UNIVERSIDAD DE MÁLAGA

ETSI Telecomunicación

GENERACIÓN DE SECUENCIAS PSEUDOALEATORIAS GAUSSIANAS MEDIANTE REGISTROS DE DESPLAZAMIENTO CON REALIMENTACIÓN LINEAL EN CUERPOS EXTENDIDOS

Tesis Doctoral

Autor:

D. Guillermo Cotrina Cuenca

Directores:

D. Alberto Peinado Domínguez

D. Andrés Ortiz García

Noviembre 2021

UNIVERSIDAD DE MÁLAGA

AUTOR: Guillermo Cotrina Cuenca

https://orcid.org/0000-0002-4221-4598

EDITA: Publicaciones y Divulgación Científica. Universidad de Málaga

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD DE LA TESIS PRESENTADA PARA OBTENER EL TÍTULO DE DOCTOR

D. Guillermo Cotrina Cuenca,

Estudiante del programa de doctorado de la ETSI Telecomunicación de la Universidad de Málaga, autor de la tesis, presentada para la obtención del título de doctor por la Universidad de Málaga, titulada: GENERACIÓN DE SECUENCIAS PSEUDOALEATORIAS GAUSSIANAS MEDIANTE REGISTROS DE DESPLAZAMIENTO CON REALIMENTACIÓN LINEAL EN CUERPOS EXTENDIDOS.

Realizada bajo la dirección de D. Alberto Peinado Domínguez y de D. Andrés Ortiz García y la tutorización de D. Alberto Peinado Domínguez.

DECLARO QUE:

La tesis presentada es una obra original que no infringe los derechos de propiedad intelectual ni los derechos de propiedad industrial u otros, conforme al ordenamiento jurídico vigente (Real Decreto Legislativo 1/1996, de 12 de abril, por el que se aprueba el texto refundido de la Ley de Propiedad Intelectual, regularizando, aclarando y armonizando las disposiciones legales vigentes sobre la materia), modificado por la Ley 2/2019, de 1 de marzo.

Igualmente asumo, ante a la Universidad de Málaga y ante cualquier otra instancia, la responsabilidad que pudiera derivarse en caso de plagio de contenidos en la tesis presentada, conforme al ordenamiento jurídico vigente.

En Málaga, a _____ de _____ de _____

Fdo.: _____ COTRINA CUENCA, GUILLERMO (AUTENTICACIÓN) Firmado digitalmente por COTRINA CUENCA, GUILLERMO (AUTENTICACIÓN) Fecha: 2021.11.22 13:34:13 +01'00' Doctorando	Fdo.: _____ ORTIZ GARCIA ANDRES - Director de Tesis
Fdo.: _____ Firmado digitalmente por PEINADO DOMINGUEZ ALBERTO Fecha: 2021.11.22 15:01:38 +01'00' Director y tutor de Tesis	

Autorización de los directores para la lectura de la Tesis

LOS DOCTORES DON ALBERTO PEINADO DOMÍNGUEZ Y DON ANDRÉS ORTIZ GARCÍA , PROFESORES DE LA UNIVERSIDAD DE MÁLAGA DEL DEPARTAMENTO DE INGENIERÍA DE COMUNICACIONES, DE LA FACULTAD DE INGENIERÍA DE TELECOMUNICACIÓN, AUTORIZAN:

Al doctorando, Guillermo Cotrina Cuenca, a la lectura y defensa de su Tesis Doctoral titulada "GENERACIÓN DE SECUENCIAS PSEUDOALEATORIAS GAUSSIANAS MEDIANTE REGISTROS DE DESPLAZAMIENTO CON REALIMENTACIÓN LINEAL EN CUERPOS EXTENDIDOS", de la cual son directores. Además, informan de que los artículos que han servido para justificar este trabajo no han sido usados en otra Tesis.

Y para que así conste, expiden y firman este informe en Málaga, a _____ de _____ de _____.

Firmado digitalmente por PEINADO DOMINGUEZ ALBERTO
Fecha: 2021.11.22 15:03:35 +01'00'

ORTIZ GARCIA ANDRES -

Resumen

La generación de números pseudoaleatorios constituye un pilar esencial para el desarrollo de un amplio rango de aplicaciones en numerosos ámbitos técnicos y científicos. La necesidad de generar secuencias reproducibles con determinadas distribuciones de probabilidad es especialmente importante en áreas como la criptografía, la comprobación de circuitos electrónicos, la transmisión de datos o las simulaciones de procesos.

Entre los diversos métodos que tradicionalmente se han utilizado para la generación pseudoaleatoria destaca la utilización del registro de desplazamiento con realimentación lineal (LFSR). La simplicidad de estos dispositivos junto con las propiedades de las secuencias generadas los ha convertido en elementos habituales en los procesos de cifrado en flujo, especialmente orientados a proteger las comunicaciones de alta velocidad; en las comunicaciones basadas en multiplexación por división de código (CDMA); o en los procesos de aleatorización propios de los canales radio que se utilizan en las comunicaciones inalámbricas. En todos estos casos, los LFSR están definidos sobre $GF(2)$ trabajando, por tanto, con secuencias binarias.

Las implementaciones software de los generadores basados en LFSR se han mostrado cada vez más ineficientes debido al aumento progresivo de la longitud de palabra de los procesadores y microcontroladores, que se encuentran actualmente en un rango que abarca desde los 8 bits de los controladores más sencillos hasta los 64 bits de los computadores portátiles y de sobremesa. Con objeto de aprovechar la capacidad de estos procesadores, se están utilizando actualmente LFSR defnidos sobre cuerpos extendidos $GF(2^n)$. De este modo, cada iteración del dispositivo genera n bits en lugar de uno solo como ocurre cuando se trabaja en $GF(2)$.

En cualquier caso, tanto sobre $GF(2)$ como sobre $GF(2^n)$, las implementaciones basadas en LFSR generan números con distribución de probabilidad uniforme, razón por la cual presentan gran interés criptográfico. Sin embargo, recientemente, la criptografía cuántica, a través de un tipo concreto de esquema cuántico de distribución de claves (QKD), está ganando protagonismo. Este tipo de esquemas necesita generar números con una distribución Gaussiana.

Con el objetivo de aprovechar la simplicidad de los LFSR y de continuar con la tradicional utilización de los mismos en criptografía, se estudia en esta tesis la posible generación de números pseudoaleatorios con distribución Gaussiana partir de LFSR definidos en $GF(2^n)$.

A partir de propuestas de otros autores, con resultados poco satisfactorios, se ha desarrollado un generador de números pseudoaleatorios con distribución Gaussiana basado en LFSR sobre $GF(2)$. Este nuevo generador mejora todas los diseños anteriores basados en LFSR.

Utilizando este nuevo generador en $GF(2)$ se ha diseñado otro sobre $GF(2^n)$, que presenta un excelente comportamiento mejorando las implementaciones sobre $GF(2)$ y a otras no basadas en LFSR que también se utilizan con frecuencia.

Como consecuencia del estudio de los LFSR en cuerpos extendidos realizado en esta tesis, se ha formalizado la relación con los LFSR definidos en $GF(2)$ mediante un modelo equivalente simplificado, que permite expresar el LFSR en $GF(2^n)$ como una combinación de secuencias entrelazadas generadas por un conjunto de LFSR en $GF(2)$.

Calificación del Tribunal

Tesis Doctoral: GENERACIÓN DE SECUENCIAS PSEUDOALEATORIAS GAUSSIANAS MEDIANTE REGISTROS DE DESPLAZAMIENTO CON REALIMENTACIÓN LINEAL EN CUERPOS EXTENDIDOS.

Autor: Guillermo Cotrina Cuenca.

Director y Tutor: Alberto Peinado Domínguez.

Director: Andrés Ortiz García.

El tribunal nombrado para juzgar la Tesis arriba indicada, compuesto por los siguientes doctores:

Presidente:

Vocales:

Secretario:

acuerdan otorgarle la calificación de:

El secretario del tribunal

Fecha

Índice general

Índice de figuras

Índice de Tablas

Glosario

A-D test de Anderson-Darling. 137–139, 141, 174

AND operador lógico "Y". 25

B-M método de Box-Muller para la generación de números con distribución Guassiana. 143, 144, 174, 175, 180

CDF función de distribución acumulada (cumulative distribution function). 173

CDMA multiplexación por división de código. 7

CHI test de la Chi-cuadrado. 138–141, 174

CLT teorema central del Límite. 5, 128, 130, 136, 152, 163, 166–168, 176, 179

CV-QKD QKD de variables continuas. 7, 63

DV-QKD QKD de variables discretas. 7, 63

GCD máximo común divisor. 34, 35, 43, 45

K-S test de Kolmogorov-Smirnov. 137, 139

LC complejidad lineal. 164

LFSR registro de desplazamiento con realimentación lineal. 4–11, 15, 17, 18, 41, 49, 115, 116, 120, 122, 125, 127–133, 136, 141, 143–146, 150–155, 157, 159, 162–168, 170, 175, 176, 179

PID dominio de ideales principales. 34, 35, 37

PRNG generador de números pseudoaleatorios. 4, 6, 49, 54, 128–134, 143, 144, 165, 167, 174–176, 179–181

QKD esquema cuántico de distribución de claves. 7, 8, 60, 62, 165, 180

RFID identificación por radiofrecuencia. 5

S-W test de Shapiro-Wilk. 138, 141, 174

SA simulated annealing. 147, 149, 150

TRNG generador de números puramente aleatorios. 5

UFD dominio de factorización única. 34, 35, 45

XOR suma módulo 2. 25, 54, 59, 129–131

Notación

$\mathbb{N}$	Semianillo de los número naturales
$\mathbb{Z}$	Anillo de los números enteros
$\mathbb{Q}$	Cuerpo de los números racionales
$\Re$	Cuerpo de los números reales
$\mathbb{C}$	Cuerpo de los números complejos
$\| \vec{v} \|$	Norma del vector $\vec{v}$
$\langle \vec{v}, \vec{w} \rangle$	Producto escalar de los vectores $\vec{v}$ y $\vec{w}$
$\|G\|$ ó $\#(G)$	Cardinal del Conjunto G
$\|A\|$	Determinante de la matriz cuadrada A
$\det(A)$	Determinante de la matriz (cuadrada) A
A^t	Matriz Transpuesta de A
A^{-1}	Inversa de la matriz A
c.t.p.	En casi todos los puntos
c.q.d.	Como queríamos demostrar
■	Como queríamos demostrar
□	Fin de la solución
e.o.c.	En cualquier otro caso
e	número e
E^x	Exponencial compleja

$Re(z)$	Parte real
$Im(z)$	Parte imaginaria
$sen(x)$	Función seno
$tan(x)$	Función tangente
$arctg(x)$	Función arco tangente
$sa(x)$	Función sampling
$sgn(x)$	Función signo
$ln(x)$	Función Logaritmo Natural
$log(x)$	Función Logaritmo Decimal
$log_b(x)$	Función Logaritmo en base b
$\frac{\partial x}{\partial y}$	Derivada parcial de x respecto a y
C_{XY}	covarianza de dos variables aleatorias reales X e Y
ρ_{XY}	Coeficiente de correlación de las variables aleatorias reales X e Y
$F_X(\cdot)$	Función de distribución de la variable aleatoria X
$f_X(\cdot)$	Función densidad de probabilidad de la variable aleatoria X
$p_X(\cdot)$	Función masa de probabilidad de la variable aleatoria discreta X
R_{XY}	correlación de dos variables aleatorias reales X e Y
$\Pr(A)$	Probabilidad del suceso A
$E[X]$	Valor esperado de la variable aleatoria X

$V(X)$	Varianza de la variable aleatoria X
$F_X(x)$	Distribuido siguiendo la función densidad de probabilidad $F_X(x)$
$X \rightsquigarrow N(\mu, \sigma)$	Distribución gaussiana para la variable aleatoria X, de media μ y varianza σ
Id_n	Matriz identidad de dimensión n
$Diag\vec{x}$	Matriz diagonal a partir del vector $\vec{x}$
$Diag\vec{A}$	Vector diagonal de la matriz A
snr	Signal-to-noise ratio
mse	Minimum square error
$/$	Tal que
$eqdef$	Igual por definición
$\left\|A\right\|$	Cardinal, número de elementos del conjunto A
D_x	Diferencial de x
$\leq$	Menor o igual
$\geq$	Mayor o igual
$\backslash$	Backslash
$\Longleftrightarrow$	Si y sólo si
$\frac{a}{b}$	Fracción con estilo pequeño, a/b
Δ	Incremento
$\rightarrow x$	Tiende hacia x
Ord	Orden
$E[x]$	Esperanza matemática de x
$\sigma^2(X)$	Varianza de x

A mi familia,

ya que sin ella

este trabajo no habría sido posible.

Agradecimientos

Antes de comenzar este documento desearía agradecer a mis tutores, D. Alberto Peinado y D. Andrés Ortiz, toda la ayuda prestada, sin la cual, la realización de esta tesis no hubiera sido posible.

Motivación personal

Me gustaría antes de comenzar con los fundamentos que justifican este documento el por qué me he decidido a realizar esta tesis doctoral. Desde que comencé mis estudios de matemáticas siempre me he sentido atraído por la investigación y más concretamente por el desarrollo de nuevas herramientas que ayuden al progreso dentro de nuestra sociedad en la que vivimos. Es por ello, que a medida que fui descubriendo los fundamentos de la matemática me sentí muy atraído por la criptografía y por la seguridad en las comunicaciones. Dentro de las posibilidades que me brindaba la universidad de Málaga, completé mis estudios en la rama de Fundamentales donde se pueden analizar las estructuras algebraicas que posteriormente se pueden utilizar como sustento de los alfabetos y medios de comunicación utilizados en la criptografía. Una vez concluidos mis estudios de la Licenciatura en Matemáticas comencé a realizar la tesis doctoral a través de la UNED, Universidad Nacional de Educación a Distancia, al mismo tiempo que desempeñaba mi labor docente como profesor de Matemáticas y de Informática en la Universidad

Les Roches Marbella, en la facultad de turismo, a su vez dependiente de la Universidad Europea de Madrid. En ella comencé a llevar a cabo mi tesis, no obstante la gran carga de trabajo me llevó a no poder consagrar el tiempo necesario para poder completar la tesis. En consecuencia, decidí iniciar una nueva singladura en la enseñanza de Secundaria. Una vez superado el concurso oposición, decidí retomar la tesis doctoral, no obstante debido a los cambios en los planes de estudio, tuve que comenzar desde cero en lo académico y es por ello que realicé el Máster de Matemáticas que me capacitaba para comenzar mis estudios de tercer ciclo. Al mismo tiempo que desarrollaba mi labor docente como profesor de secundaria, he seguido formándome en distintos aspectos entre los que destaco, los idiomas donde he logrado el nivel C1 tanto en Inglés como en Francés, he sido ponente en diversos cursos de formación y desarrollo del profesorado, he realizado los cursos de acceso la función directiva, he llevado a cabo mi labor como ponente de la PEVAU,antigua selectividad, en la materia de matemáticas aplicadas a las CCSS II, etc.

Siempre he encontrado muy interesante la labor de investigación y de divulgación de los logros obtenidos. A lo largo de este periodo en el que se ha desarrollado esta tesis doctoral, he asistido a distintos eventos en los que he podido contrastar con los compañeros los desarrollos llevados a cabo. Al mismo tiempo, he presentado los desarrollos llevados a cabo en distintas ponencias para poder compartir de este modo los desarrollos realizados, etc. Paralelamente, he podido publicar distintos artículos en distintas revistas, contribuyendo de esta manera a la difusión del conocimiento científico.

Parte I

Introducción y Estado del Arte

Capítulo 1

Introducción

1.1. Justificación y relevancia de la tesis

La generación de números pseudoaleatorios constituye un pilar esencial para el desarrollo de un amplio rango de aplicaciones en numerosos ámbitos técnicos y científicos. Los sistemas criptográficos, la comprobación de circuitos electrónicos, la transmisión de datos, la programación de juegos o las simulaciones de procesos industriales, financieros o meteorológicos hacen uso de estos generadores [9]. La búsqueda de algoritmos, métodos y sistemas capaces de generar números pseudoaleatorios comienza en los años 40 del siglo XX y continúa hoy día tratando de mejorar, entre otras, las soluciones actuales que dan soporte a los sistemas criptográficos que protegen las comunicaciones y la privacidad de los ciudadanos [31; 39; 75].

Las sucesiones de números aleatorios (o puramente aleatorios) son sucesiones en las que cada término se obtiene al azar, con una determinada probabilidad de que un rango de valores sea elegido. A pesar de que la existencia de números aleatorios es demostrable gracias a los axiomas probabilísticos de Kolmogorov [85], este resul-

tado no proporciona un método para la construcción de una sucesión de números aleatorios. Los primeros intentos de generar estas secuencias se basaron en procesos físicos mediante lanzamientos de dados, giros de ruletas, repartos de cartas de juego o extracción de datos de una tabla de doble entrada. En los años 40 (s.XX) comenzó el desarrollo de máquinas para producir números pseudoaleatorios, esto es, números con apariencia de aleatorios pero generados mediante métodos deterministas, que podían ser generados de forma más eficiente que los puramente aleatorios, pero con efectos casi idénticos en su utilización. A partir de los años 50, comenzaron a utilizarse ordenadores como generadores de números pseudoaleatorios, pocos años después de que John Von Neumann [84] propusiera, en la década de los años 40, el conocido método del cuadrado medio [87].

Si bien la simulación de procesos requiere la utilización de números aleatorios (pseudoaleatorios, en su defecto) para reproducir con la mayor precisión las condiciones de un entorno real —*nótese que los procesos naturales presentan comportamientos aleatorios*—, en otros casos, como ocurre en los sistemas criptográficos, la aleatoriedad de los números proporciona seguridad gracias a la característica implícita de no reproducibilidad de este tipo de secuencias. Estos números aleatorios están presentes en la mayor parte de los protocolos criptográficos, entre los que destacan los procesos de autenticación de usuarios y contenidos [66; 67]. Sin embargo, cierto tipo de sistemas criptográficos, conocidos como cifradores en flujo [93], basan su funcionamiento en la generación de secuencias pseudoaleatorias debido a la necesidad de ser reproducidas de forma idéntica en el transmisor y receptor de los mensajes, al tiempo que presentan la mayor aleatoriedad aparente posible.

Aunque son muchos los generadores de números pseudoaleatorios (PRNGs) propuestos hasta la fecha, los LFSR continúan siendo la pieza fundamental de la gran mayoría de los utilizados para el cifrado en flujo [10], como se puede apreciar en

las actuales implementaciones de sistemas de comunicaciones móviles e inalámbricas como Bluetooth [74], IEEE 802.11 WLAN [90], GSM [38] o LTE [83]. Los LFSR aparecen también en algunos generadores de números puramente aleatorios (TRNGs) [68] como los utilizados en los sistemas de identificación por radiofrecuencia (RFID) [64], siguiendo las recomendaciones de la especificación EPC [28].

La profusa utilización del LFSR en el cifrado en flujo se debe principalmente a su simplicidad, su bajo coste de implementación, buen comportamiento estadístico y la posibilidad de utilizar un modelo matemático que permite configurar el generador para un rendimiento óptimo [34], esto es, para generar secuencias de longitud máxima con apariencia aleatoria. En concreto, la longitud máxima de la secuencia generada por un LFSR de m celdas es $2^m - 1$ y se obtiene cuando la realimentación queda caracterizada por un polinomio primitivo de grado m [34].

No obstante, estas secuencias adolecen de una alta previsibilidad de tal manera que la secuencia completa se puede reproducir si un observador obtiene acceso a $2m$ bits. A pesar de ello, el LFSR sigue siendo una parte importante de los generadores criptográficos porque estas secuencias se utilizan para derivar en otras más robustas pero manteniendo las propiedades estadísticas originales. Se aplican dos métodos principales para corregir esa debilidad: combinaciones no lineales y filtrados no lineales. El primero combina varios LFSR, generalmente con diferente número de celdas [38], y el segundo se suele aplicar sobre un único LFSR cuya secuencia es procesada (filtrada) por una función no lineal [83].

Otra de las características principales de los LFSR es que las secuencias generadas tienen una distribución estadística uniforme. Sin embargo, existen algunas propuestas que se alejan de su uso habitual y utilizan el LFSR para generar secuencias de números con una distribución Gaussiana de probabilidad, tratando de aprovechar el teorema central del Límite (CLT) [92] para generar distribuciones

Gaussianas a partir de distribuciones uniformes. Por este motivo, la implementación más directa utiliza diferentes LFSR para generar las distintas secuencias uniformes. El ejemplo más reciente está descrito por Thomas [81]. Con objeto de disminuir el coste de la implementación, la mayoría de las propuestas para generar secuencias Gaussianas utilizan un único LFSR para generar las distintas secuencias uniformes [68; 78].

Así, en 2010, Kang [44] propone un PRNG Gaussiano, usando un LFSR de longitud $N = 4M$ bits, para generar números pseudoaleatorios de $(M + 4)$ bits, por medio de un acumulador que suma los 8 resultados consecutivos obtenidos de la suma de los cuatro números de longitud M que resultan al trocear los N del LFSR cada N instantes de tiempo. De este modo, se obtienen secuencias de longitud $(2N - 1)/8N$. Para evitar el sobredimensionamiento del LFSR utilizado en el generador de Kang, Condo y Gross [14] proponen un PRNG Gaussiano usando permutaciones sobre los estados sucesivos de un LFSR único, reduciendo así el coste de implementación. El principal inconveniente es que no todas las posibles permutaciones producen números con la distribución Gaussiana requerida y, en consecuencia, se debe realizar un gran esfuerzo de cálculo para encontrar las permutaciones adecuadas.

La utilización habitual de LFSR en cifradores para comunicaciones de alta velocidad ha puesto de manifiesto la ineficiencia de las implementaciones software de PRNG basados en LFSR debido al aumento progresivo de la longitud de palabra de los procesadores, que se encuentran actualmente en un rango que abarca desde los 8 bits de los controladores más sencillos hasta los 64 bits de los computadores portátiles y de sobremesa.

Con objeto de aprovechar la capacidad de estos procesadores, se están utilizando actualmente LFSR definidos sobre cuerpos extendidos $GF(2^n)$ [22; 23; 24]. De

este modo, cada iteración del dispositivo genera n bits en lugar de uno solo como ocurre cuando se trabaja en $GF(2)$, manteniendo en todo caso las características pseudoaleatorias y la distribución uniforme de las secuencias generadas. Un ejemplo de utilización de LFSR en $GF(2^n)$ se encuentra en el cifrado en flujo que se utiliza en el estándar de comunicaciones móviles LTE [20].

Por otra parte, en los últimos tiempos la criptografía cuántica está ganando protagonismo al presentarse como alternativa a los sistemas criptográficos actuales cuya seguridad podría verse comprometida gracias a la capacidad de cálculo que ofrecen los primeros prototipos de computadores cuánticos [47].

Actualmente, todos los países desarrollados han implementado esquemas QKD [33; 52; 71; 73], algunos de manera experimental (en ambientes controlados) y otros en sus redes de tránsito actuales [82]. Los primeros esquemas QKD se basaron en la transmisión de fotones polarizados utilizando estados no ortogonales. Estos esquemas, conocidos como esquemas QKD de variables discretas (DV-QKD) [6; 41], requieren la utilización de componentes específicos para la detección de un fotón único. Como alternativa, se han desarrollado esquemas QKD de variables continuas (CV-QKD) [5; 26; 50] para transportar información sobre algunas propiedades continuas de la luz, como los valores de los componentes en cuadratura de un estado coherente. Estos esquemas, que están siendo utilizados en China, Japón, España e Italia [82], utilizan técnicas de detección coherente y, como consecuencia, presentan un menor coste de implementación debido a la utilización de componentes de comunicaciones estándar. Además, emplean modulación Gaussiana para enviar valores aleatorios de amplitud y fase que deben generarse siguiendo una distribución Gaussiana [33; 80].

Cabe destacar, por otro lado, la importancia de la generación pseudoaleatoria con distribución Gaussiana en los sistemas de telecomunicación con objeto de si-

mular el comportamiento de una señal frente al ruido blanco Gaussiano aditivo que aparece y caracteriza gran parte de los canales de comunicación.

1.2. Propuesta de investigación y objetivos

Por todo ello, se plantea realizar una investigación centrada en el estudio de los LFSR definidos en cuerpos extendidos como posibles generadores de secuencias pseudoaleatorias con distribución de probabilidad de tipo Gaussiano. Este estudio pretende proporcionar una solución eficaz y eficiente que mejore las limitaciones de los generadores basados en LFSR propuestos hasta la fecha, comenzado por las propuestas realizadas sobre $GF(2)$. Asimismo, es interesante destacar que este estudio se encuentra alineado con el uso habitual de estos dispositivos criptográficos caracterizados por su bajo coste de implementación, facilidad de diseño y velocidad de ejecución, así como con la creciente relevancia que está tomando la criptografía cuántica, a través de los QKD.

La investigación propuesta plantea los siguientes objetivos:

- Diseñar un generador de números pseudoleatorios con distribución Gaussiana utilizando LFSR en $GF(2)$ cuyo comportamiento mejore las propuestas anteriores, tanto las que utilizan LFSR como las que no. Además, el diseño objetivo debería ser lo suficientemente genérico para ajustarlo a números de distinto tamaño (en bits) y permitir una ágil configuración de los parámetros óptimos.
- Obtener un modelo equivalente del LFSR en cuerpos extendidos que facilite el estudio de las secuencias que genera a partir de los resultados que se obtienen en el LFSR binario.
- Diseñar un generador de números pseudoleatorios con distribución Gaussia-

na utilizando LFSR en $GF(2^n)$. Como punto de partida se propone la generalización del caso binario y la utilización del modelo equivalente obtenido previamente. Al igual que en $GF(2)$, el diseño de este generador debería ser lo suficientemente genérico para ajustarlo a números de distinto tamaño (en bits) y permitir una ágil configuración de los parámetros óptimos.

Estos objetivos concretos están orientados a proporcionar métodos y herramientas que faciliten posteriormente la implementación de generadores en dispositivos de capacidades limitadas. Al mismo tiempo, el modelo equivalente que se pretende obtener tendría aplicaciones directas en el criptoanálisis de cifradores en flujo basados en LFSR sobre cuerpos extendidos.

1.3. Contribuciones

A continuación, se relacionan las principales contribuciones de la presente tesis:

- Generador pseudoaleatorio Gaussiano basado en LFSR sobre $GF(2)$. A partir de propuestas de otros autores que utilizan LFSR para generar secuencias de números pseudoaleatorios con distribución Gaussiana mediante permutaciones aplicadas a un único LFSR, se ha diseñado un nuevo generador que mejora notablemente a todos los existentes, en cuanto a las propiedades de las secuencias generadas y al mismo tiempo permite seleccionar las configuraciones óptimas de un modo sencillo, a diferencia de los anteriores. El nuevo generador utiliza rotaciones cíclicas, en lugar de permutaciones genéricas. El diseño de este generador ha sido publicado en la revista *Sensors*: (G. Cotrina, A. Peinado, A. Ortiz, "Gaussian Pseudorandom Number Generator Based on Cyclic Rotations of Linear Feedback Shift Registers", *Sensors*, **20**, article 2103, (2020); doi:10.3390/s20072103) [18].

- Modelo equivalente de LFSR en $GF(2^n)$. Con objeto de trasladar los resultados obtenidos en $GF(2)$ a un cuerpo extendido $GF(2^n)$, se ha formalizado un modelo equivalente que permite representar un LFSR de L celdas sobre $GF(2^n)$ mediante el entrelazado de n LFSR en $GF(2)$ de nL celdas cada uno, controlados todos por el mismo polinomio primitivo de realimentación. Este modelo ha sido publicado en la revista *Mathematics*: (G. Cotrina, A. Peinado, A. Ortiz, "Gaussian Pseudorandom Number Generator using Linear Feedback Shift Registers in extended fields", *Mathematics*, **9**, article 556, (2021); doi:10.3390/math9050556) [19].
- Generador pseudoaleatorio Gaussiano basado en LFSR sobre $G(2^n)$. El modelo descrito anteriormente ha permitido diseñar un generador sobre $GF(2^n)$ utilizando las relaciones implícitas de las secuencias binarias subyacentes que componen la secuencia final generada por el LFSR definido sobre $GF(2^n)$. El diseño de este generador ha sido publicado en la revista *Mathematics*: (G. Cotrina, A. Peinado, A. Ortiz, "Gaussian Pseudorandom Number Generator using Linear Feedback Shift Registers in extended fields", *Mathematics*, **9**, article 556, (2021); doi:10.3390/math9050556) [19].
- Algoritmo para la selección de polinomios primitivos en $GF(2^n)$. El modelo equivalente desarrollado para representar LFSR definidos en $GF(2^n)$ mediante un conjunto de LFSR en $GF(2)$, ha contribuido a desarrollar un algoritmo que permite identificar polinomios primitivos sobre $GF(2^n)$, necesarios para determinar la realimenación de los LFSR que se utilizan en el generador propuesto. El algoritmo de identificación de polinomios ha sido publicado en el congreso nacional de criptografía (RECSI): (G. Cotrina, A. Ortiz, A. Peinado, Ïdentificación de polinomios primitivos sobre cuerpos extendidos mediante análisis de secuencias entrelazadas", *Actas de la XV Reunión Española de*

Criptología y Seguridad de la Información (RECSI), pp. 66-70, Granada, 3-5 octubre 2018, ISBN: 978-84-09-02463-6) [17].

- Generación de secuencias criptográficas de alta complejidad lineal. La estructura del generador Gaussiano basado en un LFSR sobre $GF(2)$ ha permitido aprovechar su no linealidad para obtener secuencias binarias, generadas seleccionando algunos bits de cada número pseudoaleatorio generado, que presentan una elevada complejidad lineal (en torno a 2^{n-1}); aspecto que resulta de gran utilidad para el diseño de cifradores en flujo. Estos resultados han sido publicados en el congreso nacional de criptografía (RECSI): (G. Cotrina, A. Peinado, A. Ortiz, "Utilización de un generador con distribución gaussiana para aumentar la complejidad lineal de las m-secuencias", *Actas de la XIV Reunión Española sobre Criptología y Seguridad de la Información*, pp. 42-46, Mahón, Menorca, 26-28 octubre 2016, ISBN: 978-84-608-9470-4) [16].

1.4. Estructura del documento

El documento está divido en 5 partes. Tras una primera que contiene la introducción, en la segunda se presentan las bases algebraicas que, partiendo de la definición de grupos hasta llegar a las estructuras de los cuerpos, proporcionan el soporte necesario para el desarrollo de los generadores propuestos y el análisis de los secuencias resultantes. La tercera parte introduce los conceptos básicos de la criptografía y los aspectos más relevantes que la relacionan con los LFSR, la generación pseudoaleatoria y los esquemas cuánticos de distribución de clave. La cuarta parte presenta los fundamentos de los LFSR que permiten modelar su comportamiento y conocer las propiedades aleatorias de las secuencias que generan. Por último, la parte quinta contiene los resultados y las contribuciones de la presente tesis.

Parte II

Preliminares

Capítulo 2

Álgebra Abstracta

El rol del álgebra abstracta es fundamental dentro de muchas áreas de la ciencia y la ingeniería. En este capítulo vamos a describir las estructuras algebraicas básicas que dan soporte a la generación y análisis de las secuencias pseudoaleatorias [8; 13; 29; 30; 53; 54; 59].

2.1. Grupos

Los grupos son una de las estructuras más sencillas en el álgebra moderna. Los grupos han sido desarrollados para satisfacer la necesidad de desarrollar una gran lista de aplicaciones, entre las que se pueden destacar la criptografía, la física, la química o la biología.

Definición 2.1.1. Un grupo es un conjunto G con un elemento especial e (llamado identidad) junto con una operación $*$ que verifica los siguientes axiomas:

- *Propiedad asociativa.* $\forall a, b, c \in G$ se tiene que $a * (b * c) = (a * b) * c$.
- *Elemento Identidad* $\forall a \in G$ se tiene que $a * e = e * a = a$.
- *Elemento Opuesto o Inverso.* $\forall a \in G, \exists b \in G$ tal que $a * b = b * a = e$. Dicho

elemento b se llamará *Opuesto o Inverso de a.*

El **orden** del grupo G se denotará por $|G|$ y representará su cardinal como conjunto [13].

Definición 2.1.2. Sea G un grupo dotado de una operación $*$. Supongamos que se verifica el siguiente axioma:

- *Propiedad Conmutativa.* $\forall a, b \in G$ se tiene que $a * b = b * a$.

Entonces se dirá que el grupo es Conmutativo o Abeliano.

Proposición 2.1.1. *Sea G un grupo. Entonces se verifican las siguientes propiedades:*

1. *Si $a, b \in G$ y se verifica que $a * b = e$ entonces $b * a = e$*
2. *Cada $a \in G$ tiene un único inverso.*
3. *El elemento identidad $e \in G$ es único.*

Algunas veces utilizaremos la notación multiplicativa en los grupos. Y por lo tanto denotaremos el inverso de un elemento a mediante a^{-1}. Del mismo modo para simplificar la notación en lugar de escribir $a * b$ escribiremos directamente ab. Para la potenciación, de a^0 diremos que es e y por inducción definiremos $a^n = aa^{n-1}$. Por lo tanto $a^n a^m = a^{n+m}$ y $(an)^m = a^{n \cdot m}$

Si G es un grupo abeliano es frecuente utilizar la notación aditiva " $+$ " en lugar de " $*$ " y utizar la notación de elemento opuesto $(-a)$ en lugar de la notación de elemento inverso a^{-1}.

De esta forma $a + b = a + (-b)$ y 0 lo usaremos en lugar de e. Así se definirá por inducción $0a = 0$ y análogamente $na = a + (n-1)a$ para cualquier entero positivo n. En consecuencia $na + ma = (n+m)a$ y $n(ma) = (nm)a$.

Algunas veces escribiremos $e = e_G$ cuando consideremos varios grupos.

Ejemplo 1. Veamos algunos ejemplos

1. El conjunto de los enteros $\mathbb{Z}$ con neutro, el número entero 0, es un grupo Abeliano.
2. El conjunto de los números racionales $\mathbb{Q}$ con elemento neutro $\frac{0}{1}$ (o cualquiera equivalente a esta fracción) es un grupo Abeliano.
3. El conjunto de los números racionales (sin el 0 denotado por $\mathbb{Q}^*$) es un grupo Abeliano donde el elemento neutro es el 1.
4. Si S es cualquier conjunto. El conjunto de todas las permutaciones de S es un grupo con respecto a la operación de la composición. El orden de este grupo de permutaciones es $|S|!$.
5. Para $n \geq 1$, el conjunto de las matrices cuadradas de orden n con determinante no nulo y coeficientes racionales es un grupo no necesariamente conmutativo con respecto al producto usual de matrices. El elemento neutro es la matriz Id_n.
6. Si $N \geq 2$ y a, b son enteros, entonces diremos que a es congruente con $b \equiv N$ y se escribirá $a \equiv bmod(N)$, si N divide a $a-b$. La relación así definida es una relación de equivalencia en $\mathbb{Z}$. Vamos a denotar por $\mathbb{Z}/N$ el conjunto de todas las clases de equivalencia para esta relación. Esto es $\mathbb{Z}/N = \{a+Nb : b \in \mathbb{Z}\}$. Entonces $\mathbb{Z}/N$ es un grupo Abeliano con la operación, $(a+N\mathbb{Z})+(b+N\mathbb{Z}) = (a+b)+N\mathbb{Z}$ y $0+\mathbb{Z}$ será el elemento neutro. Se puede demostrar fácilmente que este conjunto es un grupo y que las operaciones están bien definidas.

Teorema 2.1.2. *Sea $a \in G$ un elemento en un grupo finito. Entonces bajo esta condición se verifica que $a^{|G|} = e$.*

2.2. Subgrupos

En esta sección examinaremos los subgrupos de un grupo que heredan la estructura de grupo por si solos.

Definición 2.2.1. Sea G un grupo y sea $H \subseteq G$. Diremos que H es un subgrupo de G si H es un grupo con la misma operación definida en G y además posee el mismo elemento neutro [13].

Esto significa que H es un subconjunto de G y además:

1. $e \in H$
2. Si $a, b \in H$ entonces $a + b \in H$
3. Si $a \in H$ entonces $\exists a^{-1} \in H$
4. Además si G es Abeliano entonces lo será también H

Ejemplo 2. Veamos algunos ejemplos.

- El grupo aditivo de los enteros es un subgrupo de los números racionales.
- El conjunto de las permutaciones cíclicas de $\{1, 2, \cdots, n\}$ es un subgrupo del grupo de todas las permutaciones.
- Si G_1, G_2 son dos grupos con operaciones $*_1, *_2$. Entonces el producto cartesiano de $G_1 \times G_2 = \{(a, b) : a \in G_1, b \in G_2\}$ es un grupo con la operación, $(a, b) * (c, d) = (a *_1 c, b *_2 d)$ y con elemento neutro (e_1, e_2).

Sea G un conjunto dotado de una operación interna y sea $a \in G$. Denotemos por $\langle a \rangle$ al conjunto $\{a^i : i \in \mathbb{Z}\}$. Este conjunto es un subgrupo Abeliano de G llamado subgrupo generado por a. Si $|\langle a \rangle| < \infty$ entonces diremos que el orden de a es dicha cardinalidad. En caso contrario diremos que a tiene orden infinito. Equivalentemente, el orden de a es el menor entero positivo $k > 0$ tal que $a^k = e$ si es que este k existe [54].

Un grupo G es cíclico si $G = \langle a \rangle$ para algún a, en cuyo caso diremos que a es un generador de G. Cada grupo cíclico es Abeliano. El grupo $\mathbb{Z}/N$ es cíclico y un generador es el 1. Nos referiremos a el con el grupo cíclico de orden N

Lema 2.2.1. *Sean $a, b \in \mathbb{Z}$. Entonces existen al menos dos enteros s, t tales que $m.c.d(a, b) = sx + ty$*

Teorema 2.2.2. *Cada subgrupo de un grupo cíclico es cíclico. Si además suponemos que $\langle a \rangle$ es un grupo cíclico y finito de orden n entonces se verifican:*

1. *Si k es un entero positivo, entonces $\langle a^k \rangle$ es un subgrupo de $\langle a \rangle$ de orden $\frac{n}{m.c.d.(n,k)}$*
2. *Si $d|n$ y además $d > 0$ entonces $\langle a \rangle$ contiene un subgrupo de orden d*
3. *Si $d|n$ y $d \geq 0$ entonces $\langle a \rangle$ contiene $\varphi(d)$ elementos de orden d. La función φ de Euler es aquella función que determina todos los enteros positivos relativamente primos con uno dado.*
4. *$\langle a \rangle$ contiene $\phi(d)$ elementos de orden d*

Ejemplo 3. Por ejemplo, el grupo $\mathbb{Z}$ es un grupo cíclico con generador el 1. Por lo que todo subgrupo será de la forma $m\mathbb{Z} = \{mk : k \in \mathbb{Z}\}$ para algún entero m

2.3. Homomorfismos

Más generalmente, estudiaremos las relaciones entre grupos sobre todo cuando se preserven las estructuras algebraicas y las operaciones.

Definición 2.3.1. Sean G y H dos grupos. Una función $f : G \longrightarrow H$ es un homomorfismo si preserva las operaciones entre los grupos. Esto es $\forall a, b \in G$ se tiene que $f(ab) = f(a)f(b)$. La imagen de f la denotaremos mediante $Im(f)$, y se verificará $Im(f) = \{b \in H : \exists a \in G \text{ con } f(a) = b\}$. El núcleo de f que denotaremos

por $ker(f)$ es el conjunto $\{a \in G : f(a) = e_H\}$. Si f es un homorfismo y además se verifica que $G = H$ entonces diremos que el homomorfismo es un endomorfismo. Si f es inyectiva se dirá que es un monomorfismo. Si f es sobreyectiva si dirá que es un epimorfismo. Si f es biyectiva se dirá que es un isomorfismo. Finalmente si f es un isomorfismo y un endomorfismo entonces diremos que f es un automorfismo.

Sea G un grupo y sea $a \in G$, entonces podemos definir una función $\phi(n) = a^n$. Esta función es un homomorfismo y además es un monomorfismo sii a tiene un orden finito. Si a tiene un orden finito m, entonces ϕ induce un monomorfismo de $\mathbb{Z}/m\mathbb{Z}$ en G. En particular, cada grupo cíclico de orden no finito es isomorfo a $\mathbb{Z}$ y cada grupo cíclico finito es isomorfo al grupo aditivo $\mathbb{Z}/(m)$ donde m es el orden de cualquier generador [30].

Proposición 2.3.1. *Sea $\phi : G \to H$ un homomorfismo. Entonces ϕ preserva el neutro y el inverso. Además el $ker(\phi)$ es un subgrupo de G e $Im(\phi)$ es un subgrupo de H.*

Proposición 2.3.2. *Sean $\phi : F \to G$ y $\varphi : G \to H$ dos homomorfismos. Entonces $\varphi \circ \phi$ es un homomorfismo.*

Definición 2.3.2. Sean $\phi : F \to G$ y $\varphi : G \to H$ dos homomorfismos. Diremos que estos homomorfismos son exactos en G si el núcleo de φ es igual a la imagen de ϕ. Una sucesión de aplicaciones:

$$1 \to F \to G \to H \to 1 \tag{2.1}$$

es una sucesión exacta corta si es exacta en $F, G,$ y H. En este caso 1 denotará al grupo trivial con un solo elemento.

En la ecuación (2.1) la exactitud en F es equivalente a que ϕ sea inyectiva y la exactitud en H es equivalente a que φ sea sobreyectiva.

Proposición 2.3.3. *Si* $1 \to F \to G \to H \to 1$ *es una sucesión exacta corta donde todos los grupos son finitos entonces se verifica* $|G| = |F| \cdot |H|$.
Si en la sucesión exacta corta definida en 2.1 se verifica que existe un homomorfismo $\mu : H \to G$ *y que además se verifica que* $g \cdot h = e$ *entonces diremos que la sucesión exacta corta "split"*

Proposición 2.3.4. *Sea* $1 \to F \to G \to H \to 1$ *es una sucesión exacta corta. Supongamos además que* ϕ *va de* F *en* G *y* φ *de* G *en* H. *Supongamos además que los tres grupos son Abelianos y que la sucesión exacta corta se divide vía un homomorfismo* $\mu : H \to G$. *Entonces existe un isomorfismo entre* $F \times G$ *y* G *dado por* $(a, b) \to \phi(a)\mu(b)$ *Recíprocamente si* $G = F \times H$ *entonces existe una sucesión exacta corta como en 2.1, donde* φ *es la proyección canónica y* $\phi(a) = (a, 1)$

2.4. Cocientes

Si m es un entero positivo, entonces el conjunto de los múltiplos de $m = m\mathbb{Z}$ es un subrgupo aditivo de $\mathbb{Z}$. En la sección 2.2 el grupo cociente $\mathbb{Z}/n\mathbb{Z}$ está definido como el conjunto de las clases de equivalencia de $\mathbb{Z}$ bajo la relación $a \equiv b\ mod(n)$ si $a - b \in m\mathbb{Z}$ [4].
Más generalmente, supongamos que G es un grupo y H es un subgrupo suyo. Definimos una relación de equivalencia afirmando que $a \equiv b$ sii $\exists h \in H : b = ah$. Esto es obviamente una relación de equivalencia. La clase de equivalencia de a es $aH = \{ah : h \in H\}$ y se conoce como clase lateral por la izquierda de a. El conjunto de las clases laterales se denota por G/H. En general no es posible en todos los casos formar un grupo de este tipo de clases laterales [13].

De hecho podríamos haber empezado definiendo $a \equiv' b$ si existe $h \in H$ tal que $b = ha$. Esto también es una relación de equivalencia. La clase de equivalencia de Ha de a con respecto a esta relación se llamará clase lateral por la derecha de a. El conjunto de las clases se denotará $H \setminus G$. Si el grupo G es Abeliano entonces $Ha = aH \ \forall a \in G$. Más generalmente:

Definición 2.4.1. Si H es un subgrupo de un grupo G, entonces se dirá que H es normal en G si $\forall a \in G$ se tiene que $aH = Ha$ o lo que es lo mismo $aha^{-1} \in G \ \forall h \in H$.

Teorema 2.4.1. *Si H es normal en G, entonces $G/H = H \setminus G$ es un subgrupo bajo la operación $(aH)(bH) = abH$*

En este caso, G/H se llama grupo cociente módulo H. La aplicación natural definida $f : G \to G/H$ tal que a cada $a \to aH$ es un homomorfismo. Si el conjunto de las clases laterales por la izquierda es finito, entonces decimos que H tiene un índice finito en G. El número de clases laterales por la izquierda (que coincide con el número de clases laterales por la derecha) se llama índice de H en G. Entonces si H es normal en G y tiene un índice finito se concluye que G/H es finito y se verifica que $|G/H|$ es igual al índice de H en G. Finalmente si G es finito entonces se tiene que $|G/H|$ es igual a $|G|/|H|$.

Teorema 2.4.2. *Si $\phi : G \to G'$ es un homomorfismo entonces se verifica:*

- *$Ker(\phi)$ es normal en G*
- *El cociente $G/ker(\phi)$ es isomorfo a $Im(\phi)$.*
- *Recíprocamente, si H es un subgrupo normal de G, entonces la aplicación natural definida $a \to aH$ de G en G/H es sobreyectiva y además su núcleo es H.*

Por lo tanto si H es normal en G se verifica que la sucesión:

$$1 \to H \to G \to G/H \to 1$$

es una sucesión exacta corta

2.5. Clases de conjugación

Dos elementos a y b en un grupo G son conjugados, si existe un elemento $g \in G$ que verifique $b = gag^{-1}$. Esto establece una relación de equivalencia en G cuyas clases de equivalencias se llaman clases conjugadas. Si G es un grupo Abeliano entonces cada clase de cojugación tiene un solo elemento. Sin embargo si $a \neq b$ entonces tanto a como $ba^{-1}b$ son elementos distintos en la misma clase de conjugación. Entonces el número de clases de conjugación nos da una medida de cómo de lejos un grupo está de ser Abeliano. Si H y H' son dos subgrupos de G, entonces diremos que son conjugados si existe un elemento $g \in G$ que hace que se verifique la siguiente relación: $H' = hHg^{-1}$ [57].

Una acción en un grupo G es una aplicación $G \times S \to S$ definida $(g, s) \to g \cdot s$ tal que $(gh) \cdot s = g \cdot (h \cdot s)$ para todo $g, h \in G$ y todo $s \in S$. Por lo que se deduce que la identidad $e \in G$ actúa de manera trivial $(e \cdot s = s)$ y para cada $g \in G$ actúa como una permutación. La órbita de un elemento $s \in S$ es el conjunto:

$$G \cdot s = \{g \cdot s : g \in G\} \tag{2.2}$$

Si H es un subgrupo de G entonces G actúa sobre G/H como $g \cdot xH = (gx) \cdot H$.

Si G actúa sobre S y si $s \in S$ definimos su estabilizador como:

$$Stab_G(s) = \{g \in G : g \cdot s = s\} \tag{2.3}$$

Además es trivialmente un subgrupo de G. Si $s, s' \in S$ están una sola órbita entonces sus estabilizadores son conjugados. De hecho si $s' = gs$ entonces $Stab_G(s') = gStab_G(s')g^{-1}$. Entonces la acción de G en S es transitiva si y sólamente si hay un solo elemento en el órbita, esto es, para cada $s, s' \in S$ existe $g \in G$ tal que $s' = g \cdot s$. Supongamos que este fuese el caso, escogemos un punto *base*, $s_0 \in S$, y sea $H = Stab_G(s_0)$. Entonces esta elección determina una aplicación biyectiva $\phi : G \to H$ definida como $\phi(gH) = g \cdot s_0$. Esta aplicación ϕ es compatible con las acciones del grupo G en G/H y en S, o lo que es lo mismo, $g \cdot \phi(xH) = \phi(g \cdot xH)$ para todo $g \in G$ y para todo $xH \in G/H$. Si además $|G| < \infty$ se deduce que $|S| = |G|/|H|$ divide a $|G|$

Este Grupo G actúa sobre si mismo mediante la traslación $(g \cdot x = gx)$ y mediante la conjugación $g \cdot x = gxg^{-1}$. La primera acción es transitiva pero la segunda no, y sus órbitas son las clases conjugadas de G.

2.6. Grupos abelianos finitamente generados

Definición 2.6.1. Un grupo G se dice que está finitamente generado si existe un conjunto $V \subseteq G$ tal que todo elemento de G se puede expresar como producto finito de elementos de V.

Vamos a establecer sin demostración el teorema fundamental de los grupos abelianos finitos.

Teorema 2.6.1. *Sea G un grupo finitamente generado. Entonces G es isomorfo a un producto de grupos cíclicos.*

Corolario 2.6.2. *Sea G un grupo abeliano finito con nm elementos, donde n y m son dos enteros relativamente primos entre si. Entonces existe dos grupos H_1 y H_2 con n y m elementos respectivamente de tal manera que G es isomorfo a $H_1 \times H_2$.*

Definición 2.6.2. Sea G un grupo abeliano finito. Sea $g \in G$. Diremos que g es un elemento de torsión si $g \neq 0$ y además $g + g + \cdots + g = 0$, esto es tiene orden finito. Un grupo G se dice que está libre de torsión si no tiene elementos de torsión.

Corolario 2.6.3. *Sea G un grupo abeliano finito libre de torsión. Entonces G es isomorfo a un producto directo finito de copias de $\mathbb{Z}$*

2.7. Anillos

Muchas de las estructuras algebraicas más importantes llevan relacionadas dos operaciones. Por ejemplo, la suma y el producto de enteros, racionales, reales, etc.; las funciones Booleanas que representan el operador lógico "Y" (AND) y la suma módulo 2 (XOR); la suma y la multiplicación de matrices de orden $n \times n$ sobre $\mathbb{R}$, etc.

Definición 2.7.1. Un anillo R es un conjunto dotado de dos operaciones $+$ y $\times$ junto con dos elementos denotados por O y 1 que verifica que para todos $a, b, c \in R$:

- R es un grupo abeliano con la operación $+$ e identidad 0.
- $a \times (b \times c) = (a \times b) \times c$ y además $a \times 1 = a$.
- $a \times (b + c) = a \times b + a \times c$ y además se verifica por el otro lado, esto es, $(b+c) \times a = b \times a + c \times a$. Esta propiedad es conocida como ley distributiva.

Normalmente la operación $\times$ se denota por $\cdot$, o simplemente $a \cdot b$ quedará denotado por ab.

Se sigue de la definición de manera inmediata que $a \cdot 0 = 0 \ \forall a \in R$.

Se suele denotar al grupo aditivo R^+ al grupo abeliano que se obtiene al considerar sólo R con la operación de la suma.

Un anillo R se dirá que es conmutativo si $ab = ba$ para todo $a, b \in R$. A lo largo de este documento supondremos que R es una anillo conmutativo.

2.7.1. Unidades y divisores del 0

Sea R un anillo conmutativo. Un elemento $a \in R$ se dice que es una unidad si es inversible, esto es, $\exists b \in R$ tal que $ab = 1$. En este caso b es único. El conjunto de las unidades de un anillo R quedará denotado por R^x . Se sigue de manera inmediata que R^x es un grupo abeliano. Un elemento $a \in R$ se dice que es un divisor del 0 si se verifica que existe un elemento no nulo $b \in R$ tal que $ab = 0$. El anillo de los enteros $\mathbb{Z}$ no tiene divisores de 0 y las únicas unidades son 1 y -1. Sin embargo si R es un anillo finito, entonces cada elemento o bien es un una unidad o bien es un divisor de 0. En efecto sea $\phi_a : R \to R$ la aplicación que a cada $b \in R$ le asocia ab. Si ϕ_a es una aplicación biyectiva entonces a es inversible. Si la aplicación ϕ_a no es biyectiva entonces existe $b \neq c$ tal que $ab = ac$ o lo que es lo mismo $ab - ac = 0$ o de la misma manera $a(b - c) = 0$ con lo que a es un divisor de 0. Si $a, b \in R$ son tales que $ab = 0$ entonces se dirá que b es el anulador de a y se denotará por Z_a.

Definición 2.7.2. Sea $b \in R$ una unidad. Llamaremos orden del elemento b al menor entero positivo m tal que $b^m = 1$. Se dirá que dicho orden es ∞ si éste entero no existe.

Definición 2.7.3. Un anillo R se dirá que es dominio de integridad si no tiene divisores de 0. Un anillo conmutativo R se dirá que es cuerpo si cada elemento no nulo es inversible.

2.7.2. Ideales y cocientes

Definición 2.7.4. Sea S un subconjuto de un anillo R. Se dirá que S es un subanillo si S con las operaciones de R es un anillo.

Si I es un subgrupo aditivo de R entonces el cociente R/I es un conjunto de clases de equivalencia bajo la relación $a \sim b$ si y solamente si $a - b \in I$. La clase

de equivalencia que contiene al elemento a será $a+I$. Entonces R/I será un grupo Ableiano con la operación $(a+I)+(b+I) = a+b+I$. Sin embargo la multiplicación inducida por R no define necesariamente una operación bien definida en R/I. O lo que es lo mismo si $a' \sim a$ y $b' \sim b$ entonces es verifica que existen $c, d \in I$ tales que $b' = b+d$ y también $a' = a+c$. Entonces $a'b' = ab+ad+bc+cd$ no es equivalente a ab salvo que $ad+bc+cd \in I$. La siguiente definición nos da una condición necesaria y suficiente para que esto ocurra para todo $a, b \in R$ y para todo $c, d \in I$.

Definición 2.7.5. Un ideal es un subgrupo aditivo $I \subset R$ que verifica que para cada $a \in I$ y para cada $b \in R$ se tiene que $ab \in I$ [13]

Se sigue por tanto que el conjunto de clases de equivalencia de R/I heredan una estructura de anillo de R si y solamente si I es un ideal. Dos elementos $a, b \in R$ se dicen que son congruentes módulo I si están en la misma clase de equivalencia. O lo que es lo mismo $a-b \in I$. Cada clase de equivalencia se llama clase de equivalencia residuo, módulo I. Un ideal $I \subset R$ se dice que es un ideal propio si $I \neq R$. Un ideal I se dirá que es principal si existe un elemento $a \in R$ tal que $I = \{ar : r \in R\}$, en cuyo caso esto quedará denotado por $I = (a)$. Si I, J son dos ideales entonces la suma $I + J$ será otro ideal que verifica que es el menor ideal que contiene a I y a J. La intersección $I \cap J$ es también otro ideal. El ideal producto, IJ, definido como el conjunto de las sumas finitas de $\sum a_i b_i$ donde $a_i \in I$ y $b_i \in J$ es otro ideal. Un ideal $I \subset R$ se dirá que es un ideal maximal si I es un ideal propio de tal manera que si existiera otro ideal propio I' tal que $I \subset I'$ entonces $I = I'$. Sea I un ideal de un anillo R. Se dirá que el ideal I es primo si se verifica que $ab \in I$ implica necesariamente que $a \in I$ o $b \in I$. Un ideal I se dice que es primario si cuando se verifique $ab \in I$ entonces o bien $a \in I$ o bien $b^n \in I$ para algún $n \geq 1$.
Nótese que un cuerpo sólo contiene dos ideales (0) y el (1).

Teorema 2.7.1. *Sea R un anillo conmutativo. Entonces los siguientes enunciados son ciertos:*

1. *Un ideal $P \subset R$ es maximal sii R/P es un cuerpo.*
2. *Un ideal $P \subset R$ es primo sii R/P es dominio de integridad.*
3. *Cada ideal maximal es primo.*

Consideremos el anillo de los enteros $\mathbb{Z}$. Sea I un ideal que contenga a un elemento no nulo. Si multiplicamos por -1 un elemento de I seguirá estando en I, con lo que I contendrá a un elemento no nulo. Sea m el elemento no nulo y positivo de I. Sea $a \in I$ otro elemento. Entonces por la identidad de Bezout existirán $a, u \in \mathbb{Z}$ tales que $m.c.d.(m, a) = um + va$. Por lo tanto $m.c.d.(a, m) \in I$. Además $m.c.d.(a, m) \leq m$ por lo que dado que m es minimal entonces $m.c.d.(a, m) = m$. O lo que es lo mismo m divide a a. Por lo que todo múltiplo de m está en I, en consecuencia I tiene contiene solamente a los elementos que son múltiplos de m por lo que $I = (m)$ es principal.
El ideal (m) estará contenido en el ideal (n) sii m es múltiplo de n. El ideal (m) es primo sii m es primo. En este caso maximal. Y será primario sii m es una potencia de un primo.

Definición 2.7.6. Una función $\phi : R \to S$ donde R y S son dos anillos se dirá que es un homomorfismo de anillos si $\phi(a+b) = \phi(a)+\phi(b)$ y además $\phi(ab) = \phi(a)\phi(b)$ para todo $a, b \in R$. Si un homomorfismo ϕ de este tipo verifica:

1. ϕ es inyectivo. Entonces diremos que ϕ es un monomorfismo.
2. ϕ es sobreyectivo. Entonces diremos que ϕ es un epimorfismo.
3. ϕ es biyectivo. Entonces diremos que ϕ es un isomorfismo.
4. $R = S$. Entonces diremos que ϕ es un endomorfismo.
5. ϕ es endomorfismo biyectivo. Entonces diremos que ϕ es un automorfismo.

El conjunto de los automorfismos de un anillo S en si mismo forma un grupo bajo la composición denotado mediante $Aut(S)$. Más generalmente si R es un subanillo de S entonces el conjunto de los automorfismos de S de los que restringidos a R son la identidad forman un subgrupo denotado por $Aut_R S$.

2.7.3. Característica

Sea R un anillo conmutativo. Si M es un entero no negativo, escribiremos $m \in R$ para la suma $1+1+\cdots+1$ (m) veces. Esto define un homomorfismo de $\mathbb{Z}$ en R que puede ser mostrado mediante inducción. De hecho, existe un único homomorfismo de $\mathbb{Z}$ en R dado que cada uno de estos homomorfismos está completamente determinado por el hecho de que $1_{\mathbb{Z}}$ se aplica en 1_R y las operaciones entre los anillos se preservan. En núcleo de este homomorfismo es un ideal en $\mathbb{Z}$ de la forma (m) para algún entero m Este entero se llama característica de R. Para cada $a \in R$ tenemos que $ma = a+a+\cdots+a$ m veces. Entonces si la característica es no nula, m será el menor entero positivo que hace que $ma = 0$ para cada $a \in R$. Si la característica es cero, entonces no existe este m y por lo tanto Z es isomorfo a un subanillo de R. En cualquier otro caso, $\mathbb{Z}/(m)$ es isomorfo a un subanillo de R. Si R es finito entonces su característica es positiva dado que la sucesión de los elementos $1, 2, \cdots \in R$ debe llevarnos a una repetición.

Teorema 2.7.2. *Si R es un dominio de integridad entonces su característica es o bien 0 o bien un número primo.*

Lema 2.7.3. *Sea R un anillo conmutativo. Si la característica del anillo R es k y q es una potencia positiva de k entonces se verifica:*

$$(a+b)^q = a^p + b^q \in R \tag{2.4}$$

para cada $a, b \in R$

Si k no es primo, entonces el resultado 2.4 es en general falso.

2.7.4. El anillo de los enteros módulo n y las raíces primitivas

En esta sección vamos a fijar un entero no nulo N. El anillo $\mathbb{Z}/(N)$ es un grupo cíclico de orden (N). En la sección 2.7.3 la aplicación $(mod(N)) : \mathbb{Z} \to \mathbb{Z}/(N)$ resultó ser un homomorfismo de anillos. Si $x \in \mathbb{Z}$ podemos representar $\overline{x} \in \mathbb{Z}/N$ para la clase módulo N. Recíprocamente para cada $y \in \mathbb{Z}/(N)$ representaremos por el correspondiente $\hat{y} \in \mathbb{Z}$ donde $0 \leq \hat{y} \leq N - 1$, pero habrá que poner de manifiesto que la asociación $\mathbb{Z}/N \to \mathbb{Z}$ no es ni un grupo ni un homomorfismo de anillos. Por lo que es usual omitir tanto la *barra* como el *acento circunflejo*.
Si m divide a N entonces la aplicación $(mod(N)) : \mathbb{Z}/(N) \to \mathbb{Z}/(m)$ es un homomorfismo de anillos. Si a, b son no nulos, y relativamente primos, entonces la aplicación:

$$\mathbb{Z}/(ab) \to \mathbb{Z}/(a) \times \mathbb{Z}/(a) \tag{2.5}$$

dada por $x \to (x\ mod(a), x\ mod(b))$ es un isomorfismo de anillos. Pero si x es divisible entre a y entre b entonces si a, b son relativamente primos, se verifica que x es divisible entre ab. Este es un caso especial del teorema chino del resto que veremos más adelante.

Proposición 2.7.4. *Los siguientes enunciados son equivalentes:*

1. *El elemento* $\overline{x} = x\ mod(N) \in \mathbb{Z}/(N)$ *es inversible en* $\mathbb{Z}/(N)$
2. *El elemento* $\overline{N} = N\ mod(x) \in \mathbb{Z}/(x)$ *es inversible en* $\mathbb{Z}/(x)$
3. *Los enteros* x *y* N *son relativamente primos.*

4. *Existe* $n > 0$ *tal que* $x \mid (N^n - 1)$
5. *Existe* $m > 0$ *tal que* $N \mid (N^m - 1)$
6. *El elemento* $\overline{x} \in \mathbb{Z}/(N)$ *es un generador del grupo* $\mathbb{Z}/(N)$. *Esto es el elemento* $\{0, \overline{x}, \overline{x} + \overline{x} \cdots\}$ *recorre todos los elementos de* $\mathbb{Z}/(N)$

Como ya hemos visto en la sección 2.7.3 las unidades de $\mathbb{Z}/(N)$ forman un grupo Abeliano con la multiplicación. Este grupo queda denotado por $\mathbb{Z}/(N)^x$. La función de Euler $\phi(N) =\mid \mathbb{Z}/(N)^x \mid$ se define como el número de unidades de $\mathbb{Z}/(N)^x$. Para cada $y \in \mathbb{Z}/(N)^x$ existe un menor entero d tal que $y^d = 1 \in \mathbb{Z}/(N)$. Se sigue del teorema 2.1.2 que $d \mid \phi(N)$ con lo que si y es relativamente primo con N entonces se verifica que $y^{\phi(N)} \equiv 1 \ (mod(N))$. Este hecho es conocido como el teorema pequeño de Fermat. Los menores enteros de (4) y (5) serán $n = ord_x(\overline{N})$ y $m = ord_N(\overline{x})$ respectivamente.

Lema 2.7.5. *Sea* N *un entero positivo. Entonces los siguientes resultados se verifican:*

- *Si* $N = \prod_{i=1}^{k} {p_i}^{m_i}$ *es la factorización de* N *entonces* $\phi(N) = \prod_{i=1}^{k} \phi(p^{m_i})$
- $\phi(p^m) = p^{m-1}(p-1)$ *si* p *es primo.*
- $N = \sum_{d \mid N} \phi(d)$

De los comentarios que se han realizado en el párrafo anterior se deduce que $\mathbb{Z}/(N)^x$ es un grupo cíclico si y solamente si $N = p^m, 2p^m, 2$ ó 4, donde p es un primo $p \geq 3$. En este caso el generador a de $\mathbb{N}/(N)^x$ es una raíz primitiva módulo N. Por lo tanto el número de raíces primitivas módulo N será $\phi(\phi(N))$. La conjetura de Artin [40; 55] establece que en parte cada primo primo $p \in \mathbb{Z}$ es una raíz módulo q un número finito de q veces. El siguiente resultado ayuda a la hora de encontrar raíces primitivas.

Lema 2.7.6. *Sea p un primo y sean $s \geq 1$, $t \geq 1$, $t \in \mathbb{Z}$. Entonces b es una unidad módulo p^s si y solamente si es una unidad módulo p^t*

Lema 2.7.7. *Supongamos que $N = p^t$ para algún primo $p \geq 3$ y $t \geq 2$. Sea $a \in \mathbb{Z}$ tal que $2 \leq a \leq N-1$. Entonces a es primitivo módulo N si y solamente si a es primitivo módulo p^2. Además este resultado se cumple si y solamente si a es primitivo módulo p y además p^2 no divide a $a^{p-1}-1$*

Demostración. Si a es primitivo módulo p^t entonces por el lema 2.7.6 cada unidad b módulo p ó p^2 es una unidad módulo p^t. Entonces b es congruente con una potencia de a módulo t y además módulo p y p^2. Entonces a es primitivo módulo p y p^2. Vamos a demostrar el recíproco mediante inducción. Sea $t_0 \geq 3$ un valor fijo. y supongamos que a es primitivo en $\mathbb{Z}/(p^s)$ para cada $s < t_0$. El orden de a es un divisor de $\phi(pt) = p^{t-1}(p-1)$, el cardinal del grupo de las unidades en $\mathbb{Z}/(p^t)$. Queremos demostrar que el orden de a no es un divisor de $p^{t-1}(p-1)/r$ para cualquier divisor r de $p^{t-1}(p-1)$.

Supongamos primero que $r = p$. Como $a^{p^{t-3}(p-1)} = a^{\phi(p^{t-2})} \equiv 1\ mod(p^{t-2})$. Tenemos que $a^{p^{t-3}(p-1)} = 1+cp^{t-2}$ para algún $c \neq 0$ y que verifique que es relativamente primo con p dado que a es primitivo en $\mathbb{Z}/(p^{t-1})$. Entonces $a^{p^{t-2}(p-1)} = (1+cp^{t-2})^p \equiv 1+cp^{t-1} \not\equiv 1\ mod(p^t)$, dado que $\binom{p}{i}$ es un múltiplo de p. Esto muestra que el orden de a módulo p^t no es $\phi(p^t)/r$ con $r = p$.

Supongamos por otro lado que r es un divisor primo de $p-1$ y que $a^{p^{t-1}(p-1)/r} \equiv 1\ mod(p^t)$. Sea b un elemento primitivo módulo p^t y sea $a \equiv b^k\ mod(p^t$. Entonces $b^{kp^{t-1}(p-1)/r} \equiv 1\ mod(p^t)$ por lo que $p^{t-1}(p-1)$ divide a $kp^{t-1}(p-1)/r$. O lo que es lo mismo, r divide a k. Pero entonces $a^{p^{t-2}(p-1)/r} \equiv b^{kp^{t-2}(p-1)/r} \equiv 1\ mod(p^{t-1})$. Lo que es una contradicción. Esto demuestra nuestra tesis. □

Acabamos de demostrar que si a es primitivo módulo p^t entonces también lo

será módulo p y p^2 dividirá a $a^{p-1}-1$. Ahora vamos a ver el recíproco. Si p^2 no divide a $a^{p-1}-1$ entonces la única manera en que la elección de a pueda fallar será cuando a tenga orden módulo p^2 y que éste divida a $p(p-1)/r$ para algún primo divisor de $p-1$. Pero como hemos visto en el párrafo anterior esto implica que a tiene un orden módulo p que es divisor de $(p-1)/r$ lo cual contradice que a sea primitivo módulo p.

2.7.5. Divisibilidad en Anillos

Sea R un anillo conmutativo. Si $a, b \in R$ entonces diremos que a es divisor de b si existe $c \in R$ tal que $ac = b$, en cuyo caso lo denotaremos median $a \mid b$. El elemento a es una unidad en R si es inversible, o lo que es lo mismo, si es divisor de 1. Los elementos $a, b \in R$ estarán asociados si existe una unidad ϵ tal que $a = b\epsilon$. Un elemento no nulo $c \in R$ es un divisor común de a y b si $c \mid a$ y $c \mid b$. El máximo común divisor de a y b denotado por $m.c.d.(a, b)$ es el mayor de los divisores comunes a a y a b.

Un elemento $c \neq 0$ se dirá que es múltiplo de un elemento $a \in R$ si $a \mid c$. Un elemento c se dirá que es múltiplo de a y de b dos elementos del anillo R si $a \mid c$ y además $b \mid c$. Llamaremos mínimo común múltiplo de a y de b al menor de los múltiplos comunes a a y a b. y lo denotaremos $M.C.M.(a, b)$

Definición 2.7.7. Un elemento $r \in R$ si (r) es un ideal propio y primo, o lo que es lo mismo si $ab \in (r)$ entonces $a \in (r)$ ó $b \in (r)$

Si (r) es primario entonces $ab \in (r)$ implica que $a \in (r)$ ó $b^n \in (r)$ para algún $n > 0$.

Si (r) es irreducible entonces dado un elemento no unidad c de tal manera que se verifique que $c = ab$ implica que o bien a o bien b será unidad.

Dos elementos no nulos que no sean unidades, r, s se dirán que son relativamente primos si $(r)+(s) = R$. o equivalentemente si existen $a, b \in R$ tales que $1 = ar+bs$.

Definición 2.7.8. Sea R un anillo conmutativo.

1. R es un dominio de integridad si R no tiene divisores de 0.
2. R es principal si cada ideal en R es principal. Un anillo R se dirá que es un dominio de ideales principales (PID) si es principal y es dominio de integridad.
3. R es un dominio de máximo común divisor (GCD) si cada par de elementos en R tienen máximo común divisor.
4. R es un anillo local si contiene a un único ideal maximal.
5. R es un dominio de factorización única (UFD) si es un dominio de integridad y cada elemento $a \in R$ se puede expresar de manera única (salvo orden de los factores y productos por unidades) $a = \prod_{i=1}^{m} p_i$ donde cada p_i es irreducible (no necesariamente distintos).
6. R es un anillo de factorización si cada elemento no unidad $a \in R$ tiene una factorización en producto de elementos irreducibles, (no necesariamente distintos y no necesariamente única).
7. R se dice que es Noetheriano si cada cadena de ideales $J_1 \subset J_2 \subset \cdots$ estabiliza en el algún punto finito, o lo que es lo mismo si cada ideal está finitamente generado.
8. R se dirá que es euclídeo si se verifican:
   a) Existe una función $\delta : R \to \{0, 1, 2, 3, \cdots\} \cup \{-\infty\}$ tal que para cada $a, b \in R$ no nulos a la vez, se tiene que $\delta(ab) \geq \delta(a)$.
   b) Para cada $a, b \in R$ con $b \neq 0$ existe $q \in R$ (llamado cociente) y $r \in R$ (llamado resto) tales que:

$$a = qr + r \text{ verificando que } \delta(r) < \delta(b) \tag{2.6}$$

Aunque no lo vamos a definir todavía vamos a hacer uso de un tipo especial de anillo conocido como anillo de polinomios. Sea R un anillo conmutativo y unitario. Denotaremos por $R[x]$ al anillo de todos los polinomios con coeficientes en R.

Para los siguientes resultados en esta sección se ha tenido en cuenta [8].

Teorema 2.7.8. *Sea R un anillo conmutativo y sea $R[x]$ el anillo de polinomios con coeficientes en R. Entonces se puede establecer el siguiente diagrama de implicaciones entre las distintas propiedades de R.*
Cuerpo $\rightarrow$ Euclídeo $\rightarrow$ PID $\rightarrow$ UFD $\rightarrow$ GCD $\rightarrow$ Entero $\rightarrow$ $R[x]$ es entero $\rightarrow$ $R[x]$ es Euclídeo.
Además si R es finito, entonces es un cuerpo.

Teorema 2.7.9. *Sea R Un anillo conmutativo. Sean $a, b \in R$. Entonces son ciertas:*

1. *El elemento a es primo si y solamente si se verifica la siguiente propiedad: si $a \mid cd$ entonces $a \mid c$ o bien $a \mid d$*
2. *Si a es primo y no es divisor de 0 entonces a es irreducible.*
3. *Si R es UFD, entonces a es primo si y solamente si a es irreducible.*
4. *Los elementos a y b son relativamente primos si y solamente si a es inversible en $R/(b)$*
5. *Si a y b son primos relativos entonces cada divisor de a y b es una unidad.*
6. *Si R es PID y además cada divisor de a y de b es una unidad entonces a y b son primos relativos.*
7. *Si R es PID y $a \in R$ entonces a es primo si y solamente si (a) es maximal. O lo que es lo mismo si y solamente si $R/(a)$ es un cuerpo.*

2.7.6. Fracciones

El cuerpo $\mathbb{Q}$ se construye utilizando como base el anillo $\mathbb{Q}$, donde identificamos las fracciones a/b con $(ax)/(bx)$ para cada entero no nulo x. Una construcción semejante se puede realizar en cualquier anillo R. Un subconjunto $S \subset R$ se dice que es multiplicativo si es cerrado bajo la multiplicación. Si S es un subconjunto de R multiplicativo, se define el anillo $S^{-1}R$ como la colección de todos los símbolos de la forma a/b donde $a \in R$ y $b \in S$ con la siguiente relación de equivalencia: $a/b \sim a'/b'$ si existe $s \in S$ tal que:

$$s(ab' - ba') = 0 \tag{2.7}$$

Las operaciones suma y producto se definen de la manera habitual:

$$\frac{a}{b} + \frac{a'}{b'} = \frac{ab' + a'b}{bb'} \tag{2.8}$$

$$\frac{a}{b}\frac{a'}{b'} = \frac{aa'}{bb'} \tag{2.9}$$

El anillo $S^{-1}R$ tendrá un solo elemento caso de que $0 \in S$, por lo que a veces se excluye.

Ahora supongamos que S no contiene divisores del 0, entonces la ecuación 2.7 se puede reemplazar por una relación más habitual: $ab' = a'b$. La aplicación natural $R :\to S^{-1}R$ que a cada a se le asocia $a/1$ es una aplicación inyectiva con lo que $S^{-1}R$ contendrá a R. Cada elemento de S se ha convertido en inversible en $S^{-1}R$. Si el conjunto S contiene a todos los elementos que no son divisores del O entonces un elemento de $S^{-1}R$ es o bien un divisor de 0 o bien es inversible. Es este caso la aplicación $S^{-1}R$ se conoce como anillo de fracciones de R. Si R es un dominio de integridad entonces ese anillo al no tener divisores de 0 es un cuerpo, llamado

cuerpo de las fracciones de R [59].

2.7.7. Teorema Chino del Resto

Si R_1 y R_2 son dos anillos podemos considerar el producto cartesiano de $R_1 \times R_2$. El teorema Chino del resto nos da una condición suficiente para que este anillo se pueda descomponer como un producto [62] y por otro lado [66].

Teorema 2.7.10. *Sea R un anillo y sean $I_1, \cdots, I_k$ tales que $I_i + I_j = R$ para cada $i \neq j$. Sea $I = \cap_{j=1}^{k} I_j$. Entonces para cada $a_1, a_2, \cdots a_k$ hay un elemento $a \in R$ tal que para cada i, $a \equiv a_i \ mod(I_i)$. Además se verifica que a es único módulo I y $R/I = \prod_{j=0}^{k} R/I_j$*

Demostración. Para $k = 1$ es obvio. Para $k = 2$ entonces existen elementos $b_1 \in I_1$ y $b_2 \in I_2$ tales que $1 = b_1 + b_2$. Basta tomar $a = a_1 b_2 + a_2 b_1$

Supongamos que $k > 2$. Para cada i sea $J_i = \prod_{j \neq i} I_j$. Para cada $i \geq 2$ existen elementos $c_i \in I_1$ y $b_i \in I_2$ tales que $1 = c_i + b_i$. En particular $\prod_{i=2}^{k} (c_i + b_i) = 1$. Este producto está en $I_1 + J_1$, luego $R = I_1 + J_1$, de manera análoga se ve que $R = I_j + J_j$ para cada j. Por lo que acabamos de demostrar caso de tener dos ideales, existe en un elemento $d_j \in R$ tal que $d_j \equiv 1 \ mod(I_j)$ y además $d_j \equiv 0 \ mod(J_j)$. Luego entonces $a = a_1 d_1 + \cdots a_k d_k$ como era requerido en nuestra tesis.

Para cada i consideramos el homomorfismo proyección $\phi_i : R/I$ en R/I_i. Esto induce un homomorfismo ϕ de R/I en $\prod_{j=1}^{k}$ cuyo núcleo es $I = \cap_{j=1}^{k} I_j$. Por lo tanto ϕ es inyectiva y en consecuencia biyectiva ya que habíamos visto que era sobreyectiva. Por lo tanto es un isomorfismo y en consecuencia se ve que a es único. □

Corolario 2.7.11. *Supongamos que R es un PID y sean $b_1, \cdots, b_k \in R$ elementos primos dos a dos. Si $a_1, \cdots, a_k \in R$ entonces existen un elemento $a \in R$ tal que*

para cada i, se tiene $a \equiv a_i \ (mod b_i)$

2.8. Cuerpos

Un cuerpo es un anillo donde cada elemento no nulo posee inverso. Muchos generadores de sucesiones se pueden ver como multiplicaciones por un número fijo en un anillo. Como los cuerpos finitos tienen los grupos cíclicos de las unidades, ellos son una fuente de generadores de sucesiones con un extenso periodo.

2.8.1. Grupo de Galois

Si F es un cuerpo y E es un anillo, el núcleo de cada homomorfismo no nulo $F :\to E$ es un ideal, el único ideal propio, luego cada homomorfismo es inyectivo. Diremos entonces que E es una extensión de F. Los elementos de E pueden ser sumados y multiplicados por algunos de F, con lo que E es un espacio vectorial sobre F. La dimensión de E como espacio vectorial sobre F se llamará grado o dimensión de la extensión [29].

Un anillo R es algebraicamente cerrado si cada polinomio $p(x) \in F[x]$ se puede descomponer completamente en factores, $p(x) = k(x - a_1)(x - a_2) \cdots (x - a_n)$ donde $deg(p(x)) = n$ y donde $k, a_i \in F$. Cada cuerpo F está contenido en un cuerpo algebraicamente cerrado $\overline{F}$ de grado finito sobre F llamada clausura algebraica de F [49].

Si G es un subgrupo del grupo de automorfismos de un cuerpo E, entonces el conjunto de los elementos de E que son fijos al aplicar un automorfismo en G ($\sigma(a) = a$ para cada $a \in E$ y cada $\sigma \in G$) se denota mediante E^G. Es necesariamente un cuerpo dado que es cerrado bajo la suma, el producto y el inverso. Si $F \subset E$ son cuerpos entonces el grupo $Aut_F(E)$ de automorfismos de E que fijan cada elemen-

to de F es el grupo de Galois de E sobre F y se denotará mediante $GAL(E/F)$, entonces $F \subseteq E^G$. Si $F = E^G$ entonces diremos que E es una extensión de Galois de F.

Si $F \subset E$ es una extensión finita de un cuerpo y $\overline{F}$ es una clausura algebraica de F entonces existen unas inmersiones $h_1, h_2 \cdots, h_n : E \to \overline{F}$. Si E es una extensión de Galois de F entonces todas las inmersiones tienen la misma imagen. En este caso, la elección de la inmersión determina uno a uno la correspondencia. $h_i \leftrightarrow \sigma_i$ con los elementos del Grupo de Galois $GAL(E/F)$ dado por $h_i(x) = h_1(\sigma_i(x))$

Proposición 2.8.1. *Sea $F \subset (F)$ una extensión finita y sea $T : E \to F$ una aplicación F- lineal. Entonces para cada aplicación F- lineal $f : E \to F$ existe un único $a \in E$ tal que $f(x) = T(ax)$ para todo $x \in E$*

2.8.2. Anillos locales y finitos

Sea R un anillo con un número finito de elementos. Diremos que el anillo es local si existe un único ideal maximal κ. En este caso, el ideal maximal κ estará formado por elementos que no son unidades en R [37]. El cociente $F = R/\kappa$ será un cuerpo llamado cuerpo de los residuos de R. Para cada $i \geq 0$ el cociente κ^{-1}/κ^i es un espacio vectorial sobre F, porque R actúa mediante la multiplicación y κ actúa trivialmente. Mostraremos algunos ejemplos:

- Cualquier cuerpo de Galois finito.
- $\mathbb{Z}/(p^n)$ para cada primo p, con ideal maximal (p) y cuerpo residuo $\mathbb{Z}/(p)$
- $\mathbb{F}[x]/(f^n)$ donde $\mathbb{F}$ es un cuerpo finito y f es un polinomio irreducible, con ideal maximal (f) y cuerpo de los residuos $\mathbb{F}[x]/(f)$

2.9. Cuerpos Finitos

En esta sección analizaremos la estructura de los cuerpos finitos o cuerpos de Galois [53].

2.9.1. Propiedades básicas

Teorema 2.9.1. *Sea p un número primo. Para cada $d > 0$ existe un único cuerpo (salvo isomorfismos) $\mathbb{F}_{p^d}$. Esto ocurre para todos los cuerpos finitos. Si $e > 0$ es otro entero entonces existe una inclusión $\mathbb{F}_{p^d} \subseteq \mathbb{F}_{p^e}$ si y solamente si $d \mid e$. El subcuerpo $\mathbb{F}_{p^d}$ consiste en aquellos elementos $a \in \mathbb{F}_{p^d}$ que satisfacen $a^{p^d} = a$*

El cuerpo $\mathbb{F}_{p^d}$ se denotará a veces mediante GFp^d

2.9.2. Grupos de Galois de cuerpos finitos

Sea $E = \mathbb{F}_{p^d}$ donde p es primo. Entonces la aplicación $\sigma : E \to E$ tal que $\sigma(x) = x^p$ es un automorfismo de cuerpos. Sin embargo $\sigma^e(x) = (x^p)^e = x$ con lo que $\sigma^e = Id$ es la identidad. Entonces varias potencias de σ forman un grupo cíclico de automorfismos, de orden e, que fija cada elemento de $\mathbb{F}_p$

Proposición 2.9.2. *El grupo $\{\sigma^0 = Id, \sigma_1, \cdots, \sigma^{e-1}\}$ es el grupo de Galois $Gal(E/\mathbb{F}_p)$*

Teorema 2.9.3. *Sea $F = \mathbb{F}_{p^d} \subset E = \mathbb{F}_{p^e}$, entonces el grupo de Galois $Gal(E/F)$ es un subgrupo cíclico del grupo $Gal(E/\mathbb{F}_p)$ de orden e/d. Está generado por el automorfismo $\sigma^d : x \to x^{p^d}$. El cuerpo $F \subset E$ está formado por los elementos de E que quedan fijos mediante cada elemento de $Gal(E/F)$. Por lo tanto E es una extensión de Galois del cuerpo F.*

Teorema 2.9.4. *Sea F un cuerpo finito de q elementos y sea $f(x) \in F[x]$ un polinomio de grado d con coeficientes en F. Sea E una extensión de cuerpos de*

F y supongamos que $\alpha \in E$ es una raíz del polinomio f. Entonces para cada $\sigma \in Gal(E/F)$, el elemento $\sigma(\alpha) \in E$ es una raíz de f. Si f es irreducible en $F[x]$ y E tiene grado de extensión d sobre F entonces todas las raíces de f están contenidas en E. Estas son exactamente las conjugadas de Galois,

$$\sigma_i(\alpha) = \alpha^{q^i}$$

donde $0 \leq i \leq d-1$, esto es donde σ_i recorre todos los elementos de $Gal(E/F)$

2.10. Anillos de Polinomios

En esta sección vamos a describir las propiedades básicas de los polinomios. El anillo de polinomios está dentro de las estructuras algebraicas más importantes. Se necesita para el análisis de los LFSR, [13].

2.10.1. Polinomios sobre un anillo

A lo largo de esta sección vamos a suponer que R es un anillo conmutativo. Un polinomio sobre el anillo R es una expresión algebraica de la forma:

$$f = f(x) = a_0 + a_1 x + \cdots a_d x^d = \sum_{i=0}^{d} a_i x^i \tag{2.10}$$

donde $a_0, a_1 x, \cdots, a_d \in R$ y x se conoce como indeterminada. Los elementos a_i se conocen como coeficientes de f en R. Cuando escribimos los polinomios se podrán omitir aquellos elementos que son nulos. Además se podrán escribir los términos en un orden distinto. Si $a_d \neq 0$ se dirá que el polinomio tiene grado d y lo denotaremos $deg(f(x)) = d$. En este caso se dirá que a_d es el término líder o principal del polinomio. Diremos que $deg(0) = -\infty$. Si $deg(f(x)) = 0$ entonces diremos que es un polinomio constante. Si $a_d = 1$ se dirá que el polinomio es mónico. El término

a_0 se conoce como término independiente. Se conoce como el valor numérico de $f(x)$ en b como el valor que se obtiene $a_0 + a_1 b + \cdots a_d b^d$ y se denotará por $f(b)$. Un elemento $b \in R$ se dirá que es una raíz del polinomio si $f(b) = 0$. Si $g(x) = \sum_{i=0}^{e} b_i x^i$ es otro polinomio se define:

$$(f+g) = f(x) + g(x) = \sum_{i=0}^{max(d,e)} (a_i + b_i) x^i \tag{2.11}$$

$$(fg) = f(x)g(x) = \sum_{i=0}^{d+e} \left[\sum_{j=max(0,i-e)}^{min(d,i)} (a_j + b_{i-j}) \right] x^i \tag{2.12}$$

El conjunto de todos los polinomios se denotará mediante $R[x]$. Las operaciones de suma y producto definidas anteriormente dotan a $R[x]$ de estructura de anillo donde el elemento nulo es $a_i = 0 \ \forall i$ y cuyo elemento unidad es el polinomio cuyo $a_0 = 1$ y $a_i = 0 \ \forall i \geq 1$. La demostración del siguiente lema es obvia.

Lema 2.10.1. *Si $f(x), g(x) \in R[x]$ entonces el $deg(f+g) \leq max(deg(f), deg(g)$ dándose la igualdad caso de que $deg(f) \neq deg(g)$. Además $deg(fg) \leq deg(f) + deg(g)$ siendo la igualdad cierta caso de que el producto de los términos líderes de ambos polinomios sea no nulo. En particular si R es dominio de integridad entonces $R[x]$ también lo es.*

Si R es un dominio de integridad, entonces las unidades de $R[x]$ son exactamente los polinomios de grado 0, constantes, cuyos términos independientes son las unidades de R. Esto es falso en general. Por ejemplo, si $R = \mathbb{Z}/(4)$, entonces $(1+2x)^2 = 1$, así que $1 + 2x$ es una unidad de grado 1. El siguiente resultado nos indica que algunas veces podremos realizar la división de polinomios y calcular el resto en $R[x]$ [56].

Teorema 2.10.2. *Sean $f(x), g(x) \in R[x]$ dos polinomios. Supongamos que el*

término líder de g es una unidad en R. Entonces existe dos únicos polinomios $q, r \in R[x]$ tales que $deg(r) < deg(g)$ y además:

$$f(x) = q(x)g(x) + r(x)$$

Corolario 2.10.3. *Si R es un cuerpo entonces $R[x]$ es un dominio euclidiano con $\delta(f) = deg(f)$*

Teorema 2.10.4. *Si a es una raíz de $f(x) \in R[x]$ entonces existe un polinomio $q(x) \in R[x]$ tal que*

$$f(x) = (x - a)q(x)$$

Si R es un dominio de integridad, entonces el número de raíces de f no puede exceder al grado de f.

Demostración. Por el teorema de la división euclídea de polinomios 2.10.2 aplicado al caso en que el divisor sea $(x - a)$ se tiene que existe $q(x) \in R[x]$ tal que $f(x) = q(x)(x - a) + r(x)$. El grado de r ha de ser 0. Pero al ser a una raíz entonces este valor ha de ser 0. Luego $f(x) = (x - a)q(x)$ □

Teorema 2.10.5. *Supongamos que R es un anillo GCD y dominio de factorización única. Entonces $R[x]$ es un dominio de factorización.*

Lema 2.10.6. *Sea $q = \sum_{i=0}^{m} q_i x^i \in R[x]$ con coeficientes en R. Consideremos las siguientes tesis:*

1. *q_0 es inversible en R*
2. *El polinomio x es inversible en el cociente de polinomios $R[x]/(q)$.*
3. *Los polinomios $q(x)$ y x son relativamente primos en el anillo $R[x]$.*
4. *Existe un entero $T > 0$ tal que $q(x)$ es un facotr de $x^T - 1$*
5. *Existe un entero $T > 0$ tal que $x^T = 1$ en el anillo $R[x]/(q)$.*

Entonces las tesis (1), (2) y (3) son equivalente y si se verifica cualquiera de ellas entonces

$$x^{-1} = -q_0^{-1}(q_1 + q_2x + \cdots + q_mx^{m-1}) \text{ en } R[x]/(q)$$

Las tesis expuestas en (4) y (5) son equivalentes el T es el mismo entero positivo y además $x^{-1} = x^{T-1} \in R[x]/(q)$. Las tesis (4) y (5) implican (1), (2) y (3). Si R es finito entonces (1) ó (2) implican (3), (4) ó (5)

Cuando se den las condiciones de (4) ó (5) del lema anterior entonces T es el menor valor que verifica que $q(x) \mid (x^T - 1)$. Este valor se conoce como el orden del polinomio q. Esta terminología puede llevar a error ya que en la teoría de grupos el orden del polinomio $q(x)$ es el orden de x en el grupo $(R[x]/(q))^x$. La condición (4) no se verifica con lo que q entonces no tendría orden ó si lo tiene éste sería infinito. Por ejemplo si $R = \mathbb{Q}$ entonces el polinomio $q(x) = x - 2$ tendría orden infinito.

Teorema 2.10.7. *Sean a y b dos enteros positivos. Entonces sobre cualquier anillo R el polinomio $x^a - 1$ divide al polinomio $x^b - 1$ si y solamente si $a \mid b$*

Demostración. Por el teorema de la división euclídea para enteros escribimos $b = qa + r$ donde $0 \leq r < a$. Entonces

$$x^b - 1 = (x^{b-a} + x^{b-2a} + \cdots + x^r)(x^a - 1) + x^r - 1$$

Como $deg(x^r - 1) < deg(x^a - 1)$ entonces se deduce que $(x^a - 1)$ divide a $x^b - 1$ si y solamente si $x^r - 1 = 0$. Esto último se verifica si y solamente si $r = 0$. □

Este resultado será muy útil para poder buscar raíces de la unidad.

2.10.2. Polinomios sobre un cuerpo

En esta sección vamos a trabajar con F un cuerpo en lugar de R un anillo.

Teorema 2.10.8 (cf)**.** *[70] Si F es un cuerpo, entonces $F[x]$ es euclídeo con $\delta(f) = deg(f)$. Además cada ideal en $F[x]$ tiene un único generador mónico. Por otro lado cada $f(x) \in F[x]$ se puede expresar en la forma:*

$$f(x) = ap_1{}^{e_1}p_2{}^{e_2}\cdots p_k{}^{e_k}$$

donde $a \in F$, los términos p_i son todos elementos de $F[x]$ que son distintos, mónicos e irreducibles y los e_i son enteros positivos. Esta representación es única salvo reordenación de sus factores.

Demostración. Se deduce del teorema 2.10.2 que $F[x]$ es euclídeo. Es además UFD por el teorema 2.7.8. Cada polinomio irreducible tiene un único polinomio mónico asocaido (simplemente dividimos cada coeficiente por el término líder). De esta manera a será único. □

Se sigue del mismo modo del teorema 2.7.8 que $F[x]$ es un anillo GCD. Pero para ser más preciso aplicamos el siguiente teorema.

Teorema 2.10.9. *Sea F un cuerpo y sean $f_1, f_2, \cdots, f_k \in F[x]$ no todos nulos. Existe un único polinomio $g \in F[x]$ mónico tal que:*

1. *g divide a cada f_i*
2. *si h divide a cada f_i, entonces h divide a g*

Lo que es más g se puede expresar en la forma:

$$g = h_1f_1 + h_2f_2 + \cdots h_kf_k \tag{2.13}$$

para algunos $h_1, h_2, \cdots, h_k \in F[x]$

Demostración. Sea $I = \{h_1f_1 + h_2f_2 + \cdots h_kf_k : h_1, h_2, \cdots, h_k\} \in F[x]$. Entonces I es un ideal en $F[x]$ por lo que por el teorema anterior I tiene un único generador

mónico g. Entonces como $g \in I$, g se puede expresar en la forma de 2.13. Entonces se sigue que cada h que divida a cada f_i también divide a g. Dado que $f_i \in I$, entonces g divide a f_i □

Escribiremos $g = m.c.d.(f_1, f_2, \cdots, f_k)$. Se puede encontrar este polinomio utilizando el teorema 2.10.2 de manera repetida hasta que encontremos un resto nulo. El siguiente teorema nos permitirá cuerpos finitos de todos los tamaños posibles.

Teorema 2.10.10. *Si F es un cuerpo finito y d es un entero positivo, entonces existe al menos un polinomio irreducible de grado d en $F[x]$*

Si $F \subseteq E$ son dos cuerpos y si $a \in E$ es un elemento que es raíz de algún polinomio con coeficientes en F, entonces decimos que a es algebraico sobre F. Un polinomio f $F[x]$ se dice que es un polinomio mínimo de a sobre F si es mónico, $f(a) = 0$ y además es el polinomio de menor grado que verifica estas condiciones.

Teorema 2.10.11. *Sea F un cuerpo. Supongamos que $a \in F$ es algebraico sobre F. Entonces a tiene un único polinomio mínimo $f \in F[x]$. El polinomio mínimo f es único polinomio irreducible que tiene a a como raíz. Si g es otro polinomio tal que $g(a) = 0$ entonces f divide a g en $F[x]$.*

Demostración. Si dos polinomios mónicos $f, g \in F[x]$ tienen el mismo grado mínimo y tienen ambos a a como raíz entonces $f - g$ tiene un grado menor que el de f y g y además a es una raíz de $f - G$ lo cual es un absurdo. Si f es un polinomio mínimo de a entonces $f = gh$, entonces $0 = f(a) = g(a)h(a)$ con lo que o bien $g(a) = 0$ o bien $h(a) = 0$. Dado que f es minimal entonces cualquier factor que tenga a a como raíz debe tener el mismo grado que f, con lo que f es irreducible. Supongamos que f es un polinomio mónico e irreducible tal que $f(a) = 0$. El conjunto:

$$J = \{h \in F[x] : h(a) = 0\}$$

es un ideal que contiene a f, pero f es irreducible, entonces $J = (f)$ es el ideal generado por f y f es el único polinomio mónico con esta propiedad. Si $g(a) = 0$ entonces $g \in J$ por lo que entonces g es un múltiplo de f. En particular, f es el polinomio mínimo de a. □

Más generalmente podemos pensar que en el operador de un anillo R en el anillo de polinomios $R[x]$. Estrictamente hablando esto no es una función dado que no hay un conjunto de todos los anillos.

Capítulo 3

Criptología

Aunque no de forma exclusiva ni excluyente, los LFSR mantienen una relación muy estrecha con los sistemas criptográficos de protección de datos, en especial, con los que se encuadran en la categoría de cifrado en flujo, en los que el diseño de un PRNG constituye una parte esencial de su funcionamiento. Es en esta disciplina, en los principios teóricos desarrollados y en los resultados prácticos obtenidos, en los que se sustenta el desarrollo de esta tesis, que parte del conocimiento que existe sobre la generación de secuencias pseudoaleatorias mediante LFSR. Por este motivo, hemos considerado adecuado incluir una breve introducción a los fundamentos criptográficos del cifrado en flujo, así como su relación con los LFSR.

3.1. Introducción

La criptología, asociada desde los inicios de la Escritura con el arte y la técnica de ocultar la información escrita a los destinatarios no autorizados, se considera actualmente como una disciplina matemática en la que se fundamentan los sistemas, métodos y algoritmos empleados para ocultar el contenido de un mensaje con independencia del medio de transmisión o almacenamiento que se utilice [43; 79]. La criptología se ha dividido clásicamente en dos ramas, criptografía y criptoanálisis, a la que con frecuencia —*fundamentalmente en los primeros tiempos*— se le ha unido la esteganografía, cuya principal diferencia reside en su objetivo de ocultar la mera existencia del mensaje y no solo su contenido [27; 31].

La criptografía (del griego *kryptos*: oculto, y *graphia*: escritura), aparece definida como el "Arte de escribir con clave secreta o de un modo enigmático" en el diccionario de la Real Academia Española. Es, por tanto, la disciplina encargada de diseñar métodos para ocultar el contenido de un mensaje, no el mensaje en sí mismo. Para ello realiza en el mismo una serie de transformaciones de forma que sin conocer el origen y desarrollo de éstas, así como una serie de informaciones complementarias, en general la clave, sea imposible conocer el contenido del mensaje original. A través de la criptografía, la información puede ser protegida contra el acceso no autorizado, su interceptación, su modificación y la inserción de información adicional. También puede ser usada para prevenir el acceso y uso no autorizado de los recursos de una red o sistema informático y para prevenir a los usuarios la denegación de los servicios a los que sí están permitidos. Actualmente, la criptografía proporciona seguridad a las redes telemáticas, incluyendo la identificación de entidades y autenticación, el control de acceso a los recursos, la confidencialidad de los mensajes transmitidos, la integridad de los mensajes y su

no repudio.

Las técnicas criptográficas de protección de datos requieren la aplicación de fundamentos y resultados de diversas áreas como la Teoría de la Información, la Complejidad Algorítmica, la Teoría de números o la Matemática Discreta, en general.

Por otra parte, el criptoanálisis, estrechamente relacionado con todas las áreas en las que se desenvuelve la criptografía, centra su objetivo en el análisis de los sistemas criptográficos desarrollados y en el diseño de métodos dedicados a obtener el contenido del mensaje original sin el conocimiento de parte de la información requerida para ello, principalmente la clave. De este modo, criptografía y criptoanálisis avanzan siempre en paralelo, pues se deben mutuamente su existencia y su sentido.

Definición 3.1.1. Definimos un alfabeto $A = \{a_1, a_2, \cdots, a_n\}$ como el conjunto de símbolos a_i que se utilizan para representar un mensaje.

En una operación de cifrado de la información pueden existir uno o varios alfabetos, aunque generalmente se utilizará uno para el mensaje en claro y otro para el mensaje cifrado, pueden llegar a utilizarse varios.

Definición 3.1.2. Definimos un $n-$grama como una de las combinaciones de n elementos de un alfabeto de m símbolos, siempre con $m \geq n$. En los casos $n = 2$ y $n = 3$ hablaremos respectivamente de digramas y trigramas.

Definición 3.1.3. Un criptosistema puede definirse como una quíntupla $\mathbb{S} = (M, C, K, e, \delta)$, donde :

- M es el conjunto de textos planos (sin cifrar).
- C es el conjunto de textos cifrados.

- K es el conjunto de claves utilizadas en el proceso de encriptado y desencriptado.
- $e_k : M \longrightarrow C$, $k \in K$, es la función de encriptación y transforma cada texto plano m en un texto encriptado $e_k(m)$ dependiente de cada $k \in K$.
- $\delta_k : C \longleftrightarrow M$, es la función de desencriptado y transforma los textos cifrados en textos planos dependiendo de la clave utilizada [75].

Para cada texto plano $m \in M$ y cada clave $k \in K$ se tiene

$$\delta_k\left(e_k\left(m\right)\right) = m \tag{3.1}$$

3.2. Criptografía simétrica *versus* asimétrica

Atendiendo al tipo de claves y al modo en que se utilizan en las funciones e_k y δ_k, podemos establecer una clasificación de los sistemas criptográficos que da lugar a dos grandes grupos conocidos como criptografía simétrica y asimétrica [75].

Definición 3.2.1. La criptografía simétrica, también conocida como criptografía de clave secreta, engloba todos los sistemas que utilizan la misma clave para encriptar y desencriptar.

A esta categoría pertenecen todos los sistemas clásicos o históricos que se desarrollaban de forma manual [32] e incluso aquellos que utilizaban sistemas mecánicos, como la conocida Enigma [39]. Los sistemas criptográficos simétricos modernos se dividen en dos grandes categorías, conocidas como Cifrado en bloque (*block cipher*) [21] y Cifrado en flujo (*stream cipher*) [72], [69].

Definición 3.2.2. La criptografía asimétrica, conocida también como criptografía de clave pública, se caracteriza por utilizar claves distintas en los procesos de encriptado y desencriptado. Estas dos claves están matemáticamente relacionadas y

permiten que una de ellas se haga pública (*clave pública*) mientras que la otra debe permanecer oculta al resto de los usuarios (*clave privada*) [75].

La relación que existe entre la clave privada y la clave pública permite desarrollar sistemas criptográficos en los que todo aquello que se encripta con la clave pública de un usuario solo puede ser desencriptado con la correspondiente clave privada que solo conoce el destinatario autorizado. De este modo se soluciona el inconveniente de la gestión de claves de los criptosistemas de clave simétrica en los que necesariamente ha de establecerse un canal seguro de comunicación para intercambiar la clave secreta que permite encriptar y desencriptar. La utilización de dos claves distintas, aunque relacionadas, permite también proporcionar de forma natural un servicio de firma digital, invirtiendo el orden en que se aplican las claves. Así, si un usuario utiliza su propia clave privada para encriptar, cualquier otro usuario podría desencriptarlo utilizando la correspondiente clave pública, que es conocida por todos, pero al mismo tiempo quedaría garantizada la procedencia del mensaje pues solo el usuario conocedor de la clave privada pudo haber generado el mensaje encriptado.

Por otra parte, el número de claves secretas que se necesitan en un criptosistema simétrico con N usuarios, esto es, $N(N-1)/2$, se reduce a únicamente N en los sistemas de clave asimétrica. A diferencia de la criptografía simétrica, la seguridad de criptografía asimétrica reside en la dificultad computacional de realizar determinados cálculos matemáticos, como la factorización de números enteros de gran tamaño o el logaritmo discreto. Como consecuencia de ello, en general, estos criptosistemas conllevan tiempos de ejecución mayores y, por tanto, requieren mayor capacidad de computación de los equipos en los que se utilizan [61].

Para el propósito de esta tesis, nos vamos a centrar en la criptografía simétrica y más concretamente en el cifrado en flujo.

3.3. Cifrado en flujo

El cifrado en flujo es un sistema criptográfico simétrico donde el mensaje (texto plano), generalmente en formato binario, es procesado con un flujo de bits pseudoaleatorios (secuencia cifrante) mediante la función XOR. En consecuencia, utilizando la notación introducida en este capítulo, para un mensaje m compuesto por L bits, esto es, $m = m_1, m_2, ..., m_L$, se obtiene el cifrado $c = c_1, c_2, ..., c_L$ tal que

$$c_i = m_i + y_i \tag{3.2}$$

donde y_i representa cada uno de los bits de la secuencia cifrante que han sido obtenidos a partir de un PRNG. De este modo, el mensaje original puede recuperarse a partir del texto cifrado si el destinatario puede reproducir la misma secuencia pseudoaleatoria, por lo que

$$m_i = c_i + y_i \tag{3.3}$$

La seguridad de este tipo de cifrado reside, fundamentalmente, en la robustez de la secuencia pseudoaleatoria generada y, por tanto, el diseño del PRNG se convierte en la pieza esencial de todo cifrado en flujo. En general, la secuencia se genera a partir de una semilla, que suele formar parte de la clave simétrica de estos sistemas criptográficos.

Se puede considerar el cifrado en flujo como una evolución natural de los sistemas de cifrado clásicos que se presentan a continuación.

3.3.1. Precursores del cifrado en flujo

Cifrado de César

El cifrado del César, también conocido como código César, obtiene su nombre en honor al emperador romano Julio César, que lo usó en el siglo I a.C. para comunicarse con sus generales [43]. Este cifrado es un método de sustitución simple en el que cada letra del mensaje se sustituye por la que está 3 posiciones más a la derecha en el alfabeto romano. Así, por ejemplo, el mensaje "CAZA" se transformaría en "FDCD". El descifrado consiste en aplicar el mismo desplazamiento de 3 posiciones en sentido inverso.

A continuación, se introduce una definición más formal y genérica que engloba a todas las posibilidades de este tipo de cifrado, aplicando la idea de desplazar las letras de un mensaje x posiciones en el alfabeto de referencia.

Definición 3.3.1. Definimos el cifrado de César generalizado como aquel en el que un mensaje m es transformado en un mensaje cifrado c mediante la transformación de cada uno de los símbolos m_i que componen el mensaje m en otro símbolo que se corresponde con el que se encuentra k posiciones a la derecha en su alfabeto de referencia. Así, si consideramos un alfabeto de 27 letras y establecemos una correspondencia numérica con el conjunto $\mathbb{Z}_{27}$, un mensaje de L símbolos $m = (m_1, ..., m_L)$ con $m_i \in \mathbb{Z}_{27}$, $1 \leq i \leq L$ se transforma en un mensaje cifrado $c = (c_1, ..., c_L)$ con $c_i \in \mathbb{Z}_{27}$, $1 \leq i \leq L$, mediante las siguiente funciones de encriptado y el desencriptado:

$$c_i = e_k(m_i) = m_i + k \ \ (mod 27) \tag{3.4}$$

$$m_i = \delta_k(c_i) = c_i - k \ \ (mod 27) \tag{3.5}$$

Nótese que en este caso se verifica:

$$\delta_k = e_{27-k} \tag{3.6}$$

En este caso podemos observar que se está utilizando un solo alfabeto para el cifrado ya que la clave utilizada en el proceso de cifrado es siempre la misma [25], [75].

Cifrado de Vigenère

A lo largo de la historia podemos observar que los métodos para la encriptación de los datos fueron evolucionando hacia sistemas polialfabéticos, o lo que es lo mismo para cada letra en el texto plano se utiliza una clave de encriptación distinta. El más famoso de los métodos de cifrado polialfabético lo encontramos en 1586 gracias al francés Blaise de Vigenère, donde en su libro "Traicté des Chiffes" nos muestra el siguiente método de encriptado y desencriptado [2], [75].

Definición 3.3.2. Sea $m = (m_1, ..., m_L)$ un mensaje de L símbolos $m_i \in \mathbb{Z}_{27}$, $1 \leq i \leq L$ que se transforma en un mensaje cifrado $c = (c_1, ..., c_L)$ con $c_i \in \mathbb{Z}_{27}$, $0 \leq i \leq L$, y sea $k = (k_1, k_2, \cdots, k_L)$ con $k_i \in \mathbb{Z}_{27}$ la clave de encriptación. Definimos la función de encriptación de un sistema de sustitución polialfabético como

$$e_k(m) = e_k(m_1, \cdots, m_L) = (m_1 + k_1, \cdots, m_L + k_L) \tag{3.7}$$

y la función de desencriptado como

$$\delta_k(c) = \delta_k(c_1, \cdots, c_L) = (c_1 - k_1, \cdots, c_L - k_L) \tag{3.8}$$

El cifrado de Vigenere utilizaba como clave una palabra cuya longitud era inferior a la longitud del mensaje; razón por la cual se repetía una y otra vez hasta completar dicha longitud. Así, como ejemplo, para cifrar el mensaje "MAÑANA ATACAMOS AL ALBA" con la palabra clave "ROSA", se genera la clave "ROSAROSAROSAROSAROSA" de la misma longitud (20 símbolos sin contar los espacios) que el mensaje a cifrar. De esta manera la "M" se codificará con la letra "R", la "A" con la "O" y así sucesivamente.

Debemos notar que, en este tipo de sistemas, en teoría, el número de claves distintas es 27^t, siendo t la longitud de la palabra clave, considerando un alfabeto de 27 letras. La tabla 3.3.1 muestra algunos ejemplos.

t	2	4	6	8	10
27^t	729	531,441	$3,8 \cdot 10^9$	$2,8 \cdot 10^{12}$	$2,0 \cdot 10^{15}$

Tabla 3.1: Número de claves posibles para el cifrado de Vigenère

Aunque el cifrado de Vigenere presenta una robustez mucho mayor que los sistemas de sustitución homofónica, como el del César, hay que tener en cuenta que si tenemos un patrón que aparece dos veces en el texto plano de manera que los segmentos correspondientes en el texto cifrado también son iguales, y l es la distancia entre estas dos coincidencias, entonces l debería ser un múltiplo de la longitud t de la clave.

El criptoanálisis de Kasiski [43] se basa en esta observación; si hay un patrón (de longitud 2 o superior) repetido en el texto cifrado, entonces la distancia entre estos dos segmentos debe ser un múltiplo de t. Por lo tanto, buscando patrones de longitud 2 o superior repetidos en el texto cifrado y midiendo las distancias entre

ellos se puede obtener la longitud de la palabra clave que de forma repetida forma la secuencia cifrante.

Ejemplo 4. Para ilustrar el análisis propuesto por Kasiski, consideremos el siguiente texto cifrado, resultante de aplicar el método de Vigenere, en el que se pueden observar diversos patrones repetidos.

pn xxumiei vmtiabk paxn lk yip awbqcmrvzas hvmiunyrdk, k ew bjxm diy puxarqmy mlikitpev xwit, qyr kuypeebk torbzke csa mr mlizit mrxucx eclrzhuuw dcoqn hragdrsytu havviy hevfquzew qm rm medcoza iaqmya c nauoieqw ian sgzu unkrvoqrs, eqitavq zoftie, nazds yi kypvrag ohmsnxuevziyohmamt mkxvmt sewrtreclnnz qn frzrun, tnzg eu tewjgcgvwt. xa teqsqre imxeisa kuyevpqgx, csawiude pwt ql rbuhde hr mtugqni, lge thmyfa e yi bqnxn mt yip awbqcmrvzas zrvzutvra, yuerqw yg fmairudeq qtucmnt lmcmyqzmr pn kuyurvkgoisa lk poghukztsf mtfri pwsqrgvitfew l puybvra jq nitwiuow qm larqn akorigi. g qsxn xxumiei bqrwvwt xe wvoauevbv zdew zwjqlsf kuyevpqgxew, pwthivgqkzdsfm kx msqmra di awsuneqw kzikzij qn iy uge iqcwxfargm, e ql uhm zgvs imxpahrzu qxmgw, zdaw fc gpqyvaooisa xud peebk pe pn ugdirn irqmeai kz mmy vuhegvmtfow imtfiwrqy. ql iwmxoixb irqmea kuyermw g gtmyqfmr iy loeerb jgeigb lk xa qnyaune rv sul rbdkoiiabue viabozuiim, igys hau bawb i rm pvnkzuce gwzmlmqij pe pna udgeaqfmcmbvke mmyqzmrif irqmeaiy k le wmxmruhqg zadv. mt xa qnzoza eymsmne sck oorbkopa gbv kx nszjxq di ziwgirn u. bqrwvwtqs hr tg yauhqtm ervosm fyrzuz uxvtolahna var eymsmnmn, g ufref xuferpqge diy mpq, er czgotmpisqnxr bupaw yiy ooqhvooagvwtqs zvi xmdmb g zqlitzgro. makrgss yi ozfseugoisa zkxaxvdg m lef xxqvmfquzew zmzqovbtusigna kda gvnxmde pwt xa qnyaune rvosme. hvg hevfquz cszmxoiey aoz msqqlucepquzew qm rm medcoza wr czulmmw vmre pqldav yiy ooqhvooagvwtqs qvtofavra jq lsf mybarbtke dyeitfe pn oaqrvn kohip ravmnsyi e xow vbgxieawy puvnvzq le fmmgnhn oaqrvn uazdmnt. rms gblorignkoanif lk xaw

imxeisamy ooqrziuapra jq le ziwgirn naqrsa lk ecmszgpaw cwx ormcbumneyqyfaw ozofarvkue, y temygmmotkyergm ufrsf, tu cui pwtfrmocea a pn dootseqg nrmgituce fwhde pn nrate vbgxieai kz le oizmlpn lk yaxnxgz.

Las repeticiones de las cadenas "di" y "lk" permiten observar lo siguiente:

- Distancia entre las ocurrencias lk: $402 = 6 \cdot 67$, $306 = 6 \cdot 51$, $576 = 6 \cdot 96$, $42 = 6 \cdot 7$, $126 = 6 \cdot 21$.
- Distancia entre las ocurrencias di: $504 = 6 \cdot 84$, 89, 241, pero $241 + 89 = 330 = 6 \cdot 55$, $78 = 6 \cdot 13$

Teniendo en cuenta que la clave utilizada es de longitud 6, se obtiene por fuerza bruta el resultado.

Cifrado Vernam

Este cifrado es el único sistema cuya seguridad puede ser demostrada teóricamente conforme a las condiciones de secreto perfecto establecidas por Shannon en 1949 [76]. El secreto perfecto queda garantizado cuando el mensaje cifrado no ofrece información sobre el mensaje original. La teoría desarrollada por Shannon establece unas condiciones para lograrlo que corresponden con las características del cifrado Vernam [31] y que puede resumirse en que la clave sea realmente aleatoria, lo que implica que debe ser de la misma longitud que el mensaje a cifrar y no se debe reutilizar nunca con otro mensaje distinto. Es por esto que a este método se le conoce también como *One-time Pad* en alusión al único uso que se le puede dar a cada secuencia cifrante utilizada.

En el cifrado Vernam los textos planos y cifrados tienen ya un formato binario que nos acerca a los sistemas modernos actuales. La clave es, por tanto, una secuencia aleatoria de longitud superior al mensaje y la función de cifrado es la suma en $\mathbb{B} = \mathbb{Z}_2 = \mathbb{F}_2$ XOR del texto sin cifrar y la clave, bit a bit.

Como se puede observar, la única diferencia entre el cifrado Vernam y el cifrado en flujo es que la secuencia cifrante empleada es puramente aleatoria. Este hecho, aunque garantiza la seguridad del sistema dificulta enormemente su implementación en las redes telemáticas y sistemas de comunicaciones actuales donde la gran cantidad de usuarios que los utilizan convierte la gestión de las claves en una tarea inviable. Es por ello que el cifrado en flujo emplea secuencias pseudoaleatorias, que permiten ser reproducidas en el destino sin necesidad de ser transmitidas por un canal seguro independiente. Basta con enviar la semilla (de longitud mucho más corta) para poder generar la secuencia cifrante completa.

3.4. Criptografía Cuántica

Los esquemas QKD constituyen la punta de lanza de la criptografía cuántica. En líneas generales, la protección que ofrecen estos esquemas viene asegurada por las propiedades de la mecánica cuántica. En un mismo instante se pueden tener fotones polarizados en una de dos direcciones de referencia perpendiculares entre sí (vertical u horizontal), o de manera alternativa en una superposición de estas dos direcciones al mismo tiempo. Si determinamos la polarización del fotón, y coincide con una de las dos direcciones elegidas como referencia obtenemos la medida correcta (vertical u horizontal) y este fotón mencionado no se ve alterado.

Por el contrario, en el caso de que el fotón esté vibrando en una dirección intermedia (por ejemplo a 45 grados), como consecuencia de encontrarse en una superposición de las dos direcciones que se han tomado como referencia (horizontal y vertical al mismo tiempo), al tomar la medida de esta polarización se obtendrá uno de los dos valores de forma aleatoria (por ejemplo 50 % vertical 50 % horizontal) y el fotón estará sometido a una variación y se verá obligado a cambiar a la dirección que

hemos medido, lo que proporciona un mecanismo eficiente para detectar posibles interceptaciones.

Supongamos que dos personas, Ana y Beatriz, establecen su cadena de comunicación. Ana es el emisor del mensaje encriptado y Beatriz el receptor. Para poder comunicarse es necesario que los dos compartan una clave para saber como desencriptar el mensaje inicial. Para compartir la clave con Ana, Beatriz asigna un bit a cada fotón polarizado que va enviar a Ana (donde el 0 = Horizontal, el 1 = Vertical y 0 = -45 grados, 1 = $+45$ grados). Los fotones, que tendrán la posibilidad de ir polarizados en estas cuatro direcciones diferentes, se enviarán de forma completamente aleatoria. Al otro lado, está Ana que es el destinatario y que cuenta con dos dispositivos de detección distintos para medir estos fotones que le ha enviado Beatriz. Un detector para medir los fotones verticales y horizontales y otro segundo detector que solo puede medir los fotones en las otras dos direcciones a $+45$ grados y -45 grados.

Ana también va a usar esos dispositivos de detección de forma aleatoria por lo que tiene un 50 % de probabilidades de medir los fotones que transmite Beatriz con sus detectores correspondientes. Cuando termina de medir el mensaje de Beatriz, Ana le cuenta a ella la sucesión aleatoria de detectores que ha estado utilizado y Beatriz la compara con su secuencia y le dice dónde ha usado el detector incorrecto.

Así Ana sabe qué porción de la secuencia de Beatriz ha almacenado y anotado de forma correcta y cual ha podido dar errores de forma aleatoria y debe desecharla, del cardinal de fotones enviados nos quedan, en promedio, la mitad de ellos. El mensaje cifrado y la secuencia de detectores usados por Beatriz se comparten utilizando los métodos clásicos.

Si alguna persona no deseada, intenta escuchar la secuencia de fotones que Beatriz le está mandando a Ana, al medir los fotones enviados, también está mo-

dificando su estado cuántico y lo que le llega a Ana ya no es lo mismo que envió Beatriz. ¿Cómo saben ellos que hay un intruso que ha provocado errores en la cadena de fotones? Porque Ana elige un trozo de la secuencia de bits que ha resultado al medir los fotones de Beatriz (con los detectores correctos) y se la envía a ella. Si Beatriz ve que incluso habiendo usado los detectores en el orden correcto, Ana presenta bits erróneos en esa pequeña parte, informará de la violación de la seguridad del canal, porque algo está alterando los fotones por el camino.

3.4.1. Comparativa de los sistemas de distribución de claves cuánticas

La física cuántica nos ofrece un método seguro de información teórica para distribuir información de manera segura. La QKD permite que dos personas, Ana y Beatriz, que están conectadas por un canal cuántico, como pudiera ser un enlace óptico pasivo, intercambien información de manera segura. Ana transmite información aleatoria codificada en estados cuánticos de luz a Beatriz, quien mide los estados cuánticos recibidos utilizando diferentes detectores. Después de este paso, Ana y Beatriz aplican una serie de algoritmos en sus datos para extraer una clave criptográficamente segura. Se ha demostrado que QKD es seguro en términos teóricos de la teoría de la información, lo que significa que cualquier intento de acceder a la información por un tercero será detectado inmediatamente y sólo las personas que disponen de la clave pueden acceder a ella.

En general, existen dos enfoques diferentes (pero complementarios) de QKD. De manera análoga a la dualidad partícula-onda de la luz, cada uno de los dos enfoques ponen el énfasis en el aspecto corpuscular o ondulatorio del portador cuántico de información para dar la seguridad al sistema.

El primero de estos enfoque es el uso de la variable discreta DV-QKD, en el que la naturaleza de partícula de la luz es enfatizado para lograr una distribución segura de claves. Por ejemplo, el transmisor codifica la información sobre las propiedades físicas de un solo fotón, como puede ser su estado de polarización. Los detectores de resolución de fotón único se utilizan para medir los estados cuánticos recibidos.

El segundo enfoque viene determinado por la variable continua CV-QKD, en el que se enfatiza la naturaleza ondulatoria de la luz para lograr una distribución de claves segura. En este segundo enfoque, el transmisor codifica la información en las cuadraturas de amplitud y fase de un láser coherente brillante, y el receptor mide las cuadraturas de la luz utilizando detectores homodinos equilibrados [48].

Parte III

Registros de desplazamiento para la generación de secuencias pseudoaleatorias

Capítulo 4

Máquina de estados finitos

4.1. Definición

Definición 4.1.1. [34] Una máquina de estados finitos consiste en una colección finita de estados $K = \{K_i\}_{\{i=1,\cdots,n\}}$ que de manera secuencial acepta una sucesión de valores (llamados inputs) obtenidos de un conjunto finito $A = \{a_i\}$ y a partir de estos genera secuencia de valores de salida (llamados outputs) que pertenecen a un conjunto de valores finitos $B = \{b_i\}$. Lo que es más, existe una función de salida μ que computa el resultado (output) dado un valor de entrada (input) considerado este último como el input actual en un estado determinado. Existe del mismo modo una función δ que determina y calcula el valor del siguiente estado teniendo en cuenta el estado actual. Por lo tanto $\mu(K_n a_n) = b_n$, mientras que por otro lado $\delta(K_n, a_n) = K_{n+1}$.

	0	1
R	S, α	T, β
S	S, β	S, τ
T	T, α	R, α

Tabla 4.1: Tabla de función de una máquina de estados finita

4.2. Representaciones de una máquina de estados finita

Una máquina de estados está completamente determinada por una tabla, la cual identifica el output y el siguiente estado para cada posible combinación de estados y de inputs. Un ejemplo de esta tabla lo podemos visualizar en la figura 4.1, donde la máquina de estados tiene tres posibles estados R, S y T, acepta como valores de entrada los registros 0 y 1 y genera los valores de salida α, β y τ.

Si nos dan un estado inicial de una máquina y una secuencia de entrada, entonces podemos computar la secuencia de salida y determinar el estado final de la máquina, o lo que es lo mismo, el estado de la máquina después de haber procesado el último de sus inputs. Por ejemplo, si la máquina de estados es aquella que hemos definido en 4.1, nuestro estado inicial es R e introducimos los valores 01100101, nosotros podemos ver que se obtiene la secuencia $\alpha\tau\tau\beta\alpha\tau\beta\beta$, tal y como se muestra en la figura 4.2. Además se puede observar que el estado final es T

Una alternativa a esta manera de representar el comportamiento secuencial de las máquinas de estados finitos es el método conocido como el método del diagrama. En este caso los tres estados de la máquina se representan como los nodos de un grafo. Los inputs son representados como sendas o caminos orientados en el grafo, desde un nodo a otro de acuerdo con el próximo estado. En cada uno de las aristas del grafo se indica entre paréntesis la salida u output correspondiente a un símbolo

Tiempo	Input	Estado	Salida
1	0	R	α
2	1	S	τ
3	1	S	τ
4	0	S	β
5	0	R	α
6	1	S	τ
7	0	S	β
8	1	R	β
9		T	

Tabla 4.2: Comportamiento secuencial de una máquina de estados finita

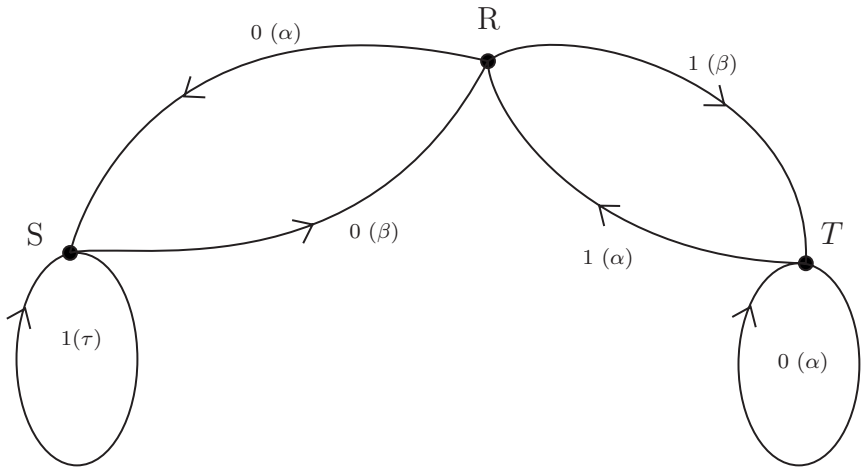

Figura 4.1: Grafo del comportamiento de una máquina de estados finita

en particular. Entonces si estamos en el estado Ry recibimos el O el diagrama muestra que avanzamos al estado S y escribimos el símbolo de salida α. Por el contrario si estamos en el estado R y recibimos el 1, avanzaremos al estado T y escribiremos el símbolo β tal y como se muestra en la figura 4.1.

En un principio podemos afirmar que cada máquina de estados finita está biunívocamente determinada por su tabla de funcionamiento o por su grafo. La tabla parece un método más amigable para representar el funcionamiento de una máquina de estados, no obstante el método a través del diagrama de su grafo es más fácil

de seguir cuando se necesita seguir rastrear el funcionamiento de una secuencia de entrada determinada.

4.3. Equivalencia entre dos máquinas de estados finitas

Desde un punto de vista práctico necesitamos establecer cuando dos máquinas de estados distintas realizan el mismo trabajo. O lo que puede resultar mucho más interesante. Si dos máquinas realizan el mismo trabajo cuál de ellas es más simple, esto es, tiene el menor número de estados.

Definición 4.3.1. [34] Diremos que dos máquinas de estados finitas que aceptan valores en el mismo conjunto de inputs y produzcan valores en el mismo conjunto de outputs, se dirá que son equivalentes si y solamente si ambas máquinas realizan las mismas transformaciones desde el conjunto de todas las secuencias inputs en el conjunto de secuencias de salida outputs. Esto es, ambas máquinas se dirá que son equivalentes si y solamente si, ambas dos procesan los datos de la misma manera obteniendo así el mismo resultado.

En la figura 4.2, podemos observar el grafo de una máquina de estados que es mucho más compleja que aquella presentada en el la figura 4.2. Además esta última conlleva un conjunto de cuatro estados en lugar de 3. Sin embargo, podemos comprovar que ambas máquinas son equivalentes. Si nos fijamos en T y en U, podemos comprobar que estos dos estados no alteran el comportamiento de la máquina de estados. Además si comprobamos el resultado obtenido en ambas secuencias de salida, éstas son idénticas para ambas máquinas para cualquier input.

En general existen distintos algoritmos que nos permiten determinar si dos

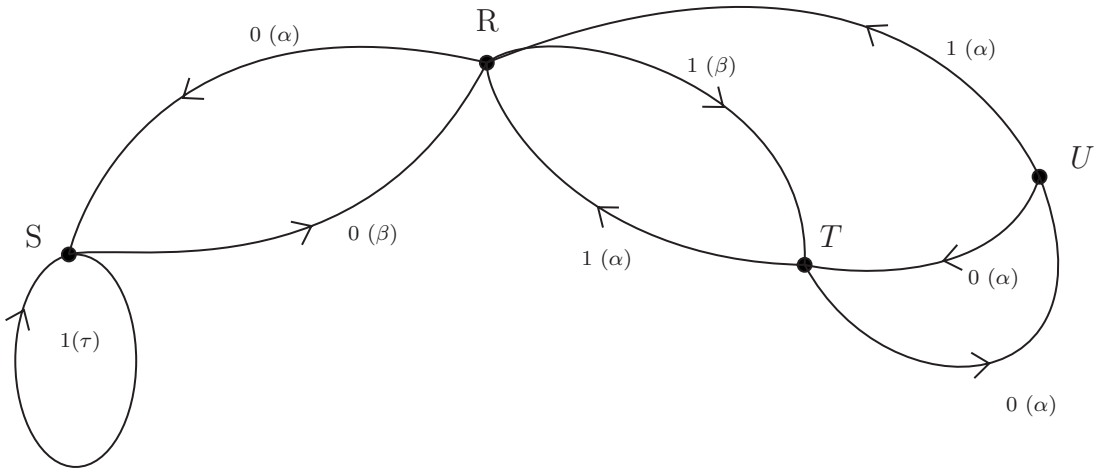

Figura 4.2: Grafo equivalente al grafo 4.1 menos eficiente

máquinas de estados finitas son o no son equivalentes, además nos ayudan a determinar dentro de el conjunto de máquinas de estados finitas, cuál de ellas es la que es minimal teniendo en cuenta cardinal del conjunto de estados.

4.4. Análisis de la autonomía de las máquinas

El comportamiento de una máquina de estados se estudiará de acuerdo a las resultados obtenidos, outputs, en relación a cómo actúa con respecto a las sucesiones de entrada, inputs. Dado que el número de secuencias de entrada finitas es infinito, es imposible testar la acción de la máquina de estados sobre el conjunto de todos los inputs posibles. Sin embargo, es muy útil y razonable, testar cómo actúa la máquina sobre ciertas secuencias de entrada o inputs [88]. Algunas secuencias que parecen razonables serán las siguientes:

1. La secuencia de nula, consistente en una secuencia de entrada en la que todos los elementos son nulos.

2. La secuencia constante, consistente en una secuencia en la que todos los términos a partir de un lugar son todos unos.
3. La secuencia de impulso, en la que el primer elemento es un 1, seguido de infinitos ceros.
4. La secuencia de paso, consistente en un número finito de unos, seguido de un número infinito de ceros.

La interacción de una máquina de estados sobre estos tipos de secuencias de entrada o inputs, no es suficiente para poder caracterizar el comportamiento de la máquina completamente. Entonces, dos máquinas pueden producir el mismo resultado sobre estos cuatro tipos de secuencias y ser al mismo tiempo no equivalentes. No obstante, cualesquiera dos máquinas equivalentes producirán las mismas secuencias de salida, cuando estén alimentadas con las mismas secuencias de entrada, siendo estas secuencias de entrada que acabamos de listar muy convenientes para poder clasificar, al menos parcialmente, a estas máquinas.

Si alimentamos una máquina de estados finita con un conjunto finito de valores no nulos, y a partir de ese lugar el conjunto de conjunto de valores es nulo, entonces la secuencia de salida que se obtiene a partir de este lugar donde los valores son todos nulos, se conoce como el comportamiento autónomo de la máquina de estados. Las secuencias de entrada marcadas como I ,III y IV, de la lista 4.4, son especialmente útiles para conocer el comportamiento autónomo de una máquina.

Existe un teorema muy importante que se aplica al conjunto de todas las máquinas de estados finita y al conjunto de los 4 tipos de inputs listados en 4.4.

El conjunto de todos los periodos de las secuencias de salida outputs generados por una máquina de estados finita son invariantes en el análisis y clasificación de una máquina de estados finita. En otras palabras.

Teorema 4.4.1. *[34] Si la secuencia de entrada aplicada a una máquina de estados*

finita es constante a partir de un lugar, entonces la secuencia de salida es periódica.

Demostración. Existe un conjunto finito de estados. Entonces durante la parte constante de la secuencia de entrada, la máquina de estados alcanzará un estado anterior. Si la máquina está en el estado S en ambos tiempos t_1 y t_2, entonces estará en el estado S' en el estado $t_1 + 1$ y $t_2 + 1$, por lo tanto en este caso tenemos determinada la periodicidad. □

Utilizando el mismo argumento podemos demostrar el siguiente teorema.

Teorema 4.4.2. *Si la secuencia de entrada de una máquina de estados finita es periódica, entonces la secuencia de salida es periódica.*

Veamos algunos ejemplos.

Tomemos como ejemplo la máquina de estados que hemos descrito en las figuras 4.1 y 4.1. Supongamos un estado inicial R, y veamos cómo actúa sobre cada uno de los 4 tipos de secuencias descritas en la lista 4.4.

1. Si la secuencia de entrada es el $000\cdots$ entonces la secuencia de salida será $\alpha\beta\alpha\beta\alpha\beta\alpha\beta\cdots$ y la secuencia de estados será $RSRSRSRSRS\cdots$.
2. Si la secuencia de entrada es $1111\cdots$ entonces la secuencia de salida será $\beta\alpha\beta\alpha\beta\alpha\beta\alpha\cdots$ y la secuencia de estados será $RTRTRTRT\cdots$.
3. Si la secuencia de estrada es $10000\cdots$ entonces la secuencia de salida será $\beta\alpha\alpha\alpha\alpha\alpha\cdots$ y la secuencia de estados será $RTTTT\cdots$.
4. Si la secuencia de entrada es $\underbrace{1111111}_{n}000\cdots$ entonces se nos pueden presentar distintos casos.
   a) Si n es impar, entonces la secuencia de salida será $\underbrace{\beta\alpha\beta\alpha\beta\cdots\beta}_{n}\alpha\alpha\alpha\cdots$ y la secuencia de estados será $\underbrace{RTRTRTRT\cdots R}_{n}TTTTT\cdots$.

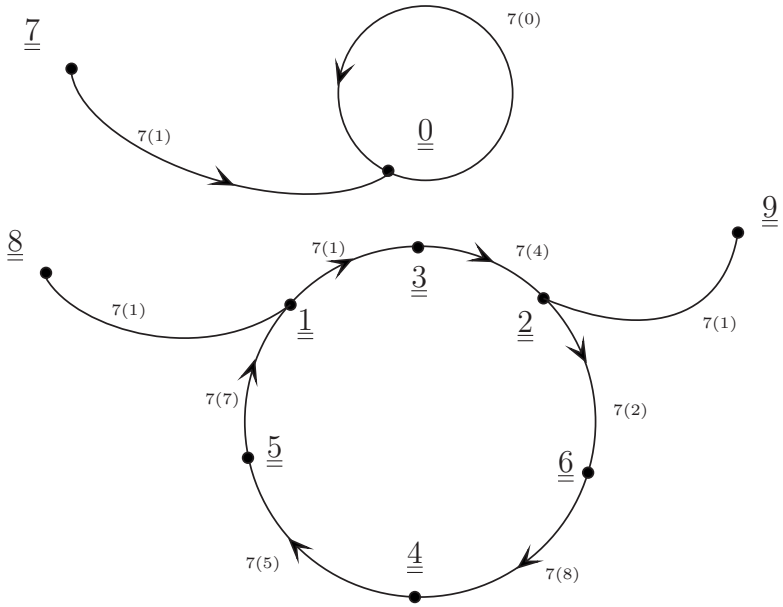

Figura 4.3: Grafo de una máquina de estados. Resultados al dividir entre 7

b) Si n es par, entonces la secuencia de salida será $\underbrace{\beta\alpha\beta\alpha\beta\cdots\alpha}_{n}\,\alpha\beta\alpha\beta\cdots$ y la secuencia de estados será $\underbrace{RTRTRTRT\cdots T}_{n}\,RSRSRSRS\cdots$.

En la figura 4.3 podemos ver parte del diagrama de estados de una máquina que representa la división decimal. Específicamente hablando representa la división decimal donde el divisor es 7. Para dividir entre 7, partimos de un estado inicial n y usamos la secuencia inicial $7777\cdots$. Siguiendo el diagrama 4.3 podemos obtener el resultado de los cálculos obtenidos.

- $0/7 = 000000\cdots$
- $1/7 = 142857\cdots$
- $2/7 = 285714\cdots$
- $3/7 = 428571\cdots$
- $4/7 = 571429\cdots$
- $5/7 = 714286\cdots$
- $6/7 = 857143\cdots$
- $7/7 = 100000\cdots$
- $8/7 = 114286\cdots$
- $9/7 = 128571\cdots$

El diagrama representado en el grafo de la figura 4.3 revela que las periodicidades posibles para este cálculo en concreto son 1 y 6.

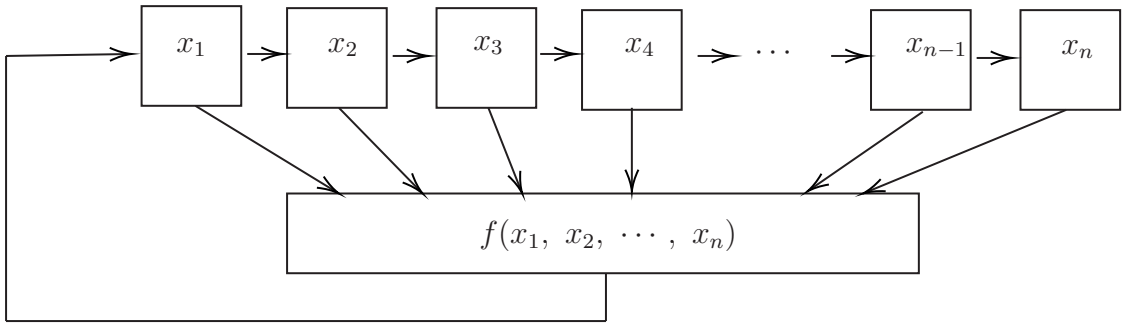

Figura 4.4: Diagrama general de un registro de desplazamiento con realimentación

4.5. Registros de desplazamiento lineales y no lineales

En la figura 4.4,podemos visualizar el esquema básico un registro de desplazamiento no lineal con realimentación. Cada una de las casillas o celdas etiquetadas con los valores $\{x_1, x_2, \cdots, x_n\}$ almacenará un dato o registro binario, con lo que cada $(x_i) \in \mathbb{Z}_2$. Con intervalos periódicos llamados impulsos de reloj, el valor de la celda x_i se transfiere a la celda x_{i+1} $\forall i \in \{1, 2, \cdots, n-1\}$. Para obtener el valor de la celda x_1 hacemos actuar la función f sobre $(x_1, x_2, \cdots, x_n)$ y ese valor se almacenará en la celda de x_1, por lo que $x_1 = f(x_1, x_2, \cdots, x_n)$.

Definición 4.5.1. [34] Una función f de n variables $\{x_1, x_2, \cdots, x_n\}$ definidas cada una de ellas sobre $\mathbb{Z}_2$ y cuya imagen se encuentre también sobre $\mathbb{Z}_2$ recibe el nombre de función booleana. Por lo tanto si f es booleana

$$f : \mathbb{Z}_2 \times \mathbb{Z}_2 \times \cdots^{(n} \cdots \times \mathbb{Z}_2 \longrightarrow \mathbb{Z}_2 \tag{4.1}$$

Nótese que existen 2^{2^n} funciones booleanas distintas definidas sobre un espacio $n-$ dimensional.

Al almacenamiento binario de cada uno de los x_i se les conoce como etapas del

registro de desplazamiento, mientras que a cada uno de los vectores considerados $n-$upla binaria de valores binarios que se almacena en $(x_1, x_2, \cdots, x_n)$ se le llama estado del registro de desplazamiento. A la función f de nuestro esquema se la conoce como función de realimentación.

En cada pulso de reloj, hay una transición de un estado al siguiente. Obviamente, existen 2^n estados distintos, tantos como partes de un conjunto de cardinal n. Si nosotros comenzamos alimentando el registro de desplazamiento con uno de estos estados, el registro de desplazamiento irá progresando hacia los siguientes. De acuerdo con el teorema 4.4.1 la sucesión resultante de estados ha de ser periódica.

En el caso en el que la función de realimentación f se puede expresar en la forma:

$$f(x_1, x_2, \cdots, x_n) = c_1x_1 \oplus c_2x_2 \oplus \cdots \oplus c_nx_n \tag{4.2}$$

donde cada $c_i \in \mathbb{Z}_2$ y $\oplus$ representa la suma binaria en $\mathbb{Z}_2$ entonces el registro de desplazamiento que observamos en la figura 4.4 diremos que el lineal. Dado que existen 2^n maneras de escoger n-tuplas binarias de constantes c_i distintas, solamente 2^n de entre las 2^{2^n} totales serán funciones lineales.

Capítulo 5

Sucesiones con propiedades aleatorias

5.1. Introducción

Podemos encontrar una amplia variedad de campos y situaciones en las que las sucesiones numéricas aleatorias tienen una importancia capital. Podemos destacar campos como la criptografía, electrónica, matemática discreta, etc... en los que el uso de secuencias de este tipo es completamente necesario para el desarrollo de este tipo de conocimientos.

En cada uno de estos campos resulta muy conveniente disponer de un método simple para que pueda generar una sucesión numérica que aunque no sea 100 % periódica al menos se parezca lo máximo posible.

Además se considera completamente necesario que las máquinas puedan generar la misma secuencia numérica. Por ejemplo, si una secuencia aleatoria se ha utilizado para encriptar un mensaje, el receptor del mensaje cifrado, deberá ser capaz de generar la misma secuencia para poder descifrarlo.

5.2. Aleatoriedad de una secuencia numérica

En un sentido, ninguna sucesión numérica finita puede ser considerada como aleatoria. Lo mejor que se puede hacer es destacar ciertas propiedades que pueden ser asociadas con el concepto de aleatoriedad y aceptar que cualquier sucesión que tenga esas propiedades sea considerada como sucesión aleatoria.

Vamos a considerar en nuestro estudio, sucesiones binarias compuestas por *ünos"* y por *"menos unos"*, correspondientes al encendido y apagado de los sistemas electrónicos.

El ejemplo más familiar y simple que podemos encontrar es el del lanzamiento de una moneda no trucada. En este ejemplo podemos identificar las caras con el valor 1 y las cruces con el valor -1. Las siguientes propiedades se asociarán a al concepto de aleatoriedad.

1. El número de caras deberá ser aproximadamente igual al de las cruces.
2. Las rachas de caras y las rachas de cruces ocurrirán con menos frecuencias si es una racha larga que si es una racha corta.
3. Además las sucesiones aleatorias poseen una función de correlación que alcanza su punto máximo en el centro y va disminuyendo hacia las colas.

5.3. Autocorrelación

Definición 5.3.1. [34] Sea $\{a_n\} = \{a_1, a_2, \cdots, a_n, \cdots\}$ una sucesión. Se define la función de autocorrelación como:

$$C(\tau) = \lim_{x\to\infty} \frac{1}{N} \sum_{n=1}^{\infty} a_n a_{n+\tau} \tag{5.1}$$

supuesto que este límite exista.

En particular si $\{a_n\}$ es una sucesión periódica con periodo p, entonces este cálculo se reduce a una suma finita.

$$C(\tau) = \frac{1}{p} \sum_{n=1}^{p} a_n a_{n+\tau} \tag{5.2}$$

En este caso en concreto, τ puede ser considerado, como una fase del registro de desplazamiento. La función $C(\tau)$ mide cuánto se parecen una sucesión y ella misma desplazada. El valor de $C(\tau)$ es siempre el mayor para $\tau = 0$ y si $\{a_n\}$ es aleatoria $C(\tau)$ tomará valores muy pequeños para cualesquiera otros valores de τ.

5.4. Introducción y antecedentes históricos

Los números pseudoaleatorios son muy útiles en una amplia variedad de aplicaciones. Por ejemplo son frecuentes las utilizaciones en usos criptográficos, test de primalidad, estadística computacional.

Por *números aleatorios* entenderemos una sucesión de números en la que cada término se obtiene al azar, con una determinada probabilidad de que un rango de valores sea elegido. La existencia de números aleatorios es demostrable gracias a los axiomas probabilísticos de Kolmogorov, sin embargo este resultado no nos da un método para la construcción de una sucesión de números aleatorios.

Anterior a 1940, los simuladores de números aleatorios se basaban en procesos físicos mediante lanzamientos de dados, el giro de ruletas, reparto de cartas de juego o extracción de datos de una tabla de doble entrada. En los años 40 se comenzó con el desarrollo de máquinas para producir números pseudoaleatorios y el los 50, se comenzó con la utilización de ordenadores para generar números aleatorios. No obstante, los números aleatorios eran producidos mediante medios mecánicos y normalmente el sistema no funcionaba del todo bien debido al mal funcionamiento de

hardware de los ordenadores. Además la mayor preocupación era que estos números no podían ser verificados con el programa informático. Métodos más simples para obtener secuencias de manera aleatoria fueron investigados y varios se investigaron varios sistemas recursivos no-lineales. El primero en realizar estos estudios fue John Von Neumann a través del método de cuadrado medio. Este método genera 4 números aleatorios dado un número aleatorio inicial de 4 dígitos. Por ejemplo, si consideramos el 6591, entonces su cuadrado es 43441281 del cual extraemos sus cuatro dígitos centrales, esto es, 4412. A este segundo número de le volvía a aplicar el método y así sucesivamente.

Obviamente esta sucesión no puede ser considerada como realmente aleatoria, dado que está completamente determinada por la semilla inicial.

Por otro lado, para las aplicaciones criptográficas, aunque el comportamiento aleatorio es deseable para garantizar un alto nivel de seguridad, desde el punto de vista computacional es condición necesaria que la secuencia se pueda reproducir de manera idéntica en los dos extremos del canal. En consecuencia centraremos nuestros esfuerzos en la generación de números pseudoaleatorios.

En cualquier caso el término pseudoaleatorio tiene el prefijo pseudo que significa falso, que parece que no es real. Un aspecto importante en el método utilizado es el periodo de repetición y la dependencia de la semilla utilizada.

Como ejemplo, el método anterior se obtienen periodos muy cortos cuando se utiliza el valor inicial 6300. Si consideramos el 6300, los valores obtenidos serán 6900, 6100, 2100, 4100, 8100, 6100, 2100, 4100, ...

Excluyendo el número 6900, la sucesión se repite indefinidamente. De hecho, aunque la sucesión original parece ser impredecible, parece que si consideramos un determinado valor inicial entonces llegamos a una repetición de números.

La llegada de los computadores hizo que se cambiara la forma de abordar la gene-

ración de números aleatorios.

5.5. Postulados de aleatoriedad

Vamos a establecer unos postulados que se han de tener en cuenta para poder asegurar que una sucesión numérica es o no pseudoaleatoria. Vamos a suponer que disponemos de sucesiones numéricas binarias.

Definición 5.5.1. Sea $\{a_n\}$ una sucesión binaria y periódica. Diremos que esta sucesión es pseudoaleatoria si verifica:

1. En cada periodo, el número de veces que aparece el 1 es casi igual al número de veces que aparece el -1. Más precisamente esta diferencia nunca será mayor a 1.
2. En cada periodo, la mitad de las rachas tienen longitud 1, una cuarta parte tienen longitud 2, una octava parte tienen longitud 3, etc... siempre y cuando la longitud de las rachas exceda a 1. Lo que es más, por cada una de estas longitudes en las rachas, hay un mismo número de rachas de ceros que de unos.
3. La función de auto correlación que hemos definido en 5.1, toma dos valores.

$$C(\tau) = \sum_{n=1}^{p} a_n a_{n+\tau} = \begin{cases} p & \text{si} \quad \tau = 0 \\ \\ K & \text{si} \quad 0 < \tau < p \end{cases} \tag{5.3}$$

[34]

Esta función que toma sólo dos valores es de duma importancia. Muchas nos puede ayudar a determinar si se verifican o no las condiciones necesarias para demostrar que una sucesión es aleatoria.

Definición 5.5.2. [34] Una sucesión numérica $\{a_n\}_{n\in\mathbb{N}}$ que verifique las condiciones establecidas en 5.5.1 se dirá que es una sucesión pseudoaleatoria.

Ejemplo 5. Vamos a lomar una sucesión periódica de periodo $p = 7$. Sea la sucesión $A = \{1, 1, 1, -1, 1, -1, -1, \cdots\}$. A partir del séptimo de los términos esta sucesión se repite. Nótese que el 1 aparece 4 veces y que el -1 aparece 3 veces con lo que entonces se verifica la primera de las condiciones. Si nos fijamos en las rachas, la mitad de ellas tienen longitud 1 y un cuarto de ellas longitud 2. La autocorrelación es 1 en la fase y $\frac{1}{7}$ fuera de fase. La condición número 2 también se cumple. Con lo que podríamos considerar esta secuencia como pseudoaleatoria.

Capítulo 6

Sucesiones y registros de desplazamiento

6.1. Registros de desplazamiento con realimentación

Definición 6.1.1. Llamaremos registro de desplazamiento a una disposición de r celdas dispuestas en linea. Cada celda almacenará un valor 1 ó 0. Atendiendo a la notación de ingeniería electrónica o bien encendido o bien apagado, de tal manera que, en cada impulso de reloj el registro rota y el contenido de cada celda se transfiere a la siguiente celda.

En la figura 6.1, podemos observar el esquema básico de un registro de desplazamiento.

Si el registro de desplazamiento no es alimentado con ninguna señal al final de los r desplazamientos entonces el registro terminará con un valor de sus celdas nulo. Una manera evitar esta situación y que el registro se mantenga activo consiste en

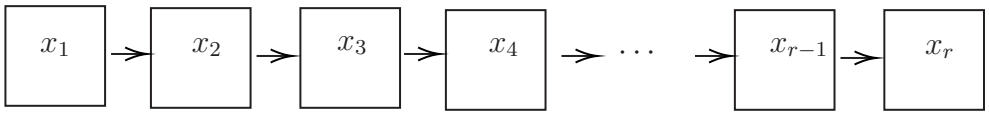

Figura 6.1: Esquema básico de un registro de desplazamiento

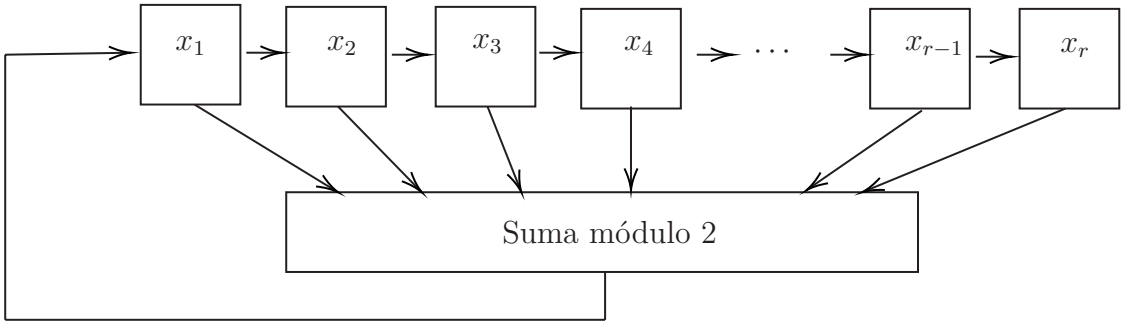

Figura 6.2: Esquema general de un registro de desplazamiento con realimentación

realimentar el registro de desplazamiento. Podríamos coger los valores de algunas de las r celdas y tras realizar unas operaciones, introducir el resultado de las mismas como valor de la primera de las celdas. O lo que es lo mismo, utilizar los valores introducidos en algunas de la r celdas o en todas y operar con ellas módulo 2 para obtener un nuevo valor que se introducirá en la celda x_1. Una vez realizadas las operaciones y antes de introducir el valor en la celda x_1, entonces realizamos un desplazamiento de los valores de las celdas, de tal manera que $x_{r-j} = x_{r-j-1}, \forall j \in \{0, 2, \cdots, r-1\}$ tal y como se muestra en la figura 6.2.

En general, los enteros que son pares se reemplazan con el valor 0 y los enteros impares se reemplazan con el valor 1.

6.2. Periodicidad de los estados de los registros de desplazamiento

No es difícil demostrar que la sucesión de los estados de un registro de desplazamiento es periódica.

Teorema 6.2.1. *La sucesión de los estados de un registro de desplazamiento es periódica con longitud de periodo p, donde* $p \leq 2^r - 1$*, donde r es el número de celdas del registro de desplazamiento.*

Demostración. Cada estado en el registro de desplazamiento está determinado por el anterior. Entonces, si uno de los estados es igual a uno de los anteriores se verificará que los siguientes también serán iguales a éste. Si disponemos de r celdas y cada una de ellas almacena dos posibles valores, entonces el conjunto de estados coincidirá con el cardinal de las partes de un conjunto de r elementos, esto es 2^r. Por lo tanto dado que tenemos ese cardinal de posibles estados, entonces se verifica que habrá una repetición después de a lo sumo $2^r + 1$ impulsos o desplazamientos. Por lo tanto la periodicidad será $p \leq 2^r - 1$ □

Debemos destacar que el resultado es cierto sin pérdida de generalidad para cualquier valor del estado inicial que se haya tomado.

6.3. Modelos de generación de números pseudo-aleatorios

La mayoría de los modelos utilizados en la actualidad se basan en uno de los tres modelos siguientes:

1. $k-$ ésimo generador por recurrencia lineal. Este tipo de generador produce una sucesión $\{x_i\}_{i\geq 0}$ de números pseudoaletorios definidos de manera recurrente mediante:

$$x_{i+k} = \sum_{j=1}^{k} a_{k-j} x_{i+k-j} + c \text{ (mod m) } 0 \leq x_i < m \tag{6.1}$$

donde $a_0, a_1, \cdots, a_{k-1}, c$ son todos valores enteros en $\mathbb{Z}_m$ y donde $a_0 \neq 0$ y m es un entero positivo. Si $k = 1$ tal generador de números se llama generador por congruencia lineal. Si además $c = 0$ entonces el método se llamará generador por congruencial puro multiplicativo.

2. Generador por recurrencia lineal módulo 2.

   Es una sucesión $\{b_i\}$ de ceros y unos dada por la relación:

$$b_i \equiv \sum_{j=1}^{k} a_j b_{i-j} \text{ (mod 2)} \tag{6.2}$$

   donde $a_0, a_1, \cdots, a_{k-1} \in \mathbb{Z}_2$, $a_k = 1$ y cada $b_j \in \mathbb{Z}_2$

3. Generadores por congruencia no lineal.

   Estos tipos de generadores son de la forma:

$$x_{i+1} \equiv f(x_i) \text{ (mod m) } ; 0 \leq x_{i+1} < m \tag{6.3}$$

   donde f es una función no lineal. Un ejemplo de este tipo de funciones es $f(x) = ax^{-1} + b$ donde $x \in \mathbb{Z}_m, x \neq 0$, donde a y b son dos enteros positivos, $m = p \geq 5$ es primo y x^{-1} es el único primo positivo menor que p tal que $xx^{-1} \equiv 1 \text{ mod} p$, llamado inverso multiplicativo de x módulo p.

6.4. Generadores lineales por recurrencia de primer orden

6.4.1. Método por congruencia lineal

Algunos casos especiales de este esquema fueron introducidos por D.H. Lehmer in 1940. Algunos de ellos son los más frecuentes como generadores de números pseudoaleatorios, ya que son muy fáciles de implementar y además son muy rápidos desde el punto de vista computacional. El método por congruencia lineal produce una sucesión $\{x_i\}_{i\geq 0}$ definida de manera recursiva por

$$x_{i+1} \equiv ax_i + c \pmod{\text{m}} \ , \ 0 \leq x_{i+1} < m \tag{6.4}$$

donde los enteros m llamado módulo, a llamado multiplicador, c llamado incremento y x_0 llamado semilla se escogen $0 \leq a, c, x_0 < m$

Como se ha mencionado anteriormente, el caso especial en el que $c = 0$ llamado método congruencial puro multiplicativo, fue introducido originalmente por Lehmer, aunque el indicó que $c \neq 0$ también era una posibilidad. Sin embargo como ya veremos más adelante $c = 0$ restringirá la longitud máxima del periodo.

Definición 6.4.1. Una sucesión $\{x_i\}$ generado por una congruencia definida en 6.4 se llama sucesión lineal por congruencia de números pseudoaleatorios, que está determinada por la cuaterna (x_0, a, c, m). De esta sucesión obtenemos la sucesión normalizada de números pseudoaleatorios $\{u_i\}$ donde $u_i = x_i/m \in [0, 1)$ para todo $i \geq 0$

Ejemplo 6. Tomemos la sucesión $\{x_i\}$ definida mediante la relación $(x_0, a, c, m) =$ $= (7, 3, 5, 15)$. Encontramos que $x_1 \equiv 3 \cdot 7 + 5 \equiv 11 \pmod{15}$, con lo que $x_1 = 11$.

Si reiteramos el proceso se obtiene $x_2 = 8$, $x_3 = 14$, $x_4 = 2$ y así sucesivamente. Con lo que se produce la sucesión $7, 11, 8, 14, 2, 11, 8, 12, 2, \cdots$ que es una sucesión periódica con anteperiodo de longitud 1 y periodo 4

Se ve claramente que $\{x_i\}$ una congruencia lineal contiene como máximo m términos distintos. Si a es relativamente primo con m entonces la sucesión dada en 6.4 puede resolverse únicamente a través de x_i ya que a tiene un único inverso módulo m. Por lo tanto la sucesión es puramente periódica con menor periodo m a lo sumo y además el primer elemento que se va a repetir es el x_0. Sin embargo si $(a, m) \neq 1$, la sucesión $\{x_i\}$ no será periódica pura como se ha visto en el ejemplo anterior. En el caso de que $\{x_i\}$ tenga periodo m diremos que tiene periodo completo.

Teorema 6.4.1. *Un generador por congruencia lineal del tipo 6.4 generará una sucesión de periodo completo si y solamente si se verifican:*

1. *c es relativamente primo con m.*
2. *$a \equiv 1$ mod(p) para cada divisor p de m.*
3. *$a \equiv 1$ mod(4) si a es un múltiplo de 4.*

Demostración. Confrontar con [42]. □

Como consecuencia inmediata de este teorema podemos destacar la importancia del siguiente corolario.

Corolario 6.4.2. *Un generador por congruencia lineal con módulos $m = 2^\beta \geq 4$, $\beta \in \mathbb{Z}^+$ genera una sucesión $\{x_i\}$ de periodo completo m si y solamente si c es impar y además $a \equiv 1$ mod(4).*

Ejemplo 7. Vamos a considerar el generador por congruencia lineal:

$$x_{i+1} \equiv 13x_i + 5 \ (\text{mod}(18)),$$

donde $x_0 = 7$, $a = 13$, $c = 5$, $m = 18$. Tenemos entonces $x_1 \equiv 13 \cdot 7 + 5 \equiv 6$ mod (18). Si repetimos el procedimiento iterando el proceso entonces se obtiene la sucesión:

$$\underbrace{7, 6, 11, 4, 3, 8, 1, 0, 5, 16, 15, 2, 13, 12, 17, 10, 9, 14}_{periodo}, 7, 6, 11, 4, \cdots$$

Después de un profuso número de artículos abordando el tema del valor del incremento c, donde $c \neq 0$ y además es conocido. El hecho de que la elección de c y de x_0 no es de gran importancia, dado que cualquier sucesión por congruencia lineal $\{x_i\}$ se puede obtener mediante una transformación afín de la sucesión fundamental $0, 1, a + 1, a^2 + a + 1, \cdots$

Teorema 6.4.3. *Sea $\{x_i\}$ una sucesión por congruencia lineal. Si $\{y_i\}$ es la sucesión fundamental $0, 1, a + 1, a^2 + a + 1, \cdots$ generada por $y_{i+1} \equiv ay_i + 1$ mod (m), entonces $\{x_i\}$ se puede obtener mediante la transformación $x_i \equiv vy_i + \omega$ mod(m) donde $v \equiv x_0(a - 1) + c$ mod (m) y $\omega \equiv x_0$ mod(m). Lo que es más el periodo de la sucesión $\{x_i\}$ es el periodo de la sucesión $\{y_i\}$ módulo m/k donde $k = (m, x_0(a - 1) + c)$*

6.4.2. Método congruencial puro multiplicativo

El método congruencial puro multiplicativo genera un sucesión congruencial de manera recursiva $\{x_i\}$ definida por:

$$x_{i+1} \equiv ax_i \text{ (mod m) } , 0 \leq x_{i+1} < m \tag{6.5}$$

con $0 < a, x_0 < m$.

Es obvio que una sucesión congruencial multiplicativa $\{x_i\}$ tendrá como posible longitud máxima $m - 1$ si $x_i = 0$ para algún i entonces la sucesión degenera hacia

0.

Antes de determinar las condiciones en el multiplicador para que $\{x_i\}$ tenga el periodo máximo necesitaremos las definiciones y resultados obtenidos con las raíces primitivas módulo m 2.2.2.

Definición 6.4.2. Un entero positivo U tal que $a^U \equiv 1$ mod(m) para cada uno de los enteros a relativamente primos con m se llamará exponente universal del entero m. El menor de todos los exponentes posibles se llamará mínimo exponente universal para m y denotará por $\lambda(m)$.

Teorema 6.4.4. *Para una sucesión congruencial pura multiplicativa x_i, determinada por $(x_0, a, 0, m)$ la longitud máxima del periodo es $\lambda(m)$ si ocurre:*

1. *x_0 es relativamente primo com m.*
2. *a es una raíz primitiva módulo m.*

Teorema 6.4.5. *Una sucesión $\{x_i\}$ obtenida mediante un generador congruencial multiplicativo puro , con generador $m = 2^\beta \geq 16$ alcanza un periodo máximo $m/4$ si y solamente si x_0 es primo y además $a \equiv 3$ ó 5 mod(8).*

Ejemplo 8. Veamos algunos ejemplos basados en una congruencia multiplicativa.

1. Para comenzar vamos a considerar el caso $m = 2^4 = 16$ y la congruencia quedará definida por

$$x_{i+1} \equiv 11x_i \text{ mod}(32) \ \forall i \geq 0$$

   Consideremos $x_0 = 21$ entonces la sucesión que se obtiene es:

$$\underbrace{21, 7, 13, 15, 5, 23, 29, 31}_{periodo}, 21, 7, 13, 15, 5, 23, 29, \cdots$$

   con periodo máximo $8 = 32/4$.

2. De manera parecida si empezamos en $x_0 = 5$ y consideramos la congruencia

$$x_{i+1} \equiv 9x_i \text{ mod } (16).$$

se obtiene

$$\underbrace{5, 13, 21, 29}_{periodo}, 5, 13, 21, 29, \cdots$$

En este caso el periodo de longitud 8 no se alcanza dado que $9 \equiv 1$ mod (8) $\not\equiv 3$ ó 5 mod(18).

6.4.3. El método general de recursión lineal de $k-$ésimo orden

Vamos a tratar de dar una generalización de lo realizado anteriormente. Es un resultado conocido que ilustraremos más adelante que este método puede generar sucesiones con periodos del orden de $m^k - 1$ en comparación con los métodos anteriores donde teníamos $m - 1$.

Sea k un entero positivo. El método de recursión lineal de $k-$ ésimo orden genera una sucesión lineal orden k $\{x_i\}_{i\geq 0}$ de números pseudoaleatorios definidos mediante:

$$x_{i+k} \equiv a_{k-1}x_{i+k-1} + a_{k-2}x_{i+k-2} + \cdots a_0 x_i \text{ mod(m)}, \forall i \geq 0 \tag{6.6}$$

donde los coeficientes $a_0, a_1, \cdots, a_{k-1} \in \mathbb{Z}_m$ y m es un entero positivo. Obviamente el periodo de longitud máxima de x_i será como máximo $m^k - 1$ dado que hay exactamente m^k distintas k uplas de elementos en $\mathbb{Z}_m$ y la nula es excluida. El objetivo que nos atañe será el de encontrar las condiciones bajo las cuales una sucesión del tipo $\{x_i\}_{i\geq 0}$ es de longitud máxima. Necesitaremos para poder trabajar adecuadamente y no encontrarnos divisores de 0 que $m = p^n$ donde p es un primo y así poder garantizar que todos los elementos salvo el neutro para la suma, tienen

inverso. Luego de aquí en adelante supondremos que $m = p^n$ donde p es primo, por lo que $\mathbb{F}_q$ es un cuerpo finito de orden p^n.

6.5. Recurrencia lineal

En el teorema 6.2.1 que acabamos de demostrar es muy útil. Se puede demostrar de la misma manera que siempre existen reordenaciones de la sucesión de partida que generan una secuencia de longitud (periodo) $2^r - 1$.

Nos vamos al concentrar en uno de los valores almacenados en las celdas de la sucesión de estados del registro de desplazamiento, digamos por ejemplo el primero. Supongamos que los valores históricos de los valores almacenados en el primera celda viene dado por la expresión $\{a_n\} = \{a_1, a_2, \cdots, a_n, \cdots\}$. Por cómo se han ido obteniendo los valores del registro de desplazamiento, a_n es la suma módulo 2 de los contenidos de las celdas que se han almacenado en el estado $n-1$. De hecho, este proceso lo podemos iterar de manera sucesiva, con lo que cada uno de los términos de la sucesión será una combinación lineal de los anteriores a partir de un cierto lugar. Con lo que hemos conseguido demostrar el siguiente teorema.

Teorema 6.5.1. *La sucesión $\{a_n\}$ de los valores almacenados en una de las celdas del registro de desplazamiento verifica la ecuación.*

$$a_n = c_1 a_{n-1} \oplus c_2 a_{n-2} \oplus \cdots \oplus c_r a_{n-r} \tag{6.7}$$

donde se verifica que $c_i \in \mathbb{Z}_2$ y no depende del valor de n

Definición 6.5.1. La relación definida en la ecuación 6.7 se conoce como recurrencia lineal y una sucesión $\{a_n\}$ que verifique esta relación se conocerá como sucesión lineal.

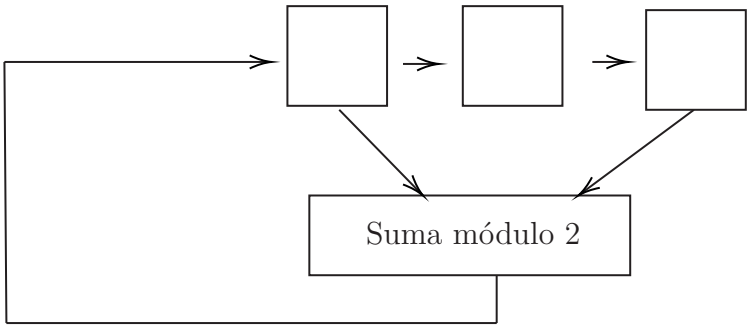

Figura 6.3: Esquema de un registro con desplazamiento lineal

Ejemplo 9. Consideremos la sucesión cuyo término inicial es $\{1,1,1\}$ y cuya recurrencia está definida mediante según nos muestra la figura 6.3.

La sucesión de estados será la siguiente:

$\{1,1,1\},\{0,1,1\},\{1,0,1\},\{0,1,0\},\{0,0,1\},\{1,0,0\},\{1,1,0\},\{1,1,1\}\cdots$

Gráficamente podemos observar esta sucesión de estados y los valores que se han obtenido en la figura 6.3.

El octavo de los elementos de la sucesión de estados es igual al primero, por lo que podemos afirmar que la sucesión de estados tiene periodo $2^3-1=7$. Por lo tanto se obtiene la longitud máxima determinada por la cota del teorema 6.2.1.

Por otro lado la sucesión de los valores almacenados en la primera de las celdas será $\{1,0,1,0,0,1,1\}$ satisface la recurrencia $a_n = a_{n-1} + a_{n-2}$.

Debemos fijarnos en los siguientes hechos que cobrarán especial importancia.

1. El histórico de los valores de la segunda de las celdas será el histórico de los valores de la primera de las celdas, excepto por el retraso de un impulso de reloj. Igualmente lo podemos establecer para las siguientes celdas.
2. Por lo tanto en cada una de las celdas, los valores de las sucesiones asociadas verificarán la misma ley de recurrencia que verifique la primera de ellas. Por lo tanto, podemos suponer que el registro de desplazamiento al completo verificará la relación lineal establecida para una de las sucesiones almacenadas

en una de sus celdas.

6.6. Función generadora

Existen básicamente dos métodos para estudiar las sucesiones recurrentes. Se pueden analizar considerando la sucesión de estados al completo con lo que se recomienda utilizar los métodos matriciales tal y como se ha visto en la sección 6.10 o considerando los valores que se obtienen en las sucesiones de las celdas. Caso de analizar las sucesiones de los valores almacenados en las celdas entonces analizaremos el registro de desplazamiento a través de la función generadora.

Definición 6.6.1. Sea $\{a_n\} = \{a_0, a_1, a_2, \cdots\}$ una sucesión generada a través de los valores almacenados en una de las celdas del registro de desplazamiento. Definimos la función que genera para esta sucesión como:

$$G(x) = \sum_{i=0}^{\infty} a_n x^n \tag{6.8}$$

Definición 6.6.2. Sea $\{C_0, C_1, \cdots\}$ la sucesión de estados asociados a un registro de desplazamiento. Al estado C_0 lo llamaremos semilla o registro inicial de la sucesión de los registros de desplazamiento. Supongamos que $C_0 = \{a_1, a_2, \cdots a_r\}$ entonces diremos que $\{a_1, a_2, \cdots a_r\}$ es la semilla de la sucesión de los valores almacenados en la primera de las celdas.

Supongamos que la sucesión $\{a_n\} = \{a_0, a_1, a_2, \cdots\}$ de desplazamiento tiene una semilla $\{a_{-1}, a_{-2}, \cdots, a_{-r}\}$. Si $\{a_n\}$ verifica la relación de recurrencia:

$$a_n = c_1 a_{n-1} \oplus c_2 a_{n-2} \oplus \cdots \oplus c_r a_{n-r} = \sum_{i=1}^{r} c_i a_{n-1} \tag{6.9}$$

Entonces

$$G(x) = \sum_{n=0}^{\infty} \sum_{i=1}^{r} c_i a_{n-i} x^n = \sum_{i=1}^{r} c_i x^i \sum_{n=0}^{\infty} a_{n-i} x^{n-i} =$$
$$= \sum_{i=1}^{r} c_i x^i \left(a_{-i} x^{-i} + \cdots + a_{-1} x^{-1} + \sum_{n=0}^{\infty} a_n x^n\right)$$

Por lo tanto,

$$G(x) = \sum_{i=1}^{r} c_i x^i \left(a_{-i} x^{-i} + \cdots + a_{-1} x^{-1} + G(x)\right)$$

en consecuencia,

$$G(x) - \sum_{i=1}^{r} c_i x^i G(x) = \sum_{i=1}^{r} c_i x^i \left(a_{-i} x^{-i} + \cdots + a_{-1} x^{-1}\right)$$

En otras palabras se tiene:

$$G(x) = \frac{\sum_{i=1}^{r} c_i x^i \left(a_{-i} x^{-i} + \cdots + a_{-1} x^{-1}\right)}{1 - \sum_{i=1}^{r} c_i x^i} \tag{6.10}$$

Con lo que esta ecuación expresa $G(x)$ en términos de la semilla inicial $\{a_{-1}, a_{-2}, \cdots, a_{-r}\}$, y los coeficientes del registro de desplazamiento $c_1, c_2, \cdots, c_r$. De hecho el denominador de la expresión resultante en 6.10, no depende de las condiciones iniciales.

Si las condiciones iniciales son $a_{-1} = a_{-2} = \cdots = a_{1-r} = 0, a_r = 1$, la expresión de la ecuación 6.10 queda reducida a:

$$G(x) = \frac{c_r}{1 - \sum_{i=1}^{r} c_i x^i} \tag{6.11}$$

Definición 6.6.3. Al polinomio que tenemos definido en el denominador de la ecuación 6.10 se le conoce como polinomio característico de la sucesión $\{a_n\}$.

$$f(x) = 1 - \sum_{i=1}^{r} c_i x^i \tag{6.12}$$

6.7. Periodo de una sucesión recurrente

El teorema 6.2.1 nos muestra que las secuencias generadas por los registros de desplazamiento son periódicas y además nos da una cota superior para la longitud del periodo. La ecuación 6.11 nos da la posibilidad de establecer una descripción más precisa en términos del periodo de la sucesión.

Teorema 6.7.1. *Sea un $A = a_n$ una sucesión generada por un registro de desplazamiento de r estados. Supongamos que la semilla de la sucesión de estados verifica $a_{-1} = a_{-2} = \cdots a_{1-r} = 0, a_{-r} = 1$, entonces el periodo de la sucesión es el menor entero positivo p para el que el polinomio característico $f(x)$ divide a $1-x^p$, módulo 2.*

Demostración. Nosotros sabemos que:

$$G(x) = \frac{1}{f(x)} = \sum_{n=0}^{\infty} a_n x^n \tag{6.13}$$

Veamos ambas implicaciones:

1. Si A tiene periodo p, entonces

$$\begin{aligned}\frac{1}{f(x)} &= \left(a_o + a_1 x + \cdots + a_{p-1}x^{p-1}\right) + x^p\left(a_0 + a_1 x + \cdots a_{p-1}x^{p-1}\right) + \\ &\quad + x^{2p}\left(a_0 + a_1 x + \cdots a_{p-1}x^{p-1}\right) + \cdots = \\ &= \left(a_0 + a_1 x + \cdots a_{p-1}x^{p-1}\right)\left(1 + x^p + x^{2p} + x^{3p} + \cdots\right) = \\ &\quad \left(a_0 + a_1 x + \cdots a_{p-1}x^{p-1}\right) \mid (1 - x^p)\end{aligned}$$

Por lo tanto se tiene que:

$$f(x)\left(a_0 + a_1x + \cdots a_{p-1}x^{p-1}\right) = 1 - x^p \tag{6.14}$$

Y entonces $f(x)$ divide a $(1 - x^p)$

2. Supongamos ahora que $f(x)$ divide a $(1 - x^p)$. Sea $(a_o + a_1x + \cdots + a_{p-1}x^{p-1})$ el cociente. Entonces:

$$\frac{1}{f(x)} = \frac{(a_0 + a_1x + \cdots a_{p-1}x^{p-1})}{1 - x^p} =$$
$$= \left(a_0 + a_1x + \cdots a_{p-1}x^{p-1}\right)\left(1 + x^p + x^{2p} + \cdots\right) =$$
$$= \left(a_0 + a_1x + \cdots a_{p-1}x^{p-1}\right) + x^p\left(a_0 + a_1x + \cdots + a_{p-1}x^{p-1}\right) + \cdots =$$
$$= G(x) = \sum_{n=0}^{\infty} a_n x^n$$

Podemos identificar término a término los coeficientes con las potencias de x, por lo tanto $a_n = a_n$, y en consecuencia A tiene periodo p o algún otro factor de éste, por lo que la sucesión es periódica.

Entonces, p es el entero positivo más pequeño para el que $f(x)$ divide a $(1 - x^p)$ es el periodo de A

□

Definición 6.7.1. En las condiciones del teorema 6.7.1 llamaremos exponente de $f(x)$ al menor entero positivo que verifica que $f(x)$ divide a $(1 - x^p)$

Corolario 6.7.2. *En las condiciones del teorema 6.7.1, supongamos que $G(x) = g(x)/f(x)$ para un cierto polinomio $g(x)$. Supongamos además que $f(x)$ es irreducible, entonces el periodo de la sucesión no depende de la semilla, (salvo el caso en el que la semilla sea nula, ya que la sucesión sería siempre nula).*

6.8. Condición necesaria para obtener una longitud máxima del periodo

Definición 6.8.1. Sea A una secuencia generada por un registro de desplazamiento de r celdas. Diremos que la sucesión A es maximal o de máxima longitud si tiene periodo $p = 2^r - 1$

Definición 6.8.2. Una serie de potencias sobre un anillo R es una expresión de la forma:

$$a(x) = a_0 + a_1 x + a_2 x^2 + \cdots$$

donde x es la indeterminada y $a_0, a_1, \cdots \in R$, como en los polinomios los a_i se llamarán coeficientes. La sucesión $(a_0, a_1, \cdots)$ se llamará sucesión de los coeficientes de la serie de potencias $a(x)$ y se denotará mediante $seq(a)$. Si $b(x) = b_0 + b_1 x + b_2 x^2 + \cdots$ es una serie de potencias sobre R entonces definimos:

$$(a+b)(x) = a(x) + b(x) = \sum_{i=0}^{\infty}(a_i + b_i)x^i$$

y

$$(ab)(x) = a(x) + b(x) = \sum_{i=0}^{\infty}\left[\sum_{j=0}^{i} a_j b_{i-j}\right] x^i$$

El conjunto de todas las series de potencias sobre R se denotará $R[[x]]$. El menor grado de una serie de potencias no nula $a(x) = \sum_{i=0}^{\infty}$ es el menor índice i tal que $a_i \neq 0$. El menor grado de 0 es ∞.

Estas operaciones dotan a $R[[x]]$ de estructura de anillo unitario.

Lema 6.8.1. *Sea $b(x) = \sum_{i=0}^{\infty} b_i x^i \in R[[x]]$ una serie de potencias. Entonces b es inversible* (1) *si y solamente si el término constante b_0 es inversible en R* (2).

Demostración. El término constante es un producto de términos constantes luego (1) $\rightarrow$ (2). Si b_0 es inversible entonces la ecuación $b(x)c(x) = 1$ se puede resolver inductivamente mediante $c(x) = \sum_{i=0}^{\infty} c_i x^i$ donde $c_0 = b_0^{-1}$ y $c_i = -b_0^{-1}(b_1 c_{i-1} + b_2 c_{i-2} + \cdots + b_i c_0)$. □

Definición 6.8.3. El anillo $E \subset R[[x]]$ es la colección de todas las series de potencias $a(x) = \sum_{i=0}^{\infty} a_i x^i$ tal que la sucesión $seq(a) = (a_0, a_1, \cdots)$ es periódica.

Teorema 6.8.2. *Sea $a(x) = \sum_{i=0}^{\infty} a_i x^i$ una serie de potencias sobre un anillo R y sea $n \geq 1$. Los siguientes enunciados son equivalentes.*

1. *La sucesión $seq(a)$ es periódica y n es el periodo de $seq(a)$*
2. *$a(x) = h(x)/(x^n - 1)$ para algún $h(x) \in R[x]$*
3. *$(a) = f(x)/g(x)$ para algunos $f, g \in R[x]$ tal que $g(x)$ es mónico y además $g(x) \mid (x^n - 1)$*
4. *$(a) = f(x)/g(x)$ para algunos $f, g \in R[x]$ tal que $g(x) \mid (x^n - 1)$*

   Además cualquiera de estos enunciados implica
5. *$a(x) = f(x)/g(x)$ para algunos $f, g \in R[x]$ tal que $g(0)$ es inversible en R*

Nótese además que si R fuese un cuerpo entonces $R(x) \subset R((x))$ y ambos serían por lo tanto cuerpos.

Teorema 6.8.3. *Si una sucesión A es maximal entonces su polinomio característico es irreducible. Esto es, si $f(x)$ es el polinomio característico de la sucesión A y existen dos polinomios tales de menor grado que el grado de $f(x)$, $g(x)$ y $h(x)$ tales que $f(x) = g(x) \cdot h(x)$ entonces o bien $g(x) = 1$ o bien $h(x) = 1$.*

Demostración. Dado que A pasa por todos $2^r - 1$ estados antes de que repita, entonces pasará por el que está compuesto por $r-1$ veces el 0 y luego después el 1. En consecuencia las condiciones del teorema 6.7.1 se verifican. Como consecuencia el periodo de A es exponente de $f(x)$.

Por reducción al absurdo, supongamos que $f(x)$ tiene dos factores $s(x)$ y $t(x)$, en consecuencia $f(x) = s(x) \cdot t(x)$ entonces $\frac{1}{f(x)} = \frac{\alpha(x)}{s(x)} + \frac{\beta(x)}{t(x)}$ dado que se puede descomponer como suma de fracciones algebraicas. Supongamos que $s(x)$ y $t(x)$ tienen grados r_1 y r_2 respectivamente tales que $r_1 + r_2 = r$. Por lo tanto $\frac{\alpha(x)}{s(x)}$ es una serie de potencias cuyos coeficientes son periódicos con periodo a lo sumo 2^{r_1-1} y $\frac{\beta(x)}{t(x)}$ es una serie de potencias cuyos coeficientes son periódicos con periodo a lo sumo 2^{r_2-1}. Entonces la suma $\frac{1}{f(x)} = \frac{\alpha(x)}{s(x)} + \frac{\beta(x)}{t(x)}$ será periódica y su coeficientes serán periódicos será a lo sumo el mínimo común múltiplo de (r_1, r_2). Entonces:

$$2^r - 1 \leq (2^{r_1} - 1) \cdot (2^{r_2} - 1) = 2^{r_1+r_2} - 2^{r_1} - 2^{r_2} + 1 \leq$$
$$\leq 2^r - 2 - 2 + 1 = 2^r - 3 \quad (6.15)$$

□

6.9. Matriz Asociada de orden k

Una sucesión recurrente $\{a_i\}_{i\geq 0}$ sobre $\mathbb{F}_q$ tiene asociada una matriz de orden k.

$$A = \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & a_0 \\ 1 & 0 & 0 & \cdots & 0 & a_1 \\ 0 & 1 & 0 & \cdots & 0 & a_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & a_{k-1} \end{pmatrix}$$

donde 0 y 1 son los neutros para la suma y el producto en $\mathbb{F}_q$. Nos debemos dar cuenta que la matriz A depende sólo de la relación de recurrencia mediante la cual se define la secuencia. Si $k = 1$, la matriz A se considera una matriz de orden 1 (a_0).

Si consideramos el vector $i-$ésimo de estados $\mathbf{x}_i = (x_i, x_{i+1}, \cdots, x_{i+k-1})$ entonces se obtiene

$$\mathbf{x}_i\mathbf{A} = (a_i, x_{i+1}, \cdots, a_{i+k-1}) \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & a_0 \\ 1 & 0 & 0 & \cdots & 0 & a_1 \\ 0 & 1 & 0 & \cdots & 0 & a_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & a_{k-1} \end{pmatrix} = \mathbf{x}_{i+1}$$

Si consideramos el conjunto de todas las matrices no singulares de orden $k \times k$ con elementos en $\mathbb{F}_q$, este forma un grupo finito de matrices bajo la multiplicación llamado grupo lineal general y se denotará $GL(k, \mathbb{F}_q)$ o alternativamente $GL(k, q)$

Teorema 6.9.1. *Si $\{a_i\}$ es una secuencia lineal recurrente de orden r sobre $\mathbb{F}_q$, donde $a_0 \neq 0$ y generada como en 6.6, entonces la longitud del menor periodo de la secuencia divide al orden de la matriz $\boldsymbol{A} \in GL(k, q)$*

Definición 6.9.1. Un impulso de respuesta de una secuencia sobre $\mathbb{F}_q$ es una sucesión lineal recurrente $\{x_i\}$ sobre $\mathbb{F}_q$ que está únicamente determinada por su vector de estado inicial $\mathbf{x}_0 = (0, 0, \cdots, 0, 1)$. Si $k = 1$ entonces $x_0 = 1$

Ejemplo 10. Consideremos una sucesión recurrente lineal de orden 4, $\{x_i\}_{i\geq 0}$ sobre $\mathbb{F}_3 \cong \mathbb{Z}_3 \cong \{-1, 0, 1\}$ definida como $x_{i+4} = x_{i+3} - x_{i+2} + x_i$ donde su matriz asociada será:

$$A = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

Es una matriz de orden 24 en $GL(4, 3)$. Si tomamos el vector inical $\mathbf{x}_0 = (-1, 0, 0, 1)$, la secuencia generada será

$$\underbrace{-1,0,0,1,0,-1,1,1}_{periodo},-1,0,0,1,0\cdots$$

con periodo mínimo 8 que divide a 24.

Si hubiéramos considerado la sucesión de del impulso de respuesta correspondiente se obtendría:

$$\underbrace{0,0,0,1,1,0,-1,0,-1,-1,-1,0,0,-1,1,-1,1,1,1,-1,-1,0,1}_{periodo},0,0,0,1\cdots$$

con periodo mínimo 24. O lo que es lo mismo el orden de $\mathbf{A} \in GL(4,3)$

Lema 6.9.2. *Supongamos que $\{x_i\}$ es una secuencia k impulso respuesta sobre $\mathbb{F}_k$ con matriz asociada $\boldsymbol{A}$. Entonces dos vectores de estados $\boldsymbol{x}_m = \boldsymbol{x}_n$ si y solamente si $\boldsymbol{A}^m = \boldsymbol{A}^n$*

Teorema 6.9.3. *Supongamos que $\{x_i\}$ es una secuencia k impulso respuesta sobre $\mathbb{F}_k$ generada como en 6.6 donde $a_0 \neq 0$. Entonces el mínimo periodo de la secuencia es igual al orden de la matriz $\boldsymbol{A} \in GL(k,q)$*

6.10. Registros de desplazamiento sobre cuerpos extendidos

De la importancia en una gran variedad de aplicaciones son las secuencias definidas sobre cuerpos finitos. Desde un punto de vista computacional el hecho de que estas sucesiones se puedan definir de manera recursiva es una gran ventaja. Son de gran relevancia las aplicaciones a la criptografía, ingeniería y teoría de códigos. En esta sección vamos a tratar de generalizar los resultados que se obtienen en $p = 2$, por lo que trabajaremos sobre cuerpos finitos de orden $q = p^n$ elementos. Como ya hemos mencionado, siempre existe un cuerpo con $q = p^n$ elementos y es también

conocido que dado dos cuerpos con $q = p^n$ elementos, estos cuerpos son isomorfos. Recordemos un poco la definición de Cuerpo de Galois de orden q.

Definición 6.10.1. El cuerpo finito de $q = p^n$ elementos, donde p es primo y n es un entero positivo se llama cuerpo de Galois de orden q y se denotará mediante $GF(q)$.

En lo que sigue denotaremos los cuerpos mediante $\mathbb{F}$ y denotaremos por $\mathbb{F}_q$ a los cuerpos de q elementos. Denotaremos mediante $\mathbb{F}_q^* = \mathbb{F} \setminus \{0\}$ y realizaremos la identificación de $\mathbb{F}_p$ con $\mathbb{Z}_p$ por lo que podremos asumir que $\mathbb{F}_p \cong \{0, 1, 2, \cdots p-1\}$.

Consecuentemente la definición dada en 6.6 quedará como sigue:

$$x_{i+k} \equiv a_{k-1}x_{i+k-1} + a_{k-2}x_{i+k-2} + \cdots a_0 x_i \text{ mod(m)}, \forall i \geq 0 \tag{6.16}$$

donde cada $x_i, a_j \in \mathbb{F}_q$. Tal generador generará un sucesión $\{x_i\}$ sobre $\mathbb{F}_q$.

Nos debemos dar cuenta de que los términos $x_0, x_1, \cdots x_{k-1}$ determinan de manera única el resto de valores de la secuencia $\{x_i\}_{i\geq 0}$. A estos valores los llamaremos valores iniciales y a la k-upla $(x_0, x_1, \cdots, x_{k-1})$ lo llamaremos vector inicial de la secuencia.

6.10.1. Propiedades de periodicidad

Teorema 6.10.1. *Cada sucesión recurrente lineal de orden k, $\{x_i\}_{i\geq 0}$ sobre $\mathbb{F}_q$ es periódica con periodo mínimo d. Además este periodo verifica que $d \leq q^k - 1$*

Por lo tanto hemos obtenido una cota del periodo.

Definición 6.10.2. Si una secuencia llega a tener periodo mínimo, $q^k - 1$, entonces diremos que la secuencia es maximal, también es frecuente encontrar en la bibliografía el término $m-$ secuencia.

Un elemento $\alpha \in \mathbb{F}_q^*$ se dice que es un elemento primitivo de $\mathbb{F}_q$ si es un generador de $\mathbb{F}_q^*$, con lo que $\mathbb{F}_q^* = \{1, \alpha, \alpha^2, \cdots \alpha^{q-1}\}$.

Ejemplo 11. Sea α un elemento primitivo de $\mathbb{F}_q$ consideremos la secuencia de primer orden lineal y recurrente $\{x_i\}$ sobre $\mathbb{F}_q$ que satisface $x_i \equiv \alpha x_i$, donde $x_0 \neq 0$.Entonces obtenemos la secuencia:

$$x_0, \alpha x_0, \alpha^2 x_0, \cdots, \alpha^{q-1} x_0, x_0 \cdots$$

con periodo mínimo $q-1$, lo que nos dice que la cota superior se puede alcanzar.

Teorema 6.10.2. *Si $\{x_i\}_{i\geq 0}$ es una secuencia lineal recurrente de orden k sobre $\mathbb{F}_q$ generada mediante la expresión 6.6 con $a_0 \neq 0$ es una secuencia periódica pura.*

6.10.2. Polinomio Característico y matriz de compañía

Definición 6.10.3. Una sucesión recurrente y lineal de orden k, $\{x_i\}$ sobre $\mathbb{F}_q$ tiene como polinomio característico $f(x) = x^k - a_{k-1}x^{k-1} - \cdots - a_1 x - a_0 \in \mathbb{F}_q[x]$. El polinomio característico $f^* = 1 - a_{k-1}x - a_{k-2}x^2 - \cdots - a_1 x^{k-1} - a_0 x^k \in \mathbb{F}_q[x]$ se llama polinomio característico recíproco de $\{x_i\}$ y se obtiene gracias a $f(x)$ a través de la igualdad $f^* = x^k f(1/x)$.

Ejemplo 12. Evidentemente el polinomio característico del ejemplo 10 es $f(x) = x^4 - x^3 + x^2 - 1 \in \mathbb{F}_3[x]$

Definición 6.10.4. La matriz de compañía de un polinomio mónico $g(x) = c_0 + c_1 x + \cdots + c_{k-1}x^{k-1} + x^k \in \mathbb{F}_q$ de grado $k \geq 1$ es una matriz de orden k, $\mathbf{M} \in GL(k, q)$ con polinomio característico $\psi(\lambda) = |\lambda \mathbf{I} - \mathbf{M}| = c_0 + c_1\lambda + \cdots c_{k-1}\lambda^{k-1} + \lambda^k = g(\lambda)$

Se puede verificar que la matriz de A asociada a la sucesión lineal recurrente $\{x_i\}$ dada por

$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & a_0 \\ 1 & 0 & 0 & \cdots & 0 & a_1 \\ 0 & 1 & 0 & \cdots & 0 & a_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & a_{k-1} \end{pmatrix}$$

tiene polinomio característico $\psi(\lambda) = |\lambda\mathbf{I} - \mathbf{M}| = c_0 + c_1\lambda + \cdots c_{k-1}\lambda^{k-1} + \lambda^k$, luego entonces $\psi(\lambda) = f(\lambda)$ donde $f(x) \in \mathbb{F}_q$ es el polinomio característico de $\{x_i\}$. En consecuencia matriz asociada y matriz de compañía vienen a representar el mismo concepto.

Teorema 6.10.3. *Supongamos que $\{x_i\}$ es una secuencia lineal recurrente de orden k, con polinomio característico $f \in \mathbb{F}_q[x]$ tal que $f(0) \neq 0$. Entonces el periodo de menor longitud de la secuencia divide al orden de f, y la menor longitud de la correspondiente sucesión del impulso de respuesta es igual al $ord(f)$. Lo que es más, ambas secuencias son periódicas puras.*

Ejemplo 13. Como hemos visto en el ejemplo 10 el polinomio característico es $f(x) = x^4 - x^3 + x^2 - 1 = (x^2 + x + 1)(x - 1)^2 \in \mathbb{F}_3[x]$. Se puede comprobar fácilmente que $f(x)$ divide a $x^{24} - 1$ y que ningún otro polinomio $x^e - 1$ con $0 < e < 24$ lo puede dividir. Con lo que el orden de $f(x)$ es 24. Este además coincide con el orden de la matriz asociada $A \in GL(4, 3)$. Con un vector inicial $(-1, 0, 0, 1)$ nosotros vimos que la secuencia obtenida es periódica pura de longitud 8, y se comprueba que divide al orden de $f(x)$. También se vio que la sucesión del impulso respuesta era periódica pura de orden 24, como se esperaba.

Teorema 6.10.4. *Supongamos que $\{x_i\}$ es una secuencia lineal recurrente periódica pura con periodo mínimo de longitud d y polinomio característico $f(x) \in \mathbb{F}_q[x]$.*

Entonces la identidad de polinomios:

$$f(x)g(x) = (1 - x^d)h(x) \tag{6.17}$$

se verifica donde

$$g(x) = x_0x^{d-1} + x_1x^{d-2} + \cdots x_{d-2}x + x_{d-1} \in \mathbb{F}_q[x],$$
$$h(x) = \sum_{j=0}^{k-1} \left[\sum_{i=0}^{k-1-j} a_{i+j+1}x_ix^j\right] \in \mathbb{F}_q[x]$$

donde además $a_k = -1$

Teorema 6.10.5. *Supongamos que* $\{x_i\}$ *es una secuencia lineal recurrente sobre* $\mathbb{F}_q$*, con un vector inicial de estados no nulo, y con polinomio característico* $f(x) \in \mathbb{F}_q[x]$ *irreducible y tal que* $f(0) \neq 0$*. Entonces la secuencia es periódica pura con periodo de longitud mínima* $ord(f)$*.*

Ejemplo 14. Consideremos la secuencia lineal recurrente de cuarto orden $\{x_i\}$ sobre $\mathbb{Z}_3 \cong \{-1, 0, 1\}$ definida mediante $x_{i+4} = x_{i+2} + x_i$

La matriz asociada será por tanto

$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \in GL(4, 3)$$

Tiene orden 16, mientras que si consideramos el polinomio característico $f(x) = x^4 - x^2 - 1 \in \mathbb{F}_3[x]$ es irreducible en $\mathbb{F}_3$ y además su orden es 16. Si tomamos el valor $\mathbf{x}_0 = (1, 0, 1, 1)$ entonces obtenemos una sucesión periódica pura de periodo mínimo 16.

$$\underbrace{1, 0, 1, 1, -1, 1, 0, -1, -1, 0, -1, -1, 1, -1, 0, 1}_{periodo}, 1, 0, 1, 1, -1, \cdots$$

6.11. Criterio para máxima longitud de Periodo

Finalmente en este capítulo vamos a demostrar el más importante de todos los resultados de esta parte del documento. Introduciremos una condición suficiente aunque no necesaria para que una secuencia lineal recurrente de orden k, $\{x_i\}$ sobre $\mathbb{F}_q$ sea de periodo máximo.

Definición 6.11.1. Un polinomio $f(x) \in \mathbb{F}_q[x]$ de grado k se dirá que es primitivo si $f(x)$ es el polinomio mínimo sobre $\mathbb{F}_q$ de un elemento primitivo de $\mathbb{F}_{q^k}$

Gracias al siguiente teorema podremos dar una definición equivalente de polinomio primitivo que es más manipulativa.

Teorema 6.11.1. *Un polinomio $f(x) \in \mathbb{F}_q[x]$ de grado $k \geq 1$ se dirá que es un polinomio primitivo sobre $\mathbb{F}_q$ si y solamente si $f(x)$ es mónico, $f(0) \neq 0$ y $ord(f) = q^k - 1$*

Teorema 6.11.2. *Sea $\{x_i\}$ es una secuencia lineal recurrente sobre $\mathbb{F}_q$ de orden k con un vector inicial no nulo. Si $\{x_i\}$ tiene polinomio característico $f(x) \in \mathbb{F}_q[x]$ y además $f(0) \neq 0$, entonces $\{x_i\}$ es periódica pura y alcanza su periodo de longitud máxima $q^k - 1$*

Demostración. Sea $\{x_i\}$ es una secuencia lineal recurrente sobre $\mathbb{F}_q$ de orden k con un vector inicial no nulo. Supongamos que la secuencia $\{x_i\}$ tiene polinomio característico f, el cual es un polinomio primitivo. Entonces por propia definición f es irreducible, luego en consecuencia el orden será $q^k - 1$ luego por el teorema 6.10.5 se tiene que $\{x_i\}$ es periódica pura y además su menor longitud de periodo será $q^k - 1$. □

Corolario 6.11.3. *Sea $\{x_i\}$ una sucesión de impulso respuesta de orden k. Entonces $\{x_i\}$ es periódica pura y alcanza la longitud máxima $q^k - 1$ si y solamente*

si su polinomio característico $f \in \mathbb{F}_q[x]$ verifica las dos siguientes condiciones:

1. $f(0) \neq 0$
2. *f es un polinomio primitivo sobre $\mathbb{F}_q$*

Ejemplo 15. Consideremos $\{x_i\}$ una secuencia lineal recurrente de orden 3 sobre $\mathbb{F}_3 \cong \mathbb{Z}_3 \cong \{-1, 0, 1\}$ definida mediante $x_{i+3} = x_{i+1} - x_i$
Si tomamos $\mathbf{x}_0 = (1, 1, 1)$ como vector inicial se obtendrá una secuencia periódica pura:

$$\underbrace{1, 1, 1, 0, 0, -1, 0, -1, 1, -1, -1, 1, 0, -1, -1, 0, 0, 1, 0, 1, -1, 1, 1, -1, 0}_{periodo}, 1, 1, 1, 0, \cdots$$

donde el periodo máximo es de $26 = 3^3 - 1$. De hecho, el polinomio característico $f(x) = x^3 - x + 1$ es primitivo sobre $\mathbb{F}_3$. Lo que es más si consideramos el impulso respuesta de la secuencia se obtiene:

$$\underbrace{0, 0, 1, 0, 1, -1, 1, 1, -1, 0, 1, 1, 1, 0, 0, -1, 0, -1, 1, -1, -1, 1, 0, -1, -1, -1}_{periodo}, 0, 0, 1, 0, 1, \cdots$$

con un menor periodo de 26 elementos. Debemos destacar que la matriz de 3×3 dada por

$$\mathbf{A} = \begin{pmatrix} 0 & 0 & -1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \in GL(3, 3)$$

es una matriz de orden 26 con lo que se demuestra que el polinomio $f(x)$ es un polinomio primitivo.

6.12. Propiedades aleatorias de las secuencias generadas por registros de desplazamiento

En esta sección vamos a demostrar que las secuencias generadas sobre registros de desplazamiento verifican los tres postulados de Golomb 5.5.1.

6.12.1. Postulado número 1

Teorema 6.12.1. *El postulado número 1 de Golomb se verifica para cualquier sucesión generada por un registro de desplazamiento maximal.*

Demostración. Supongamos que un registro de desplazamiento de r celdas recorre todos los $2^r - 1$ posibles estados de un registro de desplazamiento antes de que se produzca la repetición en la secuencia. Supongamos la sucesión representada de manera binaria. Para cada uno de los estados en el registro de desplazamiento podemos encontrar la expresión decimal correspondiente. Esto es, si el estado es $X = \{x_1, x_2, \cdots, x_r\}$ entonces podemos encontrar una expresión decimal de la siguiente manera:

$$D(X) = \sum_{i=1}^{r} x_i \cdot 2^{i-1} \tag{6.18}$$

Es obvio que si la sucesión recorre todos los estados entonces se verifica:

$$1 \leq D(X) \leq 2^r - 1 \tag{6.19}$$

y además $D(X)$ recorrerá todos los enteros intermedios. Por lo tanto, dado que la mitad de los números naturales son pares y la otra mitad son impares, entonces la mitad de los valores de la sucesión binaria serán pares y la otra mitad serán

impares. □

6.12.2. Postulado número 2

Teorema 6.12.2. *El postulado número 2 de Golomb se verifica para cualquier sucesión generada por un registro de desplazamiento maximal.*

Demostración. Sea $\{a_n\}_{n\in\mathbb{N}}$ una sucesión maximal generada por un registro de desplazamiento de r celdas. Existen 2^r-1 maneras de escoger r términos consecutivos. O lo que es lo mismo, Cada vector de longitud r en el que cada coordenada es o bien 0 o bien 1, ocurre una sola vez.

En particular la racha de r veces el 1 ocurre una sola vez. Esta racha de r veces el número 1 debe estar seguida de un 0 y precedida de un 0 o de lo contrario tendríamos otra racha de r veces el 1, cosa que no podría darse.

Un 0 es seguido de $r-1$ veces el 1 ocurre una sola vez. Pero este vector ya está contado cuando tuvimos en cuenta la racha de r veces el número 1. De manera similar $r-1$ veces el 1 seguido de un 0 ocurre una sola vez que ya ha sido tenido en cuenta una vez analizada la racha de r veces el número 1.

Supongamos que el resultado es cierto para $0 < k < r-1$. Para encontrar una racha de 1 de longitud k, consideramos r consecutivos términos que comiencen con un 0, después k veces el 1 y al final un 0, y el resto de $\{r-k-2\}$ términos de manera arbitraria. Esto ocurre de 2^{r-k-2} maneras distintas dado que cada manera de completar r términos ocurre una sola vez.

De manera análoga podríamos razonar para las rachas de k veces el 0, donde $0 < k < r-1$. No hay ninguna racha de r veces el número 0 ya que el registro de desplazamiento se pararía. Sin embargo, podemos encontrar una racha de de $r-1$ veces el 0 precedida por un 1. Este valor debe ocurrir una vez por lo que tenemos

$+$	1	0
1	0	1
0	1	0

$\times$	-1	1
-1	1	1
1	-1	1

Figura 6.4: Operaciones de suma y multiplicación en $\mathbb{Z}_2$

una sola racha de $r-1$ veces el 0. □

6.12.3. Postulado número 3

Teorema 6.12.3. *El postulado número 3 de Golomb se verifica para cualquier sucesión generada por un registro de desplazamiento maximal.*

Demostración. Sea $\{a_n\}$ una sucesión maximal generada por un registro de desplazamiento. Sea $\{b_n\}$ la sucesión definida como $b_n = 1 - 2a_n$. O lo que es lo mismo, los ceros se reemplazan por el 1, y los unos por el -1. Sean $A_0 = (0, 0, \cdots)$, $A_1 = (a_1, a_2, a_3, \cdots)$, $A_2 = (a_2, a_3, a_4, \cdots)$, $\cdots$, $A_p = (a_p, a_{p+1}, a_{p+2}, \cdots)$. Nótese que cada $\{A_i\}_{i\in\mathbb{N}}$ es una sucesión maximal. Además $(A_i, \oplus)$ tiene estructura de grupo Abeliano con las operaciones definidas en la figura 6.4.

Supongamos que en cada A_i reemplazamos los ceros se reemplazan por el 1, y los unos por el -1. Llamemos $B_0, B_1, B_2, \cdots, B_p$ a las sucesiones resultantes.

Dado que $A_i + A_j = A_k$, entonces se tiene que $B_iB_j = B_k$, donde el producto y la suma se realizan término a término. En el caso en el que $i = j$ entonces $B_iB_j = B_k = \{1, 1, \cdots\}$, por otro lado si $i \neq j$ entonces B_iB_j contendrá $\frac{(p-1)}{2}$ veces el 1 y $\frac{(p-1)}{2}$ veces el el -1. la función de autocorrelación será:

$$C(\tau) = \frac{1}{p}\sum_{n=1}^{p} b_n b_{n+\tau} = \begin{cases} 1 & \text{si} \quad \tau = 0 \\ \\ \frac{-1}{p} & \text{si} \quad 0 < \tau < p \end{cases} \tag{6.20}$$

dado que $\{b_n b_{n+\tau}\}$ es una sucesión del tipo $B_i \cdot B_j$ que se convierte en una sucesión del tipo B_k y el exceso de $+1$ con respecto al -1 es p para B_0 y -1 para cualquier otro caso.

□

Parte IV

Resultados

Capítulo 7

Identificación de polinomios primitivos sobre cuerpos extendidos

7.1. Introducción

Los registros de desplazamiento realimentados linealmente (LFSR) constituyen la base de un gran número de sistemas criptográficos simétricos debido a su simplicidad, a la longitud y aleatoriedad de las secuencias que generan y a la posibilidad de predecirlas mediante modelos lineales [3; 12; 45]. A pesar de la eficiencia de su funcionamiento, en los últimos años se ha propuesto sustituir la utilización de los LFSR tradicionales, esto es, definidos sobre $GF(2)$ por LFSR definidos sobre cuerpos extendidos, con el fin de aumentar la velocidad a la que se genera la secuencia cifrante, pasando de 1 bit a una palabra de n bits en cada generación. Un ejemplo de ello se puede encontrar en el algoritmo de cifrado simétrico empleado en las redes móviles LTE [7], donde tales secuencias se generan de 32 en 32 bits. Este

cambio, propiciado por la evolución en la longitud de palabra de los procesadores, se refleja en la utilización de LFSRs definidos en $GF(2^n)$, siendo n la longitud de palabra del procesador.

Como ocurre con los LFSR en $GF(2)$, para garantizar que las secuencias generadas satisfacen unos requisitos mínimos de seguridad, aleatoriedad y complejidad, es necesario, o al menos recomendable, que la realimentación aplicada esté caracterizada por un polinomio primitivo, lo que asegura una longitud de secuencia máxima. Si bien la búsqueda de polinomios primitivos en $GF(2)$ no es un problema, la búsqueda en un cuerpo extendido $GF(2^n)$ no es sencilla y requiere de algoritmos eficientes. La solución a este problema se ha planteado, como suele ser habitual desde dos perspectivas. Por una parte, se han propuesto métodos de construcción o generación de polinomios primitivos en $GF(p^n)$ a partir de otros polinomios, como el publicado por Cardell y Climent [11] en 2017, que utilizan polinomios de grado b y r definidos sobre $\mathbb{F}_q$ y $\mathbb{F}_{q^b}$, respectivamente. Por otra parte, se han desarrollado tests que permiten únicamente comprobar si un determinado polinomio es o no primitivo. En esta última aproximación es en la que se centra este trabajo, que tiene como antecendentes las propuestas de Wang, Zheng and Li [89] y J.A. Gordon [36] que basan su desarrollo en la generación de la secuencia en su totalidad, lo cual resulta en una implementación bastante lenta; y las más recientes de Lenstra y Hendrik W [51] y Ahmadi y Menezes [1] que utilizan métodos basados en el producto matricial.

La propuesta que se presenta en este trabajo está basada en el tratamiento de las secuencias generadas por los LFSRs en $GF(2^n)$ como secuencias binarias entrelazadas, lo que permite realizar cálculos mucho más sencillos en $GF(2)$, reduciendo así el tiempo de computación.

7.2. Modelo equivalente de secuencias entrelazadas

En [35] y en [46], tanto John J . Komo y Maurice como S. Lam Gong presentan las secuencias entrelazadas mediante la siguiente definición.

Definición 7.2.1. Sea A una secuencia sobre $GF(q^n)$, sea $d \in \mathbb{Z}^+$, se define la secuencia diezmada A sobre d y la representaremos A^d como, $A^d(t) = A(td)\ \forall t \geq 0$.

Definición 7.2.2. Sea $f(x)$ un polinomio sobre $GF(q)$ de grado n con $f(0) = 0$, y sea l un entero positivo. Para cualquier secuencia $u = (u(0), u(1), u(2), ...) = (u(t))$ sobre $GF(q)$, si las secuencias diezmadas $u_j = (u(tl + j)), t \geq 0$ son generadas por $f(x)$ para todo j, entonces u se denomina secuencia entrelazada sobre $GF(q)$ de tamaño l asociada a $f(x)$.

Al mismo tiempo, en [35], se presenta el siguiente lema.

Lema 7.2.1. *Sea u una secuencia entrelazada definida sobre $GF(q)$ de tamaño l asociada a $f(x)$, y sea $h(x)$ el polinomio minimal de u sobre $GF(q)$. Entonces, se cumple que*

1. $h(x)|f(x^l)$
2. $T(u)|per(f)^l$, *donde* $T(u)$ *es el período de la secuencia* u *y* $per(f)$ *el exponente del polinomio* f.

Por otra parte, en [46] se establece la relación entre las m-secuencias generadas sobre $GF(q^m)$ y las m-secuencias generadas sobre $GF(q)$, de tal modo que éstas últimas se pueden obtener como secuencias diezmadas de la primera, o recíprocamente, la primera es una secuencia entrelazada compuesta de m-secuencias sobre $GF(q)$. Esta relación, es la que permite analizar la secuencia en el cuerpo extendido a partir de sus componentes diezmadas en el cuerpo base.

7.3. Identificación de polinomios primitivos

Consideremos un polinomio $p(x)$ de grado m sobre $GF(2^n)$, del que se quiere comprobar si es primitivo y un polinomio $U(x)$ de grado n con coeficientes binarios, utilizado para construir $GF(2^n)$. Los siguientes teoremas proporcionan la base del método que se describe a continuación.

Teorema 7.3.1. *Dadas la sucesión generada por una máquina de estados finita* $\{A_{n,m}\}_{n,m\in\mathbb{N}} = \{\{a_{1,1}, a_{1,2}, \cdots, a_{1,n}\}, \{a_{2,1}, a_{2,2}, \cdots, a_{2,n}\}, \cdots\}$ *generada por el polinomio* $f \in \mathbb{F}_{q^n}[x]$. *Sea* $a^1 = \{a_{1,1}, a_{2,1}, \cdots\}$, $a^2 = \{a_{1,2}, a_{2,2}, \cdots\}$, $\cdots$, $a^q = \{a_{1,q}, a_{2,q}, \cdots\}$, *el conjunto de secuencias diezmadas, todas sobre* $GF(2)$. *En estas condiciones el polinomio* f *es primitivo si y solamente si las secuencias diezmadas* $\{a^1, a^2, \cdots, a^q\}$ *son secuencias maximales sobre* $GF(2)$ *y están todas generadas por el mismo polinomio. Lo que es más, en estas condiciones si el grado de* f *es* m *entonces este único polinomio que genera las secuencias* $\{a^1, a^2, \cdots, a^q\}$ *tiene grado* $m \cdot n$ *y las secuencias generadas son maximales y tienen complejidad lineal* $m \cdot n$

Demostración. Si aplicamos el **Teorema 1** dado en [35] podemos afirmar que dado un polinomio primitivo de grado $m \cdot n$ sobre $GF(q)$ donde α es una raíz que genera una secuencia maximal c y donde f_1 es un polinomio primitivo que divide a $g(x)$ sobre $GF(q^n)$ y que genera una secuencia maximal d. Por otro lado sea $h(x)$ un polinomio de grado m sobre $GF(q)$ con raíz γ donde $\gamma = \alpha^{z\omega}$ y pertenece a la base $\{1, \gamma, \gamma^2, \cdots, \gamma^{m-1}\}$ para expresar $GF(q^m)$ sobre $GF(q)$, dado $z = \left(\frac{q^{mn}-1}{q^m-1}\right)$ y donde ω y $q^m - 1$ son relativamente primos, entonces se verifica que para algún e_0

$$d = \sum_{j=0}^{m-1} \gamma^j T^{e_0 + z k_j \omega_c} \tag{7.1}$$

donde T^{i_C} indica un desplazamiento a la izquierda de la secuencia c de i ele-

mentos.

Del mismo modo como consecuencia inmediata el **teorema 1** dado en [35] podemos utilizar los **corolario 1 y 2** dado en el mismo artículo [35], todos los elementos de los registros de desplazamiento de las secuencias maximales de longitd N sobre $GF(q^m)$ se pueden obtener mediante el diezmado de esta secuencia maximal, dada una base de polinomios de $GF(q^m)$ sobre $GF(q)$. □

Teorema 7.3.2. *En las mismas condiciones del* ***teorema 7.3.1****, supongamos que el polinomio obtenido es* $a^1(x) = a_0^1 + a_1^1 x + a_2^1 x^2 + \cdots a_{m\cdot n}^1 x^{m\cdot n}$*. Consideremos la secuencia plana generada como la concatenación de las secuencias* $\{A_{n,m}\}_{n,m\in\mathbb{N}} = \{\{a_{1,1}, a_{1,2}, \cdots, a_{1,n}\}, \{a_{2,1}, a_{2,2}, \cdots, a_{2,n}\}, \cdots\}$*. Entonces esta secuencia plana tienen complejidad lineal* $m\cdot n^2$ *y está generada por el polinomio* $A(X) = a_0^1 + a_1^1\ x^{1\cdot n} + a_2^1\ x^{2\cdot n} + \cdots a_{m\cdot n^2}^1\ x^{m\cdot n^2}$.

Demostración. John J . Komo and Maurice S. Lam establecen en [46] establecen la relación existente entre las secuencias maximales sobre $GF(q^n)$ en función de las secuencias maximales de los polinomios sobre $GF(q)$ □

Vamos a ilustrar de manera resumida cómo se efectúa el método. Lo primero que realizaremos será generar $2\cdot m\cdot n+1$ elementos utilizando el polinomio que se desea analizar. Una vez obtenidos estos elementos pararemos al modelo equivalente de secuencias entrelazadas diezmando adecuadamente la secuencia. Posteriormente se analizan las secuencias entrelazadas y se les aplicará el algoritmo de Berlekamp-Massey. Finalmente se analizará si estas cumplen las condiciones pedidas en los **teoremas 7.3.1 y 7.3.2**.

Ilustremos el modelo paso a paso.

Test de primitividad. Input: polinomio $p(x)$ de grado m sobre $GF(2^n)$. Output: Primitivo/No primitivo.

- *Paso 1.* Se construye un LFSR sobre $GF(2^n)$ con m celdas, utilizando $p(x)$ como realimentación y una semilla $A_0 = \{\left(a_0^1, a_1^1, \cdots, a_{n-1}^1\right), \cdots, \left(a_0^m, \cdots, a_{n-1}^m\right)\}$, donde no todos los elementos son nulos. Se ha de tener en cuenta que no se pierde generalidad, por lo que el método es independiente de la semilla escogida.
- *Paso 2.* Se generan $2 \cdot m \cdot n + 1$ elementos de la secuencia, lo que permite disponer de $2mn^2$ bits.
- *Paso 3.* Se obtienen las secuencias entrelazadas $\{B^i\}$.
- *Paso 4.* Se analizan las secuencias binarias entrelezadas para calcular su complejidad lineal y obtener el polinomio de realimentación $\{p_1, p_2, \cdots, p_n\}$ con coeficientes binarios de cada una de ellas, aplicando, por ejemplo, el algoritmo de Berlekamp-Massey.
- *Paso 5.* El polinomio p será primitivo si y solamente si $p_i = p_j$, $\forall i, j$ y además el grado de p_i es $m \cdot n$, $\forall i, j$.

A continuación, se presenta un ejemplo de la ejecución del método aplicado a un polinomio $p(x)$ de grado 6 con coeficientes en $GF(2^4)$, esto es, $p(x) = c_0 + c_1 x + c_2 x^2 + \cdots + c_6 x^6$, con $c_i \in GF(2^4)$. En este caso, el cuerpo queda definido por el polinomio $U(x) = 1 + x + x^4$ sobre $GF(2)$. Utilizando la representación en m-uplas de los elementos del cuerpo, podemos considerar como ejemplo los siguientes coeficientes de $p(x)$:

$$
\begin{aligned}
c_0 &= \{0,1,0,1\} \\
c_1 &= \{1,1,0,1\} \\
c_2 &= \{0,0,1,1\} \\
c_3 &= \{1,1,0,1\} \\
c_4 &= \{1,0,0,1\} \\
c_5 &= \{0,0,0,1\}
\end{aligned}
\tag{7.2}
$$

o de manera equivalente,

$$
\begin{aligned}
c_0 &= 1+\alpha \\
c_1 &= 1+\alpha+\alpha^3 \\
c_2 &= \alpha^2+\alpha^3 \\
c_3 &= 1+\alpha+\alpha^3 \\
c_4 &= 1+\alpha^3 \\
c_5 &= \alpha^3
\end{aligned}
\tag{7.3}
$$

donde α es una raiz del polinomio $U(x)$ que define el cuerpo. En consecuencia, se tiene que $\alpha^4 = \alpha + 1$, quedando todos los elementos del cuerpo definidos del siguiente modo.

$$
\begin{aligned}
&1\\
&\alpha\\
&\alpha^2\\
&\alpha^4 = \alpha + 1\\
&\alpha^5 = \alpha^2 + \alpha\\
&\alpha^6 = \alpha^3 + \alpha^2\\
&\alpha^7 = \alpha^4 + \alpha^3 = \alpha^3 + \alpha + 1\\
&\alpha^8 = \alpha^4 + \alpha^2\\
&\alpha^9 = \alpha^3 + \alpha\\
&\alpha^{10} = \alpha^4 + \alpha^2 = \alpha^2 + \alpha + 1\\
&\alpha^{11} = \alpha^3 + \alpha^2 + \alpha\\
&\alpha^{12} = \alpha^4 + \alpha^3 + \alpha^2 = \alpha^3 + \alpha^2 + \alpha + 1\\
&\alpha^{13} = \alpha^4 + \alpha^3 + \alpha^2 + \alpha = \alpha^3 + \alpha^2 + 1\\
&\alpha^{14} = \alpha^4 + \alpha^3 + \alpha = \alpha^3 + 1\\
&\alpha^{15} = \alpha^4 + \alpha = 1
\end{aligned}
\tag{7.4}
$$

Definido el LFSR, seleccionamos la semilla

$$\text{Semilla} = \{\{1,0,0,1\},\{0,1,0,0\},\{0,1,0,1\},\{1,0,0,1\},\{0,1,0,1\},\{0,0,0,1\}\} \tag{7.5}$$

A continuación, se generan $2 \cdot 6 \cdot 4 + 1$ elementos que se corresponden con las filas de la siguiente matriz

$$\begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix} \tag{7.6}$$

Las columnas de esta matriz conforman las secuencias entrelazadas asociadas a este registro de desplazamiento. El análisis de estas secuencias, mediante el algoritmo de Berlekamp–Massey da como resultado el siguiente polinomio de realimentación para todas las secuencias.

$$\begin{aligned} r(x) = x^{24} + x^{23} + x^{21} + x^{19} + x^{18} + x^{17} + x^{13} + \\ + x^{11} + x^{10} + x^8 + x^6 + x^5 + x^3 + x + 1 \end{aligned} \tag{7.7}$$

Además, como se puede observar, el polinomio es de grado $6 \cdot 4 = 24$, con lo que se concluye que el polinomio $p(x)$ es primitivo en $GF(2^4)$.

7.4. Tiempos de ejecución

Para la realización de las pruebas se ha utilizado el software Mathematica en su versión 10,0,0,0 sobre una plataforma de Microsoft Windows de 64−bits. Se han programado cada una de las funciones realizadas al igual que cada uno de los bucles en el proceso de iteración. La máquina sobre la cual se han ejecutado los procesos posee un procesador Intel(R) Core (TM) $i7 - 4510U$ CPU @ 2,00 Ghz con una

memoria RAM de 16, 0 GB y un sistema operativo Windows 10 de 64 bits.

Para realizar la medida del tiempo de ejecución se ha tenido en cuenta la temperatura del procesador y el control de los procesos que en ese momento se estaban ejecutándo, con el fin de reproducir las mismas condiciones en cada experimento. Las siguientes figuras muestras la evolución comparada de los tiempos de ejecución en función del grado del polinomio, considerando las extensiones $n = 8$ y $n = 16$, en las que se puede apreciar la mejora del método propuesto.

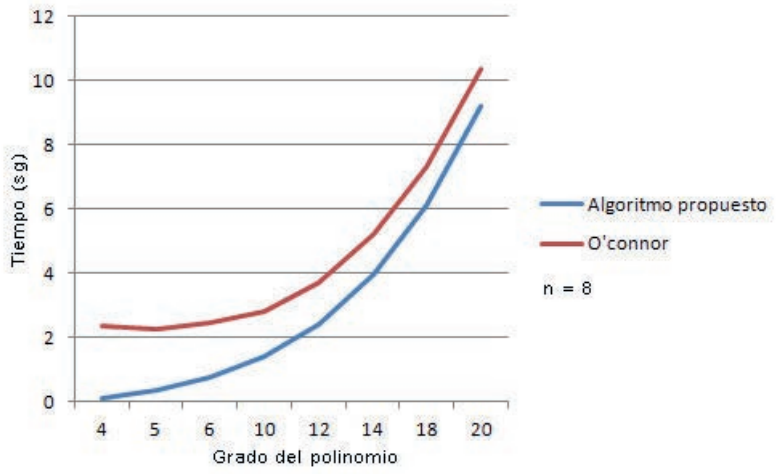

Figura 7.1: Tiempos de ejecución en $GF(2^8)$

Para poder comprobar si el método presentado mejoraba los resultados anteriores se ha realizado un estudio comparativo con el método presentado por Lenstra y Hendrik W [51]. Es evidente que para ambos métodos el número de operaciones depende del grado de los polinomios y del cardinal utilizado para la extensión del cuerpo finito. La principal ventaja presentada en nuestra propuesta es que sólo se han de generar $2 \cdot m \cdot n + 1$ bits mientras que en las propuestas que existen en la literatura se ha de generar la secuencia entera o de manera alternativa se han de efectuar tantas divisiones como elementos tenga la secuencia maximal sobre el

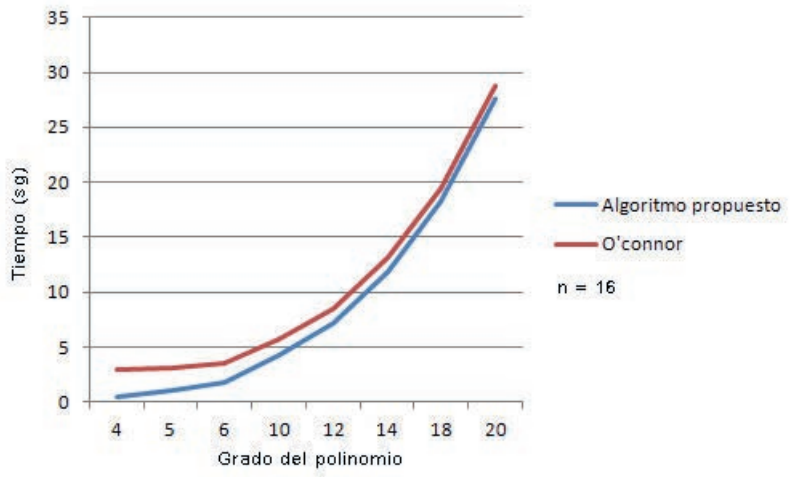

Figura 7.2: Tiempos de ejecución en $GF(2^{16})$

cuerpo extendido. Este proceso hace que la evolución de los tiempos de ejecución para un determinado cardinal del cuerpo finito en nuestro método sean lineales en lugar de geométricos/exponenciales.

En esta sección del trabajo, se ha descrito un método computacional para realizar un test que permite identificar si un polinomio con coeficientes en $GF(2^n)$ es primitivo sobre dicho cuerpo. La implementación del método analiza las secuencias binarias diezmadas a partir de la secuencia global generada por un LFSR definido sobre el cuerpo extendido, aprovechando la relación que existe entre ambos tipos de m-secuencias, que permite analizar la secuencia en el cuerpo extendido como secuencia entrelazada. El método propuesto reduce el tiempo de ejecución con respecto a las propuestas existentes.

Capítulo 8

Generador de números pseudoaleatorios gaussianos basado en rotaciones de registros de desplazamiento de realimentación lineal

8.1. Introducción

En este capítulo se presenta una nueva propuesta para generar números pseudoaleatorios óptimos con distribución Gaussiana. El generador está especialmente diseñado para la implementación de hardware de bajo coste, dado que, la implementación en hardware redunda en una mejora sustancial en los tiempos de ejecución. No obstante, también se considera la versión del software. Por esta razón, se emplean LFSR junto con rotaciones cíclicas. Esta nueva propuesta presenta un bajo

coste de implementación y supera las limitaciones de los generadores Gaussianos anteriores basados en LFSR mediante un algoritmo menos complejo para encontrar configuraciones óptimas. Como consecuencia, se convierte en un generador realmente utilizable. Además, se aplica una mejora adicional, basada en el algoritmo "*annealing*", para que los valores aleatorios se ajusten con una mayor precisión distribución normal.

Supongamos que tenemos una sucesión numérica generada por un registro de desplazamiento lineal de r estados. Supongamos que el estado inicial es $\{a_1, a_2, \cdots, a_r\}$. Supongamos que la función de realimentación que nos permite pasar de un estado al siguiente es $f(x) = a_0 + a_1 x + \cdots + a_r x^r$. En este caso denotaremos al LFSR generado bajo estas condiciones como $\langle m, f(x)\rangle$.

8.2. Generadores de números gaussianos utilizando registros de desplazamiento

Varios autores entre los que destacamos [44], [14] y finalmente [41] han propuesto la utilización de registros de desplazamiento lineales con realimentación, LFSR para generar números aleatorios con distribución Gaussiana realizando implementaciones directas del Teorema central del límite, CLT, es decir, generando varias secuencias de números aleatorios distribuidos uniformemente que luego se suman para que luego convolucionen a la distribución normal.

Para obtener implementaciones de bajo costo, Kang [44] en 2010 y Condo [14] en 2015 han propuesto un PRNG con solo un LFSR. La propuesta de Kang usa un LFSR para generar 4 sucesiones de números distintos que se suman para producir el valor aleatorio Gaussiano final. Para obtener estas 4 sucesiones de números,

el estado de un LFSR de $N = 4M$ bits se divide en bloques de M bits que se suman. El resultado de la adición se almacena en un acumulador. N ciclos de reloj más tarde, el estado del LFSR se divide nuevamente para producir una nueva entrada en el sumador. Esta operación se repite 8 veces para obtener finalmente un número pseudoaleatorio de $(N + 4)$ bits en la salida del sumador. Estos números siguen una distribución Gaussiana. Sin embargo, el PRNG no es eficiente debido al sobredimensionamiento requerido para el LFSR.

En 2015, Condo *et al* [14] propusieron también un PRNG Gaussiano usando solo un LFSR. En este caso, en lugar de dividir el estado, el sistema genera varias secuencias distribuidas uniformemente aplicando varias permutaciones a cada estado del LFSR. Más precisamente, se proponen dos versiones de PRNG en [14]. El primero, representado en la figura 8.1, produce 4 sucesiones numéricas. En cada instante o pulso de reloj t se genera un estado s^t. A partir de este se generan 3 números más aplicando a este estado s^t tres permutaciones distintas $\{\pi_i\pi_j\pi_k\}$ a $s^{(t)}$. Una vez aplicadas las permutaciones necesarias, se realiza una conversión decimal representada mediante $\gg$ y finalmente se realiza una suma decimal con acarreo. La segunda versión produce solo 3 secuencias de números porque solo aplica dos permutaciones $\{\pi_i, \pi_j\}$ a cada estado $s^{(t)}$.

Las sucesiones generadas por ambas versiones han sido analizadas el artículo publicado en 2015 [14] usando solo el LFSR $\{17, x^{17} + x^{14} + 1\}$. Sin embargo, los autores en [14] han proporcionado una estimación del coste de implementación para un PRNG genérico utilizando un LFSR de N etapas. Este diseño genérico requiere un registro de N bits, sumadores de $3N$ bits (o sumadores de $2N$ bits para la segunda versión) y r puertas XOR, siendo r el número de etapas para implementar la retroalimentación LFSR. Tengase en cuenta también que las permutaciones se pueden implementar codificando el orden de los cables que conectan el LFSR al

sumador. Por lo tanto, no requieren recursos de hardware adicionales, como puertas lógicas o registros, lo que ayuda a no aumentar el coste total de implementación. Como resultado, este PRNG tiene un coste de implementación menor que el PRNG [44] de Kang. Para generar números aleatorios de N bits, el PRNG de Kang requiere un LFSR con $4(N-4)$ etapas, $3(N-4)$- sumadores de bits y r puertas XOR.

A pesar de una baja complejidad en la implementación, este generador tiene algunos inconvenientes:

- Según los autores, solo 1/15 del conjunto todas las permutaciones $\{\pi_i, \pi_j\pi_k\}$ producen una sucesión de números con distribución de frecuencias Gaussiana. Además, el conjunto de tales permutaciones no se caracteriza, lo que implica la necesidad de tener que realizar una búsqueda exhaustiva para poder elegirlos. Además este método propuesto en [14] necesita la generación de la secuencia completa el análisis de cada combinación posible.
- Según los autores, el conjunto de permutaciones válidas depende de la semilla elegida, lo que complica enormemente su aplicación práctica.

8.3. Generador Gaussiano basado en rotaciones de LFSR

En esta sección describimos el generador propuesto siguiendo el mismo enfoque que en [14], es decir, la implementación del CLT aplicado sobre un conjunto de valores con distribución uniforme generadas por medio de un solo LFSR. La principal diferencia es que todas las secuencias de números se generan a partir de un LFSR único aplicando rotaciones cíclicas (un caso particular de permutaciones) en lugar de las permutaciones genéricas propuestas en [14]. La rotación es solo un

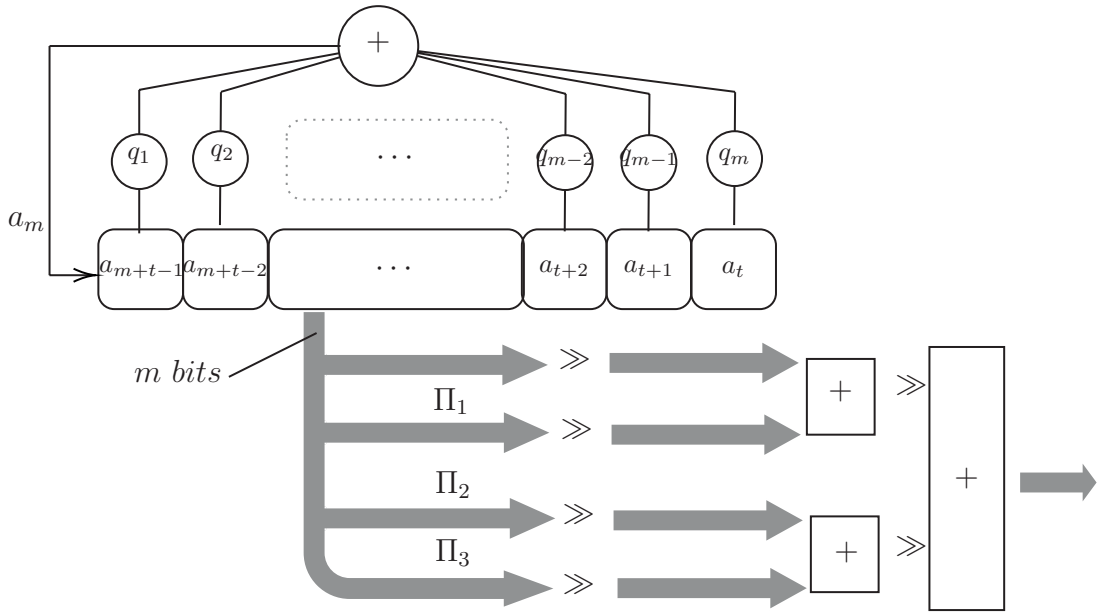

Figura 8.1: Esquema del generador propuesto por Condo y Gross

cambio cíclico del contenido de un estado dado del LFSR. Considerando el estado como un vector binario, la rotación implica el desplazamiento a la derecha de cada componente. El componente más a la derecha se desplaza ocupando un lugar más a la izquierda. Una rotación de orden k implica k rotaciones simples. Como consecuencia, dado que las rotaciones, así como las permutaciones, se pueden implementar codificando los cables que conectan el LFSR al sumador, el coste de implementación es el mismo, un registro de N bits, 3 sumadores de N bits (o 2 de Nbits en la segunda versión y J puertas lógicas XOR utilizadas para implementar el LFSR. Sin embargo, el porcentaje de rotaciones que producen números aleatorios Gaussianos es mucho mayor que el de las permutaciones genéricas. Este hecho permite seleccionar aleatoriamente las rotaciones con una alta probabilidad de que se puedan aplicar en la generación Gaussiana. De esta manera, resolvemos el principal inconveniente del PRNG de Condo *et al* [14].

El generador propuesto está diseñado utilizando dos versiones distintas (la pri-

mera usa tres rotaciones; la segunda solo dos) para facilitar la comparación con el PRNG en [14]. En ambas versiones, el LFSR está definido por un polinomio primitivo $p(x)$, por lo que produce una secuencia maximal.

En la figura 8.2, se muestra la primera versión, en la que el LFSR pasa por los $2^m - 1$ estados. En cada impulso de reloj t, el estado s^t se considera como un vector de m bits y se suma a otros tres números de m bits producidos aplicando tres rotaciones al estado $s^{(t)}$ como se indica.

Consideremos un LFSR $\langle m, p(x)\rangle$ donde $p(x)$ es un polinomio primitivo de grado m para producir una sucesión maximal de estados. Para cada estado LFSR, $s^{(t)}$ aplicamos una función de rotación $Rot^{(k)}$ se define como una rotación hacia la derecha de k desplazamientos. Entonces si $s^{(t)} = [a_{m-1+t}, \cdots, a_{1+t}, a_t]$ tenemos

$$\begin{aligned} Rot^{(1)}(s^{(t)}) &= Rot^{(1)}([a_{m-1+t}, \cdots, a_{1+t}, a_t]) = [a_t, a_{m-1+t}, \cdots, a_{1+t}] \\ Rot^{(2)}(s^{(t)}) &= Rot^{(2)}([a_{m-1+t}, \cdots, a_{1+t}, a_t]) = [a_{1+t}, a_t, a_{m-1+t}, \cdots, a_{2+t}] \\ &\cdots \end{aligned} \tag{8.1}$$

Debemos tener en cuenta que $Rot^{(m)}(s^{(t)}) = s^{(t)}$. En la primera versión del PRNG donde se aplican 3 rotaciones, denotaremos mediante $Rot_1^{(j)}, Rot_2^{(k)}, Rot_3^{(l)}$ al conjunto de rotaciones aplicadas al estado $s^{(t)}$, donde sin pérdida de generalidad, se tiene que $j \leq k \leq l$. Por otro lado, en la segunda versión y de manera similar denotaremos mediante $Rot_1^{(j)}$, $Rot_2^{(k)}$ al conjunto de pares de rotaciones aplicadas al estado $s^{(t)}$, donde igualmente y sin pérdida de generalidad podemos suponer que $j \leq k$.

Finalmente el número pseudoaleatorio generado en el instante t se representará mediante $RN^{(t)}$ y se calculará de la siguiente manera:

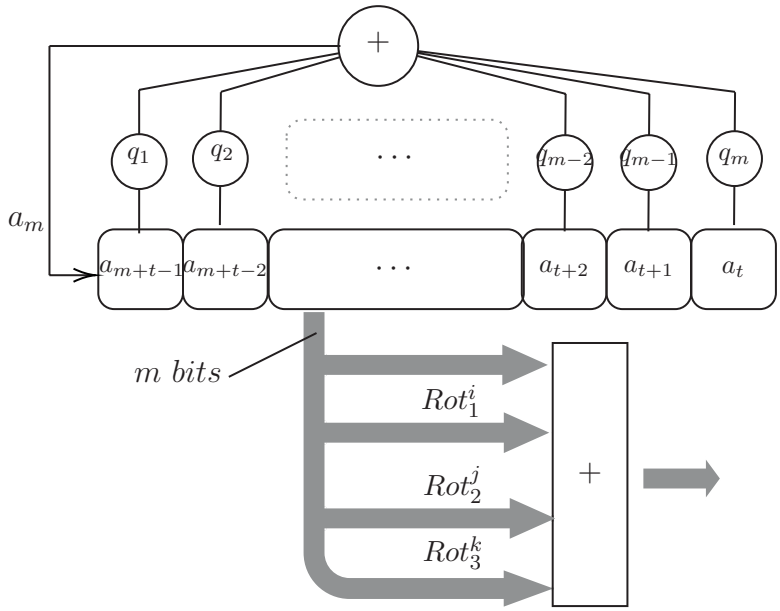

Figura 8.2: Generador Gaussiano basado en rortaciones de un LFSR.

$$\tau^{(t)} = D(s^{(t)}) + D(Rot_1^j(s^{(t)})) + D(Rot_2^k(s^{(t)})) + D(Rot_3^l(s^{(t)})) \tag{8.2}$$

Donde la función D es la función definida sobre ${\mathbb{Z}_2}^m$ en $\mathbb{R}$ tal que a cada vector de ${\mathbb{Z}_2}^m$ de $m-$bits lo transforma en su valor decimal asociado, o lo que es lo mismo:

$$D(s^t) = \sum_{i=0}^{m-1} 2^i \cdot a_{t+i} \tag{8.3}$$

También es importante hacer constar que la secuencia generada por el LFSR es siempre maximal y de longitud $2^m - 1$ independientemente de la semilla escogida, y siempre y cuando $p(x)$ sea un polinomio primitivo y la semilla sea un estado distinto de cero. Este hecho permite la utilización de cualquier polinomio primitivo y cualquier semilla distinta de cero y, por lo tanto, convierte al PRNG en uno realmente utilizable.

Debemos tener en cuenta que estas ecuaciones solo aparecen para mantener el

formalismo matemático pero no tiene que implementarse en hardware ya que los componentes electrónicos trabajan directamente con la representación binaria de los números generados.

8.4. Análisis estadístico de los resultados

En esta sección analizaremos el conjunto de valores numéricos obtenidos con el PRNG propuesto y su distribución de frecuencias. Se han aplicado distintos test de normalidad para cribar qué conjuntos de valores generados se aproximan o no a una distribución normal y caso de que sí se aproximen, hemos determinado la bondad del ajuste realizado, para así poder cuantificar cómo se ajustan los valores a una distribución normal. Se ha de tener en cuenta que se han realizado los tests para el caso de familias de 2 y de 3 rotaciones.

8.4.1. Bondad del Ajuste

Se han realizado distintos tests para conocer la bondad del ajuste realizado. En otras palabras se mide cómo se ajusta el conjunto de datos obtenidos a una distribución Normal. Para tal efecto hemos fijado una hipótesis nula H_0 en la que afirmamos que los datos se ajustan a una distribución normal de probabilidad. En contraposición hemos definido el suceso complementario dentro de este mismo espacio muestral afirmando que los datos no se encuentran normalmente distribuidos. Las pruebas realizadas nos determinan un valor p (o estadístico) que nos proporciona una medida en la que podemos afirmar si los datos siguen o no una determinada distribución de probabilidad o no. En nuestro caso la distribución normal. Para tal efecto hemos fijado un nivel de confianza del $\alpha = 95\,\%$, por lo que $1 - \alpha = 5\,\%$. Si el valor p es menor que un nivel de significancia elegido, entonces se rechaza la

Test de distribución de la Normal	**Estadístico**	p**-valor**
Anderson-Darling	0.194777	0.901172
Cramér-von Mises	0.0269862	0.896351
Jarque-Bera ALM	4.00919	0.130367
Kolmogorov-Smirnov	0.0109818	0.995996
Kuiper	0.0200591	0.993069
Mardia Combined	4.00919	0.130367
Mardia Kurtosis	−2.01337	0.0440758
Mardia Skewness	0.0502429	0.822641
Pearson χ^2	4.97263	1.0
Shapiro-Wilk	0.998453	0.504076
Watson U^2	0.0268772	0.863961

Tabla 8.1: Valores obtenidos en el $p-$ tests para el modelo de 3 rotaciones representado en la Figura 8.2, donde $n = 17$ y las rotaciones $Rot_1^2, Rot_2^6, Rot_3^8$ han sido aplicadas.

hipótesis nula de que los datos provienen de esa distribución.

Existen diferentes métodos y tests para distinguir si el rango de valores en una distribución sigue una distribución Normal. En la tabla 8.1 se muestran las pruebas de normalidad que hemos considerado durante el análisis. En la siguiente subsección describimos las más relevantes. La tabla muestra los resultados obtenidos en la aplicación de la prueba a los números generados cuando se implementan 3 rotaciones.

Para aplicar las pruebas definimos la variable estadística a analizar como:

$$X_{(i,j,k)} = \tau(s^n), \forall n \in \{0, 1, \cdots, 2^m - 1\} \tag{8.4}$$

En el caso del modelo de tres rotaciones y como

$$X_{(i,j)} = \tau(s^n), \forall n \in \{0, 1, \cdots, 2^m - 1\} \tag{8.5}$$

en el caso del modelo de dos rotaciones y donde la función τ es aquella definida en 8.2

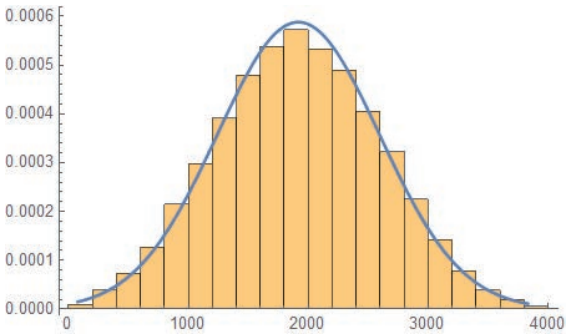

Figura 8.3: Histograma de frecuencias del modelo de 3 rotaciones generado por un LFSR con un polinomio de realimentación de grado $m = 10$

Teniendo como referencia CLT [92; 94], consideraremos $\{X_1, \cdots, X_n\}$ una sucesión de variables aleatorias independientes e idénticamente distribuidas, supongamos que todas ellas tienen media μ y desviación típica σ. Consideremos las medias muestrales definidas como $S_n = \frac{X_1+\cdots+X_n}{n}$. Entonces se verifica que la función de probabilidad para la media muestral converge hacia una distribución normal de media μ t desviación típica $\frac{\sigma}{\sqrt{n}}$

Por lo tanto si las variables aleatorias $X_1, X_2, \cdots, X_n$ forman una muestra aleatoria de tamaño n de una distribución dada como media μ y varianza σ^2, con $0 < \sigma^2 < \infty$, entonces para cada valor x_0 se verifica:

$$\lim_{n\to\infty}\left(\frac{\sqrt{n}\,(x_n-\mu)}{\sigma} \leq x_0\right) = \Phi(x_0) \tag{8.6}$$

donde Φ denota la función de distribución de una normal estándar de media $\mu = 0$ y desviación típica $\sigma = 1$.

En la figura 8.3 podemos visualizar el histograma de frecuencias de $X_{(2,5,8)}$ para 3 rotaciones donde el polinomio $p(x)$ es un polinomio primitivo de grado 10 y las rotaciones $Rot_1^2, Rot_2^5, Rot_3^8$ han sido aplicadas.

Anderson-Darling Test

El test de Anderson-Darling (A-D) es un test que se usa para comprobar si la muestra de datos obtenidos siguen una determinada distribución de probabilidad [80]. Es una modificación del test de Kolmogorov-Smirnov (K-S) en el que se da menos peso a las colas de la distribución. El test de K-S no depende de los valores críticos de la distribución que se quiere testar. Por el contrario el test de A-D hace uso de los valores críticos de la distribución por lo que es considerado un test mucho más sensible y fiable a la hora de determinar si los datos se ajustan o no a una distribución Normal de probabilidad [80]. El test de A-D también es considerado una buena alternativa al método de la Chi^2 cuando se necesita verificar una gran cantidad de valores.

La definición del test es como sigue:

Consideramos la hipótesis nula H_0: El conjunto de valores sigue una distribución de probabilidad determinada, en nuestro caso la distribución Normal de probabilidad. Por el contrario definimos H_a : El conjunto de valores no sigue una distribución de probabilidad determinada, en nuestro caso la distribución Normal de probabilidad. El test se define como:

$$A^2 = -N - S \tag{8.7}$$

donde

$$S = \sum_{i=1}^{N} \frac{(2i-1)}{N} \left[ln(F(Y_i) + ln(1 - F(Y_{N+1-i})\right] \tag{8.8}$$

y donde F es la función de distribución de los datos analizados, $Y_{i\in\mathbb{N}}$ es el conjunto de valores que se quieren testar y N es el conjunto de valores reales de la distribución, en nuestro caso denotados por N ya que se desea confrontar con la

distribución normal.

Los valores críticos del método de A-D dependen de la distribución escogida.

Shapiro-Wilk Test

El test de Shapiro-Wilk (S-W) [77] calcula un valor denotado mediante $\mathbb{W}$ que testa y verifica si los valores de una muestra tomada, $\{x_1, x_2, \cdots, x_n\}$ provienen o no de una distribución normal. Cuanto menor es el valor del estadístico mayor evidencia tendremos de que los valores se encuentran normalmente distribuidos. Los valores de comparación son obtenidos utilizando el método de Monte Carlo.

El valor del estadístico $\mathbb{W}$ se determina de la siguiente manera:

$$\mathbb{W} = \frac{\left(\sum_{i=1}^{n} \alpha_i x_{(i)}\right)^2}{\sum_{i=1}^{n} x_i - \overline{x}^2} \tag{8.9}$$

donde $x(i)$ representará al i-ésimo valor de los de la muestra una vez estos han sido ordenados de menor a mayor y por otro lado los α_i se calculan mediante:

$$(\alpha_1, \alpha_2, \cdots, \alpha_n) = \frac{m^T V^{-1}}{(m^T \cdot V^{-1} \cdot V^{-1} \cdot m)^{\frac{1}{2}}} \tag{8.10}$$

y donde $m = (m_1, m_2, \cdots, m_n)$ son los valores medios del estadístico ordenado, de variables aleatorias independientes e idénticamente distribuidas, muestreadas de distribuciones normales y V denota la matriz de covarianzas de ese estadístico de orden.

Test de la Chi^2

El test de la Chi-cuadrado (CHI) [80] se usa para testar si una muestra de datos numéricos proviene o no de una población con una distribución específica. En este caso, nos centraremos en esta prueba para comprobar si la distribución de números

se ajusta a la distribución normal.

Una característica interesante de la prueba de bondad de ajuste de CHI es que se puede aplicar a cualquier distribución univariante para la que se puede determinar la función de distribución de probabilidad acumulada. La prueba de bondad de ajuste de CHI se aplica a un conjunto de datos agrupados (es decir, los datos agrupados en clases). En realidad, esto no es una restricción, ya que para los datos no agrupados, simplemente puede calcular un histograma o una tabla de frecuencias antes de generar la prueba de CHI.

La prueba de CHI es una alternativa a las pruebas de bondad de ajuste de A-D y K-S, pero en contraposición a los anteriores, es mucho mejor en términos del análisis de colas de las distribuciones, por este hecho se considera esencial para el emulador de ruido Gaussiano.

Se considera la hipótesis nula, asumiendo que los datos siguen una distribución normal de probabilidad. El método de la CHI se define como sigue:

H_0 : Los datos siguen una distribución Normal.

H_1 : Los datos siguen no una distribución Normal.

Para estadístico de la CHI supongamos que los datos están divididos en k intervalos. Definimos el estadístico como:

$$\chi^2 = \sum_{i=1}^{k} \frac{(O_i - E_i)^2}{E_i} \tag{8.11}$$

Donde O_i es la frecuencia del $i-$ésimo valor observado en la muestra y E_i es la frecuencia real esperada de ese mismo valor en la distribución normal.

Supongamos que los valores de la variable X son $x_1, x_2, ..., x_n$. Después continuaremos con los siguientes pasos.

1. Categorizamos los (n) valores en k categorías.

2. Determinamos sus correspondientes frecuencias. f_i, $i \in \{1, 2, \cdots, k\}$, donde cada f_i es la frecuencia asociada a la $i-$ésima categoría.
3. Sea p_i la probabilidad de que, bajo la hipótesis nula, el valor real de la variable pertenezca a ese intervalo. Después calculamos el valor esperado de las frecuencias $E_i = np_i$ de las observaciones de la $i-$ésima categoría.
4. Bajo la hipótesis nula, suponemos que las variables aleatorias $f_1, f_2, \cdots, f_k$ siguen una distribución multinormal de parámetros, $n, p_1, p_2, ..., p_k$.
5. Calculamos el test estadítico de la CHI, $\chi_g^2 = \sum_{i=1}^{k} \frac{(O_i - E_i)^2}{E_i}$. El valor esperado del estadístico, bajo la hipótesis nula será $k - 1 - e$, o lo que es lo mismo $E\left[\chi_g^2\right] = k - 1 - e$.)
6. Los valores más grandes y los más pequeños del estadístico comparados con los valores reales esperados nos sugieren que la hipótesis nula no se cumple.
7. Si el $p-$ valor es suficientemente pequeño, se rechazará igualmente la hipótesis nula H_0

8.4.2. Resultados y comparación con los modelos existentes

Estas pruebas descritas se han utilizado para comprobar si el conjunto de valores $X_{(i,j,k)}$ obtenidos mediante la ecuación 8.4, se distribuyen de acuerdo con una distribución Normal. Se ha establecido un nivel mínimo de confianza del 90 %, por lo tanto, el nivel de significación se establece en 10 % y, de acuerdo con ese nivel de confianza, se han cribado la sucesiones de valores obtenidas. Es decir, dado un conjunto de valores, se han aplicado las pruebas de Normalidad para verificar si los datos siguen una distribución normal o no. El resultado de estas pruebas mencionadas es un valor de p. Si el valor de p obtenido es mayor que 90 %, la secuencia obtenida se aceptará como Normal y, en caso contrario, se ha descartado.

El generador propuesto ha sido testado para todas las combinaciones posibles

$\{Rot_1^j,\ Rot_2^k\}$ donde $0 \leq j < k \leq m$ en el caso de dos rotaciones y $\{Rot_1^j,\ Rot_2^k,\ Rot_3^l\}$ donde $0 \leq j < k < l \leq m$ en el caso de tres y se ha comprobado si generan un modelo de distribución Gaussiano. Todas las combinaciones se han probado utilizando el entorno de Mathematica, para LFSR cuyos polinomios de realimentación tienen grados $6 \leq m \leq 22$. Los tests que se han utilizando son los de CHI, los métodos A-D y S-W. Los resultados se resumen en la Tabla 8.2 donde para cada grado n, $6 \leq n \leq 22$ de los polinomios, se ha calculado el número total de rotaciones existentes en comparación con el número total de rotaciones válidas donde los $p - valores$ han superado el mínimo establecido de 90 %.

$$C_{m-1}^3 = \frac{(m-1)!}{3!(m-4)!}. \tag{8.12}$$

LFSR	$\{Rot_1^j,\ Rot_2^k,\ Rot_3^l\}$		$\{Rot_1^j,\ Rot_2^k\}$	
n	Total	Válidas(%)	Total	Válidas(%)
6	10	100	10	100
7	20	100	15	100
8	35	97,14	21	100
9	56	100	28	96,43
10	84	97,62	36	100
11	120	96,67	45	93,33
12	165	93,33	55	98,18
13	220	96,36	66	90,91
14	286	90,91	78	92,31
15	364	92,31	91	95,60
16	455	89,67	105	97,14
17	560	87,96	120	90
18	680	60,26	136	65,44
19	816	59,44	153	50,33
20	969	57,07	171	43,86
21	1140	45,79	190	31,05
22	1540	40,65	210	20,95

Tabla 8.2: Número de rotaciones válidas para un nivel de confianza del 90 %.

La figura 8.4 nos muestra un histograma de frecuencias para el caso particular

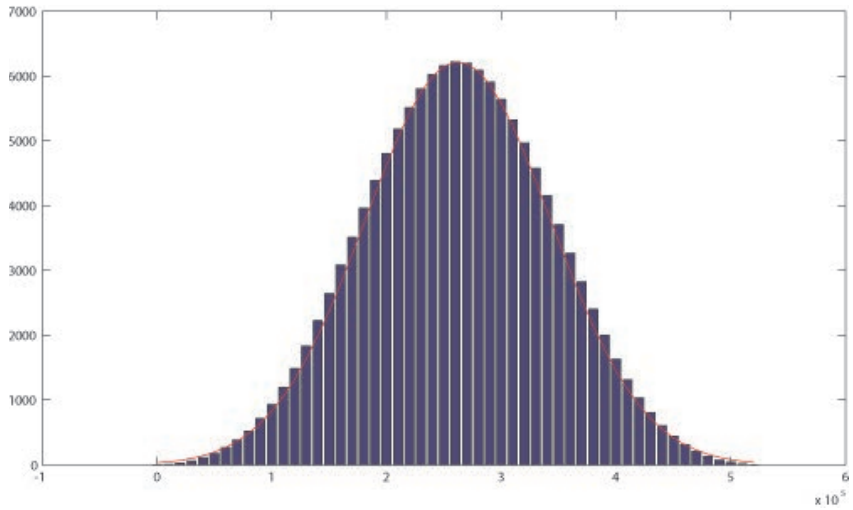

Figura 8.4: Histograma de frecuencias para $n = 17$ y el método de 3 rotaciones.

en el que el grado del polinomio primitivo aplicado es $n = 17$ y se ha aplicado un método de tres rotaciones.

Como se muestra en la Tabla 8.2, las rotaciones permiten un uso práctico del generador, ya que la elección del conjunto de rotaciones Rot_1^i, Rot_2^j, Rot_3^k es mucho más simple que la elección de permutaciones propuestas por Condo en [14]. La elección de la semilla, así como la elección del polinomio de realimentación primitivo, no son relevantes. Además, se obtiene un porcentaje mucho mayor de rotaciones válidas, 100 % en algunos casos, y estas generan una distribución Gaussiana. Las permutaciones válidas sugeridas por Condo en [14] representan solo 6,6 % del total.

El cardinal del conjunto de combinaciones válidas de dos rotaciones $Rot1^i$, $Rot2^j$, también se presentan en la tabla 8.2, considerando que el número total de combinaciones es:

$$C_{m-1}^2 = \frac{(m-1)!}{2!(m-3)!} \tag{8.13}$$

Como en el caso de tres rotaciones, la Tabla 8.2 nos muestra que dentro del

conjunto de todas las combinaciones posibles de dos rotaciones, más de un 90 % generan una sucesión de salida donde los números siguen una distribución Gaussiana, para cualquier m, longitud del LFSR. El resultado obtenido mediante las rotaciones también es independiente de la semilla y del polinomio primitivo de realimentación escogidos.

Finalmente, es importante indicar y resaltar que el coste de implementación es el mismo que el del generador Condo [14], ya que se utiliza el mismo tipo de operaciones y la misma cantidad de sumadores.

El PRNG propuesto también se ha comparado con el método de Box-Muller para la generación de números con distribución Guassiana (B-M) [91] que se diseñó como un método de generación de números pseudoaleatorios independientes y Normalmente distribuidos (Media $\mu = 0$ y varianza $\sigma^2 = 1$ unitaria). Si U_1 y U_2 son muestras independientes elegidas de la distribución uniforme en el intervalo unitario $(0, 1)$, entonces las variables definidas como:

$$Z_0 = Rcos(\Theta) = \sqrt{-2ln(U_1)} \cdot cos(2\pi U_2) \tag{8.14}$$

$$Z_1 = Rsin(\Theta) = \sqrt{-2ln(U_1)} \cdot sin(2\pi U_2) \tag{8.15}$$

son variables aleatorias independientes con una distribución Normal estándar.

Después de haber ejecutado el algoritmo B-M nos hemos encontrado con las siguientes desventajas:

- De acuerdo con los resultados presentados en la tabla 8.3, podemos ver que los $p-$ valores del test son mejores en el modelo de 3 rotaciones que en el algoritmo B-M.
- El coste computacional requerido para implementar el algoritmo de B-M es

Grado del polinomio	Número de valores	$p-$ test para B-M	$p-$ test para 3 rotaciones
$n = 6$	$2^6 = 64$	$p = 0{,}92$	$p = 0{,}99$
$n = 7$	$2^7 = 128$	$p = 0{,}91$	$p = 0{,}98$
$n = 8$	$2^8 = 256$	$p = 0{,}91$	$p = 1$
$n = 9$	$2^9 = 512$	$p = 0{,}90$	$p = 1$
$n = 10$	$2^{10} = 1024$	$p = 0{,}89$	$p = 0{,}99$
$n = 11$	$2^{11} = 2028$	$p = 0{,}86$	$p = 0{,}99$
$n = 12$	$2^{12} = 4096$	$p = 0{,}85$	$p = 0{,}99$
$n = 13$	$2^{13} = 8192$	$p = 0{,}85$	$p = 0{,}98$
$n = 14$	$2^{14} = 16384$	$p = 0{,}83$	$p = 0{,}97$
$n = 15$	$2^{15} = 32768$	$p = 0{,}83$	$p = 0{,}95$
$n = 16$	$2^{16} = 65536$	$p = 0{,}82$	$p = 0{,}93$
$n = 17$	$2^{17} = 131072$	$p = 0{,}81$	$p = 0{,}92$
$n = 18$	$2^{18} = 262144$	$p = 0{,}81$	$p = 0{,}92$
$n = 19$	$2^{19} = 524288$	$p = 0{,}78$	$p = 0{,}92$
$n = 20$	$2^{20} = 1048576$	$p = 0{,}77$	$p = 0{,}91$
$n = 21$	$2^{21} = 2197152$	$p = 0{,}76$	$p = 0{,}91$
$n = 22$	$2^{22} = 4194394$	$p = 0{,}76$	$p = 0{,}90$

Tabla 8.3: Evolución de los $p-$valores obtenidos después de haber aplicado el test CHI al método B-M y al modelo de 3 rotaciones.

mucho mayor.

8.5. Mejoras en los resultados obtenidos

En secciones anteriores, hemos demostrado que los PRNG propuestos mejoran los resultados obtenidos en [14] y en [44]. El PRNG propuesto ha sido diseñado como una particularización directa del sistema de Condo, utilizando solo rotaciones en lugar de una permutación genérica, tratando de obtener la solución más fácil (con la mínima modificación) a los problemas detectados en [14]. No obstante, la precisión de la distribución en el PRNG propuesto se puede mejorar aún más, es decir, se pueden aumentar los valores de los $p-test$. Dado que los resultados son más estables en la versión de tres rotaciones, hemos decidido basarnos en esta versión. Aunque el número de rotaciones válidas es más o menos el mismo, se ha obtenido una mejora sustancial en cuanto a su precisión. Para ello, se ha considerado un LFSR generado con un polinomio primitivo.

Consideramos un LFSR controlado por un polinomio primitivo de grado m, $\langle m, p(x)\rangle$. Sea $s^{(t)} = [a_{m-1+t}, a_{m-2+t}, \cdots, a_{t+1}, a_t]$ un estado del LFSR. Luego definimos las proyecciones Π_1 and Π_2 de la siguiente manera:

$$\Pi_1(s^t) = [0, a_{m-2+t}, \cdots, a_{t+1}, 0] \tag{8.16}$$

y

$$\Pi_2(s^t) = [0, 0, a_{m-3+t}, \cdots, a_{t+2}, 0, 0] \tag{8.17}$$

Entonces, el número aleatorio ahora se genera de la siguiente manera:

$$\begin{aligned} \tau\left(s^t\right) = \\ = D\left(Rot_1^0\left(s^t\right)\right) + D\left(Rot_2^i\left(s^t\right)\right) + D\left(Rot_3^j\left(s^t\right)\right) + D\left(Rot_4^k\left(s^t\right)\right) + \\ + D\left(\pi_1\left(Rot_4^k\left(s^t\right)\right)\right) + D\left(\pi_2\left(Rot_4^k\left(s^t\right)\right)\right) \end{aligned} \tag{8.18}$$

donde la función D está definida en la ecuación 8.2.

La figura 8.5 muestra la implementación de esta mejora. Como podemos observar, las proyecciones no incrementan el coste de implementación dado que no necesitan puertas lógicas o nuevos registros.

Hemos analizado todos los valores de n que verifican $6 \leq n \leq 22$. La cantidad de rotaciones y variables válidas es similar a las del modelo anterior, sin embargo, el nivel mínimo de aceptación que se ha establecido es del 95 % que corresponde a un error de estimación máximo por debajo del umbral de 5 %. La tabla 8.4 muestra los valores que se han alcanzado.

En este caso particular donde se han aplicado las proyecciones, hemos probado

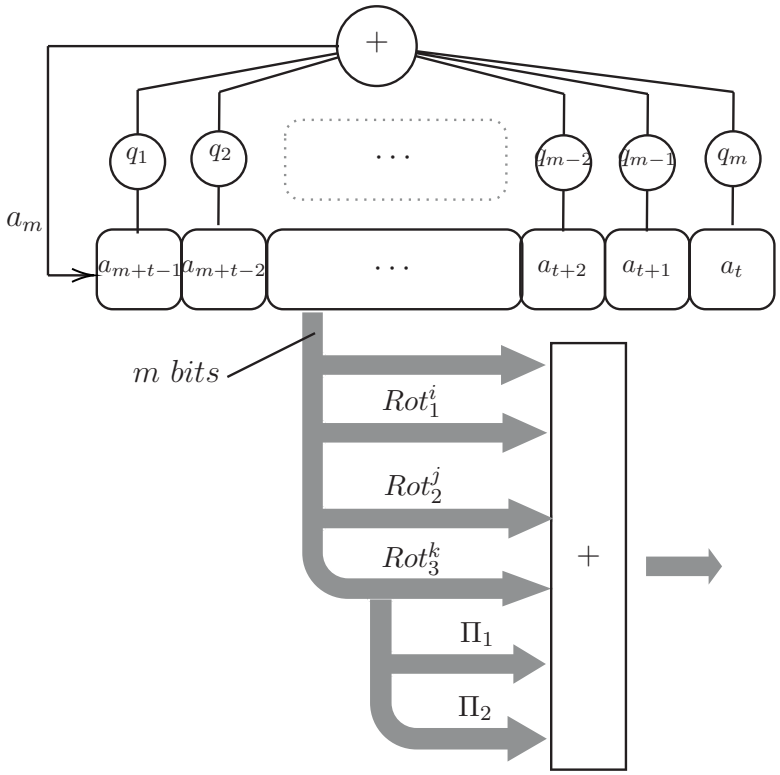

Figura 8.5: Generador mejorado basado en rotaciones de un LFSR

LFSR	Rot_1^i, Rot_2^j, Rot_3^k ,$\Pi_1(Rot_3^k)$, $\Pi_2(Rot_3^k)$	
n	Total	Valid(%)
6	60	100
7	105	100
8	168	100
9	252	94.04
10	360	94.44
11	495	94.34
12	660	94.09
13	858	93.70
14	1092	82.12
15	1365	79.96
16	1680	76.96
17	2040	72.84
18	2448	65.81
19	2907	68.04
20	3420	67.69
21	3990	68.25
22	4620	67.58

Tabla 8.4: % de rotaciones válidas para un nivel de confianza del 95 % donde las proyecciones Π_1 and Π_2 han sido aplicadas.

que, aunque el nivel de aceptación se ha fijado en un 95 % obtenemos más o menos el mismo número de rotaciones válidas. Es decir, no solo hemos encontrado un método que supera los modelos propuestos hasta la fecha, sino que también se ha depurado el sistema para adaptar el conjunto de valores a una distribución Normal. Si bien el sistema ha dado buenos resultados experimentales, hemos podido comprobar que cuando aumentamos el valor del grado del polinomio para obtener un mayor número de registros, entonces se obtiene una disminución en el número de rotaciones válidas.

Una vez obtenidos estos resultados, hemos intentado buscar un método que nos permita obtener un número similar de rotaciones válidas similar al obtenido en el apartado anterior y de la misma forma, si es posible, incrementar la eficiencia del sistema. Para ello, hemos utilizado el método simulated annealing (SA). Este método, también llamado recocido simulado, cristalización simulada o enfriamiento simulado, es un algoritmo de búsqueda metaheurística para problemas de optimización global [86]. El objetivo general de este tipo de algoritmos es encontrar una buena aproximación al valor óptimo de una función en un espacio de búsqueda grande. Dicho "óptimo globalçorresponde a la solución del problema de interés para el que no existe un mejor valor. En el caso de que tal problema sea de minimización, el óptimo global será aquél para el cual la función objetivo tenga el más pequeño posible de todos los de su espacio de búsqueda. Por el contrario, para un problema de maximización, el óptimo global es aquél con el valor más alto posible.

En cada iteración del algoritmo SA, se genera aleatoriamente un nuevo punto. La distancia del nuevo punto desde el punto actual, o la extensión de la búsqueda, se basa en una distribución de probabilidad con una escala proporcional a la temperatura. El algoritmo acepta todos los puntos nuevos que se acercan al objetivo, pero también, con cierta probabilidad, los puntos que se alejan del objetivo. Al

aceptar puntos que incrementan el objetivo, el algoritmo evita quedar atrapado en mínimos locales y es capaz de explorar globalmente en busca de más soluciones posibles. Se selecciona un programa de recocido para disminuir sistemáticamente la temperatura a medida que avanza el algoritmo. A medida que la temperatura disminuye, el algoritmo reduce el alcance de su búsqueda para converger al mínimo.

Para ilustrar cómo se ha implementado este método, ilustraremos el procedimiento a través de un ejemplo. Si tomamos el caso donde el grado del polinomio $n = 12$, partimos de vectores de dimensión 12 en los que estamos aplicando el método de rotaciones descrito en el apartado anterior. Una vez aplicado este método, se ha determinado cuál de las posibles rotaciones tiene mayor coeficiente y por tanto una distribución de valores más cercana a la distribución normal. En este caso los valores a tomar son los de Rot_1^2,Rot_2^4,Rot_3^8,$\Pi_1(Rot_3^8)$,$\Pi_2(Rot_3^8)$. Si usamos esta combinación de rotaciones, se obtiene un valor del $p-$ test cercano a 97,5 %. A partir de este punto, esta distribución de valores se tomará como referencia fija.

Una vez fijada la distribución de valores, hemos definido una función en la cual los valores obtenidos mediante las proyecciones definidas en 8.16 y 8.17, han sido multiplicadas por una dupla de factores fijos α y β. Entonces los valores obtenidos han sido calculados de la siguiente manera:

$$\begin{aligned} D\left(s^{(t)}\right) + D\left(Rot_1^2(s^{(t)})\right) + D\left(Rot_2^4(s^{(t)})\right) + D\left(Rot_3^8(s^{(t)})\right) + \\ +\alpha \cdot D\left(\Pi_1(Rot_3^8(s^{(t)}))\right) + \beta \cdot D\left(\Pi_2(Rot_8(s^{(t)}))\right) \end{aligned} \tag{8.19}$$

Donde la función D es la función que convierte a valores en base 10 definida en 8.2.

Para aplicar este método se ha definido una nueva función que depende de la sucesión de valores y de los parámetros de α y β y cuya salida será el valor

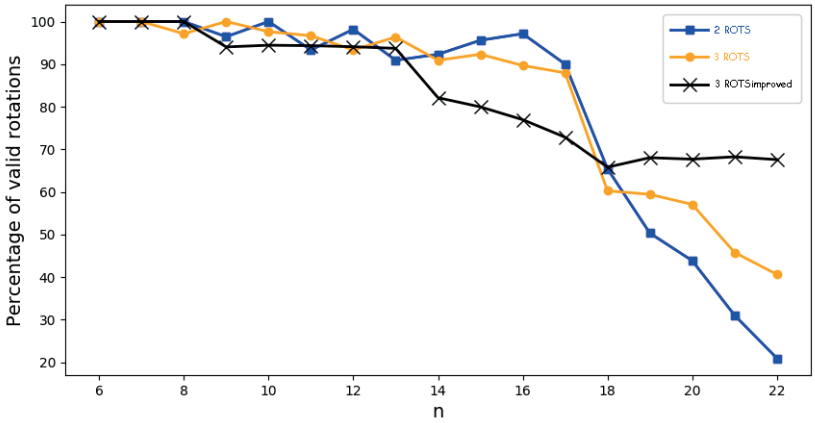

Figura 8.6: Evolución de los porcentajes de rotaciones válidas para los distintos modelos y para los distintos grados de los polinomios primitivos

1− y del $p-$valor. A esta función es a la que aplicaremos el método del SA para minimizar el valor de esta función. En el caso donde $n = 12$ se ha obtenido que $\alpha \to 1{,}48289, \beta \to 3{,}16175$. Con este resultado se ha incrementado significativamente el número de rotaciones válidas y de la misma forma se ha conseguido que estén mucho más cerca de la distribución Normal.

Para otros valores de n como es el caso de $n = \{10, 11\}$ se han obtenido siguiendo el mismo método los valores de α y β siguientes: $\alpha \to 0{,}882965$ y $\beta \to 4{,}53362$ por $n = 10$ y $\alpha \to 1{,}30476$ y $\beta \to 3{,}64635$ para el caso donde $n = 11$, los casos restantes ser considerado para estudios futuros.

En la figura 8.6 podemos observar la evolución del porcentaje de rotaciones válidas para cada grado del polinomio primitivo y para cada modelo (2 rotaciones, 3 rotaciones y 3 rotaciones con proyecciones 2). Como se ve en la línea de tendencia con cuadrados, que representa la tendencia del modelo de rotaciones de 2, el porcentaje de rotaciones válidas disminuye a medida que aumentamos el grado del polinomio primitivo. La situación mejora en el caso de la línea de tendencia con círculos donde se analiza el modelo de rotaciones de 3. Para polinomios pri-

mitivos con grado pequeño, la situación es aceptable, sin embargo, a medida que aumenta el grado del polinomio, disminuye el porcentaje de rotaciones válidas. Esta disminución no es tan notoria, sin embargo, no se obtienen valores adecuados. Finalmente, en la línea de tendencia de las cruces, (aunque los valores de α y β no son los óptimos sino que son los valores obtenidos para el caso en el que $n = 11$) representa el caso de las rotaciones de 3 y las 2 proyecciones, se observa que los porcentajes de rotaciones válidas son aceptables y constantes, incluso para valores suficientemente grandes para del grado del polinomio primitivo.

8.6. Conclusión

Esta sección de la tesis propone un nuevo generador de números pseudoaleatorios con distribución gaussiana utilizando un método que reduce el coste de implementación, como se aplica a [14]. La principal diferencia está en la caracterización de configuraciones válidas. Mientras que en [14] la propuesta se basa en el uso de permutaciones y realizar una búsqueda preliminar exhaustiva de las mismas, dependiendo de la semilla y retroalimentación polinomial, este trabajo propone el uso de un subconjunto de estas permutaciones, como son las rotaciones cíclicas, concluyendo que más del 90 % de las combinaciones de dichas rotaciones son adecuadas para su uso práctico. Además, el generador es independiente de la semilla y la realimentación polinomial una vez que se fija la longitud LFSR. Para mejorar el modelo, se han aplicado proyecciones adicionales al diseño inicial obteniendo números con mejores resultados en las pruebas de normalidad sin incrementar el coste de implementación. Finalmente, se ha aplicado el algoritmo de recocido simulado SA para optimizar los resultados obtenidos en las pruebas.

Capítulo 9

Generación de secuencias de alta complejidad lineal

9.1. Introducción

Dado que gran parte de la motivación que justifica la utilización de LFSR en la generación de números con distribución Gaussiana parte del extendido uso criptográfico de los LFSR, se describe en esta sección una utilización del generador diseñado en el capítulo anterior con posibles fines criptográficos. En particular, se presenta el resultado del estudio de la complejidad lineal de las secuencias binarias formadas a partir de un subconjunto de bits tomados de las secuencias con distribución normal producidas por el generador del capítulo anterior. Las secuencias finalmente obtenidas presentan unos valores de complejidad lineal de 2^{n-1}.

El generador Gaussiano diseñado utiliza rotaciones ciclicas de los sucesivos estados del LFSR para generar la secuencia final con distribución Gaussiana. En consecuencia, se analizarán las secuencias binarias obtenidas a partir de las secuencias decimales pseudoaleatorias producidas por el generador de [15], para su posible

utilización como secuencias cifrantes. Para ello se estudia la complejidad lineal, la distribución de ceros y unos y la autocorrelación de estas secuencias, dando como resultado una configuración que permite alcanzar un valor de complejidad lineal de 2^{n-1}.

El esquema general se basa en una implementación del CLT mediante la suma de cuatro secuencias psuedoaleatorias con distribución uniforme tal y como se observa en la figura 9.1). Estas secuencias son las que genera el LFSR aplicando las rotaciones, del siguiente modo.

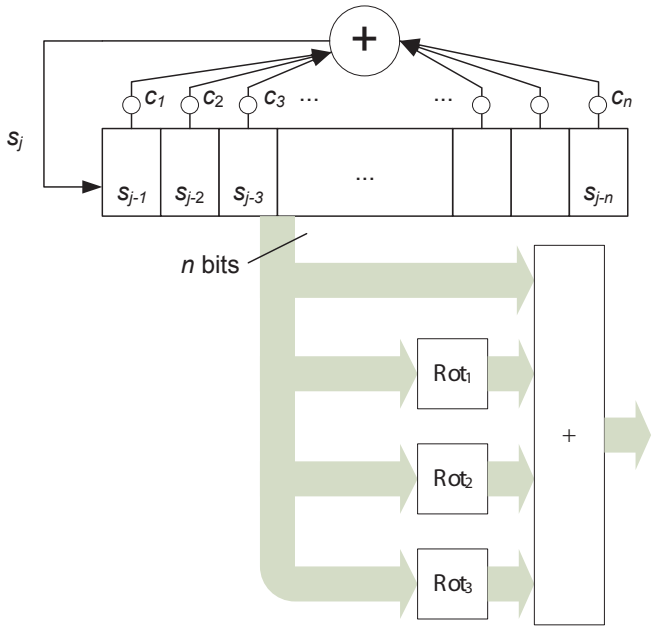

Figura 9.1: Generador Gaussiano basado en rotaciones

Sea $S_0 = (S_0(t)); t \geq 0$ la sucesión de estados del LFSR en la que $S_0(t+1) = f_p(S_0(t))$, siendo $f_p : GF(2^n) \longrightarrow GF(2^n)$ la función que genera el siguiente estado del LFSR aplicando la realimentación definida por el polinomio $p(x)$. En consecuencia, $S_0(t+k) = {f_p}^k(S_0(0))$ donde $S_0(0)$ es la semilla del generador.

Por otra parte, las rotaciones cíclicas Rot_1, Rot_2, Rot_3 definidas sobre los ele-

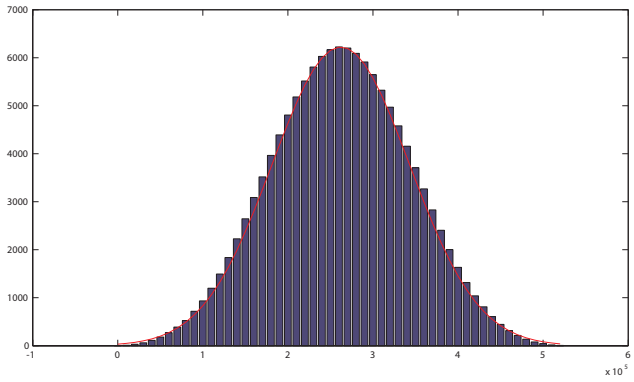

Figura 9.2: Distribución con $n = 17$ y tres rotaciones

mentos de S_0 generan las sucesiones

$$S_i = (S_i(t)) = (Rot_i(S_0(t))); 1 \leq i \leq 3 \tag{9.1}$$

En consecuencia, en cada instante de tiempo t, los n bits que constituyen el estado actual del LFSR se transforman en otros tres valores mediante sendas rotaciones Rot_1, Rot_2, Rot_3. De este modo, se obtienen cuatro secuencias diferentes de números de n bits, que una vez sumados generarán la secuencia de salida $S_T = (S_T(t))$, como

$$S_T(t) = \sum_{i=0}^{3} S_i(t), \tag{9.2}$$

donde el operador suma se aplica sobre los valores decimales que representan los elementos de cada una de las secuencias. Más concretamente, se sigue el convenio de considerar que la realimentación del LFSR se aplica sobre el bit menos significativo (LSB) de los n que constituyen el estado del LFSR. Como se puede comprobar en la figura 9.2, la sucesión de salida para el caso de utilizar $n = 17$ bits sigue un distribución normal.

El período de la secuencia S_T está determinado por el polinomio de realimen-

tación del LFSR. Al ser éste primitivo, la secuencia S_0 tiene período de longitud máxima, $2^n - 1$. Del mismo modo, la aplicación de cualquiera de las rotaciones Rot_i sobre los elementos de S_0 determinan que el período de S_i sea también máximo. En consecuencia, S_T es también una secuencia de longitud $2^n - 1$.

La distribución estadística de S_T no depende del polinomio primitivo $p(x)$ de realimentación aplicado al LFSR. Al tener S_0 una longitud máxima, contiene a todos los elementos no nulos del $GF(2^n)$. Un polinomio primitivo $p'(x) \neq p(x)$, genera también todos los elementos no nulos. Por tanto, la única diferencia está en el orden en que aparecen en la sucesión S_0. Desde el punto de vista del análisis de la distribución estadística, no importa el orden en el que aparecen los elementos en la secuencia, sino cuáles son estos elementos. Por idéntica razón, la modificación de la semilla afecta únicamente al orden en el que van apareciendo los elementos de la sucesión. Por todo ello, se puede afirmar que la distribución estadística de la secuencia final S_T es independiente tanto del polinomio de realimentación como de la semilla elegida, y solo depende de la combinación de rotaciones Rot_1, Rot_2, Rot_3 que se apliquen al generador.

Los resultados experimentales realizados es [15] indican que para $6 \leq n \leq 17$ al menos el 90 % de las rotaciones generan datos con distribución Gaussiana, llegando al 100 % en algunos casos.

9.2. Secuencias de alta complejidad lineal

Como es bien sabido, las secuencias generadas por un LFSR no pueden ser utilizadas como secuencias cifrantes debido a su alta predictibilidad. Una m-secuencia completa, generada por un LFSR de n bits, puede reproducirse a partir de sólo $2n$ bits. Con el fin de reducir dicha predictibilidad, causada por la linealidad inherente

del LFSR, se aplican habitualmente diferentes técnicas a las secuencias generadas por el LFSR, como el filtrado no lineal, la decimación irregular, las combinaciones no lineales de varios LFSR, o la actualización dinámica de la realimentación (DLFSR) [63; 65].

Para medir la mejora que se consigue con estas técnicas se utiliza el concepto de complejidad lineal, definida de la siguiente forma.

Definición 9.2.1. La complejidad lineal de una secuencia $S = (S(t)); t \geq 0$, representada por $LC(S)$, es la longitud del LFSR más corto que puede generar dicha secuencia. De forma equivalente, se define $LC(S)$ como el menor entero e para el que existe un polinomio de grado e

$$q(x) = x^e + \sum_{i=0}^{e-1} a_i \cdot x^i, \tag{9.3}$$

tal que la secuencia S es generada por la recurrencia

$$S(t) = \sum_{i=1}^{e} a_i \cdot S(t-i). \tag{9.4}$$

Aplicando la definición, las m-secuencias generadas por un LFSR de n bits tienen complejidad lineal $LC(S) = n$. El objetivo de las técnicas aplicadas sobre estas secuencias no es sólo aumentar considerablemente la comlejidada lineal de la secuencia total S, sino hacerlo de manera que también aumente a medida que se va generando la secuencia, ofreciendo un comportamiento similar al de las secuencia puramente aleatorias. Para ello, se estudia el gráfico de perfil de complejidad que representa los valores $LC(S^j)$, para $1 \leq j \leq T(S)$, siendo $T(S)$ el período de la secuencia y $S^j = (S(1), S(2), \ldots, S(j))$, los j primeros elementos de la secuencia S.

Observando el generador de la figura 9.1, se puede apreciar que la secuencia de salida S_T es una sucesión de números que pueden ser presentados con $n+2$ bits a lo sumo, puesto que la suma decimal aplicada en la ecuación9.2 genera números menores o iguales que $4 \cdot (2^n - 1)$.

A continuación, se presenta el análisis de las secuencias binarias extraídas de la secuencia S con distribución Gaussiana, producida por el generador definido en [15].

9.3. Secuencias binarias

Partiendo de la secuencia S_T producida por el generador de la figura 9.1, se definen las secuencias binarias $S_{T,i} = (S_{T,i}(t))$ con $0 \leq i \leq n+2$, donde

$$S_{T,i}(t) = bit_i(S_T(t)), \tag{9.5}$$

siendo $bit_i : GF(2^n) \longrightarrow GF(2)$ la función que convierte el estado del LFSR en su bit i-ésimo, considerando $i = 0$ el bit menos significativo (LSB) e $i = n+2$ el bit más significativo (MSB).

Tras una primera observación, se descartan las secuencias $S_{T,0}$ y $S_{T,n+2}$, es decir, las que corresponden con el LSB y el MSB, respectivamente. La secuencia binaria que determina el MSB estará compuesta de una gran cantidad de ceros debido a que tanto el bit $n+1$ como el $n+2$ aparecen debido al acarreo producido por la suma decimal. En consecuencia, la gran mayoría de los elementos tendrán estos bits a cero, por lo que la distribución estadística no cumplirá los criterios de Golomb. En cuanto a la secuencia binaria definida por el LSB, la ausencia de acarreo en la suma de este bit, lo convierte en una combinación lineal de las cuatro secuencias S_0, S_1, S_2 y S_3, es decir,

n	Rotaciones	LC
7	2-3-4	64
7	2-3-6	64
7	2-5-6	64
12	2-4-6	769
12	4-7-11	793
17	3-7-14	3213

Tabla 9.1: Complejidad lineal de secuencias binarias $S_{T,3}$

$$S_{T,0}(t) = bit_0\left(S_0\left(t\right)\right) + bit_0\left(S_1\left(t\right)\right) + bit_0\left(S_2\left(t\right)\right) + bit_0\left(S_3\left(t\right)\right) \tag{9.6}$$

y en consecuencia, no mejora la complejidad lineal.

El estudio experimental realizado sobre el resto de secuencias binarias refleja, en término generales, un aumento considerable de la complejidad lineal que asciende hasta un valor aproximado de $n2^{\frac{n}{2}}$. La tabla 9.1 muestra los resultados más representativos de la aplicación del algoritmo de Massey-Berlekammp para LFSR de longitudes 7, 12 y 17, utilizando conjuntos de rotaciones que garantizan una distribución Gaussiana, conforme a lo establecido en [15].

A pesar del aumento considerable de la complejidad lineal, la gráfica de autocorrelación muestra valores altos y la distribución de rachas se aleja del estándar para longitudes elevadas; es decir, se mantiene en los límites en cuanto al número total de ceros y unos, y de rachas de poca longitud, pero está desequilibrada en rachas de longitudes elevadas.

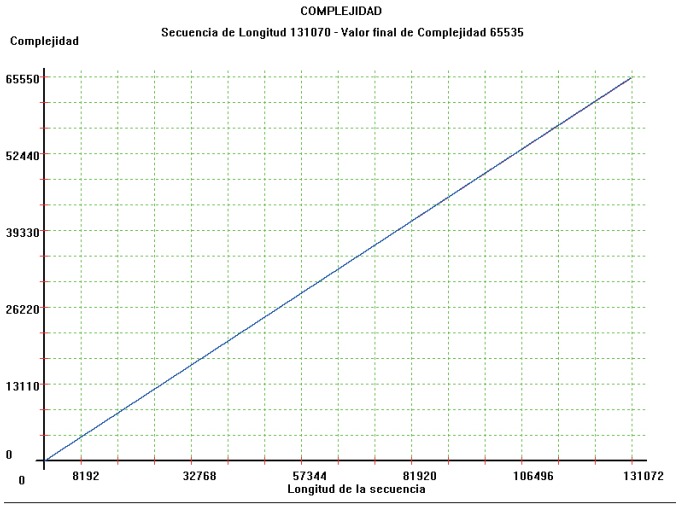

Figura 9.3: Perfil de Complejidad lineal para $n = 17$

9.4. Combinación de secuencias binarias

A la vista de los resultados obtenidos con las secuencias binarias, se consideran en este apartado las secuencias resultantes de la combinación de algunas de estas secuencias. Los primeros resultados experimentales demuestran que la combinación lineal de tres secuencias binarias producen resultados excelentes. Así, para el caso de $n = 12$ y $n = 17$, utilizando las rotaciones $4-7-11$ que garantizan distribución gaussiana en S_T, las secuencias formadas por la suma $S_{T,3} + S_{T,6} + S_{T,7}$, tienen una complejidad lineal $LC(S) = 2^{n-1}$, esto es, 2048 y 65535, respectivamente.

Además, presentan una buena autocorrelación y distribución de rachas. Las figuras 9.3 y 9.4 muestran el perfil de complejidad lineal y la autocorrelación para $n = 17$.

9.5. Conclusión

Se han presentado los resultados del estudio experimental realizado sobre las secuencias binarias extraídas a partir de la salida de un generador pseudoaleatorio

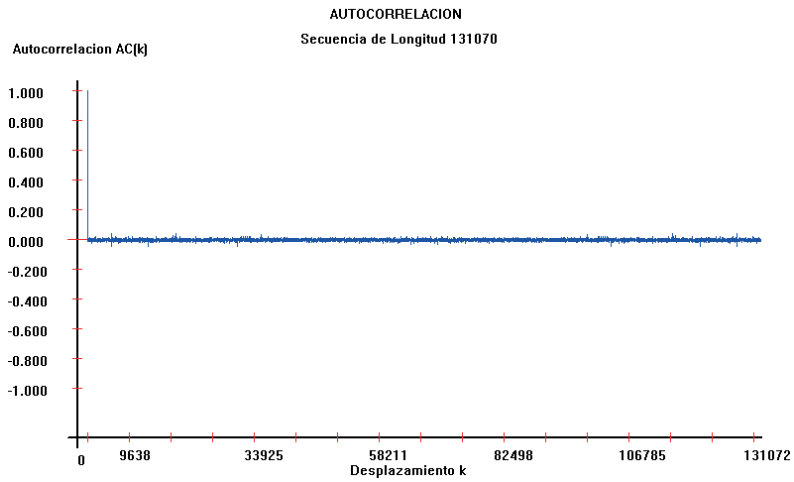

Figura 9.4: Autocorrelación para $n = 17$

con distribución normal construido mediante LFSR. A pesar de que la distribución normal no es adecuada para las aplicaciones criptográficas, las secuencias binarias obtenidas mediante la combinación de algunos bits de la salida presentan una complejidad lineal que alcanza valores de 2^{n-1}, manteniendo una buena autocorrelación y una buena distribución según los criterios de Golomb.

Capítulo 10

Generador de números Gaussianos sobre $GF(2^n)$

10.1. Introducción

A lo largo de este documento hemos resaltado la necesidad de poder generar números pseudoaleatorios con una distribución Gaussiana de probabilidad. Hemos podido observar que uno de los restos a los que nos podemos enfrentar es el del cardinal del conjunto de números generados. Paralelamente, el desarrollo de los nuevos procesadores nos permiten trabajar con una longitud de palabra mucho mayor, es por esto que para poder aumentar el cardinal del conjunto de números generados nos planteamos la posibilidad de que la generación de números se produzca a través de registros de desplazamiento con realimentación lineal definidos sobre un cuerpo extendido de Galois $GF(2^n)$. Las ventajas son evidentes ya que al trabajar sobre un cuerpo extendido la secuencia de estados generada tendrá en cada impulso de reloj $m \cdot n$ bits, siendo m el polinomio primitivo de realimentación sobre el cual está definido el cuerpo. Con lo que una vez generada las secuencias podemos obtener un

modelo sobre $GF(2)$ de secuencias diezmadas equivalente al modelo sobre $GF(2^n)$. Como venimos indicando con los nuevos procesadores no tenemos el inconveniente de trabajar con un cuerpo extendido ya que los procesadores actuales pueden ejecutar el procedimiento con un rendimiento adecuado. Finalmente, tal y como se podrá observar al final de este capítulo podemos afirmar que la generación que se va a proponer supera a los modelos existentes en la bibliografía dando respuesta a las exigencias criptográficas de los nuevos modelos de encriptación de datos.

10.2. Modelo equivalente para este generador Gaussiano

En la sección 7.2 hemos descrito cómo a través de las secuencias diezmadas podemos obtener un modelo equivalente desde $GF(2^n)$ a $GF(2)$. Vamos a profundizar en este aspecto para tratar de llegar a una mejor compresión del modelo que proponemos.

Como conocemos [60], existe una relación entre las secuencias maximales obtenidas en $GF(2^n)$ y aquellas secuencias maximales generadas en $GF(2)$, de tal modo que estas segundas se pueden obtener a través de las primeras y viceversa. La relación se establece a través de los polinomios primitivos que definen en cada caso el LFSR.Más concretamente, si $q(x)$ es un polinomio primitivo de grado m que genera la sucesión en $GF(2^n)$, las secuencias diezmadas obtenidas a través de las proyecciones sobre el $j-$ésimo elemento de la secuencia de estados de la $m-$secuencia generan un LFSR binario. Este último LFSR puede ser generado a través de un polinomio $h(x)$ de grado $m \cdot n$, además se verifica que $h(x)$ divide a $q(x)$ Esto nos permite desarrollar un modelo equivalente de LFSR binario, ya que si dispone-

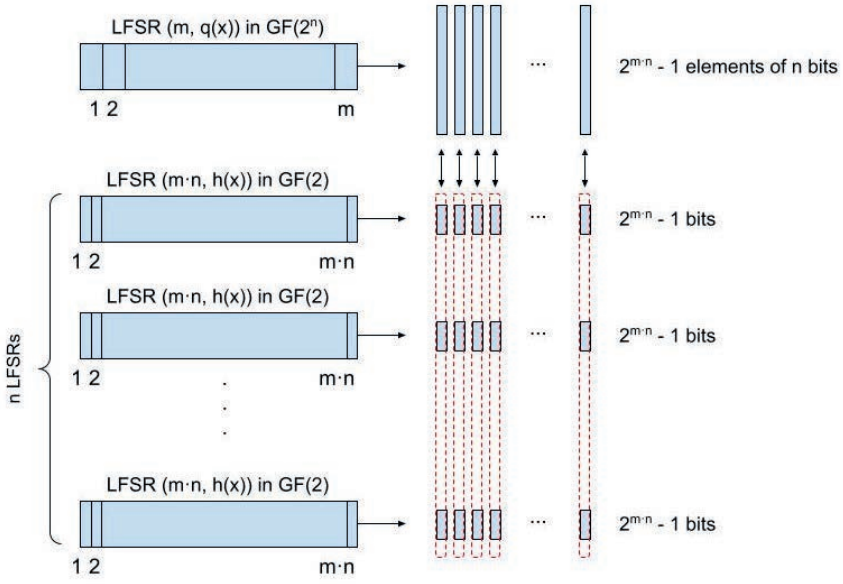

Figura 10.1: Modelo binario equivalente de un LFSR sobre $GF(2^n)$.

mos de un LFSR definido sobre $GF(2^n)$, podemos obtener n secuencias maximales sobre $GF(2)$ tal y como se muestra en la figura 10.1, donde el $j-$ésimo bit de cada elemento en la secuencia generada es el elemento que se añade a la secuencia diezmada. Nótese que todos los polinomios $h(x)$ en un modelo de secuencias diezmadas son todos iguales, aunque su semilla sea distinta. Entonces, las secuencias generadas son $m-$ secuencias en $GF(2)$ y por lo tanto las secuencias obtenidas son secuencias desplazadas y/o rotadas de la misas. En este sentido la generación de una $m-$secuencia en $GF(2^n)$ implica la generación de n secuencias binaras de m bits maximales, que pueden ser aplicadas como distribuciones uniformes para poder aplicar posteriormente el CLT

Definición 10.2.1. Sea $\langle m, q(x)\rangle$ un LFSR sobre $GF(2^n)$. Para cada $t \in \mathbb{N}$, sea $s(t) = [a_{m-1+t}, a_{m-2+t}, \cdots, a_t]$ el estado t del LFSR, donde $a_i \in GF(2^n)$ puede ser representada mediante el vector $[a_{i,n-1}, \ldots, a_{i,1}, a_{i,0}]$ con $a_{i,j} \in GF(2)$ $\forall 0 \leq j \leq n-1$. Entonces, Π_k se define como la proyección sobre la compo-

nente $k-$ésima de un elemento. Entonces, $\Pi_k(a_i) = a_{i,k}$ y $\Pi_k(s(t)) = s(t)_k = [a_{m-1+t,k}, a_{m-2+t,k}, \cdots, a_{t,k}]$.

$$\Pi_k(A) = (\Pi_k(a_0), \Pi_k(a_1), \Pi_k(a_2), \cdots) \tag{10.1}$$

Teorema 10.2.1. *Sea $\langle m, q(x)\rangle$ un LFSR definido sobre $GF(2^n)$ y sea $\langle r_k, f_k(x)\rangle$ el conjunto de todas las secuencias diezmadas $\forall k \in \{1, 2, \cdots, n\}$ entonces, $q(x)$ es un polinomio primitivo sobre $GF(2^n)$ sii $\forall k \in \{1, 2, \cdots, n\}$ f_k es primitivo sobre $GF(2)$. Además si se cumplen estas condiciones, entonces $f_j(x) = f_k(x) = h(x)$ $\forall j, k \in \{1, 2, \cdots, n\}$ y se verifica que $m \mid r_k$*

Por lo tanto, si tenemos un LFSR $\langle m, q(x)\rangle$ sobre $GF(2^n)$ donde $q(x)$ es primitivo, entonces podemos considerar que tenemos n secuencias maximales sobre $GF(2)$.

En otras palabras, este modelo equivalente representa el proceso de entrelazado que genera la secuencia en $GF(2^n)$; es decir, la secuencia generada por el LFSR en $GF(2^n)$ puede expresarse como una secuencia entrelazada, en el sentido descrito por Gong en [35], compuesta por n secuencias componentes, correspondiente a las secuencias diezmadas. Más precisamente, será una secuencia entrelazada primitiva ya que todas las secuencias componentes son generadas por el mismo polinomio primitivo en $GF(2)$.

Por otro lado, las secuencias pseudoaleatorias deben ser difíciles de reproducir. La complejidad lineal complejidad lineal (LC) de una secuencia se define como el grado del polinomio mínimo que la genera, o, de manera equivalente, la longitud del LFSR más corto que la genera [58]. En consecuencia, la complejidad lineal de una secuencia generada por una LFSR donde el polinomio es primitivo es la longitud de esa LFSR. Por lo tanto, considerando la secuencia producida por un LFSR de

m celdas en $GF(2^n)$ como una secuencia intercalada, su complejidad lineal LC_{ext} es n veces la complejidad lineal LC_{bin} de sus componentes originales; eso es,

$$LC_{ext} = n \cdot LC_{bin} = m \cdot n^2 \tag{10.2}$$

10.3. Generador Gaussiano propuesto

Teniendo en cuenta que una de las aplicaciones potenciales de un PRNG gaussiano basado en LFSR podría ser un esquema QKD [52], es importante tener en cuenta los siguientes aspectos:

- La PRNG debería permitir que se genere un conjunto discreto de valores lo suficientemente grande como para aproximarse a la distribución de probabilidad continua.
- El conjunto de valores generado debe tener una distribución de probabilidad Gaussiana.
- La seguridad del sistema debe permitir la generación de un conjunto de valores con un cardinal suficientemente grande.
- La generación de los valores obtenidos debe realizarse lo más rápido posible y con el menor coste de implementación. Además, para que el sistema sea eficaz, se debe considerar la posibilidad de una implementación de hardware.
- El sistema debe permitir que la generación de los valores pseudoaleatorios con distribución Gaussiana sean diferentes para cada una de sus distintas ejecuciones.

Para cumplir con todos estos requisitos, proponemos un PRNG basado en un LFSR en $GF(2^n)$. El PRNG está compuesto por dos unidades: la unidad de control y la unidad de procesamiento (Véase la Figura 10.2). La unidad de control consiste

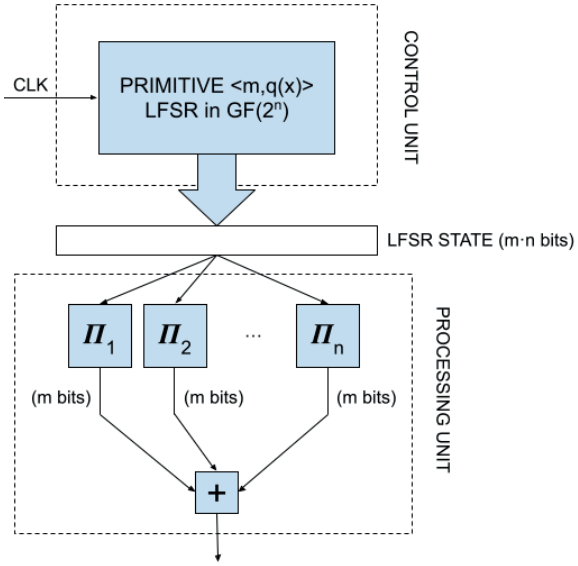

Figura 10.2: Gaussian number generator proposed.

en un LFSR con m celdas, generadas en cada estado por un polinomio primitivo $q(x)$ sobre $GF(2^n)$. Esta *unidad* es responsable de la generación de la secuencia $m-$ a través la cual se obtienen todas las secuencias con distribución uniforme para la posterior aplicación de CLT. Como se describe en las secciones anteriores, si $q(x)$ es primitivo, el LFSR genera una secuencia maximal o $m-$ secuencia es decir una secuencia de período máximo $2^{n \cdot m} - 1$.

La unidad de procesamiento se ha diseñado con un doble objetivo. Por un lado, como muchos otros criptosistemas, aplica un filtrado no lineal a la secuencia producida por el LFSR en la unidad de control para aumentar la dificultad de que un observador no autorizado reproduzca toda la secuencia. Por otro lado, esta unidad implementa las operaciones que transforman la distribución estadística de uniforme a Gaussiana, es decir, implementa el CLT. Para hacer eso, el operador Π_k se aplica en cada estado LFSR, produciendo así n cadenas de m bits que se corresponden con segmentos de las n secuencias binarias en el modelo equivalente descrito en la Sección 10.2 . Por lo tanto, aplicando el CLT, el número pseudoaleatorio B se

obtiene mediante la suma entera de esas n cadenas de bits como:

$$B = D(s_1^t) + D(s_2^t) + \cdots + D(s_n^t) \tag{10.3}$$

donde D es la función que opera un vector de $m-$bits $x = (x_0, x_1, \cdots, x_{m-1})$ en su correspondiente valor decimal,

$$D(x) = \sum_{i=1}^{n-1} 2^i \cdot x_i \tag{10.4}$$

El período, rango y precisión del PRNG propuesto se puede configurar usando dos parámetros principales: m y n. La secuencia de números pseudoaleatorios tiene un período de $2^{mn} - 1$ porque el polinomio de retroalimentación en el LFSR es un polinomio primitivo de grado m en $GF(2^n)$. El rango de los números está determinado por m ya que cada estado del LFSR contiene $n \cdot m$ bits que se dividen en n cadenas de m bits que luego se suman para producir el número pseudoaleatorio. El parámetro n aumenta ligeramente, hasta $m + log_2(n)$, el número de bits de los números generados debido al acarreo de la suma entera. Por lo tanto, para $m = 5$ y $n = 8$, el PRNG genera valores con un rango de $5 + 3 = 8$ bits.

Finalmente, cuando se aplica el CLT para obtener una distribución Gaussiana, su precisión está relacionada con el número de números distribuidos uniformemente que se suman. En este caso, cada número pseudoaleatorio se obtiene sumando valores de n. Por lo tanto, la generación de números pseudoaleatorios de $16-$ bits requiere la utilización de un LFSR con 16 celdas. Si el LFSR está definido en $GF(2^4)$, el PRNG presentará un período de $2^{4 \cdot 16} - 1 = 2^{64} - 1$, obtenido de la suma de 4 secuencias uniformemente distribuidas. La precisión puede aumentar usando valores de 8 en lugar de 4. En ese caso, el LFSR funcionaría en $GF(2^8)$ lo que también aumentaría el período a $2^{16 \cdot 8} - 1 = 2^{128} - 1$.

En cuanto a la velocidad de generación, aunque en general, el uso de LFSRs en $GF(2^n)$ está motivado por el aumento de velocidad que se logra al generar n bits en lugar de 1 en cada iteración, en este caso, el uso de el LFSR en $GF(2^n)$ persigue un segundo objetivo: aprovechar la relación implícita con n secuencias binarias diferentes. Esto facilita la aplicación del CLT mediante el uso de un solo LFSR que es equivalente a n LFSRs binarios. Por lo tanto, la tasa final de generación de números con distribución Gaussiana es la misma que la tasa a la que un LFSR binario genera un bit. Sin embargo, teniendo en cuenta que los números generados tienen una longitud de m bits, la velocidad de generación en bits por segundo es m veces mayor.

10.4. Análisis estadístico

Para comprobar que los resultados se aproximan debidamente a la distribución normal se han aplicado los mismos Tests descritos en la sección 8.4. No obstante, debido a la gran cantidad de números generados, estos análisis de han completado con los siguientes tests.

Medidas de tendencia central, Dispersión, Curtosis y Asimetría

Para comprobar si estas medidas hacen que los valores encajen de forma fehaciente con los datos de una distribución Gaussiana, hemos normalizado los resultados aplicando la transformación elemental

$$Z = \frac{X - \mu}{\sigma} \tag{10.5}$$

A partir de aquí, hemos determinado las medidas de tendencia central de la variable, las medidas de dispersión y las medidas de asimetría y curtosis. Lo primero

que tenemos que verificar es que si se aumentan los valores del grado del polinomio de retroalimentación y se aumenta el cardinal del cuerpo extendido, los valores obtenidos se ajustan mejor a la distribución normal, obteniendo en cada caso valores más próximos a los valores reales de la distribución Normal.

Si los datos numéricos se han normalizado, en el sentido de que hemos aplicado la tipificación 10.5, entonces podemos establecer los diversos parámetros de control para verificar si los datos siguen una distribución normal.

Los valores esperados para la distribución normal son los siguientes:

1. Los cuartiles Q_1 y Q_3 deberán ser $Q_1 = -0{,}67448, Q_3 = 0{,}67448$.
2. Aproximadamente el 95 % de las observaciones están dentro del intervalo centrado en la media y amplitud 2 veces la desviación típica. O lo que es lo mismo, el 95 % de los valores estarán dentro del intervalo $(-1{,}96, 1{,}96)$. Aproximadamente 68 % de las observaciones estarán dentro del intervalo centrado en 0 y de amplitud $1 \times \sigma$ o lo que es lo mismo $(-0{,}47, +0{,}47)$, y alrededor del $99,7\,\%$ de las observaciones estarían dentro del intervalo centrado en la media y de amplitud $3 \times \sigma$ o lo que es lo mismo $(-2{,}75, +2{,}75)$.
3. La desviación típica deberá ser $\sigma = 1$ y la media $\mu = 0$.
4. La asimetría de una distribución normal es cero, y cualquier dato simétrico debe tener una asimetría cercana a cero. Los valores negativos para la asimetría indican datos sesgados hacia la izquierda y los valores positivos para la asimetría indican datos sesgados hacia la derecha. Por asimétrico a la izquierda, queremos decir que la cola izquierda es mayor en relación con la cola derecha. De manera similar, asimétrico a la derecha significa que la cola derecha es mayor en relación con la cola izquierda.
5. El coeficiente de curtosis para una distribución normal estándar debe ser 3.

Grado (q_i)	Polinomio $q_i(x)$
4	$q_4 = x^4 + x + 1$
5	$q_5 = x^5 + x^2 + 1$
6	$q_6 = x^6 + x + 1$
7	$q_7 = x^7 + x + 1$
8	$q_8 = x^8 + x^4 + x^3 + x^2 + 1$
16	$q_{16} = x^{16} + x^{12} + x^3 + x^1 + 1$

Tabla 10.1: Polinomios utilizados en la generación de los cuerpos extendidos.

10.5. Resultados

En esta sección pasaremos a mostrar los resultados que se han obtenido para la generación de números con distribución gaussiana.

Para ilustrar los resultados obtenidos en la generación de números con distribución gaussiana, se han tenido en cuenta los siguientes polinomios $\{q_i\}_{i\in\mathbb{N}}$ para la generación de los cuerpos extendidos y los siguientes polinomios como polinomios de polinomios de realimentación $\{p_j\}_{j\in\mathbb{N}}$.

Para generar todos los cuerpos extendidos se han utilizado diferentes polinomios primitivos. Los grados de estos polinomios han oscilado entre 4 y 16. La lista de polinomios primitivos que se han utilizado se representa en la Tabla 10.1.

Continuaremos describiendo los polinomios primitivos que hemos utilizado como polinomio de realimentación para cada campo extendido. Se ha de tener en cuenta que debido a la gran longitud de cada uno de los términos de cada polinomio, se ha realizado la conversión a hexadecimal.

A partir de ahora denotaremos que κ_{ij} como el LFSR $\langle j, p_j\rangle$ y del mismo modo $j \in \{4, 5, 6, 7, 8, 16\}$ representará el grado del polinomio primitivo sobre el cuerpo extendido $GF(2^i)$. Por lo tanto, de acuerdo con la Tabla 10.2, el LFSR $\kappa_{45}(x)$ será el LFSR generado por

$$p(x) = \{\{1,0,0,0\},\{0,1,0,1\},\{0,0,1,1\},\{1,1,0,1\},\{1,0,0,1\},\{0,0,0,1\}\}$$

Polinomio $q_i(x)$	Grado (p_i)	$p_i(x)$
q_4	4	$\{a, 5, 9, 6\}$
q_4	5	$\{6, 5, a\ , 8, 9\}$
q_4	6	$\{6, 3, c, a\ , 9, b\}$
q_4	7	$\{6, 3, c, a\ , 2, 9, b\}$
q_4	8	$\{6, 3, c, a\ , d, 2, 9, b\}$
q_5	4	$\{15, a\ , 11, 16\}$
q_5	5	$\{15, a\ , 6, 17, 13\}$
q_5	6	$\{15, a\ , 17, 6, 17, 13\}$
q_5	7	$\{15, a\ , 17, 16, 6, 17, 13\}$
q_5	8	$\{15, a\ , 17, 14, 16, 6, 17, 13\}$
q_6	4	$\{2d, 25, 31, 36\}$
q_6	5	$\{2d, 25, 31, 36, 25\}$
q_6	6	$\{2d, 23, 25, 31, 36, 25\}$
q_6	7	$\{2d, 23, 25, 31, 14, 36, 25\}$
q_6	8	$\{2d, 23, 25, 31, 14, 36, 26, 25\}$
q_7	4	$\{45, 65, 55, 76\}$
q_7	5	$\{45, 65, 55, 65, 76\}$
q_7	6	$\{45, 65, 55, 55, 65, 57\}$
q_7	7	$\{45, 65, 55, 55, 61, 65, 57\}$
q_7	8	$\{45, 65, 63, 66, 55, 61, 65, 57\}$
q_8	4	$\{95, e5, 55, ed\}$
q_8	5	$\{aa, 55, 93, 65, 99\}$
q_8	6	$\{aa, 55, 93, b1, 65, 99\}$
q_8	7	$\{aa, 55, 74, 93, b1, 65, 99\}$
q_8	8	$\{80, aa, 0, 0, 0, 1\}$
q_{16}	24	$\{1002d, 10039, 1003f, 10053, 100bd, 100d7, 1012f, 1013d\ , 1014f,$ $1015d,\ 10197, 101a1, 101ad, 101bf, 101c7, 10215, 10219, 10225,$ $1022f, 1025d, 66157, 10285, 10291, 102a1\}$

Tabla 10.2: Tabla de los polinomios primitivos usados para la generación del cuerpo extendido para cada $q_i(x)$.

sobre $GF(2^4)$.

Primero, determinaremos la media aritmética, la desviación típica y los cuartiles de los valores numéricos obtenidos. De acuerdo con la Tabla 10.3, se puede observar que todos los valores obtenidos están dentro del rango de valores esperados y que estos se ajustan a los valores teóricos de una distribución Gaussiana.

De la misma forma hemos generado todos los cuantiles y los hemos comparado con los valores teóricos de los de la distribución Gaussiana. En cada uno de los casos se puede verificar que los valores se ajustan al modelo Gaussiano teórico. En la Figura 10.3 representamos gráficamente la lista de cuantiles obtenida enfrentada a la lista de cuartiles teóricos de una distribución Normal para alguno de los casos estudiados.

Generador	Media μ	Desv Típ σ	$\{Q_1, Q_2, Q_3\}$
κ_{44}	$1{,}28113 \cdot 10^{-17}$	1.	$\{-0{,}652405, -0{,}000815332, 0{,}650774\}$
κ_{45}	$7{,}92835 \cdot 10^{-18}$	1.	$\{-0{,}652215, -0{,}000012122, 0{,}650214\}$
κ_{46}	$7{,}21716 \cdot 10^{-18}$	1.	$\{-0{,}6754, -0{,}0100635, 0{,}699629\}$
κ_{47}	$4{,}02607 \cdot 10^{-18}$	1.	$\{-0{,}699102, 0{,}0137925, 0{,}696134\}$
κ_{48}	$2{,}62951 \cdot 10^{-17}$	1.	$\{-0{,}696669, -0{,}000285917, 0{,}696097\}$
κ_{54}	$-2{,}60767 \cdot 10^{-17}$	1.	$\{-0{,}660901, -0{,}00813484, 0{,}644631\}$
κ_{55}	$-3{,}87947 \cdot 10^{-17}$	1.	$\{-0{,}671761, -0{,}0087515, 0{,}676358\}$
κ_{56}	$4{,}76993 \cdot 10^{-17}$	1.	$\{-0{,}671761, -0{,}0087515, 0{,}676358\}$
κ_{57}	$1{,}74443 \cdot 10^{-17}$	1.	$\{-0{,}680723, 0{,}00183247, 0{,}684388\}$
κ_{58}	$3{,}47529 \cdot 10^{-17}$	1.	$\{-0{,}687545, -0{,}0026389, 0{,}691847\}$
κ_{64}	$3{,}90608 \cdot 10^{-17}$	1.	$\{-0{,}659401, 0{,}000866493, 0{,}661134\}$
κ_{65}	$3{,}20183 \cdot 10^{-17}$	1.	$\{-0{,}711603, -0{,}026917, 0{,}706675\}$
κ_{66}	$2{,}84824 \cdot 10^{-17}$	1.	$\{-0{,}685646, -0{,}000279343, 0{,}685087\}$
κ_{67}	$7{,}32401 \cdot 10^{-17}$	1.	$\{-0{,}688653, 0{,}0113709, 0{,}680959\}$
κ_{68}	$2{,}24457 \cdot 10^{-17}$	1.	$-0{,}67203, -0{,}00582178, 0{,}68299$
κ_{74}	$1{,}26656 \cdot 10^{-17}$	1.	$\{-0{,}65119, 0{,}000928774, 0{,}653048\}$
κ_{75}	$8{,}49908 \cdot 10^{-18}$	1.	$\{-0{,}699352, -0{,}0176537, 0{,}712738\}$
κ_{76}	$8{,}49908 \cdot 10^{-18}$	1.	$\{-0{,}673098, -0{,}00289631, 0{,}667306\}$
κ_{77}	$1{,}39497 \cdot 10^{-18}$	1.	$\{-0{,}694148, 0{,}00888319, 0{,}701576\}$
κ_{78}	$-4{,}53049 \cdot 10^{-18}$	1.	$\{-0{,}693528, 0{,}0000646841, 0{,}703291\}$
κ_{84}	$5{,}43506 \cdot 10^{-17}$	1.	$\{-0{,}649619, 0{,}0029409, 0{,}655501\}$
κ_{85}	$3{,}3536 \cdot 10^{-18}$	1.	$\{-0{,}698031, -0{,}0244844, 0{,}697172\}$
κ_{86}	$1{,}42056 \cdot 10^{-17}$	1.	$\{-0{,}693034, -0{,}00748555, 0{,}700178\}$
κ_{87}	$-3{,}13666 \cdot 10^{-18}$	1.	$\{-0{,}685563, -0{,}00883622, 0{,}69865\}$
κ_{88}	$-3{,}93139 \cdot 10^{-19}$	1.	$\{-0{,}687156, -0{,}0022108, 0{,}682735\}$
κ_{1624}	$1{,}90913 \cdot 10^{-17}$	1.	$\{-0{,}690607, -0{,}005302, 0{,}672002\}$

Tabla 10.3: Media, Desviación típica y cuartiles para los valores obtenidos usando los polinomios de realimentación y para la generación del cuerpo extendido determinados por $\kappa_{i,j}$.

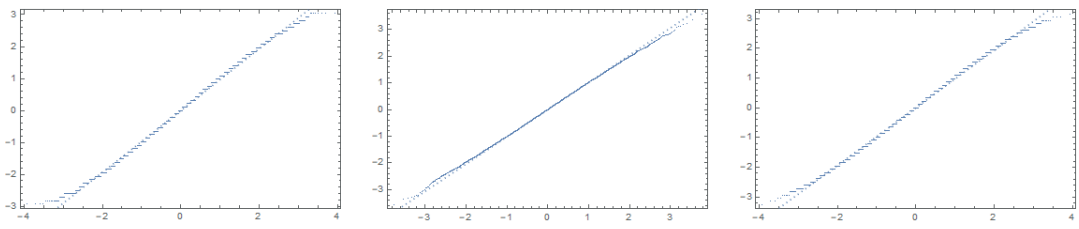

Figura 10.3: Comparación mediante representación gráfica de los cuantiles de una distribución Normal teórica frente a los obtenidos mediante nuestro modelo para el caso κ_{44}, κ_{58} y κ_{74} respectivamente.

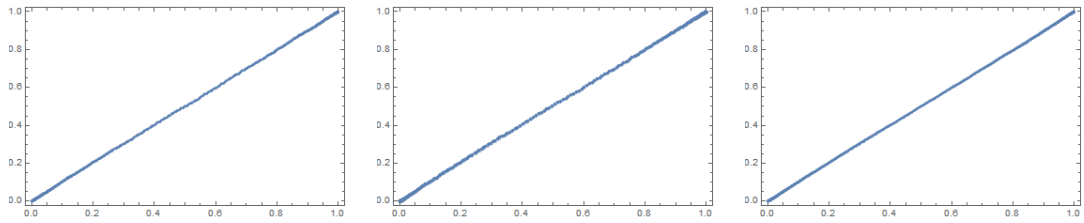

Figura 10.4: Representación gráfica de las CDF de los valores teóricos de una distribución normal confrontados a las CDF de los valores obtenidos para los casos κ_{67}, κ_{86} y κ_{88} respectivamente.

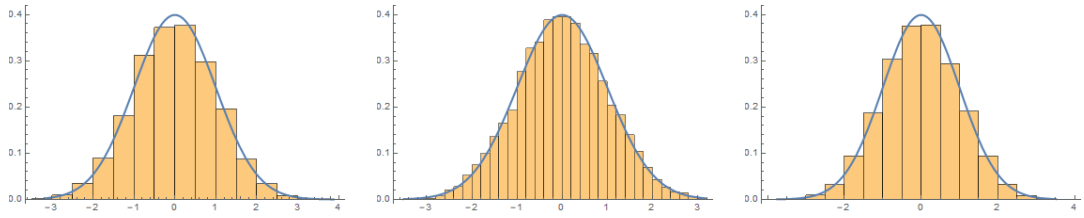

Figura 10.5: Representación gráfica de los histogramas obtenidos confrontados a la curva de una distribución teórica normal para los casos κ_{57}, κ_{68} y κ_{88} respectivamente.

Para sustentar mejor los resultados presentados, hemos representado la función de distribución acumulada (cumulative distribution function) (CDF) y los resultados obtenidos y los hemos comparado con los esperados en la distribución normal. En la Figura 10.4 confrontamos los valores teóricos de la función 10.4 de la distribución normal con los resultados obtenidos.

En este sentido, también cabe destacar que se han analizado los histogramas de los resultados obtenidos y estos se han comparado dentro de los correspondientes histogramas de la distribución normal. Se ha aplicado la corrección de continuidad ya que se está utilizando un conjunto discreto de valores y también hemos podido verificar que los datos se ajustan a una distribución normal de valores. En la Figura 10.5 podemos ver cómo los histogramas tienden hacia la curva de distribución normal de CDF.

Se utilizaron diferentes tests de normalidad para comprobar si los datos de los

Test de Bondad del ajuste	Estadístico	*P*-valor
Anderson–Darling	0.16486	0.956321
Cramér–von Mises	0.0248859	0.924378
Jarque–Bera ALM	0.214813	0.894745
Mardia Combined	0.214813	0.894745
Mardia Kurtosis	0.00564325	0.995497
Mardia Skewness	0.205783	0.650093
Pearson χ^2	13.4135	0.966447
Shapiro–Wilk	0.997609	0.907292

Tabla 10.4: Tests de bondad del ajuste para el caso κ_{44}.

modelos se ajustaban a la distribución Normal, como podemos ver en la Tabla 10.4.

Estas pruebas descritas se han utilizado para comprobar si las variables estadísticas generadas por los modelos, se distribuyen según una distribución Normal. Se ha establecido un nivel mínimo de confianza del 90 %, por lo tanto, el nivel de significancción se establece es del 10 % y, de acuerdo con ese nivel de confianza, se ha examinado la secuencia obtenida. Es decir, dado un conjunto de valores, se han aplicado las pruebas de Normalidad para verificar si los datos siguieron una distribución normal o no. Si el valor de prueba p obtenido es mayor que 90 %, la secuencia obtenida se considera aceptada y, de lo contrario, ha sido descartada.

El generador propuesto se ha probado para todos los polinomios posibles. Todas las combinaciones de polinomios primitivos se han probado utilizando el entorno de Mathematica. Debido al cardinal de los datos generados, las pruebas se han realizado utilizando para el cribado de los modelos han sido CHI, los métodos A-D y S-W. Los resultados se ejemplifican en la Tabla 10.4.

El PRNG propuesto también se ha comparado con el algoritmo B-M, descrito anteriormente en las ecuaciones 8.14, [91].

Después de haber ejecutado el algoritmo de B-M hemos encontrado las siguientes desventajas.

LFSR	B-M	Modelo	Número de valores
0.95112	0.87231	$\kappa_{4,4}$	65,535
0.90900	0.761121	$\kappa_{4,5}$	1,048,575
0.90101	0.755181	$\kappa_{6,5}$	1,073,741,823
0.90012	0.699876	$\kappa_{6,6}$	68,719,476,735

Tabla 10.5: Evolución de los valores de los $p-$ test después de haber aplicado la prueba de la CHI al conjunto de valores obtenidos por el método del LFSR propuesto y por el método de B-M

- De acuerdo con los resultados presentados en la Tabla 10.5, podemos ver que los valores de la prueba p son mejores para nuestro modelo basado en los LFSR, que en el algoritmo B-M.
- El coste computacional requerido para implementar el algoritmo es mucho mayor en el caso del método de B-M.

Otra forma de mejorar la precisión de la distribución Gaussiana es modificar la forma en que se generan las cadenas de m -bit. Si se utilizan todos los estados del LFSR, cada celda se utiliza en la generación de n números pseudoaleatorios consecutivos. Aunque las pruebas de ajuste revelan muy buenos resultados (consultar la Sección 10.5), disminuir o eliminar la cantidad de números afectados por la misma celda ayudaría a mejorar la precisión. Por lo tanto, proponemos, como alternativa, utilizar uno de cada m de los estados. Más formalmente, proponemos diezmar en m la producción LFSR. De esta manera, el período sería $(2^{mn}-1)/gcd((2^{mn}-1),m)$ dando lugar a seleccionar m,n tal que $gcd\left((2^{mn}-1),m\right)=1$ para alcanzar el mismo período $2^{mn}-1$.

En cualquier caso, es importante señalar que el período es mucho mayor que el rango, es decir, $2^{m\cdot n}-1 \gg 2^{m+log_2(n)}$, dando lugar a una probabilidad de aproximadamente 0,005 de generar el mismo número en cada $10,000$ valores generados.

En esta sección se ha propuesto un nuevo PRNG gaussiano. Se basa en un LFSR único, utilizando el mismo enfoque que las propuestas anteriores [44; 14; 18; 19],

para generar un cierto número de secuencias de números distribuidos uniformemente, necesarios para aplicar el CLT. A diferencia de las propuestas anteriores, no se han aplicado permutaciones o rotaciones explícitas a los sucesivos estados LFSR. En cambio, el PRNG se opera en $GF(2^n)$ para aprovechar la relación entre los estados y secuencias generadas en $GF(2^n)$ y $GF(2)$, que permite representar la $m-$ secuencias en $GF(2^n)$ como secuencias primitivas entrelazadas compuestas por $m-$ secuencias en $GF(2)$.

El PRNG, presentado en la Sección 10.3, permite configurarlo mediante dos parámetros principales, m (el número de celdas en el LFSR) y n (la dimensión de $GF(2^n)$), determinando el período como $2^{m \cdot n} - 1$, y el rango $2^{m+log_2(n)}$. El análisis estadístico revela un excelente comportamiento cuando se aplican las pruebas de ajuste.

Finalmente, este PRNG es una forma de seguir usando los LFSR en aplicaciones criptográficas donde no se requiere una distribución uniforme. Además, como en otras aplicaciones, el uso de los LFSR en $GF(2^n)$ está motivado por el aumento de velocidad que se logra al generar n bits en lugar de 1 en cada iteración; en este caso, el uso del PRNG en $GF(2^n)$ persigue un segundo objetivo: aprovechar la relación implícita con n secuencias binarias diferentes buscando una implementación más fácil del CLT. Por lo tanto, la tasa final de generación de números con distribución Gaussiana es la misma que la tasa a la que un PRNG binario cuando se genera un bit. Sin embargo, teniendo en cuenta que los números generados tienen una longitud de $m + log_2(n)$ bits, la velocidad de generación en bits por segundo es $m + log_2(n)$ veces mayor.

Parte V

Conclusiones

Capítulo 11

Conclusiones

Para la realización de esta tesis se ha llevado a cabo una revisión detallada de las referencias bibliográficas sobre la generación de PRNG con distribución Gaussiana relacionada con la utilización de LFSR y/o con la aplicación del CLT. Se han analizado los métodos existentes en la literatura y se han mejorado en todos los aspectos deseables. En primer lugar, se ha podido comprobar que los números que se han generado se ajustan con una mayor precisión a la distribución Gaussiana de probabilidad. Por lo tanto los resultados que ellos se obtienen, poseen una mayor precisión. Las simulaciones que con ellos se han obtenido aproximan con un porcentaje mucho mayor los resultados que se obtienen de manera experimental. Además, se ha desarrollado un sistema que resulta mucho más rápido, debido a que la implementación se realiza a través de Hardware y no es necesario la implementación por Software. De esta manera, la generación se puede realizar en tiempo real con la gran ventaja que ello supone. Hemos podido observar que en otros esquemas los números se han de generar con anterioridad para posteriormente ser utilizados. Esto produce un retraso considerable a la hora de implementar las simulaciones. Por otro lado los sistemas que planteamos generan esquemas que se

pueden repetir o no según los valores iniciales que se introduzcan en el sistema. Es por esto, que el investigador podrá realizar generaciones de estos números en los que se repita o no el resultado ya que para poder realizar la comprobación de resultados este factor es de vital importancia. Debemos añadir del mismo modo, que se ha reducido de manera significativa el coste de implementación de la generación de PRNG con distribución Gaussiana. Hemos podido comprobar que en el mundo de la investigación se utilizan métodos tales como el B-M que dependen del cálculo de funciones trigonométricas. Como ya sabemos los resultados obtenidos en el cálculo de las mismas son aproximaciones que se realizan a través del desarrollo en serie de potencias de las mismas. Es por ello, que el coste computacional de la generación de tales números no posee la misma eficiencia computacional como se ha mostrado en los distintos apartados del presente documentos. Desde el punto de vista de la Criptografía, cabe destacar que el sistema de generación de PRNG con distribución Gaussiana es fundamental en el proceso de QKD. Para poder dar respuesta a esta necesidad se han de generar unas secuencias de números suficientemente grandes y con distribución Gaussiana. Es por ello, que hemos desarrollado un sistema que puede generar números de hasta 2^{64} (posteriormente estos se pueden o tipificar), o lo que es lo mismo 64 bits, pudiéndose observar que, las aproximaciones de las distribuciones de valores obtenidos de manera discreta aproximar con una gran precisión a los valores de la distribución de probabilidad continua. Así mismo, también cabe destacar que los valores obtenidos no dependen de la elección de unas buenas condiciones iniciales o no. Tal como hemos visto en las publicaciones de [14], se había de escoger un sistema adecuado en el que sólo un $\frac{1}{17}$ de las condiciones iniciales eran propicias. En contraposición a este sistema, nosotros hemos determinado un sistema basado en la elección de un polinomio primitivo. Lo que es más, hemos desarrollado un sistema para que se puede determinar si un

polinomio es o no primitivo de manera eficiente. Una vez escogido este polinomio la generación de PRNG con distribución Gaussiana, es fluida y obtiene un conjunto de valores que se aproximan de manera eficiente a la distribución Gaussiana de probabilidad. Finalmente, hemos podido observar que en un futuro no muy lejano, la arquitectura de los ordenadores va a ser completamente nueva ya que se espera una introducción de una computación cuántica. Lo que es más, se ha puesto de manifiesto que esta nueva arquitectura tecnológica llevará consigo un cambio en el concepto la seguridad de la información, puesto que se ha de desarrollar una nueva criptografía basada en este nuevo paradigma tecnológico. Con estos resultados que planteamos en esta tesis nos estamos adelantando a las necesidades que se van a plantear en el futuro y que formarán parte de nuestros días.

Parte VI

Referencias

Referencias

[1] Ahmadi, O., and Menezes, A. On the number of trace-one elements in polynomial bases for $\mathbb{F}_{2^n}$. *Designs, Codes and Cryptography 37*, 3 (2005), 493–507. 7.1

[2] Ali, F. M. S., and Sarhan, F. H. Enhancing security of vigenere cipher by stream cipher. *International Journal of Computer Applications 100*, 1 (2014), 1–4. 3.3.1

[3] Babbage, S., and Dodd, M. *The MICKEY Stream Ciphers*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 191–209. 7.1

[4] Baer, R. Groups with abelian central quotient group. *Transactions of the American Mathematical Society 44*, 3 (1938), 357–386. 2.4

[5] Bai, D., Huang, P., Zhu, Y., Ma, H., Xiao, T., Wang, T., and Zeng, G. Unidimensional continuous-variable measurement-device-independent quantum key distribution. *Quantum Information Processing 19*, 2 (2020), 1–21. 1.1

[6] Bennett, C. H., and Brassard, G. Quantum cryptography: Public key distribution and coin tossing. *arXiv preprint arXiv:2003.06557* (2020). 1.1

[7] Bohm, P. Statistical evaluation of stream cipher snow 3g. In *Constantin Brancusi University of Targu Jiu Engineering Faculty Scientific Conference with international participation,* (2008), pp. 363–366. 7.1

[8] Bourbaki, N. *Algebra II: Chapters 4-7.* Springer Science & Business Media, 2013. 2, 2.7.5

[9] Brandimarte, P. *Handbook in Monte Carlo: Applications in Financial Engineering, Risk Management, and Economics.* John Wiley & Sons, 2014. 1.1

[10] Caballero, P. *Introducción a la criptografía.* RaMa, 2002. 1.1

[11] Cardell, S. D., and Climent, J.-J. A construction of primitive polynomials over finite fields. *Linear and Multilinear Algebra 65*, 12 (2017), 2424–2431. 7.1

[12] Cid, C., Kiyomoto, S., and Kurihara, J. The rakaposhi stream cipher. In *Information and Communications Security* (Berlin, Heidelberg, 2009), S. Qing, C. J. Mitchell, and G. Wang, Eds., Springer Berlin Heidelberg, pp. 32–46. 7.1

[13] Cohn, P. M. *Basic algebra: groups, rings and fields.* Springer Science & Business Media, 2012. 2, 2.1.1, 2.2.1, 2.4, 2.7.5, 2.10

[14] Condo, C., and Gross, W. Pseudo-random gaussian distribution through optimised lfsr permutations. *Electronics Letters 51*, 25 (2015), 2098–2100. 1.1, 8.2, 8.3, 8.4.2, 8.4.2, 8.5, 8.6, 10.5, 11

[15] Cotrina, G., Peinado, A., and Ortiz, A. Generador pseudoaleatorio con distribución gaussiana implementado con lfsr. In *Actas del XXXI SIM-*

POSIUM NACIONAL DE LA UNIÓN CIENTÍFICA INTERNACIONAL DE RADIO (2016), p. 100. 9.1, 9.1, 9.2, 9.3

[16] Cotrina, G., Peinado, A., and Ortiz, A. Utilización de un generador con distribución gaussiana para aumentar la complejidad lineal de las *m*-secuencias. In *Actas de la XIV Reunión Española de Criptología y Seguridad de la Información (RECSI)* (2016), pp. 42–46. 1.3

[17] Cotrina, G., Peinado, A., and Ortiz, A. Ïdentificacion de polinomios primitivos sobre cuerpos extendidos mediante analisis de secuencias entrelazadas. In *Actas de la XV Reunión Española de Criptología y Seguridad de la Información (RECSI)* (2018), pp. 66–70. 1.3

[18] Cotrina, G., Peinado, A., and Ortiz, A. Gaussian pseudorandom number generator based on cyclic rotations of linear feedback shift registers. *Sensors 20*, 7 (2020), 2103. 1.3, 10.5

[19] Cotrina, G., Peinado, A., and Ortiz, A. Gaussian pseudorandom number generator using linear feedback shift registers in extended fields. *Mathematics 9*, 5 (2021), 556. 1.3, 10.5

[20] Cox, C. *An introduction to LTE: LTE, LTE-advanced, SAE and 4G mobile communications*. John Wiley & Sons, 2012. 1.1

[21] Daemen, J., and Rijmen, V. The block cipher rijndael. In *International Conference on Smart Card Research and Advanced Applications* (1998), Springer, pp. 277–284. 3.2

[22] Datta, D., Datta, B., and Dutta, H. S. Design and implementation of multibit lfsr on fpga to generate pseudorandom sequence number. In *2017 Devices for Integrated Circuit (DevIC)* (2017), IEEE, pp. 346–349. 1.1

[23] Delgado-Mohatar, O., and Fúster-Sabater, A. Software implementation of linear feedback shift registers over extended fields. In *International Joint Conference CISIS'12-ICEUTE´ 12-SOCO´ 12 Special Sessions* (2013), Springer, pp. 117–126. 1.1

[24] Delgado-Mohatar, O., and Fúster-Sabater, A. Software implementation of cryptographic sequence generators over extended fields, 2015. 1.1

[25] Dey, S., Nath, J., and Nath, A. An advanced combined symmetric key cryptographic method using bit manipulation, bit reversal, modified caesar cipher (sd-ree), djsa method, ttjsa method: Sja-i algorithm. *International Journal of Computer Applications 46*, 20 (2012), 46–53. 3.3.1

[26] Diamanti, E., and Leverrier, A. Distributing secret keys with quantum continuous variables: principle, security and implementations. *Entropy 17*, 9 (2015), 6072–6092. 1.1

[27] Dooley, J. F. *History of cryptography and cryptanalysis: Codes, Ciphers, and their algorithms*. Springer, 2018. 3.1

[28] Finkenzeller, K. *RFID handbook: fundamentals and applications in contactless smart cards, radio frequency identification and near-field communication*. John Wiley & Sons, 2010. 1.1

[29] Fukuda, T. Remarks on $\mathbb{Z}_p$-extensions of number fields. *Proceedings of the Japan Academy, Series A, Mathematical Sciences 70*, 8 (1994), 264–266. 2, 2.8.1

[30] Fulton, W., and Lang, S. *Riemann-roch algebra*, vol. 277. Springer Science & Business Media, 2013. 2, 2.3

[31] Fúster Sabater, A., Guía Martínez, D. d. l., Hernández Encinas, L., Montoya Vitini, F., and Muñoz Masqué, J. *Técnicas criptográficas de protección de datos*. RaMa, 2004. 1.1, 3.1, 3.3.1

[32] García Carmons, J. *Tratado de criptografía con aplicación especial al ejército*, re-editado por ministerio de defensa, 2011 ed. Sucesores de Rivadeneyra, 1894. 3.2

[33] Gehring, T., Händchen, V., Duhme, J., Furrer, F., Franz, T., Pacher, C., Werner, R. F., and Schnabel, R. Implementation of continuous-variable quantum key distribution with composable and one-sided-device-independent security against coherent attacks. *Nature communications 6*, 1 (2015), 1–7. 1.1

[34] Golomb, S. W. *Shift register sequences: secure and limited-access code generators, efficiency code generators, prescribed property generators, mathematical models*. World Scientific, 2017. 1.1, 4.1.1, 4.3.1, 4.4.1, 4.5.1, 5.3.1, 5.5.1, 5.5.2

[35] Gong, G. Theory and applications of q-ary interleaved sequences. *IEEE Transactions on Information Theory 41*, 2 (1995), 400–411. 7.2, 7.2, 7.3, 7.3, 10.2

[36] Gordon, J. A. Very simple method to find the minimum polynomial of an arbitrary nonzero element of a finite field. *Electronics Letters 12*, 25 (1976), 663–664. 7.1

[37] Greferath, M., and Schmidt, S. E. Finite-ring combinatorics and macwilliams' equivalence theorem. *Journal of Combinatorial Theory, Series A 92*, 1 (2000), 17–28. 2.8.2

[38] Gundaram, P. K., Allu, S. N., Yerukala, N., and Tentu, A. N. Rainbow tables for cryptanalysis of a5/1 stream cipher. In *Second International Conference on Networks and Advances in Computational Technologies* (2021), Springer, pp. 251–261. 1.1

[39] Hernández Encinas, L. *La criptografía.* ¿Qué sabemos de ... Editorial CSIC, 2015. 1.1, 3.2

[40] Hooley, C. On artin's conjecture. 2.7.4

[41] Hu, Y., Wu, Y., Chen, Y., Wan, G. C., and Tong, M. S. Gaussian random number generator: Implemented in fpga for quantum key distribution. *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields 32*, 3 (2019), e2554. 1.1, 8.2

[42] Hull, T. E., and Dobell, A. R. Random number generators. *SIAM review 4*, 3 (1962), 230–254. 6.4.1

[43] Kahn, D. *The Codebreakers: The Comprehensive History of Secret Communication from Ancient Times.* Scribner, 1996. 3.1, 3.3.1, 3.3.1

[44] Kang, M. Fpga implementation of gaussian-distributed pseudo-random number generator. In *6th International Conference on Digital Content, Multimedia Technology and its Applications* (2010), IEEE, pp. 11–13. 1.1, 8.2, 8.5, 10.5

[45] Kiyomoto, S., Tanaka, T., and Sakurai, K. K2: a stream cipher algorithm using dynamic feedback control. In *Secrypt* (2007), pp. 204–213. 7.1

[46] Komo, J. J., and Lam, M. S. Primitive polynomials and m-sequences over GF(q^m). *IEEE transactions on information theory 39*, 2 (1993), 643–647. 7.2, 7.2, 7.3

[47] Ladd, T. D., Goldman, J., Yamaguchi, F., Yamamoto, Y., Abe, E., and Itoh, K. M. All-silicon quantum computer. *Physical Review Letters 89*, 1 (2002), 017901. 1.1

[48] Lance, A., and Leiseboer, J. Quantum key distribution systems compared. 3.4.1

[49] Lang, S. On quasi algebraic closure. *Annals of Mathematics* (1952), 373–390. 2.8.1

[50] Laudenbach, F., Pacher, C., Fung, C.-H. F., Poppe, A., Peev, M., Schrenk, B., Hentschel, M., Walther, P., and Hübel, H. Continuous-variable quantum key distribution with gaussian modulation—the theory of practical implementations. *Advanced Quantum Technologies 1*, 1 (2018), 1800011. 1.1

[51] Lenstra, H. W. Primality testing with gaussian periods. In *International Conference on Foundations of Software Technology and Theoretical Computer Science* (2002), Springer, pp. 1–1. 7.1, 7.4

[52] Leverrier, A. Security of continuous-variable quantum key distribution via a gaussian de finetti reduction. *Physical review letters 118*, 20 (2017), 200501. 1.1, 10.3

[53] Lidl, R., and Niederreiter, H. *Introduction to finite fields and their applications*. Cambridge university press, 1994. 2, 2.9

[54] Littlewood, D. E., and Richardson, A. R. Group characters and algebra. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character 233*, 721-730 (1934), 99–141. 2, 2.2

[55] Moree, P., and Stevenhagen, P. A two-variable artin conjecture. *Journal of Number Theory 85*, 2 (2000), 291–304. 2.7.4

[56] Nagata, M. On the theory of henselian rings. *Nagoya Mathematical Journal 5* (1953), 45–57. 2.10.1

[57] Neumann, B. Groups with finite classes of conjugate subgroups. *Mathematische Zeitschrift 63*, 1 (1955), 76–96. 2.5

[58] Niederreiter, H. Linear complexity and related complexity measures for sequences. In *International Conference on Cryptology in India* (2003), Springer, pp. 1–17. 10.2

[59] Ormsby, C. Basic rules of algebra, algebraic fractions, laws of exponents, and roots/radicals, 2005. 2, 2.7.6

[60] Park, W. J., and Komo, J. J. Relationships between m-sequences over gf(q) and $gf(q^m)$. *IEEE Transactions on Information Theory 35*, 1 (1989), 183–186. 10.2

[61] Patarin, J. Asymmetric cryptography with a hidden monomial. In *Annual International Cryptology Conference* (1996), Springer, pp. 45–60. 3.2

[62] Pei, D., Salomaa, A., and Ding, C. *Chinese remainder theorem: applications in computing, coding, cryptography*. World Scientific, 1996. 2.7.7

[63] Peinado, A., and Fúster-Sabater, A. Generation of pseudorandom binary sequences by means of linear feedback shift registers (lfsrs) with dynamic feedback. *Mathematical and Computer Modelling 57*, 11-12 (2013), 2596–2604. 9.2

[64] Peinado, A., Munilla, J., and Fúster-Sabater, A. Epcgen2 pseudorandom number generators: Analysis of j3gen. *Sensors 14*, 4 (2014), 6500–6515. 1.1

[65] Peinado, A., Munilla, J., and Fúster-Sabater, A. Optimal modes of operation of pseudorandom sequence generators based on dlfsrs. *Logic Journal of the IGPL 24*, 6 (2016), 933–943. 9.2

[66] Peinado, A., Ortiz, A., and Cotrina, G. Cryptanalysis of the improvement of an authentication scheme based on the chinese remainder theorem for multicast communications. In *International Joint Conference SOCO'13-CISIS'13-ICEUTE'13* (2014), Springer, pp. 509–516. 1.1, 2.7.7

[67] Peinado, A., Ortiz, A., and Cotrina, G. Content-based image authentication using sparse features and som trajectories. *Logic Journal of the IGPL 24*, 1 (2016), 4–15. 1.1

[68] Prasad, R. S., Siripagada, A., Selvaraj, S., and Mohankumar, N. Random seeding lfsr-based trng for hardware security applications. In *Inte-*

grated Intelligent Computing, Communication and Security. Springer, 2019, pp. 427–434. 1.1

[69] Preneel, B. Cryptographic hash functions. *European Transactions on Telecommunications 5*, 4 (1994), 431–448. 3.2

[70] Reed, I. S., and Solomon, G. Polynomial codes over certain finite fields. *Journal of the society for industrial and applied mathematics 8*, 2 (1960), 300–304. 2.10.8

[71] Renner, R. Security of quantum key distribution. *International Journal of Quantum Information 6*, 01 (2008), 1–127. 1.1

[72] Rueppel, R. A. Stream ciphers. In *Analysis and Design of Stream Ciphers*. Springer, 1986, pp. 5–16. 3.2

[73] Scarani, V., Bechmann-Pasquinucci, H., Cerf, N. J., Dušek, M., Lütkenhaus, N., and Peev, M. The security of practical quantum key distribution. *Reviews of modern physics 81*, 3 (2009), 1301. 1.1

[74] Scarfone, K., Padgette, J., et al. Guide to bluetooth security. *NIST Special Publication 800*, 2008 (2008), 121. 1.1

[75] Shallit, J. Handbook of applied cryptography. by alfred j. menezes, paul c. van oorschot, and scott a. vanstone, the cryptographic imagination: Secret writing from edgar poe to the internet. by shawn james rosenheim. *The American Mathematical Monthly 106*, 1 (1999), 85–88. 1.1, 3.1.3, 3.2, 3.2.2, 3.3.1, 3.3.1

[76] Shannon, C. Communication theory of secrecy systems. *Bell System Technical Journal 28*, 4 (1949), 656–715. 3.3.1

[77] Shapiro, S. S., and Wilk, M. B. An analysis of variance test for normality (complete samples). *Biometrika 52*, 3/4 (1965), 591–611. 8.4.1

[78] Singh, P. L., Majumder, A., Chowdhury, B., Singh, R., and Mishra, N. A novel realization of reversible lfsr for its application in cryptography. In *2015 2nd International Conference on Signal Processing and Integrated Networks (SPIN)* (2015), IEEE, pp. 601–606. 1.1

[79] Singh, S. *Los códigos secretos*. Debate, 2000. 3.1

[80] Stephens, M. A. Edf statistics for goodness of fit and some comparisons. *Journal of the American Statistical Association 69*, 347 (1974), 730–737. 1.1, 8.4.1, 8.4.1

[81] Thomas, D. B. Fpga gaussian random number generators with guaranteed statistical accuracy. In *2014 IEEE 22nd Annual International Symposium on Field-Programmable Custom Computing Machines* (2014), IEEE, pp. 149–156. 1.1

[82] Travagnin, M., and Lewis, A. Quantum key distribution in-field implementations: technology assessment of qkd deployments. *EUR 29865 EN, Publications Office of the European Union, Luxembourg* (2019). 1.1

[83] UEA2&UIA, I. Specification of the 3gpp confidentiality and integrity algorithms uea2& uia2. document 2: Snow 3g specifications. version: 1.1. etsi, 2006. 1.1

[84] Ulam, S., Kuhn, H. W., Tucker, A. W., and Shannon, C. E. *John von Neumann, 1903-1957*. Harvard University Press, 2013. 1.1

[85] Vallander, S. Some remarks on axioms of probability theory. *Vestnik St. Petersburg University: Mathematics 46*, 3 (2013), 126–128. 1.1

[86] Van Laarhoven, P. J., and Aarts, E. H. Simulated annealing. In *Simulated annealing: Theory and applications*. Springer, 1987, pp. 7–15. 8.5

[87] Von Neumann, J., Kent, R., Bellinson, H., and Hart, B. The mean square successive difference. *The Annals of Mathematical Statistics 12*, 2 (1941), 153–162. 1.1

[88] Walkinshaw, N., Taylor, R., and Derrick, J. Inferring extended finite state machine models from software executions. *Empirical Software Engineering 21*, 3 (2016), 811–853. 4.4

[89] Wang, J., Huang, Z., Zheng, D., and Li, Q. An application of newton formula on the computation of finite field trace. In *2014 IEEE International Conference on Information and Automation (ICIA)* (2014), IEEE, pp. 137–140. 7.1

[90] Wang, J., and Park, J. G. Ieee 802.11 wlan based indoor positioning algorithm using weight grey prediction model. In *2020 IEEE 22nd International Conference on High Performance Computing and Communications; IEEE 18th International Conference on Smart City; IEEE 6th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)* (2020), IEEE, pp. 1158–1165. 1.1

[91] Wang, Y., and Bie, Z. A novel hardware gaussian noise generator using box-muller and cordic. In *2014 Sixth International Conference on Wireless Communications and Signal Processing (WCSP)* (2014), IEEE, pp. 1–6. 8.4.2, 10.5

[92] Weisstein, E. W. Central limit theorem. from mathworld–a wolfram web resource. 1.1, 8.4.1

[93] Wu, H. The stream cipher hc-128. In *New stream cipher designs*. Springer, 2008, pp. 39–47. 1.1

[94] Zabell, S. L. Alan turing and the central limit theorem. *The American Mathematical Monthly 102*, 6 (1995), 483–494. 8.4.1