SIMETRÍAS EN FUNCIONES BOOLEANAS INMUNES A LA CORRELACIÓN.

Tesis de Licenciatura en Computación.
6 de octubre de 2017

Francisco Castro

Tutor: Alfredo Viola

Facultad de Ingeniería
Universidad de la República

Índice

1. Introducción y aplicaciones.

En este informe se presenta una serie de teoremas con sus demostraciones sobre funciones booleanas de n variables inmunes a la correlación de orden k, (abreviado k-CI de aquí en más). A lo largo del documento se explican los conceptos necesarios, tratando de que la lectura lineal del mismo sea suficiente para su comprensión. Debido a lo confuso que pueden resultar algunas de las definiciones, se hace énfasis aportando ejemplos.

Antes que nada, hay muchas maneras de definir funciones booleanas, algunas de ellas combinatorias, otras basadas en álgebra lineal, en cuerpos finitos, transformadas de Fourier, u otras. A los efectos de este documento una función booleana será vista como una función que toma como parámetro un vector de n booleanos y devuelve un booleano.

Por un relevamiento de funciones booleanas y sus uso en criptografía y teoría de códigos se puede leer la referencia internacional en el tema [6, 7] de Claude Carlet.

1.1. Motivación en generadores pseudo aleatorios

Debido a que la generación de números aleatorios es muy importante pero muy costoso, se utilizan generadores pseudo aleatorios.[9] En particular para aplicaciones criptográficas es importante que además de pasar varios tests aleatorios[9] el resultado del próximo bit conociendo los anteriores, sea "impredecible".

Por "impredecible" queremos decir de una manera informal que para un atacante que conozca la estructura del generador pseudo aleatorio pero no la semilla inicial (clave secreta), la manera "más eficiente" (en el sentido de algoritmo polinomial) sea equivalente a tirar una moneda con probabilidad 1/2.

En otras palabras, si bien los generadores pseudo aleatorios son *determinísticos* (el próximo bit está determinado conociendo los bits anteriores) deberán ser diseñados de tal manera que un atacante conociendo el generador y la secuencia completa generada hasta el momento, pero no la clave privada (semilla inicial), no tenga maneras eficientes (polinomiales) en la práctica de predecir el próximo bit con probabilidad de un medio.[14]

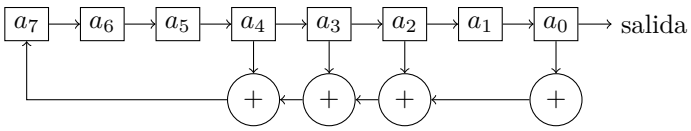
Por referencias a las aplicaciones a la criptografía de los generadores pseudo aleatorios, ver capítulo 11 del libro CryptoSchool [14].

Tradicionalmente se utilizó LFSR (*linear feedback shift registers*) como generadores pseudo aleatorios. Los mismos se componen de una memoria FIFO de n bits ($a_{n-1} \dots a_0$), en donde para generar cada nuevo bit se hace un XOR de un subconjunto fijo de los bits, insertándose el resultado por la entrada y devolviéndose el bit desplazado.

Como ventaja, las implementaciones de hardware permiten generar un bit por cada ciclo de reloj con costos insignificantes, siendo suficientes n flip flops más unas pocas compuertas XOR para poder obtener un generador de periodo muy largo (de hasta periodo $p = 2^n - 1$, repitiendose los mismos p bits una y otra vez).

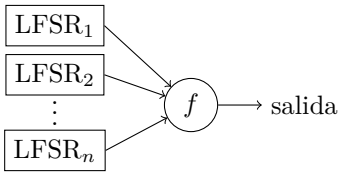
Ejemplo 1.1. Con tan solo 8 bits de RAM y 4 compuertas XOR se puede

fabricar el siguiente generador LFSR de período 255:



El problema que tienen esta familia de generadores es que por medio del algoritmo Berlekamp-Massey, los LFSR son fácilmente quebrables. Tras capturar n bits generados, es posible encontrar en tiempo polinómico el estado del generador y de esta forma predecir los siguientes bits. Este algoritmo de hecho se puede utilizar para decodificar códigos de la familia BCH como los Reed-Solomon.

Para aumentar la fortaleza de los generadores aleatorios, una herramienta comúnmente utilizada era la de colocar n LFSR en paralelos combinándolos mediante una función booleana, donde para $i \in \{1 \dots n\}$ la salida del LFSR_i estaba conectado a la función booleana correspondiendo su entrada x_i.



Ya en [11, página 10] se decía que uno de los problemas comunes de utilizar este esquema de combinación mediante funciones booleanas, es que muchas de las funciones booleanas tienden a exponer información sobre los generadores internos; en el sentido en que la salida final aparece correlacionada con una o más de las salidas de los generadores internos y que en tales casos dicho esquema es fácilmente rompible.

Por tanto para aumentar la dificultad de estos ataques es que necesitamos que la función sea inmune a la correlación.

1.2. Motivación general del proyecto.

Dentro de la literatura existente, se encuentran varios usos para las funciones booleanas inmunes a la correlación, a modo de ejemplo lado Carlet et al. presentaron otra aplicación de estas funciones para proteger ataques a AES[1]. [3, 5]

A tales efectos es necesario conocer funciones k-CI de bajo peso (cantidad de x para los cuales $f(x) = 1$), en algunos casos con cantidades de variables relativamente grandes. Resultan por tanto interesantes los siguientes problemas abiertos:

1. Dados n y k, saber cuál es el peso mínimo de una función booleana k-CI de n variables.

[1] En particular utilizan funciones booleanas k-CI en técnicas de enmascarado para proteger a implementaciones en hardware contra ataques de canal secundario.

2. Dados n, k y w, saber cuántas funciones booleanas de n variables, de inmunidad a la correlación de orden k y peso w existen.

3. Dados los parámetros antes mencionados, encontrar criterios para poder enumerar estas funciones, de forma tal que sea fáctible:

4. poder tener un algoritmo razonablemente eficiente de generación aleatoria uniforme.

Definición 1.1 (función booleana)**.** Llamamos funciones booleanas a aquellas que toman un vector de n bits como entrada y devuelven un bit como salida; y debido a que el dominio de las mismas puede verse ya sea como un vector o como n parámetros de un bit cada uno, se utiliza indistintamente ambas perspectivas. Para referirnos a los bits, en el área se suele designar al conjunto $\{0, 1\}$ mediante $\mathbb{F}_2$ (formalmente campo de Galois de tamaño 2) con la suma definida como el `XOR` (denotado con $\oplus$) y la multiplicación como en los enteros. Por tanto a las funciones booleanas de n variables se las denota mediante $\mathbb{F}_2^n \mapsto \mathbb{F}_2$.

A continuación se muestra una tabla de verdad con todas las funciones booleanas de 2 variables ($\mathbb{F}_2^2 \mapsto \mathbb{F}_2$), viéndose fácilmente cómo para n variables existen 2^{2^n} funciones booleanas.

x_2	x_1	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
0	0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
0	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	0	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Debido a que la cantidad de funciones booleanas crece muy rápidamente en función de la cantidad de variables $\mathcal{O}(2^{(2^n)})$, mientras que las cantidades de funciones k-CI son significativamente más chicas; resultan por tanto inviables las búsquedas por extensión sobre todo el espacio.

n	# funciones	# 1-CI	2-CI	3-CI	4-CI
1	4	2			
2	16	2	2		
3	256	14	2	2	
4	65.536	636	8	2	2
5	4.294.967.296	3.139.004	1044	10	2
6	18.446.744.073.709.551.616	95.259.103.924.394	¿?	¿?	¿?
10	$2^{1024} \approx 1,8 \times 10^{308}$	¿?	¿?	¿?	¿?

2. Definiciones básicas.

2.1. Peso y distancia de Hamming.

A lo largo de este trabajo utilizaremos repetidas veces el concepto de peso y distancia de Hamming, definida tanto para vectores como para funciones.

Definición 2.1 (peso de Hamming de un vector)**.** Sea u un vector en $\mathbb{F}_2^n$, llamamos peso de Hamming del vector a la cantidad de entradas distintas de 1: $w_H(u) = \#\{i/u_i = 1\} = \sum u_i$.

Definición 2.2 (peso de Hamming de una función)**.** Sea $f : \mathbb{F}_2^n \mapsto \mathbb{F}_2$, llamamos peso de Hamming de la función booleana a la cantidad de unos en la imagen de la función: $w_H(f) = \#\{x/f(x) = 1 \forall x \in \mathbb{F}_2^n\} = \sum_{x \in \mathbb{F}_2^n} f(x)$.

Definición 2.3 (distancia hamming de vectores)**.** Para vectores $u, v \in \mathbb{F}_2^n$, la distancia de hamming entre u y v: $d_H(u, v) = \#\{i/u_i \neq v_i\}$.

Definición 2.4 (distancia hamming de funciones)**.** Mientras que para funciones $f, g : \mathbb{F}_2^n \mapsto \mathbb{F}_2$, la distancia de Hamming la definimos como: $d_H(f, g) = \#\{x/f(x) \neq g(x)\}$.

Ejemplo 2.1. A modo de ejemplo la distancia entre los vectores (1101) y (1011) es 2 al haber dos bits distintos.

Se puede ver que tanto para vectores como para funciones, se cumple: $d_H(u, v) = w_H(u \oplus v)$ y $d_H(f, g) = w_H(f \oplus g)$.

2.2. Función atómica

Definición 2.5. Decimos que $f : \mathbb{F}_2^n \mapsto \mathbb{F}_2$ es una función atómica si $w_H(f) = 1$.

Dada una constante $c \in \mathbb{F}_2^n$, vemos que este tipo de funciones tienen la forma:

$$f(x) = (x_1 \oplus c_1)(x_2 \oplus c_2) \dots (x_n \oplus c_n),$$

ya que claramente vale 1 si y solo si $x = \bar{c}$.

Ejemplo 2.2. Función atómica que vale 1 únicamente en la entrada (010), $f = (x_3 \oplus 1)(x_2 \oplus 0)(x_1 \oplus 1)$:

x_3	x_2	x_1	f
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

2.3. Forma Algebraica Normal (ANF).

Para expresar funciones, además de listar los valores de la función para cada combinación de los parámetros (como en la tabla de verdad), las mismas se pueden escribir en la forma normal algebraica (ANF), la cual se caracteriza por ser una suma de términos en donde cada término es una multiplicación de un subconjunto de los parámetros. Vemos en el ejemplo 2.3 como la función f_{11} puede escribirse como $1 \oplus x_2 \oplus x_1 x_2$ en su forma normal.

Notar que toda función booleana se puede escribir como el `XOR` de funciones atómicas, por tanto si conocemos la tabla de verdad de una función, un método trivial de obtener su forma algebraica normal es, tras escribirla como

XOR de funciones atómicas, desarrollarlas expandiendo primero con la propiedad distributiva, y tachando después términos repetidos.

De esta manera la conversión a la forma algebraica siempre existe, y termina siendo única salvando el orden.

Ejemplo 2.3. Desarrollo en base a funciones atómicas.

Vemos aquí como $f_{11} = f_1 \oplus f_2 \oplus f_8$.

x_2	x_1	f_1	f_2	f_8	f_{11}
0	0	1	0	0	1
0	1	0	1	0	1
1	0	0	0	0	0
1	1	0	0	1	1

Expandiendo y desarrollando obtenemos:

$$\begin{aligned} f_{11}(x) &= f_1 \oplus f_2 \oplus f_8 \\ &= (x_2 \oplus 1)(x_1 \oplus 1) \ \oplus \ (x_2 \oplus 1)(x_1 \oplus 0) \ \oplus \ (x_2 \oplus 0)(x_1 \oplus 0) \\ &= (x_2 \oplus 1)(x_1 \oplus 1) \ \oplus \ (x_2 \oplus 1)x_1 \ \oplus \ x_1x_2 \\ &= x_1x_2 \oplus x_1 \oplus x_2 \oplus 1 \ \oplus \ x_1x_2 \oplus x_1 \ \oplus \ x_1x_2 \\ &= x_1x_2 \oplus x_2 \oplus 1. \end{aligned}$$

Recordamos que en $x_i \in \mathbb{F}_2$; $x_i \oplus x_i = 0$, por lo que se cancelan los términos repetidos cantidad par de veces. De la misma forma la multiplicación corresponde a la operación booleana AND, por lo cual $x_i^2 = x_i$, no apareciendo entonces exponentes mayores a 1.

2.4. Transformada de Fourier.

Otra manera de representar funciones booleanas es por medio de sus transformadas de Fourier o Walsh (transformada de fourier de la función $\lambda x.(-1)^f(x)$).

Esta representación nos permite caracterizar elegantemente a las funciones k-CI, y es ésta la caracterización sobre la cual se sustentan nuestros algoritmos combinatorios.

La transformada de fourier de una función booleana $f : \mathbb{F}_2^n \mapsto \mathbb{F}_2$ se denota como $\hat{f} : \mathbb{F}_2^n \mapsto \mathbb{Z}$, con el codominio en $[-2^n, 2^n]$. Para poder comprender fácilmente, hay que tener en mente que:

- hay un caso particular: $\hat{f}(0) = w_H(f)$, el cual obviamente es siempre no negativo.
- Para calcular el resto de los casos de $\hat{f}(u)$; el dominio ($\mathbb{F}_2^n$) se separa en dos clases de equivalencia con igual cantidad de elementos (separación que depende de u). Se suma (en enteros, no usando XOR) por un lado los valores $f(x)$ para los x de la primera clase, y se le resta los valores de $f(x)$ para los x de la otra clase.

  Para determinar cuáles valores suman y cuáles restan, nos basamos en la función afín $\ell_u = \sum_{i \in \{1 \ldots n\}} u_i x_i$. Sumamos los valores de $f(x)$ tales que $\ell_u(x) = 0$ y restamos aquellos con $\ell_u(x) = 1$.

- Debido a que $\ell_{\vec{0}}(x) = 0$, el punto anterior es consistente con el hecho de que $\hat{f}(0)$ sea el $w_H(f)$.

Definición 2.6 (transformada de Fourier)**.** La definición formal es la siguiente (donde $x \cdot u$ es el producto interno de x por u, o sea $x \cdot u = \bigoplus_{i \in \{1 \dots n\}} x_i u_i$):

$$\hat{f}(u) = \sum_{x \in \mathbb{F}_2^n} f(x)(-1)^{x \cdot u}$$

Como notación para expresar vectores $u \in \mathbb{F}_2^n$, a veces utilizaremos la notación $u = \langle i \rangle_n$ con $i \in \{0 \dots 2^n - 1\}$, donde u se interpreta como la representación binaria de i en bits, formalmente: $i = \sum\{2^{j-1} / x_j = 1\}$. Según el contexto puede llegar a omitirse el largo del vector, asumiéndose largo n.

Ejemplo 2.4. Vemos a continuación un ejemplo en donde: $\ell_{\langle 0 \rangle}(x) = 0$, $\ell_{\langle 1 \rangle}(x) = x_1$, $\ell_{\langle 2 \rangle}(x) = x_2$ y $\ell_{\langle 3 \rangle}(x) = x_1 \oplus x_2$:

x_2	x_1	f_{11}	$\widehat{f_{11}}$	$\ell_{\langle 0 \rangle}$	$\ell_{\langle 1 \rangle}$	$\ell_{\langle 2 \rangle}$	$\ell_{\langle 3 \rangle}$
0	0	1	3	0 (+)	0 (+)	0 (+)	0 (+)
0	1	1	-1	0 (+)	1 (−)	0 (+)	1 (−)
1	0	0	1	0 (+)	0 (+)	1 (−)	1 (−)
1	1	1	1	0 (+)	1 (−)	1 (−)	0 (+)

Se ve en el ejemplo 2.4 como salvo la afinidad nula, todas las otras afinidades tienen peso 2^{n-1}. Esto explica por qué la transformada de fourier separa los valores en dos mitades; de aquí se desprende por qué es relevante tener en mente estas afinidades al momento de trabajar con las transformadas de Fourier.

Es entonces importante recalcar que el hecho que la transformada de fourier valga cero para ciertos conjuntos de vectores (como los de peso menor a un k) nos muestra una propiedad de "equilibrio" o "balance" que es interesante entender; más específicamente los valores de x tales que $f(x) = 1$ deben ser seleccionados con criterios específicos para que satisfagan estas condiciones.

Definición 2.7 (deltas de clases)**.** Como "notación alternativa" utilizada en este y otros trabajos llamamos $\delta_i(f) \stackrel{def}{=} \hat{f}(u)$ donde $u \in \mathbb{F}_2^n$ tiene solamente la entrada i en 1.

Generalizamos el delta anterior para conjuntos de enteros $I \subseteq \{1 \dots n\}$; donde $\delta_I(f) \stackrel{def}{=} \hat{f}(u)$ donde $u \in \mathbb{F}_2^n$ es el vector que tiene solamente las entradas $i \in I$ en 1; en otras palabras, donde I es el soporte de u.

Para calcular eficientemente simultáneamente todos los valores de $\hat{f}$ se puede utilizar el algoritmo conocido como transformada rápida de fourier (FFT).

2.5. Restricción de una función.

Definición 2.8 (restricción de una función)**.** Este concepto que informalmente venía diciendo como "ver los casos de f para los cuales $x_i = c$" se conoce como restricción de una función, denotándose $f|_{x_i = c}$.

Estas restricciones pueden verse como una aplicación parcial en la que el resultado es una función con un argumento menos. $(f|_{x_i=c})(x_1, \ldots, x_{n-1}) \stackrel{def}{=} f(x_1, \ldots, x_{i-1}, c, x_i, \ldots, x_{n-1})$.

La restricción de una función se puede también generalizar de manera análoga para múltiples argumentos como podemos ver en el ejemplo 2.5

Ejemplo 2.5. Vemos a continuación un ejemplo de restricciones de una función la función $f = 1 \oplus x_1 \oplus x_2 \oplus x_3$, en donde primero aplicamos parcialmente x_3 y luego x_1.

x_3	x_2	x_1	f
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

$\Longrightarrow$

x_2	x_1	$f\|_{x_3=0}$	$f\|_{x_3=1}$
0	0	1	0
0	1	0	1
1	0	0	1
1	1	1	0

$\Downarrow$

x_2	$f\|_{x_3=1,x_1=1}$
0	1
1	0

3. Inmunidad a la correlación.

Para entender la motivación se puede pensar en la siguiente aproximación al siguiente problema; en el que queremos definir una propiedad que capture la siguiente situación:

> Supongamos que tenemos una función booleana $f : \mathbb{F}_2^n \mapsto \mathbb{F}_2$ desconocida por el atacante. Asumamos que el atacante controla a lo sumo k entradas cualquiera de la función (pudiendo elegir cualquier valor arbitrario para estas entradas, ver motivación en sección 1), y puede observar el peso de la restricción de la función (el peso de la función resultante $\mathbb{F}_2^{n-k} \mapsto \mathbb{F}_2$ al dejar libres solamente las otras $n - k$ entradas, atando las k variables a los valores elegidos).
>
> Pedir que el atacante obtenga siempre el mismo peso por más que varíe los valores de estas a lo sumo k entradas elegidas, es pedir que la función sea k-CI (inmune a la correlación de orden k).

Definición 3.1 (inmunidad a la correlación de orden k)**.** Una función f es k-CI para $k \geq 0$, si además de ser $(k-1) - CI$ se cumple que para todo conjunto de k variables, todas las restricciones de f dejando fijas estas k variables tienen la misma proporción de ceros y unos; y por tanto también el mismo peso.

En el ejemplo 2.5 podemos observar inmediatamente que $w_H(f|_{x_3=0}) = w_H(f|_{x_3=1})$, ídem para x_1 y x_2; por lo cual sabemos que f es 1-CI. Si miramos los pesos de las restricciones aplicando parcialmente x_3 y x_2 vemos que las 4 restricciones tienen el mismo peso; ídem con x_3 y x_1, y con x_2 y x_1 por lo que sabemos que f es además 2-CI.

Teorema 3.1. Pedir que una función booleana $f : \mathbb{F}_2^n \mapsto \mathbb{F}_2$ sea k-CI es equivalente a que $\hat{f}(u) = 0$ para todo $u : 1 \leq w_H(u) \leq k$.

La demostración de esta propiedad (originalmente demostrado en [15].) para 1-CI es bastante directa (disponible en teorema A.1); sin embargo para el caso general se puede ver a continuación una idea de una posible prueba, mientras que damos una demostración formal y sorprendentemente corta aunque complicada en el apéndice B.

Esta idea de prueba no solo se presenta por completitud, sino porque ayuda a entender los conceptos fundamentales.

Idea de la prueba.
($\Rightarrow$)

Sabemos que el hecho de que f sea k-CI implica que, dado un u con $0 < w_H(u) \leq k$, si agrupamos los valores de $f(x)$ según los $(x_i)_{i \in supp(u)}$, al sumarlos obtendremos el mismo peso para cada uno de estos grupos.

Como $\hat{f}(u)$ es la suma y resta de los pesos de estos grupos, donde la mitad de dichos grupos se suman y la otra mitad se restan; el resultado necesariamente debe ser cero.

El signo de cada uno de dichos grupos, tal como se puede ver en la definición de la transformada de f, queda determinado mediante la paridad de los bits en común entre x y u, esto es el efecto de realizar el producto interno: $(-1)^{x \cdot u}$

($\Leftarrow$)

Dado un u arbitrario con $0 < w_H(u) = r \leq k$, sabemos que los bits de u en 1 nos separan a los valores de f en 2^r grupos, que queremos probar tienen el mismo peso.

Sabemos también que la transformada de Fourier de f vale 0 en los $2^r - 1$ vectores u' tales que $\emptyset \neq supp(u') \subseteq supp(u)$, ya que el peso de u' está entre 1 y k. Al escribir las ecuaciones de estas $2^r - 1$ evaluaciones mediante la suma y resta de 2^r variables, donde estas variables corresponden a los pesos de los grupos mencionados en el párrafo anterior; llegamos a un sistema de ecuaciones de 2^r variables y $2^r - 1$ ecuaciones.

Más aún, si nos quedamos con la matriz de coeficientes del sistema y agregamos una fila de unos al principio (que hubiese correspondido a evaluar la transformada en el vector nulo), llegamos a la famosa construcción de Sylvester de matrices de Hadamard. Dada por:

$$H_1 = (1), \qquad H_{2^k} = \begin{pmatrix} H_{2^{k-1}} & H_{2^{k-1}} \\ H_{2^{k-1}} & -H_{2^{k-1}} \end{pmatrix}$$

Las matrices de Hadamard además de invertibles son ortogonales, esto es, las filas forman un espacio *ortonormal*. De forma tal que volviendo al sistema original en donde no estaba presente la fila de unos, nos queda una variable libre, concluyendo en que las 2^r variables tienen que ser iguales.

Entonces como se mantiene la misma relación entre los pesos de todas las restricciones según los $r \leq k$ bits de u arbitrario, f tiene que ser k-CI. $\square$

4. Fundamentación formal del proyecto.

El desafío es mucho más difícil del que parece. Para empezar porque no se conocen teoremas generales tal que dados n, k indiquen cuál es el peso mínimo $\bar{m}(n,k)$ para el cual existan funciones k-CI.

Más aún, existen métodos no constructivos[3] que en algunos casos encuentran dicho mínimo pero en el cual no se conocen construcciones.

Por otro lado este problema para el caso $k = 2$ está profundamente relacionado para n en general con la famosa conjetura de Hadamard. [4]

Estas evidencias son una muestra clara de la complejidad del problema.

Es importante indicar que en [3] hay dos tablas: en una de ellas se presentan los $\bar{m}(n,k)$ conocidos hasta n=14, donde en muchos lugares figuran interrogación indicando desconocimiento de esos valores; ya ni mencionemos *encontrar una función óptima*.

Hay además otra tabla de las mejoras cotas inferiores conocidas.

Por tal motivo se realizaron dos proyectos de grado de Ingeniería en Computación en el marco de colaboración conjunta con la universidad de Caen para extender los métodos presentados en [10, 8] a este problema.

La dificultad más importante de este método *constructivo* es la explosión combinatoria de la dificultad del problema al crecer n y k. Mostramos alguna evidencia al respecto.

Es importante notar que las funciones k-CI son invariantes a ciertas transformaciones, por ejemplo si f es k-CI y g es la función f en la cual la variable x_i la llamamos x_j y viceversa ($x_i \leftrightarrow x_j$) es claro que si bien son funciones diferentes, a los efectos de este problema, es esencialmente la misma función.

Definir, entender y caracterizar algunas de estas simetrías es el objetivo de este proyecto. Más aún, la idea es ayudar a desarrollar algoritmos más eficientes (en tiempo y espacio) para resolver el problema planteado por Carlet. En este sentido hemos hecho avances parciales y seguimos trabajando en los aspectos algorítmicos.

El método combinatorio introducido en la sección 5 y detallado en [10, 8] se sustenta en definir clases de equivalencia de funciones. Cómo generalizar estas clases para $k > 1$ y cómo agrupar clases que capturen las simetrías del problema; buscando definir de una menera clara y precisa una clase que la represente que llamaremos “clase normal”, es el objetivo central de este proyecto.

5. Métodos para construcciones de funciones booleanas k-CI.

Tanto en el survey [6] como en la sección 2.1 de [12] se presentan métodos para encontrar funciones inmunes a la correlación. Sin embargo ninguno de estos métodos es exhaustivo y no se da métodos verificables para generar funciones k-CI de menor peso. mucho menos aún generación aleatoria uniforme de dichas funciones.

En 2011 Carrasco et al. presentan un novedoso método combinatorio pa-

ra contar, enumerar y generar aleatoriamente de manera uniforme funciones 1-resiliente de hasta 9 variables.[8].

En una reunión del proyecto ANR Boole realizada en París en junio del 2014, Claude Carlet le planteó a los autores de [10, 8] la posibilidad de generalizar estos métodos para hallar funciones k-CI de menor peso para valores de n hasta 14.

Para comprender fácilmente dicho algoritmo se recomienda recurrir a la tesis de Sebastián Fonseca y Cecilia García [13] en donde se presenta y explica una implementación en alto nivel.

El método inicial de Le Bars y Viola,[10] se basaba en agrupar las funciones en clases de equivalencia según los valores que presentaban las mismas en las entradas de peso 1 de sus transformadas de Fourier. Las clases, llamadas clases de correlación de primer orden, eran simplemente una lista ordenada de la forma: $\langle w_H(f)|\delta_n(f), \ldots, \delta_1(f)\rangle$ (ver definición de δ en 2.7).

El método de Carrasco et al. [8] introdujo como nueva contribución propiedades que, relacionando las clases de correlación entre sí, permitían no tener que guardar en la base de datos todas las clases. Alcanzaba entonces guardar solamente las clases normales. Estas propiedades son:

- Existe la misma cantidad de funciones con la clase $\langle w_H|\delta_n, \ldots, \delta_i, \ldots, \delta_1\rangle$, que con la clase: $\langle w_H|\delta_n, \ldots, -\delta_i, \ldots, \delta_1\rangle$.

  Esto significaba que no tenía sentido guardar en la base de datos las clases de correlaciones con δ_i negativos.

- Existe la misma cantidad de funciones con la clase

  $$\langle w_H|\delta_n, \ldots, \delta_i, \ldots, \delta_j, \ldots, \delta_1\rangle\,,$$

  que con la clase:

  $$\langle w_H|\delta_n, \ldots, \delta_j, \ldots, \delta_i, \ldots, \delta_1\rangle\,.$$

  Significando que la cantidad de funciones no dependía del orden en que apareciesen estos δ_i; por tanto alcanzaba guardar solamente las clases en que los deltas estaban ordenados.

Se llamó clases de correlaciones normales al pequeño subconjunto de las clases de correlaciones que cumplían con estos requisitos (tener deltas positivos y ordenados: $\delta_n \leq \ldots \leq \delta_1$).

Debido a la sencillez de estas propiedades, la normalización era un proceso bastante sencillo, alcanzaba con tomar los valores absolutos y ordenar los deltas. También es sencillo a partir de la normal saber cuáles son las no normales (cambiando los signos de los deltas distintos de cero y tomando todos los órdenes posibles).

Más aún, se encontró que si se cambiaban las variables x_i por x_j en la entrada de una función, se alternaban también los valores de δ_i y δ_j. Y de la misma forma, si en la entrada de la función booleana se negaba una de las variables, cambiaba el signo del delta correspondiente. Presentamos la demostración de este teorema en el apéndice A.

6. Simetrías.

Apareció entonces la siguiente pregunta: ¿Se cumplen estas propiedades antes mencionadas al tomar clases de correlación de orden k?, o mejor dicho ¿qué propiedades similares pueden llegar a cumplirse?

Tomando como punto de partida los trabajos realizados con clases de correlación de primer orden, una primer definición de clases de correlación de orden k fue simplemente tomar una generalización en donde aparecían no solo los valores de la transformada de Fourier para las entradas de peso 0 ($w_H(f)$) y 1 ($\delta_i(f)$), sino todos los $u : w_H(u) \leq k$.

6.1. Traslación

Decimos que g es *traslación* de f cuando se cumple $g(x) = f(x + w)$ para algún w vector constante en $\mathbb{F}_2^n$; sin embargo nos basta en un principio el considerar el caso más sencillo: en donde w tiene un 1 solamente en la posición i.

Recordamos que en $\mathbb{F}_2$ la única traslación posible de una variable es hacerle `XOR` con 1, en otras palabras hacer la negación de la misma.

Teorema 6.1 (traslación en una variable para clases de correlación de orden k)**.** Dados $i \in \{1 \dots n\}$, f función booleana de n variables, y $g(x) = f(x \oplus a)$ con a en $\mathbb{F}_2^n$ valiendo 1 solamente en a_i; entonces $\hat{f}(u) = -\hat{f}(u)$ cuando $u_i = 1$, y $\hat{f}(u) = \hat{f}(u)$ en otro caso.

Este mismo teorema lo podemos expresar también mediante los cambios en las clases de equivalencia de orden k (que recordamos las tomamos es una lista ordenada de los $\hat{f}(u)$ para los $u : w_H(u) \leq k$).

Se cumple $M_i(\Omega^{[k]}(f)) = \Omega^{[k]}(m_i(f))$. Donde:

- $m_i(f)(x) = f(x \oplus a)$: con $a \in \mathbb{F}_2^n$ tal que a vale 1 solamente en a_i.
- M_i es definido para clases de correlaciones de orden k; donde dado $\omega = \Omega^{[k]}(f)$, $M_i(\omega) = \langle w_H, \alpha_n, \dots, \alpha_1, \alpha_{n,n-1}, \dots, \alpha_{2,1}, \dots \rangle$ con $\alpha_I = -\delta_I(f)$ si $i \in I$, mientras que $\alpha_I = \delta_I(f)$ si $i \notin I$ para todo $I \subseteq \{1, \dots, n\}$ / $|I| \leq k$.

Demostración disponible en apéndice B.

En otras palabras, al trasladar f negando el valor de la variable x_i podemos observar que en la clase de correlación cambia el signo de aquellos $\delta_I(f)$ en los cuales $i \in I$.

Ejemplo 6.1. Al trasladar la función $f(x_3, x_2, x_1) = x_1 \oplus x_2x_1 \oplus x_3x_2x_1$ cambiando el valor de x_2, obtenemos $m_2(f)(x_3, x_2, x_1) = x_1 \oplus (x_2 \oplus 1)x_1 \oplus x_3(x_2 \oplus 1)x_1 = x_2x_1 \oplus x_3x_1 \oplus x_3x_2x_1$. Se puede ver en las columnas con $x_2 = 1$ el cambio de los signos:

$f(x) =$	0	1	0	0	0	1	0	1
$m_2(f)(x) =$	0	0	0	1	0	1	0	1
$\hat{f}(x) =$	3	-3	1	-1	-1	1	1	-1
$\widehat{m_2(f)}(x) =$	3	-3	-1	1	-1	1	-1	1

x_1	0	1	0	1	0	1	0	1
x_2	0	0	1	1	0	0	1	1
x_3	0	0	0	0	1	1	1	1

6.2. Permutación de las variables

Aprovechamos para recordar que $\delta_I(f)$ se define como $\hat{f}(u)$ con $supp(u) = I$ (I es el conjunto de los $\{i/u_i = 1\}$).

De manera similar a como ocurría con clases de correlación de orden 1, en orden k lo que ocurre es que si cambiamos x_i por x_j, veremos que cambiarán a su vez los δ_I por $\delta_{\mathrm{map}_\pi I}$, con $\pi = (i\ j)$. En resumen, cambiarán $i \leftrightarrow j$ también dentro de I.

Más formalmente, dados $i, j \in \{1 \dots n\}$ distintos y $\pi_{i,j}$ la permutación $(i\ j)$, llamamos $\pi_{i,j}(f)$ a la función con las variables permutadas de f, definido como: $\pi_{i,j}(f)(x_1, \dots, x_n) = f(x_{\pi_{i,j}(1)}, \dots, x_{\pi_{i,j}(n)})$.

Teorema 6.2. Se cumple $\Pi_{i,j}(\Omega^{[k]}(f)) = \Omega^{[k]}(\pi_{i,j}(f))$. Donde:

- $\Pi_{i,j}$ es definido para clases de correlaciones de orden k; donde dado $\omega = \Omega^{[k]}(f)$, $\Pi_{i,j}(\omega) = \langle w_H, \alpha_n, \dots, \alpha_1, \alpha_{n,n-1}, \dots, \alpha_{2,1}, \dots \rangle$ con $\alpha_I = \delta_{I'}(f)$, $I' = \{\pi_{i,j}(\ell)/\ell \in I\}$.

Demostración disponible en apéndice B.

Al intercambiar las variables x_i y x_j en f, se puede observar como en la clase de correlación se intercambian los deltas de las entradas I con los de $\{\pi_{i,j}(\ell)/\ell \in I\}$. Nótese además que la cardinalidad del índice de los deltas no cambia al permutarse, por lo que sabemos que tanto f como su permutación tienen la misma inmunidad a la correlación.

Ejemplo 6.2. En una función f de tres variables, al intercambiar x_2 con x_3 cambian $\delta_{\{2\}} \leftrightarrow \delta_{\{3\}}$, y $\delta_{\{1,2\}} \leftrightarrow \delta_{\{1,3\}}$.

6.3. Composición con transformaciones lineales.

Hasta el momento el núcleo del trabajo se centró en el estudio de cómo cambiaba la transformada de Fourier de una función booleana al componerla con una permutación de variables o una traslación, ya que mediante el aprovechamiento de simetrías existentes en estas composiciones es posible diseñar algoritmos que ayuden a obtener sistemáticamente funciones booleanas inmunes a la correlación de orden k.

Sin embargo es importante estudiar también la relación entre $\widehat{f}$ y $\widehat{f \circ T}$ para afinidades T invertibles; ya que esto nos permitiría reducir aún más la cantidad de clases normales. Se puede ver como la parte no lineal de la afinidad (la

traslación) implica sobre los valores de la transformada de fourier únicamente cambios de signo; mientras que la parte lineal (la transformación lineal) va a implicar únicamente la permutación de los valores de la transformada de fourier.

6.3.1. Composición con automorfismos afines.

En este contexto de funciones booleanas, vemos las afinidades invertibles (también llamadas automorfismos afines) como funciones de $\mathbb{F}_2^n \mapsto \mathbb{F}_2^n$, que tomando un vector $(x_1, \ldots, x_n)$, lo aplican a una transformación lineal invertible también sobre $\mathbb{F}_2^n$ y al resultado le suman un vector de $\mathbb{F}_2^n$. Podemos verlo como multiplicación de matrices:

$$\begin{pmatrix} \alpha_{11} & \cdots & \alpha_{1n} & \beta_1 \\ \vdots & \ddots & \vdots & \vdots \\ \alpha_{n1} & \cdots & \alpha_{nn} & \beta_n \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ \vdots \\ x_n \\ 1 \end{pmatrix} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$$

En otras palabras $y_i = \beta_i \oplus \bigoplus_{j=1}^{n} \alpha_{ij} x_i$.

En las secciones anteriores, las simetrías vistas fueron por un lado las permutaciones de los argumentos de las funciones booleanas, y por otro la traslación de los argumentos, o dicho en otras palabras la suma de una constante. Ambas simetrías se pueden ver como composición de funciones booleanas con afinidades: $f \circ A$.

Ejemplo 6.3. Afinidad de permutación entre x_2 y x_3:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ 1 \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} x_1 \\ x_3 \\ x_2 \end{pmatrix}$$

Ejemplo 6.4. Afinidad de traslación que devuelve el complemento para los valores x_1 y x_3 (o sea $y_1 = 1 \oplus x_1$, $y_2 = x_2$ e $y_3 = 1 \oplus y_3$):

$$\begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ 1 \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} 1 \oplus x_1 \\ x_2 \\ 1 \oplus x_3 \end{pmatrix}$$

Para este caso de traslación, la afinidad sería $A(x) = Id(x) \oplus a = x \oplus a$. Quedando clara la generalización de las simetrías sobre las clases de correlaciones para composición con estas afinidades (en las que la parte de transformación lineal es la identidad), $\forall I \subseteq \{1, \ldots, n\}$:

$$\delta_I(f \circ A) = \delta_I(f) \cdot (-1)^{|I \cap supp(a)|}.$$

Como toda afinidad $A(x) = T(x) + a$ sobre $\mathbb{F}_2^n$ invertible se puede escribir como $A(x) = T(x + a')$ con $a' = T \backslash a$, se cumple:

$$\delta_I(f \circ A) = \delta_I(f \circ T) \cdot (-1)^{|I \cap supp(a')|}.$$

Lo cual nos indica que la composición con toda afinidad invertible termina siendo un problema de dos partes mayormente independientes, por un lado los cambios de signo sobre los deltas son ocasionados por la parte no lineal de la afinidad (la suma del despejable vector a), y por otro la composición con transformaciones lineales que, ya veremos en esta a continuación, simplemente permutan los valores.

6.3.2. Composición con transformaciones lineales invertibles.

En esta sección se muestra como los cambios generados por la composición de funciones booleanas con transformaciones lineales invertibles, terminan siendo únicamente permutaciones de los valores de las transformadas de Fourier.

Teorema 6.3. Sean $i, j \in \{1, \ldots, n\}$ con $i \neq j$; $L_{i,j} : \mathbb{F}_2^n \mapsto \mathbb{F}_2^n$ transformación lineal invertible[2], con la matriz asociada de $L_{i,j}$ valiendo 1 solamente en la diagonal y en la posición i, j. Entonces:

$$\begin{aligned} &\delta_I(f \circ L_{i,j}) = \delta_{I \triangle \{j\}}(f), && \forall I / \{i\} \in I \subseteq \{1, \ldots, n\}, y \\ &\delta_I(f \circ L_{i,j}) = \delta_I(f), && \forall I / \{i\} \notin I \subseteq \{1, \ldots, n\}. \end{aligned}$$

Demostración. Siendo $I = supp(u)$, y recordando que $L_{i,j}$ es también una involución, procedemos con la prueba, en donde primero expandimos la definición de delta, después hacemos un cambio de variable $x \leftrightarrow L_{i,j}^{-1}(x)$ y separamos el producto escalar en casos.

$$\begin{aligned} \delta_I(f \circ L_{i,j}) &= \sum_{x \in \mathbb{F}_2^n} (f \circ L_{i,j})(x)(-1)^{x \cdot u} \\ &= \sum_{x \in \mathbb{F}_2^n} f(x)(-1)^{L_{i,j}^{-1}(x) \cdot u} \\ &= \sum_{x \in \mathbb{F}_2^n} f(x)(-1)^{L_{i,j}(x) \cdot u} \end{aligned}$$

Primer caso: $i \notin I$: por tanto $u_i = 0$, y $L_{i,j}(x)_k = x_k \ \forall k \neq i$.

$$L_{i,j}(x) \cdot u = \bigoplus_{k=1}^{n} L_{i,j}(x)_k u_k = \bigoplus_{k=1}^{n} x_k u_k = x \cdot u$$

Caso restante: $i \in I$: con $v = supp(I \triangle \{j\})$ donde v es el vector u con el bit en j alternado.

$$\begin{aligned} L_{i,j}(x) \cdot u &= \bigoplus_{k=1}^{n} L_{i,j}(x)_k u_k = x_i \oplus x_j \oplus \bigoplus_{\substack{k=1 \\ k \neq i}}^{n} x_k u_k = x_j \oplus \bigoplus_{k=1}^{n} x_k u_k \\ &= x_j \oplus x_j u_j \oplus \bigoplus_{\substack{k=1 \\ k \neq j}}^{n} x_k u_k = (1 \oplus x_j) u_j \oplus \bigoplus_{\substack{k=1 \\ k \neq j}}^{n} x_k u_k = \bigoplus_{k=1}^{n} x_k v_k = x \cdot v \end{aligned}$$

Quedando entonces la sumatoria, en el primer caso igual a $\delta_I(f)$ y en el otro caso igual a $\delta_{I \triangle \{j\}}(f)$. □

[2]Este tipo de matrices elementales son además involuciones al encontrarnos en $\mathbb{F}_2$. Sobre $\mathbb{R}$ sin embargo, $L_{i,j}^{-1} = -L_{i,j}$.

Teorema 6.4. Sea T una transformación lineal invertible sobre $\mathbb{F}_2$. Entonces existe $m \in \mathbb{N}$ tal que T puede escribirse como la composición de transformaciones $T = T_1 \circ \ldots \circ T_m$, donde $T_1, \ldots, T_m : \mathbb{F}_2^n \mapsto \mathbb{F}_2^n$ son transformaciones lineales, cuyas matrices asociadas valen 1 en la diagonal y en a lo sumo otra posición (no necesariamente la misma en todas las matrices).

Demostración. Demostración por inducción en el tamaño de T. Para $\mathcal{M}_{1x1}(\mathbb{F}_2)$ existe una única matriz invertible (la que tiene un uno), siendo ésta la propia identidad.

Tomamos como hipótesis inductiva que cualquier matriz $n \times n$ invertible S puede escribirse como el producto de matrices como la identidad pero con un uno más.

El primer paso es entonces construir la matriz $(n+1)\times(n+1)$ mediante las siguientes multiplicaciones:

$$\begin{pmatrix} Id_n & \begin{matrix}0\\\vdots\\0\end{matrix} \\ \alpha'_1 \ \ldots \ \alpha'_n & 1 \end{pmatrix} \quad \begin{pmatrix} S & \begin{matrix}0\\\vdots\\0\end{matrix} \\ 0 \ \ldots \ 0 & 1 \end{pmatrix} \quad \begin{pmatrix} S & \begin{matrix}\beta_1\\\vdots\\\beta_n\end{matrix} \\ \alpha_1 \ \ldots \ \alpha_n & 1 \end{pmatrix} \quad \begin{pmatrix} Id_n & \begin{matrix}\beta'_1\\\vdots\\\beta'_n\end{matrix} \\ 0 \ \ldots \ 0 & 1 \end{pmatrix}$$

Como S es invertible, existen por tanto $(\alpha'_1, \ldots, \alpha'_n)$ solución del sistema $S^\intercal \backslash (\alpha_1, \ldots, \alpha_n)$; y $(\beta'_1, \ldots, \beta'_n)$ solución de $S \backslash (\beta_1, \ldots, \beta_n)$.

Es importante notar que la matriz con la identidad y los α' también puede escribirse de la forma buscada.

$$\begin{pmatrix} Id_n & \begin{matrix}0\\\vdots\\0\end{matrix} \\ \alpha'_1 \ \ldots \ \alpha'_n & 1 \end{pmatrix} = \begin{pmatrix} Id_n & \begin{matrix}0\\\vdots\\0\end{matrix} \\ \alpha'_1 \ 0 \ \ldots \ 0 & 1 \end{pmatrix} \cdot \ldots \cdot \begin{pmatrix} Id_n & \begin{matrix}0\\\vdots\\0\end{matrix} \\ 0 \ \ldots \ 0 \ \alpha'_n & 1 \end{pmatrix}$$

Análogamente, para colocar elementos sobre la última columna nos valemos además de que $(B_1 \cdot B_n)^\intercal = B_n^\intercal \cdot B_1^\intercal$.

Falta entonces solamente considerar el caso en el que se desea que el elemento de más abajo a la derecha no sea 1. Para esto se deberá intercambiar la fila $n+1$-ésima con la i-ésima, haciendo que dicho 1 quede en lugar del β_i.

Debido a que para que la matriz resultante quede invertible, por lo menos uno de los elementos de cada columna debe valer 1; hace que por más que dicha permutación de filas obligue a que exista un uno en la última columna, termine no siendo esto una restricción adicional.

Dicha matriz de permutación (a premultiplicar a la resultante) podrá formarse mediante $P \cdot P^\intercal \cdot P$, donde $P : \mathcal{M}_{(n+1)\times(n+1)}(\mathbb{F}_2)$ es aquella con unos en la diagonal y en la posición $(i, n+1)$.

Una demostración alternativa se desprende de la utilización del algoritmo de escalerización gaussiana para escalerizar la matriz deseada, ya que el algoritmo no es más que la iteración de los pasos (visible como multiplicación de matrices):

- Suma de una combinación lineal de una fila j a la fila i. Representable como la premultiplicación por una matriz en la que hay unos solamente en la diagonal y en la posición i, j.
- Intercambio de dos filas i, j, representable mediante la premultiplicación por $L_{i,j} \cdot L_{i,j}^{\intercal} \cdot L_{i,j}$; equivalente a las siguientes transformaciones de las filas.
  - $r_i \leftarrow r_i \oplus r_j$
  - $r'_j \leftarrow r_j \oplus r'_i = r_j \oplus (r_i \oplus r_j) = r_i$
  - $r''_i \leftarrow r'_i \oplus r'_j = (r_i \oplus r_j) \oplus r_i = r_j$

□

Corolario 6.1. Al generalizar el teorema 6.3 sabemos que para toda transformación lineal T invertible, $\delta_I(f \circ T) = \delta_{\Pi(I)}(f)$ donde Π termina siendo una permutación determinable a partir de T, para lo cual se puede utilizar el algoritmo constructivo indicado en la prueba del teorema 6.4.

Cabe destacar que no todas las permutaciones se pueden lograr mediante la composición con una transformación lineal; ya que mientras que hay $(2^n - 1)!$ permutaciones posibles de los deltas de la clase de correlación que mantengan el peso, solo hay $\Pi_{i=0}^{n-1}(2^n - 2^i)$ transformaciones lineales binarias invertibles. Por ejemplo con n=4 son 1307674368000 las transformaciones, pero solo 20160 las permutaciones.

Teorema 6.5. Dada $T : \mathbb{F}_2^n \mapsto \mathbb{F}_2^n$ transformación lineal invertible, el conjunto de vectores en $\mathbb{F}_2^n$: $\{supp^{-1}(I)/\delta_i(f \circ T) = \delta_I(f)\ \forall i \in \{1, \ldots, n\}\}$ es linealmente independiente.

Demostración. Debido a que la composición de una función booleana con cualquier transformación lineal invertible es equivalente a la composición con múltiples transformaciones elementales $L_{i,j}$ cuyas matrices asociadas son de la forma (identidad + matriz de única entrada), es suficiente demostrar que se mantiene la independencia lineal tras la composición con una única matriz $L_{i,j}$ con $i \neq j$.

Basta que demostrar la independencia lineal del siguiente conjunto; para el cual expandimos $\delta_k(f \circ L_{i,j})$ según el teorema 6.3, recordando que $L_{i,j}$ es invertible por ser una involución:

$$\begin{aligned}
&\{supp^{-1}(I)/\delta_k(f \circ L_{i,j}) = \delta_I(f)\ \forall k \in \{1, \ldots, n\}\} \\
&= \{supp^{-1}(\{k\})/\forall k \in \{1, \ldots, n\}, k \neq i\} \cup \{supp^{-1}(\{i, j\})\} \\
&= \{u/u \text{ es fila de } (L_{i,j})\}.
\end{aligned}$$

□

De esto se desprende que existe tantas transformaciones lineales invertibles $(T) : \mathcal{M}_{n \times n}(\mathbb{F}_2)$ como posibles mapeos L.I. para los δ_i.

Teorema 6.6. Sea $L_{i,j} : \mathbb{F}_2^n \mapsto \mathbb{F}_2^n$ una transformación lineal elemental, o sea, cuya matriz asociada tiene unos en la diagonal y en la posición i, j.

$$\delta_I(f \circ L_{i,j}) = \delta_{supp(L_{j,i}(supp^{-1}(I)))}(f).$$

Demostración. Basándonos en el teorema 6.3, hay que probar $supp(L_{j,i}(supp^{-1}(I)))$ igual a I para $i \notin I$, e igual a $I \triangle \{j\}$ para $i \in I$. Desarrollando primero el caso general.

$$\begin{aligned} supp(L_{j,i}(supp^{-1}(I))) &= \{i'/L_{j,i}(supp^{-1}(I))_{i'} = 1 \forall i'\} \\ &= \left\{ i'/\bigoplus_{j'=1}^{n} (\mathbb{1}j' = i' \vee \langle i', j' \rangle = \langle j, i \rangle)(\mathbb{1}j' \in I) = 1 \ \forall i' \right\} \end{aligned}$$

Caso en que $i \notin I$.

$$\left\{ i'/\bigoplus_{j'=1}^{n} (\mathbb{1}j' = i')(\mathbb{1}j' \in I) = 1 \ \forall i' \neq i \right\} \ = \ \{i'/(\mathbb{1}i' \in I) = 1 \ \forall i' \neq i\} \ = \ I$$

Caso en que $i \in I$.

$$\begin{aligned} &\left\{ i'/\bigoplus_{j'=1}^{n} (\mathbb{1}j' = i')(\mathbb{1}j' \in I) = 1 \ \forall i' \right\} \triangle \left\{ i'/\bigoplus_{j'=1}^{n} (\mathbb{1}\langle i', j' \rangle = \langle j, i \rangle)(\mathbb{1}j' \in I) \ \forall i' \right\} \\ &= \{i'/(\mathbb{1}i' \in I) = 1 \ \forall i'\} \triangle \{i'/(\mathbb{1}i' = j)(\mathbb{1}i \in I) = 1 \ \forall i'\} \\ &= I \triangle \{i'/(\mathbb{1}i' = j) = 1 \ \forall i'\} \\ &= I \triangle \{j\} \end{aligned}$$

□

Teorema 6.7. Sea $T : \mathbb{F}_2^n \mapsto \mathbb{F}_2^n$ una transformación lineal invertible.

$$\delta_I(f \circ T) = \delta_{supp(T^\intercal \setminus supp^{-1}(I))}(f).$$

Afirmación también expresable mediante la transformada de Fourier:

$$\widehat{f \circ T} = \widehat{f} \circ T^{\intercal\,-1}.$$

Demostración. Se puede realizar mediante una demostración inductiva al descomponer T en transformaciones elementales T_i, en donde el paso base es el teorema 6.6 teniendo en mente que $L_{j,i} = L_{j,i}^{-1}$. Para la hipótesis inductiva se asume $\delta_I(f \circ T_1 \circ \ldots \circ T_{m-1}) = \delta_{supp((T_1 \circ \ldots \circ T_{m-1})^\intercal \setminus supp^{-1}(I))}(f)$, y se prueba para T:

$$\begin{aligned} \delta_I(f \circ T) &= \delta_I((f \circ T_1 \circ \ldots \circ T_{m-1}) \circ T_m) \\ &= \delta_{supp((T_1 \circ \ldots \circ T_{m-1})^\intercal \setminus supp^{-1}[supp(T_m^\intercal(supp^{-1}(I)))])}(f) \\ &= \delta_{supp((T_{m-1}^\intercal \circ \ldots \circ T_1^\intercal)^{-1}(T_m^\intercal)^{-1}(supp^{-1}(I)))}(f) \\ &= \delta_{supp((T_m^\intercal \circ T_{m-1}^\intercal \circ \ldots \circ T_1^\intercal) \setminus supp^{-1}(I))}(f) \\ &= \delta_{supp((T_1 \circ \ldots \circ T_m)^\intercal \setminus supp^{-1}(I))}(f) \\ &= \delta_{supp(T^\intercal \setminus supp^{-1}(I))}(f). \end{aligned}$$

□

6.3.3. Simetrías entre f y $\lambda x.(f \circ A + \vec{b} \cdot x + c)$

Ya en [2] se propuso la agrupación de funciones booleanas en clases de equivalencias según las biyecciones de la forma $g(x) = f \circ A + \vec{b} \cdot x + c$. Con A automorfismo afín, b vector en $\mathbb{F}_2^n$ y $c \in \mathbb{F}_2$.

Sin embargo al sumar a f una combinación lineal de x se pierde la propiedad de que todas las clases de equivalencias tengan el mismo peso. Por ejemplo (en dos variables) $f(x) = x_1 \oplus x_2$ tiene peso 2, sin embargo la función nula $g(x) = 0$ se encuentra en la misma clase: $g(x) = f(x) \oplus \vec{1} \cdot x)$. En este ejemplo tampoco se mantienen los valores de los deltas ($-2 = \delta_{\{1,2\}}(f) \neq \delta_{\{1,2\}}(g) = 0$).

7. Preservación de inmunidad a la correlación.

En este punto ya conocemos la importante propiedad que nos relaciona las transformadas de fourier de f y $f \circ T$ para f función booleana y T transformación lineal invertible; que recordamos: $\widehat{f \circ T}(u) = \hat{f}(T^{\intercal\,-1}(u))$.

Resulta entonces interesante saber cuáles son las transformaciones lineales invertibles tal que compuestas a una función booleana k-CI, nos dan una función también k-CI.

Definición 7.1 (preservación de k-CI)**.** Se dice que T preserva k-CI si para toda función booleana k-CI $f : \mathbb{F}_2^n \mapsto \mathbb{F}_2$, ocurre que $f \circ T$ es también k-CI.

Teorema 7.1. Las transformaciones lineales T invertibles que preservan k-CI, son aquellas que cumplen con $w_H(T^{\intercal\,-1}(u)) \leq k \ \forall u : w_H(u) \leq k$. En otras palabras, si miramos la matriz asociada de $(T^{\intercal\,-1})$ veremos que si hacemos el `XOR` de a lo sumo k columnas, el peso resultante tendrá que ser a lo sumo k.

Demostración. Comenzamos planteando las siguientes equivalencias (para el producto interno), que habrán de utilizarse más adelante:

$$\begin{aligned} T^{-1}y \cdot u =& u \cdot T^{-1}y = u^{\intercal}(T^{-1}y) = (u^{\intercal}T^{-1})y = \\ =& (T^{-1\intercal}(u))^{\intercal}y = (T^{-1\intercal}(u)) \cdot y = y \cdot T^{-1\intercal}(u) = y \cdot T^{\intercal\,-1}(u) \end{aligned}$$

Expandiendo la definición de transformada de fourier de $f \circ T$:

$$\begin{aligned} \widehat{f \circ T}(u) &= \sum_{x \in \mathbb{F}_2^n} f(T(x))(-1)^{x \cdot u} && \text{cambiando } x \leftrightarrow T^{-1}(y)\text{:} \\ &= \sum_{T^{-1}(y) \in \mathbb{F}_2^n} f(y)(-1)^{T^{-1}(y) \cdot u} && \text{y como } T \text{ es invertible:} \\ &= \sum_{y \in \mathbb{F}_2^n} f(y)(-1)^{T^{-1}(y) \cdot u} && \text{aplicando derivación previa:} \\ &= \sum_{y \in \mathbb{F}_2^n} f(y)(-1)^{y \cdot T^{\intercal\,-1}(u)} \end{aligned}$$

($\Rightarrow$) Si T preserva k-CI, $f \circ T$ es k-CI, esto significa que en la última parte de la ecuación anterior $T^{\intercal\,-1}(u)$ tendría que darnos pesos menores o iguales a k para que se cumpla que cualquier f sea también k-CI.

($\Leftarrow$) Tomando hipótesis $w_H(T^{\intercal\,-1}(u)) \leq k\ \forall u : w_H(u) \leq k$ y como f es k-CI (o sea que $\hat{f}(u) = 0$ para los $u\colon w_H(u) \leq k$), sabemos que la transformada de fourier de $f \circ T$ también vale cero para los $u\colon w_H(u) \leq k$, y por tanto $f \circ T$ es también k-CI; entonces T preserva k-CI. □

Corolario 7.1. Las transformaciones lineales T que conservan 1-CI son únicamente las matrices de permutación.

Corolario 7.2. Si T es una transformación lineal invertible que conserva k-CI, entonces también conserva k'-CI para todo $k' > k$.

Por ejemplo las T que conservan 1-CI también conservan 2-CI.

Teorema 7.2. La preservación de k-CI es cerrado ante la composición.

Demostración. Sean S y T transformaciones lineales invertibles que conservan k-CI. Para toda función booleana $f : \mathbb{F}_2^n \mapsto \mathbb{F}_2$ k-CI: $(f \circ S) \circ T = f \circ (S \circ T)$ es también k-CI, por tasto $S \circ T = S \cdot T$ preserva k-CI. □

7.1. Preservación de 2-CI.

Ejemplo 7.1. Una transformación lineal invertible que preserva 2-CI:

$$T = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix}, \quad T^{\intercal\,-1} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

Procederemos entonces a generalizar las matrices que preservan 2-CI.

Ejemplo 7.2. Presentamos a continuación una transformación lineal T invertible que preserva 2-CI, la cual en particular es una involución (igual a su inversa):

$$T = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 1 & 1 & 0 & \vdots \\ \vdots & 0 & \ddots & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}$$

Demostración de preservación de 2-CI del ejemplo 7.2. Si consideramos una función booleana f, y $g = f \circ T$, observaremos que $g(x_1 \dots x_n) = f(T(x_1 \dots x_n)) = f(x_1, x_1 + x_2, \dots, x_1 + x_n)$.

Supongamos que f es 2-CI, o sea $\hat{f}(u) = 0 \forall u : 1 \leq w_H(u) \leq 2$. Sabemos que $\hat{g} = \hat{f}(T^{\intercal\,-1}(u))$, y que en este caso es igual a $\hat{f}(T^{\intercal}(u))$ (por ser una involución), donde T transpuesta es:

$$T^{\intercal} = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 0 & 1 & 0 & 0 \\ \vdots & 0 & \ddots & 0 \\ 0 & \cdots & 0 & 1 \end{pmatrix}$$

Obteniéndose: $\hat{g}(u_1 \dots u_n) = \hat{f}(u_1 + \dots + u_n,\ u_2, \dots, u_n)$.

Para $u = (1, 0 \dots 0)$ queda $\hat{f}(1, 0 \dots 0)$ también cero.

Para u de peso 1 en otra posición queda $\hat{f}(1, 0 \dots 0, 1, 0 \dots 0) = 0$ (por 2-CI).

Para u de peso 2, sabemos que como la primer entrada de $T^\intercal(u)$ es la suma de los bits de u, al ser u de peso par, se obtiene 0. Las otras entradas de u se mantienen y por tanto el peso de $T^\intercal(u) \leq 2$.

En particular, con u de peso 2 y $u_1 = 1$, queda $\hat{f}(0, 0 \dots 0, 1, 0 \dots 0)$, y el caso restante $u_1 = 0$ con $w_H(u) = 2$, da $T^\intercal(u) = u$.

Por tanto T conserva 2-CI. □

Nótese que las matrices de permutación compuestas con T, también preservan 2-CI, debido a que: la preservación de k-CI es cerrada ante la composición, y las permutaciones al preservar 1-CI, también preservan 2-CI.

Construyamos entonces las transformaciones lineales invertibles que preservan 2-CI (exceptuando las de permutación, que sabemos preservan 1-CI), pensando en ${T^\intercal}^{-1}$:

- todas las columnas deben tener peso 1 o 2; mayor que cero para que sea invertible y a lo sumo 2 por condición necesaria para preservar 2-CI.
- Si todas las columnas tuviesen peso 1, llegaríamos a una matriz de permutación, así que por lo menos una columna tiene peso 2.
- Dadas dos columnas de peso 1 (con entradas en 1 en i y j con $i \neq j$ (para que sea invertibe)) y una de peso 2, sabemos que es necesario que las entradas en la de peso 2 correspondan con i y j, ya que sino al hacer `XOR` entre ellas ebtendríamos peso 3 (no preservando 2-CI). Entonces la columna de peso 2 sería una combinación lineal de las anteriores, haciendo que la transformación sea no invertible (absurdo),
- por tanto no puede haber más de una columna con peso 1, y esta tendría que compartir un bit con todas las de peso 2 (el soporte de la de peso 1 sería subconjunto del soporte de todas las de peso 2).
- Por otro lado, dado un par cualquiera de columnas de peso 2, la distancia entre ambas puede ser 0, 2 o 4, si las columnas fuesen iguales (distancia cero) no sería invertible la transformación; si la distancia fuese 4, no preservaría 2-CI (recordamos que la distancia de dos columnas es el `XOR` de ambas); por lo que tienen que tener un bit en 1 en común. Generalizando esto para todas las columnas de peso 2, sabemos que tendrían que compartir esta entrada en 1.
- Si hubiese n columnas de peso 2 compartiendo un bit pero con el otro bit repartido entre las otras n-1 filas, por el principio del palomar habrían dos columnas repetidas (absurdo).

Resultando en que toda transformación lineal invertible que preserve k-CI tendría que: tener una columna de peso 1 y n-1 columnas de peso 2; tener una fila en 1 (correspondiendo a la posición del bit de la columna de peso 1).

Como podemos ver, todas estas transformaciones son ni más ni menos que una permutación de la transformación del ejemplo 7.2, que habíamos visto preservaba 2-CI. De esta forma logramos caracterizarlas.

8. Clases normales en base a permutaciones y traslaciones.

El objetivo de trabajar con clases normales en lugar de trabajar directamente con todas las clases, es reducir el costo de almacenamiento y procesamiento de los algoritmos aquí mencionados para la construcción enumerativa de funciones booleanas k-CI. Como se verá más adelante, para cualquier clase ω es posible encontrar su clase normal $N(\omega)$, así como encontrar todas las clases con la misma normal: $\{\omega'/N(\omega')=N(\omega)\ \forall\omega'\in\Omega^{[k]}\}$

Entonces se nos presentan inmediatamente dos enfoques posibles para trabajar con clases normales de orden k:

- Un primer enfoque, el trabajado en esta sección, es el de: 1. almacenar en la clase de correlación de orden k únicamente los deltas de peso menor o igual que k; 2. dar una relación de equivalencia que permita separar las clases de correlación en clases más pequeñas, utilizando en este caso la existencia de biyecciones en la forma de permutaciones y traslaciones; 3. dar un orden total bien definido para estas clases de correlación de forma tal que alcance con decir que el mínimo es la clase normal.

- El segundo enfoque se vale, no de solamente las permutaciones y las traslaciones, sino en general de los automorfismos afines, almacenando como "clase de correlación" la transformada de Fourier entera de la función. Sin embargo al separar estas transformadas de Fourier en clases de correlación mediante la existencia de un automorfismo afín, es posible reducir significativamente la cantidad de clases a almacenar.

  El considerar automorfismos afines en general, acarrea dos problemas a solucionar:

  - Las "clases de correlaciones" al ser las transformadas de Fourier, terminan siendo una por cada función booleana, lo cual ya sabemos es un gran número.
  - Las afinidades que no cambian el peso de la función son solamente las permutaciones con traslaciones; de forma tal que si buscamos encontrar funciones de peso mínimo, se deberá pensar nuevos algoritmos.

Sin dudas esta segunda alternativa, nos abre un amplio panorama a investigar; aunque por sencillez nos resulta conveniente comenzar en esta sección por primera.

Definición 8.1. (*clase normal para k=1.*) Una clase $\omega = \langle w_H, \delta_n, \ldots, \delta_1\rangle$ es normal cuando $0 \leq \delta_n \leq \delta_{n-1} \leq \ldots \leq \delta_1$.

Se puede observar cómo para toda función es posible obtener una composición de transformaciones (tomando únicamente traslaciones y permutaciones de parámetros) de forma tal que la clase de correlación de la función transformada sea normal; buscándose que en definitiva, las clases normales sean un subconjunto representativo que nos permita mantener las mismas propiedades de las clases de correlaciones.

Definición 8.2. (*clase normal de orden k*) Una clase $\omega = \Omega^{[k]}(f)$ es normal si $\omega = max_{\geq}\{\Pi_\pi(M_I(\omega))\ \forall$ permutación π de $\{1,\ldots,n\}\ \forall I \subseteq \{1,\ldots,n\}\}$, siendo:

- $\Pi_\pi(\Omega^{[k]}(f))$, donde llamando α_I a los componentes de la clase devuelta, se define $\alpha_I = \delta_{\{\pi(i)\forall i \in I\}}(f)$,
- M_I es definido para clases de correlaciones de orden k; donde dado $\omega = \Omega^{[k]}(f)$, $M_I(\omega) = \langle w_H, \alpha_n, \ldots, \alpha_1, \alpha_{n,n-1}, \ldots, \alpha_{2,1}, \ldots\rangle$ con $\alpha_J = -\delta_J(f)$ si $|I \cap J|$ es impar, mientras que $\alpha_J = \delta_J(f)$ si $|I \cap J|$ es par, para todo $J \subseteq \{1,\ldots,n\}\ /\ |J| \leq k$,
- el orden total estricto $\geq$ para clases de correlaciones de mismo orden, donde dadas las clases ω y ω' con componentes δ_I y δ'_I respectivamente, se define $\omega = \omega'$ si $\delta_I = \delta'_I \forall I$, y

$$\omega > \omega' \text{ sii } \exists I: \quad \begin{array}{l} \forall J.\ (|J| < |I| \vee |J| = |I| \wedge J >_L I) \Rightarrow \delta_J = \delta'_J, \\ |\delta_I| > |\delta'_I| \ \vee \ |\delta_I| = |\delta'_I| \wedge \delta_I > \delta'_I \end{array}$$

Al considerar los δ_I con $|I| = 1$ en una clase normal de orden k, ubicados después del peso de la función, se puede observar que se encuentran en orden ascendente, siendo por tanto esta definición una generalización de las de primer orden.

Ejemplo 8.1. Se presenta a continuación una la función de tres variables: $f = x_1x_2x_3 \oplus x_1 \oplus x_2 \oplus x_3$, la cual tiene como clase de correlación de orden 3: $\langle 3,\ 1,1,1,\ -1,-1,-1,\ -3\rangle$, en donde los deltas de peso 1 son todos 1 mientras que los de dos son todos -1.

Podemos ver también como todas las permutaciones de los subíndices de las variables de f dan la misma función, mientras que las transformaciones M_I solo producen cambios de signos de formas tales que salvo $M_\varnothing$ (que no realiza ningún cambio) todas dejan algún delta de una variable negativo.

8.1. Algoritmos para clases normales

Dada f función booleana de n variables, y ω su clase de correlación de orden k, podemos utilizar el siguiente algoritmo para normalizarla: $N(\omega)$.

La primera parte ordena los deltas según el valor absoluto.

Iterando $k' = 1 \ldots k$:

- Para cada I tal que $|I| = k'$, iterando en orden decreciente según $<_L$ (de derecha a izquierda):
  - Elegir el J con mayor $|\delta_J(f)|$ tal que:
    - $|J| = k'$,
    - $J <_L I$ (que esté a la izquierda de I),
    - $\exists\, \pi = (a_1\ b_1)(a_2\ b_2)\ldots(a_d\ b_d)$ tal que $\{a_1,\ldots,a_d\} = I \backslash J$, $\{b_1,\ldots,b_d\} = J \backslash I$ y $|I \backslash J| = |J \backslash I| = d$,
    - $\delta_K(\Pi(f)) = \delta_K(f)\ \ \forall K\ /\ |K| < k' \vee |K| = k' \wedge K >_L I$, donde $\Pi(f)(x_1,\ldots,x_n) = f(x_{\pi(1)},\ldots,x_{\pi(n)})$.
  - si existe dicha J y $|\delta_J(f)| > |\delta_I(f)|$, reasignar $f := \Pi(f)$.

La segunda parte del algoritmo se encarga de ajustar los signos de la clase de correlación tratando de dejar positivos los deltas con índice de menor peso, y dentro de los de mismo peso le da prioridad a aquellos que se encuentran a la derecha.

$S :=$ sistema de ecuaciones $\{m_i = 0/\delta_i(f) \neq 0\}$,

$f := M_{\{i/\delta_{\{i\}}(f)<0\}}(f)$,

Iterando $k' = 2, \ldots, k$:

- Para cada I tal que $|I| = k'$, iterando en orden decreciente según $<_L$ (de derecha a izquierda):
  - Si $\delta_I(f) > 0$ redefinir $S := S \cup \big(0 = \bigoplus_{i\in I} m_i\big)$.
  - Si $\delta_I(f) < 0$, y $S \cup \big(1 = \bigoplus_{i\in I} m_i\big)$ es un sistema compatible, redefinir $S := S \cup \big(1 = \bigoplus_{i\in I} m_i\big)$.
  - Si el sistema S queda determinado ($FV(S) = \emptyset$), terminar las iteraciones.

Finalmente $f := M_{\{i/m_i=1\}}(f)$ donde el valor de los m_i queda determinado por el sistema S, eligiéndose $m_i = 0$ para aquellos que hubiesen quedado libres.

Nótese que los únicos casos en que quedan variables libres en el sistema es cuando los deltas de la función booleana tienen ceros en posiciones tales que trasladando la función en dichas variables se obtiene una misma función con la misma clase de orden k.

Para obtener todas las clases de correlaciones que normalizadas dan una cierta clase dada θ, se itera sobre todas las permutaciones variando los signos de aquellas variables que no hubiesen quedado libres en S.

9. Conclusiones.

En base a que el problema de entender desde un punto de vista combinatorio las funciones booleanas inmunes a la correlación me pareció muy interesante es que comencé este trabajo.

Del amplio espectro de problemas al respecto me interesó particularmente entender las simetrías y en este tema nos concentramos al respecto (no empezamos) en este marco tiene que leer bastante materila y notamos que hay muchos aspectos importantes de investigación científica en problemas que no son muy bien comprendidos.

En este sentido la mayor parte del tiempo dedicado al proyecto se asignó a tratar de entender estas simetrías, a formalizarlas adecuadamente y a pensar como aplicar estas ideas en mejorar la eficiencia de los algoritmos de búsqueda y generación uniforme y aleatoria de funciones booleanas inmunes a la correlación de menor peso.

La visión combinatoria del problema es novedosa en muchos aspectos fundamentales, y por eso llevó bastante tiempo entenderlas y llevar a cabo pruebas de concepto que permitiesen hacer comprobaciones empíricas.

Específicamente en este proyecto considero que las mayores contribuciones originales son: la fórmula para la transformada de Fourier de la composición

de una función booleana con una afinidad; la caracterización de los automorfismos afines que preservan inmunidad de la correlación; y el hecho de que la investigación de simetrías generales abriese el camino al estudio de algoritmos combinatorios que mediante la existencia de automorfismos afines, permita la separación en clases de equivalencias.

Estas contribuciones permiten avanzar en el estudio de nuevos algoritmos de generación y búsqueda de funciones booleanas inmunes a la correlación de orden k.

Como en todo trabajo de investigación, quedan quedan por tanto aspectos sin resolver:

- La integración entre las clases de equivalencias expuestas en este trabajo y las simetrías presentadas en [1] que permitirían construir directamente funciones de n+1 argumentos $2k+1$-CI desde funciones de n argumentos $2k - CI$.
- El estudio respecto a representaciones eficientes de las transformadas de fourier de funciones booleanas en general, y las normales presentadas en este trabajo en particular.
- La optimización de los algoritmos de normalización de clases de equivalencias de funciones booleanas.

Referencias

[1] Jean-Marie Le Bars and Alfredo Viola. Correlation-immune functions of order k with minimal hamming weight. 2016.

[2] E Berlekamp and L Welch. Weight distributions of the cosets of the (32, 6) reed-muller code. *IEEE Transactions on Information Theory*, 18(1):203–207, 1972.

[3] Claude Bhasin, Shivam; Carlet and Sylvain Guilley. Theory of masking with codewords in hardware: low-weight dth-order correlation-immune boolean functions. *IACR Cryptology ePrint Archive*, 2014:303, 2014.

[4] E Rodney Canfield, Zhicheng Gao, Catherine Greenhill, Brendan D McKay, and Robert W Robinson. Asymptotic enumeration of correlation-immune boolean functions. *Cryptography and Communications*, 2(1):111–126, 2010.

[5] Claude Carlet. Emerging applications of finite fields: Correlation-immune Boolean functions and counter-measures to side channel attacks. `https://www.ricam.oeaw.ac.at/specsem/specsem2013/workshop4/slides/carlet.pdf`.

[6] Claude Carlet. Boolean functions for cryptography and error correcting codes. *Boolean models and methods in mathematics, computer science, and engineering*, 2:257, 2010.

[7] Claude Carlet. Vectorial boolean functions for cryptography. *Boolean models and methods in mathematics, computer science, and engineering*, 134:398–469, 2010.

[8] Nicolas Carrasco, Jean-Marie Le Bars, and Alfredo Viola. Enumerative encoding of correlation immune boolean functions. In *Information Theory Workshop (ITW), 2011 IEEE*, pages 643–647. IEEE, 2011.

[9] D.E. Knuth. *Art of Computer Programming, Volume 2: Seminumerical Algorithms*. Pearson Education, 2014.

[10] Jean-Marie Le Bars and Alfredo Viola. Equivalence classes of boolean functions for first-order correlation. *IEEE Transactions on Information Theory*, 56(3):1247–1261, 2010.

[11] James L Massey. Cryptography and system theory. In *Proceedings of the 24th Allerton Conference on Communication, Control, and Computers*, pages 1–8, 1986.

[12] Enes Pasalic, Subhamoy Maitra, Thomas Johansson, and Palash Sarkar. New constructions of resilient and correlation immune boolean functions achieving upper bound on nonlinearity. *Electronic Notes in Discrete Mathematics*, 6:158–167, 2001.

[13] Sebastián Fonseca, Cecilia García. Generación aleatoria de funciones inmunes a la correlación de menor peso.

[14] J. von zur Gathen. *CryptoSchool*. Springer Berlin Heidelberg, 2015.

[15] G-Z Xiao and James L Massey. A spectral characterization of correlation-immune combining functions. *IEEE Transactions on information theory*, 34(3):569–571, 1988.

Apéndices.

A. Simetrías para clases de correlación en primer orden.

Se agrega este apéndice por completitud; sin embargo el resultado aquí expuesto ya era conocido y demostrado en el artículo de Carrasco [8].

Con el fin de mantener consistencia entre la definición de $\delta_i(f)$ utilizada a lo largo de este texto (expresado como la evaluación de $\hat{f}$), con la definición alternativa dada en [6] y [8] (expresado como diferencia de pesos de hamming), se presenta a continuación una pequeña prueba de la equivalencia de ambas.

Teorema A.1 (elementos de las clases como peso de hammings)**.** Dada una función booleana f de n variables con clase de correlación de primer orden $\langle w_H(f), \delta_n(f), \ldots, \delta_1(f)\rangle$. Se cumple $\delta_i(f) = w_H(f|_{x_i=0}) - w_H(f|_{x_i=1})$.

Demostración.

$$\begin{aligned}
\delta_i(f) &= \hat{f}(u) \text{ con } u \in \mathbb{F}_2^n / u_j = \mathbb{1}_{i=j} \\
&= \sum_{x \in \mathbb{F}_2^n} f(x)(-1)^{x \cdot u} \\
&= \sum_{x \in \mathbb{F}_2^n / x_i = 0} f(x) - \sum_{x \in \mathbb{F}_2^n / x_i = 1} f(x) \\
&= \sum_{x \in \mathbb{F}_2^{(}n-1)} (f|_{x_i=0})(x) - \sum_{x \in \mathbb{F}_2^{(}n-1)} (f|_{x_i=1})(x) \\
&= w_H(f|_{x_i=0}) - w_H(f|_{x_i=1})
\end{aligned}$$

□

Teorema A.2 (permutación de variables para primer orden)**.** El intercambio de x_i por x_j en f, cambia los valores de δ_i por δ_j.

Dados:

- $i, j \in \{1, \ldots, n\}$ con $i < j$;
- $\pi_{i,j}$ la permutación de i por j [3];
- la sobrecarga de la definición de π también para la permutación de los argumentos en funciones n-arias $\pi_{i,j}(f)(x_1, \ldots, x_n) = f(x_{\pi_{i,j}(n)}, \ldots, x_{\pi_{i,j}(1)})$;
- definiendo además las permutaciones para la clases de correlaciones $\Pi_{i,j}(\Omega(f)) = \langle w_H(f), \delta_{\pi_{i,j}(n)}(f), \ldots, \delta_{\pi_{i,j}(1)}(f) \rangle$.

Se cumple que $\Pi_{i,j}(\Omega(f)) = \Omega(\pi_{i,j}(f))$.

O dicho de otra forma: $\delta_{\pi_{i,j}(k)}(f) = \delta_k(f \circ P)$, donde $P : \mathbb{F}_2^n \mapsto \mathbb{F}_2^n$ es la matriz de permutación que cambia x_i por x_j.

Demostración. Expandiendo las definiciones de Π y Ω, el teorema anterior es equivalente a probar que $\forall\ k \in \{1, \ldots, n\}$ se cumple $\delta_{\pi_{i,j}(k)}(f) = \delta_k(\pi_{i,j}(f))$.

Primero tomando el caso para todo $k \notin \{i, j\}$:

$$\begin{aligned}
\delta_{\pi_{i,j}(k)}(f) &= \delta_k(f) \\
&= w_H(f|_{x_k=0}) - w_H(f|_{x_k=1}) \\
&= w_H(\pi_{i,j}(f|_{x_k=0})) - w_H(\pi_{i,j}(f|_{x_k=1})) \\
&= w_H(\pi_{i,j}(f)|_{x_k=0}) - w_H(\pi_{i,j}(f)|_{x_k=1}) \\
&= \delta_k(\pi_{i,j}(f))
\end{aligned}$$

Siguiendo el mismo razonamiento, para $k = i$ tenemos:

$$\begin{aligned}
\delta_{\pi_{i,j}(i)}(f) = \delta_j(f) &= w_H(f|_{x_j=0}) - w_H(f|_{x_j=1}) \\
&= w_H(\pi_{i,j}(f|_{x_j=0})) - w_H(\pi_{i,j}(f|_{x_j=1})) \\
&= w_H(\pi_{i,j}(f)|_{x_i=0}) - w_H(\pi_{i,j}(f)|_{x_i=1}) \\
&= \delta_i(\pi_{i,j}(f))
\end{aligned}$$

Siendo análogo el restante caso para $k = j$. □

[3] Se define $\pi_{i,j}(k) = k$ si $k \notin \{i, j\}$; $\pi_{i,j}(i) = j$, y $\pi_{i,j}(j) = i$

Teorema A.3 (traslación para primer orden)**.** El negar el valor de x_i en f, implica el cambio de signo en δ_i.

Dados:

- $i \in \{1, \ldots, n\}$;
- $m_i(f)(x_1, \ldots, x_n) = f(y_1, \ldots, y_n)$, donde $y_i = \bar{x_i}$ e $y_j = x_j$ para los j distintos de i;
- y la función M_i (que cambia el signo de δ_i) definida sobre clases de correlaciones de primer orden como: $M_i(\langle w_H, \delta_n, \ldots, \delta_1 \rangle) = \langle w_H, \alpha_n, \ldots, \alpha_1 \rangle$, con $\alpha_i = -\delta_i$ y $\alpha_j = \delta_j$ para $j \neq i$.

Se cumple $M_i(\Omega(f)) = \Omega(m_i(f))$.

O dicho de otra forma: sea $u \in \mathbb{F}_2^n$ el vector que solo tiene prendido el bit i:

$$\delta_k(f \circ (\lambda x.\ x \oplus u)) = \begin{cases} -\delta_i(f) & \text{si } k = i \\ \delta_i(f) & \text{en otro caso} \end{cases}$$

Demostración. Nuevamente, tras expandir las definiciones de M_i y Ω probar el teorema anterior es equivalente a probar las identidades $-\delta_i(f) = \delta_i(m_i(f))$ y $\delta_j(f) = \delta_j(m_i(f))$ para $i \neq j$.

Para el caso en que $j \in \{1, \ldots, n\} \backslash \{i\}$, tenemos:

$$\begin{aligned} \delta_j(m_i(f)) &= w_H(m_i(f)|_{x_j=0}) - w_H(m_i(f)|_{x_j=1}) \\ &= w_H(m_i(f|_{x_j=0})) - w_H(m_i(f|_{x_j=1})) \\ &= w_H(f|_{x_j=0}) - w_H(f|_{x_j=1}) \\ &= \delta_j(f) \end{aligned}$$

Para la otra identidad:

$$\begin{aligned} \delta_i(m_i(f)) &= w_H(m_i(f)|_{x_i=0}) - w_H(m_i(f)|_{x_i=1}) \\ &= w_H(m_i(f|_{x_i=1})) - w_H(m_i(f|_{x_i=0})) \\ &= w_H(f|_{x_i=1}) - w_H(f|_{x_i=0}) \\ &= -(w_H(f|_{x_i=0}) - w_H(f|_{x_i=1})) \\ &= -\delta_i(f) \end{aligned}$$

□

B. Demostraciones de simetrías para clases de correlación en orden k.

Mientras que en 3.1 se introdujo la relación entre que una función sea k-CI y que su transformada de Fourier valga cero en los elementos de peso entre 1 y k, acá generalizamos aún más dicho teorema dando una equivalencia de las dos definiciones posibles para $\delta_I(f)$, permitiéndonos trabajar tranquilamente con cualquiera de las dos:

- Como sumas de los pesos de Hamming de las restricciones de las funciones, tal como se presenta en [10]; y la menos engorrosa:

- Como el valor de $\hat{f}(u)$ donde $I = supp(u)$.

Teorema B.1. Dado $I \subseteq \{1, \ldots, n\}$, sea $I = supp(u)$, E un espacio vectorial sobre $\mathbb{F}_2^n$, tal que $E = \{x / supp(x) \subseteq I\}$; entonces:

$$\delta_I(f) = \sum_{y \in E} w_H(f | proy_E(x) = y)(-1)^{y \cdot u}.$$

Demostración.

$$\begin{aligned}
\delta_I(f) = \hat{f}(u) &= \sum_{x \in \mathbb{F}_2^n} f(x)(-1)^{x \cdot u} && \text{al reescribir x=y⊕z} \\
&= \sum_{y \in E, z \in E^\perp} f(y \oplus z)(-1)^{(y \oplus z) \cdot u} && z \in E^\perp \Rightarrow z \cdot u = 0 \\
&= \sum_{y \in E, z \in E^\perp} f(y \oplus z)(-1)^{y \cdot u} && \text{factor común y} \\
&= \sum_{y \in E} \left(\sum_{z \in E^\perp} f(y \oplus z) \right) (-1)^{y \cdot u} && \text{def. de peso de proy.} \\
&= \sum_{y \in E} w_H(f | proy_E(x) = y)(-1)^{y \cdot u}
\end{aligned}$$

□

Reiterando el teorema 6.2, se afirma: $\Pi_{i,j}(\Omega^{[k]}(f)) = \Omega^{[k]}(\pi_{i,j}(f))$.

Demostración. Expandiendo las definiciones de Ω, Π, queda equivalente a probar que $\delta_{\pi_{i,j}(I)}(f) = \delta_I \pi_{i,j}(f)$, al sustituir también por la definición deδ, queda: $\hat{f}(\pi_{i,j}(u)) = \widehat{\pi_{i,j}(f)}(u)$. Expandiendo además la transformada de Fourier sería necesario probar que $\sum_{x \in \mathbb{F}_2^n} f(x)(-1)^{x \cdot \pi_{i,j}(u)} = \sum_{x \in \mathbb{F}_2^n} \pi_{i,j}(f)(x))(-1)^{x \cdot u}$ para $I = supp(u)$:

$$\begin{aligned}
\hat{f}(\pi_{i,j}(u)) &= \sum_{x \in \mathbb{F}_2^n} f(x)(-1)^{x \cdot \pi_{i,j}(u)} && \text{por def. de } \hat{f} \\
&= \sum_{x \in \mathbb{F}_2^n} f(x)(-1)^{\pi_{i,j}(x) \cdot u} && (\divideontimes) \\
&= \sum_{x \in \mathbb{F}_2^n} f(\pi_{i,j}(\pi_{i,j}(x)))(-1)^{\pi_{i,j}(x) \cdot u} && \pi_{i,j} \text{ es una involución} \\
&= \sum_{x \in \mathbb{F}_2^n} \pi_{i,j}(f)(\pi_{i,j}(x)))(-1)^{\pi_{i,j}(x) \cdot u} && \text{por def. de } \pi_{i,j}(f) \\
&= \sum_{\pi_{i,j}(x) \in \mathbb{F}_2^n} \pi_{i,j}(f)(\pi_{i,j}(x)))(-1)^{\pi_{i,j}(x) \cdot u} && \pi_{i,j} \text{ es una biyección} \\
&= \sum_{x \in \mathbb{F}_2^n} \pi_{i,j}(f)(x))(-1)^{x \cdot u} = \widehat{\pi_{i,j}(f)}(u) && \text{cambio } x \leftrightarrow \pi_{i,j}(x)
\end{aligned}$$

La equivalencia utilizada en (※) se muestra a continuación, mediante el desarrollo de la definición de producto escalar sobre $\mathbb{F}_2$:

$$\begin{aligned}
x \cdot \pi_{i,j}(u) &= \bigoplus_{i'=1}^{n} \mathbb{1}(x_{i'} = 1) \wedge (\pi_{i,j}(u)_{i'} = 1) \\
&= \bigoplus_{i'=1}^{n} \mathbb{1}(x_{\pi_{i,j}(\pi_{i,j}(i'))} = 1) \wedge (\pi_{i,j}(u)_{i'} = 1) \\
&= \bigoplus_{i'=1}^{n} \mathbb{1}(\pi_{i,j}(x)_{\pi_{i,j}(i')} = 1) \wedge (u_{\pi_{i,j}(i')} = 1) \\
&= \bigoplus_{\pi_{i,j}(i')=1}^{n} \mathbb{1}(\pi_{i,j}(x)_{i'} = 1) \wedge (u_{i'} = 1) \\
&= \bigoplus_{i'=1}^{n} \mathbb{1}(\pi_{i,j}(x)_{i'} = 1) \wedge (u_{i'} = 1) \\
&= \pi_{i,j}(x) \cdot u
\end{aligned}$$

□

Se coloca aquí también la demostración del teorema 6.1, el cual nos indica que al negar la entrada x_i de la función, los δ_I cambian de signo en las posiciones I tales que $i \in I$.

Demostración. Comenzamos definiendo $I = supp(u)$, expandiendo la transformada de Fourier y haciendo el cambio de variables $x \leftrightarrow m_i(x)$.

$$\begin{aligned}
\delta_I(m_i(f)) = \widehat{m_i(f)}(u) &= \sum_{x \in \mathbb{F}_2^n} m_i(f)(x)(-1)^{x \cdot u} \\
&= \sum_{m_i(x) \in \mathbb{F}_2^n} m_i(f)(m_i(x))(-1)^{m_i(x) \cdot u}
\end{aligned}$$

como m_i es una involución ($m_i(m_i(x)) = x$) sobre $\mathbb{F}_2^n$, es biyectiva y por tanto $\mathbb{F}_2^n = \{m_i(x)/x \in \mathbb{F}_2^n\}$,

$$\begin{aligned}
\delta_I(m_i(f)) &= \sum_{x \in \mathbb{F}_2^n} m_i(f)(m_i(x))(-1)^{m_i(x) \cdot u} \\
&= \sum_{x \in \mathbb{F}_2^n} f(m_i(m_i(x)))(-1)^{m_i(x) \cdot u} \\
&= \sum_{x \in \mathbb{F}_2^n} f(x)(-1)^{m_i(x) \cdot u}
\end{aligned}$$

debido a que el producto escalar $m_i(x) \cdot u$ se define como $\bigoplus_{j=1}^{n} \mathbb{1}m_i(x)_j = 1 \wedge u_j = 1$, es que separamos en dos casos: si $i \in I$ sabemos que solo cuando $j = i$ la función m_i cambia dicho bit, cambiando el resultado de la indicatriz (porque $u_i = 1$), dando:

$$\delta_i(m_i(f)) = \sum_{x \in \mathbb{F}_2^n} f(x)(-1)^{x \cdot u \oplus 1} = -\sum_{x \in \mathbb{F}_2^n} f(x)(-1)^{x \cdot u} = -\delta_i(f),$$

sin embargo cuando $i \notin I$, al m_i cambiar el bit i, dicho bit es ignorado al ser $u_i = 0$ (por no estar i en el soporte de u), dando:

$$\delta_i(m_i(f)) = \sum_{x \in \mathbb{F}_2^n} f(x)(-1)^{x \cdot u} = \delta_i(f).$$

□

C. Notación

$\mathbb{F}_2^n$	campo de Galois de tamaño 2^n, en este contexto visto como un vector de n bits (ceros o unos).
f^0	restricción de la función f, $f^0(x_1, \ldots, x_{n-1}) = f(x_1, \ldots, x_{n-1}, 0)$.
f^1	restricción de la función f, $f^1(x_1, \ldots, x_{n-1}) = f(x_1, \ldots, x_{n-1}, 1)$.
$\Omega(f)$	clase de correlación de la función f de primer orden.
$\Omega^{[k]}(f)$	clase de correlación de la función f de orden k.
$\delta_i(f)$	diferencia de los pesos de las restricciones de f a x_i, $\delta_{\{i\}}(f)$.
$\delta_I(f)$	generalización en donde se toma las restricciones según la paridad de los valores de $x_i / i \in I$, definido como $\hat{f}(u)$, con $I = supp(u)$.
$\langle i \rangle_n$	vector en $\mathbb{F}_2^n$ tal que $i = \sum\{2^{j-1} \ / \ u_j = 1\}$, donde $u = \langle i \rangle_n$; se asume largo n si este se llegase a omitir.
ℓ_u	la función afín que dada por el vector u, queda definida como $\ell_u(x) = \sum_{i \in \{1 \ldots n\}} u_i x_i$
$\hat{f}$	transformada de Fourier de la función f.
$supp(u)$	función que devuelve el conjunto de enteros de los índices en que u vale 1; $supp(u) : \mathbb{F}_2^n \mapsto \{1, \ldots, n\} = \{i / u_i \neq 0\}$
$f\|_{x_i=c}$	restricción de la función f, pudiendo verse como una aplicación parcial en la que el resultado es una función con un argumento menos. $(f\|_{x_i=c})(x_1, \ldots, x_{n-1}) \stackrel{def}{=} f(x_1, \ldots, x_{i-1}, c, x_i, \ldots, x_{n-1})$.
$T \backslash b$	solución del sistema lineal, equivalente a $T^{-1}b$.
$A \backslash B$	resta de conjuntos, equivalente a $A \cap B^{\complement}$.
$A \triangle B$	or exclusivo de conjuntos, equivalente a $(A \cup B) \backslash (A \cap B)$.