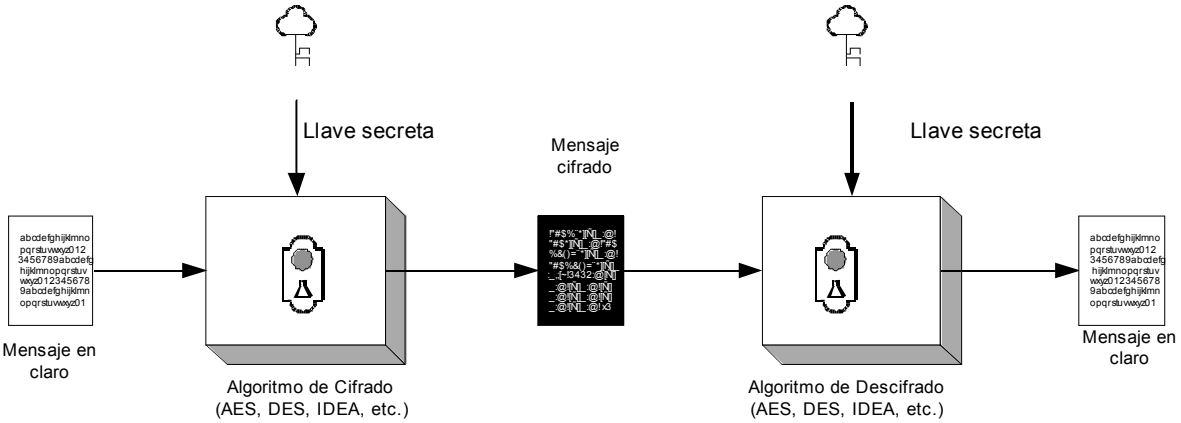

Fig 1. Modelo convencional de cifrado con llave simétrica

La figura 1 muestra esquemáticamente el proceso de cifrado y descifrado llevado a cabo en un sistema simétrico. En los algoritmos de llave simétrica (o secreta) se presupone que cada una de las partes legítimamente involucradas en la comunicación son las únicas entidades que tienen conocimiento de la llave empleada en el proceso de cifrado/descifrado. El conocimiento de esta llave permite el descifrado del texto, de ahí la razón de que ésta deba permanecer en el más estricto secreto.

En la terminología criptográfica al mensaje a ser cifrado se le conoce como *texto en claro*; al proceso de codificación al que se somete al mensaje para que no pueda ser develado por entidades no autorizadas se le llama *cifrado*; al documento que resulta de cifrar el mensaje se le conoce como *criptograma*; al proceso de recuperar el mensaje en claro a partir del criptograma se le llama *descifrado* [27]. Finalmente, el término *llave* o *clave* se refiere a un valor numérico

utilizado para alterar información haciéndola segura y visible únicamente a los individuos que tienen la llave correspondiente para recuperar dicha información [21].

Formalmente un *criptosistema* puede ser definido como una quíntupla {*P,C,K,E,D*}, donde [7]:

- *P* es el conjunto finito de los posibles textos en claro.
- *C* es el conjunto finito de los posibles textos cifrados.
- *K* el espacio de llaves, es un conjunto finito de todas las llaves posibles.
- $\forall k \in K \exists E_k \in E \text{ (regla de cifrado)} \exists D_k \in D \text{ (regla de descifrado)}$
  $\text{Cada } E_k : P \rightarrow C \text{ y } D_k : C \rightarrow P \text{ son funciones tales que } \forall x \in P, D_k(E_k(x)) = x$

Una subclase importante de algoritmos de llave simétrica está conformada por los cifradores simétricos por bloque que se caracterizan por dividir el texto en claro a ser cifrado en bloques de longitud fija, los cuales pueden ser o no procesados de forma independiente de acuerdo al modo de operación en que el cifrador por bloque sea utilizado.[3]

Algunos ejemplos famosos de cifradores por bloque son: el venerable estándar de cifrado de datos (DES) adoptado en el lejano año de 1974 [8,27,31], y su sucesor, el estándar avanzado de encriptación (AES), escogido en octubre de 2000 por el Instituto Nacional de Estándares y Tecnología (NIST[4] por sus siglas en inglés) como el estándar oficial en Estados Unidos para cifrar/descifrar documentos [20,24,26]. La principal ventaja de este tipo de esquemas es la sencillez matemática y consecuente eficiencia computacional de sus algoritmos, y su principal debilidad el manejo y distribución de las llaves secretas entre las partes interesadas[5]. Los tamaños de las llaves utilizadas para cifrar/descifrar varían desde los 64 bits (aun cuando hoy en día se necesitan al menos 80 bits para ser consideradas realmente seguras) hasta 256 bits [24,27,31].

En el caso del estándar DES, se utiliza una longitud de llave de apenas 56 bits. A pesar que en el tiempo de su creación esta longitud de llave fue considerada muy segura, los avances tecnológicos han permitido el desarrollo de técnicas para encontrar las llaves por búsqueda exhaustiva en tiempos relativamente cortos. Por ejemplo, ya desde 1999 un proyecto de cómputo distribuido *rompió* DES en un tiempo de 22 horas con 15 minutos. Debido a ello, desde hace mucho tiempo DES no es considerado suficientemente robusto para aplicaciones de alta seguridad, por lo que en la práctica profesional se utiliza una variante conocida como *triple*

---

[3] El estudio de los modos de operación en cifradores por bloque está fuera del enfoque de este artículo. Para una descripción más amplia de los modos de operación en cifradores por bloques, se recomienda el excelente artículo introductorio en:
http://en.wikipedia.org/wiki/Block_cipher_modes_of_operation.

[4] National Institute of Standards and Technology, http://www.nist.gov/.

[5] El número de llaves secretas crece cuadráticamente con el número *n* de usuarios en el sistema, dado que necesita un intercambio de $n(n-1)/2$ llaves para que todas las entidades puedan comunicarse unas con otras de manera confidencial.

DES, la cual brinda una seguridad equivalente a la proporcionada por una llave de 112 bits [27].

Los cifradores por bloque ofrecen diversos grados de seguridad, determinados esencialmente por el ya mencionado tamaño en bits de la llave secreta y por la propia calidad criptográfica en el diseño de cada cifrador. Hoy en día, muchos de los diseños más importantes de cifradores por bloque siguen el modelo de Feistel. Ello se debe a que este modelo se caracteriza por su simplicidad, buenas propiedades criptográficas y una robustez inherente. Por otro lado, los cifradores por bloque de Feistel han adquirido un enorme prestigio tras resistir exitosamente el escrutinio exhaustivo que sobre ellos ha realizado la comunidad criptográfica de manera implacable a lo largo de los últimos treinta años [31].

Los cifradores de Feistel utilizan transformaciones lineales en forma de corrimientos lógicos, operadores booleanos a nivel bit, etc., y transformaciones no lineales que son implementadas con bloques de substitución de bits, conocidos en la literatura especializada como cajas S. Dado que las cajas S son los únicos bloques no lineales presentes en el modelo de Feistel, se acepta de manera general que la calidad en eficiencia y seguridad de un algoritmo cifrador depende en buena medida del buen diseño de dichos módulos.

Por ejemplo, en el caso del cifrador DES están definidas un total de 8 cajas S. Cada una de estas cajas puede ser representada como una tabla con 64 casillas dispuestas en 4 renglones y 16 columnas [27,31]. El proceso de substitución de un valor de entrada de 6 bits por uno de salida de 4 bits[6] opera de la siguiente manera:

Dado el dato de entrada, $a_0a_1a_2a_3a_4a_5$, el primer y último bit, esto es, a_0a_5, representan el número de renglón, mientras que los 4 bits restantes, esto es, $a_1a_2a_3a_4$, representan el número de columna. Así, cualquiera de las 8 cajas S de DES substituirá *A*=101011 con el valor almacenado en el cuarto renglón (11) y la sexta casilla (0101). Por ejemplo si al dato *A* se le aplica la caja de substitución S_3 (véase figura 2), el valor substituido será 1001 (9). Si en cambio el mismo dato *A* es substituido utilizando la caja S_7, el resultado de la substitución será 0100 (4).

Como se muestra en figura 2, todos los renglones de todas las cajas S de DES son permutaciones de los dieciséis números enteros que pueden representarse con 4 bits, esto es: 0,1,…,15. Además los valores contenidos en las cajas S fueron diseñados así que si se tienen dos datos de entrada que difieren por un solo bit, las correspondientes salidas diferirán por al menos dos bits.[7]

---

[6] De acuerdo a la información pública que se tiene sobre el diseño de DES, se decidió que las cajas S tuvieran seis bits de entrada y 4 de salida, por ser el máximo valor práctico que permitía la tecnología de la época (mediados de los años 70's) [27].

[7] Esta propiedad, conocida como *criterio de avalancha*, se discute formalmente en la siguiente sección.

Resulta interesante señalar que aunque los criterios completos de diseño de las cajas S de DES no han sido nunca desclasificados, la perspectiva que dan más de treinta años de criptoanálisis permite tener hoy la casi certeza que sutiles debilidades fueron introducidas a propósito en las propiedades no lineales de las cajas S, debilidades que hicieron a DES vulnerable al criptoanálisis diferencial y lineal inventados en los años noventa [31].[8]

Renglón	Columnas																Cajas
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	S
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7	S1
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8	
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0	
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13	
0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10	S2
1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5	
2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15	
3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9	
0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8	S3
1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1	
2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7	
3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12	
0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15	S4
1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9	
2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4	
3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14	
0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9	S5
1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6	
2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14	
3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3	
0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11	S6
1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8	
2	9	14	15	5	2	8	12	3	7	0	14	10	1	13	11	6	
3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13	
0	4	11	2	14	15	0	8	13	3	2	9	7	5	10	6	1	S7
1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6	
2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2	
3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12	
0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7	S8
1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2	
2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8	
3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11	

[8] Se especula que la estadounidense agencia nacional de seguridad (NSA por sus siglas en inglés) quiso, de manera sigilosa, reservarse la capacidad de *romper* DES cuando así lo estimara necesario, una inquietante posibilidad que fue sospechada desde siempre por grupos activistas [32-33].

Fig. 2 Cajas S del algoritmo de cifrado DES

La figura 3 muestra las diversas transformaciones que un bloque de texto en claro de 64 bits sufre en la iteración principal del algoritmo DES. Se puede apreciar en el extremo derecho inferior de dicha figura que las 8 cajas S se encargan de procesar en paralelo 48 bits de entrada produciendo una salida de 32 bits. En una implementación en hardware, cada una de las 8 cajas S de DES utiliza un espacio de memoria de 64 × 4 = 256 bits, por lo que se necesitan 2K bits de memoria para almacenar las 8 cajas S a la manera descrita en la figura 2.

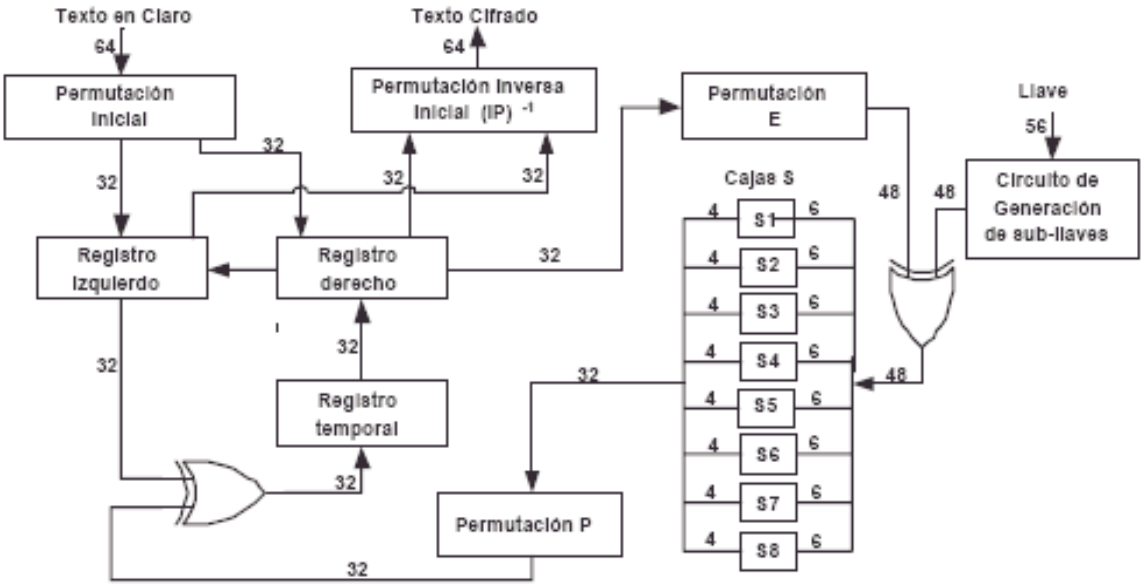

Fig 3. Bloques Principales del Algoritmo DES

Como se mencionó en la Introducción, para que las cajas *S* puedan ser consideradas útiles en aplicaciones criptográficas, deben de cumplir ciertas propiedades tales como una alta no linealidad, un alto grado algebraico, un alto orden de inmunidad de correlación y una baja autocorrelación, entre otras. Para poder describir estas propiedades de una manera formal se revisan en la siguiente sección las definiciones matemáticas más relevantes a este problema.

3 Definiciones matemáticas Básicas

Como se mencionó en la Introducción, una caja S de *n* bits × *m* bits es una relación tal que $S : Z_2^n \rightarrow Z_2^m$. Así, una caja *S* puede ser representada como 2^n números de *m* bits, denotados por $r_0, \ldots, r_{2^n-1}$, en donde $S(x) = r_x, 0 \le x \le 2^n$ y donde r_x representa los renglones de la caja *S*. Alternativamente, una caja *S* puede también representarse mediante una matriz binaria *M* de $2^n \times m$ bits, donde la entrada *i, j* es el bit *j* del renglón *i*-ésimo.

En la práctica, resulta suficiente estudiar cajas S de *n* variables de entrada con un solo bit de salida,[9] a las que llamaremos por simplicidad funciones booleanas de *n* variables, y las cuales serán el objeto de estudio en el resto de esta sección.

Una función booleana de *n* variables, $f(x): Z_2^n \rightarrow Z_2$, es entonces una relación de *n* entradas binarias a una sola salida binaria. Llamaremos B_n al conjunto de las 2^{2^n} funciones booleanas de *n* variables. La representación básica de una función booleana es su *tabla de verdad*, la cual consiste en una cadena binaria de longitud 2^n, tal que:

$$f = [f(0,\ldots 0,0), f(0,\ldots 0,1), \ldots, f(1,\ldots 1,1)] \quad (1)$$

Es importante remarcar que por simplicidad, en muchas ocasiones la cadena binaria en (1) se escribe utilizando su correspondiente representación hexadecimal.

El peso de Hamming de una cadena *S* es el número de unos en *S* y se denota como *H*(*S*). Se dice que una función booleana de *n*-variables está balanceada si su tabla de verdad contiene un número igual de ceros y unos, esto es, si acaso $H(f) = 2^{n-1}$. En B_n, existen un total de $\binom{2^n}{2^{n-1}}$ funciones balanceadas.

Como se mencionó arriba, las 2^n salidas de una tabla de verdad tradicional están definidas sobre Z_2, esto es sus salidas pueden tomar los valores {0, 1}. Sin embargo, notando que el grupo {0, 1, ⊕} es isomórfico a {1,−1, ∗}, resulta útil considerar funciones booleanas con símbolos de salida {1, -1}. Llamaremos a esa representación la tabla de verdad polar de la función booleana $\hat{f}(x)$.

Por razones históricas, la representación polar se prefiere para la mayoría de los cálculos en funciones booleanas [5,9,22-23]. Tal representación puede ser obtenida a partir de la tabla de verdad de *f* usando la sencilla fórmula $\hat{f}(x) = (-1)^{f(x)} = 1 - 2f(x)$.

La llamada *forma normal algebraica* es una tercera alternativa para representar una función booleana de *n* variables, la cual consiste en expresar la función booleana como un polinomio multi-variable a través de la suma XOR[10] mínima de productos AND, es decir, dado *S* = {1, 2, . . . , *n*},

$$f(x_1,\ldots,x_n) = a_0 \oplus a_1 x_1 \ldots \oplus a_n x_n \oplus a_{1,2} x_1 x_2 \ldots \oplus a_{n-1,n} x_{n-1} x_n \ldots \oplus a_{1,2,\ldots,n} x_1 x_2 \ldots x_n = \bigoplus_{I \subseteq S} a_I \prod_{i \in I} x_i \quad (2)$$

[9] Puesto que existen métodos eficientes que combinan *m* funciones booleanas de *n* bits para que juntas conformen una caja S de m-bits de salida.

[10] La operación Or-Exclusiva (XOR, ⊕) consiste en la adición módulo 2 de dos variables de un bit cada una.

Se dice que una función booleana L_ω es *lineal* si puede ser definida como la suma XOR de productos AND de un subconjunto de variables de entrada y los coeficientes de ω así que:

$$L_\omega(x) = \omega_1 x_1 \oplus \omega_2 x_2 \oplus \ldots \oplus \omega_n x_n \tag{3}$$

donde $n, \omega \in Z_2^n$. El conjunto de *funciones afines* está conformado por el conjunto de funciones lineales y sus complementos $A_{\omega,c}(x) = L_\omega(x) \oplus c$ con $c \in \{0,1\}$.

La *distancia de Hamming* entre dos funciones $f \in B_n$ y $g \in B_n$ está definida como el número de posiciones en la tabla de verdad en donde las funciones difieren y pueden ser expresadas como el peso de Hamming de la suma XOR de las dos funciones, esto es, *dist*(*f, g*) = *H*(*f*⊕ *g*). Entonces, la *correlación* entre las funciones *f* y *g* está dada por:

$$c(f,g) = 1 - \frac{dist(f,g)}{2^n - 1} \tag{4}$$

Sean $x = (x_1, \ldots x_n)$ y $\omega = (\omega_1, \ldots \omega_n)$ dos vectores binarios de *n* bits, cuyo producto punto está definido como, $x \cdot \omega = x_1\omega_1 \oplus \ldots xn\omega_n$, y sea *f* una función booleana de *n* variables. Entonces, la transformada de Walsh-Hadamard de una función *f* es la función *F* definida en el dominio de la frecuencia ω como:

$$F(\omega) = \sum_{x \in Z_2^n} (-1)^{\hat{f}(x) \oplus x \cdot \omega} \tag{5}$$

con $\omega \in Z_2^n$.

A partir de (5) se colige que una función booleana *f* de *n* variables es balanceada si y sólo si *F*(ω) = 0.[11] El vector $\{F(0), \ldots, F(2^n - 1)\}$ es el espectro de Walsh-Hadamard de la función *f*, y se denota al valor absoluto del máximo coeficiente del espectro como:

$$WH_{\max}(f) = \max_{w \in Z_2^n} |F(\omega)| \tag{6}$$

Correspondientemente, la transformada inversa de Walsh-Hadamard de una función *F*, formulada a través de la representación polar de una cierta función $\hat{f}$ está dada por:

---

[11] Dado que: $F(\omega)_{\omega=0} = \sum_{x \in Z_2^n} (-1)^{\hat{f}(x) \oplus x \cdot \omega} = \sum_{x \in Z_2^n} (-1)^{\hat{f}(x) \oplus x \cdot 0} = \sum_{x \in Z_2^n} (-1)^{\hat{f}(x)}$, y es fácil ver que si efectivamente *f* es una función balanceada, la anterior sumatoria se hace cero.

$$\hat{f}(x) = 2^{-n} \sum_{\omega} F(\omega)(-1)^{\omega \cdot x} \tag{7}$$

Nótese que un cálculo directo del espectro de Walsh-Hadamard completo utilizando (5) implica una complejidad de N^2 pasos, con $N=2^n$. Sin embargo tal y como ocurre con la transformada rápida de Fourier, es posible definir un procedimiento rápido para el cálculo de la transformada de Walsh-Hadamard que puede ser computado con únicamente $N\log(N)$ pasos. Para lograr esa aceleración, la Transformada Rápida de Walsh-Hadamard (TRWH) utiliza el concepto de *diagrama de mariposa*. Un diagrama de mariposa de tamaño 2 (el tamaño más pequeño), toma dos bits de entrada, (x_0, x_1), y produce dos bits de salida (y_0, y_1), de la siguiente manera:

$$\begin{aligned} y_0 &= x_0 + x_1 \\ y_1 &= x_0 - x_1 \end{aligned} \tag{8}$$

En general, la TRWH divide recursivamente el cálculo de un vector de tamaño $n=rm$, en r transformaciones más pequeñas de tamaño m, donde r es la base de la transformación. Estas r transformaciones pequeñas son combinadas utilizando diagramas de mariposa de tamaño r, las cuales a su vez, son TRWH de tamaño r.[12]

La no linealidad de una función booleana *f* está definida como el número de bits que deben cambiarse en la tabla de verdad en esa función booleana para obtener la función afín más cercana (en el sentido de la distancia de Hamming) [1,3,11]. Se demuestra que la no linealidad de una función *f* puede calcularse directamente a partir de $|WH_{\max}(f)|$, el máximo valor absoluto en el espectro de Walsh-Hadamard a través de la siguiente expresión:

$$N(f) = \frac{1}{2}\left(2^n - |WH_{\max}(f)|\right) \tag{9}$$

La aplicación del celebrado *Teorema de Parseval* al dominio de funciones booleanas establece que la suma de los cuadrados de cada uno de los coeficientes del espectro de Walsh-Hadamard es siempre igual a 2^{2n}, esto es:

$$\sum_{\omega \in Z_2^n} \left(\hat{F}(\omega)\right)^2 = 2^{2n} \tag{10}$$

[12] véase el ejemplo 2 y la figura 4 en la siguiente subsección.

Una consecuencia inmediata de este resultado es que $WH_{\max}(f) \geq 2^{n/2}$. Basado en esta observación se definen las funciones curvas,[13] las cuales son funciones booleanas de *n* variables de entrada tales que,

$$\left| F(\omega) = 2^{\frac{n}{2}} \right|, \forall \omega \in 0,\ldots,2^n - 1 \qquad (11)$$

Las funciones booleanas curvas sólo están definidas para un número de variables de entrada par y siempre resultan ser funciones booleanas desbalanceadas [14] de máxima no linealidad (lo cual se puede demostrar a partir de la definición (7) y el hecho que $|WH_{\max}(f)|$ toma su valor mínimo teórico: $2^{n/2}$).

Se define la *función de autocorrelación* $r_{\hat{f}}(s)$, de una función booleana *f* a partir de su representación polar como:

$$r_{\hat{f}}(s) = \sum_x \hat{f}(x)\hat{f}(x \oplus s) \qquad (12)$$

con $s \in Z_2^n$.

Finalmente mencionaremos el teorema de Titsworth [3], que establece que *F* evaluado en el dominio de la frecuencia corresponde al espectro de Walsh-Hadamard de una función booleana si y sólo si:

$$\sum_{\omega \in Z_2^n} F(\omega)F(\omega \oplus s) = \begin{cases} 2^{2n} & \text{si } s = 0 \\ 0 & \text{en otro caso} \end{cases} \qquad (13)$$

con $s \in Z_2^n$.

3.1 Ejemplos.

En esta subsección se ilustran las definiciones dadas en la subsección precedente con varios ejemplos.

Tabla 1. Número de Funciones Balanceadas en B_n.

N	***B_n***	**Func. Balanceadas**	**Porcentaje**
1	2^2=4	$\binom{2^1}{2^0} = \binom{2}{1} = 2$	50.0%

[13] Funciones *Bent* en inglés.

[14] puesto que $F(0) \neq 0$.

2	2^4=16	$\binom{2^2}{2^1} = \binom{4}{2} = 6$	37.5%
3	2^8=256	$\binom{2^3}{2^2} = \binom{8}{4} = 70$	27.3%
4	2^{16}=64 Kilos	$\binom{2^4}{2^3} = \binom{16}{8} = 12870$	19.6%
5	2^{32}=4 Gigas	$\binom{2^5}{2^4} = \binom{32}{16} \approx 601Meg$	14.0%
6	2^{64}=16 Exas	$\binom{2^6}{2^5} = \binom{64}{32} \approx 1.6exas$	9.9%

Ejemplo 1: Número de funciones balanceadas.
La tabla 1 muestra el número de funciones booleanas balanceadas y respectivo porcentaje en el total de B_n funciones booleanas para n=1,2,…,6. Como puede apreciarse, hay una relativa abundancia de funciones balanceadas en el enorme universo de funciones booleanas B_n, que sin embargo decae rápidamente conforme n se incrementa.

Ejemplo 2: Función booleana de tres entradas, balanceada y de máxima no linealidad.
Consideremos la función booleana f de tres entradas descrita algebraicamente como:

$$f(x) = x_2x_1 + x_3x_1 + x_3x_2 \tag{14}$$

Tabla 2. Ejemplo de una función booleana de tres entradas

x_3	x_2	x_1	$f(x)$	$\hat{f}(x)$	$F(\omega)$
0	0	0	0	1	0
0	0	1	0	1	4
0	1	0	0	1	4
0	1	1	1	-1	0
1	0	0	0	1	4
1	0	1	1	-1	0
1	1	0	1	-1	0
1	1	1	1	-1	-4

La tabla 2 presenta la tabla de verdad de la función f en su versión tradicional y polar, junto con su respectivo espectro de Walsh-Hadamard. El espectro de f puede ser hallado a través de la ecuación (5), o aún mejor, utilizando la Transformada Rápida de Walsh-Hadamard mostrada esquemáticamente en la figura 4.

Como se explicó en la subsección precedente, el espectro $F(\omega)$ de una función booleana *f* brinda una rica información sobre las características de dicha función. Por ejemplo, el espectro de Walsh-Hadamard de la tabla 2 nos indica que *f* es una función balanceada, puesto que el primer coeficiente del espectro, $F(0)$, tiene valor cero. Asimismo, se determina a partir de (9) que *f* tiene no linealidad 2 puesto que[15]:

$$N(f) = \frac{1}{2}\left(2^n - |WH_{\max}(f)|\right) = \frac{1}{2}\left(2^3 - 4\right) = 2$$

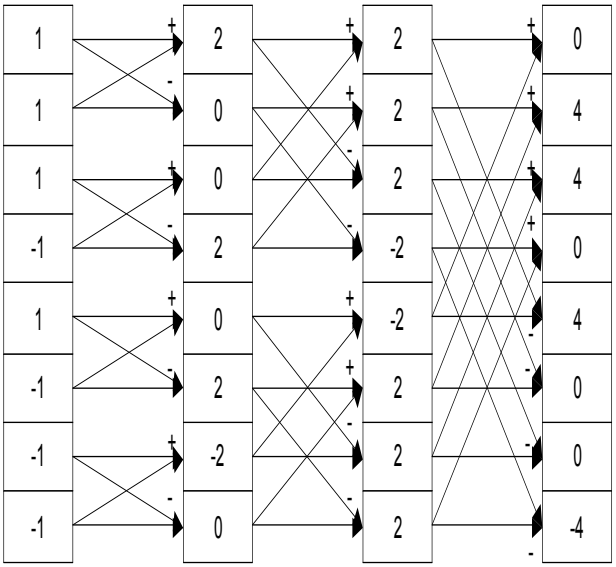

Figura 4. Transformada rápida de Walsh-Hadamard para la función

$$f(x) = x_2x_1 + x_3x_1 + x_3x_2 .$$

Ejemplo 3: Función curva de 4 entradas.

Considere la función booleana *f* de cuatro entradas descrita algebraicamente como:

$$f(x) = \bar{x}_4\bar{x}_3x_2x_1 + x_4x_3\left(\bar{x}_2 + \bar{x}_1\right) + \bar{x}_2\bar{x}_1\left(x_3 + x_4\right) \qquad (15)$$

La tabla 3 presenta la tabla de verdad de la función *f* en su versión tradicional y polar, junto con su respectivo espectro de Walsh-Hadamard. Del espectro de Walsh-Hadamard de la Tabla 3 podemos deducir que la función booleana *f* en (15), satisface la definición de función curva de la ecuación (11). Nótese que como se afirmó en la subsección precedente, la tabla de verdad de *f* corresponde a la de una función booleana no balanceada con máxima no linealidad 6, puesto que:

[15] Un valor de no linealidad 2 corresponde a la máxima no linealidad alcanzable por una función booleana de tres entradas.

$$N(f) = \frac{1}{2}\left(2^n - |WH_{\max}(f)|\right) = \frac{1}{2}\left(2^4 - 4\right) = 6$$

Tabla 3. Tabla de verdad de una función curva de 4 entradas

x_4	x_3	x_2	x_1	$f(x)$	$\hat{f}(x)$	$F(\omega)$
0	0	0	0	0	1	4
0	0	0	1	0	1	-4
0	0	1	0	0	1	-4
0	0	1	1	1	-1	-4
0	1	0	0	1	-1	4
0	1	0	1	0	1	4
0	1	1	0	0	1	4
0	1	1	1	0	1	-4
1	0	0	0	1	-1	-4
1	0	0	1	0	1	4
1	0	1	0	0	1	4
1	0	1	1	0	1	4
1	1	0	0	1	-1	-4
1	1	0	1	1	-1	4
1	1	1	0	1	-1	4
1	1	1	1	0	1	4

3.2 Propiedades criptográficas deseables en funciones booleanas

A continuación se enlistan varios de los principales criterios utilizados en la práctica profesional para diseñar cajas S con buenas propiedades criptográficas:

1. Balance: Esta propiedad es muy deseable para evitar ataques cripto-diferenciales tales como los introducidos por A. Shamir contra el algoritmo DES [27,31-33].
2. Alta no linealidad: Esta propiedad reduce el efecto de los ataques por criptoanálisis lineal. Como se discutió antes, la no linealidad de una función booleana puede ser calculada directamente de la transformada de Walsh-Hadamard (a través de la ecuación (5)).
3. Autocorrelación: Este valor es proporcional al desbalance de todas las derivadas de primer orden de la función booleana. Valores pequeños son considerados como buenos mientras que un valor grande es considerado un símbolo de debilidad. Las funciones curvas, estudiadas en la subsección precedente, gozan de una autocorrelación mínima, por lo que optimizan esta propiedad.
4. Indicador absoluto de una función booleana denotado por *M*(*f*) está dado por $|r_{max}|$ el máximo valor absoluto en $r_{\hat{f}}(s)$ (véase (12)). Se considera que una función booleana con un *M*(*f*) pequeño es criptográficamente deseable.

Nuevamente, las funciones curvas tienen una autocorrelación óptima pues su indicador absoluto es cero [1,3,11].

5. Efecto avalancha: Está relacionado con la autocorrelación y se define con respecto a un bit específico de entrada tal que al complementarlo resulta en un cambio en el bit de salida con una probabilidad de 1/2. El criterio de avalancha estricto (SAC por sus siglas en inglés), [16] requiere los efectos avalancha de todos los bits de entrada. Se dice que una función booleana satisface el criterio de avalancha estricto si al complementar un solo bit de entrada resulta en un cambio en un bit de salida con una probabilidad de 1/2. Puede demostrarse fácilmente que una función booleana *f* con función de autocorrelación $r_{\hat{f}}(s)$, satisface el criterio de avalancha estricto si y sólo si $r_{\hat{f}}(s)=0$ para toda *s* con peso de Hamming $H(s)=1$ [13-14].

6. Grado algebraico: El grado algebraico de una función *f*, denotado como *deg*(*f*), es el número de entradas más grande que aparece en cualquier producto de la forma normal algebraica. Esto es $x_1 \oplus x_2$ tiene grado 1 (es decir, es lineal) mientras que $x_1 \oplus x_1 x_2 x_3$ tiene grado 3 [3,16-17].

7. Orden de **Inmunidad de Correlación:** Una función *f* tiene un orden de inmunidad de correlación *m* si y sólo si [9]:

$$\hat{F}(\omega)=0; 1 \le H(\omega) \le m$$

8. **Resistencia**: Una función *f* que tiene inmunidad de correlación de orden *m*, es *resistente* si y sólo si también es balanceada [9]:

$$\hat{F}(\omega)=0; 0 \le H(\omega) \le m$$

3.3 Discusión de compromisos y conflictos en las propiedades de las cajas *S*

De manera ingenua, uno podría plantearse buscar funciones booleanas que reúnan todas las propiedades criptográficas descritas en la subsección anterior. Así, podría ensayarse el vano intento de hallar funciones booleanas balanceadas, con máxima no linealidad, alto grado algebraico, alto orden de inmunidad de correlación y baja autocorrelación.

Sin embargo es *imposible* que ninguna función booleana pueda satisfacer al mismo tiempo todos esos criterios.

Quizás el ejemplo más socorrido para ilustrar esa realidad, son las funciones curvas, las cuales por definición (véase ecuación (9)), son máximamente no lineales *pero* desbalanceadas. Si desistimos de las funciones curvas y nos concentramos en funciones balanceadas (esto es $F(0)=0$), entonces, y como consecuencia del teorema de Parseval de la ecuación (10), algún otro coeficiente del espectro deberá necesariamente compensar ese faltante teniendo una

[16] strict avalanche criterion (SAC).

magnitud mayor que $2^{n/2}$, lo cual reducirá la no linealidad de esa función. Otro conflicto más, ocurre al intentar maximizar el orden de inmunidad, lo cual sólo puede llevarse a cabo en detrimento de la no linealidad [3]. Se conocen funciones booleanas curvas que exhiben máxima no linealidad y sin embargo tienen bajísimos grados algebraicos. Por otro lado es posible hallar funciones con baja no linealidad pero con alto grado algebraico [1].

Debido a los conflictos existentes en las propiedades deseables para una función booleana, es necesario establecer compromisos. De esa manera, se ha ido adoptando más y más en la literatura especializada [1,3-5,16-17], el perfil de una función booleana *f* balanceada, dado por la cuádrupla (*n*, *m*, *d*, *nl*), donde *n* denota el número de variables de entrada, *m* el orden de inmunidad, *d* el orden algebraico y *nl* la no linealidad de la función *f*.[17]

4 Búsqueda de funciones booleanas por métodos heurísticos.

Para poder realizar una búsqueda basada en técnicas heurísticas evolutivas, es indispensable contar con una *representación* que permita codificar las soluciones potenciales del problema en una población inicial de *individuos*. En seguida es necesario definir operadores que, generación tras generación, alteren las características de los individuos así que en cada generación, a los individuos con mejores características se les dé una mayor oportunidad de reproducirse, mejorando así sus oportunidades de sobrevivir.

Los operadores que típicamente se utilizan en este tipo de heurísticas son la *mutación*, la *selección* y la *cruza*. En particular, para poder implementar el mecanismo de *selección*, resulta indispensable contar con una función de *aptitud* que permita medir el desempeño de la solución representada en cada uno de los individuos de la población bajo análisis [13-15,19].

En el caso de una búsqueda heurística de funciones booleanas, el problema de diseño más importante es decidir cuál será la función de aptitud que se utilizará para medir las bondades criptográficas de los individuos (funciones booleanas) que constituyen la población de cada generación [4-5,13]. En los últimos años, se han propuesto diversas funciones de aptitud, de las cuales, en el resto de esta sección se discutirán las siguientes tres: funciones de aptitud tradicionales, basadas en inversión de espectro y basadas en búsquedas en espacios restringidos.

4.1 Funciones de aptitud tradicionales

La abrumadora mayoría de los trabajos reportados antes del año 2000 [16-17] enfilaban todos los cañones hacia la búsqueda de funciones altamente no lineales, sin reparar, ni poco ni mucho, en otras propiedades criptográficas. Así se propuso

[17] A propósito de los conflictos entre las propiedades, desde hace mucho tiempo se sabe que para funciones booleanas balanceadas se cumple siempre que $m+d \leq n-1$ [3].

la medida de no linealidad de un individuo dado (esto es, alguna función booleana *f*) como su medida de aptitud:

$$Aptitud(f) = \frac{1}{2}\left(2^n - |WH_{\max}(f)|\right)$$

o visto como un problema de minimización, la función de aptitud se planteó también como:

$$\cos to(f) = |WH_{\max}(f)| = \max_{\omega} |\hat{F}(\omega)|$$

De manera similar, en los raros casos en que se fijó la baja autocorrelación como función objetivo, se utilizó una función de costo dada por:

$$\cos to(f) = AC(f) = \max_{s \neq 0} \left| \sum_{x} \hat{f}(x)\hat{f}(x \oplus s) \right| = \max_{s \neq 0} |\hat{r}(s)| \text{ con } s \in Z_2^n$$

4.2 Funciones de aptitud basadas en inversión de espectro

Como se ha mencionado anteriormente, el espectro de Walsh-Hadamard de una función booleana *f*, permite evaluar rápidamente si los diferentes criterios de diseño han sido alcanzados o no. Es por ello que en años recientes se propuso desarrollar motores de búsqueda basados en las características que el espectro debiera tener en una buena función booleana. Esta estrategia realiza entonces una suerte de "ingeniería en reversa", en el sentido que la búsqueda se enfoca primero en diseñar el espectro con las características que se desean, para después, a través de la aplicación de la transformada inversa de Walsh-Hadamard, hallar la función booleana a la que le corresponde tal espectro.

En concreto, supongamos que se cuenta con el espectro de Walsh-Hadamard $F(\omega)=\{F(0), F(1),\ldots, F(2^n-1)\}$, de una función con perfil criptográfico (*n*, *m*, *d*, *nl*), esto es, el espectro correspondiente al de una función booleana balanceada de *n* variables de entrada con no linealidad *nl*, grado algebraico *d* y orden de inmunidad *m*.[18] Consideremos entonces el conjunto de espectros *P* dado por todas las posibles permutaciones del espectro original $F(\omega)$ tales que $P(\omega) = 0; 0 \leq H(\omega) \leq m.$ Entonces cualquier espectro *G* incluido en el conjunto *P* disfruta de los mismos valores y propiedades criptográficos con los que cuenta el espectro original *F*.

Desafortunadamente, esta estrategia no garantiza que un espectro permutado *G* en el conjunto *P* corresponderá a alguna función booleana legítima. En efecto, cuando se aplica la transformada inversa a *G*:

[18] Note que el método de inversión espectral supone que en un principio *no* se conoce el valor de tal función booleana.

$$\hat{p}(x) = 2^{-n} \sum_{\omega} G(\omega)(-1)^{\omega \cdot x} ,$$

la función resultante $\hat{p}$ tendrá, en general, coeficientes reales, en vez de tener todos sus coeficientes en {1,-1}, como corresponde a la representación polar de toda verdadera función booleana. Debido a ello, en [3] se propuso utilizar una asignación heurística para evaluar la *desviación* del espectro *G* a un espectro legítimo. Se define la función booleana $\hat{b}$ así que:[19]

$$\hat{b}(x) = \begin{cases} +1 & \text{si } \hat{p}(x) > 0 \\ -1 & \text{si } \hat{p}(x) < 0 \\ +1 \text{ o } -1 & \text{si } \hat{p}(x) = 0 \end{cases}$$

Con lo que de manera natural, surge como función de costo la ecuación que mide cuan lejos quedó la permutación espectral *G* de una verdadera función booleana, es decir [3]:

$$Costo(G) = \sum_{x=0}^{2^n-1} \left(\hat{p}(x) - \hat{b}(x)\right)^2 \qquad (16)$$

La función de costo en (16) tiene el defecto de hacer las evaluaciones en el dominio booleano abandonando el dominio de la frecuencia ω donde está definida la permutación espectral *G*. Es por ello que en [3] se definió una función de costo en el dominio de la frecuencia, fundamentada en el teorema de Titsworth enunciado en la sección precedente (véase la ecuación (13)):

$$\cos to(G) = \sum_{s} \left(\left| \sum_{\omega \in Z_2^n} G(\omega) G(\omega \oplus s) \right| \right) - 2^{2n} = 0 \qquad (17)$$

con $s \in Z_2^n$.

Utilizando la función de costo (17), se hicieron en [6] experimentos para hallar funciones booleanas con perfil criptográfico (7, 0, 6, 56), correspondiente a una función booleana de siete variables de entrada, balanceada, con orden de inmunidad 0, grado algebraico 6 y no linealidad 56.[20] Se utilizó un algoritmo genético simple con porcentaje de mutación en el rango de [1/100, 1/128] y porcentaje de cruza 0.7, obteniéndose resultados favorables en todas las corridas. Por ejemplo, una de las funciones halladas en [6] que satisface el perfil buscado es:

---

[19] En caso que $\hat{p}(x) = 0$ el valor correspondiente de $\hat{b}(x)$ se escoge aleatoriamente.

[20] Máximo valor de no linealidad para funciones de siete entradas.

6A65 33AC D05E C840 07BA 4597 BD81 BE7B

Asimismo, en [6] se encontró la siguiente función booleana que satisface el perfil criptográfico (7,2,4,56):

039CE9F8D781253EA6555E4A22E8F11F

4.3 Búsquedas en espacios restringidos

A pesar que el método de búsqueda por inversión espectral ha dado en los últimos tres años excelentes resultados [3-5], sigue estando limitado por el hecho que el espacio de búsqueda B_n tiene un crecimiento doblemente exponencial con n. Por ello, en trabajos más recientes [16-17,28] se ha utilizado un refinamiento del método de búsqueda por inversión espectral restringiendo el espacio de búsqueda al asociado a las Funciones booleanas de Rotación Simétrica (FBRS).[21]

Las FBRS son funciones booleanas que mantienen el mismo valor para todas las rotaciones cíclicas de sus entradas. Por ejemplo, para una FBRS de 5 variables de entrada se definen las siguientes ocho clases u *órbitas* [3,16-17,25,34-35] de rotación:

Órbita 1: $f(00000)$;
Órbita 2: $f(00001)=f(00010)=f(00100)=f(01000)=f(10000)$;
Órbita 3: $f(00011)=f(00110)=f(01100)=f(11000)=f(10001)$;
Órbita 4: $f(00101)=f(01010)=f(10100)=f(01001)=f(10010)$;
Órbita 5: $f(01011)=f(10110)=f(01101)=f(11010)=f(10101)$;
Órbita 6: $f(00111)=f(01110)=f(11100)=f(11001)=f(10011)$;
Órbita 7: $f(01111)=f(11110)=f(11101)=f(11011)=f(10111)$;
Órbita 8: $f(11111)$;

La tabla 5 muestra el número de funciones booleanas de rotación simétrica en el universo total de B_n funciones booleanas para n=2,3,…,9.

Tabla 5. Número de Funciones booleanas de rotación simétrica en B_n.

	2	**3**	**4**	**5**	**6**	**7**	**8**	**9**
B_n	2^4	2^8	2^{16}	2^{32}	2^{64}	2^{128}	2^{256}	2^{512}
FBRS	2^3	2^4	2^6	2^8	2^{14}	$\approx 2^{19}$	$\approx 2^{32}$	$\approx 2^{57}$

Una propiedad muy útil de las FBRS es que el espectro de Walsh-Hadamard toma el mismo valor para todos los elementos que pertenezcan a la misma órbita [28]. Además, a pesar que el conjunto de FBRS representa sólo una pequeña fracción de todas las posibles funciones booleanas, el conjunto de funciones FBRS tiende a tener una muy rica no linealidad [16-17].

[21] En inglés: Rotation Symmetric Boolean Functions.

Utilizando una búsqueda heurística del máximo gradiente restringida al subespacio de las funciones FBRS y empleando la técnica de inversión espectral, se reportó en diciembre de 2006 la siguiente función booleana de nueve variables con no linealidad 241 [17]:

977F 3FFA 0EFA AEC9 55F8 FACD CCA9 A083 7666 EBC0 FA88 E0B3 F4E0 8983 C845 915E 7F7C 2C29 FCCB A101 EA98 C085 E811 8B5E FE21 E911 8483 851E E195 2136 9716 76E9

En importante señalar que desde 1974 se había conjeturado que tal función podría existir, pero tuvieron que pasar más de 30 años para poder confirmar esa afirmación con evidencia experimental [17].

5. De conjeturas, retos y perspectivas.

La ecuación (11) de la sección 3, define a las funciones curvas, las cuales se caracterizan por alcanzar una no linealidad máxima (con valor de $2^{n-1} - 2^{n/2-1}$), para funciones booleanas de *n* variables con *n* par. Sin embargo, como se ha mencionado, las funciones curvas son siempre desbalanceadas, por lo que cabe preguntarse:

¿Cuál es la máxima no linealidad alcanzable por una función booleana balanceada de *n* variables, con *n* par?

Hasta diciembre del año pasado sólo se conocían ejemplos de funciones booleanas con valores de no linealidad $2^{n-1} - 2^{n/2}$, pues por ejemplo, para *n*=8, se sabía de funciones booleanas balanceadas con no linealidad de $112 = 2^{8-1} - 2^{8/2}$. Sin embargo, en [16] se reportó una función booleana de 10 variables de entrada con no linealidad de $492 = 2^{10-1} - 2^{10/2} + 12$, valor que ya había sido predicho en [30] como el máximo teóricamente posible para funciones balanceadas de ese número de variables.

Por otro lado, en 1972 y en 1980 se demostró que la máxima no linealidad alcanzable con funciones booleanas de 5 y 7 variables es de 12 y 56, respectivamente, por lo que se supo que para $n \leq 7$ impar la máxima no linealidad para funciones de *n* variables es $2^{n-1} - 2^{(n-1)/2}$ y se planteó la conjetura de si acaso ese valor se mantendría para *n* impar $n \geq 9$ [16].

Sin embargo, en 1983, tal conjetura fue refutada al hallarse que existen funciones booleanas de 15 variables con no linealidad de $16276 = 2^{15-1} - 2^{(15-1)/2} + 20$. Este descubrimiento fue utilizado para demostrar que para *n* impar con $n \geq 15$, las funciones booleanas alcanzan una no linealidad de al menos $2^{n-1} - 2^{(n-1)/2} + 20 \cdot 2^{(n-15)/2}$.

Con estos resultados, el valor exacto de la máxima no linealidad alcanzable para *n*=9,11,13, quedó como un problema abierto, puesto que los mejores resultados que se conocían, apuntaban a funciones booleanas con no linealidad de $2^{n-1} - 2^{(n-1)/2}$. Sin embargo, como se mencionó al final de la sección anterior, el año pasado, utilizando una búsqueda exhaustiva en subespacios restringidos FBRS, se reportó en [17] que para *n*=9 existen funciones con no linealidad $241 = 2^{9-1} - 2^{(9-1)/2} + 1 \cdot 2^{(9-9)/2}$.

Por ser 241 un número impar, este hallazgo permite conjeturar que acaso el valor máximo de no linealidad para funciones booleanas con 9 variables deba ser mayor o igual que 242. [22]

A manera de resumen general, en la tabla 6 se listan las cotas superiores teóricas y mejores valores reportados para la no linealidad de funciones booleanas balanceadas.

Con respecto a resultados de funciones booleanas con otras propiedades criptográficas (además de balance y no linealidad), únicamente mencionamos dos resultados interesantes, ambos reportados el año pasado. Utilizando el método de inversión espectral junto con la heurística evolutiva de cúmulo de partículas [18-19] se encontró una familia de funciones booleanas de 9 variables con perfil (9,3,5,240), esto es, con orden de inmunidad 3, orden algebraico 5 y no linealidad 240 [28]. Con este hallazgo se contestó afirmativamente una conjetura sobre la existencia de tales funciones lanzada desde el año 2000 en [30]. Un segundo resultado significativo se reportó en [1], donde se determinó que existen exactamente 36 y 10272 funciones del tipo FBRS con perfiles criptográficos de (7,2,4,56) y (8,1,6,116), respectivamente. [23]

Tabla 6. Cotas superiores teóricas y mejores valores alcanzados para no linealidad en funciones booleanas balanceadas.

	5	6	7	8	9	10	11	12
Cota superior teórica	12	26	56	118	244	492	1000	2014
Mejor caso reportado	12	26	56	116	241	492	992	2010

A manera de conclusión señalamos que en los últimos tres años se han encontrado funciones booleanas con excelentes propiedades criptográficas, las cuales han permitido confirmar/refutar diversas conjeturas planteadas desde hacía

[22] Esta conjetura estaría sustentada en el hecho que los valores de las máximas no linealidades teóricas conocidas hasta ahora, son todos números pares, por lo que se piensa que un valor impar como el obtenido en [17] es de alguna manera antiestético y hasta *contra natura* [29].
[23] Todas esas funciones toman un valor *f*(0)=0, en el dominio booleano.

más de tres décadas. No es exagerado afirmar que este gran avance ha sido posible gracias al círculo virtuoso conformado por la combinación de ingeniosos y refinados resultados combinatorios teóricos junto con el empleo de poderosos motores de búsqueda heurísticos (esencialmente a través de técnicas evolutivas).

Consideramos que las perspectivas con respecto a este ilustre problema combinatorio son muy prometedoras, pues desde hace unos tres años hemos entrado de lleno en una etapa de desarrollo acelerada en la que en períodos muy cortos se anuncian nuevos y espectaculares resultados. No es aventurado predecir que en los próximos años, se encontrarán muchas funciones booleanas con características criptográficas aún mejores que las reportadas hasta ahora gracias al empleo de algoritmos y heurísticas evolutivas cada vez más sofisticados.

Por último, nos gustaría finalizar este manuscrito con una pregunta que consideramos obligada:

¿Se harán algunos de los descubrimientos de nuevas y mejores funciones booleanas en México, y más específicamente en el CINVESTAV-IPN?
Nos atrevemos a conjeturar que sí.

Agradecimientos

El autor de este artículo desea hacer público su agradecimiento a los doctores Nareli Cruz Cortés y Guillermo Morales Luna por sus valiosos comentarios y sugerencias que ayudaron a mejorar el contenido y forma de este artículo.

REFERENCIAS

[1] C. Carlet, D. K. Dalai, K. C. Gupta y S. Maitra. Algebraic Immunity for Cryptographically Significant Boolean Functions: Analysis and Construction. IEEE Transactions on Information Theory 52(7): 3105-3121 2006.

[2] H. Chen y D. G. Feng. An effective genetic algorithm for finding highly nonlinear boolean functions. In *CEC 2004: International Conference on Evolutionary Computation, Portland OR, USA, June 2004*, páginas 2120–2123. IEEE, 2004.

[3] J. A. Clark, J. L. Jacob, S. Maitra, y P. Stnic. Almost boolean functions: The design of boolean functions by spectral inversion. In *Computational Intelligence 20 (3)*, páginas 450–462, 2004.

[4] J. A. Clark, J. L. Jacob, y S. Stepney. The design of S-boxes by simulated annealing. In *CEC 2004: International Conference on Evolutionary Computation, Portland OR, USA, June 2004*, páginas 1533–1537. IEEE, 2004.

[5] J. A. Clark, J. L. Jacob, S. Stepney, S. Maitra, y W. Millan. Evolving boolean functions satisfying multiple criteria. In *Proceedings of the Third International Conference on Cryptology*, páginas 246–259. Springer-Verlag, 2002.

[6] N. Cruz-Cortés, comunicación personal no publicada, Diciembre de 2006.

[7] J. Daemen y V. Rijmen. *The Design of Rijndael*. Springer-Verlag New York, Inc., 2002. ISBN:3540425802.

[8] A. Díaz-Pérez, N. A. Saqib, y F. Rodríguez-Henríquez. Some Guidelines for Implementing Symmetric-Key Cryptosystems on Reconfigurable-Hardware. In IV Jornadas de Computación Reconfigurable y Aplicaciones, páginas 379_387, septiembre 2004.
[9] R. Forré. Methods and instruments for designing s-boxes. *J. of Cryptology*, 2(3):115–130, 1990.
[10] J. Fuller, W. Millan, y E. Dawson. Multiobjective optimization of bijective s-boxes. In *CEC 2004: International Conference on Evolutionary Computation, Portland OR, USA, June 2004*, páginas 1525–1532. IEEE, 2004.
[11] K. C. Gupta y P. Sarkar. Improved construction of nonlinear resilient s-boxes. In *Proceedings of the 8th International Conference on the Theory and Application of Cryptology and Information Security*, páginas 466–483. Springer-Verlag, 2002.
[12] J. C. Hernández-Castro, P. Isasi, y C. Luque del Arco-Calderón. Finding efficient nonlinear functions by means of genetic programming. In *KES 2003*, Seventh International Conference on Knowledge-Based Intelligent Information & Engineering Systems, páginas 1192–1198, 2003.
[13] E. Hernández-Luna. Documento de Propuesta Doctoral, enero de 2005.
[14] E. Hernández-Luna. Criterio de avalancha estricto en funciones booleanas. Reporte técnico, mayo de 2005.
[15] E. Hernández-Luna, C. A. Coello Coello y A. Hernández-Aguirre. On the use of a population-based particle swarm optimizer to design combinational logic circuits. In Ricardo S. Zebulum, David Gwaltney, Gregory Hornby, Didier Keymeulen, Jason Lohn, y Adrian Stoica, editores, *Proceedings of the 2004 NASA/DoD Conference on Evolvable Hardware*, pages 183–190. IEEE Computer Society, June 2004.
[16] S. Kavut, S. Maitra, S. Sarkar y M. D. Yücel. Enumeration of 9-Variable Rotation Symmetric Boolean Functions Having Nonlinearity > 240. Rana Barua, Tanja Lange (Eds.): Progress in Cryptology - INDOCRYPT 2006, 7th International Conference on Cryptology in India, Proceedings. Lecture Notes in Computer Science 4329, páginas 266-279, Springer, 2006.
[17] S. Kavut, S. Maitra y S. Sarkar. There exist Boolean functions on n (odd) variables having nonlinearity > $2^{n-1} - 2^{n-1/2}$ if and only if $n > 7$, Cryptology ePrint Archive, Report 2006/181, Disponible en: http://eprint.iacr.org/, 2006.
[18] J. Kennedy y R. C. Eberhart. Particle Swarm Optimization. In *Proceedings of the 1995 IEEE International Conference on Neural Networks*, pages 1942–1948, Piscataway, New Jersey, 1995. IEEE Service Center.
[19] J. Kennedy y R. C. Eberhart. *Swarm Intelligence*. Morgan Kaufmann Publishers, San Francisco, California, 2001.
[20] E. López-Trejo, F. Rodríguez-Henríquez y A. Díaz- Pérez. An Efficient FPGA implementation of CCM Using AES. The 8th International Conference on Information Security and Cryptology (ICISC'05), Lecture Notes in Computer Science, Vol. 3935, páginas.208-215, 2005.
[21] A. J. Menezes, S. A. Vanstone, y P. C. Van Oorschot. *Handbook of Applied Cryptography*. CRC Press, Inc., 1997. ISBN: 0-8493-8523-7.
[22] W. Millan, J. Fuller, y E. Dawson. Evolutionary generation of bent functions for cryptography. In Ruhul Sarker, Robert Reynolds, Hussein Abbass, Kay Chen Tan,

Bob McKay, Daryl Essam y Tom Gedeon, editores, *CEC*, páginas 149–158, Canberra, 8-12, IEEE Computer Society Press, December 2003.
[23] W. Millan, J. Fuller, y E. Dawson. New concepts in evolutionary search for boolean functions in cryptology. In *Computational Intelligence 20 (3)*, páginas 463–474, 2004.
[24] NIST. Announcing the Advanced Encryption Standard (AES). Federal Information Standards Publication, Nov. 2001. Disponible en: http://csrc.nist.gov/CryptoToolkit/aes/index.html.
[25] J. Pieprzyk y C. X. QU. Fast hashing and rotation-symmetric functions. Journal of Universal Computer Science, **5**(1):20–31. 1999.
[26] F. Rodríguez-Henríquez, N. A. Saqib y A. Díaz-Pérez. 4.2 Gbit/s Single-Chip FPGA Implementation of AES Algorithm. Electronic Letters, 39(15):1115-1116, July 2003.
[27] F. Rodríguez-Henríquez, N.A. Saqib, A. Díaz Pérez y Ç. K. Koç. "Cryptographic Algorithms on Reconfigurable Hardware", Springer First Edition, Noviembre 2006, 362 páginas. ISBN: 0387338837.
[28] Z. Saber, M. F. Uddin, A. Youssef: On the existence of (9, 3, 5, 240) resilient functions. IEEE Transactions on Information Theory 52(5): 2269-2270 (2006).
[29] P. Sarkar, comunicación personal (a través de D. Chakraborty) no publicada, Diciembre de 2006.
[30] P. Sarkar y S. Maitra. Nonlinearity bounds and construction of resilient Boolean functions. In Advances in Cryptology - Crypto 2000, páginas 515–532, Berlin, 2000. Springer-Verlag. Lecture Notes in Computer Science Volume 1880.
[31] B. Schneier. *Applied Cryptography. Protocols, Algorithms, and Source Code in C*. John Wiley & Sons, segunda edición, 1996. ISBN 0-471-11709-9.
[32] S. Singh. The Code Book. Fourth Estate; segunda edición (June 8, 2000). ISBN 978-1857028898. Disponible electrónicamente en:
http://www.simonsingh.net/.
[33] S. Singh. Los Códigos Secretos. Debate, 2000. ISBN: 84-8306-278-X.
[34] P. Stănică, S. Maitra y J. Clark. Results on Rotation Symmetric Bent and Correlation Immune Boolean Functions. Fast Software Encryption Workshop (FSE 2004), Nueva Delhi, INDIA, LNCS 3017, Springer Verlag, 161–177, 2004.
[35] P. Stănică y S. Maitra. Rotation symmetric Boolean functions—Count and cryptographic properties. *In* R. C. Bose Centenary Symposium on Discrete Mathematics and Applications, December 2002, Indian Statistical Institute, Calcutta and Technical Report of Cryptology Research Group, Reporte técnico CRG/2002/10, Noviembre, 2002. Disponible en:
http://www.isical.ac.in/~crg/tech_reports.html.