CACiC 2023
CONGRESO ARGENTINO DE CIENCIAS DE LA COMPUTACIÓN

Simulación de Montecarlo, con Dos Variables Aleatorias, Utilizando Generador Binario Pseudoaleatorio, para Problemas de Reparación de Máquinas

Andrés Francisco Farías[1], Germán Antonio Montejano[2], Ana Gabriela Garis[3], Pablo Marcelo García[4], Andrés Alejandro Farías[5]

Universidad Nacional de La Rioja, La Rioja [1,5] – Argentina,
Universidad Nacional de San Luis, San Luis – Argentina [2,3],
Universidad Nacional de La Pampa, Santa Rosa – Argentina [4]

afarias665@yahoo.com.ar [1], gmonte@unsl.edu.ar [2], agaris@gmail.com [3], pablogarcia@unlpam.edu.ar [4], andres_af86@hotmail.com

Abstract. Simulación de Montecarlo de dos variables aleatorias, con números aleatorios entregados por un generador binario. Se trabaja sobre un caso de reparación de máquinas similares de una planta industrial, donde se realizaron 200 observaciones de los tiempos entre fallas y el tiempo que demandó la reparación de las máquinas. Para ello se ejecuta un experimento para un ciclo de 20 fallas y luego se repite el procedimiento 200 veces, de donde se obtiene el promedio de las mismas.

Para realizar esta simulación de Montecarlo se necesitan obtener números aleatorios, para lograrlo podemos contar con distintos dispositivos, uno de ellos es el generador de secuencias binarias pseudoaleatorias.

En este trabajo se expone el procedimiento de diseño de dicho mecanismo para producir secuencias binarias, basadas en la combinación de registros de desplazamiento con retroalimentación lineal (Linear Feedback Shift Register, LFSR). El proceso incluye la descripción del modelo, la estructura de cada generador, selección de las funciones booleanas que cuenten con las mejores propiedades criptográficas, la definición de la combinación final. Luego se verifica la aleatoriedad de las secuencias obtenidas, se aplican a las mismas un conjunto de pruebas estadísticas de aleatoriedad. Luego la secuencia binaria se convierte en secuencia decimal, mediante un protocolo específico.

Keywords: LFSR, cipher, key, boolean function, non-linearity, simulation

1. Introducción

El trabajo consta de siguientes etapas:

- Diseño del generador binario pseudoaleatorio
- Pruebas de aleatoriedad
- Conversión de la secuencia binaria a decimal
- Simulación de Montecarlo de dos variables aleatorias
- Caso práctico: Reparación de máquinas.

2. Generador binario pseudoaleatorio

Entre otras características de este generador tenemos la facilidad de implementación, pero, fundamentalmente un período con una longitud significativa.

Es en esos términos que se propone un modelo que responda a tales exigencias. La modalidad elegida se basa en la combinación no lineal de secuencias producidas por cuatro LFSR [1], [2].

El procedimiento de construcción de un generador pseudoaleatorio de ese estilo requiere de varias etapas:

- Definición esquemática del modelo.
- Elección de los distintos LFSR.
- Selección de funciones booleanas en base a sus propiedades criptográficas.
- Conformación del generador con los componentes ya seleccionados.
- Clave y el procedimiento para generar los estados iniciales de los LFSR.
- Pruebas estadísticas y los criterios de interpretación de los resultados.
- Puesta en funcionamiento y realización de las pruebas de aleatoriedad.

2.1 Definición esquemática del modelo

El generador propuesto en este trabajo, está conformado por cuatro LFSR, que tienen cada uno, una función booleana de filtrado no lineal, que producen secuencias binarias, las que luego son combinadas mediante otras funciones booleanas de combinación, y una operación final XOR.:

2.2 Elección de los distintos LFSR

Las longitudes y polinomios primitivos de cada LFSR, que componen el generador, son las siguientes [3], [4], [5], en la tabla 1 y en la figura 2 se indican como queda el dispositivo.

Tabla 5. LFSR, longitudes y polinomios primitivos del Generador

LFSR	Longitud	Polinomios primitivos
1	47	$P(x) = x^{47} + x^{32} + x^{24} + x^{11} + 1$
2	61	$P(x) = x^{61} + x^{57} + x^{26} + x^{3} + 1$
3	59	$P(x) = x^{59} + x^{54} + x^{46} + x^{26} + 1$
4	53	$P(x) = x^{53} + x^{50} + x^{41} + x^{20} + 1$

2.3 Selección de las funciones booleanas

Propiedades criptográficas deseables. A continuación se indican algunas de las propiedades criptográficamente más significativas, adoptadas para este trabajo [6], [7], [8]: Función Balanceada, Alta No Linealidad, Alto Grado Algebraic y Criterio de Avalancha Estricto.

Siguiendo los criterios arriba indicados las funciones booleanas aceptadas, son observadas en tabla 2:

Tabla 6. Funciones de cuatro variables adoptadas

$f_{5775} = a \oplus b \oplus a \cdot b \oplus a \cdot c \oplus a \cdot d$
$f_{4529} = a \oplus c \oplus a \cdot c \oplus b \cdot c \oplus c \cdot d$
$f_{4722} = a \oplus b \oplus a \cdot c \oplus b \cdot c \oplus c \cdot d$
$f_{4393} = a \oplus c \oplus a \cdot d \oplus b \cdot d \oplus c \cdot d$
$f_{3981} = a \oplus a \cdot c \oplus b \cdot c \oplus d \oplus c \cdot d$
$f_{3672} = a \oplus a \cdot b \oplus c \oplus a \cdot c \oplus a \cdot d$

2.4 Conformación del generador combinacional

El generador combinacional se indica en la figura 1:

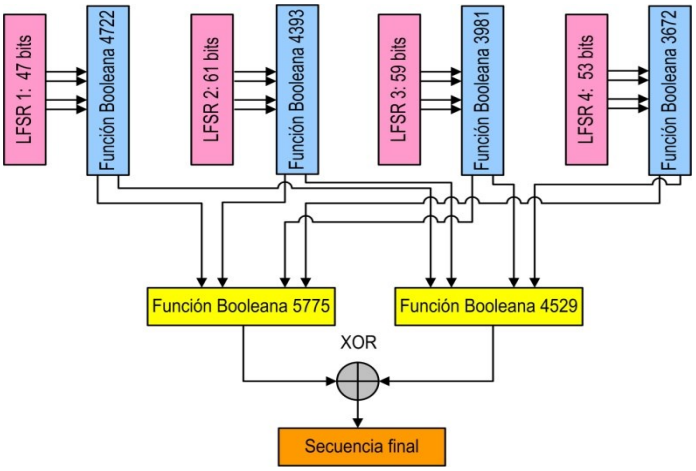

Fig. 1. Generador Combinacional

2.5 Clave

Para originar los estados iniciales de los distintos LFSR se realiza un proceso que utiliza una clave de 32 caracteres, que expresada en código ASCII (American Standard Code for Information Interchange), tiene longitud de 256 bits.

La clave es sometida a un proceso criptográfico, De la operación resulta un vector SK[j] de 256 bits, que es el que proveerá los estados iniciales de los LFSR, en forma secuencial.

3. Pruebas de aleatoriedad

3.1 Elección de las pruebas estadísticas

Fueron seleccionadas algunas pruebas de la Norma NIST Special Publication 800-

22, del trabajo de Rukhin (et al.) [10], estas son las siguientes: Prueba de frecuencia, Prueba de frecuencia en un bloque, Prueba de rachas, Prueba de rachas de unos en un bloque, Prueba de sumas acumuladas, Prueba de entropía aproximada

3.2 Pruebas sobre el generador

Se analizaron cien secuencias binarias de 100.000 bits, obtenidas del generador a partir de cien claves distintas. El nivel de significancia adoptado para las pruebas estadísticas es de $\alpha = 0{,}01$. La hipótesis nula es:

$$H_0 \rightarrow p\_valor > 0{,}01$$

3.3 Proporción de muestras que pasan las pruebas

Para el análisis de los resultados, se determina la proporción de muestras que superan las pruebas, y con esos datos se construye un gráfico de puntos, luego se verifica si los mismos caen dentro de los límites superior e inferior, donde k es el número de muestras. $LS, LI = (1-\alpha) \pm 3 \cdot \sqrt{\alpha \cdot (1-\alpha)/k}$

En nuestro caso $k = 100$ y el nivel de significancia elegido es: $\alpha = 0.01$, los límites quedan: $LS = 1{,}02$ y $LI = 0{,}96$

Se consideran todas pruebas, los resultados se indican en la tabla 3.

Tabla 3. Pruebas estadísticas

Pruebas	Proporción	Superior	Inferior
Frecuencias	1,00	1,02	0,96
Frecuencias en un Bloque	1,00	1,02	0,96
Rachas	1,00	1,02	0,96
Rachas de Unos en un Bloque	0,98	1,02	0,96
Sumas Acumuladas Adelante	1,00	1,02	0,96
Sumas Acumuladas Atrás	1,00	1,02	0,96
Entropía Aproximada	1,00	1,02	0,96

En la figura 2 se aprecia el resultado, en definitiva, las secuencias que entrega el generador superan las pruebas de aleatoriedad.

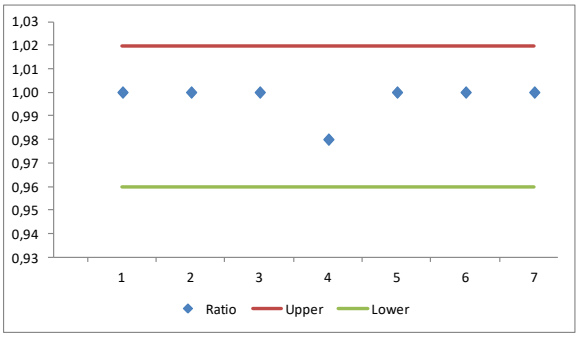

Fig. 2. Gráfico de puntos

4. Conversión de secuencia binaria a decimal

4.1 Procedimiento de conversión de secuencia binaria a secuencia decimal

Como la secuencia entregada por el generador es binaria y se necesita trabajar con números decimales, se adopta un criterio de conversión que es el siguiente:

- Dividir la secuencia binaria en bloques de 7 bits cada uno.
- Convertir esos bloques de 7 bits en número decimal.
- Como necesitamos números decimales comprendidos entre 0 y 99, los valores superiores a 99 se descartan.

Del resultado de la operación queda una secuencia decimal de 0 a 99 pseudoaleatoria. Finalmente queda verificar la aleatoriedad, se realiza una prueba de χ^2sobre la secuencia decimal obtenida.

4.2 Prueba de χ^2 sobre la secuencia decimal.

Realizamos la prueba de χ^2 sobre la secuencia decimal, y nos queda la siguiente tabla 4.

Tabla 4. Tabla de frecuencias y cálculo de χ^2

Clase *A*	Observada *B*	Esperada *C*	C-B *D*	D*D *E*	E/10 *F*
1	1000	1002	2	4	0,00
2	1000	1027	27	729	0,73
3	1000	966	-34	1156	1,16
4	1000	963	-37	1369	1,37
5	1000	1047	47	2209	2,21
6	1000	1002	2	4	0,00
7	1000	1036	36	1296	1,30
8	1000	980	-20	400	0,40
9	1000	1009	9	81	0,08
10	1000	968	-32	1024	1,02
	10000	**10000**			**8,27**

Finalmente comparamos el χ^2 de aceptación para un $\alpha = 0{,}01$ y 9 grados de libertad, con el χ^2 obtenido en tabla 4, de modo tal que resulte:

$$\boldsymbol{\chi^2 Aceptación > \chi^2 Obtenido}$$

$$\mathbf{21,66 > 8,27}$$

Cae en la zona de aceptación.

4.3 Generar números aleatorios para la primera y segunda variable

Se hace correr el generador de números aleatorios con la clave: 1X92ZQXu8f3oDzSEZRN7OcCM1LR97P9a y con los datos obtenidos, se realiza la tabla 5. Se repite el procedimiento con la clave: **P1kAlMiG2Kb7FzP5tM1QBI6DSS92c31A** y con los datos obtenidos, nos queda la tabla 6.

Tabla 5. Números aleatorios

67	82	30	21	97	91	47	19	94	11
89	94	14	58	44	17	74	21	13	75
…	…	…	…	…	…	…	…	…	…

Tabla 6. Números aleatorios

71	52	18	20	42	4	35	43	51	1
0	12	60	22	42	61	91	80	62	7
…	…	…	…	…	…	…	…	…	…

5. Simulación de Montecarlo

Se aplica el método de simulación de Montecarlo, cuando componentes del sistema son de comportamiento aleatorio [11]. El procedimiento tiene las siguientes etapas:

- Datos de las variables en estudio (observaciones).
- Confección Tabla de de frecuencias, probabilidades, probabilidades acumuladas e intervalos de números aleatorios.
- Generar los números aleatorios, con el generador presentado en este trabajo.
- Simular el experimento para un ciclo de 20 fallas.
- Realizar 200 repeticiones del experimento y obtener el promedio.

6. Reparación de máquinas

6.1 Procedimiento para simular dos variables

A continuación se realiza una simulación de Montecarlo donde se presenta dos variables aleatorias. En este caso se analiza un problema de reparación de máquinas, y se desarrolla un experimento de simulación para 20 fallas.

Datos de la primera variable.

En la tabla 7 figuran los datos del tiempo entre fallas de 200 observaciones.

Tabla 7. Tiempo entre fallas

3	4	1	3	2	2	2	3	6	4	4	5	6	2	6	1	4	1	1	2
4	6	3	5	5	1	4	4	5	3	2	4	4	2	4	5	5	3	1	4
5	1	4	1	6	2	6	2	6	4	4	4	6	3	1	2	6	4	1	2
3	5	5	1	1	1	1	4	1	4	3	3	3	5	1	6	4	5	6	2
4	2	2	2	1	4	6	2	1	1	1	1	4	1	2	6	3	4	3	5
4	6	1	3	6	2	5	6	4	1	5	4	6	1	6	6	5	6	4	6
4	4	3	2	2	1	4	6	6	3	3	5	2	6	1	5	4	6	2	6
2	4	2	4	6	5	5	5	4	1	5	2	2	6	6	4	2	6	1	6
3	3	1	6	2	2	2	2	5	4	1	5	6	4	2	5	4	1	1	3
2	5	4	5	2	6	3	3	2	3	2	6	1	2	5	5	5	5	5	2

Tabla de frecuencias, probabilidades, probabilidades acumuladas e intervalos de números aleatorios

En la tabla 8, aparece la distribución de frecuencia, las probabilidades, las probabilidades acumuladas e intervalos de números aleatorios del tiempo entre fallas.

Tabla 8. Intervalos de números aleatorios del tiempo entre fallas

Tiempo entre fallas	Frecuencia	Probabilidad	Probabilidad acumulada	Números aleatorios
1	34	0,17	0,17	0 al 16
2	38	0,19	0,36	17 al 35
3	23	0,11	0,47	36 al 46
4	39	0,20	0,67	47 al 66
5	31	0,15	0,82	67 al 81
6	35	0,18	1,00	82 al 99
	200	1,00		

Tiempos entre fallas aleatorios.

A partir de las tablas 5 y 8, en tabla 9 se establecen los tiempos entre fallas aleatorios.

Tabla 9. Tiempos entre fallas aleatorios

5	6	2	2	6	6	4	2	6	1
6	6	1	4	3	2	5	2	1	5
6	2	3	1	4	1	6	1	4	2
2	2	1	4	1	4	2	4	4	6
…	…	…	…	…	…	…	…	…	…

Datos de segunda variable.

En la tabla 10 figuran los tiempos de reparación para 200 observaciones.

Tabla 10. Tiempo de reparaciones

1	4	4	1	4	1	3	4	1	3	2	4	2	2	4	4	3	4	2	1
4	2	2	4	2	4	1	1	1	3	2	1	1	1	2	4	3	3	2	4
1	3	4	3	3	3	3	1	4	2	4	1	1	4	4	4	2	2	2	4
2	2	1	3	1	2	3	4	3	2	3	2	2	3	1	2	1	2	1	2
1	1	4	4	1	4	3	2	1	2	4	3	1	1	3	1	2	1	4	4
2	4	3	1	2	2	1	3	2	4	1	3	2	1	4	4	3	1	3	1
2	1	2	3	1	4	3	2	4	2	2	1	2	1	3	1	2	3	2	4
3	3	2	1	3	1	3	3	1	2	4	2	3	1	3	3	1	4	4	4
3	4	3	2	1	4	4	3	2	3	4	2	3	2	2	3	4	1	2	1
1	2	1	4	4	1	2	1	3	3	4	3	1	2	1	4	3	3	1	3

Tabla de frecuencias, probabilidades, probabilidades acumuladas e intervalos de números aleatorios

En la tabla 11, aparece la distribución de frecuencia, las probabilidades, las probabilidades acumuladas e intervalos de números aleatorios del tiempo de reparaciones.

Tabla 11. Intervalos de números aleatorios

Tiempo de reparación	Frecuencia	Probabilidad	Probabilidad acumulada	Números aleatorios
1	54	0,27	0,27	0 al 26
2	51	0,26	0,53	27 al 52
3	48	0,24	0,77	53 al 76
4	47	0,23	1,00	77 al 99
	200	1,00		

Tiempos de reparación aleatorios.

A partir de las tablas 6 y 11, en tabla 12 se establecen los tiempos de reparaciones aleatorios.

Tabla 12. Tiempo de reparaciones aleatorios

3	2	1	1	2	1	2	2	2	1
1	1	3	1	2	3	4	4	3	1
3	4	2	3	4	3	4	3	1	2
4	4	1	3	4	2	1	2	2	4
…	…	…	…	…	…	…	…	…	…

Simular el experimento.

De la tabla 5 y 6 de números aleatorios se seleccionan 20 números de cada una de ellas, para realizar el experimento en un ciclo de 20 fallas, entonces nos queda la tabla 13.

Tabla 13. Experimento para determinar el tiempo inactivo de las máquinas

Número de fallas	Aleatorio tabla 5	Tiempo entre fallas	Inicio de falla	Inicio reparación	Aleatorio de tabla6	Tiempo de reparación	Fin de reparación	**Tiempo inactivo**	Espera
1	67	5	5	5	71	3	8	**3**	0
2	82	6	11	11	52	2	13	**2**	0
3	30	2	13	15	18	1	14	**1**	0
4	21	2	15	17	20	1	16	**1**	0
5	97	6	21	21	42	2	23	**2**	0
6	91	6	27	27	4	1	28	**1**	0
7	47	4	31	31	35	2	33	**2**	0
8	19	2	33	33	43	2	35	**2**	0
9	94	6	39	39	51	2	41	**2**	0
10	11	1	40	43	1	1	42	**2**	1
11	89	6	46	46	0	1	47	**1**	0
12	94	6	52	52	12	1	53	**1**	0
13	14	1	53	56	60	3	56	**3**	0
14	58	4	57	57	22	1	58	**1**	0
15	44	3	60	60	42	2	62	**2**	0

16	17	2	62	62	61	3	65	**3**	0
17	74	5	67	67	91	4	71	**4**	0
18	21	2	69	70	80	4	75	**6**	2
19	13	1	70	71	62	3	78	**8**	5
20	75	5	75	75	7	1	79	**4**	3
							79	**51**	

Tiempo inactivo de las máquinas: **51 horas** en un ciclo de 20 fallas.

Realizar varias simulaciones y obtener el resultado.

Para obtener un mejor resultado realizamos muchas simulaciones del experimento anterior, en nuestro caso adoptamos 200 simulaciones y calculando el promedio con los resultados obtenidos de las simulaciones, que figura en la tabla 14:

Tabla 14**.** Simulaciones de tiempo inactivo de las máquinas

51	131	60	58	44	53	88	66	132	65	85	70	64	74	65	47	54	48	60	43
71	51	70	55	53	87	77	62	49	61	68	54	48	53	99	44	75	48	74	63
64	46	101	58	52	62	63	65	70	50	43	57	101	55	69	63	90	67	61	58
68	65	52	68	48	55	84	53	74	53	68	62	57	43	58	57	69	72	62	54
54	60	54	101	52	94	111	74	127	65	59	47	55	68	58	56	90	55	110	60
68	64	105	45	63	85	64	65	76	60	44	60	63	90	77	53	55	46	65	65
65	57	69	84	63	72	71	55	60	73	117	58	77	69	85	51	58	69	66	54
64	51	56	56	69	85	83	100	46	60	60	115	68	59	54	69	75	56	59	49
89	55	71	61	93	50	61	69	55	78	57	53	60	54	48	53	51	66	62	82
57	91	71	63	73	54	99	65	67	48	43	77	45	68	49	60	67	54	51	71

El promedio de las 200 simulaciones, indicadas en la tabla 14, es de **65,74 horas** de tiempo inactivo de las máquinas, en el ciclo estudiado de 20 fallas.

7. Conclusiones

Uno de los usos de los generadores de números aleatorios es la realización de las simulaciones de Montecarlo, donde se requieren variables aleatorias para el cálculo de los experimentos. Existen muchos casos de simulaciones donde aparecen una, dos, tres o múltiples variables aleatorias.

En este trabajo aparecen dos variables y se generaron dos tablas aleatorias a partir del generador presentado. Con esos valores se realizaron las 200 simulaciones que determinaron el promedio del tiempo inactivo de las máquinas para un ciclo de 20 fallas.

El uso del generador permite, con respecto al cálculo mediante planillas de números aleatorios, un proceso más rápido y con mayor cantidad de simulaciones.

Referencias

1. Massodi, F., Alam, S. and Bokhari, M., "A Analysis of Linear Feedback Shift Registers in Stream Ciphers", International Journal of Computer Application, 16 (17), pp. 0975 – 887, 2012.
2. Menezes, A.,Van Oorschot, P. and Vanstone, S., "Handbook of Applied Cryptography", Massachusetts Institute of Technology, 1996.
3. Parr, C. and Pelzl, L., Understanding Cryptography, Springer, 2010.
4. Stahnke, W., "Primitive Binary Polynomials", Mathematics of Computation, 27. 124, pp. 977-980, 1973.
5. Seroussi, G., "Table of Low-Weight Binary Irreducible Polynomials", Computer Systems Laboratory, 1998.
6. Clark, J., Jacob, J., Maitra, S., Stanica, P.: Almost Boolean Functions: The Design of Boolean Functions by Spectral Inversion. Computational intelligence. 20. (3), 450—462 (2004)
7. Braeken, A.: Cryptographic Properties of Boolean Functions and S-Boxes. Faculteit Ingenieurswetenschappen. Katholieke Universiteit Leuven (2003)
8. 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)
9. Fishman, G.: Multiplicative Congruential Random Number Generators with Modulus 2ß : An Exhaustive Analysis for ß = 32 and a Partial Analysis for ß = 48. Mathematics of Computation. 54. (189), 33--344 (1990)
10. Rukhin, A., Soto, J., Nechvatal, J., Smid, M., Barker, E., Leigh, S., Levenson, M., Vangel, M., Banks, D., Heckert, A., Dray, J., and Vo, S., "A Statistical Prueba Suite for Random and Pseudorandom Number Generators for Cryptographic Applications", National Institute of Standards and Technology, (2000).
11. Heizer, J. y Render, B. Dirección de la producción y de operaciones. Decisiones tácticas, 8.ª edición PEARSON EDUCACIÓN, S.A., Madrid, (2008)