Construcción de un Generador Aleatorio Binario mediante Combinación No Lineal de LFSR

Germán Montejano[1], Pablo García[2], AndrésFarías.[3]

[1] FCFMyN (UNSL) – Ejercito de Los Andes 50, San Luis,
gmonte@unsl.edu.ar
[2] FCEyN (UNLPam) – Uruguay 151, Santa Rosa, La Pampa
pablogarcia@exactas.unlpam.edu.ar
[3] DCE (Unlar) – Av. De La Fuente S/N - La Rioja
afarias@yahoo.com.ar

Abstract. El objetivo de este documento es mostrar los primeros avances en el desarrollo de un software generador de bits, a partir de la combinación no lineal de generadores Geffe, que produzca secuencias de alto nivel de aleatoriedad. Dicha combinación persigue la finalidad de romper la linealidad de los registros; para ello se utiliza una función booleana que cumple con una serie de requisitos matemáticos para asegurar las mejores prestaciones criptográficas. Al final del proceso se debe someter a diversas pruebas de aleatoriedad a la secuencia obtenida, para verificar que el generador realmente está entregando los resultados aleatorios esperados, para lo cual se recurre a herramientas estadísticas.

Palabras Claves: Generación de Bits Aleatorios, LFSR, Geffe, Prueba de Aleatoriedad, Complejidad Lineal.

1 Generador Aleatorio mediante Combinación No Lineal de LFSR

1.1 Descripción del Generador Aleatorio

El generador propuesto en este trabajo, está conformado por tres generadores Geffe, cuyas secuencias son combinadas de forma no lineal mediante una función booleana de tres variables $f_{booleana} = f_{booleana}(x, y, z)$. El esquema del generador se indica en la figura 1.

Para esta propuesta se analizan distintas funciones booleanas que son obtenidas mediante un procedimiento de búsqueda.

Las variables x, y, z de la función son las secuencias obtenidas de cada generador Geffe $x = S_{Geffe_1} = S_x$; $y = S_{Geffe_2} = S_y$; $z = S_{Geffe_3} = S_z$.

El propósito de combinar las secuencias S_{Geffe_1}, S_{Geffe_2}, S_{Geffe_3}, obtenidas por cada generador Geffe, consiste en lograr una secuencia final $w = S_{Generador}$ con un período y una complejidad lineal muchos mayores de las que se obtendría con cada

uno de ellos trabajando de manera individual.

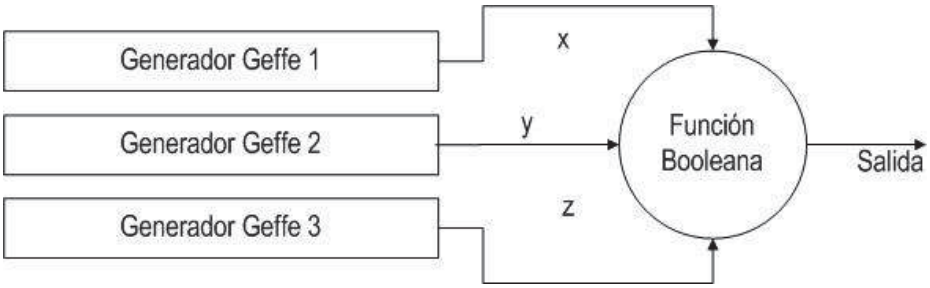

Figura 1. Esquema del Generador Aleatorio

2 Generador Geffe

Como las secuencias obtenidas de Registros de Desplazamiento Lineal con Retroalimentación (RDLR), conocidos por sus siglas en inglés LFSR (Linear Feedback Shift Register) son lineales, se desarrollaron distintas estructuras que permiten obtener secuencias no lineales a partir de los LFSR que componen el generador. Existen tres metodologías principales en ese sentido, (Massodi (et al.) [1], Canteaut [2] y Menezes (et al.) [3]). En particular, el generador Geffe (fig. 2), combina tres LFSR mediante una función booleana que tiene la siguiente forma:

$$f = f(a, b, c) = (c \cdot b) \oplus (b \cdot a) \oplus a$$

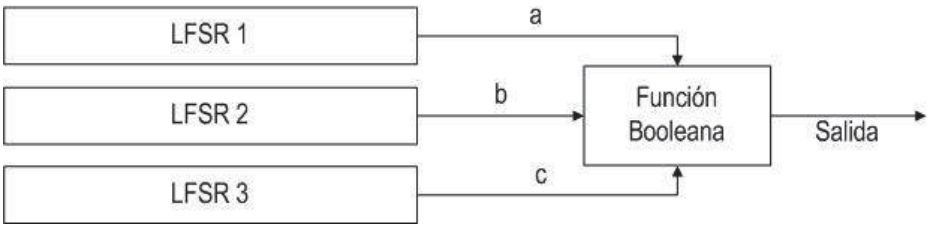

Figura 2. Generador Geffe

2.1 LFSR

La gran mayoría de los sistemas de cifrado de flujo modernos utilizan uno o varios LFSR como bloques de construcción. La principal razón es que tienen un período grande, buenas propiedades estadísticas, y se pueden implementar rápidamente tanto en hardware como en software.

Los LFSR, tienen un estado inicial de carga llamado semilla (clave) y están asociados a un polinomio de conexión que preferentemente debería ser un polinomio

primitivo.

Polinomios Primitivos. Según Parr (et al.) [4]: "Los polinomios primitivos son un tipo especial de polinomio irreductible. Los polinomios irreductibles son aproximadamente comparables con los números primos, es decir, sus únicos factores son 1 y el propio polinomio"

Recomiendan Stahnke [5], Seroussi [6], Mioc y Stratulat [7] la elección de un polinomio primitivo con la mínima cantidad de XOR, ya que de esa manera se incrementa la velocidad del procesamiento.

Período de un LFSR. Las características de cada LFSR son las siguientes. Según indican Wardlaw [8] y Guarino [9], desde una secuencia recurrente lineal de orden L homogénea cuyo polinomio característico sea primitivo. Entonces es periódica y su período es: $p = 2^L - 1$

Complejidad Lineal y Algoritmo de Berlekamp Massey. La complejidad lineal de la secuencia s generada por un LFSR, es el tamaño del LFSR más pequeño que genera esa misma secuencia. Para calcular la complejidad lineal, se recurre al algoritmo de Berlekamp Massey.

Es un algoritmo que proporciona un polinomio de conexión, dado solo unos pocos términos de una secuencia de LFSR, según detallan Campbell [10] y Constantinescu [11]

2.2 Estructura de los Generadores Geffe

Para la selección de los LFSR, que componen cada generador Geffe, además del requisito que los polinomios de conexión sean primitivos, hay otra condición adicional a cumplir y se refiere a que las complejidades lineales de los LFSR que componen cada generador Geffe, sean primos relativos de a dos.

En la tabla 1, se indican los componentes de los generadores:

Tabla 1: Características de los LFSR

Geffe	LFSR	Longitud	Polinomios primitivos	Períodos
1	$LFSR\ 1$	23	$P(x)_1 = x^{23} + x^5 + 1$	$2^{23} - 1 = 8388607$
1	$LFSR\ 2$	19	$P(x)_2 = x^{19} + x^5 + x^2 + x^1 + 1$	$2^{19} - 1 = 52487$
1	$LFSR\ 3$	18	$P(x)_3 = x^{18} + x^5 + x^2 + x^1 + 1$	$2^{18} - 1 = 262143$
2	$LFSR\ 4$	22	$P(x)_4 = x^{22} + x^1 + 1$	$2^{22} - 1 = 4194303$
2	$LFSR\ 5$	17	$P(x)_5 = x^{17} + x^3$	$2^{17} - 1 = 131071$
2	$LFSR\ 6$	25	$P(x)_6 = x^{25} + x^3 + 1$	$2^{25} - 1 = 33554431$
3	$LFSR\ 7$	21	$P(x)_7 = x^{21} + x^2 + 1$	$2^{21} - 1 = 2097151$
3	$LFSR\ 8$	16	$P(x)_8 = x^{16} + x^5 + x^3 + x^2 + 1$	$2^{16} - 1 = 65535$
3	$LFSR\ 9$	23	$P(x)_9 = x^{23} + x^5 + 1$	$2^{23} - 1 = 8388607$

Claves. Para originar las semillas de los distintos LFSR se realiza un proceso, que parte de la clave de una longitud de 32 caracteres, que expresados en código ASCII

(American Standard Code for Information Interchange), tiene 256 bits.

Para simplificar el procedimiento de introducción de la clave, se aceptan solamente las letras del alfabeto (minúsculas y mayúsculas) y los números del sistema de numeración decimal, es decir un total de 62 caracteres permitidos.

3 Generador

3.1 Período del Generador

Gammel, Gottfert y Kniffler [12], proponen que el período del generador es igual al orden del polinomio mínimo m_{S_w} de la secuencia S_w, entonces $período(S_w) = orden(m_{S_w})$. La secuencia S_w, se obtiene de la combinación de las secuencias: S_w, S_y, S_z, que a su vez tienen polinomios mínimos: $m_{S_x}, m_{S_y}, m_{S_z}$, y en el caso que sean primos relativos de a dos, el orden del polinomio mínimo de la secuencia S_w, resulta igual al mínimo común múltiplo del orden de cada polinomio mínimo, es decir $orden(m_{S_w}) = mcm(orden(m_{S_x}), orden(m_{S_y}), orden(m_{S_z}))$.

Además, para los polinomios mínimos se cumple $orden(m_{S_i}) = per(S_i)$. Por esta razón se obtiene: $período(S_w) = orden(m_{S_w}) = mcm(per(S_x), per(S_y), per(S_z))$.

Entonces las tres secuencias obtenidas de los generadores Geffe, al ser combinadas con una función booleana final, tendrán una secuencia resultante cuyo período será igual a $período(S_w) = mcm(per(S_x), per(S_y), per(S_z))$. El valor de los períodos, según al diseño adoptado, será: $período(S_w)$ =1,4272E+45.

3.2 Complejidad Lineal del Generador

Según Gammel, Gottfert y Kniffler [12], la complejidad lineal del generador es igual al grado del polinomio mínimo de la secuencia S_w, $\Lambda(S_w) = grado(m_{S_w})$

El algoritmo de Berlekamp-Massey, permite encontrar el polinomio de menor grado que puede generar una secuencia igual a la que se está analizando. Para el caso en estudio: $\Lambda(S_w) = 1836740$.

3.3 Funciones Booleanas

Las formas en las que se pueden representar las funciones booleanas incluyen: tabla de verdad, vector y forma polar, además de la forma algebraica normal (FNA), las cuales resultarán de interés en este caso. Las funciones booleanas que se necesitan deben ser balanceadas.

Tablas de Verdad de Funciones Balanceadas. Para seleccionar las funciones booleanas a utilizar, se parte de la tabla de verdad balanceadas para tres variables, donde se presentan distintas alternativas de salidas que tiene igual cantidad de ceros y unos. En total se obtuvieron 70 funciones balanceadas, indicadas en la tabla 2.

Funciones Expresadas en FNA. Se determina la forma normal algebraica, a partir de las tablas de verdad anterior, mediante la expresión: $f = A_n.tabla\ verdad(f)$

En la tabla 3 se indican los coeficientes de la FNA:

$f_{FNA} = a_1 1 \oplus a_2 x \oplus a_3 y \oplus a_4(xy) \oplus a_5 z \oplus a_6(xz) \oplus a_7(yz) \oplus a_8(xyz)$

Tabla 2: Funciones balanceadas

f()									f()								f()							
A	1	1	1	1	0	0	0	0	Z	1	1	0	0	0	1	1	1	AY	0	0	1	1	0	1	0	1
B	1	1	1	0	1	0	0	0	AA	1	0	1	0	0	1	1	1	AZ	1	0	0	0	1	1	0	1
C	1	1	0	1	1	0	0	0	AB	0	1	1	0	0	1	1	1	BA	0	1	0	0	1	1	0	1
D	1	0	1	1	1	0	0	0	AC	1	0	0	1	0	1	1	1	BB	0	0	1	0	1	1	0	1
E	0	1	1	1	1	0	0	0	AD	0	1	0	1	0	1	1	1	BC	0	0	0	1	1	1	0	1
F	1	1	1	0	0	1	0	0	AE	0	0	1	1	0	1	1	1	BD	1	1	0	0	0	0	1	1
G	1	1	0	1	0	1	0	0	AF	1	0	0	0	1	1	1	1	BE	1	0	1	0	0	0	1	1
H	1	0	1	1	0	1	0	0	AG	0	1	0	0	1	1	1	1	BF	0	1	1	0	0	0	1	1
I	0	1	1	1	0	1	0	0	AH	0	0	1	0	1	1	1	1	BG	1	0	0	1	0	0	1	1
J	1	1	0	0	1	1	0	0	AI	0	0	0	1	1	1	1	1	BH	0	1	0	1	0	0	1	1
K	1	0	1	0	1	1	0	0	AJ	1	1	1	0	0	0	0	1	BI	0	0	1	1	0	0	1	1
L	0	1	1	0	1	1	0	0	AK	1	1	0	1	0	0	0	1	BJ	1	0	0	0	1	0	1	1
M	1	0	0	1	1	1	0	0	AL	1	0	1	1	0	0	0	1	BK	0	1	0	0	1	0	1	1
N	0	1	0	1	1	1	0	0	AM	0	1	1	1	0	0	0	1	BL	0	0	1	0	1	0	1	1
O	0	0	1	1	1	1	0	0	AN	1	1	0	0	1	0	0	1	BM	0	0	0	1	1	0	1	1
P	1	1	1	0	0	0	1	0	AO	1	0	1	0	1	0	0	1	BN	1	0	0	0	0	1	1	1
Q	1	1	0	1	0	0	1	0	AP	0	1	1	0	1	0	0	1	BO	0	1	0	0	0	1	1	1
R	1	0	1	1	0	0	1	0	AQ	1	0	0	1	1	0	0	1	BP	0	0	1	0	0	1	1	1
S	0	1	1	1	0	0	1	0	AR	0	1	0	1	1	0	0	1	BQ	0	0	0	1	0	1	1	1
T	1	1	0	0	1	0	1	0	AS	0	0	1	1	1	0	0	1	BR	0	0	0	0	1	1	1	1
U	1	0	1	0	1	0	1	0	AT	1	1	0	0	0	1	0	1									
V	0	1	1	0	1	0	1	0	AU	1	0	1	0	0	1	0	1									
W	1	0	0	1	1	0	1	0	AV	0	1	1	0	0	1	0	1									
X	0	1	0	1	1	0	1	0	AW	1	0	0	1	0	1	0	1									
Y	0	0	1	1	1	0	1	0	AX	0	1	0	1	0	1	0	1									

Transformada Rápida de Walsh-Hadamard. El espectro correspondiente puede ser calculado por la expresión: $F(\omega) = \sum_{x \in \mathbb{Z}_2^n} (-1)^{\hat{f}(x) \oplus x \cdot \omega}$.

3.2 Propiedades Criptográficas Deseables en Funciones Booleanas

A continuación se indican las propiedades criptográficamente más significativas, según Elhosary, Hamdy, Farag y Rohiem [13] y Clark, Jacob, Maitra y Stanica [14].

Función Balanceada. Una función booleana de n-variables f se dice ser balanceada si $w(f) = 2n - 1$. Esta propiedad es deseable para evitar ataques criptodiferenciales. La función es balanceada cuando el primer coeficiente del espectro de Walsh-

Hadamard, es igual a cero: $F(0) = 0$

No Linealidad. Esta propiedad reduce el efecto de los ataques por criptoanálisis lineal. La No Linealidad de una función booleana puede ser calculada directamente de la transformada de Walsh-Hadamard $NL_f = \frac{1}{2} \cdot (2^n - |WH_{max}(f)|)$

Tabla 3: Funciones en FNA (Coeficientes ai)

f()	1	2	3	4	5	6	7	8	f()	1	2	3	4	5	6	7	8	f()	1	2	3	4	5	6	7	8
A	1	0	0	0	1	0	0	0	Z	1	0	1	0	1	1	0	0	AY	0	0	1	0	0	1	1	0
B	1	0	0	1	0	1	1	0	AA	1	1	0	0	1	0	1	0	AZ	1	1	1	1	0	1	0	0
C	1	0	1	1	0	1	0	0	AB	0	1	1	0	0	0	0	0	BA	0	1	0	1	1	1	1	0
D	1	1	0	1	0	0	1	0	AC	1	1	1	0	1	0	0	0	BB	0	0	1	1	1	0	0	0
E	0	1	1	1	1	0	0	0	AD	0	1	0	0	0	0	1	0	BC	0	0	0	1	1	0	1	0
F	1	0	0	1	1	1	0	0	AE	0	0	1	0	0	1	0	0	BD	1	0	1	0	1	0	0	0
G	1	0	1	1	1	1	1	0	AF	1	1	1	1	0	1	1	0	BE	1	1	0	0	1	1	1	0
H	1	1	0	1	1	0	0	0	AG	0	1	0	1	1	1	0	0	BF	0	1	1	0	0	1	0	0
I	0	1	1	1	0	0	1	0	AH	0	0	1	1	1	0	1	0	BG	1	1	1	0	1	1	0	0
J	1	0	1	0	0	0	0	0	AI	0	0	0	1	1	0	0	0	BH	0	1	0	0	0	1	1	0
K	1	1	0	0	0	1	1	0	AJ	1	0	0	1	1	0	0	0	BI	0	0	1	0	0	0	0	0
L	0	1	1	0	1	1	0	0	AK	1	0	1	1	1	0	1	0	BJ	1	1	1	1	0	0	1	0
M	1	1	1	0	0	1	0	0	AL	1	1	0	1	1	1	0	0	BK	0	1	0	1	1	0	0	0
N	0	1	0	0	1	1	1	0	AM	0	1	1	1	0	1	1	0	BL	0	0	1	1	1	1	1	0
O	0	0	1	0	1	0	0	0	AN	1	0	1	0	0	1	0	0	BM	0	0	0	1	1	1	0	0
P	1	0	0	1	1	0	1	0	AO	1	1	0	0	0	0	1	0	BN	1	1	1	1	1	0	0	0
Q	1	0	1	1	1	0	0	0	AP	0	1	1	0	1	0	0	0	BO	0	1	0	1	0	0	1	0
R	1	1	0	1	1	1	1	0	AQ	1	1	1	0	0	0	0	0	BP	0	0	1	1	0	1	0	0
S	0	1	1	1	0	1	0	0	AR	0	1	0	0	1	0	1	0	BQ	0	0	0	1	0	1	1	0
T	1	0	1	0	0	1	1	0	AS	0	0	1	0	1	1	0	0	BR	0	0	0	0	1	0	0	0
U	1	1	0	0	0	0	0	0	AT	1	0	1	0	1	1	1	0									
V	0	1	1	0	1	0	1	0	AU	1	1	0	0	1	0	0	0									
W	1	1	1	0	0	0	1	0	AV	0	1	1	0	0	0	1	0									
X	0	1	0	0	1	0	0	0	AW	1	1	1	0	1	0	1	0									
Y	0	0	1	0	1	1	1	0	AX	0	1	0	0	0	0	0	0									

Grado algebraico. Es el grado algebraico máximo de una función booleana.

SAC. El criterio de avalancha estricto (SAC por sus siglas en inglés), requiere los efectos avalancha de todos los bits de entrada.

Una función booleana se dice que satisface SAC sí y solo sí $f(x) \oplus f(x \oplus u)$ es balanceada para toda u con $w(u) = 1$.

3.3 Tablas de resultados

Son rechazadas las funciones que tienen los siguientes resultados: espectro de Walsh-Hadamard $F(0) > 0$, no linealidad $Nf = 0$, grado algebraico $g(f) = 1$ y no

cumplen el SAC, ($SAC = No$).

Son seleccionadas, por lo tanto, las funciones indicadas en la tabla 5.

Tabla 4: Resultados

f()	F(0)	NL_f	gf()	SAC	f()	F(0)	NL_f	g(f)	SAC	f()	F(0)	NL_f	gf()	SAC
A	0	0	2	No	Z	0	2	2	No	AY	0	2	2	Sí
B	0	2	2	Sí	AA	0	2	2	No	AZ	0	2	2	Sí
C	0	2	2	Sí	AB	0	0	2	No	BA	0	2	2	Sí
D	0	2	2	Sí	AC	0	0	2	No	BB	0	2	2	No
E	0	2	2	No	AD	0	2	2	No	BC	0	2	2	Sí
F	0	2	2	Sí	AE	0	2	2	No	BD	0	0	2	No
G	0	2	2	Sí	AF	0	2	2	Sí	BE	0	2	2	Sí
H	0	2	2	No	AG	0	2	2	Sí	BF	0	2	2	No
I	0	2	2	Sí	AH	0	2	2	Sí	BG	0	2	2	No
J	0	0	2	No	AI	0	2	2	No	BH	0	2	2	Sí
K	0	2	2	Sí	AJ	0	2	2	No	BI	0	0	2	No
L	0	2	2	No	AK	0	2	2	Sí	BJ	0	2	2	Sí
M	0	2	2	No	AL	0	2	2	Sí	BK	0	2	2	No
N	0	2	2	Sí	AM	0	2	2	Sí	BL	0	2	2	Sí
O	0	0	2	No	AN	0	2	2	No	BM	0	2	2	Sí
P	0	2	2	Sí	AO	0	2	2	No	BN	0	2	2	No
Q	0	2	2	No	AP	0	0	1	No	BO	0	2	2	Sí
R	0	2	2	Sí	AQ	0	0	2	No	BP	0	2	2	Sí
S	0	2	2	Sí	AR	0	2	2	No	BQ	0	2	2	Sí
T	0	2	2	Sí	AS	0	2	2	No	BR	0	0	2	No
U	0	0	2	No	AT	0	2	2	Sí					
V	0	2	2	No	AU	0	0	2	No					
W	0	2	2	No	AV	0	2	2	No					
X	0	0	2	No	AW	0	2	2	No					
Y	0	2	2	Sí	AX	0	0	2	No					

3.3 Funciones seleccionadas y Complejidad Lineal

En la tabla 5, se detallan las funciones seleccionadas, conforme a los criterios de tabla 4, agrupadas, en orden descendente, según la complejidad lineal estimada.

Tabla 5: Funciones seleccionadas y su complejidad lineal

Función	Complejidad	Función	Complejidad	Función	Complejidad	Función	Complejidad
AF	1836741	BJ	1255291	AZ	1241516	AT	1178222
AM	1836740	I	1255290	S	1241515	Y	1178221
G	1836664	AK	1255214	AL	1241420	BE	1178203
BL	1836663	AH	1255213	AG	1241419	N	1178202
R	1836645	D	1254470	C	1240714	T	1177497
BA	1836644	BO	1254469	BP	1240713	AY	1177496

B	1835118	P	1254393	F	1240618	K	1177478
BQ	1835117	BC	1254392	BM	1240617	BH	1177477

4 Pruebas

4.1 Pruebas Básicas para Secuencias Binarias

Se utilizarán cinco pruebas estadísticas básicas, recomendadas por Dragomir [15], para secuencias binarias.

Prueba de Frecuencia (Monobit). El propósito de esta prueba es determinar si el número de 0 y 1 en s es aproximadamente el mismo, como se esperaría para una secuencia aleatoria, el estadístico χ_1 se determina con la expresión: $\chi_1 = \frac{(n_0 - n_1)^2}{n}$. El valor calculado se compara con la distribución χ^2con un grado de libertad $\nu = 1$.

Prueba de Series. El propósito de esta prueba es determinar si el número de ocurrencias de las cuatro formas (00, 01, 10, 11) de dos bits es aproximadamente el mismo número esperado para una secuencia aleatoria.

Las secuencias aleatorias son uniformes, por lo que cada forma de 2 bits tiene la misma probabilidad de ocurrencia, el estadístico χ_2 se determina con la expresión: $\chi_2 = \frac{4}{n-1} \cdot (n_{00}^2 + n_{01}^2 + n_{10}^2 + n_{11}^2) - \frac{2}{n} \cdot (n_0^2 + n_1^2) + 1$. El valor calculado se compara con la distribución χ^2con un grado de libertad $\nu = 2$.

Prueba de Póker. Esta prueba se usa para probar la aleatoriedad de un patrón de 3 bits en una secuencia, el estadístico χ_3 se determina con la expresión $\chi_3 = \sum_{\mathrm{i}}^{\mathrm{d}} \frac{(\mathrm{A(d)_i} - \mathrm{E_i})^2}{\mathrm{E_i}}$, donde $\mathrm{E_i} = \mathrm{C_i^d} \cdot \left(\frac{\mathrm{L}}{2^{\mathrm{d}} \cdot \mathrm{d}}\right)$. El valor calculado se compara con la distribución χ^2con un grado de libertad $\nu = 2^d - 1$.

Prueba de Runs. El objetivo de esta prueba es calcular el número total de secuencias ininterrumpidas de bits igual a cero o uno. Una secuencia ininterrumpida de bits idénticos de longitud k es una subsecuencia compuesta por k bits del mismo valor (cero o uno) y que está limitada por un bit de valor diferente. El propósito de la prueba de ejecuciones es determinar si el número de ejecuciones (de ceros o unos) de varias longitudes en la secuencia s es el esperado para una secuencia aleatoria. El número esperado de espacios (o bloques) de longitud i en una secuencia aleatoria de longitud n viene dado por la fórmula $e_i = \frac{(n-i+3)}{2^{i+2}}$.

El estadístico χ_4 se determina con la expresión $\chi_4 = \sum_{i=1}^{k} \frac{(B_i - e_i)^2}{e_i} + \sum_{i=1}^{k} \frac{(G_i - e_i)^2}{e_i}$.

El valor calculado se compara con la distribución χ^2con un grado de libertad $\nu = 2k - 2$.

Prueba de Autocorrelación. El propósito de esta prueba es verificar las

correlaciones entre la secuencia s y las versiones desplazadas (no cíclicas) de la misma. Sea d un entero fijo $1 \leq d \leq \left\lfloor \frac{n}{2} \right\rfloor$. El número de bits en s que no es igual a los desplazados d es $A_{(d)} = \sum_{i=0}^{n-d-1} s_i \oplus s_{i+d}$.

El estadístico χ_5 es $\chi_5 = \frac{2 \cdot \left(A_{(d)} - \frac{n-d}{2}\right)}{\sqrt{n-d}}$.

4.2 Pruebas de los Generadores con las Funciones Booleanas Seleccionadas

A las funciones de la tabla 5, se las somete a pruebas estadísticas en los párrafos anteriores, con 1.000 claves distintas y secuencias de 100.000 de bits y un nivel de significancia de $\propto = 0{,}10$. Los resultados obtenidos se muestran en tabla 6.

Cualquiera de ellas puede ser utilizada como función booleana de combinación de las secuencias provenientes de los tres generadores Geffe.

Tabla 6: Funciones que superaron las pruebas estadísticas

$f_{booleana}(x,y,z)$	$f_{booleana}(x,y,z)$
$f_{AF} = 1 \oplus x \oplus y \oplus (x \cdot y) \oplus (x \cdot z) \oplus (y \cdot z)$	$f_C = 1 \oplus y \oplus (x \cdot y) \oplus (x \cdot z)$
$f_{AM} = x \oplus y \oplus (x \cdot y) \oplus (x \cdot z) \oplus (y \cdot z)$	$f_{BP} = y \oplus (x \cdot y) \oplus (x \cdot z)$
$f_G = 1 \oplus y \oplus (x \cdot y) \oplus z \oplus (x \cdot z) \oplus (y \cdot z)$	$f_{AT} = 1 \oplus y \oplus z \oplus (x \cdot z) \oplus (y \cdot z)$
$f_{BL} = y \oplus (x \cdot y) \oplus z \oplus (x \cdot z) \oplus (y \cdot z)$	$f_Y = y \oplus z \oplus (x \cdot z) \oplus (y \cdot z)$
$f_B = 1 \oplus (x \cdot y) \oplus (x \cdot z) \oplus (y \cdot z)$	$f_{BE} = 1 \oplus x \oplus z \oplus (x \cdot z) \oplus (y \cdot z)$
$f_{BQ} = (x \cdot y) \oplus (x \cdot z) \oplus (y \cdot z)$	$f_N = x \oplus z \oplus (x \cdot z) \oplus (y \cdot z)$
$f_{BJ} = 1 \oplus x \oplus y \oplus (x \cdot y) \oplus (y \cdot z)$	$f_T = 1 \oplus y \oplus (x \cdot z) \oplus (y \cdot z)$
$f_I = x \oplus y \oplus (x \cdot y) \oplus (y \cdot z)$	$f_{AY} = y \oplus (x \cdot z) \oplus (y \cdot z)$
$f_D = 1 \oplus x \oplus (x \cdot y) \oplus (y \cdot z)$	$f_K = 1 \oplus x \oplus (x \cdot z) \oplus (y \cdot z)$
$f_{BO} = x \oplus (x \cdot y) \oplus (y \cdot z)$	$f_{BH} = x \oplus (x \cdot z) \oplus (y \cdot z)$

5 Conclusiones

Se desarrolló un generador de secuencias binarias pseudoaleatorias de elevado período y complejidad lineal. Para ello se diseñó un dispositivo que combina en forma no lineal las secuencias producidas por generadores Geffe, mediante una función booleana de tres variables.

Los LFSR que componen cada generador Geffe tienen polinomios de conexión primitivos y las longitudes de los mismos son primos relativos de a pares, lo que asegura un elevado período en la secuencia resultante.

La función booleana que es la responsable de la combinación no lineal, debe asegurar las mejores prestaciones criptográficas, partiendo de funciones balanceadas expresadas en la forma normal algebraica.

Realizado el proceso de selección, las funciones fueron incorporadas al generador y luego puestas a funcionar para generar las secuencias respectivas con distintos valores de claves y ser sometidas a las pruebas de aleatoriedad.

Las funciones que superaron las pruebas, son la que en definitiva pueden utilizar

como función de combinación del generador propuesto.

Los generadores de secuencias pseudoaleatorias, en nuestro caso binarias, son útiles en una amplia variedad de aplicaciones: simulaciones por computadoras, estudio de fenómenos físicos, investigación operativa, simulación de Monte Carlo, pruebas de primalidad, estadísticas computacionales, juegos de azar, juegos de computadora y en los cifrados de flujo.

Referencias

1. Massodi, F., Alam, S., Bokhari, M.: An Analysis of Linear Feedback Shift Registers in Stream Ciphers. International Journal of Computer Application. 16 (17). 0975 – 8887 (2012).
2. Canteaut, A. y Filio, E.: Ciphertext only reconstruction of stream ciphers based on combination generators. Fast Software Encryption 2000, Lecture Notes in Computer Science. 1978. 165–180 (2001).
3. Menezes, A.; Van Oorschot, P.; Vanstone, S.: Handbook of Applied Cryptography. USA: Massachusetts Institute of Technology (1996).
4. Paar, C.; Pelzl, L.: Understanding Cryptography. Alemania: Springer. (2010).
5. Stahnke, W.: Primitive Binary Polynomials. Mathematics of computation. 27. 124. 977-980 (1973).
6. Seroussi, G.: Table of Low-Weight Binary Irreducible Polynomials. Computer Systems Laboratory (1998).
7. Mioc, M., Stratulat, M.: Study of Software implementation for Linear Feedback Shift Register based on 8th degree irreducible polynomials. International Journal of Computers. 8 (2014).
8. Wardlaw, W.: A Matrix Model for the Linear Feedback Shift Register. Identification Systems Branch Radar Division. NRL Report 9179 (1989).
9. Guarino, S.: Ciphertext-only reconstruction of LFSR-based stream ciphers. Facoltá di Scienze Matematiche Fisiche e Naturali, Universitá degli studi di Roma Tre (2010).
10. Campbell, P.: An Implementation of the Berlekamp-Massey Linear Feedback Shift-Register Synthesis Algorithm in the C Programing Language. Sandia National Laboratories (1999).
11. Constantinescu, N.: Combining Linear Feedback Shift Registers. Annals of University of Craiova, Math. Comp. Sci. Ser. 36 (2). 42–46 (2009).
12. Gammel, B., Gottfert, R. y Kniffler, O.: The Achterbahn Stream Cipher. Infineon Technologies AG (2005).
13. Elhosary, A., Hamdy, N., Farag, I., Rohiem, I.: State of the ART in Boolean Functions Cryptographic Assessment. International Journal of Computer Networks and Communications Security.1. (3). 88-94 (2013).
14. Clark, J., Jacob, J., Maitra, S. y Stanica, P.: Almost Boolean Functions: The Design of Boolean Functions by Spectral Inversion. Computational Intelligence. 20. (3). 450-462 (2004).
15. Dragomir, I.: Statistical Assessment of Binary Sequences Generated by Cryptographic Algorithms. Social Economic Debates. 5 (2) (2016).