Universidad Central "Marta Abreu" de Las Villas

Facultad de Matemática–Física-Computación

Departamento de Matemática

Homomorfismos de inmersión y matrices MDS en la Criptografía

Tesis presentada en opción al grado científico de Doctor en Ciencias Matemáticas

Oristela Cuellar Justiz

Santa Clara

2016

Universidad Central "Marta Abreu" de Las Villas

Facultad de Matemática–Física-Computación

Departamento de Matemática

Homomorfismos de inmersión y matrices MDS en la Criptografía

Tesis presentada en opción al grado científico de Doctor en Ciencias Matemáticas

Autora: M. Sc. Oristela Cuellar Justiz

oristela@uclv.edu.cu

Tutores: Dr. C. Pablo Freyre Arrozarena

pfreyre@matcom.uh.cu

Dr. C. Eberto Rodobaldo Morgado Morales

morgado@uclv.edu.cu

Santa Clara.2016

PENSAMIENTO

“El futuro de nuestra Patria tiene que ser, necesariamente, un futuro de hombres de ciencia”.

Fidel Castro

15 de enero de 1960.

AGRADECIMIENTOS

A mis padres, que ya no están por darme la vida y educarme.

A mis tutores, por impulsarme a la investigación, confiar en que podía lograrlo y recorrer junto a mí este largo camino.

A Yanet Rodríguez Sarabia decana de la facultad por su apoyo.

Al Laboratorio de Criptografía y en especial a Evaristo Madarro Capó por su trabajo en la implementación computacional de todas las ideas que fueron surgiendo.

A Guillermo Sosa Gómez por el financiamiento de las publicaciones de los artículos en el exterior.

A Yusimi Guerra Veliz por sus acertadas y valiosas sugerencias.

A mis hermanas: Maritza y Mercedes y a toda mi familia por su amor y por darme aliento para seguir adelante.

A mis amigas y amigos por ser mi sostén.

A Aniel López Pérez por su valiosa contribución a la edición de la tesis.

A mis compañeros del Departamento de Matemática, por su apoyo.

A todos los que me han ayudado en estos años y han contribuido a la conclusión de este trabajo.

A José Enrique Milord por su apoyo.

Agradecimiento especial

A mi hija Ana Beatriz, que es mi razón de vivir, por su apoyo, paciencia y comprensión.

DEDICATORIA

A mis padres: Juana Josefa y Eusebio Atanasio.

A mi hermano Rolando Epifanio.

A mi hija Ana Beatriz.

SÍNTESIS

En esta tesis se presenta un estudio de los homomorfismos de inmersión entre campos finitos y de las matrices MDS. Se exponen los fundamentos teóricos del algoritmo, para la determinación de los homomorfismos de inmersión, propuesto en este trabajo. Se realiza su análisis de complejidad y un estudio de los métodos de construcción de matrices MDS reportados en la literatura y, en particular, del algoritmo de generación aleatoria de matrices MDS propuesto por uno de los tutores de esta tesis. Se diseña un nuevo algoritmo para la generación aleatoria de dichas matrices usando otra caracterización de ellas, que mejora en cuanta a complejidad el anterior lo que hace más factible utilizar las matrices MDS como elemento llave en los algoritmos criptográficos de Cifrado en Bloques. Se formulan variantes de este algoritmo, con sus análisis de complejidad. Se concluye la tesis con la fundamentación y ejemplificación de un procedimiento que permite obtener a partir de matrices MDS con entradas en un campo, nuevas matrices MDS con entradas en otro campo, de mayor cardinalidad, en el cual se sumerge el primero.

TABLA DE CONTENIDOS

PENSAMIENTO I

AGRADECIMIENTOS II

DEDICATORIA III

SÍNTESIS IV

TABLA DE CONTENIDOS V

INDICE DE TABLAS IX

NOTACIÓN Y ACRÓNIMOS X

INTRODUCCIÓN 1

CAPÍTULO 1. MARCO TEÓRICO Y ANTECEDENTES 11

1.1 Los campos finitos 11

1.1.1 Definiciones básicas 12

1.2 Antecedentes en el tema de los homomorfismos de inmersión 17

1.2.1 Los homomorfismos entre campos finitos en los softwares profesionales GAP, MAGMA y SAGE. 23

1.2.2 Algunas aplicaciones de los homomorfismos en la criptografía 25

1.3. Antecedentes en el tema de las matrices MDS 27

1.3.1 Concepto y propiedades de las matrices MDS 28

1.3.2 Métodos para la obtención de matrices MDS 30

1.4 Conclusiones del capítulo 40

CAPÍTULO 2. ALGORITMOS PARA LA DETERMINACIÓN DE LOS HOMOMORFISMOS DE INMERSIÓN 41

2.1 Homomorfismos de inmersión 41

2.1.1 Obtención de los elementos del campo como los componentes de una sucesión recurrente lineal 43

2.1.2 Condición necesaria y suficiente para la existencia de los homomorfismos de inmersión 44

2.1.3 Ejemplo de determinación de los homomorfismos entre los campos $GF(2^3)\ y\ GF(2^6)$ 47

2.2 Relación entre los homomorfismos de inmersión y el automorfismo de *Frobenius* 53

2.2.1 Relación entre los grupos de automorfismos de ambos campos 53

2.2.2 Automorfismo de *Frobenius* del campo $GF(2^6)$ 55

2.2.3 Automorfismo de *Frobenius* de $GF(2^3)$ 56

2.3 Algoritmo para la determinación de los homomorfismos de inmersión entre campos de *Galois* 58

2.3.1 Algoritmo EVOR 58

2.3.2 Complejidad del algoritmo EVOR 61

2.3.3 Ejemplo de aplicación del algoritmo EVOR 62

2.4 Conclusiones del capítulo 64

CAPÍTULO 3. LAS MATRICES MDS Y LOS HOMOMORFISMOS DE INMERSIÓN 65

3.1 Nuevo algoritmo para la generación aleatoria de matrices MDS 66

3.1.1 Ejemplo de aplicación del nuevo algortimo. 68

3.2 Análisis de complejidad 71

3.3 Variantes del nuevo algoritmo propuesto 74

3.3.1 Variante 1 del algoritmo propuesto 74

3.3.2 Variante 2 del algoritmo propuesto 76

3.3.3.Complejidad de la variante dos del algoritmo propuesto 79

3.3.4.Ejemplos de aplicación del nuevo algoritmo 80

3.4 Uso de los homomorfismos de inmersión en la obtención de matrices MDS 83

3.4.1 Aplicación de los homomorfismos de inmersión a la obtención de matrices MDS 84

3.4.2 Ejemplos de aplicación de los homomorfismos a la obtención de matrices MDS 88

3.5 Conclusiones del capítulo 91

CONCLUSIONES Y RECOMENDACIONES 92

Conclusiones 92

Recomendaciones 94

BIBLIOGRAFÍA 95

ANEXOS 118

Anexo I: Elementos del campo $GF(2^6)$ con polinomio definidor $x^6 + x + 1$ 118

Anexo II: Otros algoritmos para la determinación de los homomorfismos de inmersión entre campos de Galois 119

II.1 Algoritmo ORISTO 119

II.2 Algoritmo ORISTO. Variante 2 122

Anexo III: Algoritmo EVOR en pseudocódigo 125

Anexo IV: Isomorfismos entre distintas variantes del campo $GF(5^2)$ 126

Anexo V: El logaritmo de *Zech´s* para el campo $GF(2^8)$ 129

Anexo VI. Nuevo algoritmo del epígrafe 3.3 134

VI.1 Pseudocódigo de la variante dos del nuevo algoritmo 134

VI.2 Precomputo de los determinantes de las submatrices de orden dos. 135

VI.3 Obtención de expresiones recursivas para el cálculo de los elementos de la matriz para la variante 1 del nuevo algoritmo 136

VI.4 Algoritmo para la determinación de la inversa de una matriz y sus variantes ... 138

VI.5 Determinantes de las submatrices de orden dos del epígrafe 3.1.1 141

INDICE DE TABLAS

Tabla I.1 Elementos del campo $GF(2^6)$ con polinomio definidor $x^6 + x + 1$

Tabla IV.1 Elementos primitivos con sus raíces

Tabla IV.2 Variantes del campo $\mathrm{GF}(5^2)$

Tabla IV.3 Isomorfismos entre distintas variantes del campo $\mathrm{GF}(5^2)$

Tabla V.1 Logaritmo de *Zech´s* para los elementos del campo $\mathrm{GF}(2^8)$

NOTACIÓN Y ACRÓNIMOS

Dado el uso generalizado del inglés como el lenguaje universal de la ciencia y de la bibliografía en este idioma, algunos términos característicos de este ámbito no han sido traducidos, por cuanto su equivalente en español pueda llevar a confusión o simplemente no existir.

Para ciertos términos, que sí han sido traducidos, se ha optado por escribir, entre paréntesis y también en cursiva, la palabra original con la que se denominan en inglés, la primera vez que son referidos en el texto.

$\mathbb{Z}_p$- grupo de restos módulo p.

$GF(p)$- campo finito o de *Galois*, de p elementos, para p primo.

$GF(q)$-campo finito o de *Galois*, de q elementos, para q potencia de un primo p.

GF(q)[x]- anillo de polinomios con coeficientes en el campo GF(q).

F/K–El campo F es una extensión del campo K.

$irr(\alpha, F)$- polinomio mínimo del elemento α sobre el campo F.

SRL- Sucesión recurrente lineal.

$gr(\alpha^i)$ –grado del elemento α^i en el campo finito $GF(q)$, que es el grado de su polinomio mínimo.

$GL_n(GF(p^n))$ – Grupo lineal general de las matrices inversibles de orden n con coeficientes sobre el campo $GF(p^n)$.

FREDIA- algoritmo de generación aleatoria de matrices MDS propuesto por Pablo Freyre, Nelson Díaz y otros.

Las matrices las denotaremos de la forma usual. La matriz de 4×4 cuando sus entradas sean elementos de campos finitos de dimensión ≥ 8 utilizaremos la notación siguiente:

$A = \big((a_{10}, a_{11}, a_{12}, a_{13}), \cdots, (a_{40}, a_{41}, a_{42}, a_{43})\ \big) -$ matriz de orden 4×4.

$(a_{i0}, a_{i1}, a_{i2}, a_{i3})$- fila i-ésima de la matriz A.

I_{n-} matriz identidad de orden n.

$C\ =\ [n;\ k;\ d]$- código lineal de longitud n, dimensión k y distancia d.

$gr\big(f(x)\big)$ –grado del polinomio $f(x)$.

INTRODUCCIÓN

La Matemática surgió de la actividad productiva de los hombres y como ciencia surge del resultado de la aplicación de las teorías Matemáticas existentes a problemas prácticos y de la elaboración de nuevos métodos para su resolución. Posee un objeto de estudio que tiene la característica de no ser un reflejo directo de la realidad objetiva debido a que tiene un carácter abstracto. De ahí que para investigar desde el punto de vista matemático cualquier objeto o fenómeno, es necesario abstraerse de todas sus cualidades particulares, excepto de aquellas que lo caracterizan directamente: su cantidad o su forma (Ribnikov 1987).

En la actualidad, la Matemática se desarrolla bajo la influencia directa de las exigencias de nuevas ramas de la ciencia y de la técnica. La teoría Matemática puede adelantarse en su desarrollo y no tener de momento una aplicación directa que luego puede aparecer o viceversa, es decir, desarrollarse por las exigencias de otras ciencias particulares. Como sucedió con el Álgebra Abstracta, en el primer caso o con la criptografía en el segundo.

Los orígenes de la criptografía se remontan a la antigüedad y es la ciencia que estudia los modos de transformación de la información con el objetivo de protegerla de las acciones de intrusos. Es una disciplina científica, técnico-ingenieril que se dedica a la elaboración, desarrollo, análisis y argumentación de la fortaleza de los medios de enmascaramiento y de protección de la información. Una de sus herramientas básicas es la matemática.

La criptografía moderna nace al mismo tiempo que las computadoras. Tras la conclusión de la Segunda Guerra Mundial, ella tuvo un desarrollo teórico importante, siendo *Claude Shannon* y sus investigaciones sobre la Teoría de la Información esenciales para su desarrollo.

La criptografía ha tenido un gran avance en el ámbito académico directamente ligado al desarrollo de las matemáticas y que ha conllevado a la resolución de múltiples problemas prácticos. Este hecho ha contribuido a que sus resultados sean utilizados por círculos más amplios de la población, siendo esencial en asuntos tan importantes como el comercio electrónico, la telefonía móvil, las nuevas plataformas de distribución de contenidos multimedia, etc.

Actualmente, tanto a nivel internacional como nacional, una de las necesidades sociales que más demanda la aplicación de la criptografía es la informatización de la sociedad. Nuestro país es subdesarrollado y bloqueado, pero su elevado potencial científico y humano le permite desarrollar las herramientas teóricas necesarias para diseñar algoritmos criptográficos propios y evaluar los criptosistemas que se aplican en el mundo globalizado actual.

Una de las herramientas matemáticas, utilizadas en el diseño de los algoritmos criptográficos modernos son las matrices MDS (*Maximal Distance Separable*), definidas sobre campos finitos y las operaciones con ellas. Las matrices MDS son aquellas para las cuales todas sus submatrices cuadradas son no singulares.

En el diseño de algoritmos de cifrado en bloques, la confusión y la difusión son dos propiedades requeridas (Shannon 1949). En la formalización de la noción de difusión perfecta en el diseño de algoritmos criptográficos existen varios enfoques. Uno de ellos usa matrices MDS, otro utiliza el concepto de multipermutación (Santis et al. 1995; Vaudenay 1994; Vaudenay 1995).

La confusión y la difusión contribuyen a enmascarar el texto claro y proteger el secreto de la información. La difusión indica que un cambio en al menos un bit del texto claro, debería modificar la mayor cantidad posible de bits en el texto cifrado. La confusión permite

ocultar la relación entre el texto cifrado y la clave secreta. (Andrade & Cedillo 2004; Salinas Rosales 2012).

Las matrices MDS garantizan que el coeficiente de difusión (*branch number*) de la transformación lineal sea máximo (Daemen 2002), obteniéndose así grandes valores de difusión en el cifrado (Barreto & Rijmen 2000a; Junod & Macchetti 2009; Xu et al. 2014).

La búsqueda de nuevos y eficientes algoritmos para la generación aleatoria de matrices MDS es un área de investigación abierta y muy actual para la Criptografía (Randal 2011).

La mayoría de los algoritmos criptográficos codifican los mensajes mediantes secuencias finitas de ceros y unos, que pueden ser consideradas como elementos de un campo finito.

La Teoría de los Campos Finitos, es una rama del Álgebra Moderna que ha tenido un desarrollo vertiginoso en los últimos 60 años debido a sus múltiples aplicaciones en los códigos algebraicos, los esquemas criptográficos, los generadores de números aleatorios, el procesamiento digital de señales, las comunicaciones, la ingeniería eléctrica, la ciencia de la computación, etc.

En la criptografía, los campos finitos se utilizan en el criptosistema *Diffie Hellman*, la firma digital el *Gamal*, en el RSA (*Rivest, Shamir and Adleman*), (Lawrence 2006), los sistemas de curvas elípticas e hiperelípticas o de *Chor-Rivest* (Van Tilborg & Jajodia 2014), los códigos de *Goppa* (McEliece 1977; Lidl & Niederreiter 1994), el secreto compartido de *Shamir*, en los cifradores de flujo *Welch-Gong* y RC4 (Goutam & Subhamoy 2012), en los cifradores de bloque AES (*Advanced Encryption Standard*), *Rijndael* (Daemen 2002; Kim et al. 2007; Nakahara & Abrahao 2009; Elumalai & Reddy 2011; Saleem et al. 2012) y RC6 (Rivest et al. 1998; Contini et al. 1998; Van Tilborg & Jajodia 2014), entre otros.

La seguridad de los algoritmos criptográficos depende del diseño del algoritmo y de la longitud de la llave. Si el algoritmo está bien diseñado, un texto cifrado sólo se podrá

descifrar si se conoce la clave. Como los ordenadores modernos son mucho más rápidos que los humanos, es importante elegir claves suficientemente largas, de manera que con la potencia de cálculo actual sea imposible probarlas todas en un tiempo razonable (Preuskchat 2014).

El avance computacional hace entonces que sea necesario que los algoritmos se implementen sobre campos finitos de mayor cardinalidad, y debido a la dificultad de realizar operaciones aritméticas sobre ellos, muchos investigadores se han dedicado a encontrar algoritmos eficientes para realizar estas operaciones. Ejemplos de ellos son *Thomas C. Bartee* y *David I. Schneider* que en (Bartee & Schneider 1963), presentan una técnica para la generación de representaciones de elementos de campos finitos que se utiliza para la implementación en paralelo de la aritmética de campos finitos de 2^n elementos; *Nils-Hassan Quittnet* que en (Quttineh 2003) trata la búsqueda de una base que minimice el número de operaciones a nivel de bits involucradas en la multiplicación sobre campos finitos; *Kenneth Wong* que en (Wong 2008) trabaja la aritmética en extensiones de campos para criptosistemas de llave pública, *Peter Montgomery* que en (Montgomery1985; Montgomery1992; Van Tilborg & Jajodia 2014) ha trabajado los algoritmos de multiplicación y reducción modular en campos finitos. Otros trabajos que pueden citarse como ejemplos son (WU 1998; Wu 2000a; Wu 2000b; Wu et al.2002).

En el empeño de encontrar algoritmos eficientes para realizar la aritmética la determinación de los homomorfismos entre campos finitos es de gran importancia en la demostración de teoremas y propiedades, y desde el punto de vista computacional al permitir realizar los cómputos de manera más eficiente. La existencia de los homomorfismos muestra que los campos asociados no son esencialmente diferentes, sino que únicamente se diferencian en

los nombres de sus elementos o en los símbolos de las operaciones. Esto implica que cualquier resultado conocido para un campo es válido para el otro (Díaz et al. 2005).

Por otro lado, como se ha señalado, muchos sistemas de cifrado en bloques modernos utilizan matrices MDS para alcanzar una alta difusión.

El concurso realizado para seleccionar el nuevo estándar criptográfico avanzado AES concluyó en octubre del 2000, con la elección del algoritmo *Rinjdael* (Daemen 2002; Kim et al. 2007; Nakahara & Abrahao 2009; Elumalai & Reddy 2011;Saleem et al. 2012).Varios de los algoritmos finalistas, incluyendo el nuevo AES, utilizaron matrices MDS. Los autores del algoritmo criptográfico *Twofish* (Schneier et al. 1998; Schneier et al. 1999), que fue uno de los finalistas, diseñaron S-cajas que varían en función de la llave y evaluaron la conveniencia de hacer variar en función de la llave la matriz MDS utilizada en la función de cifrado del algoritmo. Finalmente esta idea fue desechada por la complejidad que se le introducía al algoritmo de expansión de la llave (*key schedule*).

Esta idea de hacer variables las S-cajas en los algoritmos criptográficos en función de la llave ha sido retomada por los diseñadores de algoritmos criptográficos. Esto se puede constatar en los siguientes trabajos (Fahmy et al. 2005; Krishnamurthy & Ramaswamy 2008; Abd-ElGhafar et al. 2009; Seon 2011; Murtaza & Khan 2011; Ahmed & Elkamchouchi 2013; Arrag et al. 2013; Mahmoud et al. 2013; Freyre et al. 2015).

Las variantes dinámicas del AES en particular o de los algoritmos de Cifrado en Bloques consisten en hacer variar las funciones de estos algoritmos en función de la llave secreta. En la literatura especializada actual se observa una tendencia al diseño de variantes dinámicas del AES en las que se aprecia un marcado interés por mejorar su fortaleza a través del dinamismo de sus operaciones. Dentro de estas variantes dinámicas diseñadas del AES se pueden destacar las propuestas en (Fahmy et al. 2005; Krishnamurthy &

Ramaswamy 2008; Mohammad et al. 2009; Abd-ElGhafar et al. 2009; Mohammad et al. 2009; Kazlauskas & Kazlauskas 2009; Malik & Jong-Seon, 2011; Arrag et al. 2014; Jacob et al. 2015; Freyre et al. 2015). La construcción dinámica de las matrices MDS es uno de los criterios utilizados en el diseño de estas variantes del AES.

En (Ahmed & Elkamchouchi 2013) los autores proponen el uso de un conjunto de S-cajas y de matrices MDS dependientes de la clave. En (Murtaza & Khan 2011) se propone una transformación que comprende la obtención de matrices MDS dinámicas, basadas en la matriz MDS predefinida del AES y m-bits de llave adicional. En (Reboll 2016) la autora realiza un análisis comparativo de variantes del AES dinámico. En todos los trabajos analizados por ella se realiza la comparación entre cada una de las variantes y el AES, pero sin existir un análisis consistente de la mejora que aporta cada modificación propuesta.

En (Freyre et al. 2009a; Freyre et al. 2009b; Freyre et al. 2010; Freyre et al. 2014), los autores proponen varios algoritmos para el trabajo con matrices. Estos trabajos unidos a (Freyre et al. 2015) tienen el propósito de desarrollar un algoritmo que permita variar la matriz MDS en el proceso de cifrado en función de la llave y de lograr que la matriz inversa pueda ser calculada de manera eficiente. El algoritmo propuesto por Freyre, Díaz y otros (FREDIA) en (Freyre et al. 2014; Freyre et al. 2015), tiene la ventaja con respecto a otros reportados en la literatura que permite la generación aleatoria de una matriz MDS del conjunto de todas las matrices MDS posibles, pero su complejidad es alta porque parte de calcular los determinantes de todos las submatrices de la matriz. Es por esta razón que resulta conveniente trabajar en la idea de mejorar la eficiencia de este algoritmo.

De lo expresado anteriormente resulta el **problema científico:** la necesidad de encontrar algoritmos para la generación aleatoria de matrices MDS con una baja complejidad computacional que varíen de forma dinámica, en el proceso de cifre-descifre, en función de

la llave secreta del algoritmo criptográfico de cifrado en bloque. Se delimita como **objeto de investigación** las matrices con coeficientes sobre campos finitos y como su **campo de acción** las matrices MDS y los homomorfismos entre campos finitos.

De la definición del problema se plantea como **hipótesis de investigación** que partiendo del algoritmo FREDIA (Freyre et al. 2014; Freyre et al. 2015), se pueden diseñar otros algoritmos, con menor complejidad computacional que permitan variar dinámicamente la matriz MDS en función de la llave secreta en el algoritmo de expansión de la llave y ampliar las posibilidades de variar la matriz MDS, de forma dinámica, en el proceso de cifrado y usando los homomorfismos de inmersión entre campos finitos es posible obtener otras matrices MDS con entradas en la imagen homomorfa del campo de partida.

El **objetivo general** de esta investigación consiste en diseñar algoritmos con baja complejidad computacional para la determinación de los homomorfismos de inmersión de campos finitos y para la generación aleatoria de matrices MDS de manera que puedan ser usadas como llaves en los algoritmos de cifrado en bloques.

Para alcanzar este objetivo general fue necesario plantearse una serie de **objetivos específicos** que se pueden resumir de la manera siguiente**:**

1. Caracterizar los homomorfismos de campos finitos, las matrices MDS y los métodos para su obtención.
2. Demostrar los teoremas de condiciones necesarias y suficientes para la existencia de los homomorfismos de inmersión entre campos finitos.
3. Diseñar algoritmos para la determinación de los homomorfismos de inmersión entre campos finitos.

4. Diseñar un algoritmo para la generación aleatoria de matrices MDS con baja complejidad computacional, que permita la utilización de las matrices MDS de forma dinámica en los algoritmos de cifrado y en el proceso de cifre descifre.
5. Proponer un procedimiento para la obtención de matrices MDS a partir de otras matrices MDS, utilizando los homomorfismos de inmersión de campos finitos.

La **novedad científica** está dada en que:

1. El algoritmo para la determinación de los homomorfismos de inmersión del campo finito $GF(p^n)$ en el campo finito $GF(p^m)$, se ha diseñado combinando propiedades de las raíces de un polinomio, la distribución de los exponentes de las raíces en cosetos p-ciclotómicos módulo $p^n - 1$, el hecho de que el elemento primitivo del campo de partida se transforma en un elemento de su mismo orden en el campo de llegada. Se logra así reducir de manera significativa la complejidad del algoritmo.
2. La obtención de un nuevo algoritmo para la generación aleatoria de matrices MDS A, $A \in GL_4\left(GF(2^8)\right)$, poco consumidor de memoria, y baja complejidad que permite variar dinámicamente la matriz MDS y su inversa en función de la llave secreta en el algoritmo de expansión de la llave y ampliar las posibilidades de variar la matriz MDS, de forma dinámica, en el proceso de cifrado. Este algoritmo permite generar matrices MDS del conjunto de todas las MDS posibles, utilizando otra caracterización de las matrices MDS de tamaño 4×4 sobre campos de característica dos.
3. El uso de los homomorfismos de inmersión en la obtención de matrices MDS.

Constituyen aportes de esta investigación

Teórico: procedimientos para la determinación de los homomorfismos de inmersión entre los campos finitos $GF(p^n)$y $GF(p^m)$, donde p es un número primo cualquiera y para la obtención de nuevas matrices MDS con entradas en el campo $GF(p^m)$, en el que se sumerge $GF(p^n)$, a partir de matrices MDS con entradas en $GF(p^n)$, y viceversa.

Práctico: algoritmos para la determinación de los homomorfismos de inmersión y para la generación aleatoria de matrices MDS.

Los **métodos de investigación** empleados son:

Métodos teóricos: hipotético deductivo que permitió elaborar la hipótesis de investigación a partir de los resultados de la revisión bibliográfica y facilitó la formulación de teoremas y el desarrollo de sus demostraciones; el método inductivo deductivo que se pone de manifiesto en el estudio de las fuentes de información y de la extracción de sus regularidades y sus tendencias relacionadas con el tema de investigación y la lógica de pensamiento, cuyas interrelaciones y generalizaciones permiten la argumentación y la coherencia de las propuestas que se realizan; el método analítico-sintético que se evidenció en el análisis de casos particulares de los homomorfismos entre campos finitos y de la detección de regularidades que llevaron a la formulación de teoremas y sus demostraciones; el histórico-lógico que permitió el análisis de la trayectoria evolutiva de la investigación a partir de su objeto, su antecedente, desarrollo y la determinación de las tendencias actuales de la aplicación de los campos finitos y de las matrices MDS en los sistemas criptográficos.

Métodos matemáticos: demostración para demostrar los teoremas enunciados.

La tesis está estructurada en introducción, tres capítulos, conclusiones, recomendaciones bibliografía y anexos. En el primer capítulo se abordan los elementos de la Teoría de los Campos Finitos necesarios para la comprensión y lectura de la tesis y el estado del arte de

los homomorfismos de inmersión y de las matrices MDS. El segundo y tercer capítulos de la tesis resumen los resultados del trabajo de investigación de la autora. En el segundo capítulo se dan los fundamentos matemáticos de los algoritmos para la determinación de los homomorfismos entre campos finitos obtenidos en el transcurso de la investigación, la versión final del algoritmo con su análisis de complejidad. En el tercer capítulo se diseña por la autora un nuevo algoritmo para la generación aleatoria de matrices MDS y se proponen variantes del nuevo algoritmo de generación de matrices MDS. El capítulo termina con el uso de los homomorfismos de inmersión en la obtención de matrices MDS. La tesis consta de 6 anexos que la complementan. En el anexo 1 se dan los elementos del campo $GF(2^6)$ a partir de un polinomio primitivo específico. El anexo II está dedicado a los otros algoritmos para la determinación de los homomorfismos de inmersión. El anexo III contiene el algoritmo EVOR en pseudocódigo. En el anexo IV se presenta un ejemplo de determinación de los isomorfismos entre distintas variantes del campo $GF(5^2)$. En el anexo V contiene la tabla del logaritmo de *Zech´s* para el campo $GF(2^8)$. El anexo VI brinda información complementaria relacionada con el nuevo algoritmo de generación de matrices MDS que se presenta en el epígrafe 3.1.

CAPÍTULO 1. MARCO TEÓRICO Y ANTECEDENTES

Este capítulo está dedicado al marco teórico de la investigación y al estado del arte de los homomorfismos de inmersión y las matrices MDS. En él se caracterizan los campos finitos y los homomorfismos entre ellos, las matrices MDS y sus métodos de obtención. El capítulo está dividido en tres epígrafes: el primero dedicado a aspectos de los campos finitos necesarios para la comprensión y lectura de la tesis, el segundo al estado del arte de los homomorfismos de inmersión y el tercero a los métodos fundamentales de obtención de las matrices MDS.

1.1 Los campos finitos

La teoría de los campos finitos tiene sus orígenes en los siglos XVII y XVIII, con el estudio de una clase especial de campos finitos, los llamados campos primos, gracias a los aportes de matemáticos tan reconocidos como *Pierre de Fermat* (1601-1665), *Leonhard Euler* (1707-1783), *Joseph-Louis Lagrange* (1736-1813) y *Adrien-Marie Legendre* (1752-1833).

La teoría general sobre los campos finitos comienza con los trabajos de *Carl F. Gauss* (1777-1855) y *Evariste Galois* (1811-1832) a principios del siglo XIX. Los trabajos de Gauss se conocieron después de su muerte y los trabajos de *Galois* marcan el principio de los campos finitos o campos de *Galois*. El siguiente paso en la construcción de los campos finitos fue dado por *Richard Dedekind* en 1857, quien caracterizó los campos finitos de orden p^n como anillos de clases de restos $GF(p)[x]/(f(x))$, donde f es un polinomio

irreducible de grado n sobre $GF(p)$. Finalmente, *Eliakim Moore* en 1893 probó que el campo finito debe tener p^n elementos, donde p es un número primo. Desde finales del siglo XIX toda la estructura de los campos finitos es conocida. El libro de *Dickson* de 1901 ya tiene todos los elementos importantes de su estructura.

1.1.1 Definiciones básicas

Las definiciones, teoremas y propiedades que se enuncian en este subepígrafe se pueden encontrar en (Lidl & Niederreiter 1998). Puede consultarse también (Fraleigh 2006; Mullen & Panario 2013).

Definición 1.1: Un campo es un conjunto F no vacío con dos operaciones internas $+$ y . donde $(F,+)$ y $(F\backslash\{0\},.)$ son grupos conmutativos y se cumple la ley distributiva del producto con respecto a la suma. O sea para todo $a,b,c \in F$

$$a\cdot(b+c)=a\cdot b+a\cdot c;\ (b+c)\cdot a=b\cdot a+c\cdot a$$

Es decir, un campo es un anillo conmutativo y unitario en el que todo elemento distinto de cero posee inverso.

La característica de un campo F es el menor entero positivo n tal que $nr=0$ para todo r en F, se dice que F tiene característica n.

Un campo finito no es más que un campo con un número finito de elementos y se prueba que su característica es necesariamente un número primo p.

Teorema 1.1: En un campo F de característica p y k natural, se cumple que

$(a+b)^{p^k}=a^{p^k}+b^{p^k}$, $(a-b)^{p^k}=a^{p^k}-b^{p^k}$, para todo $a,b\in F$.

Definición 1.2: Sea $F[x]$ el anillo de polinomios con coeficientes en el campo F. Un polinomio $f\in F[x]$ se dice irreducible sobre F si f tiene grado positivo y no puede descomponerse como el producto de dos polinomios de $F[x]$, ambos de grado positivo.

Como $F[x]$ es un dominio de integridad, con la propiedad adicional de que se cumple la división con resto, todo polinomio $f \in F[x]$ se puede descomponer de manera única, salvo el orden de los factores, como producto de polinomios irreducibles, por lo cual se dice que $F[x]$ es un anillo factorial.

Definición 1.3: Sí Fes un campo y $K \subseteq F, K$ un subcampo de F. Entonces diremos que F es una extensión de K, y lo denotaremos por F/K. Si $K \neq F$ decimos que K es un subcampo propio de F. Un campo que no contiene subcampos propios se denomina campo primo

Un campo *GF(q)* de característica p se dice que tiene a $GF(p)$ como subcampo primo, ya que éste se sumerge homomórficamente en aquel.

Teorema 1.2: Sea $f \in GF(q)[x\,]$. Para que el anillo de restos $GF(q)[x]/\langle \mathrm{f} \rangle$ sea un campo es necesario y suficiente que el polinomio f sea irreducible sobre $\mathrm{GF}(q)$.

Si f es un polinomio mónico irreducible de grado m, el anillo de restos $GF(q)[x]/\langle f \rangle$ es el campo $GF(q^m)$, que se llama extensión algebraica de grado m de $GF(q)$, (y se obtiene al adjuntarle a $GF(q)$ una raíz del polinomio irreducible f).

Este campo está conformado por los polinomios de $GF(q)$[x] de grado menor que m , por tanto tiene q^m elementos. Sea α una raíz de f, se puede representar $GF(q^m)$ como el conjunto de las expresiones polinómicas en α de grado $\leq m-1$ con coeficientes en $GF(q)$. En este caso la raíz α se llama elemento definitorio de $GF(q^m)$. Es decir,

$$GF(q^m) = \{c_{m-1}\alpha^{m-1} + \cdots + c_1\alpha + c_0 |\ \forall i, c_i \in GF(q)\ \}$$

brinda una forma de representar los elementos del campo $GF(q^m)$ mediante una raíz de un polinomio f irreducible sobre $GF(q)$, a este f se le llama polinomio característico (polinomio definidor) de la extensión $GF(q^m)$, *de* $GF(q)$.

Otra forma de ver la extensión $GF(q^m)$ es como espacio vectorial de dimensión m sobre $GF(q)$. La representación que se dio permite tomar al conjunto $\{1, \alpha, \alpha^2, \cdots, \alpha^{m-1}\}$ (donde α es un elemento definitorio de $GF(q^m)$) como una base de $GF(q^m)$. A esta base se le llama base polinomial. Los elementos de $GF(q^m)$ quedarán representados entonces en la forma $(c_0, c_1, \ldots, c_{m-1})$, donde para todo $i, c_i \in GF(q)$.

A continuación se enuncian algunos teoremas importantes sin demostraciones que pueden verse en (Lidl & Niederreiter 1994; Mullen & Panario 2013).

Teorema1.3: Sea $\mathbb{F}$ un campo finito que contiene como subcampo a $GF(q)$. Entonces $\mathbb{F}$ tiene q^m elementos, para algún $m \in \mathbb{N}$.

Del teorema anterior se deduce una característica esencial de los campos finitos.

Teorema 1.4: Sea $\mathbb{F}$ un campo finito de característica p, entonces $\mathbb{F}$ tiene p^n elementos, para algún $n \in \mathbb{N}$.

Teorema 1.5: Dado el campo finito $GF(q)$ con $q = p^m$ elementos, todo subcampo de $GF(q)$ tendrá entonces p^n elementos, donde n es un divisor de m.

Definición 1.4: Sean los campos F y K, donde F es una extensión de K. Si un polinomio $f \in K[x]$ tiene todas sus raíces en F *y* F es la menor extensión de K con esta característica, se dice que F es el campo de descomposición de f sobre K, o que f se descompone en F.

Teorema 1.6: Sea K un campo y f un polinomio de grado positivo en $K[x]$, entonces existe un campo de descomposición de f sobre K.

Dos campos de descomposición de f sobre K son isomorfos, bajo un isomorfismo que deja fijos los elementos de K y deja invariante al conjunto de las raíces de f.

Teorema 1.7: Si $f \in GF(q)[x]$ es un polinomio irreducible sobre $GF(q)$ de grado m, entonces f tiene una raíz α en $GF(q^m)$. Es más, todas las raíces de f se hallan en $GF(q^m)$ y están dadas por los m elementos $\alpha, \alpha^q, \ldots, \alpha^{q^{m-1}}$ de $GF(q^m)$.

Teorema 1.8: Si $f \in GF(q)[x]$ es irreducible sobre $GF(q)$ de grado m, entonces $GF(q^m)$ es su campo de descomposición. Si f no es irreducible sobre $GF(q)$, su campo de descomposición es el menor campo en que se descomponen todos los factores irreducibles de f.

Definición 1.5: Los elementos $\alpha, \alpha^q, \ldots, \alpha^{q^{m-1}}$ de una extensión $GF(q^m)$ de $GF(q)$ se llaman conjugados con α con respecto a $GF(q)$.

Teorema 1.9: Los elementos conjugados de un $\alpha \in GF(q)$ con respecto a cualquier subcampo de $GF(q)$ tienen un mismo orden en el grupo multiplicativo $GF(q)^*$.

Teorema 1.10: Todo elemento $a \in GF(q)$ cumple que $a^q = a$.

Teorema 1.11: El grupo multiplicativo de un campo $GF(q)$ (denotado por $GF(q)^*$) es cíclico.

Definición 1.6: Un generador del grupo cíclico $GF(q)^*$ se llama elemento primitivo de $GF(q)$.

Teorema 1.12: Toda extensión $GF(q^m)$ de un campo $GF(q)$ se puede considerar como una extensión de $GF(q)$ al adjuntarle un solo elemento. A este tipo de extensión se le llama extensión simple.

Definición 1.7: Si $\alpha \in GF(q^m)$ es algebraico sobre $GF(q)$, entonces el polinomio mónico unívocamente determinado $f \in GF(q)[x]$, de menor grado que tiene a α como raíz se denomina polinomio mínimo de α sobre $GF(q)$. Diremos que el grado de α sobre $GF(q)$ es el grado de f.

Teorema 1.13: El polinomio mínimo de un elemento $\alpha \in GF(q^m)$ está dado por

$$\prod_{i=0}^{d-1}(x - \alpha^{q^i})$$

y su grado d es un divisor de m .

Definición 1.8: Sea $GF(q^m)$ una extensión de $GF(q)$ cuyo elemento definitorio α es raíz de un polinomio f irreducible sobre $GF(q)$ y de grado m. Si todo elemento de $GF(q^m)^*$ se puede expresar como una potencia de α (equivalentemente, α es un elemento primitivo de $GF(q^m)$) se dice entonces que f es un polinomio primitivo**.**

Al ser $GF(q^m)^*$ un grupo cíclico, siempre existe en él un elemento primitivo, y, por consiguiente, existe un polinomio primitivo.

Esto permite plantear $GF(q^m)^* = \{\alpha^k | \ 0 \leq k \leq q^m - 1 \}$, siempre que α sea raíz de un polinomio primitivo sobre $GF(q)$.

Definición 1.9: El automorfismo de *Frobenius* f de un campo finito *F* de característica p es la función que eleva cada elemento x de F a su p- ésima potencia, o sea $f(x) = x^p$.

Sea E una extensión del campo *F* y se considera al conjunto de todos los automorfismos de campo E/F, esto es los isomorfismos φ del campo E en sí mismo tales que $\varphi(x) = x$, $\forall x \in F$. Este conjunto con la operación de composición de funciones forma un grupo G denotado por $Aut(E/F)$, que es el grupo de las funciones F-lineales, de E en sí mismo, visto E como un F-espacio vectorial. Si E/F es una extensión de *Galois*, entonces G es llamado grupo de *Galois* de la extensión y se denota por $Gal(E/F)$ (Wilson 2000; Stewart 2015). Cada automorfismo del campo F es una potencia de f. Por lo tanto f es un generador para el grupo de *Galois* de F con respecto a su campo primo y una potencia apropiada de f es un generador del grupo de *Galois* de F sobre su subcampo primo.

1.2 Antecedentes en el tema de los homomorfismos de inmersión

Definición 1.10: Un homomorfismo de campos no es más que un homomorfismo de anillos.

Este es siempre inyectivo, porque su núcleo, siendo un ideal propio, tiene que ser cero.

A continuación se aborda el tratamiento que le dan a los homomorfismos de inmersión en la literatura consultada. Para la conformación de este epígrafe se revisaron varios materiales entre artículos, capítulos de libros, libros y páginas de INTERNET.

H, W. Lenstra en su artículo "*Finding Isomorphisms between Finite Fields*" (Lenstra 1991) analiza la existencia de algoritmos de tiempo polinomial para la determinación de los isomorfismos de campos finitos. En su análisis utiliza el término de información explícita. Considera que un campo finito está dado explícitamente si, para algunas bases del campo sobre su subcampo primo, se conoce el producto de cualquier par de elementos base, expresados en las mismas bases. En otras palabras, sea p un número primo y n un entero positivo, entonces por datos explícitos, para campos finitos de cardinalidad p^n se entiende un sistema de n^3 elementos $(a_{ijk})_{i,j,k=1}^{n}$ del campo primo $GF(p)$ tal que $GF(p)^n$ sea un campo, con la adición y multiplicación usual de elementos de $GF(p)$ y la multiplicación determinada por $e_i e_j = \sum_{k=1}^{n} a_{ijk} e_k$, donde $e_1, e_2, \dots, e_n$ denotan la base estándar de $(\mathrm{GF(p)})^n$ sobre $\mathrm{GF(p)}$. No se conoce si existe un algoritmo de tiempo polinomial que, dados p y n, construya información explícita para un campo finito de cardinalidad p^n. Si la hipótesis generalizada de *Riemann* fuera válida, entonces este algoritmo existe (Lenstra 1991). También *V. Shoup* (Shoup 1990), ha mostrado que el problema puede ser reducido a la factorización de polinomios en una variable sobre campos finitos.

Si dos campos finitos de la misma cardinalidad están dados explícitamente, el isomorfismo será representado por medio de su matriz en las bases dadas de los campos sobre el campo principal. *Lenstra* demuestra que existe un algoritmo de tiempo polinomial que, dada información explícita para dos campos finitos de la misma cardinalidad, encuentra un isomorfismo entre ellos (Lenstra 1991).

En la tesis, no fue necesario construir esta información explícita, aunque si se utiliza la representación matricial de los homomorfismos en las bases consideradas.

En (Wilson 2000) para dos campos K y L, siendo L una extensión de K denotada por L/K, el autor analiza las inmersiones de campos y las extensiones de las inmersiones. Cuando F es un campo y $\varphi: K \to F$ es una inmersión. Si $K \subseteq L$, se dice que la inmersión $\pi: L \to F$ es una extensión de φ si $\pi|_K = \varphi$. Si el campo L es una extensión de F, entonces una inmersión de K/F en L es una inmersión de K en L que se restringe a la identidad en F. El conjunto de todas las inmersiones K/F en L es denotado $Emb(K/F, L)$.

Se utiliza la notación $Emb_{\varphi}(L, F)$ para el conjunto de todas las inmersiones $\pi: L \to F$ que extienden a φ. En el caso que φ es la inclusión de K en F se escribe $Emb_K(L, F)$ para denotar el conjunto de inmersiones sobre K.

$$Emb_{\varphi}(L, w) = \{inmersiones\ \pi: L \to w\ tal\ que\ \varphi|_K = \varphi\}$$

$$Emb_K(L, F) = \{inmersiones\ \pi: L \to F\ tal\ que\ \varphi(x) = x\ para\ x \in K\}$$

Un caso especial es cuando $F = L$ y L es una extensión de K. Entonces $Emb_k(L, L)$ es un grupo. La composición de inmersiones sobre K es una inmersión y el conjunto $Emb_k(L, L)$ tiene estructura de monoide. Si L es una extensión de dimensión finita, cada inmersión sobre K es inversible, así $Emb_k(L, L)$ es un grupo. Este grupo es el grupo de *Galois* de L sobre K, denotado por $Gal(L/K)$.

En el caso más general, cuando L no es necesariamente una extensión finita, $Emb_k(L, L)$ no siempre es un grupo, pero el subconjunto de inmersiones inversibles si lo es. En este artículo no se analizan procedimientos para la determinación de los homomorfismos de inmersión.

K.M. Lux y M. Szöke, en su artículo "*Computing Homomorphism Spaces between Modules over Finite Dimensional Algebras"* del 2002, estudian el problema de la determinación de los homomorfismos de un A-módulo M a un A-módulo N sobre un álgebra A. Ellos describen un algoritmo para el caso de que A es un álgebra finito dimensional sobre un campo finito F y ambos módulos M y N son finitamente generados como A- módulos (Lux &Szöke 2002).

Se puede decir que los campos finitos $GF(p^n)$ y $GF(p^m)$ son módulos sobre su sub-campo primo $GF(p) = \mathbb{Z}_p$. Ellos conforman una subcategoría plena de la categoría de los campos o cuerpos conmutativos, y, al mismo tiempo, una subcategoría, no plena, de la categoría de los $\mathbb{Z}_p$- módulos.

La condición de no plena está dada por el hecho de que los $\mathbb{Z}_p$-homomorfismos de módulos, es decir, las transformaciones $\mathbb{Z}_p$-lineales, no son necesariamente homomorfismos multiplicativos esto es, homomorfismos de campos. La metodología aplicada en (Lux & Szöke 2002), serviría solo para encontrar los $\mathbb{Z}_p$- homomorfismos de módulos, pero sin precisar cuáles son, además de aditivos, multiplicativos. En nuestro caso, la estructura de campo permite abordar el asunto de una manera más directa. Una herramienta importante ha sido la existencia del llamado automorfismo de *Frobenius*, el cual permite, una vez hallado un homomorfismo de inmersión, de un campo en otro, obtener los restantes por composición con las diferentes potencias del automorfismo de

Frobenius. Este automorfismo, del campo $GF(p^n)$, genera al grupo de *Galois* de este campo sobre su subcampo primo, el cual es un grupo cíclico de orden n. De aquí ha resultado que existen exactamente n maneras de realizar la inmersión de un campo en el otro.

No fue usado, ni se hizo necesario, un enfoque basado en la Teoría de representaciones de módulos, lo cual sí se hace en (Lux &Szöke 2002). Este enfoque, que sería factible, tal vez en un futuro trabajo, no se hizo necesario para el cumplimiento de los objetivos propuestos con la realización de esta tesis.

Marc Culler en su artículo "*Notes on Algebra"* (Culler 2005), trata aspectos relacionados con los isomorfismos y homomorfismos entre campos finitos y los homomorfismos de inmersión define de manera similar a *Wilson* en (Wilson 2000), las inmersiones de campo y las extensiones de estas inmersiones.

Definición 1.11: Si F y K son dos campos finitos, tales que K es una extensión de F. Una aplicación φ: F→K es llamada una inmersión si φ un es un homomorfismo no nulo de F en K (Cualquier homomorfismo no nulo de campo es inyectivo ya que un campo no tiene ningún ideal propio). Si K es una extensión del campo F y φ: F→L es una inmersión entonces, una aplicación $\hat{\varphi}: K \to L$ es llamada extensión de φ si $\hat{\varphi}|_F = \varphi$.

Para el caso que F, K y L son campos tales que $F \subseteq K \subseteq L$.

Teorema1.14: Si el campo L es una extensión del campo F, entonces una inmersión de K/F en L es una inmersión de K en L que se restringe a la identidad en F. El conjunto de todas las inmersiones K/F en L se denota por $Emb(K/F, L)$.

Cuando C es una extensión algebraicamente cerrada de F y K es una extensión finita de F , *Marc Culler* en (Culler 2005) demuestra cuántas inmersiones existen de $K/F en\ C$.

Teorema 1.15: Sea F un campo finito y sea C una extensión algebraicamente cerrada de F. Si K y L son extensiones finitas de F tales que $F \subseteq K \subseteq L$, entonces cualquier inmersión de K/F dentro de C se extiende a una inmersión de L/F en C.

Teorema 1.16: Sea F un campo y $f(x)$ un polinomio irreducible en $F[x]$. Si $\eta: F \to K$ es una inmersión de F en el campo K, el número de extensiones distintas de η a inmersiones de $F[x]/(f)$ dentro de K es igual al número de distintas raíces de $f(x)$ en K.

En (Kumar 2009) el autor aborda las inmersiones de anillo y enfoca las inmersiones de campo similar a José Luis en (Barla 2012), pero no dan procedimientos para determinar dichas inmersiones.

En (Barla 2012), el autor define los monomorfismos de cuerpos y los K - homomorfismos. Si K es un cuerpo, una extensión de cuerpos F/K, es un monomorfismo de K sobre un cuerpo F, con $K \subset F$. Si $F_1 \; y \; F_2$ son extensiones de cuerpos de K, un homomorfismo $\sigma: F_1 \to F_2$ que deja fijo todo elemento de K, es llamado K -homomorfismo. Se dice que dos extensiones $F_1 \; y \; F_2$ son K -isomórficas si existe algún K -isomorfismo: $\sigma: F_1 \to F_2$.

Los homomorfismos de inmersión pueden ser considerados como $\mathbb{Z}_p$- homomorfismos.

En (Mandal 2013) *Satya Mandal* trabaja los isomorfismos conjugados que pueden verse también en (Fraleigh 2006).

Para el campo F su clausura algebraica es denotada por $\bar{F}$. En particular $i: F \to \bar{F}$ significa que $i: F \to \bar{F}$ es algebraica y $\bar{F}$.es algebraicamente cerrado.

Teorema 1.17: Sean: $F \to F(\alpha)$, una extensión algebraica; $p(x) = irr(\alpha, F)$, un polinomio irreducible; $\bar{F}$ la clausura algebraica de F; $j: F \to \bar{F}$ la inclusión; $\beta \in \bar{F}$, raíz del polinomio $p(x)$. Entonces existe un único homomorfismo $\varphi: F(\alpha) \to \bar{F}$ tal que $\varphi(\alpha) = \beta \, , \varphi(x) = x$, $\forall \, x \in F$.

Definición 1.12: Sea E una extensión algebraica del campo F. Dos elementos $\alpha\ y\ \beta \in E$ se dicen conjugados sobre F *si* $irr(\alpha, F) = irr(\beta, F)$.

Teorema 1.18 (Teorema de los isomorfismos conjugados): Sea F un campo y $\alpha, \beta \in \bar{F}$ ($\bar{F}$ es la clausura algebraica de F). Asumiendo que $gr(\alpha) = n$. Se considera una aplicación $\psi_{\alpha,\beta}: F(\alpha) \to F(\beta)$ definida por

$\psi_{\alpha,\beta}(a_0 + a_1\alpha + \ ... \ + a_{n-1}\alpha^{n-1}) = a_0 + a_1\beta + \ ... \ + a_{n-1}\beta^{n-1}$, donde $a_i \in F$.

Entonces $\psi_{\alpha,\beta}: F(\alpha) \to F(\beta)$ es un isomorfismo si y sólo si $\alpha\ y\ \beta$ son conjugados sobre F.

En (Vajakas et al. 2014) se da un método para determinar los isomorfismos de campos finitos determinados por diferentes polinomios.

Sean F y G dos campos finitos. Un homomorfismo de campos es una función

$h: F \to G$ que satisface:

$h(a + b) = h(a) + h(b)\ , h(ab) = h(a).h(b).$ (1)

$h(0) = 0\ y\ h(1) = 1.$ (2)

Sean F y G dos campos finitos de característica p y de diferente cardinalidad, determinados por los polinomios irreducibles $f\ y\ g$, con $\alpha\ y\ \beta$ raíces de estos polinomios respectivamente. Para determinar un homomorfismo es suficiente determinar el valor de $h(\alpha)$ que satisface las igualdades (1) y (2).

Teorema 1.19: Un homomorfismo $h: F \to G$ tal que $h(\alpha) = s(\beta)$ existe si y sólo si $s(\beta)$ es raíz del polinomio f.

De acuerdo a lo analizado para determinar los homomorfismos entre campos finitos hay que hallar cuáles elementos del campo de llegada son raíces del polinomio que define el campo de partida, en otras palabras hallar las raíces del polinomio que define el campo de

partida en el campo de llegada. Esta idea se utiliza en el capítulo 2 de la tesis en el diseño del algoritmo para la determinación de los homomorfismos de inmersión.

A continuación se hace referencia al tratamiento que se le da a los homomorfismos de campos en los *softwares* profesionales GAP, MAGMA y SAGE.

1.2.1 Los homomorfismos entre campos finitos en los softwares profesionales GAP, MAGMA y SAGE.

En la biblioteca del *software* profesional GAP (Rainbolt & Gallian 2003;GAPGroup 2015) existe una sección completa dedicada a explicar las funciones del GAP para la creación y análisis de campos finitos. Este *software* soporta a lo máximo campos de tamaño (2^{16}).

Dentro de la sección de campos finitos se definen varias secciones, las cuales son: introducción de los elementos de campos finitos y cómo este *software* los devuelve (*Finite Field Elements*); las operaciones aplicables a campos finitos (*Operations for finites fields elements*); la función que evalúa si un objeto es un elemento de un campo finito (IsFFE); las funciones que dan una información básica acerca de los elementos de campos finitos (CharFFE, DegreeFFE, y OrderFFE); las funciones que dan otras representaciones de elementos de campos finitos (IntFFE y LogFFE); la función que crea un campo finito (*GaloisField*).

Aunque este *software* posee aplicaciones para campos finitos y sus elementos, en su biblioteca no se encuentra nada relacionado con los homomorfismos de inmersión a excepción del Automorfismo de *Frobenius*.

En el *software* MAGMA (Cannon 2006; Kohel 2006), al igual que en el GAP, se le dedica todo un capítulo a los campos finitos el cual está dividido en las siguientes subsecciones: representación de campos finitos; creación de estructuras; campos de extensión y

relaciones; homomorfismos; creación de elementos; elementos especiales; conversiones de secuencia; opciones especiales.

Ejemplo 1.1:

hom< F -> G | x >: FldFin, Rng -> Map

Dado un campo F, crea un homomorfismo con F como su dominio y G como su codomino.

Embed (E, F, x): FldFin, FldFin ->

Para campos finitos E y F de cardinalidad p^d y p^n, donde d divide a n, da las funciones de inmersión entre E y F.

El SAGE (*Software for Algebra and Geometry Experimentation*), es un entorno de cálculos matemáticos (MCE – *Mathematics computing enviroment*) de código abierto para llevar a cabo cálculos algebraicos, simbólicos y numéricos.

El SAGE en su biblioteca *Givaro* tiene una función que determina los homomorfismos de inmersión.

sage: from sage. rings. finite_rings. hom_finite_field_givaro import

finiteFieldHomomorphism_givaro

sage: $k.< t >= GF(3^2)$

sage: $K.< T >= GF(3^4)$

sage: f = FiniteFieldHomomorphism_givaro (Hom(k, K))

sage: g = f. section(); g

Section of Ring morphism:

From: Finite Field in t of size 3^2

To: Finite Field in T of size 3^4

Defn: $t \mid \rightarrow 2T^3 + 2T^2 + 1$

El MAGMA es un *software* propietario, y no se tiene modo de conocer los algoritmos utilizados para la determinación de estos homomorfismos de inmersión. En el caso del SAGE es de código abierto y se han hecho varios intentos de comunicación con los autores del mismo, vía correo electrónico, para conocer los algoritmos para la determinación de los homomorfismos de inmersión usados por ellos, pero no se ha recibido respuesta. Se puede destacar que ellos usan polinomios *conway* (Sawollek 1999; Joyner & Kim 2011), para definir los campos finitos. En la propuesta que se hace en la tesis se puede usar cualquier polinomio primitivo de grado n sobre $GF(p)$ para definir el campo finito $GF(p^n)$, este hecho le proporciona seguridad al algoritmo criptográfico en el cual sean usados. Si se usan los homomorfismos como un componente de un algoritmo criptográfico, el desconocimiento del polinomio utilizado para definir el campo podría proporcionar mayor seguridad o incertidumbre en la medida de que el cardinal del conjunto de polinomios primitivos a elegir sea significativamente grande.

1.2.2 Algunas aplicaciones de los homomorfismos en la criptografía

Se reportan en la literatura aplicaciones de los homomorfismos en varios esquemas de cifrado y en la Teoría de Códigos. Se hará referencia a algunas de ellas encontradas en los materiales consultados.

Los autores del artículo "*Robust Finite Field Arithmetic for Fault –Tolerant Public – Key Cryptography*" (Gabautz & Sunar 2005) utilizan los homomorfismos de inmersión para sumergir los elementos del campo en un anillo grande (redundante). Los cómputos se realizan en el anillo y el homomorfismo asegura que las operaciones sean preservadas. También utilizan el homomorfismo dado a partir de las extensiones de campo $f: GF(q^m) \to GF((q^m)^n)$, el cual a veces resulta demasiado restrictivo cuando $n > 2$.

Se puede mencionar también al cifrado completamente homomórfico, originalmente llamado homomorfismo de privacidad, propuesto en 1978 (Rivest et al. 1978), poco después de la invención del RSA (Rivest et al. 1978).

En el año 2009 *Gentry* presentó el primer sistema de cifrado completamente homomórfico y la investigación de este tipo de sistema entró en una nueva era (Gentry 2009a; Gentry 2009b; Gentry & Halevi 2011). En (Gentry 2009a), se ve la seguridad de los esquemas de cifrado homomórfico contemporáneos sobre campos numéricos ciclotómicos, de grandes dimensiones. Todavía hay problemas en este tipo de cifrado que afectan su aplicación en la práctica (Smart & Vercauteren 2010; Armknecht et al. 2012; Zhang 2013). En el año 2015 y 2016 se han publicado en la revista de la IACR (*International Association for Cryptologic Research*) varios trabajos relacionados con este tipo de cifrado (Cetin et al.2015; Armknecht et al.2015; Bogos et al. 2015; Yagisawa 2015; Tanping et al. 2015; Cheon et al. 2015; Yagisawa 2015; Yagisawa 2016a; Yagisawa 2016b; Yagisawa 2016c; Yagisawa 2016d; Arita & Nakasato 2016). En (Grigorescu et al. 2006), puede encontrarse otra aplicación de los homomorfismos en la criptografía. Con este artículo los autores comienzan un estudio sistemático de la decodificación local basada en los grupos de homomorfismos y proponen una lista eficaz de decodificadores para la clase de homomorfismos de cualquier grupo abeliano G y un grupo abeliano fijo H.

En (Lauter et al. 2010), los autores presentan un esquema práctico de firma digital para ser usado en unión con las redes de codificación. Estos esquemas de firmas son el primer ejemplo de esquemas de firmas homomórficos. Esta idea proporciona autenticación y detecta nodos maliciosos que intencionalmente adulterados están contenidos en la red. Otros esquemas de firmas que utilizan homomorfismo se pueden encontrar en (Traverso et al.2015; Derler & Slamanig 2016; Lai et al.2016).

En el artículo "*Homomorphic Encryption with Access Policies: Characterization and New Constructions"* los autores presentan una caracterización de cifrado predicado (CP) que utiliza homomorfismos de grupos (Clear & Hughes 2013).

En este epígrafe hemos hecho referencia solo a algunas de las aplicaciones de los homomorfismos que hemos encontrado en la literatura.

1.3. Antecedentes en el tema de las matrices MDS

Ya se señaló en la introducción que la confusión y difusión son dos propiedades requeridas en el diseño de los algoritmos de cifrado en bloques (Menezes 1996; Aguirre 2003; Lucena 2010). Estas dos propiedades son componentes vitales en la criptografía moderna. En la formalización de la noción de difusión perfecta uno de los enfoque usa matrices MDS. Las matrices MDS garantizan que el coeficiente de difusión (*branch number*) (Daemen 2002) de la transformación lineal sea máximo, obteniéndose así grandes valores de difusión en el cifrado (Barreto & Rijmen 2000b; Barreto & Rijmen 2000c; Junod & Macchetti 2009; Xu et al. 2014). Las matrices MDS (*Maximal Distance Separable*) tienen aplicación en el diseño de los algoritmos de cifrado en bloques, las funciones Hash y en la Teoría de Códigos, que es de donde proviene su nombre.

Las matrices MDS se usan tanto en los algoritmos de cifrado en bloques AES (Daemen 2002; Nakahara & Abrahao 2009; Elumalai & Reddy 2011; Freyre et al. 2015;), *TwoFish* (Schneier et al. 1998; Schneier et al. 1999; Freyre et al. 2015), *Shark* (Gollmann et al. 1996), *Square* (Daemen et al. 1997), *Khazad* (Barreto & Rijmen 2000c), *Clefia* (Shirai et al. 2007; Tsunoo et al. 2008; Katagi & Moriai 2011) y MDS- AES (Nakahara & Abrahao 2009), como en los algoritmos de cifrado en flujo *MUGI* (Watanabe et al. 2002; Daemen &

Rijmen 2003), y las funciones de *Hash Whirlpool* (Barreto & Rijmen 2000a; Wu et al. 2013).

Otros resultados relacionados con las matrices MDS pueden verse en (Murtaza & Ikram 2008; Krishnamurthy & Ramaswamy 2008; Nakaharo & Abrahao 2009; Choy et al. 2011; Murtaza & Ikram 2011; Sajadieh et al. 2012; Augot & Finiasz 2013; Gupta & Ray 2013a; Gupta & Ray 2013b; Gupta & Ray 2014; Kolay & Mukhopadhyay 2014; Augot & Finiasz 2014; Dehnavi et al. 2015; Sim et al. 2015).

1.3.1 Concepto y propiedades de las matrices MDS

Comenzamos este epígrafe haciendo referencia a algunos aspectos relacionados con la teoría de códigos (Mac Williams & Sloane 1977; Williams 1986; Huffman 2003; Lawrence 2006)

Definición 1.13: Un código lineal C = [n; k; d] sobre $GF(2^p)$ es un subespacio k-dimensional del espacio vectorial $(GF(2^p))^n$, donde cualesquiera dos vectores diferentes del subespacio tienen distancia de *Hamming* a lo sumo d. (Williams 1986; Barreto & Rijmen 2000a; Junod & Macchetti 2009).

Los códigos lineales pueden ser descritos a través de las siguientes matrices:

- La matriz G generadora del código lineal, la cual es una matriz de $n \times k$, cuyas columnas forman una base del subespacio vectorial C. La forma escalonada de esta matriz es $G = [I_k \; A_{k\times(n-k)}]$.
- La matriz H del código C es una matriz de $(n - k) \times k$, con la propiedad de que un vector $x \in C$ si y sólo si $Hx^T = 0$.

La Teoría de Códigos Lineales trata los problemas de determinación de la distancia de un código lineal y la construcción de códigos lineales con una distancia dada.

Teorema 1.20 (Cota de *Singleton*): Sea C un código $[n;\ k;\ d]$, entonces se cumple que: $d \leq n - k + 1$.

Un código que alcanza la cota de *Singleton* es conocido como código MDS (Williams 1986; Huffman 2003; Gupta & Ray 2013 b; Gupta & Ray 2014).

Teorema 1.21: Un código [n; k; d] con matriz generadora $G = \left[I_k\ A_{k\times(n-k)}\right]$, es MDS si y sólo si toda submatriz cuadrada de A es no singular. (Williams 1986; Huffman 2003; Lawrence 2006; Gupta & Ray 2014).

Definición1.14: Sea $\mathbb{F}$ un campo finito y $p\ y\ q$ dos números enteros. Sea $x \to M \times x$ una aplicación de $\mathbb{F}^p\ en\ \mathbb{F}^q$ definida por una matriz M de tamaño $q \times p$. Decimos que una matriz es una matriz MDS si el conjunto de todos los pares $(x, M \times x)$ es un código MDS, es decir un código lineal de dimensión p, longitud $p + q$ y distancia mínima $q + 1$. (Gupta & Ray 2014).

Caracterizaciones y algunas propiedades de las matrices MDS

Las caracterizaciones y propiedades de las matrices MDS que se exponen fueron tomadas de (Gupta & Ray 2013a; Gupta & Ray 2014).

1. Una matriz A es una matriz MDS si y sólo si todas las submatrices cuadradas de A son no singulares.
2. Cualquier matriz 2×2 de $GF(2^n)$ es MDS si y sólo si es de rango completo y todas las entradas de su matriz inversa son no nulas.
3. Cualquier matriz 3×3 de $GF(2^n)$ con todas las entradas no nulas es MDS si y sólo si es de rango completo y todas las entradas de su matriz inversa son no nulas.

4. Cualquier matriz 4×4 de $GF(2^n)$ con todas las entradas no nulas es MDS si y sólo si es de rango completo, con su matriz inversa teniendo todas las entradas no nulas y todas las submatrices de tamaño 2×2 de rango completo.
5. Todas las entradas de una matriz MDS son no nulas.
6. Si A es una matriz MDS sobre $GF(2^n)$, entonces la matriz obtenida multiplicando una fila (columna) por cualquier $c \in GF(2^n)^*$ o por la permutación de filas (columnas) de A es MDS. Además si A es MDS, su traspuesta A^T también lo es.
7. Si A es una matriz MDS sobre $GF(2^n)$, entonces la matriz cA es MDS para cualquier $c \in GF(2^n)^*$.
8. La inversa de una matriz cuadrada MDS, también es MDS.

Sea $A \in M_{mxn}$ entonces, si $m \neq n$ y $rg(A) = min\{m, n\}$, se dice que la matriz A es una matriz de rango completo. En caso contrario se dice que A no es de rango completo.

Si $m = n$ entonces si $rg(A) = n$, se dice que A es una matriz no singular o regular

1.3.2 Métodos para la obtención de matrices MDS

Se reportan en la literatura varios métodos para la obtención de matrices MDS como son: el de *Reed-Solomon* (Williams 1986; Huffman 2003); las matrices de *Cauchy* (Youssef et al 1997); la construcción de matrices MDS con máxima cantidad de unos (Junod& Vaudenay 2004); la construcción de matrices MDS a partir de una constante aleatoria (Murtaza & Ikram 2008; Malik & No 2011); la construcción de matrices MDS a partir de matrices de *Vandermonde* (Sajadieh et al. 2012; Gupta & Ray 2013a); la construcción de matrices MDS a partir de la matriz acompañante de un polinomio (Gupta & Ray 2014; Gupta & Ray 2013b); la construcción de matrices MDS a partir de matrices similares al circulante (Gupta & Ray 2014); algoritmo FREDIA para la generación aleatoria de matrices MDS de todo

$GL_4\left(GF(2^8)\right)$ (Freyre et al. 2012; Freyre et al. 2014); método para obtener matrices MDS de difusión, cuyos bloques son todos polinomios (Zhao et al. 2015).

Se presentan a continuación de forma resumida las ideas esenciales de cada uno de estas formas de obtención de matrices MDS.

Una forma de obtener matrices en $GL_n(GF(q))$, q potencia de un primo p, donde todas sus submatrices cuadradas son no singulares, es a través del algoritmo de *Reed-Solomon* para la obtención de códigos lineales MDS (Williams 1986; Huffman 2003).

Construcción de matrices MDS con máxima cantidad de unos

La construcción de matrices MDS con máxima cantidad de unos tiene el propósito de obtener matrices MDS con máxima cantidad de unos y la menor diferencia posible entre sus elementos, con el objetivo de ser implementadas eficientemente en plataformas de bajo perfil. Este método fue propuesto por *Pascual Junod* y *Serge Vaudenay* en el año 2004 (Vaudenay & Junod 2004; Ruizsanchez 2014) y trabaja con la definición de arreglos bi-regulares.

Sea K un conjunto con elemento neutro.

Definición 1.15: Se dice que un arreglo de 2×2 es un arreglo bi-regular si al menos una fila y una columna tienen dos entradas diferentes.

Se dice que un arreglo de $q \times p$ con entradas en K es un arreglo bi-regular si todos sus arreglos de 2×2 son bi-regulares.

En una matriz la bi-regularidad es una condición necesaria pero no suficiente para que la matriz sea MDS.

En este método las matrices son seleccionadas de un subconjunto pequeño del total de matrices MDS. No necesita de otros métodos para iniciar el algoritmo.

Construcción de matrices MDS dinámicas a partir de una constante aleatoria

Este es un método simple basado en la propiedad 7 del epígrafe 1.3.1 (Murtaza & Ikram 2008). Sólo se obtienen tantas matrices como el cardinal del grupo multiplicativo del campo donde están definidos sus elementos. Necesita de una matriz MDS para iniciar el algoritmo. Se puede utilizar de modo dinámico pero con un alto costo de memoria. Este método puede introducir no linealidad en la transformación MDS, sin embargo en el caso de $GF(2^8)$ el costo es muy elevado (Malik & No 2011).

Construcción de matrices MDS a partir de matrices de *Cauchy*

Definición 1.16: Dados $x_0, x_1, \cdots, x_{d-1} \in GF(2^n)$ y $y_0, y_1, \cdots, y_{d-1} \in GF(2^n)$ tales que $x_i + y_j \neq 0, \forall\ 0 \leq i, j \leq d-1$, la matriz $A = (a_{ij}),\ 0 \leq i, j \leq d-1$ donde $a_{ij} = \frac{1}{x_i + y_j}$ es llamada matriz de *Cauchy* (Youssef et al. 1997).

$$detA = \frac{\prod_{0 \leq i \leq j \leq d-1}(x_j - x_i)(y_j - y_i)}{\prod_{0 \leq i,j \leq d-1} x_i + y_j}$$

Para distintos $x_0, \cdots, x_{d-1} \in GF(2^n)$ e $y_0, \cdots, y_{d-1} \in GF(2^n)$ tales que $x_i + y_j \neq 0$, $\forall\ 0 \leq i, j \leq d-1$, $detA \neq 0$ y la matriz de *Cauchy* es no singular.

Cualquier submatriz de una matriz de *Cauchy* es una matriz de *Cauchy*.

Lema1.1: Para distintos $x_0, x_1, \cdots, x_{d-1} \in GF(2^n)$ e $y_0, y_1, \cdots, y_{d-1} \in GF(2^n)$ tales que $x_i + y_j \neq 0, \forall\ 0 \leq i, j \leq d-1$, la matriz $A = (a_{ij}),\ 0 \leq i, j \leq d-1$ donde $a_{ij} = \frac{1}{x_i + y_j}$ es una matriz MDS.

Las matrices MDS obtenidas a partir de matrices de *Cauchy* poseen un gran valor teórico, sin embargo, estos diseños simples propician que existan relaciones entre los elementos de la matriz que pudiesen hacer más efectivos algunos ataques (en la práctica esto no ha sido comprobado) (Ruizsanchez 2014). Estas matrices son seleccionadas de un subconjunto del

total de matrices MDS y el método para la obtención de las mismas no necesita de otros métodos para iniciar el algoritmo y puede ser diseñado para ser utilizado de manera dinámica.

Construcción de matrices MDS a partir de matrices de *Vandermonde*

Las matrices de *Vandermonde* proporcionan otro método de obtención de matrices MDS (Sajadieh et al. 2012; Gupta & Ray 2013a).

Definición1.17: Una matriz de *Vandermonde* $A = van_d(a_0, a_1, \cdots, a_{m-1})$ es una matriz $m \times d$ construida de $(a_0, a_1, \cdots, a_{m-1})$ donde los a_i son elementos de un campo finito.

$$A = van_d(a_0, a_1, \cdots, a_{m-1}) = \begin{pmatrix} 1 & a_0 & a_0^2 & \cdots & a_0^{d-1} \\ 1 & a_1 & a_1^2 & \cdots & a_1^{d-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & a_{m-1} & a_{m-1}^2 & \cdots & a_{m-1}^{d-1} \end{pmatrix}$$

Teorema1.22: Si $A = vand(a_0, a_1, \cdots, a_{m-1})$ y $B = vand(b_0, b_1, \cdots, b_{m-1})$ son dos matrices de *Vandermonde* con diferentes elementos $a_i \neq b_j$, entonces la matriz $A.B^{-1}$ es una matriz MDS (Gupta & Ray 2013a).

Definición 1.18: Una matriz involutiva M_{mxm} es una matriz que satisface que

$M^2 = I_m$ o sea $M = M^{-1}$.

Definición 1.19: Una matriz ortogonal M_{mxm} es una matriz que satisface que

$$M\,M^T = I_m \; o \; sea \; M^{-1} = M^T.$$

Las matrices MDS involutivas pueden ser construidas a partir de matrices de *Vandermonde*.

Teorema1.23: Si $A = vand(a_0, a_1, \cdots, a_{m-1})$ y $B = vand(b_0, b_1, \cdots, b_{m-1}) = vand(a_0 + \Delta, a_1 + \Delta, \cdots, a_{m-1} + \Delta)$ son dos matrices de *Vandermonde* donde Δ es un elemento no nulo de $GF(2^q)$. Las matrices $A\,B^{-1}$ y $B\,A^{-1}$ son involutivas y si los a_i y b_j

son $2m$ valores diferentes, entonces las matrices $A\,B^{-1}\,y\,B\,A^{-1}$ serán matrices MDS involutivas (Gupta & Ray 2013a).

En este caso sólo se obtiene un subconjunto del total de matrices MDS, se necesitan dos matrices de *Vandermonde* para iniciar el algoritmo y necesita calcular la inversa que representa un costo adicional.

Construcción de matrices MDS a partir de la matriz acompañante de un polinomio

En (Guo et al. 2011; Gupta & Ray 2013b; Gupta & Ray 2014) se define el $Serial(z_0, z_1, \cdots, z_{d-1})$ como la matriz acompañante de un polinomio $z_0 + z_1 x + z_2 x^2 + \cdots + z_{d-1} x^{d-1} + x^d$, donde $z_i \in GF(2^n)$

$$A = Serial(z_0, z_1, \cdots, z_{d-1}) = \begin{pmatrix} 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & 0 & \cdots & 1 \\ z_0 & z_1 & z_2 & z_3 & \cdots & z_{d-1} \end{pmatrix}$$

En (Gupta & Ray 2014) se estudia para qué valores de $z_0, z_1, \cdots, z_{d-1}$, la matriz A^d es MDS y los casos cuando los coeficientes $z_i \in \{1, \beta, \beta^2, \beta + 1\}$, para algún $\beta \in GF(2^n)$ que conducen a matrices MDS.

Una desventaja del método es que no se conocen todos los posibles valores $\{z_0, z_1, \cdots, z_{d-1}\}$, para los cuales la matriz $A^d, d \in \mathbb{N}$, es MDS.

Es posible encontrar valores $z_0, z_1, \cdots, z_{d-1} \in GF(2^n)$ para los cuales $A^k, k \geq d$ es MDS.

Construcción de matrices MDS a partir de matrices similares al circulante

En (Gupta & Ray 2014), los autores dan las definiciones de matriz circulante y matriz circulante del tipo I y II y a partir de ellas obtienen matrices MDS.

Definición 1.20: Una matriz dxd $\begin{pmatrix} a_0 & a_1 & \cdots & a_{d-1} \\ a_{d-1} & a_0 & \cdots & a_{d-2} \\ \vdots & \vdots & \ddots & \vdots \\ a_1 & a_2 & \cdots & a_0 \end{pmatrix}$ es llamada matriz circulante y es denotada por $Circ(a_0, a_1, \cdots, a_{d-1})$.

Definición 1.21: Una matriz similar al circulante de tipo I es una matriz $\begin{pmatrix} a & 1 \\ 1^T & A \end{pmatrix}$ de tamaño dxd donde $A = Circ(a_0, a_1, \cdots, a_{d-2})$, $1 = \underbrace{1,1,\cdots,1}_{d-1\ veces}$, los a_i y a son elementos no nulos del campo y se denota por $TypeI(a, Circ(a_0, a_1, \cdots, a_{d-1}))$.

Definición 1.22: Una matriz similar al circulante de tipo II es una matriz $\begin{pmatrix} A & A^{-1} \\ A^3 + A & A \end{pmatrix}$ de tamaño $2dx2d$ donde $A = Circ(a_0, a_1, \cdots, a_{d-1})$ y se denota por $TypeII(Circ(a_0, a_1, \cdots, a_{d-1}))$.

En (Gupta & Ray 2014), los autores estudian importantes propiedades de estas matrices, proponen matrices circulantes MDS eficientes de orden 4×4 y 8×8 y prueban que tales matrices $d \times d$ similares al circulante no pueden ser involutivas ni ortogonales. Demuestran la inexistencia de matrices MDS involutivas de $2d \times 2d$ cuando d es par. Para $d = 3$ se construyen matrices MDS involutivas de orden 6×6 que son adecuadas para redes SPN redes de sustitución permutación (S*ubstitution-Permutation Network*). (Youssef et al. 1996; Youssef et al. 1997a; Youssef et al. 1997b).

Construcción de matrices MDS a partir de matrices *Finite Field Hadamard* (FFHadamard) y de las matrices especiales de Vandermonde

En (Gupta & Ray 2013a), se definen las matrices *Finite Field Hadamard* (FFHadamard) y las matrices especiales de *Vandermonde*.

Definición 1.23: Una matriz H de tamaño $2^m \times 2^m$ es FFHadamard en $GF(2^n)$ si puede ser representada como sigue $H = \begin{pmatrix} U & V \\ V & U \end{pmatrix}$, donde las submatrices U y V son también FFHadmard.

Definición 1.24: Sea $H = (h_{ij})$ una matriz de $2^m \times 2^m$, cuya primera fila es $(x_0, x_1, \cdots, x_{2^m-1})$ y $h_{ij} = x_{i \oplus j}$; entonces H es una matriz FFHadamard y es denotada por $H = had(x_0, x_1, \cdots, x_{2^m-1})$.

Sea $G = \{x_0, x_1, \cdots, x_{2^m-1}\}$ un subgrupo aditivo de $GF(2^n)$, donde $x_0 = 0, x_i + x_j = x_{i \oplus j}$. Si $H = (h_{ij})$ es una matriz de orden $2^m \times 2^m$ sobre $GF(2^n)$, donde $h_{ij} = \frac{1}{r + x_{i \oplus j}}$ y $r \in GF(2^n) \setminus G$, entonces H es FFHadamard.

Definición 1.25: Sea G un subgrupo aditivo de $GF(2^n)$, de orden 2^m, el cual es un espacio lineal de m elementos linealmente independientes $\{x_1, x_2, \cdots, x_{2^{m-1}}\}$, tales que $x_i = \sum_{k=0}^{m-1} b_k x_{2^k}$, donde $(b_0, b_1, \cdots, b_{m-1})$ es la representación binaria de i. Una matriz $van(y_0, y_1, \cdots, y_{2^m-1})$ es llamada matriz especial de *Vandermonde* (MEV) si $y_i = r + x_i$ donde $r \in GF(2^n)$.

En (Gupta & Ray 2013a), los autores enuncian lemas que fundamentan la construcción de matrices MDS involutivas a partir de MEV que son además de involutivas FFHadamard y proponen un algoritmo para la construcción de matrices MDS de $2^m \times 2^m$ sobre $GF(2^n)$ donde $m < n$. Este algoritmo de generación de matrices MDS involutivas de tamaño $d \times d$ sobre $GF(2^n)$, donde $d = 2^m$, tiene una complejidad de $O(d^2)$, puede ser diseñado para ser usado de manera dinámica en el proceso de cifrado en función de la llave.

Otros métodos

En (Sim et al.2015; Peyrin2015) se trabaja la búsqueda de matrices MDS involutivas de bajo peso. En (Bai &Wang 2016) se trabaja la búsqueda de matrices MDS de orden 4 de bajo peso y en (Liu & Sim 2016) se obtienen matrices de bajo peso a partir de una generalización de la estructura circulante.

En (Zhao et al. 2015), los autores proponen un método para obtener una especie de matrices MDS de difusión, cuyos bloques son todos polinomios sobre el campo finito *GF(2)*, estudian un nuevo tipo de transformación que puede preservar la propiedad MDS. Por otra parte, obtienen una relación de equivalencia de este tipo de transformación y la propiedad de ser MDS que es una invariante con respecto a la cual se puede reducir considerablemente el volumen de cálculos cuando se buscan matrices MDS. Los polinomios mínimos de matrices desempeñan un papel importante en su estrategia. Estas ideas no fueron utilizadas en la propuesta que se hace en la tesis.

En estos métodos descritos se obtiene sólo un subconjunto del conjunto de todas las matrices MDS posibles por eso no fue seleccionado ninguno de ellos para nuestro trabajo.

Algoritmo FREDIA para la generación aleatoria de matrices MDS de todo $\boldsymbol{GL_4\left(GF(2^8)\right)}$

Este fue propuesto por un colectivo de investigadores cubanos y su autor principal es uno de los tutores de esta tesis (Freyre et al. 2014). Este algoritmo tiene la ventaja con respecto a los antes descritos de que las matrices MDS obtenidas, son seleccionadas sobre el conjunto de todas las MDS posibles, lo que permite utilizarlo como llave. Este algoritmo no necesita de otros métodos para iniciarse, se puede utilizar de modo dinámico y eficiente en el proceso de cifrado.

El algoritmo tiene como punto de partida los algoritmos de generación aleatoria de matrices invertibles (Freyre et al. 2010; Freyre et al. 2012), donde para todo $i\ y\ j$, $a_{i,j} \in GF(q)$. El algoritmo FREDIA es importante en el diseño del nuevo algoritmo de generación de matrices MDS que se propone en el capítulo 3. Por esta razón se describe completamente a continuación.

Entrada: Polinomios primitivos $g_1(x), g_2(x), g_3(x) \in GF(q)[x]$ arbitrarios y seleccionados a priori y que cumplen que:

$$gr(g_1(x)) = 4, gr(g_2(x)) = 3, gr(g_3(x)) = 2).$$

La matriz aleatoria $M = \begin{pmatrix} \mathrm{b}_{1,0} & \mathrm{b}_{1,1} & \mathrm{b}_{1,2} & \mathrm{b}_{1,3} \\ \mathrm{c}_{2,0} & \mathrm{b}_{2,0} & \mathrm{b}_{2,1} & \mathrm{b}_{2,2} \\ \mathrm{c}_{3,0} & \mathrm{c}_{3,1} & \mathrm{b}_{3,0} & \mathrm{b}_{3,1} \\ \mathrm{c}_{4,0} & \mathrm{c}_{4,1} & \mathrm{c}_{4,2} & \mathrm{b}_{4,0} \end{pmatrix}$, con $\mathrm{b}_{1,0}$ como incógnita.

Donde: $c_{i,j} y\ b_{k,t} \in GF(q), q$ potencia de un primo p.

$$\left(b_{k,0}, b_{k,1}, \ldots, b_{k,n-k}\right) \neq\ 0, k = \overline{1,4}, t = \overline{0,3}, i = \overline{2,4}\ y\ j = \overline{0,2}, \mathrm{b}_{1,i} \neq\ 0, \forall\ i$$

Salida: Matriz MDS $A = \begin{pmatrix} Fila_1 \\ Fila_2 \\ Fila_3 \\ Fila_4 \end{pmatrix}$

Pasos del algoritmo:

1. Cálculo de la primera fila de la matriz $A = \{a_{i,j}\}_{4x4}$.

De la primera fila se toma el valor $b_{1,0}$ como incógnita.

Entrada $(a_0, a_1, a_2\ , a_3) :=\ (1,0,0,0)$

$$\hat{a}_0 +\ \hat{a}_1 x + a_2\, x^2\ + \hat{a}_3 x^3 =$$

$$(a_0 + a_1 x + a_2\, x^2 + a_3 x^3)\left(\mathrm{b}_{1,0} + \mathrm{b}_{1,1} x + b_{1,2}\, x^2 + \mathrm{b}_{1,3} x^3\right) mod g_1(x)$$

Salida: $Fila_1 =\ (\hat{a}_0, \hat{a}_1, \hat{a}_2, \hat{a}_3) := \left(\mathrm{b}_{1,0}, \mathrm{b}_{1,1}, \mathrm{b}_{1,2}, \mathrm{b}_{1,3}\right)$.

2. Cálculo de la fila j – ésima de A, $2 \leq j \leq 4$.

Para cada $j \in [2,4]$, hacer

Entrada: $(a_0, a_1, a_2, a_3) :=$ el vector canónico que tiene un uno en la posición j – ésima y cero en las restantes posiciones.

Para i desde i = j hasta 2, hacer

$$\hat{a}_0 := a_0 + c_{i,0}a_{i-1}, \hat{a}_1 := a_1 + c_{i,1}a_{i-1}, \cdots, \hat{a}_{i-2} := a_{i-2} + c_{i,i-2}a_{i-1}$$

$$\left(a_{i-1} + a_i x + \cdots + a_3 x^{4-i}\right)\left(\mathrm{b}_{i,0} + \mathrm{b}_{i,1}x + \cdots + \mathrm{b}_{i,4-i}x^{n-i}\right) mod g_i(x)$$

$$= \hat{a}_{i-1} + \hat{a}_i x + \cdots + \hat{a}_3 x^{4-i}$$

$$(a_0, a_1, a_2, a_3) := (\hat{a}_0, \hat{a}_1, \hat{a}_2, \hat{a}_3)$$

$$(a_0 + a_1 x + a_2 x^2 + a_3 x^3)\left(\mathrm{b}_{1,0} + \mathrm{b}_{1,1}x + b_{1,2}x^2 + \mathrm{b}_{1,3}x^3\right) mod g_1(x) =$$

$$= \hat{a}_0 + \hat{a}_1 x + a_2 x^2 + \hat{a}_3 x^3.$$

Salida: $Fila_j := (\hat{a}_0, \hat{a}_1, \hat{a}_2, \hat{a}_3)$

Los elementos de la fila j-ésima se igualan a cero y se obtienen ecuaciones lineales con $\mathrm{b}_{1,0}$ como incógnita, se resuelven estas ecuaciones. Los valores que no satisfacen las ecuaciones son almacenados.

Los determinantes de todas las sub-matrices hasta de orden $i \times i, 1 \leq i \leq j$ que no fueron obtenidos en el cálculo de la fila anterior son calculados. Los determinantes se igualan a cero obteniéndose, ecuaciones lineales o no lineales con $b_{1,0}$ como incógnita. Los valores que no satisfacen estas ecuaciones son almacenados.

De los valores almacenados, se selecciona uno al azar para la incógnita y se obtiene la matriz MDS siguiendo los pasos del algoritmo, para un valor especifico de $b_{1,0}$.

3. Imprimir**:** Matriz $A = \begin{pmatrix} Fila_1 \\ Fila_2 \\ Fila_3 \\ Fila_4 \end{pmatrix}$

Observación:

Se tiene que cumplir que el número máximo posible de soluciones de las ecuaciones que se forman debe ser menor que el cardinal del grupo multiplicativo del campo $GF(q)$.

Hasta ahora se han descrito varios métodos encontrados en la literatura consultada. Se revisaron también todos los materiales de los *Latincryp, Eurocrypt y Africacrypt* de los últimos años.

1.4 Conclusiones del capítulo

En este capítulo se caracterizaron los campos finitos, los homomorfismos de inmersión y los métodos de obtención de matrices MDS. Se estudiaron los elementos teóricos relacionados con los campos finitos, el tratamiento que se le da a los homomorfismos de inmersión en la literatura, en los *softwares* profesionales, así como algunas de sus aplicaciones. De igual forma se presentaron varios métodos de obtención de matrices MDS, reportados en la literatura especializada sobre todo aquellos que pudieran ser utilizados como elemento llave. Después de analizar las diferentes formas encontradas en la literatura especializada para la generación de matrices MDS se decide tomar el algoritmo FREDIA por ser el que garantiza que genera todas las matrices MDS posibles.

CAPÍTULO 2. ALGORITMOS PARA LA DETERMINACIÓN DE LOS HOMOMORFISMOS DE INMERSIÓN

En este capítulo se abordan los fundamentos matemáticos de los algoritmos para la determinación de los homomorfismos de inmersión de campos finitos, se demuestran los teoremas de condiciones necesarias y suficientes para la existencia y la cantidad de homomorfismos de inmersión entre campos finitos. Se expone un algoritmo diseñado por la autora para la determinación de estos homomorfismos con su respectivo análisis de complejidad. El capítulo está dividido en cuatro epígrafes. Los epígrafes 2.1 y 2.2 forman parte de artículos publicados por la autora de la tesis en la Revista Ciencias Matemáticas (Cuellar & Sosa 2013; Cuellar et al. 2015b).

2.1 Homomorfismos de inmersión

Los campos $GF(p^n)$ y $GF(p^m)$, donde p es un número primo $n\ y\ m$ números naturales, son extensiones algebraicas del campo primo $GF(p)$, de grados $n\ y\ m$ respectivamente y cada uno se obtiene al adjuntarle a $GF(p)$ una raíz, primitiva, de un polinomio irreducible de grado n, o grado m, según el caso.

Si n es un divisor de m, la aplicación $h: GF(p^n) \rightarrow GF(p^m)$, es llamada una inmersión de campos si h es un homomorfismo no nulo del campo $GF(p^n)\ en\ GF(p^m)$.

Si α y β son elementos primitivos en $GF(p^n)$ y $GF(p^m)$ respectivamente sus órdenes son $p^n - 1$ y $p^m - 1$ en el grupo multiplicativo correspondiente, entonces un homomorfismo

de inmersión $h: GF(p^n) \rightarrow GF(p^m)$, debe convertir α en un elemento β^k de su mismo orden, es decir, de orden $p^n - 1$. Para ello es necesario que k sea múltiplo del entero $\frac{p^m-1}{p^n-1}$. (Este número es entero, ya que por ser n divisor de m, $p^n - 1$ divide a $p^m - 1$). La cantidad de valores que puede tomar k es igual a $\varphi\ (p^n - 1)$, siendo φ la llamada función tótem de *Euler*, que asigna a cada número natural la cantidad de números naturales menores que, y primos relativos con él.

Como un homomorfismo de campos necesariamente convierte cada elemento neutro, el aditivo y el multiplicativo, en el neutro correspondiente, convierte también cada elemento del campo primo $GF(p)$, siendo p la característica de ambos campos, en sí mismo.

Por consiguiente, el homomorfismo de inmersión h es también una aplicación lineal, o sea un homomorfismo de espacios vectoriales, considerando ambos campos como espacios vectoriales sobre su subcampo primo $GF(p)$.

De acuerdo con ello el homomorfismo puede ser representado matricialmente, tomando como bases los sistemas $\{1, \alpha, \alpha^2, \cdots, \alpha^{n-1}\}\ y\ \{1, \beta, \beta^2, \cdots, \beta^{m-1}\}$ de potencias, linealmente independientes, de los elementos primitivos $\alpha\ y\ \beta$, de los campos $GF(p^n)$ y $GF(p^m)$, respectivamente. Estas bases, que pueden considerarse como las bases canónicas de ambos $GF(p)$) - espacios vectoriales, dan lugar a matrices $m \times n$, esto es, de m filas y n columnas, representantes de los diferentes homomorfismos de inmersión de $GF(p^n)$ en $GF(p^m)$.

Denotando al entero $\frac{p^m-1}{p^n-1}$ como k_1 y como k_t a cada entero $t\ k_1$ para cada $t\ < p^n - 1$ y primo relativo con el mismo, resulta que los elementos β^{k_t} son los de orden $p^n - 1$ en el campo $GF(p^m)$. Esto significa que las funciones h_t de $GF(p^n)$ en $GF(p^m)$, definidas

como $h_t(0) = 0\ y\ h_t(\alpha^i) = \beta^{ik_t}$, para cada $i \in \{0,1,2,\cdots,p^n-2\}$, son homomorfismos multiplicativos entre los monoides $\left(GF(p^n)\right)^*$ y $(GF(p^m))^*$.

Por el llamado teorema de existencia de las aplicaciones lineales para cada función h_t existe una única aplicación lineal $f_t: GF(p^n) \to GF(p^m)$ tal que $f_t(\alpha^i) = h_t(\alpha^i)$, para todo $i \in \{0,1,2,\cdots,n-1\}$, ya que $\{1,\alpha,\alpha^2,\cdots,\alpha^{n-1}\}$ es una base del espacio de partida $GF(p^n)$, (Cuellar & Sosa 2013). Entonces, los homomorfismos de inmersión son las funciones h_t que coinciden con su aplicación lineal asociada en todo su dominio $GF(p^n)$, es decir, son las que son multiplicativas y, al mismo tiempo aditivas y $\mathbb{Z}_p$-lineales.

Con estos elementos se determinaron los isomorfismos entre dos variantes de los campos $GF(3^2)\ y\ GF(5^2)$ y los homomorfismos de inmersión entre los campos $GF(2^2)\ y\ GF(2^4)$, $GF(2^2)\ y\ GF(2^6), GF(2^3)\ y\ GF(2^6), GF(3^2)\ y\ GF(3^4), GF(5^2)\ y\ GF(5^4)$.

El análisis de los resultados obtenidos permitió detectar regularidades, formular algunas hipótesis para cuyas demostraciones fue necesario representar los elementos de un campo finito como elementos de una sucesión recurrente lineal (SRL).

2.1.1 Obtención de los elementos del campo como los componentes de una sucesión recurrente lineal

Al polinomio irreducible $p(x) = x^n - \sum_{i=0}^{n-1} c_i x^i$ se le asocia la matriz cuadrada

(Lidl & Niederreiter 1998; Cuellar & Sosa 2013), $\begin{pmatrix} 0 & 0 & \cdots & 0 & c_0 \\ 1 & 0 & \cdots & 0 & c_1 \\ 0 & 1 & \cdots & 0 & c_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & c_{n-1} \end{pmatrix}$ a la que se le llama matriz acompañante del polinomio $p(x)$.

El polinomio característico $Det\left(x\ I_n - M_{p(x)}\right)$ es precisamente $p(x)$ y representando como matrices filas a los elementos $0,1,\alpha,\alpha^2,\alpha^3,\cdots,\alpha^{n-1},\alpha^n = \sum_{i=0}^{n-1}(-c_i)\alpha^i,\alpha^{n+1},\cdots,\alpha^{p^n-2}$, del campo $GF(p^n)$, siendo α un elemento primitivo, dicha matriz convierte cada elemento no nulo en su siguiente como potencia de α, siendo las correspondientes matrices filas los componentes vectoriales de una SRL cuyo período es un divisor del número $p^n - 1$. Si el polinomio es, además de irreducible, un polinomio primitivo, entonces, el período de la sucesión es exactamente $p^n - 1$, que es el máximo posible. Esto es necesario para que el elemento α, raíz de $p(x)$, sea un generador del grupo multiplicativo $(GF(p^n))^*$.

2.1.2 Condición necesaria y suficiente para la existencia de los homomorfismos de inmersión

El teorema siguiente da una condición necesaria y suficiente para que las funciones $h_t\ y\ f_t$ sean la misma función y es un resultado de la tesis (Cuellar & Sosa 2013), usado en el diseño de los algoritmos para la determinación de los homomorfismos.

Teorema 2.1:

Sean $GF(p^n)$ y $GF(p^m)$ dos campos de Galois, siendo n un divisor de m; los elementos primitivos $\alpha \in GF(p^n)\ y\ \beta \in GF(p^m)$; la función $h_t: GF(p^n) \to GF(p^m)$ definida por: $h_t(0) = 0\ , h_t\left(\alpha^i\right) = \beta^{i\,k_t}$, donde $k_t = tk_1$, siendo $k_1 = \frac{p^m-1}{p^n-1}$ y t primo relativo con $p^n - 1$ y f_t la aplicación lineal, única $f_t : GF(p^n) \to GF(p^m)\ \ con\ \ h_t\left(\alpha^i\right) = f_t\left(\alpha^i\right)$ para todo $i \in \{0,1,2,\cdots,n-1\}$ (esto es, para el cero y para los elementos de la base $(1,\alpha,\cdots,\alpha^{n-1})$ de $GF(p^n)$). Entonces, para que $h_t\ y\ f_t$ sean la misma función es suficiente, y necesario, que se verifique la igualdad $h_t(\alpha^n) = f_t(\alpha^n)$.

Demostración:

La condición es, obviamente, necesaria. Se prueba que es suficiente:

El elemento β^{k_t} es del mismo orden que α, que es igual a $p^n - 1$. La función h_t, así definida, es obviamente un homomorfismo de monoides, entre los monoides multiplicativos $\left(GF(p^n)\right)^* y\ (GF(p^m))^*$y, restringida, es un homomorfismo de grupos entre los grupos multiplicativos cíclicos $\left(GF(p^n)\right)^* y\ (GF(p^m))^*$.

Siendo α raíz del polinomio primitivo $p(x) = x^n - \sum_{i=0}^{n-1} c_i x^i$, se tiene la igualdad

$\alpha^n = \sum_{i=0}^{n-1} c_i \alpha^i$. Los elementos del campo $GF(p^n)$ son, además del cero, los términos de la sucesión recurrente lineal $\left(\alpha^j\right)_{j=0}^{\infty}$, donde los primeros n términos son $1, \alpha\ , \alpha^2, \cdots, \alpha^{n-1}$ y para $r \geq 0$, se cumple la relación de recurrencia lineal:

$\alpha^{n+r} = \sum_{i=0}^{n-1} c_i \alpha^{j+r}$, esto es, para $j \geq n$, $\alpha^j = \sum_{i=0}^{n-1} c_i \alpha^{i+j-n}$.

Esta es la S.R.L. cuyo polinomio generador es el polinomio primitivo.

$p(x) = x^n - \sum_{i=0}^{n-1} c_i x^i$, siendo su período igual a $p^n - 1$, que es el orden del elemento primitivo α, generador del grupo multiplicativo $\left(GF(p^n)\right)^*$.

Se necesita probar que, para todo r natural se verifica la igualdad $h_t(\alpha^{n+r}) = f_t(\alpha^{n+r})$. Lo que se prueba por inducción completa, aplicado a la variable natural r.

Para $r = 1$.

$$f_t(\alpha^{n+1}) = f_t(\alpha^n\ .\alpha) = f_t\left(\left(\sum_{i=0}^{n-1} c_i\ .\alpha^i\right)\ .\alpha^1\right) = f_t\left(\sum_{i=0}^{n-1} c_i\ .\alpha^{i+1}\right),$$

por ley distributiva,

$= \sum_{i=0}^{n-1} c_i\ f_t\left(\alpha^{i+1}\right)$, por ser f_t una aplicación lineal,

$= \sum_{i=0}^{n-1} c_i\ h_t\left(\ \alpha^{i+1}\right)$, por ser $f_t\ y\ h_t$ coincidentes para todos los exponentes menores o iguales que n,

$= \sum_{i=0}^{n-1} c_i \, h_t(\alpha^i) \, . \, h_t(\alpha)$, por ser h_t multiplicativa,

$= \left(\sum_{i=0}^{n-1} c_i \, h_t(\alpha)^i\right) . \, h_t(\alpha)$, de nuevo por ley distributiva,

$= \left(\sum_{i=0}^{n-1} c_i \, f_t(\alpha)^i\right) . \, h_t(\alpha)$, de nuevo por la coincidencia de ambas para exponentes menores que n ,

$= f_t\left(\sum_{i=0}^{n-1} c_i \, (\alpha)^i\right) . \, h_t(\alpha) = f_t(\alpha^n) . \, h_t(\alpha) = h_t(\alpha^n) . \, h_t(\alpha) = h_t(\alpha^n . \alpha) = h_t(\alpha^{n+1})$, de nuevo por la coincidencia y por ser h_t multiplicativa.

Queda pues probado que $f_t(\alpha^{n+1}) = h_t(\alpha^{n+1})$.

Suponiendo ahora cierta la igualdad para todo exponente menor a $n + r$, para $r \geq 2$, se obtiene:

$f_t(\alpha^{n+r}) = f_t(\alpha^n \, . \, \alpha^r) = f_t\left(\left(\sum_{i=0}^{n-1} c_i \, . \, \alpha^i\right) . \, \alpha^r\right) = f_t\left(\sum_{i=0}^{n-1} c_i \, . \, \alpha^{i+r}\right)$, por ley distributiva,

$= \sum_{i=0}^{n-1} c_i \, f_t\left(\alpha^{i+r}\right)$, por ser f_t una aplicación lineal,

$= \sum_{i=0}^{n-1} c_i \, h_t\left(\, \alpha^{i+r}\right)$, por ser f_ty h_t coincidentes para todos los exponentes i menores que $n + r$.

$= \sum_{i=0}^{n-1} c_i \, h_t(\alpha)^i \, . \, h_t(\alpha^r)$, por ser h_t multiplicativa,

$= \left(\sum_{i=0}^{n-1} c_i \, h_t(\alpha)^i\right) . \, h_t(\alpha^r)$, de nuevo por ley distributiva,

$= \left(\sum_{i=0}^{n-1} c_i \, f_t(\alpha)^i\right) . \, h_t(\alpha^r)$, de nuevo por la coincidencia de ambas para exponentes menores que $n + r$,

$= f_t\left(\sum_{i=0}^{n-1} c_i \, (\alpha)^i\right) . \, h_t(\alpha^r) = f_t(\alpha^n) . \, h_t(\alpha^r) = h_t(\alpha^n) . \, h_t(\alpha^r) = h_t(\alpha^n . \alpha^r) = h_t(\alpha^{n+r})$, de nuevo por la coincidencia y por ser h_t multiplicativa.

Queda así probado que $f_t(\alpha^{n+r}) = h_t(\alpha^{n+r})$, para todo r natural.

Por consiguiente, ambas funciones coinciden en todo su dominio.■

El teorema 2.1 significa que, siendo las SRL $\left(f_t(\alpha^j)\right)_{j=0}^{\infty}$ y $\left(h_t(\alpha^j)\right)_{j=0}^{\infty}$, del mismo grado n, con la misma relación de recurrencia y con las primeras n componentes iguales, ambas son necesariamente iguales, en todos sus términos.

2.1.3 Ejemplo de determinación de los homomorfismos entre los campos $GF(2^3)$ y $GF(2^6)$

El campo $GF(2^3)$, se obtiene como extensión del campo $GF(2) = \{0,1\}$ mediante la adjunción de una raíz de un polinomio irreducible de grado 3.

El polinomio $x^3 + x + 1$ es irreducible sobre el campo $GF(2)$ y es además primitivo, ya que cualquiera de sus raíces es de orden $7 = 2^3 - 1$. Sea α una raíz de este polinomio y por tanto un generador del grupo multiplicativo $GF(2^3)^*$, del campo $GF(2^3)$. El campo se describe como

$$GF(2^3) = \{0,1, \alpha, \alpha^2, 1+\alpha, \alpha+\alpha^2, 1+\alpha+\alpha^2, 1+\alpha^2\}$$

El campo $GF(2^6) = GF(2)(\beta)$ donde β es una raíz del polinomio irreducible de grado 6 $x^6 + x + 1 \in GF(2)[x]$.

Los elementos del campo $GF(2^6)$ aparecen en el Anexo I**.**

Un homomorfismo inyectivo $h: GF(2^3) \to GF(2^6)$, convierte al grupo cíclico $GF(2^3)^*$ en un subgrupo del grupo cíclico $GF(2^6)^*$. Luego, h convierte al generador α en un elemento que es también de orden 7, que es el orden del grupo multiplicativo $GF(2^3)^*$, dentro del grupo $GF(2^6)^*$, que es de orden $2^6 - 1 = 63$. En el campo $GF(2^6)$ el elemento β^9 es de orden 7 y en general, elementos de orden 7 son los de la forma $\beta^{9\,t}$ donde cada t es un entero positivo que es primo relativo con 7. Es decir $t \in \{1,2,3,4,5,6\}$. Por tanto $\beta^9, \beta^{18}, \beta^{27}, \beta^{36}, \beta^{45}, \beta^{54}$ son los elementos de orden 7.

Por consiguiente, hay 6 posibilidades para definir h:

$$h_1: \alpha \to \beta^9 = \beta^3 + \beta^4 = (0,0,0,1,1,0)$$

$$h_2: \alpha \to \beta^{18} = 1 + \beta + \beta^2 + \beta^3 = (1,1,1,1,0,0)$$

$$h_3: \alpha \to \beta^{27} = \beta + \beta^2 + \beta^3 = (0,1,1,1,0,0)$$

$$h_4: \alpha \to \beta^{36} = \beta + \beta^2 + \beta^4 = (0,1,1,0,1,0)$$

$$h_5: \alpha \to \beta^{45} = 1 + \beta^3 + \beta^4 = (1,0,0,1,1,0)$$

$$h_6: \alpha \to \beta^{54} = 1 + \beta + \beta^2 + \beta^4 = (1,1,0,0,1,0)$$

Aquí, se representa cada h_t como polinomio en β de grado menor que 6 y como el vector binario de dimensión 6 asociado a la misma.

Cada función h_t tiene una aplicación lineal asociada f_t. Sean $\{1, \alpha, \alpha^2\}, \{1, \beta, \beta^2, \beta^3, \beta^4, \beta^5\}$ las bases de los campos $GF(2^3)$ y $GF(2^6)$ respectivamente. Las matrices asociadas a dichas aplicaciones lineales con respecto a estas bases son

$$M(f_1) = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad M(f_2) = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}, M(f_3) = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

$$M(f_4) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \quad M(f_5) = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, M(f_6) = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

Representando como matrices columnas a los elementos de ambos campos, se tiene:

$$GF(2^3) = \left\{ \begin{array}{c} 0 = \begin{bmatrix}0\\0\\0\end{bmatrix}, 1 = \begin{bmatrix}1\\0\\0\end{bmatrix}, \alpha = \begin{bmatrix}0\\1\\0\end{bmatrix}, \alpha^2 = \begin{bmatrix}0\\0\\1\end{bmatrix}, \alpha^3 = 1+\alpha = \begin{bmatrix}1\\1\\0\end{bmatrix}, \\ \alpha^4 = \alpha+\alpha^2 = \begin{bmatrix}0\\1\\1\end{bmatrix}, \alpha^5 = 1+\alpha+\alpha^2 = \begin{bmatrix}1\\1\\1\end{bmatrix}, \alpha^6 = 1+\alpha^2 = \begin{bmatrix}1\\0\\1\end{bmatrix} \end{array} \right\}$$

mientras que los de $GF(2^6)$ se representan según la tabla del Anexo I tomando la traspuesta de la matriz fila correspondiente a cada potencia obtenida.

Se analiza que ahora cada uno de los casos por separado y según el teorema 1, para que $f_t \; y \; h_t$ sean la misma función es suficiente, y necesario, que se verifique la igualdad $f_t(\alpha^3) = \; h_t(\alpha^3)$.

Caso 1: $h_1(\alpha) = \beta^9$

$$h_1(0) = 0 = \begin{bmatrix}0\\0\\0\\0\\0\\0\end{bmatrix}, h_1(1) = 1 = \begin{bmatrix}1\\0\\0\\0\\0\\0\end{bmatrix}, h_1(\alpha) = \beta^9 = \begin{bmatrix}0\\0\\0\\1\\1\\0\end{bmatrix},$$

$$h_1(\alpha^2) = \beta^{18} = \begin{bmatrix}1\\1\\1\\1\\0\\0\end{bmatrix}, h_1(\alpha^3) = \beta^{27} = \begin{bmatrix}0\\1\\1\\1\\0\\0\end{bmatrix}$$

La matriz, $M\ (f_1)$ actúa sobre los elementos $0 \; y \; \alpha^i, 0 \; \leq i \leq 3$ de la manera siguiente

$$\begin{bmatrix}0\\0\\0\end{bmatrix} \rightarrow \begin{bmatrix}0\\0\\0\\0\\0\\0\end{bmatrix}, \begin{bmatrix}1\\0\\0\end{bmatrix} \rightarrow \begin{bmatrix}1\\0\\0\\0\\0\\0\end{bmatrix}, \begin{bmatrix}0\\1\\0\end{bmatrix} \rightarrow \begin{bmatrix}0\\0\\0\\1\\1\\0\end{bmatrix}, \begin{bmatrix}0\\0\\1\end{bmatrix} \rightarrow \begin{bmatrix}1\\1\\1\\1\\0\\0\end{bmatrix}, \begin{bmatrix}1\\1\\0\end{bmatrix} \rightarrow \begin{bmatrix}1\\0\\0\\1\\1\\0\end{bmatrix},$$

Se aprecia que el elemento imagen, obtenido por el producto matricial es el que corresponde, como potencia de β, según la sustitución $h_1(\alpha) = \; \beta^9$ excepto en el caso de

$$h_1(\alpha^3) = \beta^{27} = \begin{bmatrix}0\\1\\1\\1\\0\\0\end{bmatrix} \neq \begin{bmatrix}1\\0\\0\\1\\1\\0\end{bmatrix} = f_1(\alpha^3)$$

Es decir, ambas funciones, la aditiva f_1 y la multiplicativa h_1, no son la misma. De aquí resulta que la sustitución $h_1: \alpha \to \beta^9$ no define un homomorfismo de campos.

Caso 2: $h_2(\alpha) = \beta^{18}$

$$h_2(\alpha^3) = \beta^{54} = \begin{bmatrix}1\\1\\1\\0\\1\\0\end{bmatrix}$$

Si se hace lo mismo con la matriz $M(f_2)$, se aprecia que la misma transforma el elemento α^3 de $GF(8)$ en un elemento de $GF(2^6)$ diferente de β^{54}.

$$\begin{bmatrix}1&1&0\\0&1&1\\0&1&1\\0&1&0\\0&0&1\\0&0&0\end{bmatrix}\begin{bmatrix}1\\1\\0\end{bmatrix} = \begin{bmatrix}0\\1\\1\\1\\0\\0\end{bmatrix} \neq \begin{bmatrix}1\\1\\1\\0\\1\\0\end{bmatrix} \therefore f_2(\alpha^3) \neq h_2(\alpha^3)$$

Al igual que en el caso 1 las funciones, la aditiva f_2 y la multiplicativa h_2 no son la misma. De aquí resulta que la sustitución $h_2: \alpha \to \beta^{18}$ tampoco define un homomorfismo de campos.

Caso 3: $h_3(\alpha) = \beta^{27}$

$$h_3(\alpha^3) = \beta^{81} = \beta^{18} = 1 + \beta + \beta^2 + \beta^3 = \begin{bmatrix}1\\1\\1\\1\\0\\0\end{bmatrix}$$

Si se hace lo mismo con la matriz $M(f_3)$, se ve que transforma el elemento α^3 de $GF(2^3)$ en un elemento de GF(2^6) igual a β^{18}.

$$\begin{bmatrix}1&0&1\\0&1&1\\0&1&1\\0&1&0\\0&0&1\\0&0&0\end{bmatrix}\begin{bmatrix}1\\1\\0\end{bmatrix}=\begin{bmatrix}1\\1\\1\\1\\0\\0\end{bmatrix}=\begin{bmatrix}1\\1\\1\\1\\0\\0\end{bmatrix}\therefore f_3(\alpha^3)=h_3(\alpha^3)$$

En este caso ambas funciones la aditiva f_3 y la multiplicativa h_3 son la misma función De aquí resulta que la sustitución h_3: $\alpha \rightarrow \beta^{27}$ define un homomorfismo de campos que inserta $GF(2^3)$ en $GF(2^6)$.

Caso 4: $h_4(\alpha)=\beta^{36}$

$$h_4(\alpha^3)=\beta^{108}=\beta^{45}=1+\beta^3+\beta^4=\begin{bmatrix}1\\0\\0\\1\\1\\0\end{bmatrix}$$

Si se hace lo mismo con la matriz $M(f_4)$, se ve que la misma transforma el elemento α^3 de $GF(2^3)$ en un elemento de $GF(2^6)$ desigual de β^{45}.

$$\begin{bmatrix}1&0&0\\0&1&0\\0&1&0\\0&0&1\\0&1&1\\0&0&0\end{bmatrix}\begin{bmatrix}1\\1\\0\end{bmatrix}=\begin{bmatrix}1\\1\\1\\0\\1\\0\end{bmatrix}\neq\begin{bmatrix}1\\0\\0\\1\\1\\0\end{bmatrix}\therefore f_4(\alpha^3)\neq h_4(\alpha^3)$$

En este caso ambas funciones f_4 y la multiplicativa h_4 no son la misma función. De aquí resulta que la sustitución h_4: $\alpha \rightarrow \beta^{36}$ no define un homomorfismo de campos.

Caso 5: $h_5(\alpha^3)=\beta^{45}$

$$h_5(\alpha^3) = \beta^{135} = \beta^9 = \beta^3 + \beta^4 = \begin{bmatrix}0\\0\\0\\1\\1\\0\end{bmatrix}$$

Si se hace lo mismo con la matriz $M(f_5)$, se ve que la misma transforma el elemento α^3 de $GF(2^3)$ en un elemento de $GF(2^6)$ igual a β^9.

$$\begin{bmatrix}1&1&0\\0&0&1\\0&0&1\\0&1&1\\0&1&1\\0&0&0\end{bmatrix}\begin{bmatrix}1\\1\\0\end{bmatrix} = \begin{bmatrix}0\\0\\0\\1\\1\\0\end{bmatrix} = \begin{bmatrix}1\\0\\0\\1\\1\\0\end{bmatrix} \therefore f_5(\alpha^3) = h_5(\alpha^3)$$

Por tanto la sustitución $h_5: \alpha \to \beta^{45}$ define un homomorfismo de campos que inserta $GF(2^3)$ en $GF(2^6)$.

Caso 6: $h_6(\alpha^3) = \beta^{54}$

$$h_6(\alpha^3) = \beta^{162} = \beta^{36} = \beta + \beta^3 + \beta^4 = \begin{bmatrix}0\\1\\0\\1\\1\\0\end{bmatrix}$$

Si se hace lo mismo con la matriz $M(f_6)$, se ve que la misma transforma el elemento α^3 de $GF(2^3)$ en un elemento de $GF(2^6)$ igual a β^{36}.

$$\begin{bmatrix}1&1&1\\0&1&0\\0&1&0\\0&0&1\\0&1&1\\0&0&0\end{bmatrix}\begin{bmatrix}1\\1\\0\end{bmatrix} = \begin{bmatrix}0\\1\\1\\0\\1\\0\end{bmatrix} = \begin{bmatrix}0\\1\\1\\0\\1\\0\end{bmatrix} \therefore f_6(\alpha^3) = h_6(\alpha^3)$$

En este caso ambas funciones f_6 y h_6 son la misma función. De aquí resulta que la sustitución $h_6: \alpha \to \beta^{54}$ define un homomorfismo de campos que inserta $GF(2^3)$ en $GF(2^6)$.

Se ha visto que el campo $GF(2^3)$, se sumerge de tres maneras diferentes en el campo $GF(2^6)$. Las funciones que representan estos homomorfismos son: $h_3(\alpha) = \beta^{27}$, $h_5(\alpha) = \beta^{47}$, $h_6(\alpha) = \beta^{54}$, aunque en los tres casos el subcampo isomorfo a $GF(2^3)$ es el mismo, pues está formado por los mismos elementos: 0,1,β^9, β^{18}, β^{27}, β^{36}, β^{45}, β^{54}.

2.2 Relación entre los homomorfismos de inmersión y el automorfismo de *Frobenius*

En el epígrafe anterior se obtuvieron las diferentes maneras de definir un homomorfismo de inmersión, del campo de *Galois* $GF(p^n)$ en el también campo de *Galois* $GF(p^m)$ siendo p un número primo m y n naturales. En este epígrafe se prueba que la cantidad de maneras de realizar la inmersión es igual al orden del grupo cíclico $Aut(GF(p^n))$, de automorfismos del campo $GF(p^n)$, el cual es de orden n y es generado por el llamado automorfismo de *Frobenius* F_p. Se ilustra el resultado anterior tomando como ejemplo las funciones de inmersión del campo $GF(2^3)$ en el campo $GF(2^6)$ (Cuellar et al. 2015b).

2.2.1 Relación entre los grupos de automorfismos de ambos campos

Se considera la función $Rest: Aut(GF(p^m)) \rightarrow Aut(GF(p^n))$ tal que: ${F_p}^t \rightarrow {B_p}^t$, la cual asigna a cada automorfismo de $GF(p^m)$, expresado como una potencia del automorfismo de *Frobenius* F_p de $GF(p^m)$, la correspondiente potencia del automorfismo de *Frobenius* B_p de $GF(p^n)$. Esto no es más que la restricción del automorfismo ${F_p}^t$ de $GF(p^m)$ al automorfismo ${B_p}^t$ del subcampo $GF(p^n)$, visto como subcampo de $GF(p^m)$. Esta función es, evidentemente, un epimorfismo, cuyo *kernel* es el subgrupo formado por los ${F_p}^t$ cuyo exponente t es múltiplo de n, siendo el orden del kernel igual al número k tal que $m = nk$. Se está en condiciones de enunciar y probar el siguiente teorema (Cuellar et al. 2015b).

Teorema 2.2: La cantidad de maneras de sumergir el campo de *Galois* $GF(p^n)$ en el también campo de *Galois* $GF(p^m)$, es igual al número n, que es el orden del grupo de automorfismos del campo $GF(p^n)$.

Demostración:

Sean $G = Aut_{GF(p)}\big(GF(p^n)\big) = \{AutomorfismoB_{p^j}: GF(p^n) \to GF(p^n)\}$.

Si $\alpha \in GF(p^n)$ entonces $\alpha \to \alpha^p$.

$B_{p^j}(\alpha) = \alpha^{p^j}$, $j = \overline{0, n-1}$.

Y $G' = Aut_{GF(p)}\big(GF(p^m)\big) = \{Automorfismo\ F_{p^j}: GF(p^m) \to GF(p^m)\}$.

Si $\beta \in GF(p^m)$ entonces $\beta \to \beta^p$. $F_{p^j}(\beta) = \beta^{p^j}$, $j = \overline{0, m-1}$.

El grupo G'es generado por el automorfismo de *Frobenius* $F_p: GF(p^m) \to GF(p^m)$ por tanto $F_{p^j} = \big(F_p\big)^j$.

Sean $h_r y\, h_s$ ambos homomorfismos de inmersión, tales que $h_r(\alpha) = \beta^{k_r}\ y\ h_s(\alpha) = \beta^{k_s}$, siendo $k_r = rk_1$, $k_S = sk_1$ y $k_1 = \frac{p^m-1}{p^n-1}$.

Se toma uno de estos automorfismos, por ejemplo para un t tal que $0 < t < m-1$ y se evalúa $F_{p^t} = \big(F_p\big)^t$ en β^{k_r}

$$F_{p^t}(\beta^{k_r}) = \Big(F_p(\beta^{k_r})\Big)^t = ((\beta^{k_r})^p)^t \Longrightarrow F_{p^t}(\beta^{k_r}) = (\beta^{k_r})^{p^t} = \beta^{k_r p^t}$$

Sea $k_r p^t = rk_1.p^t = (rp^t)k_1$

Como $r < p^n - 1$ y primo relativo con $p^n - 1$, el $mcd(rp^t, p^n - 1) = 1$, si se considera $rp^t = s$, entonces $sk_1 = k_s$ y

$$k_r p^t = rk_1.p^t = (rp^t)k_1 = sk_1 = k_s \Longrightarrow F_{p^t}(\beta^{k_r}) = (\beta^{k_r})^{p^t} = \beta^{k_r p^t} = \beta^{k_s}$$

$$F_{p^t}(\beta^{k_r}) = \Big(F_p(\beta^{k_r})\Big)^t = \beta^{k_s}$$

Se ha probado que existe un automorfismo $F_{p^t} = \left(F_p\right)^t$, del campo $GF(p^m)$ tal que $F_{p^t}(\beta^{k_r}) = \beta^{k_s}$. Esto implica que $F_{p^t} \circ h_r = h_r \circ B_{p^t} = h_s$, es decir, que, dados dos homomorfismos de inmersión, uno se obtiene del otro por composición con un automorfismo. Esto, a su vez, significa que el número de homomorfismos de inmersión es igual al número n, de automorfismos del campo $GF(p^n)$.■

En el epígrafe 2.1.3 se analizaron los posibles homomorfismos de inmersión del campo de *Galois* $GF(2^3)$ en el campo $GF(2^6)$ y se mostró que el campo de *Galois* $GF(2^3)$ se sumerge de tres maneras diferentes en el campo de *Galois* $GF(2^6)$.

$$h_3(\alpha) = \beta^{27}, h_5(\alpha) = \beta^{27}, h_6(\alpha) = \beta^{27}$$

Como estas funciones son aquellas para las cuales se cumple que coinciden con su aplicación lineal asociada $h_3 = f_3, h_5 = f_3, h_6 = f_6$, entonces se pueden considerar dadas, si se conocen las matrices que representan los homomorfismos de inmersión $M(f_3), M(f_5), M(f_6)$.

A continuación se ofrece la relación que existe entre los automorfismos de *Frobenius* y las diferentes maneras de sumergir un campo en el otro.(Cuellar et al. 2015b).

2.2.2 Automorfismo de *Frobenius* del campo $GF(2^6)$

Es fácil notar que el automorfismo F_2 de $GF(2^6)$ es el que tiene por Matriz: $M(F_2)$

$$M(F_2) = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, M(F_2^2) = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$M(F_2^3)=\begin{bmatrix}1&0&1&1&1&1\\0&0&1&0&0&1\\0&1&0&0&0&1\\0&1&0&0&1&1\\0&0&1&1&0&0\\0&0&1&0&0&1\end{bmatrix}, M(F_2^4)=\begin{bmatrix}1&1&1&1&0&0\\0&1&0&0&1&1\\0&0&0&1&0&0\\0&0&1&0&0&0\\0&1&0&0&0&0\\0&0&0&0&0&1\end{bmatrix}$$

$$M(F_2^5)=\begin{bmatrix}1&1&0&0&0&0\\0&0&1&1&0&0\\0&0&0&0&1&1\\0&1&0&0&0&0\\0&0&0&1&0&0\\0&0&0&0&0&1\end{bmatrix}, M(F_2^6)=\begin{bmatrix}1&0&0&0&0&0\\0&1&0&0&0&0\\0&0&1&0&0&0\\0&0&1&1&0&0\\0&0&0&0&1&0\\0&0&0&0&0&1\end{bmatrix}=I_6$$

Multiplicando cada una de estas matrices a la izquierda de $M(f_3),\ M(f_5)\ y\ M(f_6)$, se ve que, aplicando cada uno de los automorfismos después de aplicada la función de inmersión, h_i se obtiene que

$$F_2 \circ h_3 = h_6\,, (F_2)^2 \circ h_3 = h_5\,, (F_2)^3 \circ h_3 = h_3$$

$$(F_2)^4 \circ h_3 = h_6\,, (F_2)^5 \circ h_3 = h_5\,, (F_2)^6 \circ h_3 = h_3$$

$$F_2 \circ h_5 = h_3\,, (F_2)^2 \circ h_5 = h_6\,, (F_2)^3 \circ h_5 = h_5$$

$$(F_2)^4 \circ h_5 = h_3\,, (F_2)^5 \circ h_5 = h_6\,, (F_2)^6 \circ h_5 = h_5$$

$$F_2 \circ h_6 = h_5\,, (F_2)^2 \circ h_6 = h_3\,, (F_2)^3 \circ h_6 = h_6$$

$$(F_2)^4 \circ h_6 = h_5\,, (F_2)^5 \circ h_6 = h_3\,, (F_2)^6 \circ h_6 = h_6$$

Es decir, que la composición con el automorfismo de *Frobenius* F_2 convierte cada homomorfismo de inmersión en otro, no existiendo otros homomorfismos de inmersión salvo $h_3\,, h_5\,, h_6$. En el caso analizado la cantidad de inmersiones posibles coincide con el orden n, del grupo de automorfismos $Aut(GF(2^3))$ del campo de partida.

2.2.3 Automorfismo de *Frobenius* de $GF(2^3)$

El automorfismo B_2 de $GF(2^3)$ es el que tiene por Matriz: $M(B_2)$

$$M(B_2)=\begin{bmatrix}1&0&0\\0&0&1\\0&1&1\end{bmatrix} y\ cumple\ que\ M(B_2^2)=\begin{bmatrix}1&0&0\\0&1&1\\0&1&0\end{bmatrix} y\ M(B_2^3)=\begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix}=I_3$$

Multiplicando la matriz $M(B_2)$ a la derecha de $M(h_3), M(h_5), M(h_6)$ se ve que, aplicando la función de inmersión h_i, después de aplicado el automorfismo B_2, se obtiene que

$$M(h_3)M(B_2)=\begin{bmatrix}1&0&1\\0&1&1\\0&1&1\\0&1&0\\0&0&1\\0&0&0\end{bmatrix}\begin{bmatrix}1&0&0\\0&0&1\\0&1&1\end{bmatrix}=\begin{bmatrix}1&1&1\\0&1&0\\0&1&0\\0&0&1\\0&1&1\\0&0&0\end{bmatrix}=M(h_6)$$

$$M(h_5)M(B_2)=\begin{bmatrix}1&1&0\\0&0&1\\0&0&1\\0&1&1\\0&1&0\\0&0&0\end{bmatrix}\begin{bmatrix}1&0&0\\0&0&1\\0&1&1\end{bmatrix}=\begin{bmatrix}1&0&1\\0&1&1\\0&1&1\\0&1&0\\0&0&1\\0&0&0\end{bmatrix}=M(h_3)$$

$$M(h_6)M(B_2)=\begin{bmatrix}1&1&1\\0&1&0\\0&1&0\\0&0&1\\0&1&1\\0&0&0\end{bmatrix}\begin{bmatrix}1&0&0\\0&0&1\\0&1&1\end{bmatrix}=\begin{bmatrix}1&1&0\\0&0&1\\0&0&1\\0&1&1\\0&1&0\\0&0&0\end{bmatrix}=M(h_5)$$

lo cual significa que $h_3 \circ B_2 = h_6$, $h_5 \circ B_2 = h_3$, $h_6 \circ B_2 = h_5$. Es decir, que en este caso también ocurre que la composición con el automorfismo de *Frobenius* convierte cada homomorfismo de inmersión en otro existente, no existiendo ningún otro homomorfismo de inmersión salvo h_3 , h_5 , h_6.

Se ha corroborado que la cantidad de inmersiones posibles coincide con el orden, 3, del grupo de automorfismos $Aut(GF(2^3))$ del campo de partida.

Todos los cálculos necesarios para obtener los resultados que se presentan fueron realizados con el *Mathematica* 9.0 y con el *software* BiGFSoP[1].

2.3 Algoritmo para la determinación de los homomorfismos de inmersión entre campos de *Galois*

A partir de los resultados obtenidos en los epígrafes 2.1 y 2.2 fueron diseñados tres algoritmos ORISTO, ORISTO II y EVOR para la determinación de los homomorfismos de inmersión. Los dos primeros algoritmos pueden verse en el Anexo II. Cada nuevo algoritmo constituyó una mejora del anterior (Cuellar et al. 2015a). En este epígrafe se presenta el mejor de estos algoritmos, el EVOR para determinar los homomorfismos de inmersión del campo finito $GF(p^n)$ en el campo $GF(p^m)$ donde p es un número primo, $n\ y\ m$ números naturales.

2.3.1 Algoritmo EVOR

De acuerdo a lo expuesto en el capítulo 1 epígrafe 1.2 otra manera de determinar los homomorfismos entre campos finitos es hallar cuáles elementos del campo de llegada, son raíces del polinomio que define el campo de partida, en otras palabras hallar las raíces del polinomio que define el campo de partida, en el campo de llegada. Aunque en la literatura consultada (Vajakas et al. 2014) se utiliza este procedimiento para determinar los isomorfismos de campos finitos definidos por diferentes polinomios, él puede ser generalizado a la determinación de los homomorfismos entre campos finitos de diferente cardinalidad.

Sean F y G dos campos finitos de característica p y de diferente cardinalidad. Sean el campo F determinado por el polinomio irreducible $f\ y\ \alpha$ una raíz de dicho polinomio por tanto $f(\alpha) = 0$ y el campo G determinado por el polinomio irreducible g. Si β es una raíz de dicho polinomio $g(\beta) = 0$.

La cantidad de homomorfismos de inmersión es igual al número de raíces del polinomio irreducible que define el campo de partida. Por tanto el método de búsqueda de los homomorfismos de inmersión se reduce a la búsqueda de las raíces de f en el campo de llegada $GF(p^m)$. Aquí se necesita disponer de los algoritmos de búsqueda de raíces, la búsqueda exhaustiva puede ser útil para campos relativamente pequeños, pero para campos grandes no, ya que el cardinal del espacio de búsqueda crece exponencialmente con el incremento de la dimensión del campo sobre el cual se realiza la búsqueda. Incluso si se utilizan los algoritmos de búsqueda de raíces (Chen 1982; Lidl & Niederreiter 1998; Gathen & Panario 2001; Madarro et al. 2013), a medida que el grado del polinomio en cuestión sea mayor, el costo computacional se incrementa considerablemente.

Antes de exponer el último algoritmo diseñado, se recuerda la definición de coseto ciclotómico dada por *Golomb* en (Golomb 1982; Villalba 2000; Rani 2013). Estos cosetos son utilizados en el diseño de nuestro algoritmo.

Definición 2.1: Se denomina coseto q –ciclotómico de S módulo n al conjunto $C_s = \{s, sq, sq^2, \cdots, sq^{r-1}\}$, donde r es el menor entero positivo tal que $s\, q^r \equiv s\ (mod\ n)$.

De manera particular el coseto 2 –ciclotómico de S módulo n es el conjunto $C_s = \{s, s.2, s.2^2, \cdots, s.2^{r-1}\}$, donde r es el menor entero positivo tal que $s\ 2^r \equiv s\ (mod\ n)$.

Se propone a continuación un algoritmo híbrido diseñado por la autora que evalúa el polinomio f en busca de las raíces y usa también las ideas desarrolladas en los epígrafes anteriores de este capítulo. Esta propuesta que reduce el espacio de búsqueda de manera significativa, tiene en cuenta también que si para un polinomio f de grado n sobre un campo $GF(p)$ una de sus raíces es α, el resto de las raíces son los elementos conjugados con α con respecto al campo $GF(p)$ o sea $\alpha, \alpha^p, \alpha^{p^2}, \cdots, \alpha^{p^{n-1}}$(Lidl & Niederreiter 1998;

Mullen & Panario 2013). Los exponentes de dichas potencias pertenecen al mismo coseto p-ciclotómico módulo $p^n - 1$. Por tanto es suficiente encontrar una de dichas raíces y además a la hora de realizar la búsqueda no es necesario explorar todo el campo es necesario sólo determinar cuál de las potencias del elemento primitivo del campo de llegada cuyo exponente está asociado a uno de los cosetos líderes (el menor de los elementos de la clase) es raíz del polinomio que define el campo de partida.

También se tiene en cuenta que si el polinomio que define el campo de partida es primitivo de grado n, sus raíces son elementos de orden $p^n - 1$ y estas raíces se transforman por el homomorfismo en elementos de su mismo orden en el campo de llegada.

Algoritmo EVOR.

Entrada: Elemento primitivo β del segundo campo finito y el polinomio P que define el campo finito de partida y Q que define el campo de llegada, la dimensión n, m de los dos campos respectivamente, la característica p de los campos finitos, el conjunto CL de todos los cosetos líderes módulo $p^n - 1$, que son primos relativos con el orden del grupo multiplicativo del campo de partida.

Salida: Funciones que son homomorfismos de inmersión entre los campos finitos especificados.

1. Hacer $k := \frac{p^m - 1}{p^n - 1}$; $r := |\mathrm{CL}|$ y $s := p^n - 1$;
2. Para cada elemento $i \in \mathrm{CL}$, calcular $h_i = i.k$, hacer :
   2.1. Calcular β^{h_i} y guardar $t := \beta^{h_i}$.
   2.2. Si t es raíz del polinomio P, calcular los elementos conjugados con t con respecto a $GF(p)$ e ir al paso 3.
3. Imprimir las funciones que representan los homomorfismos de inmersión.

El algoritmo en pseudocódigo aparece en el **Anexo III.**

2.3.2 Complejidad del algoritmo EVOR

La complejidad del algoritmo depende de la complejidad del paso dos es decir, principalmente, de las operaciones exponenciación y evaluación de polinomios. De esta manera, se tiene la siguiente notación asintótica:

$$O\big(c\ \max\big((E_v);\ (E)\big)\big)$$

Donde (E) es la complejidad de exponenciación y (E_v) es la complejidad de evaluar un elemento en un polinomio y c es la cantidad de cosetos líderes chequeados hasta encontrar el primer homomorfismo, con $c < r = \frac{\varphi(p^n-1)}{n} = \frac{\varphi(\mathrm{s})}{n}$

El polinomio $P(x) \in GF(p)[x]$. La evaluación de este polinomio de grado n sobre $GF(p^m)$, se realiza mediante el método de *Horner* que tiene un costo $O(n)$ operaciones. (Das 1999; Panario 2015), mientras que la operación de exponenciación en $GF(p^m)$, tiene una complejidad de un $O(m^2 \log p \log m \log\log m.)$, usando métodos basados en la transformada rápida de *Fourier* (Gao et al. 2000; Panario 2015).

Se tiene entonces,

$$O\big(c \max\ \big((n\, m \log m \log\log m\,)\,;\ \ (m^2 \log p \log m \log\log m)\big)\big)\rightarrow$$

$$O(c(m^2 \log p \log m \log\log m))$$

El cálculo de las p-ésimas potencias del homomorfismo es una simple multiplicación del exponente por p de manera sucesiva y reducción del mismo módulo p^m-1, en caso de ser necesario.

También puede tenerse en cuenta que si $(a_0, a_1, \cdots, a_{m-1})$ son las coordenadas de un elemento $a \in GF(q) = GF(p^m)$ en una base normal $\big(\alpha, \alpha^p, \cdots, \alpha^{p^{m-1}}\big)$, entonces el

elemento a^p tiene por coordenadas $(a_{m-1}, a_0, \cdots, a_{m-2})$. La elevación a la potencia p implica simplemente una permutación circular de los coeficientes y por tanto, su costo computacional es despreciable. Si $p = 2$ es la elevación al cuadrado, la que tiene costo cero. (Huguet et al. 2012).

Este algoritmo mejora la complejidad computacional de determinar los homomorfismos de inmersión entre campos finitos con respecto a ORISTO y ORISTO II.

No fue posible comparar el algoritmo propuesto con otros reportados en la literatura. Los *softwares* profesionales MAGMA y SAGE tienen funciones que determinan los homomorfismos de inmersión entre campos finitos. Las funciones obtenidas usando el EVOR coinciden con las obtenidas aplicando estos *softwares* pero no se pudieron hacer análisis comparativos de complejidad.

2.3.3 Ejemplo de aplicación del algoritmo EVOR

Los campos $GF(2^4) = GF(2)(\alpha), GF(2^8) = GF(2)(\beta)$ y $GF(2^{16}) = GF(2)(\gamma)$ se obtienen como extensiones del campo $GF(2) = \{0,1\}$ siendo $\alpha, \beta\, y\, \gamma$ raíces de los polinomios irreducibles sobre $GF(2)$de grado 4, 8 y 16 respectivamente. Se ha tomado en este caso los polinomios primitivos sobre $GF(2)$:

$$p_1(x) = x^4 + x + 1,$$

$$p_2(x) = x^8 + x^4 + \ x^3 + x^2 + 1$$

$$p_3(x) = x^{16} + x^{15} + \ x^{14} + x^{12} + x^{10} + x^9 + x^4 + x^2 + 1\,.$$

Los cosetos líderes módulo 15 que son primos relativos con 15 son $\mathrm{CL} = \{1,7\}$.

Los cosetos líderes módulo 255 que son primos relativos con 255 son

$\mathrm{CL} = \{1,7,11,13,19,23,29,31,43,47,53,59,61,91,119,127\}$.

$$\frac{2^8-1}{2^4-1}=17\,;\,\frac{2^{16}-1}{2^8-1}=257$$

β^{17} es raíz de $p_1(x)$ en el campo $GF(2^8)$. El coseto 2-ciclotómico módulo 15 correspondiente a esta solución es $\{1,2,4,8\}$, por tanto las raíces del polinomio $p_1(x)$ en el campo $GF(2^8)$ son $\beta^{17},\beta^{34},\beta^{68}$ y β^{136}.

Las funciones que representan los homomorfismos entre los campos $GF(2^4)$ y $GF(2^8)$ son

$$h_1(\alpha)=\beta^{17},h_2(\alpha)=\beta^{34},h_3(\alpha)=\beta^{68},h_4(\alpha)=\beta^{136}.$$

$(\gamma^{257})^7$ es raíz del polinomio $p_2(x)$ en el campo $GF(2^{16})$. El coseto 2-ciclotómico módulo 255 correspondiente a esta solución es $\{7,14,28,56,112,224,193,131\}$, por tanto las raíces de este polinomio en el campo $GF(2^{16})$, son:

$$(\gamma^{257})^7,(\gamma^{257})^{14},(\gamma^{257})^{28},(\gamma^{257})^{56},(\gamma^{257})^{112},(\gamma^{257})^{224},(\gamma^{257})^{193},(\gamma^{257})^{131}$$

Las funciones que representan los homomorfismos entre los campos $GF(2^8)$ y $GF(2^{16})$ son:

$$g_1(\beta)=(\gamma^{257})^7=\gamma^2+\gamma^3+\gamma^4+\gamma^5+\gamma^8+\gamma^{10}+\gamma^{14}+\gamma^{15}$$

$$g_2(\beta)=(\gamma^{257})^{14}=1+\gamma+\gamma^2+\gamma^3+\gamma^7+\gamma^8+\gamma^9+\gamma^{10}+\gamma^{12}+\gamma^{13}+\gamma^{15}$$

$$g_3(\beta)=(\gamma^{257})^{28}=\gamma+\gamma^2+\gamma^6+\gamma^7+\gamma^8+\gamma^{10}+\gamma^{15}$$

$$g_4(\beta)=(\gamma^{257})^{56}=\gamma+\gamma^5+\gamma^6+\gamma^8+\gamma^9+\gamma^{11}+\gamma^{15}$$

$$g_5(\alpha)=(\gamma^{257})^{112}=1+\gamma+\gamma^7+\gamma^8+\gamma^9+\gamma^{10}+\gamma^{13}+\gamma^{14}$$

$$g_6(\alpha)=(\gamma^{257})^{224}=\gamma+\gamma^3+\gamma^4+\gamma^5+\gamma^6+\gamma^7+\gamma^8+\gamma^{11}+\gamma^{12}$$

$$g_7(\alpha)=(\gamma^{257})^{193}=\gamma^3+\gamma^4+\gamma^5+\gamma^6+\gamma^8+\gamma^9+\gamma^{10}+\gamma^{13}$$

$$g_8(\beta)=(\gamma^{257})^{131}=\gamma+\gamma^2+\gamma^4+\gamma^8+\gamma^{10}+\gamma^{13}$$

En el Anexo IV se presenta otro ejemplo de determinación de los isomorfismos entre distintas variantes del campo $GF(5^2)$.

2.4 Conclusiones del capítulo

En este capítulo se demostraron los teoremas de condiciones necesarias y suficientes para la existencia de los homomorfismos de inmersión, se abordaron los fundamentos teóricos de los algoritmos para la determinación los homomorfismos de inmersión entre campos finitos, su relación con el automorfismo de *Frobenius*, se diseñó un algoritmo para la determinación de estos homomorfismos y se presentaron ejemplos ilustrativos de la aplicación del mismo.

CAPÍTULO 3. LAS MATRICES MDS Y LOS HOMOMORFISMOS DE INMERSIÓN

El objetivo de este capítulo es encontrar una forma más eficiente para generar matrices MDS dentro de los algoritmos de cifrado en función de la llave, a partir del estudio de otras caracterizaciones de las matrices MDS vistas en el epígrafe 1.3.1 y del algoritmo FREDIA para la generación aleatoria de matrices MDS $A \in GL_n\big(GF(q)\big)$, (Freyre et al. 2014; Freyre et al. 2015), para el caso $n = 4$ y proponer un procedimiento para a partir de matrices MDS con coeficientes sobre un campo obtener matrices MDS con coeficientes sobre un campo homomorfo al anterior.

El algoritmo FREDIA fue descrito en el epígrafe 1.3.2 y tiene como entrada: 3 polinomios primitivos, sobre el campo $GF(q)$, seleccionados a priori y una matriz M seleccionada aleatoriamente donde sus elementos pertenecen al campo $GF(q)$.

$$M = \begin{pmatrix} \mathrm{b}_{1,0} & \mathrm{b}_{1,1} & \mathrm{b}_{1,2} & \mathrm{b}_{1,2} \\ \mathrm{c}_{2,0} & \mathrm{b}_{2,0} & \mathrm{b}_{2,1} & \mathrm{b}_{2,2} \\ \mathrm{c}_{3,0} & \mathrm{c}_{3,1} & \mathrm{b}_{3,0} & \mathrm{b}_{3,1} \\ \mathrm{c}_{4,0} & \mathrm{c}_{4,1} & \mathrm{c}_{4,2} & \mathrm{b}_{4,0} \end{pmatrix}$$

Donde: $c_{i,j} y b_{k,t} \in GF(q)$, q potencia de un primo p.

$$\big(b_{k,0}, b_{k,1}, \ldots, b_{k,n-k}\big) \neq 0, k = \overline{1,4}, t = \overline{0,3}, i = \overline{2,4}\ yj = \overline{0,2},\ b_{1,i} \neq 0, \forall\, i.$$

Este algoritmo trabaja a partir de la caracterización 1 de las matrices MDS estudiada en el epígrafe 1.3.1. Una matriz A es una matriz MDS si y sólo si todas las submatrices cuadradas de A son no singulares.

La caracterización 4 de este propio epígrafe afirma que cualquier matriz 4×4 de $GF(2^m)$ con todas las entradas no nulas es MDS si y sólo si es de rango completo, con su matriz inversa teniendo todas las entradas no nulas y todas las submatrices de tamaño 2×2 de rango completo.

Esta caracterización permite diseñar un nuevo algoritmo para la generación aleatoria de matrices MDS de 4×4 sobre campos de característica dos o sea $GF(2^m)$.

3.1 Nuevo algoritmo para la generación aleatoria de matrices MDS

El nuevo algoritmo para la generación aleatoria de matrices MDS de tamaño 4×4 sobre el campo $GF(2^8)$, trabaja con la matriz y su inversa y chequea que todos los elementos de ambas, así como los determinantes de todas las submatrices de orden dos, sean no nulos.

Entrada: Polinomio $x^8 + x^4 + x^3 + x^2 + 1$, primitivo definidor del campo $GF(2^8)$. Polinomios primitivos $g_1(x), g_2(x), g_3(x) \in GF(2^8)[x]$ seleccionados a priori y que cumplen que: $gr\big(g_1(x)\big) = 4,\ gr\big(g_2(x)\big) = 3,\ gr(g_3(x)) = 2)$.

La matriz aleatoria $M = \begin{pmatrix} \mathrm{b}_{1,0} & \mathrm{b}_{1,1} & \mathrm{b}_{1,2} & \mathrm{b}_{1,3} \\ \mathrm{c}_{2,0} & \mathrm{b}_{2,0} & \mathrm{b}_{2,1} & \mathrm{b}_{2,2} \\ \mathrm{c}_{3,0} & \mathrm{c}_{3,1} & \mathrm{b}_{3,0} & \mathrm{b}_{3,1} \\ \mathrm{c}_{4,0} & \mathrm{c}_{4,1} & \mathrm{c}_{4,2} & \mathrm{b}_{4,0} \end{pmatrix}$, con $\mathrm{b}_{1,0}$ como incógnita.

Donde: $c_{i,j}\, y\ b_{k,t} \in GF(2^8), \big(b_{k,0}, b_{k,1}, \dots, b_{k,n-k}\big) \neq 0,$

$$k = \overline{1,4}, t = \overline{0,3}, i = \overline{2,4}\ y\ j = \overline{0,2}, \mathrm{b}_{1,i} \neq 0, \forall\, i.$$

Tabla de las operaciones suma y producto del campo $GF(2^8)$, expresando los elementos del campo como potencias del elemento primitivo.

Salida: Matriz MDS $A\ y\ A^{-1}$.

1. $Fila_1 := (b_{1,0}, b_{1,1}, b_{1,2}, b_{1,3})$
2. Cálculo de la fila j – ésima de A, $2 \leq j \leq 4$.

Pasos desde 1 hasta $j-1$.

Entrada: $(a_0, a_1, a_2, a_3) :=$ el vector canónico que tiene un uno en la posición j – ésima y cero en las restantes posiciones.

Para i desde $i = j$ hasta 2, hacer

Inicio

$$\hat{a}_0 := a_0 + c_{i,0}a_{i-1}, \hat{a}_1 := a_1 + c_{i,1}a_{i-1}, \cdots, \hat{a}_{i-2} := a_{i-2} + c_{i,i-2}a_{i-1}$$

$$\left(a_{i-1} + a_i x + \cdots + a_3 x^{4-i}\right)\left(\mathrm{b}_{i,0} + \mathrm{b}_{i,1}x + \cdots + \mathrm{b}_{i,4-i}x^{4-i}\right) mod g_i(x)$$

$$= \hat{a}_{i-1} + \hat{a}_i x + \cdots + \hat{a}_3 x^{4-i}$$

$$(a_0, a_1, a_2, a_3) := (\hat{a}_0, \hat{a}_1, \hat{a}_2, \hat{a}_3)$$

Fin

Paso j:

Entrada: (a_0, a_1, a_2, a_3)

$$(a_0 + a_1 x + a_2 x^2 + a_3 x^3)\left(\mathrm{b}_{1,0} + \mathrm{b}_{1,1}x + b_{1,2}x^2 + \mathrm{b}_{1,3}x^3\right) mod g_1(x)$$

$$= \hat{a}_0 + \hat{a}_1 x + \hat{a}_2 x^2 + \hat{a}_3 x^3.$$

Salida: $Fila_j := (\hat{a}_0, \hat{a}_1, \hat{a}_2, \hat{a}_3)$.

Fin

Los elementos de la fila j-ésima se igualan a cero y los determinantes de todas las submatrices hasta de orden dos que no fueron obtenidos en el cálculo de la fila anterior son determinados e igualados a cero, obteniéndose, ecuaciones lineales o no lineales con $b_{1,0}$ como incógnita. Los valores que no satisfacen las ecuaciones son almacenados.

De los valores almacenados, se selecciona uno al azar para la incógnita, se corren los pasos 1 y 2 y se obtiene la matriz A.

3. Calcular la inversa de la matriz obtenida y verificar que sus entradas son no nulas. Si todas las entradas de la inversa son no nulas ir al paso 4 en caso contrario volver a generar la matriz M.
4. Imprimir las matrices $A\ y\ A^{-1}$.

3.1.1 Ejemplo de aplicación del nuevo algortimo.

Obtener una matriz MDS sobre el campo $GF(2^8)$ utilizando el algoritmo propuesto.

Para facilitar la comprensión en este primer ejemplo se trabajó con la representación de los elementos del campo $GF(2^8)$ como polinomios de grado menor o igual que 7.

Se selecciona aleatoriamente la matriz M, considerando como incógnita al elemento b_{10}.

$$M = ((\ b_{10}\ , \alpha^3 + \alpha^4 + \alpha^5 + \alpha^6, \alpha^2 + \alpha^4 + \alpha^5 + \alpha^7, \alpha^3 + \alpha^4 + \alpha^5 + \alpha^6),$$

$$(\alpha + \alpha^2 + \alpha^4 + \alpha^7, \alpha^6, \alpha + \alpha^3 + \alpha^5 + \alpha^7, \alpha^3 + \alpha^5),$$

$$(1 + \alpha^3 + \alpha^4 + \alpha^5 + \alpha^7, \alpha + \alpha^2 + \alpha^3 + \alpha^4 + \alpha^7, \alpha + \alpha^2 + \alpha^3 + \alpha^5, 1 + \alpha^6),$$

$$(\alpha^2 + \alpha^4 + \alpha^5 + \alpha^6, \alpha + \alpha^2 + \alpha^3 + \alpha^4 + \alpha^5, \alpha^3 + \alpha^4 + \alpha^5 + \alpha^7, 1 + \alpha + \alpha^2 + \alpha^3 +$$

$$+\alpha^4 + \alpha^5 + \alpha^6 + \alpha^7))$$

Los polinomios primitivos seleccionados a priori son:

$g_1 = x^4 + \alpha^9 x^3 + \alpha^2 x^2 + \alpha^9; g_2 = x^3 + \alpha^{16} x + \alpha^{53}$; $g_3 = x^2 + \alpha^{67} x + \alpha^{14}$

Aplicando el algoritmo FREDIA para el cálculo de los elementos de las filas se obtiene

$Fila1 = (b_{10}, \alpha^3 + \alpha^4 + \alpha^5 + \alpha^6, \alpha^2 + \alpha^4 + \alpha^5 + \alpha^7, \alpha^3 + \alpha^4 + \alpha^5 + \alpha^6)$

$Fila2 = (1 + \alpha^2 + \alpha^3 + \alpha^7 + \alpha b_{10} + \alpha^2 b_{10} + \alpha^4 b_{10} + \alpha^7 b_{10}, 1 + \alpha + \alpha^3 + \alpha^4 + \alpha^7 + \alpha^6 b_{10}, 1 + \alpha^4 + \alpha^5 + \alpha^7 + \alpha b_{10} + \alpha^3 b_{10} + \alpha^5 b_{10} + \alpha^7 b_{10}, 1 + \alpha^2 + \alpha^3 + \alpha^4 + \alpha^5 + \alpha^7 + \alpha^3 b_{10} + \alpha^5 b_{10})$

$Fila3 = (1 + \alpha + \alpha^4 + \alpha^5 + \alpha^7 + b_{10} + \alpha b_{10} + \alpha^2 b_{10} + \alpha^5 b_{10} + \alpha^6 b_{10}, 1 + \alpha + \alpha^2 + \alpha^3 + \alpha^7 + b_{10} + \alpha b_{10} + \alpha^3 b_{10} + \alpha^4 b_{10}, 1 + \alpha^5 + \alpha^6 + b_{10} + \alpha b_{10} + \alpha^3 b_{10} + \alpha^4 b_{10} + \alpha^6 b_{10}, 1 + \alpha + \alpha^2 + \alpha^4 + \alpha^6 + \alpha^7 + b_{10} + \alpha^2 b_{10})$

$Fila\ 4 = (\alpha + \alpha^2 + \alpha^3 + \alpha^4 + \alpha^6 + \alpha^7 + b_{10} + \alpha b_{10} + \alpha^2 b_{10} + \alpha^5 b_{10} + \alpha^7 b_{10}, 1 + \alpha^2 + \alpha^4 + \alpha^5 + b_{10} + \alpha b_{10} + \alpha^2 b_{10} + \alpha^3 b_{10} + \alpha^4 b_{10} + \alpha^5 b_{10} + \alpha^7 b_{10}, 1 + \alpha^3 + \alpha^4 + \alpha^6 + \alpha^7 + \alpha^3 b_{10} + \alpha^4 b_{10} + \alpha^7 b_{10}, 1 + \alpha^4 + \alpha^5 + \alpha^6 + b_{10} + \alpha^4 b_{10} + \alpha^5 b_{10} + \alpha^7 b_{10})$

Hallando los valores que anulan los elementos de la matriz se obtiene un primer conjunto de valores inadmisibles (VI) para b_{10}.

$VI1 = \{0, \alpha^3 + \alpha^4 + \alpha^7, \alpha + \alpha^4 + \alpha^5 + \alpha^7, \alpha + \alpha^2 + \alpha^3 + \alpha^5 + \alpha^7, 1 + \alpha^2 + \alpha^3 + \alpha^4 + \alpha^6, 1 + \alpha + \alpha^2 + \alpha^3 + \alpha^5 + \alpha^6, \alpha^6 + \alpha^7, \alpha + \alpha^3 + \alpha^4 + \alpha^5 + \alpha^6, 1 + \alpha + \alpha^3 + \alpha^4 + \alpha^5, 1 + \alpha^7, \alpha^3 + \alpha^4 + \alpha^5 + \alpha^7, \alpha^4 + \alpha^5 + \alpha^6 + \alpha^7, 1 + \alpha^2 + \alpha^3 + \alpha^4\}.$

Se calculan los determinantes de las submatrices de orden dos y se igualan a cero, obteniéndose 9 ecuaciones lineales y 27 ecuaciones cuadráticas. Las expresiones de estos determinantes pueden verse en el Anexo VI.4.

Las soluciones de las ecuaciones lineales son valores inadmisibles para la variable b_{10}.

$$VI2 = (\alpha^5 + \alpha^6, 1 + \alpha + \alpha^5 + \alpha^6, 1 + \alpha^4 + \alpha^5 + \alpha^6 + \alpha^7, \alpha + \alpha^3 + \alpha^5 + \alpha^6, 1 + \alpha + \alpha^2 + \alpha^4 + \alpha^5 + \alpha^7, 1 + \alpha^4 + \alpha^6 + \alpha^7, \alpha + \alpha^2 + \alpha^4 + \alpha^5, 1 + \alpha^4 + \alpha^7, \alpha + \alpha^2 + \alpha^5 + \alpha^7\}$$

Las ecuaciones cuadráticas obtenidas tienen como soluciones los valores

$$VI3 = (\alpha^4 + \alpha^6 + \alpha^7, \alpha + \alpha^4 + \alpha^5 + \alpha^6 + \alpha^7, 1 + \alpha + \alpha^3 + \alpha^6 + \alpha^7, \alpha + \alpha^4 + \alpha^7, \alpha + \alpha^5 + \alpha^7, \alpha + \alpha^2 + \alpha^7, 1 + \alpha + \alpha^2 + \alpha^3 + \alpha^6 + \alpha^7, \alpha^6, \alpha^3 + \alpha^5, 1 + \alpha + \alpha^2 + \alpha^4 + \alpha^7, 1 + \alpha^2 + \alpha^6, 1 + \alpha + \alpha^2 + \alpha^3 + \alpha^4 + \alpha^7, \alpha + \alpha^2 + \alpha^3 + \alpha^5 + \alpha^6, 1 + \alpha + \alpha^2 + \alpha^3 + \alpha^5 + \alpha^6, 1 + \alpha + \alpha^2 + \alpha^4 + \alpha^5 + \alpha^7, 1 + \alpha^2 + \alpha^3, 1 + \alpha^4 + \alpha^7, 1 + \alpha^2 + \alpha^4 + \alpha^7, \alpha^2 + \alpha^3 + \alpha^5 + \alpha^7\}$$

Los valores admisibles (VA) para b_{10} que garantizan que tanto los elementos de las filas como los determinantes de las submatrices de orden dos sean no nulos son

$VA = GF(2^8) \setminus (VI1 \cup VI2 \cup VI3)$.

Se selecciona aleatoriamente un valor de VA para b_{10} , por ejemplo $b_{10} = 1 + \alpha + \alpha^2$. Si se guardan los elementos calculados de las filas solo se necesita evaluarlos para el valor b_{10} seleccionado.

La matriz

$$A = ((1 + \alpha + \alpha^2, \alpha^3 + \alpha^4 + \alpha^5 + \alpha^6, \alpha^2 + \alpha^4 + \alpha^5 + \alpha^7, \alpha^3 + \alpha^4 + \alpha^5 + \alpha^6), (\alpha^3 + \alpha^6, \alpha + \alpha^2 + \alpha^6, \alpha^6 + \alpha^7, 1 + \alpha^2 + \alpha^5 + \alpha^6), (1 + \alpha + \alpha^3 + \alpha^4 + \alpha^7, \alpha + \alpha^2 + \alpha^3 + \alpha^6 + \alpha^7, 1 + \alpha^2 + \alpha^3 + \alpha^4 + \alpha^5 + \alpha^6 + \alpha^7, \alpha^2 + \alpha^3 + \alpha^6 + \alpha^7), (\alpha^2 + \alpha^3 + \alpha^7, 1 + \alpha + \alpha^2 + \alpha^3 + \alpha^5, \alpha + \alpha^2 + \alpha^4 + \alpha^5, 1 + \alpha^6))$$

Se halla su inversa utilizando el *Mathematica* 9.0 y se verifica que todas sus entradas son no nulas

$A^{-1} = ((1+\alpha+\alpha^2+\alpha^3+\alpha^7, 1+\alpha^2+\alpha^4+\alpha^5, 1+\alpha+\alpha^6, \alpha+\alpha^2+\alpha^3+\alpha^7), (1+\alpha+\alpha^2+\alpha^7, \alpha+\alpha^2+\alpha^7, 1+\alpha+\alpha^2+\alpha^3+\alpha^5+\alpha^7, \alpha+\alpha^2+\alpha^3+\alpha^4+\alpha^5+\alpha^7), (1+\alpha+\alpha^3+\alpha^5+\alpha^7, 1+\alpha^3+\alpha^6+\alpha^7, 1+\alpha+\alpha^2+\alpha^3+\alpha^4+\alpha^5, 1+\alpha^3+\alpha^7), (1+\alpha+\alpha^2+\alpha^3+\alpha^6+\alpha^7, 1+\alpha^3+\alpha^4+\alpha^6, \alpha^2+\alpha^4+\alpha^6, 1+\alpha^5+\alpha^6))$

Por tanto la matriz A y su inversa A^{-1} son MDS.

Se seleccionaron al azar varios valores del conjunto VA y para todos ellos se obtuvieron matrices MDS. Los valores seleccionados son:

$1+\alpha+\alpha^7, \alpha^2+\alpha^4, 1+\alpha^2+\alpha^5+\alpha^7, \alpha+\alpha^2+\alpha^3+\alpha^4+\alpha^6, 1+\alpha+\alpha^3+\alpha^4+\alpha^6.$

3.2 Análisis de complejidad

Se realiza un análisis comparativo de las complejidades del nuevo algoritmo y el algoritmo FREDIA.

Sea $n = 4$ el orden de la matriz cuadrada generada, $GF(q) = GF(p^m) = GF(2^8)$.

La **complejidad del algoritmo FREDIA** depende de la complejidad de:

- El cálculo de las filas de la matriz A, $2 \leq j \leq n$.
- La determinación de los determinantes de las sub-matrices de $i \times i, 1 \leq i \leq j$, que no fueron obtenidos después del cálculo de la fila anterior.
- La solución de las ecuaciones lineales y no lineales que se obtienen al igualar a cero los determinantes de las sub-matrices de $i \times i, 1 \leq i \leq j$, considerando como incógnita el elemento considerado de la i-ésima fila.

Se considera que las operaciones en el campo $GF(q) = GF(2^8)$ se realizan mediante consultas a tablas para comparar los algoritmos en igualdad de condiciones.

Cálculo de las filas de la matriz $A, 2 \leq j \leq n$

La operación que predomina es la multiplicación de polinomios con coeficientes sobre extensiones de campos. Estos polinomios pueden ser considerados en el caso peor como elementos de la extensión $GF(q^n)$, siendo n el orden de las matrices a generar y el grado del polinomio primitivo g_1. Este producto tiene una complejidad de $O(n \log n \log\log n)$ operaciones en $GF(q)$, (Knuth 1968; Graham et al.1990; Knuth 1998). El costo de cada operación en $GF(q)$ es de un $O(1)$ ya que se realiza mediante consulta a tablas. Se realizan $2n$ operaciones de este tipo.

El cálculo de las filas tiene una complejidad menor que $O\left((2n)\, n \log n \log\log n\right) \rightarrow O(n^2 \log n \log\log n)$.

Determinación de los determinantes de las submatrices de $i \times i$, $1 \leq i \leq j$

La cantidad de submatrices de $i \times i, 1 \leq \mathrm{i} \leq \mathrm{j}$ que no fueron obtenidas en la fila anterior es T donde T=$\sum_{i=2}^{j}\binom{j-1}{i-1}\binom{n}{i}$. La complejidad para calcular el determinante de una matriz $i \times i$, $2 \leq i \leq j$ es $O(i!)$ o $O(i^3)$ multiplicaciones en $GF(q)$. La complejidad es $O(i!)$ si se utiliza el método de menores (Poole 2010) y $O(i^3)$ si se utiliza el método de triangulación. Como en nuestro caso $2 \leq i \leq j < 4$, $O(i!) < O(i^3)$. La complejidad del cálculo de los T determinantes es $O(T\, i!)$.

Solución de las ecuaciones lineales y no lineales

Si se utiliza búsqueda exhaustiva y se tiene en cuenta la complejidad de la evaluación de polinomios de orden $n-1$ (en el caso peor) en el campo $GF(q)$ vista en el epígrafe 2.3.2. La complejidad es entonces de $O(Tq(n-1)))$.

La cota superior de la complejidad del algoritmo FREDIA es menor que $O(T\, q\, (n-1))$.

Complejidad del algoritmo propuesto

Sea $n = 4$ el orden de la matriz cuadrada generada y $k = 36$ la cantidad de submatrices de orden dos de la misma, $GF(q) = GF(p^m) = GF(2^8)$.

A continuación se analiza la complejidad de cada paso del algoritmo

Cálculo de los elementos de las filas y chequear que son no nulos.

Esta operación es la misma que realiza el algoritmo FREDIA por tanto su complejidad es de $O(n^2 log\, n log log\, n)$.

Cálculo de los K determinantes de las submatrices de orden dos

Los determinantes de las submatrices están pre calculados en función de los elementos de la matriz. Para el cálculo de cada determinante hay que realizar dos productos y una suma sobre $GF(2^m) = GF(2^8)$, operaciones que se realizan mediante consultas a tablas. La complejidad de este paso es de $O(K(n-1))$, siendo $K = \binom{n}{2}^2 = \left[\frac{n(n-1)}{2}\right]^2$.

Cálculo de la inversa de la matriz obtenida.

El cálculo de la inversa de una matriz $A \in GL_n(GF(q))$, no singular tiene una complejidad de $O(n^3 log\, n log log\, n)$ si se usa el método propuesto en (Freyre et al. 2009b) y que puede verse en el Anexo VI.4.

La complejidad del algoritmo es menor que $O(n^3 log\, n log log\, n)$ ya que

$O(K(n-1)) < O(n^3 log\, n log log\, n)$.

Observación

La tabla de la suma de los elementos del campo, expresados como potencia de elemento primitivo, que se da como entrada del algoritmo se construye utilizando la tabla del

logaritmo de *Zech´s* (Chen 2005) para el campo $GF(2^8)$ (Cuellar et al. 2016) que aparece en el Anexo V.

El análisis de complejidad lleva a concluir que el nuevo algoritmo es más eficiente que el algoritmo FREDIA en cuanto a costo y cantidad de operaciones pues

$$O\big(Tq(n-1)\big) > O\big(\, n^3 log\, n log log\, n\big).$$

3.3 Variantes del nuevo algoritmo propuesto

Las variantes que se estudian están relacionadas con la forma de la matriz M seleccionada aleatoriamente

Variante 1

$$\begin{pmatrix} b_{1,0} & b_{1,1} & b_{1,2} & b_{1,2} \\ 0 & b_{2,0} & 0 & 0 \\ 0 & 0 & b_{3,0} & 0 \\ 0 & 0 & 0 & b_{4,0} \end{pmatrix}$$

Variante 2

$$\begin{pmatrix} b_{1,0} & b_{1,1} & b_{1,2} & b_{1,2} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

3.3.1 Variante 1 del algoritmo propuesto

Se considera el elemento $b_{i,0}, .\, i = \overline{2,4}$ como incógnita en la construcción de la i- ésima fila de la matriz MDS A.

El i-ésimo vector canónico será la entrada del algoritmo en el cálculo de la fila i.

Se analiza el comportamiento de las expresiones para el cálculo de las filas de la matriz

Cálculo de la primera fila de A

$Fila_1 :=\ \big(b_{1,0}, b_{1,1}, b_{1,2}, b_{1,3}\big)$

Cálculo de la i- ésima fila de A ($2 \leq i \leq 4$) se reduce a un solo paso.

Entrada:

$$(a_0, a_1, a_2, a_3) := \text{vector que tiene en la i} - \text{ésima posición el producto } \prod_{k=2}^{i} b_{k,0}$$

$$(a_0 + a_1x + a_2x^2 + a_3x^3)\left(b_{1,0} + b_{1,1}x + b_{1,2}x^2 + b_{1,3}x^3\right) mod g_1(x)$$

$$= \left(\prod_{k=2}^{i} b_{k,0}\right) x^{i-1}\left(b_{1,0} + b_{1,1}x + b_{1,2}x^2 + b_{1,3}x^3\right) mod g_1(x)$$

$$= \widehat{a_0} + \widehat{a_1}x + \widehat{a_2}x^2 + \widehat{a_3}x^3$$

Salida: $Fila_i := [\widehat{a_0}, \widehat{a_1}, \widehat{a_2}, \widehat{a_3}]$

En la medida en que se calculan los elementos de las filas se va chequeando que sean no nulos.

La aplicación del algoritmo en este caso, tiene la particularidad de que después de obtenida la fila i, calculados e igualados a cero los determinantes de las submatrices hasta de orden dos y resolver las ecuaciones que resultan, teniendo al elemento $b_{i,0}$ como incógnita, se obtiene un solo valor para $b_{i,0}, b_{i,0} = 0,$ por tanto si se trabaja con la condición de que $b_{i,0} \neq 0$, los determinantes de las submatrices serán iguales a cero o no en dependencia de la relación que exista entre los elementos de la primera fila de la matriz M y los coeficientes del polinomio g_1.

Denotando los elementos de la fila i por $f_{i\,j}$, $con\ i = \overline{1,4}$, $j = \overline{0,3}$, el polinomio primitivo $g_1(\mathrm{x}) = g_{10} + g_{11}x + g_{12}\,x^2 + g_{1\,3}x^3 + x^4 \in GF(q)[x]$, expresando los elementos de las filas en función de f_{ij} y los coeficientes de g_1 y realizando operaciones algebraicas se obtienen expresiones recursivas para el cálculo de los elementos de las filas. La obtención de estas expresiones aparece en el Anexo VI.3.

Las expresiones siguientes permiten calcular los elementos de la fila i a partir de los elementos de la fila $i - 1$.

$$Fila_1 = (f_{10}, f_{11}, f_{12}, f_{13})$$

$$Fila_2 = \left(b_{2,0}(g_{10}f_{13}), b_{2,0}(f_{10} + g_{11}f_{13}), b_{2,0}(f_{11} + g_{12}f_{13}), b_{2,0}(f_{12} + g_{13}f_{13})\right)$$

$$Fila_3 = \left(b_{3,0}g_{10}f_{23}, b_{3,0}(f_{20} + g_{11}f_{23}), b_{3,0}(f_{21} + g_{12}f_{23}), b_{3,0}(f_{22} + g_{13}f_{23})\right)$$

$$Fila_4 = \left(b_{4,0}g_{10}f_{33}, b_{4,0}(f_{30} + g_{11}f_{33}), b_{4,0}(f_{31} + g_{12}f_{33}), b_{4,0}(f_{32} + g_{13}f_{33})\right)$$

De manera general se tiene que la i-ésima fila de la matriz, con $i \geq 2$, puede ser expresada como

$$Fila_i = \left(b_{i,0}g_{10}f_{(i-1)3}, b_{i,0}\left(f_{(i-1)0} + g_{11}f_{(i-1)3}\right), b_{i,0}\left(f_{(i-1)1} + g_{12}f_{(i-1)3}\right), b_{i,0}\left(f_{(i-1)2} + g_{13}f_{(i-1)3}\right)\right)$$

La obtención de estas expresiones permite reducir la complejidad del algoritmo de manera significativa, pues el costo computacional del algoritmo no tiene ahora en cuenta las operaciones de multiplicación de polinomios sobre campos extendidos, el cálculo de los determinantes de las submatrices y la solución de ecuaciones.

En este caso se obtiene un subconjunto de las matrices MDS.

3.3.2 Variante 2 del algoritmo propuesto

En el epígrafe 1.3.1 se hacia referencia a la propiedad 6 de las matrices MDS. Teniendo en cuenta esta propiedad se tiene que si en la matriz MDS A obtenida en la variante 1 del algoritmo se extrae en la fila i-ésima el elemento $b_{i,0}$ como factor común la matriz sigue siendo MDS. Se justifica entonces el estudio de esta nueva variante en la cual todos los $b_{i,0} = 1$, $\forall\, i = \overline{2,4}$, que reduce considerablemente el tiempo de ejecución del algoritmo. En este caso el cálculo de la i- ésima fila de A *($2 \leq i \leq 4$)* se reduce a un solo paso y adopta la forma

Entrada: $(a_0, a_1, a_2, a_3) := i - ésimo\ vector\ canónico$

$$(a_0 + a_1x + a_2x^2 + a_3x^3)\left(b_{1,0} + b_{1,1}x + b_{1,2}x^2 + b_{1,3}x^3\right) mod g_1(x)$$

$$= x^{i-1}\left(b_{1,0} + b_{1,1}x + b_{1,2}x^2 + b_{1,3}x^3\right) mod g_1(x) = \widehat{a_0} + \widehat{a_1}\,x + \widehat{a_2}x^2 + \widehat{a_3}x^3$$

Salida: $Fila\ i \coloneqq (\widehat{a_0}, \widehat{a_1}, \widehat{a_2}, \widehat{a_3})$

Las expresiones para el cálculo de los elementos de la fila i a partir de los elementos de la fila $i-1$.

$$Fila_1 = (f_{10}, f_{11}, f_{12}, f_{13})$$

$$Fila_2 = (g_{10}f_{13}, f_{10} + g_{11}f_{13}, f_{11} + g_{12}f_{13}, f_{12} + g_{13}f_{13})$$

$$Fila_3 = (g_{10}f_{23}, f_{20} + g_{11}f_{23}, f_{21} + g_{12}f_{23}, f_{22} + g_{13}f_{23})$$

$$Fila_4 = (g_{10}f_{33}, f_{30} + g_{11}f_{33}, f_{31} + g_{12}f_{33}, f_{32} + g_{13}f_{33})$$

De manera general se tiene que la i -ésima fila de la matriz con $2 \leq i \leq 4$.

$$Fila_i = \left(g_{10}f_{(i-1)3}, f_{(i-1)0} + g_{11}f_{(i-1)3}, f_{(i-1)1} + g_{12}f_{(i-1)3}, f_{(i-1)2} + g_{13}f_{(i-1)3}\right).$$

En (Freyre et al. 2009b; Freyre et al. 2015) los autores presentan un algoritmo para el cálculo de la matriz A MDS y su inversa. El algoritmo para el cálculo de la inversa tiene la misma estructura solo que trabaja con los polinomios inversos asociados a los elementos $b_{j,t} \in GF(2^8), j = \overline{1,4}, t = \overline{0,3}$ de la j-ésima fila de la matriz M seleccionada aleatoriamente.

Para el caso de esta variante el algoritmo de cálculo de la inversa adopta la forma obtenida en el Anexo VI.4.

Fila i - ésima de A^{-1}.

Entrada: $(a_0, a_1, a_2, a_3) \coloneqq i - ésimo\ vector\ canónico$

$$x^{i-1}\left(b_{1,0} + b_{1,1}x + b_{1,2}x^2 + b_{1,3}x^3\right)^{-1} mod g_1(x) = \widehat{a_0} + \widehat{a_1}\,x + \widehat{a_2}x^2 + \widehat{a_3}x^3$$

Salida: $Fila_i \coloneqq (\widehat{a_0}, \widehat{a_1}, \widehat{a_2}, \widehat{a_3})$

El nuevo algoritmo de generación aleatoria de matrices para esta variante seria:

Entrada: Polinomio $p(x)$ definidor del campo $GF(2^8)$, polinomio primitivo

$$g_1(\mathrm{x}) = g_{10} + g_{11}x + g_{12}\,x^2 + g_{1\,3}x^3 + x^4 \ \in GF(q)[x]$$

Las tablas de la suma y el producto de los elementos del campo $GF(2^8)$.

Salida: Matriz MDS A y su inversa A^{-1}.

1. Generar aleatoriamente los 4 elementos $f_{1\,t} \neq 0, con\ \ t \in [0,3]$, de la primera fila de la matriz A en $GF(2^8)^*$

$$Fila_1 = (f_{10}, f_{11}, f_{12}, f_{13})$$

2. Calcular $Fila_i = (f_{i0}, f_{i1}, f_{i2}, f_{i3}),\ con\ f_{ij}\ \in GF(2^8)^*$, siendo

$$Fila_i = \left(g_{10}f_{(i-1)3}, f_{(i-1)0} + g_{11}f_{(i-1)3}, f_{(i-1)1} + g_{12}f_{(i-1)3}, f_{(i-1)2} + g_{13}f_{(i-1)3}\right)$$

Si $f_{ij} = 0\ \ para\ algun\ i,j,\ \ i = \overline{2,4}\ , j = \overline{0,3}$, volver al paso1, en caso contrario ir a 3.

3. Chequear que los K determinantes de las submatrices de orden dos sean no nulos.

   Los determinantes son denotados por $r_k, r_k \in GF(2^8)^*$con $k = \overline{1,36}$ y para el cálculo de estos determinantes se utilizan sus expresiones ya pre computados en función de los elementos de la matriz que pueden verse en el Anexo VI.2.

   Si para algún $k\ , r_k = 0,$ volver al paso 1, en caso contrario ir al paso 4.

4. Calcular el polinomio inverso del polinomio $\mathrm{f}_{10} + f_{11}x + f_{12}x^2 + f_{13}x^3$. Sea el polinomio inverso denotado por $\tilde{f}_{10} + \tilde{f}_{11}x + \tilde{f}_{12}\,x^2 + \tilde{f}_{13}x^3$.

5. Calcular la i-ésima fila de la matriz inversa.

   Se denota al elemento de la fila i columna j de la matriz inversa por $\tilde{f}_{ij}$ y la fila i por $\widetilde{Fila_\imath} = \left(\tilde{f}_{i0}, \tilde{f}_{i1}, \tilde{f}_{i2}, \tilde{f}_{i3}\right)$ con $\tilde{f}_{ij} \in GF(2^8),$ siendo

$$\widetilde{Fila_i} = \left(g_{10}\tilde{f}_{(i-1)3}, \tilde{f}_{(i-1)0} + g_{11}\tilde{f}_{(i-1)3}, \tilde{f}_{(i-1)1} + g_{12}\tilde{f}_{(i-1)3}, \tilde{f}_{(i-1)2} + g_{13}\tilde{f}_{(i-1)3}\right)$$

Si $\tilde{f}_{ij} = 0,\ para\ algún\ i\ , j\ con\ i = \overline{2,4}\ , j = \overline{0,3}$, volver al paso 1, en caso contrario ir al paso 6.

6. Imprimir las matrices $A = \begin{pmatrix} Fila\ 1 \\ Fila\ 2 \\ Fila\ 3 \\ Fila\ 4 \end{pmatrix}$ y $A^{-1} = \begin{pmatrix} \widetilde{\text{Fila1}} \\ \widetilde{\text{Fila2}} \\ \widetilde{\text{Fila3}} \\ \widetilde{\text{Fila4}} \end{pmatrix}$.

El pseudocódigo de este algoritmo puede verse en el Anexo VI.1.

3.3.3.Complejidad de la variante dos del algoritmo propuesto

Sea $n = 4$ el orden de la matriz cuadrada a generar y $k = 36$ la cantidad de submatrices de orden dos de la misma, $GF(q) = GF(p^m) = GF(2^8)$.

Se muestra la complejidad de cada paso del algoritmo de acuerdo a su pseudocódigo.

Paso 1: $O(n)$

Pasos 2-8: La suma y el producto se realizan mediante consultas a tablas. Cada suma y cada producto es un O (1).

Para el cálculo de cada fila, es necesario calcular n sumas y $(n-1)$ productos. La complejidad de este paso es entonces $O\big((n-1)(2n-1)\big)$.

Pasos 9-13: Chequear que los determinantes de las submatrices de orden dos, son distintos de cero. Los determinantes de las submatrices están pre calculados en función de los elementos de la matriz. Para el cálculo de cada determinante hay que realizar dos productos y una suma.

La complejidad de chequear la no igualdad a cero de los determinantes es $O\big(K(n-1)\big)$.

Paso 14: El polinomio inverso se halla utilizando el algoritmo extendido de Euclides que tiene una complejidad de $O(n\ (\log n)^2 \log\log n)$ (Gao et al.2000).

Pasos 15-23: $O\big((n-1)(2n-1)\big)$.

La complejidad del algoritmo es menor que $O\big(K(n-1)\big)$. La complejidad del nuevo algoritmo propuesto es mayor que la complejidad de esta variante $O\big(\ n^3 log\ n log log\ n\big) > O\big(K(n-1)\big)$.

La variante dos del algoritmo disminuye la complejidad de las operaciones que emplea lo que repercute en su complejidad y en su tiempo de ejecución.

3.3.4.Ejemplos de aplicación del nuevo algoritmo

Ejemplo 3.3.1

Sea la matriz M tal que $M = \begin{pmatrix} 1 & \alpha & \alpha^3 & \alpha^5 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$,

$$g_1 = \alpha^{13} + \alpha^2 x^2 + \alpha^9 x^3 + x^4 \ \in GF(2^8)[x];$$

$Fila1 \ = (1, \alpha, \alpha^3, \alpha^5)$,

Calculando el resto de las filas de la matriz usando las expresiones recursivas y las tablas del logaritmo de *Zech´s* se obtiene

$$Fila2 \ = (\alpha^{13}\alpha^5, 1, \alpha + \alpha^7, \alpha^3 + \alpha^{14}) = (\alpha^{18}, 1, \alpha^{192}, \alpha^{248})$$

$$Fila3 \ = (\alpha^{13}\alpha^{248}, \alpha^{18}, 1 + \alpha^2\alpha^{248}, \alpha^{192} + \alpha^9\alpha^{248}) = (\alpha^6, \alpha^{18}, \alpha^{133}, \alpha^{99})$$

$$Fila3 \ = (\alpha^{13}\alpha^{99}, \alpha^6, \alpha^{18} + \alpha^2\alpha^{99}, \alpha^{133} + \alpha^9\alpha^{99}) = (\alpha^{112}, \alpha^6, \alpha^{75}, \alpha^{109})$$

$$A = \begin{pmatrix} 1 & \alpha & \alpha^3 & \alpha^5 \\ \alpha^{18} & 1 & \alpha^{192} & \alpha^{248} \\ \alpha^6 & \alpha^{18} & \alpha^{133} & \alpha^{99} \\ \alpha^{112} & \alpha^6 & \alpha^{75} & \alpha^{109} \end{pmatrix}$$

Todas las entradas de la matriz son no nulas por tanto se continúa con la aplicación del algoritmo.

Calculando con el *Mathematica* 9.0 se obtienen los determinantes de las submatrices de orden dos y se verifica que son no nulos.

Se halla la matriz inversa utilizando también el *Mathematica* 9.0

$$A^{-1} = \begin{pmatrix} \alpha^{70} & \alpha^{240} & \alpha^{154} & \alpha^{229} \\ \alpha^{242} & \alpha^{70} & \alpha^{96} & \alpha^{194} \\ \alpha^{207} & \alpha^{242} & \alpha^{180} & \alpha^{154} \\ \alpha^{167} & \alpha^{207} & \alpha^{143} & \alpha^{231} \end{pmatrix}$$

$$((\alpha + \alpha^2 + \alpha^3 + \alpha^4 + \alpha^6, \alpha^2 + \alpha^3 + \alpha^5, 1 + \alpha^3 + \alpha^4 + \alpha^5, \alpha + \alpha^3 + \alpha^4 + \alpha^5 + \alpha^6), (\alpha^4 + \alpha^5 + \alpha^7, \alpha + \alpha^2 + \alpha^3 + \alpha^4 + \alpha^6, 1 + \alpha^3 + \alpha^4 + \alpha^6 + \alpha^7, \alpha + \alpha^4 + \alpha^5), (\alpha + \alpha^2 + \alpha^5 + \alpha^7, \alpha^4 + \alpha^5 + \alpha^7, \alpha + \alpha^2 + \alpha^4 + \alpha^7, 1 + \alpha^3 + \alpha^4 + \alpha^5), (\alpha + \alpha^2 + \alpha^3 + \alpha^4 + \alpha^5 + \alpha^6, \alpha + \alpha^2 + \alpha^5 + \alpha^7, \alpha^2 + \alpha^4 + \alpha^6, 1 + \alpha^2 + \alpha^4 + \alpha^5 + \alpha^6 + \alpha^7))$$

Todas las entradas de la matriz inversa son no nulas. La matriz A y su inversa A^{-1} son MDS.

Ejemplo 3.3.2

Sea la matriz M tal que $M = \begin{pmatrix} b & b & b & b \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$ donde $b \in GF(2^8)^*$ y

$$g_1 = \alpha^{13} + \alpha^2 x^2 + \alpha^9 x^3 + x^4 \in GF(2^8)[x];$$

Calculando las filas de la matriz se obtienen

$Fila\ 1 = (b, b, b, b)$,

$Fila\ 2 = \left(b\alpha^{13}, b, b(1 + \alpha^2), b(1 + \alpha^9)\right)$,

$Fila\ 3 = \left(b(\alpha^{13} + \alpha^{22}), b\alpha^{13}, b(1 + \alpha^2 + \alpha^{11}), b(1 + \alpha^2 + \alpha^9 + \alpha^{18})\right)$,

$$Fila\ 4\ = (b(\alpha^{13} + \alpha^{15} + \alpha^{22} + \alpha^{31}), b(\alpha^{13} + \alpha^{22}), b(\alpha^{2} + \alpha^{4} + \alpha^{11} + \alpha^{13} + \alpha^{20}), b(1 + \alpha^{2} + \alpha^{9} + \alpha^{18} + \alpha^{27}))$$

Efectuando las sumas utilizando las tablas del logaritmo de *Zech´s*.

$$Fila1\ = (b, b, b, b),$$

$$Fila2\ = (b\alpha^{13}, b, b\alpha^{50}, b\alpha^{120})$$

$$Fila3\ = (b\alpha^{133}, b\alpha^{13}, b\alpha^{117}, b\alpha^{170}),$$

$$Fila4\ = (b\alpha^{237}, b\alpha^{133}, b\alpha^{246}, b\alpha^{76})$$

Como $b \neq 0$, todas las entradas de la matriz obtenida son no nulas.

Reduciendo módulo el polinomio que define el campo

$$Fila1\ = (b, b, b, b),$$

$$Fila2 = \left(b(1 + \alpha + \alpha^{2} + \alpha^{7}), b, b(1 + \alpha^{2}), b(1 + \alpha + \alpha^{3} + \alpha^{4} + \alpha^{5})\right)$$

$$Fila3\ = \left(b(1 + \alpha^{2} + \alpha^{3} + \alpha^{5} + \alpha^{6}), b(1 + \alpha + \alpha^{2} + \alpha^{7}), b(1 + \alpha^{2} + \alpha^{3} + \alpha^{5} + \alpha^{6} + \alpha^{7}), b(\alpha + \alpha^{4})\right),$$

$$Fila4\ = \left(b(1 + \alpha + \alpha^{3} + \alpha^{7}), b(1 + \alpha^{2} + \alpha^{3} + \alpha^{5} + \alpha^{6}), b(1 + \alpha + \alpha^{2} + \alpha^{3} + \alpha^{6} + \alpha^{7}), b(\alpha + \alpha^{2} + \alpha^{3} + \alpha^{4})\right)$$

Los determinantes de las submatrices de orden dos son:

$$b^2\alpha^{99}, b^2\alpha^{192}, b^2\alpha^{35}, b^2\alpha^{2}, b^2\alpha^{9}, b^2\alpha^{114}, b^2\alpha^{22}, b^2\alpha^{7}, b^2\alpha^{87},$$

$$b^2\alpha^{40}, b^2\alpha^{184}, b^2\alpha^{175}, b^2\alpha^{160}, b^2\alpha^{102}, b^2\alpha^{184}, b^2\alpha^{209}, b^2\alpha^{159}, b^2\alpha^{161},$$

$$b^2\alpha^{84}, b^2\alpha^{22}, b^2\alpha^{127}, b^2\alpha^{16}, b^2\alpha^{87}, b^2\alpha^{161}, b^2\alpha^{100}, b^2\alpha^{197}, b^2\alpha^{188},$$

$$b^2\alpha^{238}, b^2\alpha^{210}, b^2\alpha^{144}, b^2\alpha^{140}, b^2\alpha^{100}, b^2\alpha^{174}, b^2\alpha^{187}, b^2\alpha^{188}, b^2\alpha^{173}$$

Como $b \neq 0$, todos los determinantes de las submatrices de orden dos son no nulos.

Calculando la inversa de la matriz usando el *Mathematica* 9.0

$$Fila\ 1 = (b^{-1}\alpha^{98}, b^{-1}\alpha^{183}, b^{-1}\alpha^{103}, b^{-1}\alpha^{209}),$$

$$Fila2 = (b^{-1}\alpha^{222}, b^{-1}\alpha^{98}, b^{-1}\alpha^{121}, b^{-1}\alpha^{209}),$$

$$Fila3 = (b^{-1}\alpha^{104}, b^{-1}\alpha^{222}, b^{-1}\alpha^{231}, b^{-1}\alpha^{110}),$$

$$Fila4 = (b^{-1}\alpha^{123}, b^{-1}\alpha^{104}, b^{-1}\alpha^{238}, b^{-1}\alpha^{126})$$

Reduciendo módulo el polinomio que define el campo

$$Fila1 = ((b^{-1}(1 + \alpha + \alpha^6), b^{-1}(\alpha^2 + \alpha^6 + \alpha^7), b^{-1}(\alpha^3 + \alpha^7), b^{-1}(\alpha + \alpha^5 + \alpha^7)),$$

$$Fila2 = ((b^{-1}(\alpha + \alpha^3 + \alpha^7), b^{-1}(1 + \alpha + \alpha^6), b^{-1}(\alpha + \alpha^2 + \alpha^4 + \alpha^5 + \alpha^6),$$

$$b^{-1}(\alpha + \alpha^5 + \alpha^7)),$$

$$Fila\ 3 = ((b^{-1}(1 + \alpha^2 + \alpha^3), b^{-1}(\alpha + \alpha^3 + \alpha^7), b^{-1}(1 + \alpha^2 + \alpha^4 + \alpha^5 + \alpha^6 + \alpha^7),$$

$$b^{-1}(1 + \alpha + \alpha^2 + \alpha^5 + \alpha^6)),$$

$$Fila\ 4 = ((b^{-1}(1 + \alpha^2 + \alpha^6 + \alpha^7), b^{-1}(1 + \alpha^2 + \alpha^3), b^{-1}(1 + \alpha + \alpha^3),$$

$$b^{-1}(\alpha + \alpha^2 + \alpha^5 + \alpha^6)),$$

Como $b \neq 0$, todas las entradas de la inversa son no nulas. La matriz A y su inversa obtenidas para valores fijos de b y para el polinomio primitivo g_1, son MDS para todos los valores de $b \neq 0$.

En este caso la propiedad de ser MDS se preserva en dependencia del polinomio primitivo seleccionado.

3.4 Uso de los homomorfismos de inmersión en la obtención de matrices MDS

El objetivo de este epígrafe es presentar un método para la obtención de matrices MDS sobre un campo $GF(p^n)$ a partir matrices MDS sobre un campo $GF(p^m)$, donde p es un número primo y n divide a m. El método trabaja en ambos sentidos y utiliza los homomorfismos de inmersión del campo $GF(p^n)$ en el campo $GF(p^m)$.

Es necesario destacar que para que el método funcione en ambos sentidos, en la matriz MDS $A \in GL_n\big(GF(p^m)\big)$, las entradas deben estar sobre el subcampo de $GF(p^m)$, isomorfo a $GF(p^n)$, siendo m un múltiplo entero de n.

Un homomorfismo de inmersión del campo $GF(p^n)$ en el campo $GF(p^m)$ es un homomorfismo inyectivo no nulo de $GF(p^n)$ en $GF(p^m)$. En el epígrafe 2.1 se obtuvieron las diferentes maneras de definir un homomorfismo de inmersión, del campo de *Galois* $GF(p^n)$ en el campo de *Galois* $GF(p^m)$ siendo p un número primo y *n* un divisor de *m*, también se determinó el número de maneras de realizar la inmersión que es igual al orden del grupo cíclico $Aut(GF(p^n))$, de automorfismos del campo $GF(p^n)$, el cual es de orden n y es generado por el llamado automorfismo de *Frobenius* F_P.

3.4.1 Aplicación de los homomorfismos de inmersión a la obtención de matrices MDS

A continuación se desarrolla una aplicación de los homomorfismos de inmersión a la obtención de matrices MDS y se solucionan dos casos que pueden presentarse.

1. Dadas las matrices rectangulares y MDS A, de orden $h \times k$, de componentes en un campo finito $GF(p^n)$, hallar un procedimiento para convertirlas en matrices, también $h \times k$ y MDS, pero de entradas en otro campo finito $GF(p^m)$. Siendo n un divisor de m.
2. Dadas las matrices rectangulares y MDS A, de orden $h \times k$, de componentes en el subcampo de $GF(p^m)$ isomorfo a $GF(p^n)$, hallar un procedimiento para convertirlas en matrices, también $h \times k$ y MDS, pero de entradas en el campo finito $GF(p^n)$. Siendo m un múltiplo entero de n.

Caso 1

Sean $h_r: GF(p^n) \to GF(p^m), 0 \leq r \leq n,$ los homomorfismos de inmersión del campo $GF(p^n)$ en el campo $GF(p^m)$, todos inyectivos, o sea, $h_r(0) = 0\ y\ h_r(\alpha^i) = \beta^{ik_r}$, para i $\in\{0,1,2,\ldots \mathrm{p^n} - 2\}$. Para todo r, $h_r = h_1 \circ \varphi^{r-1}$, donde $\varphi: x \to x^p$ es el llamado automorfismo de *Frobenius*.

Cada homomorfismo de campos puede ser extendido a un homomorfismo entre espacios vectoriales $H_r: M_{h\times k}(GF(p^n)) \to M_{h\times k}(GF(p^m))$ entre los espacios matriciales, de matrices rectangulares $h \times k$, con entradas en cada uno de los campos respectivos, de modo que si $A = (a_{ij})$, $H_r(A) = (h_r(a_{ij}))$.

Teorema 3.1.

Sea $A \in GL_n(GF(p^n))$ una matriz $A = (a_{ij})$ con $i = \overline{1,k}, j = \overline{1,k}$, y $f: GF(p^n) \to GF(p^m)$, un homomorfismo que inserta el campo de Galois $GF(p^n)$ en el también campo de Galois $GF(p^m)$ para p primo y m, n naturales, siendo n un divisor de m. Sea la matriz $B = f(A) = (f(a_{ij}))$ con $i = \overline{1,n}, j = \overline{1,n}$. Si la matriz $A \in GL_k(GF(p^n))$ es MDS entonces la matriz $B \in GL_k(GF(p^m))$ también es MDS.

Demostración

Si la matriz A es MDS entonces toda submatriz cuadrada de A es no singular y esto significa que el determinante de toda submatriz cuadrada es distinto de cero.

Sea A_K una cualquiera de las submatrices cuadradas de A de orden s ($s \leq k$). Sin pérdida de generalidad se supone que A_K está formada por las primeras k columnas y las primeras k filas.

$$A_S = \begin{pmatrix} a_{11} & \cdots & a_{1s} \\ \vdots & \ddots & \vdots \\ a_{s1} & \cdots & a_{ss} \end{pmatrix} f(A_S) = \begin{pmatrix} f(a_{11}) & \cdots & f(a_{1s}) \\ \vdots & \ddots & \vdots \\ f(a_{s1}) & \cdots & f(a_{ss}) \end{pmatrix}$$

Se halla el determinante de $f(A_s)$ y como f es un homomorfismo de campos preserva las operaciones del campo y por tanto

$$|f(A_S)| = \begin{vmatrix} f(a_{11}) & \cdots & f(a_{1s}) \\ \vdots & \ddots & \vdots \\ f(a_{s1}) & \cdots & f(a_{ss}) \end{vmatrix} = f\left(\begin{vmatrix} a_{11} & \cdots & a_{1s} \\ \vdots & \ddots & \vdots \\ a_{s1} & \cdots & a_{ss} \end{vmatrix}\right)$$

La matriz A es MDS, $\begin{vmatrix} a_{11} & \cdots & a_{1s} \\ \vdots & \ddots & \vdots \\ a_{s1} & \cdots & a_{ss} \end{vmatrix} \neq 0$, y como el homomorfismo transforma cada elemento neutro en el elemento neutro correspondiente, transforma cada elemento no nulo en un elemento no nulo, por tanto $f\left(\begin{vmatrix} a_{11} & \cdots & a_{1s} \\ \vdots & \ddots & \vdots \\ a_{s1} & \cdots & a_{ss} \end{vmatrix}\right) \neq 0$, la matriz $A_S\, es\, MDS$, como $f(A_S)$ es una submatriz cualquiera de B, entonces B es también MDS.■

El teorema 3.1 permite obtener matrices MDS a partir de matrices MDS conocidas utilizando los homomorfismos de inmersión entre campos finitos e incluso los isomorfismos entre dos variantes distintas de un mismo campo.

Como cada homomorfismo $h_r: GF(p^n) \rightarrow GF(p^m)$ de inmersión es un homomorfismo inyectivo de campos, cada imagen $h_r\big(GF(p^n)\big)$ es un subcampo del campo $GF(p^m)$. Por consiguiente, para la aplicación lineal H_r, que es también inyectiva, la imagen $H_r\left(M_{h\times k}\big(GF(p^n)\big)\right)$ es un subespacio vectorial del espacio $H_r\left(M_{h\times k}\big(GF(p^m)\big)\right)$ y resulta la igualdad, $H_r\left(M_{h\times k}\big(GF(p^n)\big)\right) = M_{h\times k}\left(h_r\big(GF(p^n)\big)\right)$ que es el espacio de matrices con entradas en el subcampo $h_r\big(GF(p^n)\big)$.

Caso 2:

Dada una matriz $B \in M_{h\times k}\left(h_r\left(GF(p^n)\right)\right)$, es decir, una matriz de tamaño $h \times k$ con entradas en el subcampo imagen $h_r\left(GF(p^n)\right)$; se determina su preimagen A, es decir $A \in M_{h\times k}\left(GF(p^n)\right)$ tal que $H_r(A) = B$.

Como cada $h_r: GF(p^n) \to GF(p^m)$ es inyectivo, restringiendo su codominio $GF(p^m)$ a su imagen $h_r\left(GF(p^n)\right)$, el mismo se convierte en un isomorfismo

$$\bar{h}_r: GF(p^n) \to h_r\left(GF(p^n)\right)$$

Entonces mediante el isomorfismo inverso ${\bar{h}_r}^{-1}$, se define el isomorfismo lineal

$${\bar{H}_r}^{-1}: M_{h\times k}\left(h_r\left(GF(p^n)\right)\right) \to M_{h\times k}\left(GF(p^n)\right)$$

de modo que, para toda

$$B \in M_{h\times k}\left(h_r\left(GF(p^n)\right)\right), B = \left(b_{ij}\right), {\bar{H}_r}^{-1}(B) = {\bar{h}_r}^{-1}\left(b_{ij}\right) = \text{A}.$$

Esta es, obviamente, la matriz tal que $H_r(A) = B$.

Se hace notar que el isomorfsimo ${\bar{H}_r}^{-1}$ es el inverso del isomorfismo lineal $H_r : M_{h\times k}\left(GF(p^n)\right) \to M_{h\times k}\left(GF(p^m)\right)$ que resulta de la restricción del codominio de $H_r : M_{h\times k}\left(GF(p^n)\right) \to M_{h\times k}\left(GF(p^m)\right)$ al subespacio imagen

$$H_r(M_{h\times k}\left(GF(p^n)\right)) = M_{h\times k}\left(h_r\left(GF(p^n)\right)\right)$$

Resulta fácil probar que si $B \in M_{h\times k}\left(h_r\left(GF(p^n)\right)\right)$ es MDS, entonces $A = {\bar{H}_r}^{-1}(B)$ también lo es. La demostración se basa en el hecho de que la imagen de una submatriz cuadrada, de determinante no nulo, es también de determinante no nulo, pues por ser ${\bar{H}_r}^{-1}$ un homomorfismo inyectivo, este transforma elementos no nulos en elementos no nulos.

3.4.2 Ejemplos de aplicación de los homomorfismos a la obtención de matrices MDS

A continuación se presentan dos ejemplos de aplicación de los homomorfismos de inmersión a la obtención de matrices MDS.

Ejemplo 3.4.1

Dada una matriz A MDS tal que $A \in GL_4(GF(2^8))$ donde $A = (a_{ij})$ con $i = \overline{1,4}, j = \overline{1,4}$, obtenga las matrices MDS $B \in GL_4(GF(2^8))$ y $C \in GL_4(GF(2^{16}))$.

Los campos $GF(2^4) = GF(2)(\alpha), GF(2^8) = GF(2)(\beta)$ y $GF(2^{16}) = GF(2)(\gamma)$ se obtienen como extensiones del campo $GF(2) = \{0,1\}$ siendo α, β y γ raíces de polinomios irreducibles sobre $GF(2)$ de grados 4, 8 y 16 respectivamente. Para garantizar que α, β y γ sean elementos primitivos de los campos $GF(2^4), GF(2^8)$ y $GF(2^{16})$ respectivamente se utilizan en este caso los polinomios primitivos sobre $GF(2)$

$$p_1(x) = x^4 + x + 1,$$

$$p_2(x) = x^8 + x^4 + x^3 + x^2 + 1,$$

$$p_3(x) = x^{16} + x^{15} + x^{14} + x^{12} + x^{10} + x^9 + x^4 + x^2 + 1\,.$$

Utilizando el s*oftware* BiGFSoP[1], herramienta computacional elaborada en el Laboratorio de Criptografía Académica de la UCLV, se obtuvo que las 4 funciones que insertan el campo $GF(2^4)$ en $GF(2^8)$ tienen la forma $h_i(\alpha) = \beta^{17i}$ donde i toma valores en el conjunto $\{1, 2, 4, 8\}$ y que las 8 funciones que insertan el campo $GF(2^8)$ en $GF(2^{16})$ tienen la forma $g_j(\alpha) = \beta^{257j}$ donde j toma valores en el conjunto $\{7, 14, 28, 56, 112, 224, 193, 131\}$.

Las funciones que representan los homomorfismos entre los campos $GF(2^4)$ y $GF(2^8)$, $GF(2^8)$ y $GF(2^{16})$ fueron obtenidas en el epígrafe 2.3.3.

Sea la matriz A

$$A = \begin{pmatrix} 1 & \alpha & \alpha^2 & \alpha^2+\alpha \\ \alpha & 1 & 1+\alpha & \alpha \\ \alpha^2 & 1+\alpha & 1 & 1 \\ \alpha^2+\alpha & \alpha & 1 & 1+\alpha^3 \end{pmatrix}$$

Si se expresan los elementos de la matriz A como potencias del elemento primitivo α del campo $GF(2^4)$ se obtiene que

$$A = \begin{pmatrix} 1 & \alpha & \alpha^2 & \alpha^5 \\ \alpha & 1 & \alpha^4 & \alpha \\ \alpha^2 & 1+\alpha & 1 & 1 \\ \alpha^5 & \alpha & 1 & \alpha^{14} \end{pmatrix}$$

Se selecciona uno de los homomorfismos de inmersión entre los campos $GF(2^4)$ y $GF(2^8)$, $GF(2^8)$ y $GF(2^{16})$ respectivamente $h_1(\alpha) = \beta^{17}$ y $g_1(\alpha) = (\gamma^{257})^7$. Se calculan $h_1(A) = h_1(a_{ij}) = h_1(\alpha^k) = (\beta^{17})^k$ y $g_1(B) = g_1(b_{ij}) = g_1(\beta^k) = (\gamma^{257})^{7k}$.

Se obtienen las matrices B y C

$$B = \begin{pmatrix} 1 & \beta^{17} & \beta^{34} & \beta^{85} \\ \beta^{17} & 1 & \beta^{68} & \beta^{17} \\ \beta^{34} & \beta^{68} & 1 & 1 \\ \beta^{85} & \beta^{17} & 1 & \beta^{238} \end{pmatrix}, C = \begin{pmatrix} 1 & \gamma^{30583} & \gamma^{61166} & \gamma^{21845} \\ \gamma^{30583} & 1 & \gamma^{56797} & \gamma^{30583} \\ \gamma^{61166} & \gamma^{56797} & 1 & 1 \\ \gamma^{21845} & \gamma^{30583} & 1 & \gamma^{34952} \end{pmatrix}$$

Se pueden escribir las matrices B y C representando los elementos del campo como polinomios en β y γ de grado menor o igual que 7 y 15 respectivamente, según una de las formas de representar los elementos de un campo finito dada en 1.1.1.

$$B = \begin{pmatrix} 1 & \beta^3+\beta^4+\beta^7 & \beta+\beta^2+\beta^3+\beta^6 & \beta+\beta^2+\beta^4+\beta^6+\beta^7 \\ \beta^3+\beta^4+\beta^7 & 1 & 1+\beta^3+\beta^4+\beta^7 & \beta^3+\beta^4+\beta^7 \\ \beta+\beta^2+\beta^3+\beta^6 & 1+\beta^3+\beta^4+\beta^7 & 1 & 1 \\ \beta+\beta^2+\beta^4+\beta^6+\beta^7 & \beta^3+\beta^4+\beta^7 & 1 & 1+\beta+\beta^3 \end{pmatrix}$$

C= $((1, \gamma^2+\gamma^4+\gamma^5+\gamma^6+\gamma^9+\gamma^{10}+\gamma^{11}+\gamma^{13}+\gamma^{15},$

$1+\gamma+\gamma^2+\gamma^3+\gamma^4+\gamma^5+\gamma^7+\gamma^9+\gamma^{13}+\gamma^{15}, 1+\gamma+\gamma^3+\gamma^6+\gamma^7+\gamma^{10}+\gamma^{11})$,

$$(\gamma^2+\gamma^4+\gamma^5+\gamma^6+\gamma^9+\gamma^{10}+\gamma^{11}+\gamma^{13}+\gamma^{15}, 1, 1+\gamma^2+\gamma^4+\gamma^5+\gamma^6+\gamma^9+\gamma^{10}+\gamma^{11}+\gamma^{13}+\gamma^{15}, \gamma^2+\gamma^4+\gamma^5+\gamma^6+\gamma^9+\gamma^{10}+\gamma^{11}+\gamma^{13}+\gamma^{15}),$$

$$(1+\gamma+\gamma^2+\gamma^3+\gamma^4+\gamma^5+\gamma^7+\gamma^9+\gamma^{13}+\gamma^{15}, 1+\gamma^2+\gamma^4+\gamma^5+\gamma^6+\gamma^9+\gamma^{10}+\gamma^{11}+\gamma^{13}+\gamma^{15}, 1, 1),$$

$$(1+\gamma+\gamma^3+\gamma^6+\gamma^7+\gamma^{10}+\gamma^{11}, \gamma^2+\gamma^4+\gamma^5+\gamma^6+\gamma^9+\gamma^{10}+\gamma^{11}+\gamma^{13}+\gamma^{15}, 1, 1+\gamma+\gamma^2+\gamma^4+\gamma^{10}+\gamma^{11}+\gamma^{12}+\gamma^{13}+\gamma^{14}))$$

Ejemplo 3.4.2

A partir de la matriz $B \in GL_4(GF(2^8))$ MDS del ejemplo anterior tal que $B = (b_{ij})$ con $i = \overline{1,4},\ j = \overline{1,4}$, obtener la matriz $A \in GL_4(GF(2^4))$.

Los campos $GF(2^8)\ y\ GF(2^4)$ están definidos como en el ejemplo anterior.

El subcampo imagen del homomorfismo $h_1(\alpha) = \beta^{17}$ es

$$h_1(GF(2^4)) = \{1,0,\beta^{17},\beta^{34},\beta^{51},\beta^{68},\beta^{85},\beta^{102},\beta^{119},\beta^{136},\beta^{153},\beta^{170},\beta^{187},\beta^{204},\beta^{221},\beta^{238}\}$$

Sea $h_1^{-1}(0) = 0, h_1^{-1}(\beta^j) = \alpha^{\frac{j}{17}}$ y $h_1^{-1}(B) = h_1^{-1}(b_{ij}) = h_1^{-1}(\beta^k) = \alpha^{\frac{k}{17}}$

La matriz A tiene la forma

$$A = \begin{pmatrix} 1 & \alpha & \alpha^2 & \alpha^5 \\ \alpha & 1 & \alpha^4 & \alpha \\ \alpha^2 & 1+\alpha & 1 & 1 \\ \alpha^5 & \alpha & 1 & \alpha^{14} \end{pmatrix} = \begin{pmatrix} 1 & \alpha & \alpha^2 & \alpha^2+\alpha \\ \alpha & 1 & 1+\alpha & \alpha \\ \alpha^2 & 1+\alpha & 1 & 1 \\ \alpha^2+\alpha & \alpha & 1 & 1+\alpha^3 \end{pmatrix}$$

En este epígrafe se desarrolló un procedimiento para dadas matrices rectangulares y MDS A de tamaño $h \times k$ de componentes en un campo finito $GF(p^n)$, obtener matrices MDS del mismo orden, pero de entradas en otro campo finito $GF(p^m)$, siendo n un divisor de m a partir de la extensión de un homomorfismo de campos finitos, que puede ser visto como homomorfismo entre espacios vectoriales, a un homomorfismo entre los espacios vectoriales de las matrices de tamaño $h \times k$ y con elementos sobre dichos campos finitos.

Esta aplicación de los homomorfismos de inmersión a la obtención de matrices MDS, que son de alto impacto en el diseño de algoritmos criptográficos, es novedosa.

3.5 Conclusiones del capítulo

En este capítulo utilizando otra caracterización de las matrices MDS de orden 4 sobre $GF(2^m)$, se diseñó un nuevo algoritmo para la generación aleatoria de matrices MDS, de orden 4 sobre el campo $GF(2^8)$. Se realizó un análisis comparativo de las complejidades del nuevo algoritmo y el algoritmo FREDIA. Se presentaron dos variantes del nuevo algoritmo que disminuyen la complejidad del algoritmo para el cálculo de la matriz MDS y de su inversa, al reducir la cantidad de parámetros de entrada, no necesitar de la solución de ecuaciones, ni de la restricción en cuanto a la cardinalidad del conjunto de submatrices cuadradas que se obtienen hasta el cálculo de la fila $n-ésima$ de la matriz. Se determinaron expresiones recursivas para el cálculo de los elementos de la matriz y de su inversa. Se mostró una aplicación novedosa de los homomorfismos a la obtención de matrices MDS.

CONCLUSIONES Y RECOMENDACIONES

Conclusiones

1. Se realizó un estudio sobre los campos finitos como elemento base de la investigación, así como el tratamiento dado en la literatura a los homomorfismos de inmersión, los métodos de obtención de las matrices MDS y su aplicación al diseño de algoritmos criptográficos.
2. Se demostró el teorema de la condición necesaria y suficiente para la existencia de los homomorfismos de inmersión entre campos finitos,utilizando otra forma de representar los elementos de un campo finito como elementos de una sucesión recurrente lineal.
3. Partiendo de la condición necesaria y suficiente para la existencia de los homomorfismos de inmersión de campos finitos, la relación existente entre los homomorfismos de inmersión y el automorfismos de *Frobenius* de los campos que son homomorfos y para los cuales se quieren hallar las funciones representantes de sus homomorfismos, la relación entre los homomorfismos de inmersión y las raíces de los polinomios definidores de los campos, se diseñó el algoritmo EVOR para la determinación de los homomorfismos de inmersión entre campos finitos.
4. Se diseñó un nuevo algoritmo para la generación aleatoria de matrices MDS que, al igual que el algoritmo FREDIA, permite seleccionar las matrices MDS del conjunto de todas las matrices MDS posibles y al mismo tiempo ganar en eficiencia.

Se diseñaron dos variantes de este nuevo algoritmo para la generación aleatoria de matrices MDS, que disminuyen la complejidad del algoritmo, al reducir la cantidad de parámetros de entrada, el número de operaciones que realizan y eliminar la restricción en cuanto a la cardinalidad del conjunto de submatrices cuadradas que se obtienen hasta el cálculo de la fila $n-ésima$ de la matriz.

5. Se propuso un procedimiento para la obtención de matrices MDS a partir de otras matrices MDS con coeficientes sobre campos homomorfos, considerándose esto como una aplicación novedosa.

Todo lo anterior permite afirmar que la hipótesis de investigación es válida y se resolvió el problema de investigación planteado y permite formular algunas recomendaciones.

Recomendaciones

1. Implementar el algoritmo EVOR para campos finitos no binarios.
2. Extender el nuevo algoritmo de generación de matrices MDS pertenecientes a $GL_n\left(GF(2^8)\right)$ para $n > 4$.
3. Obtener de manera más eficiente el polinomio y la matriz inversas cuando el coeficiente $b_{1,0}$ es tomado como incógnita en el algoritmo de generación aleatoria de matrices MDS.
4. Diseñar un algoritmo que integre la generación de matrices MDS y el uso de los homomorfismos de inmersión entre campos finitos.
5. Utilizar los resultados de la tesis en el diseño de algoritmos criptográficos.

BIBLIOGRAFÍA

1. ABD-ELGHAFAR, A. R., DIAA, A. & MOHAMMED, F. Generation of AES key dependent S-boxes using RC4 algorithm. 13 th International Conference on Aerospace Sciences & Aviation Technology, 2009. 26-28.
2. ABHIRAM, L., SRIROOP, B., GOWRAV, L., PUNITH, K. H. & LAKKANNAVAR, M. C.2015. FPGA implementation of dual key based AES encryption with key Based S-Box generation. Computing for Sustainable Global Development (INDIA Com), 2nd International Conference 2015. IEEE, 577-581.
3. AGUIRRE, R. 2003. Curso de Seguridad Informática y Criptografía. Madrid.
4. AHMED, F. & ELKAMCHOUCHI, D. STRONGEST. AES with S-Boxes bank and dynamic key MDS matrix (SDK-AES). International Journal of Computer and Communication Engineering, Vol. 2, No. 4, July 2013.
5. AHO, J A. V., HOPCROFT, E. & ULLMAN, J. D. Data Structures and Algorithms Addison-Wesley, Massachusetts, 1983.
6. ANDRADE, C., & CEDILLO, D. (2004). Criptografía en la seguridad de sistemas informáticos. http://dspace.uazuay.edu.ec/bitstream/datos/2142/1/04282.pdf.
7. ARITA, S. & NAKASATO, S. Fully Homomorphic Encryption for Point Numbers. Cryptology ePrint Archive: Report 2016/402. http://eprint. iacr. org.
8. ARRAG S., HAMDOUN A., TRAGHA A., & KHAMLICH S. E. Implementation of stronger AES by using dynamic S-box dependent of master key Journal of Theoretical and Applied Information Technology. Marruecos. 2013.

9. ARMKNECHT, F., KATZENBEISSER, S. & PETER, A. 2012. Shift-type homomorphic encryption and its application to fully homomorphic encryption. Progress in Cryptology-AFRICACRYPT 2012. Springer.
10. ARMKNECHT, F., BOYD, C., CARR, C., GJØSTEEN, K., JÄSCHKE, A., REUTER, C. A., & STRAND, M. (2015). A guide to fully homomorphic encryption. IACR Cryptology ePrint Archive (2015/1192).
11. ARRAG, S., HAMDOUN, A., TRAGHA, A. & KHAMLICH, S. E. 2013. IMPLEMENTATION OF STRONGER AES BY USING DYNAMIC S-BOX DEPENDENT OF MASTER KEY. Journal of Theoretical & Applied Information Technology, 53.
12. AUGOT, D. & FINIASZ, M. Exhaustive search for small dimension recursive MDS diffusion layers for block ciphers and hash functions. Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on, 2013. IEEE, 1551-1555.
13. AUGOT, D. & FINIASZ, M. Direct construction of recursive MDS diffusion layers using shortened BCH codes. Fast Software Encryption, 2014. Springer, 3-17.
14. BAI, J. & WANG, D. The Lightest 4x4 MDS Matrices over GL(4, F_2)

    Cryptology ePrint Archive, Report 2016/686, 2016. http://eprint. iacr. org.
15. BARLA, J. L. 2012. Cuerpos finitos Disponible en rinconmatematico.com/foros/index. php?action=dlattach;topic
16. BARRETO, L. & RIJMEN, V. 2000. The WHIRLPOOL Hashing Function. First open NESSIE Workshop. Leuven, Belgium (Vol. 13, p. 14).

17. BARRETO, P. & RIJMEN, V. 2000. The Anubis block cipher, NESSIE Algorithm Submission (2000).
18. BARRETO, P. & RIJMEN, V. 2000. The Khazad Legacy-Level Block Cipher, In Proceedings First open NESSIE Workshop. Leuven
19. BARRETO, P. S. & RIJMEN, V. 2011. Whirlpool. Encyclopedia of Cryptography and Security. Springer.
20. BARTEE, T. C. & SCHNEIDER, D. I. 1963. Computation with finite fields. Information and Control, 6, 79-98.
21. BERLEKAMP, E. R. 1968. Algebraic Coding Theory, McGraw-Hill.
22. BOGOS, S., GASPOZ, J., & VAUDENAY, S. Cryptanalysis of a Homomorphic Encryption Scheme. IACR Cryptology ePrint Archive 2015, 775
23. BOS, J. W., MONTGOMERY, P. L., SHUMOW, D. & ZAVERUCHA, G. M. 2013. Montgomery multiplication using vector instructions. Selected Areas in Cryptography--SAC 2013. Springer.
24. BOSMA, W., CANNON, J. & PLAYOUST, C. 1997. The Magma algebra system I: The user language. Journal of Symbolic Computation, 24, 235-265.
25. BOSMA, W., CANNON, J. & STEEL, A. 1997. Lattices of compatibly embedded finite fields. Journal of Symbolic Computation, 24, 351-369.
26. BOTANAA, F., ESCRIBANOB, J. & ABÁNADESC, M. 2011. SAGE: una aplicación libre para matemáticas.
27. BRICKELL, E., GORDON, D., MCCURLEY, K., WILSON, D. (1992). Fast exponentiation with precomputation. In Proceedings of Eurocrypt'92. Hungary, Balatonf¨ured.

28. CANNON, J. 2006. Handbook of Magma functions. In: Editors, W. B. (ed.) Language, Aggregates and Semigroups. Sydney: disponible en https://secure.msri.org/about/computing/docs/magma/HandbookVolume01.pdf.
29. ÇETIN, G. S., DAI, W., DORÖZ, Y., & SUNAR, B. (2015). Homomorphic autocomplete. IACR Cryptology ePrint Archive 2015, 1194.
30. CHARLES, D., JAIN, K. & LAUTER, K. 2009. Signatures for network coding. International Journal of Information and Coding Theory, 1, 3-14.
31. CHEN, C. L. 1982. Formulas for the solution of algebraic equations over $GF(2^m)$. Journal: IEEE Transactions on Information Theory - TIT vol. 28.
32. CHEN, Y. 2005. Computation of Initial State for Tail-Biting Trellis. A thesis presented to the faculty of Engineering and Technology of Ohio University. In partial fulfillment of the requirements for the degree Master of Science.
33. CHEON, J.H. & STEHLE, D. Fully Homomorphic Encryption over the Integers Revisited. Cryptology ePrint Archive, Report 2016/837, 2016. http://eprint. iacr. org.
34. CHEON, J.H., KIM, M., & LAUTER, K. Homomorphic Computation of Edit Distance .Cryptology ePrint Archive, Report 2015/132, 2015. http://eprint. iacr. org.
35. CHOR, B. & RIVEST, R. L. 1988. A knapsack-type public key cryptosystem based on arithmetic in finite fields. Information Theory, IEEE Transactions on, 34, 901-909.
36. CID, C., MURPHY, S. & ROBSHAW, M. 2006. Algebraic Aspects of the Advanced Encryption Standard, publisher 2006, In: Security, S. A. I. I. (ed.). Springer.

37. CIET, M., JOYE, M., LAUTER, K. & MONTGOMERY, P. L. 2006. Trading inversions for multiplications in elliptic curve cryptography. Designs, codes and cryptography, 39, 189-206.
38. CLEAR, M., HUGHES, A. & TEWARI, H. 2013. Homomorphic Encryption with Access Policies: Characterization and New Constructions. 61-87.
39. CNUDDE, T., REPARAZ, O., BILGIN, B., NIKOVA, S., NIKOV, V. & RIJMEN, V. Masking AES with d+1 Shares in Hardware. Cryptology ePrint Archive: Report 2016/631. http://eprint. iacr. org.
40. CONTINI, S., RIVEST, R. L., ROBSHAW, M. B. & YIN, Y. L. 1998. The Security of the RC6 Block Cipher. in: https://www.grc.com/r&d/rc6-security.pdf.
41. CUELLAR, O., MADARRO, E., FREYRE, P. & PALENCIA, G. 2015. Algoritmo para la determinación de los homomorfismos de inserción entre campos finitos de diferente cardinalidad. Boletín de la Sociedad Cubana de Matemática y Computación, Número Especial 2015.
42. CUELLAR, O., MORGADO, E., MADARRO, E. & SOSA, G. 2015. Relación entre el automorfismo de Frobenius y los homomorfismos de inmersión del campo $GF(p^n)$ en el campo $GF(p^m)$. Revista Ciencias Matemáticas, Vol 29, 127-130.
43. CUELLAR, O. & SOSA, G. 2013. Inmersión de un campo de Galois GF(p^n), en otro de mayor cardinalidad. Revista Ciencias Matemáticas. Habana .Cuba.Vol. 27. Número 2, 2013, pp 70-77.
44. CUELLAR, O., MADARRO, E., FREYRE, P. & SOSA, G. 2016. A Reduced table of the Zech´s logarithm. Journal of Advances in Mathematics. Vol 12, No 7, 2016.pp 6422-6428. https://cirworld.com/index.php/jam/issue/view/578

45. CULLER, M. 2005. Notes on Algebra (Fields). in: http://homepages.math.uic.edu/~culler/notes/fields.pdf.
46. DAEMEN, J. 2002. The Design of Rijndael. AES-The Advanced Encryption Standard. Heidelberg, Alemania: Springer-Verlag.
47. DAEMEN, J., KNUDSEN, L. & RIJMEN, V. The block cipher Square. Fast software encryption, 1997. Springer, 149-165.
48. DAEMEN, J. & RIJMEN, V. 2003. A new keystream generator MUGI. Proceedings of the 9th International Workshop on Fast Software Encryption, 2365, 179-194.
49. DAEMEN, J. & RIJMEN, V. 2005. Rijndael/AES. Encyclopedia of Cryptography and Security. Springer.
50. DAEMEN, J., KNUDSEN, L.R. & RIJMEN, V.1997. The block cipher, SQUARE. 4th Fast Software Encryption Workshop. LNCS. Springer-Verlag, 1267, pp.149-165.
51. DAS, A. 1999. Galois Field computations: Implementation of a library and a study of the discrete logarithm problem. A thesis submitted for the degree of Doctor of Philosophy in the Faculty of Engineering Computer Science and Automation Indian Institute of Science. Bangalore 560 012.122 pp
52. DEHNAVI, S. M., RISHAKANI, A. M., SHAMSABAD, M. M., MAIMANI, H., & PASHA, E. 2014. Construction of New Families of MDS Diffusion Layers. IACR Cryptology ePrint Archive, 2014, 11.
53. DERLER, D. & SLAMANIG, D. Key-Homomorphic Signatures and Applications to Multiparty Signatures. Cryptology ePrint Archive, Report 2016/792, 2016. http://eprint. iacr. org.

54. DÍAZ, J. F., SIERRA, J. M. R., & URIARTE, E. A. (2005). Introducción al álgebra.https://dialnet.unirioja.es/servlet/libro?codigo=277812

55. EL-SHEIKH, H. M., EL-MOHSEN, O. A., ELGARF, S. T. & ZEKRY, A. 2012. A New Approach for Designing Key-Dependent S-Box Defined over $GF\ (2^4)$ in AES. International Journal of Computer Theory and Engineering, 4, 158.

56. ELUMALAI, J. & REDDY, A. R. 2011. "Improving Diffusion Power of AES Rijndael with 8x8 MDS Matrix". International Journal of Scientific & Engineering Research 2.

57. FAHMY, A., SHAARAWY, M., EL-HADAD, K., SALAMA, G. & HASSANAIN, K. A proposal for A key-dependent AES. 3rd International Conference: Sciences of Electronic, Technologies of Information and Telecommunications. Tunisia: SETIT, 2005.

58. FERGUSON, N., SCHROEPPEL, R. & WHITING, D. A simple algebraic representation of Rijndael. Selected Areas in Cryptography, 2001. Springer, 103-111.

59. FIORE, D., MITROKOTSA, A., NIZZARDO, L. &PAGNIN, E. Multi-Key Homomorphic Authenticators. Cryptology ePrint Archive, Report 2016/804, 2016. http://eprint. iacr. org.

60. FRALEIGH, J. B., A First Course in Abstract Algebra. Addison Wesley 2006, 7th edition. Part VI y X.

61. FREYRE, P., DÍAZ, N. & MORGADO, E. 2009. Fast algorithm for the multiplication of a row vector by a randomly selected matrix. Journal of Discrete Mathematical Science & Cryptography, 12, 533-549.

62. FREYRE, P., DÍAZ, N. & MORGADO, E. 2009. “Some algorithms related to matrices with entries in a finite field”. Journal of Discrete Mathematical Sciences & Cryptography, 12, 509-519.
63. FREYRE, P., DÍAZ, N. & MORGADO, E. 2010. “Algoritmo para la generación aleatoria de matrices booleanas invertibles”. Revista Investigación Operacional, 31, 258-263.
64. FREYRE, P., DIAZ, N. & MORGADO, E. 2012. Algoritmo para la generación aleatoria de matrices invertibles. Revista Ciencias Matemáticas, Vol 26.pp131-134.
65. FREYRE, P., DÍAZ, N., DÍAZ, R. & PEREZ, C. Random Generation of MDS matrices. Mathematical Aspects of Cryptography. Proceeding CD. CTCrypt 2014. Moscú, 2014.
66. FREYRE, P., DÍAZ, N. & CUELLAR, O. 2015. Variations to the cryptographics algorithms AES and Twofish. Cryptology ePrint Archive: Report 2015/1080 .avaliable at http://eprint. iacr. org/2015/1080
67. GAAL, L. 1998. Classical Galois Theory: with examples, American Mathematical Soc.
68. GABAUTZ, G. & SUNAR, B. 2005. Robust Finite Field Arithmetic for Fault-Tolerant Public-Key Cryptography Worcester Polytechnic Institute. Massachusetts, FDT05.
69. GATHEN, V. Z. & PANARIO, D. Factoring Polynomials Over Finite Fields Academic Press 2001. https://people.csail.mit.edu/dmoshkov/courses/codes/poly-factorization.pdf.

70. GAO, S., GATHEN, V.Z., PANARIO, D. & SHOUP, V.2000. Algorithms for Exponentiation in Finite Fields. 2000. Journal Simbolic Computation. Vol 29.pp 879-889.doi:10.1006/jsco.1999.0309. http://www.ideallibrary.com .
71. GENTRY, C. 2009. A Fully Homomorphic Encryption Scheme. Ph.D. thesis Stanford University.
72. GENTRY, C. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009, pages 169–178. ACM, 2009.
73. GENTRY, C., HALEVI, S. Implementing gentry's fully-homomorphic encryption scheme. In Kenneth G. Paterson, editor, Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings, volume 6632 of Lecture Notes in Computer Science, pages 129–148. Springer, 2011.
74. GOLLMANN, D., RIJMEN, V. & DAEMEN, J. 1996. The cipher SHARK. Fast Software Encryption. LNCS 1039.Springer-Verlag, 99-112.
75. GOLOMB, S. W. 1982. Shift register sequences, Aegean Park Press.
76. GOUTAM, P. & SUBHAMOY, M. 2012. RC4.Streams Cipher and Its Variants. In: ROSEN, D. M. I. A. S. E. K. H. (ed.). CRC Press. Taylor & Francis Group.
77. GRAHAM, R. L., KNUTH, D. E. & PATASHNIK, O. Concrete Mathematics Addison-Wesley, 1990.

78. GRIGORESCU, E., KOPPARTY, S. & SUDAN, M. 2006. Local decoding and testing for homomorphisms. Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques. Springer.
79. GROUP, T. G. 2015. GAP-Reference Manual. In: http://www.gap-system.org/manuals/doc/ref/manual.pdf, t. g. g. d. e. (ed.).
80. GUO, J., PEYRIN, T. & POSCHMANN, A. 2011. The PHOTON family of lightweight hash functions. Advances in Cryptology–CRYPTO 2011. Springer.
81. GUPTA, K. C. & RAY, I. G. 2013. On constructions of involutory MDS matrices. Progress in Cryptology–AFRICACRYPT 2013. Springer.
82. GUPTA, K. C. & RAY, I. G. 2013. On constructions of MDS matrices from companion matrices for lightweight cryptography. Security Engineering and Intelligence Informatics. Springer.
83. GUPTA, K. C. & RAY, I. G. 2014. On constructions of circulant MDS matrices for lightweight cryptography. Information Security Practice and Experience. Springer.
84. GUPTA, K. C. & RAY, I. G. 2015. Cryptographically significant MDS matrices based on circulant and circulant-like matrices for lightweight applications. Cryptography and Communications, 7, 257-287.
85. HOSSEINKHANI, R. & JAVADI, H. H. S. 2012. Using cipher key to generate dynamic S-box in AES cipher system. International Journal of Computer Science and Security (IJCSS), 6, 19-28.
86. Huber K. 1990. Some Comments on Zech's Logarithms. IEEE Transactions on Information Theory. Vol. 36, Issue: 4, pp 946 - 950.

87. Huguet, L., Rifa, J., Tena, J.G., 2012. Cuerpos finitos. Material docente de la Universidad Oberta de Catalunya. Disponible http://hdl.handle.net/10609/50542
88. HUBIČKA, J. & NEŠETŘIL, J. 2009. Homomorphism and embedding universal structures for restricted classes. arXiv preprint arXiv:0909.4939.
89. HUFFMAN, V. W. C. 2003. Fundamentals of Error -Correcting Codes, New York.
90. JACOB G., MURUGAN A. &VIOLA I. Towards the Generation of a Dynamic Key-Dependent S-Box to Enhance Security. IACR Cryptology. ePrint Archive. 2015.
91. JOYNER D. &KIM, J. L. 2011. Selected Unsolved Problems in Coding Theory. Springer Science +Business Media. LLC
92. JUNOD, P. & MACCHETTI, M. Revisiting the IDEA philosophy. Fast Software Encryption, 2009. Springer, 277-295.
93. JUNOD, P. & VAUDENAY, S. Perfect diffusion primitives for block ciphers. Selected Areas in Cryptography, 2004. Springer, 84-99.
94. KATAGI, M., & MORIAI, S. 2011. The 128-Bit Blockcipher CLEFIA (No. RFC 6114).
95. KAUFFMAN, L. H. & SALEUR, H. 1991. Free fermions and the Alexander-Conway polynomial. Communications in mathematical physics, 141, 293-327.
96. KAZLAUSKAS, K. & KAZLAUSKAS, J. 2009. Key-dependent S-box generation in AES block cipher system. Informatica, 20, 23-34.
97. KHAMLICH, E. 2013. Implementation of stronger AES by using dynamic s-box dependent of master key. Journal of Theoretical and Applied Information Technology, 53.

98. KIM, J., HONG, S. & PRENEEL, B. 2007. Related-key rectangle attacks on reduced AES-192 and AES-25. 4593, 225-241.

99. KNUTH, D. E. (1968). The Art of Computer Programming. Vol. 1 y 2: Fundamental Algorithms, Seminumerical Algorithms. Addision-Wesley Publishing Company.

100. KNUTH, D. E. 1998. The art of computer programming: sorting and searching.Vol. 3. Pearson Education.

101. KOHEL, D. R. 2006. A Brief Magma Tutorial. In: INSTITUTE, M. S. R. (ed.). https://etudes.univrennes1.fr/digitalAssets/26/26222_tutoriali_magma.pdf.

102. KOLAY, S. & AND MUKHOPADHYAY, D. 2014. Lightweight difusion layer from kth root of the MDS matrix. IACR Cryptology. ePrint Archive, 2014, 498.

103. KRISHNAMURTHY, G.N. & RAMASWAMY V. 2008. Making AES stronger: AES with key dependent S-box. IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.9, pp 388-398.

104. KUMAR, R. 2009. Rings and Fields Theorems. PMATH 334.Intro to Rings and Fields. Fall 2009.

105. KWON, D., SUNG, S. H., SONG, J. H. & PARK, S. 2005. Design of block ciphers and coding theory. Trends in Mathematics Information Center for Mathematical Sciences, 8, 13-20.

106. LACAN, J. & FIMES, J. 2004. Systematic MDS erasure codes based on Vandermonde matrices. IEEE Communications Letters, 8, 570-572.

107. LAI, R. W. F., TAI,R.K.H., WONG, W. H. & CHOW,S.S.M. A Zoo of Homomorphic Signatures: Multi-Key and Key-Homomorphism. Cryptology ePrint Archive, Report 2016/837, 2016. http://eprint. iacr. org.

108. LAUTER, K. E., CHARLES, D. X. & JAIN, K. 2010. Digital signature for network coding. Google Patents.

109. LAWRENCE, C. 2006. Introduction to Cryptography with Coding Theory. Second Edition. Washington Person Prentice Hall. Pearson Education.

110. LENSTRA, J. H. W. 1991. Finding isomorphisms between finite fields. Math.Comp. 56 (193), 329-347.

111. LIDL, R. & NIEDERREITER, H. 1994. Introduction to finite fields and their applications, Cambridge University Press.

112. LIDL, R. & NIEDERREITER, H. 1998. Konechnye Polia, Mir.Moscú.

113. LIU, M. & SIM, S .M. Lightweight MDS Generalized Circulant Matrices (Full Version). Cryptology ePrint Archive: Report 2016/186. http://eprint. iacr. org.

114. LUCENA, M. 2010. Criptografía y Seguridad en Computadores.

115. LUX, K.M. & SZÖKE, M. Computing Homomorphism Spaces between Modules over Finite Dimensional Algebras.. Experimental Mathematics. Vol. 12, No. 1, page 91, 2002. http://www.emis.de/journals/EM/expmath/volumes/12/12.1/pp91_98.pdf

116. MAC WILLIAMS, F. J. & SLOANE, N. J. A. 1977. The theory of error correcting codes, Elsevier.

117. MADARRO, E., CUELLAR, O., SOSA, G., PERFETTI, L., MORGADO, E. & Y. C. R. 2013. Comparación de los métodos de solución de ecuaciones cuadráticas y

cúbicas sobre campos finitos de característica dos. Boletín de la Sociedad Cubana de Matemática y Computación. Volumen especial del 2013.

118. MAHMOUD, E. M., ABD, A., HAFEZ, E. & ELGARF, T. A. 2013. Dynamic AES-128 with key-dependent S-box. International Journal of Engineering Research and Applications, págs. 1662-1670. Egypt.
119. MALIK, M. Y. & NO, J-S. Dynamic MDS matrices for substantial cryptographic strength. Seoul National University. 2011. https://arxiv.org/ftp/arxiv/papers/1108/1108.6302.pdf.
120. MANDAL, S. 2013. Automorphisms and Galois Theory. GaloisTheory. University of Kansas. USA: Lawrence KS 66045.
121. MC ELIECE, R. J. 1977. The theory of information and coding in volume 3 of Encyclopedia of Mathematics and its applications. Addison-Wesley Reading, Mass.
122. MENEZES, V. S. 1996. Handbook of Applied Cryptography.
123. MOHAMMAD, F. Y., ROHIEM, A. E. & ELBAYOUMY, A. D. A novel S-box of AES algorithm using variable mapping technique. Proceedings of the 13th International Conference on Aerospace Sciences and Aviation Technology, 2009. 1-10.
124. MONTGOMERY, P. L. 1985. Modular multiplication without trial division. Mathematics of Computation, 44, 519-521.
125. MONTGOMERY, P. L. 1992. An FFT extension of the elliptic curve method of factorization. UNIVERSITY OF CALIFORNIA Los Angeles.
126. MULLEN, G. & PANARIO, D. 2013. Handbook of Finite Fields, CRC Pres. Taylor& Francis Group.

127. MURTAZA, G. & IKRAM, N. New Methods of Generating MDS Matrices. Proceedings of International Cryptology Workshop and Conference, 2008.

128. MURTAZA, G. & IKRAM, N.2011. Direct exponent and scalar multiplication classes of an MDS matrix. IACR Cryptology. ePrint Archive, 2011,151.

129. MURTAZA G., KHAN A. A., ALAM S. W. & FAROOQ A. 2011.Fortification of AES with dynamic mix-column Transformation. National University of Science and Technology. Islamabad. Pakistan.

130. NAKAHARA, J. & ABRAHAO, E. 2009. A New Involutory MDS Matrix for the AES. IJ Network Security, 9, 109-116.

131. PANARIO, D. 2015. Minicourse on Arithmetic in Finite Fields and Cryptographical Applications. Congreso Internacional COMPUMAT 2015 UCI. Habana. noviembre 2015

132. PETERSON, W. W. & WELDON, J. E. 1972. Error Correcting codes, New York, John Wiley and Sons.

133. PEYRIN, T. Lightweight MDS Involution Matrices. Fast Software Encryption: 22nd International Workshop, FSE 2015, Istanbul, Turkey, March 8-11, 2015, Revised Selected Papers, 2015. Springer, 471.

134. POOLE, D. 2010. Linear Algebra. A Modern Introduction. Cengage Learning. Third edition ISBN**:** 9780538735452 / 0538735457.

135. PREUKSCHAT, A. 2014. Criptografía simétrica: Sistemas de cifra con clave secreta. Disponible en https://www.oroyfinanzas.com/2014/01/criptografia-simetrica-sistemas-cifra-clave-secreta/

136. QUTTINEH, N.-H. 2003. Computational complexity of finite field multiplication. Linköping University, Department of Electrical Engineering. Student thesis.105 p. http://www.diva-portal.org/smash/record.jsf?pid= diva2:19295

137. RAINBOLT, J. G. & GALLIAN, J. 2003. Abstract Algebra with GAP. A Manual to be used with Contemporary Abstract Algebra. 5th Edition .Disponible en http://college.cengage.com/mathematics/gallian/abstract_algebra/5e/ shared/gap/gap_manual.pdf.

138. RANDAL, D. 2011. Efficient generation of random non-singular matrices [Online]. Available: http://citeseer.ist.psu.edu. [Accessed Abril.

139. RANI, M. M. 2013. J. Cyclic codes of lenth n over GF (q) q-cyclotomic cosets modulo n and application of burnside's lemma. International Journal of Scientific and Research Publications, *3*(3), 1-5.

140. RAY, K. C. G. A. I. G. 2013. On Constructions of Involutory MDS Matrices. Lecture Notes in Computer Science. Springer. Progress in Cryptology - AFRICACRYPT 2013.6th International Conference on Cryptology in Africa. Proceedings, vol.7918, 43-60.

141. RAZA, S., MALIK, N. S., SHAKEEL, A. & KHAN, M. I. 2013. Implementation and Comparative Analysis of the Fault Attacks on AES. Int. Arab J. Inf. Technol., 10, 625-634.

142. REBOLL CABALLERO, H. 2016. Análisis comparativo de variantes de AES dinámico. Tesis presentada en opción al título de Máster en ciencias en la Facultad de Matemática y Computación. Universidad de la Habana.

143. RIBNIKOV, K. 1987. Historia de las matemáticas, Moscú, Editorial Mir.

144. RIJMEN, V., DAEMEN, J., PRENEEL, B., BOSSELAERS, A. & DE WIN, E. The cipher SHARK. Fast Software Encryption, 1996. Springer, 99-111.

145. RIVEST, R. L., ROBSHAW, M. B., SIDNEY, R. & YIN, Y. L. 1998. The RC6. Block Cipher. disponible en https://people.csail.mit.edu/rivest/pubs/RRSY98.pdf.

146. RIVEST, R. L., ADLEMAN, L. & DERTOUZOS, M. L.1978. On data banks and privacy homomorphisms. Foundations of Secure Computation, Academia Press, pages 169–179.

147. RIVEST, R. L., SHAMIR, A., & ADLEMAN, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, *21*(2), 120-126.

148. ROZANSKY, L. & SALEUR, H. 1992. Quantum field theory for the multi-variable Alexander-Conway polynomial. Nuclear Physics B, 376, 461-509.

149. RUIZSANCHEZ, C. P. 2014. Matrices MDS y Funciones Booleanas Vectoriales. Diseño de un Algoritmo criptográfico. Tesis presentada en opción al título de Master en Ciencias Matemáticas Universidad de la Habana.

150. SAJADIEH, M., DAKHILALIAN, M., MALA, H. & OMOOMI, B. 2012. On construction of involutory MDS matrices from Vandermonde Matrices in $GF\ (2^{q})$. Designs, Codes and Cryptography, 64, 287-308.

151. SAJADIEH, M., DAKHILALIAN, M., MALA, H. & SEPEHRDAD, P. Recursive diffusion layers for block ciphers and hash functions. Fast Software Encryption, 2012. Springer, 385-401.

152. SALEEM, R., MALIK, N. S., SHAKEEL, A. & KHAN, M. I. 2012. Implementation and Comparative Analysis of the Fault Attacks on AES. Int. Arab J. Inf. Technol. **10**(6): 625-634.

153. SALINAS ROSALES, M. (2012). Temario Tópicos Selectos de la Computación: Introducción a la Criptografía. http://www.repositoriodigital.ipn.mx/handle/123456789/6840

154. SANTIS, A. D., VAUDENAY, S. & SCHNORR, C. 1995. Black Box Cryptanalysis of Hash Networks based on Multipermutations. Advances in Cryptology-EUROCRYPT 94.Proceedings. LNCS, Vol. 950, pp47-57.

155. SAWOLLEK, J. 1999. On Alexander-Conway polynomials for virtual knots and links. arXiv preprint math/9912173.

156. SCHNEIER, B., KELSEY, J., WHITING, D., WAGNER, D., HALL, C. & FERGUSON, N. 1998. Twofish: A 128-bit block cipher. NIST AES Proposal, 15.

157. SCHNEIER, B., KELSEY, J., WHITING, D., WAGNER, D., HALL, C. & FERGUSON, N. 1999. The Twofish encryption algorithm: a 128-bit block cipher, John Wiley & Sons, Inc.

158. SEON, J. 2011. Dynamic MDS Matrices for Substantial Cryptographic Strength.

159. SHANNON, C. E. 1949. Communication Theory of Secrecy Systems. Bell Syst. Technical J., 28, 656-715.

160. SHIRAI, T., SHIBUTANI, K., AKISHITA, T., MORIAI, S. & IWATA, T. 2007. The 128-bit blockcipher CLEFIA. In International Workshop on Fast software encryption, 2007. Springer Berlin Heidelberg, 181-195.

161. SHOUP, V. 1990. New algorithms for finding irreducible polynomials over finite field. Math.Comp. 54, 435-447.

162. SIM, S. M., KHOO, K., OGGIER, F. & PEYRIN, T. Lightweight MDS involution matrices. Fast Software Encryption, 2015. Springer, 471-493.

163. SIMS, C. C. 2003. Recursive Difusion Layers for Block Ciphers and Hash Functions [Online]. Available: http://www.math.rutgers.edu/˜sims/publications [Accessed Noviembre 2014].

164. SKVARCIUS, R & ROBINSON, W. B. Discrete Mathematics with Computer Science Applications Benjamin/Cummings, Menlo Park, California, 1986.

165. SMART, N. P. & VERCAUTEREN, F. 2010. Fully homomorphic encryption with relatively small key and ciphertext sizes. Public Key Cryptography–PKC 2010. Springer.

166. STEWART, I. N. 2015. Galois Theory, CRC Press.

167. TANPING, Z., XIAOYUAN, Y., WEI, Z., & LIQIANG, W. Efficient Fully Homomorphic Encryption with Circularly Secure Key Switching Process. Cryptology ePrint Archive, Report 2015/466, 2015. http://eprint. iacr. org.

168. THE GAP GROUP. Gap Reference Manual .Released 4.7.9, 29 nov. 2015. Disponible en http: //www.gap-system.org/Manuals/doc/ ref/ manual. pdf.

169. TRAVERSO, G., DEMIREL, D. & BUCHMANN, J. 2015. Homomorphic Signature Schemes-A survey. Cryptology ePrint Archive: Report 2015/954, 2015. avaliable at http://eprint. iacr.org/2015/954.

170. TSUNOO, Y., TSUJIHARA, E., SHIGERI, M., SAITO, T., SUZAKI, T., & KUBO, H. 2008. Impossible differential cryptanalysis of CLEFIA. In International Workshop on Fast Software Encryption (pp. 398-411). Springer Berlin Heidelberg.

171. VAJAKAS, J., LISSITSIN, A., JOHANSON, M., LAUR, S., UNRUH, D.& LAUD, P. 2014. Isomorphisms, homomorphisms, automorphisms. Classification of all finite fields. Disponible en http://mathwiki.cs.ut.ee/finite_fields/ [Accessed 10 de abril 2015].

172. VAN TILBORG, H. C. & JAJODIA, S. 2014. Encyclopedia of cryptography and security, Springer Science & Business Media.

173. VAUDENAY, S.1995. On the Need for Multipermutations:Cryptanalysis of MD4 and SAFER, Springer, Verlag, B. Preneel.

174. VAUDENAY, S. 1994. On the need for multipermutations: Cryptanalysis of MD4 and SAFER. Fast Software Encryption, 1994. Springer, 286-297.

175. VAUDENAY, S. & JUNOD, P. 2004. S.Perfet Difusion Primitives For Block Ciphers. Building eficient MDS Matrices. Ecole Polytechnique Federale de Lausanne (Switzerland).

176. VILLALBA, A. S. L. 2000. Sobre el parámetro de complejidad lineal y los filtros no lineales de segundo orden. Revista Matemática Complutense, XIII, 119-134.

177. WANG, H. & TANG, T. Methods for Efficient Homomorphic Integer Polynomial Evaluation based on GSW FHE. Cryptology ePrint Archive: Report 2016/488. http://eprint. iacr. org.

178. WATANABE, D., FURUYA, S., YOSHIDA, H., TAKARAGI, K. & PRENEEL, B. 2002. A new key stream generator MUGI. In International Workshop on Fast Software Encryption, 2002. Springer, 179-194. Springer Berlin Heidelberg

179. WEI, Y. & HU, Y. 2009. New related-key rectangle attacks on reduced AES-192 and AES-256. Science in China Series F: Information Sciences, 52, 617-626.

180. WILLIAMS, F. M. 1986. The Theory of Error Correcting Codes.

181. WILSON, D. P. 2000. Galois Theory. Michaelmas Term 2000.disponible en http://www.jchl.co.uk/maths/Galois.pdf.

182. WONG, K. K.-H. 2008. Applications of finite field computation to cryptology: Extension field arithmetic in public key systems and algebraic attacks on stream ciphers. Doctoral dissertation, Queensland University of Technology.

183. WU, H. 1998. Efficient computations in finite fields with cryptographic significance. PhD Thesis University of Waterloo.

184. WU, H. 2000. Montgomery Multiplier and squarer in $GF(2^n)$. Tecnical Report University of Waterloo The Center for applied cryptographic research.

185. WU, H. 2000. On computation of polynomial modular reduction. Waterloo. Ontorio. Canada: Tecnical Report. University of Waterloo. The Center for Aplied Cyptographic Research.

186. WU, H., HASAN, M. A., BLAKE, I. F. & GAO, S. 2002. Finite field multiplier using redundant representation. Computers, IEEE Transactions on, 51, 1306-1316.

187. WU, S., WANG, M. & WU, W. 2013. Recursive Diffusion Layers for (Lightweight) Block Ciphers and Hash Functions. SAC 2012. LNCS. , vol. 7707. Springer. pp. 355-371.

188. XU, H., TAN, L. & LAI, X. 2014. On the Recursive Construction of MDS Matrices for Lightweight Cryptography. Information Security Practice and Experience. Springer.

189. YAGISAWA, M. (2015). Fully homomorphic encryption on octonion ring. Cryptology ePrint Archive, Report 2015/733, 2015. http://eprint. iacr. org.

190. YAGISAWA, M. Improved Fully Homomorphic Encryption with Composite Number Modulus. Cryptology ePrint Archive: Report 2016/050. http://eprint. iacr. org.

191. YAGISAWA, M. Fully Homomorphic Public-key Encryption Based on Discrete Logarithm Problem. Cryptology ePrint Archive: Report 2016/054. http://eprint. iacr. org.

192. YAGISAWA, M. Fully Homomorphic Encryption with Isotropic Elements Cryptology ePrint Archive: Report 2016/462. http://eprint. iacr. org.

193. YAGISAWA, M. Fully Homomorphic Encryption with Zero Norm Cipher Text. Cryptology ePrint Archive: Report 2016/653. http://eprint. iacr. org.

194. YOUSSEF, A., MISTER, S. & TAVARES, S. 1997. On the design of linear transformations for substitution permutation encryption networks. Workshop on Selected Areas of Cryptography (SAC'96): Workshop Record, 1997. 40-48.

195. YOUSSEF, A. M., MISTER, S., TAVARES, S. E. & YOUSSEF, T. S. 1997. On the Design of Linear Transformations for Substitution Permutation Encryption Networks. SAC'97, pp. 1-9.

196. YOUSSEF, A. M., TAVARES, S.E. & HEYS, H. M. 1996. A New Class of Substitution Permutation Networks. Workshop on Select Areas in Cryptography. Workshop Record, Vols. SAC 96, 132-147.

197. ZHANG, Q. Y. L. 2013. A fast integer-based batch full-homomorphic encryption scheme over finite fields.

198. ZHAO, R., ZHANG, R., LI, Y. AND WU, B. 2015. On Constructions of a Sort of MDS Block Diffusion Matrices for Block Ciphers and Hash Functions. https://eprint.iacr.org/2015/449.pdf.

BiGFSoP[1] - MADARRO, E., CID, Y., CUELLAR, O., SOSA, G., MORGADO, E. & PERFETTI, L. 2014. Software para la solución de problemas sobre campos finitos binarios. Cuba patent application. Número de registro 2741-09-2014.

ANEXOS

Anexo I: Elementos del campo $GF(2^6)$ con polinomio definidor $x^6 + x + 1$

TablaI.1: Elementos del campo $GF(2^6)$con polinomio definidor $x^6 + x + 1$

Potencia de β	Polinomio de grado menor o igual que 5 y secuencia binaria correspondiente	Potencia de β	Polinomio de grado menor o igual que 5 y secuencia binaria de 6 elementos correspondiente
β^0	$1=(1,0,0,0,0,0)$	β^{32}	$1+\beta^3=(1,0,0,1,0,0)$
β^1	$\beta=(0,1,0,0,0,0)$	β^{33}	$\beta+\beta^4=(0,1,0,0,1,0)$
β^2	$\beta^2=(0,0,1,0,0,0)$	β^{34}	$\beta^2+\beta^5=(0,0,1,0,0,1)$
β^3	$\beta^3=(0,0,0,1,0,0)$	β^{35}	$1+\beta+\beta^3=(1,1,0,1,0,0)$
β^4	$\beta^4=(0,0,0,0,1,0)$	β^{36}	$\beta+\beta^2+\beta^4=(0,1,1,0,1,0)$
β^5	$\beta^5=(0,0,0,0,0,1)$	β^{37}	$\beta^2+\beta^3+\beta^5=(0,0,1,1,0,1)$
β^6	$1+\beta=(1,1,0,0,0,0)$	β^{38}	$1+\beta+\beta^3+\beta^4=(1,1,0,1,1,0)$
β^7	$\beta+\beta^2=(0,1,1,0,0,0)$	β^{39}	$\beta+\beta^2+\beta^4+\beta^5=(0,1,1,0,1,1)$
β^8	$\beta^2+\beta^3=(0,0,1,1,0,0)$	β^{40}	$1+\beta+\beta^2+\beta^3+\beta^5=(1,1,1,1,0,1)$
β^9	$\beta^3+\beta^4=(0,0,0,1,1,0)$	β^{41}	$1+\beta^2+\beta^3+\beta^4=(1,0,1,1,1,0)$
β^{10}	$\beta^4+\beta^5=(0,0,0,0,1,1)$	β^{42}	$\beta+\beta^3+\beta^4+\beta^5=(0,1,0,1,1,1)$
β^{11}	$1+\beta+\beta^5=(1,1,0,0,0,1)$	β^{43}	$1+\beta+\beta^2+\beta^4+\beta^5=(1,1,1,0,1,1)$
β^{12}	$1+\beta^2=(1,0,1,0,0,0)$	β^{44}	$1+\beta^2+\beta^3+\beta^5$ (1,0,1,1,0,1)
β^{13}	$\beta+\beta^3=(0,1,0,1,0,0)$	β^{45}	$1+\beta^3+\beta^4=(1,0,0,1,1,0)$
β^{14}	$\beta^2+\beta^4=(0,0,1,0,1,0)$	β^{46}	$\beta+\beta^4+\beta^5=(0,1,0,0,1,1)$
β^{15}	$\beta^3+\beta^5=(0,0,0,1,0,1)$	β^{47}	$1+\beta+\beta^2+\beta^5=(1,1,1,0,0,1)$
β^{16}	$1+\beta+\beta^4=(1,1,0,0,1,0)$	β^{48}	$1+\beta^2+\beta^3=(1,0,1,1,0,0)$
β^{17}	$\beta+\beta^2+\beta^5=(0,1,1,0,0,1)$	β^{49}	$\beta+\beta^3+\beta^4=(0,1,0,1,1,0)$
β^{18}	$1+\beta+\beta^2+\beta^3=(1,1,1,1,0,0)$	β^{50}	$\beta^2+\beta^4+\beta^5=(0,0,1,0,1,1)$
β^{19}	$\beta+\beta^2+\beta^3+\beta^4=(0,1,1,1,1,0)$	β^{51}	$1+\beta+\beta^3+\beta^5=(1,1,0,1,0,1)$

β^{20}	$\beta^2+\beta^3+\beta^4+\beta^5=(0,0,1,1,1,1$	β^{52}	$1+\beta^2+\beta^4=(1,0,1,0,1,0)$
β^{21}	$1+\beta+\beta^3+\beta^4+\beta^5=(1,1,0,1$	β^{53}	$\beta+\beta^3+\beta^5=(0,1,0,1,1,0)$
β^{22}	$1+\beta^2+\beta^4+\beta^5=(1,0,1,0,1,1)$	β^{54}	$1+\beta+\beta^2+\beta^4=(1,1,1,0,1,0)$
β^{23}	$1+\beta^3+\beta^5=(1,0,0,1,0,1)$	β^{55}	$\beta+\beta^2+\beta^3+\beta^5=(0,1,1,1,0,1)$
β^{24}	$1+\beta^4=(1,0,0,0,1,0)$	β^{56}	$1+\beta+\beta^2+\beta^3+\beta^4=(1,1,1,1,1,0)$
β^{25}	$\beta+\beta^5=(0,1,0,0,0,1)$	β^{57}	$\beta+\beta^2+\beta^3+\beta^4+\beta^5=(0,1,1,1,1,1$
β^{26}	$1+\beta+\beta^2=(1,1,1,0,0,0)$	β^{58}	$1+\beta+\beta^2+\beta^3+\beta^4+\beta^5=(1,1,1,$
β^{27}	$\beta+\beta^2+\beta^3=(0,1,1,1,0,0)$	β^{59}	$1+\beta^2+\beta^3+\beta^4+\beta^5=(1,0,1,1,1,1$
β^{28}	$\beta^2+\beta^3+\beta^4=(0,0,1,1,1,0)$	β^{60}	$1+\beta^3+\beta^4+\beta^5=(1,0,0,1,1,1)$
β^{29}	$\beta^3+\beta^4+\beta^5=(0,0,0,1,1,1)$	β^{61}	$1+\beta^4+\beta^5=(1,0,0,0,1,1,)$
β^{30}	$1+\beta+\beta^4+\beta^5=(1,1,0,0,1,1)$	β^{62}	$1+\beta^5=(1,0,0,0,0,1)$
β^{31}	$1+\beta^2+\beta^5=(1,0,1,0,0,1)$	β^{63}	1

Anexo II: Otros algoritmos para la determinación de los homomorfismos de inmersión entre campos de Galois

II.1 Algoritmo ORISTO

Entrada: Elementos primitivos α,β, las dimensiones n, m de los campos finitos, respectivamente, la característica p de los campos finitos, los polinomios definidores de los campos, el conjunto PR de todos los elementos primos relativos con el orden $p^n - 1$, del grupo multiplicativo del campo de partida.

Salida: Funciones que son homomorfismos de inmersión entre los campos finitos especificados.

1 Calcular $k := \frac{p^{\mathrm{m}}-1}{p^{\mathrm{n}}-1}$; y hacer $r := |\mathrm{PR}|;\ s = p^n - 1; b := 0;$

2 Calcular y guardar la n-ésima potencia del elemento primitivo del campo de partida $t := \alpha^n$ como vector columna de n componentes.

3. Para cada entero $i \in PR$, se tienen los posibles homomorfismos $h_w(\alpha) = \beta^{kw}, siendo\ w = i, s - i$, mientras $b < n - 1$ hacer :

3.1. Calcular las matrices M_w de orden $m \times n$ asociadas a las aplicaciones lineales $f_w(\alpha) = \beta^{kw}$(sus columnas son las imágenes de los vectores de la base $\{1, \alpha, \alpha^2, \alpha^3, \cdots, \alpha^{n-1}\}$ del campo de partida).

3.2. Calcular $h_w(\alpha^n) = \beta^{kw.n} = \beta^{kw.(n-1)}\beta^{kw} = f_w(\alpha^{n-1})\beta^{kw}$

3.3. Calcular $f_w(\alpha^n) = M_w \,.\, t$, producto de una matriz M de orden $m \times n$ por un vector columna de dimensión n

3.4. Si para α^n se cumple que $f_w(\alpha^n) = h_w(\alpha^n)$, almacenar h_w e incrementar b

4. Imprimir las funciones que representan los homomorfismos de inmersión

En el paso tres del algoritmo se tiene en cuenta que si

$mcd(s, i) = 1 \Rightarrow\ mcd(s, s - i) = 1$ y además $r = |PR| = \varphi(p^n - 1)$

. Pseudocódigo del algoritmo

INPUT: Elementos primitivos α,β, las dimensiones n, m de los campos finitos, respectivamente, la característica p de los campos finitos y el conjunto PR de todos los elementos primos relativos con el orden $p^n - 1$, del grupo multiplicativo del campo de partida.

OUTPUT: Funciones que son homomorfismos entre los campos finitos especificados.

1. k: = $\frac{p^m - 1}{p^n - 1}$;
2. $r \coloneqq |PR|$;

3. s:=$p^n - 1$
4. g:= α^n;
5. x:=0;
6. for i: = 1 to r do
7. for j: = 0 to n-1 do
8. $M_i^j = \beta^{k \cdot i \cdot j}$;
9. $M_{s-i}^j = \beta^{k \cdot (s-i) \cdot j}$
10. end for
11. $t_i := M_i^{n-1} . \beta^{ki}$;
12. $t_{s-i} := M_{s-i}^{n-1} . \beta^{k(s-i)}$;
13. if ($M_i \times g = t_i$)
14. $\lambda_x := \beta^{k \cdot i}$; $x++$;
15. $else\ if$ ($M_{s-i} \times g = t_{s-i}$)
16. $\lambda_{x+1} := \beta^{k \cdot (s-i)}$; x++;
17. end if
18. if $(x = n)$ go to step 21
19. end if
20. End for
21. Return λ;

M_i^j es la columna j-ésima de la matriz M_i , asociada a la aplicación lineal f_i cuyas columnas son las imágenes de los vectores de la base $\{1, \alpha, \alpha^2, \alpha^3, \cdots, \alpha^{n-1}\}$

Complejidad Computacional

La complejidad del algoritmo depende de la complejidad del paso tres es decir de las operaciones exponenciación y el producto de un vector fila por una matriz. De esta manera, se tiene la siguiente notación asintótica:

$$O\ (r\ max\ ((M);\ (n\ E)))$$

Donde (E) es la complejidad de exponenciación y (M) la complejidad del producto de un vector fila por una matriz binaria.

La complejidad producto de una matriz de orden $m \times n$ por un vector columna de dimensión n es de $O(mn)$. La operación de exponenciación en $GF(p^m)$, tiene una complejidad de un $O(m^2 \log p \log m \log\log m)$, usando métodos basados en la transformada rápida de Fourier (Gao et al. 2000; Panario 2015).

$O\ (r\ \ max\ ((mn);\ \ ((n\ m^2 \log p \log m \log\log m)))) \rightarrow O\ \left(r\ (\ n\ m^2 \log p \log m \log\log m)\right)$

Este algoritmo resulta costoso, hay que recorrer el espacio de búsqueda hasta encontrar todas las funciones que sumergen un campo en otro, es decir, los homomorfismos entre ambos campos. De manera que, a medida que el cardinal de los campos aumenta el tiempo de ejecución del algoritmo aumenta considerablemente y este puede llegar a ser impráctico.

II.2 Algoritmo ORISTO. Variante 2

Desde el punto de vista práctico, en cuanto a tiempo, el algoritmo ORISTO puede ser mejorado teniendo en cuenta la relación existente entre los homomorfismos de inmersión y el automorfismo de *Frobenius* analizada en el epígrafe 2.2. A partir de esto se diseñó la siguiente variante del algoritmo anterior.

Entrada: Elementos primitivos α,β, la dimensión n, m de los campos finitos, respectivamente, la característica p de los campos finitos, el conjunto PR de todos los

elementos primos relativos con el orden $p^n - 1$, del grupo multiplicativo del campo de partida.

Salida: Funciones que son homomorfismos de inmersión entre los campos finitos especificados.

1 Calcular $k := \frac{p^{\mathrm{m}}-1}{p^{\mathrm{n}}-1}$; y hacer $r := |PR|;\ s := p^n - 1;$

2 Calcular y guardar la n-ésima potencia del elemento primitivo del campo de partida $t := \alpha^n$ como vector columna de n componentes.

3 Para cada entero $i \in PR$, se tienen los posibles homomorfismos $\mathrm{h}_w(\alpha) = \beta^{kw}, siendo\ w = i, s - i$,hacer :

3.1. Calcular las matrices M_w de orden $m \times n$ asociadas a las aplicaciones lineales $\mathrm{f}_w(\alpha) = \beta^{kw}$, (sus columnas son las imágenes de los vectores de la base $\{1, \alpha, \alpha^2, \alpha^3, \cdots, \alpha^{n-1}\}$, del campo de partida).

3.2. Calcular $\mathrm{h}_w(\alpha^n) = \beta^{k\,w\cdot\,n} = \beta^{kw\cdot(n-1)}\beta^{kw} = \mathrm{f}_w(\alpha^{n-1})\beta^{kw}$

3.3. Calcular $f_w(\alpha^n) = M_w \cdot t$, producto de una matriz M de orden $m \times n$ por un vector columna de dimensión n.

3.4. Si para α^n se cumple que $f_w(\alpha^n) = \mathrm{h}_w(\alpha^n)$, terminar e ir al paso 4.

4. Calcular las p-ésimas potencias de β^{kw} hasta $(\beta^{kw})^{p^{n-1}}$.

5. Imprimir las funciones que representan los homomorfismos de inmersión.

Pseudocódigo del algoritmo

INPUT: Elementos primitivos α,β, las dimensiones $n\ y\ m$ de los campos finitos, respectivamente, la característica p de estos campos, el conjunto PR de todos los

elementos primos relativos con el orden $p^n - 1$, del grupo multiplicativo del campo de partida.

OUTPUT: Funciones que son homomorfismos entre los campos finitos especificados.

1. k : = $\frac{p^{m}-1}{p^{n}-1}$;
2. $r \coloneqq |PR|$;
3. s:=$p^{n} - 1$
4. g:= α^{n};
5. for $i \coloneqq 1\ to\ r$ do
6. for j: = 0 to n-1 do
7. $M_i^j = \beta^{k \cdot i \cdot j}$;
8. $M_{s-i}^j = \beta^{k \cdot (s-i) \cdot j}$
9. end for
10. $t_i := M_i^{n-1} . \beta^{ki}$;
11. $t_{s-i} := M_{s-i}^{n-1} . \beta^{k(s-i)}$;
12. if ($M_i \times g = t_i$)
13. $\lambda_0 := \beta^{k \cdot (i)}$; go to step 18
14. $else\ if$ ($M_{s-i} \times g = t_{s-i}$)
15. $\lambda_0 := \beta^{k \cdot (s-i)}$; go to step 18
16. end else if
17. end for
18. for $i \coloneqq 1\ to\ n-1$ do
19. $\lambda_i := (\lambda_{i-1})^p$

20. end for

21. return λ;

Complejidad Computacional

La complejidad computacional del algoritmo varía en cuanto al parámetro que determina el espacio de búsqueda. De manera que, el nuevo algoritmo solo analiza c elementos, donde c es la cantidad de elementos chequeados hasta encontrar el primer homomorfismo, con $c < r$. Así, se tiene la siguiente notación asintótica para esta variante es $O\ (c\ \ max\ ((mn);\ \ ((n\,m^2 \log p \log m \log\log m)))) \rightarrow O\ \big(c\ (\ n\,m^2 \log p \log m \log\log m)\big)$

En el paso 4 del algoritmo las potencias se calculan elevando la función obtenida a la p-ésima potencia y así sucesivamente hasta encontrar los n homomorfismos de inmersión.

Esta variante mejora el tiempo de cómputo pues alcanza de manera más rápida la solución final.

Anexo III: Algoritmo EVOR en pseudocódigo

INPUT: Elemento primitivo β del segundo campo finito y el polinomio P que define al primer campo finito. La dimensión n, m de los dos campos, respectivamente. La característica p de los campos finitos. El conjunto CL de todos los cosetos líderes módulo $p^n - 1$, que son primos relativos con el orden del grupo multiplicativo del campo de partida.

OUTPUT: Funciones que son homomorfismos entre los campos finitos especificados.

1. k := $\frac{p^n-1}{p^m-1}$;
2. $r := |CL|$;
3. $s := p^{\mathrm{n}} - 1$

4. for i = 1 to r do
5. $h_i = i \bullet k$;
6. $t \coloneqq \beta^{h_i}$
7. $if\ (t\ is\ a\ root\ of\ P)$
8. $\lambda_0^i \coloneqq t$
9. $for\ j = 1\ to\ n-1\ do$
10. $\lambda_j^i =: \left(\lambda_{j-1}^i\right)^p$;
11. end for and go to step 14
12. end if
13. end for
14. return λ;

Anexo IV: Isomorfismos entre distintas variantes del campo $GF(5^2)$

Se considera el caso particular del campo $GF(5^2)$. El campo $GF(5^2)$ es una extensión algebraica de grado dos, del campo primo $\mathrm{GF}(5) = \{0,1,2,3,4\}$.

Los polinomios primitivos de grado dos sobre el campo $\mathbb{F}_5$ son

$$x^2 + x + 2\ , \qquad x^2 + 2\,x + 3\ , x^2 + 3x + 3, \qquad x^2 + 4\,x + 2.$$

Los polinomios mónicos asociados a los reversos de $x^2 + x + 2$ y $x^2 + 2\,x + 3$ son $x^2 + 3x + 3$ y $x^2 + 4\,x + 2$ respectivamente

Las raíces de estos 4 polinomios primitivos son los 8 elementos primitivos del campo $GF(5^2)$.

Si α es raíz de $f(x) = x^2 + x + 2$, que es irreducible sobre $\mathbb{F}_5$. Entonces $GF(5^2)$ puede representarse por el conjunto $GF(5^2) = \{c_1\alpha + c_0 | c_i \in GF(5)\}$ con la suma y la multiplicación usuales en $\mathrm{GF}(5)[\alpha]$ y reduciendo los elementos de grado mayor que uno mediante la relación $\alpha^2 + \alpha + 2 = 0$.

Los elementos primitivos de este campo son las potencias de α que son primos relativos con el orden 24. Tales elementos son $8 = \varphi(24) = \varphi(3.2^3) = \varphi(3)\varphi(2^3) = 2.4$

Los elementos primitivos son α , $\alpha^5, \alpha^7, \alpha^{11}, \alpha^{13}, \alpha^{17}, \alpha^{19}, \alpha^{23}$.

Tabla IV.1 .Polinomios primitivos con sus raíces

Polin.prim	$x^2 + x + 2$	$x^2 + 2\,x + 3$	$x^2 + 3\,x + 3$	$x^2 + 4\,x + 2$
Raíces	α , α^5	α^7, α^{11}	α^{19}, α^{23}	α^{13}, α^{17}

En dependencia de cuál de estos 4 polinomios se utilice para definir el campo se obtienen cuatro variantes del campo $GF(5^2)$. Los campos resultantes van a ser isomorfos, pero ¿cuáles son las funciones que representan estos isomorfismos?. El objetivo de este anexo es presentar estas funciones

Se agrupan en la Tabla IV.2 las cuatro variantes del campo.

Tabla IV.2 Variantes del campo $GF(5^2)$

	Variante 1	Variante 2	Variante 3	Variante 4
	$x^2 + x + 2$	$x^2 + 2\,x + 3$	$x^2 + 3\,x + 3$	$x^2 + 4\,x + 2$
	$Ep: \alpha$	$Ep: \beta$	$Ep: \gamma$	$Ep: \delta$
0	1	1	1	1
1	α	β	γ	δ
2	$4\alpha + 3$	$3\beta + 2$	$2\gamma + 2$	$\delta + 3$
3	$4\alpha + 2$	$\beta + 1$	$\gamma + 4$	$4\delta + 3$
4	$3\alpha + 2$	$4\beta + 2$	$\gamma + 2$	$2\delta + 2$

5	$4\alpha + 4$	$4\beta + 3$	$4\gamma + 2$	$4\delta + 1$
6	2	3	3	2
7	2α	3β	3γ	2δ
8	$3\alpha + 1$	$4\beta + 1$	$\gamma + 1$	$2\delta + 1$
9	$3\alpha + 4$	$3\beta + 3$	$3\gamma + 2$	$3\delta + 1$
10	$\alpha + 4$	$2\beta + 1$	$3\gamma + 1$	$4\delta + 4$
11	$3\alpha + 3$	$2\beta + 4$	$2\gamma + 1$	$3\delta + 2$
12	4	4	4	4
13	4α	4β	4γ	4δ
14	$\alpha + 2$	$2\beta + 3$	$3\gamma + 3$	$4\delta + 2$
15	$\alpha + 3$	$4\beta + 4$	$4\gamma + 1$	$\delta + 2$
16	$2\alpha + 3$	$\beta + 3$	$4\gamma + 3$	$3\delta + 3$
17	$\alpha + 1$	$\beta + 2$	$\gamma + 3$	$X + 4$
18	3	2	2	3
19	3α	2β	2γ	3δ
20	$2\alpha + 4$	$\beta + 4$	$4\gamma + 4$	$3\delta + 4$
21	$2\alpha + 1$	$2\beta + 2$	$2\gamma + 3$	$2\delta + 4$
22	$4\alpha + 1$	$3\beta + 4$	$2\gamma + 4$	$\delta + 1$
23	$2\alpha + 2$	$3\beta + 1$	$3\gamma + 4$	$2\delta + 3$

Ep: significa elemento primitivo

En total se consideran 6 posibles parejas de campos

1. $\mathrm{GF}(5)(\alpha) y\ \mathrm{GF}(5)(\beta)$
2. $\mathrm{GF}(5)(\alpha) y \mathrm{GF}(5)(\gamma)$
3. $\mathrm{GF}(5)(\alpha) y \mathrm{GF}(5)(\delta)$
4. $\mathrm{GF}(5)(\beta) y \mathrm{GF}(5)(\gamma)$
5. $\mathrm{GF}(5)(\beta) y \mathrm{GF}(5)(\delta)$
6. $\mathrm{GF}(5)(\gamma) y \mathrm{GF}(5)(\delta)$

Se hallan los isomorfismos en el primer caso $\mathrm{GF}(5)(\alpha)\ y\ \mathrm{GF}(5)(\beta)$

Usando el algoritmo EVOR se tiene que las raíces de polinomio $x^2 + x + 2$ en el campo

$\mathrm{GF}(5)(\beta)$ son $\beta^{7} y\, \beta^{21}$. Por tanto los isomorfismos entre estas dos variantes están dados por las funciones $f_1(\alpha) = \beta^7$ y $f_2(\alpha) = \beta^{11}$ que tienen la característica de que una puede ser obtenida a partir de la otra utilizando el automorfismo de *Frobenius* $\Phi: GF(5^2) \rightarrow GF(5^2), \Phi(x) = x^5$.

Se hallan los isomorfismos en el segundo primer caso $\mathrm{GF}(5)(\alpha)\ y\ \mathrm{GF}(5)(\gamma)$

Usando el algoritmo EVOR se tiene que las raíces del polinomio $x^2 + x + 2$ en el campo $\mathrm{GF}(5)(\gamma)$ son $\gamma^{19}\ \ y\ \ \gamma^{23}$. Por tanto los isomorfismos entre estas dos variantes están dados por las funciones $f_1(\alpha) = \gamma^{19}$ y $f_2(\alpha) = \gamma^{23}$.

De manera similar se obtienen los isomorfismos en los otros casos, obteniéndose los siguientes resultados.

Tabla IV.3. Isomorfismos entre distintas variantes del campo $GF(5^2)$

Variantes	Funciones que representan los isomorfismos
1. $\mathbb{F}_5(\alpha) y\, \mathbb{F}_5(\beta)$	$f_1(\alpha) = \beta^7$ y $f_2(\alpha) = \beta^{11}$
2. $\mathbb{F}_5(\alpha) y\, \mathbb{F}_5(\gamma)$	$f_1(\alpha) = \gamma^{19}$ y $f_2(\alpha) = \gamma^{23}$
3. $\mathbb{F}_5(\alpha) y\, \mathbb{F}_5(\delta)$	$f_1(\alpha) = \delta^{13}$ y $f_2(\alpha) = \delta^{17}$
4. $\mathbb{F}_5(\beta) y\, \mathbb{F}_5(\gamma)$	$f_1(\beta) = \gamma^{13}$ y $f_2(\beta) = \gamma^{17}$
5. $\mathbb{F}_5(\beta) y\, \mathbb{F}_5(\delta)$	$f_1(\beta) = \delta^{19}$ y $f_2(\beta) = \delta^{23}$
6. $\mathbb{F}_5(\gamma) y\, \mathbb{F}_5(\delta)$	$f_1(\gamma) = \delta^7$ y $f_2(\gamma) = \delta^{11}$

Anexo V: El logaritmo de Zech´s para el campo $GF(2^8)$

Este anexo resume el artículo "*A reduced table of the Zech´s Logarithm"* publicado en la revista *Journal of Advanced in Mathematics* en agosto de este año. (Cuellar et al. 2016).

Al trabajar con campos finitos se realizan operaciones aritméticas con sus elementos, por lo que se necesita encontrar algoritmos eficientes para efectuar dichas operaciones, siendo éste un problema de gran importancia actualmente.

Teniendo en cuenta que la multiplicación de elementos de un campo finito es una multiplicación de polinomios resulta conveniente, expresar los elementos del campo como potencias del elemento primitivo, de manera que la multiplicación de ellos se reduzca a la suma de sus exponentes, también es necesario disponer de un procedimiento que permita realizar de manera eficiente la suma de dos elementos del campo expresados como potencia del elemento primitivo.

El propósito de este Anexo es presentar la tabla del logaritmo de *Zech´s* para el campo $GF(2^8)$.

Sea el campo finito $GF(q) = GF(p^n)$, donde p es un número primo y n un número entero positivo. Sea α un elemento generador del grupo multiplicativo del campo $GF(p^n)$.

El producto $\alpha^i \alpha^j = \alpha^{i+j(mod(p^n-1))}$

La adición de los elementos del campo finito expresados como potencias del elemento primitivo se facilita si se construye la tabla, llamada tabla del logaritmo de *Zech´s* (Huber 1990) donde para cada entero $i,\ 0 \leq i \leq p^n - 2$, se determina y tabula el entero $j = z(i)$ tal que $1 + \alpha^i = \alpha^j = \alpha^{z(i)}$. (1)

$\alpha^i + \alpha^j = \alpha^i\left(1 + \alpha^{(j-i)(mod(p^n-1))}\right) = \alpha^i . \alpha^{z(j-i)} = \alpha^{i+z(j-i)}$, donde $z(j-i)$, se toma de la tabla del logaritmo de Zech´s

Teniendo en cuenta (1) y las propiedades de los campos finitos se tiene

$$\left(1 + \alpha^i\right)^p = \left(\alpha^{z(i)}\right)^p \Rightarrow 1 + \alpha^{ip} = \alpha^{pz(i)}$$

´ $\left(1 + \alpha^i\right)^{p^k} = \left(\alpha^{z(i)}\right)^{p^k} \Rightarrow 1 + \alpha^{ip^k} = \alpha^{z(i)\,.p^k}$

Se presenta la tabla del logaritmo de *Zech´s* para el campo $GF(2^8)$, con polinomio primitivo definidor $x^8 + x^4 + x^3 + x^2 + 1 = 0$, agrupando los valores de j de acuerdo al coseto ciclotómico al que pertenecen.

$$1 + \alpha^i = \alpha^{z(i)}$$

Tabla V.1 Logaritmo de *Zech´s* para el campo $GF(2^8)$

i	$z(i)$	i	$z(i)$	i	$z(i)$	i	$z(i)$
1	25	3	223	5	138	7	112
2	50	6	191	10	21	14	224
4	100	12	127	20	42	28	193
8	200	24	254	40	84	56	131
16	145	48	253	80	168	112	7
32	35	96	251	160	81	224	14
64	70	192	247	65	162	193	28
128	140	129	239	130	69	131	56
9	120	11	245	13	99	15	33
18	240	22	235	26	198	30	66
36	225	44	215	52	141	60	132
72	195	88	175	104	27	120	68
144	135	176	95	208	54	240	136
33	15	97	190	161	108	225	17
66	30	194	125	67	216	195	34
132	60	133	250	134	177	135	68
17	68	51	238	85	170	119	153
34	136	102	221	170	85	238	51

68	17	204	187	∞	0	221	102
136	34	153	119	0	∞	187	204
19	92	21	10	23	196	25	1
38	184	42	20	46	137	50	2
76	113	84	40	92	19	100	4
152	138	168	80	184	38	200	8
49	197	81	160	113	76	145	16
98	139	162	65	226	152	35	32
196	23	69	130	197	49	70	64
137	46	138	5	139	98	140	128
27	104	29	181	31	45	37	179
54	208	58	107	62	90	74	103
108	161	116	214	124	180	148	206
216	67	232	173	248	105	41	157
177	134	209	91	241	210	82	59
99	13	163	182	227	165	164	179
198	26	71	109	199	75	73	103
141	52	142	218	143	150	146	206
39	106	43	121	45	31	47	101
78	212	86	242	90	62	94	202
156	169	172	229	180	124	188	149
57	83	89	203	105	248	121	43
114	166	178	151	210	241	242	86
228	77	101	47	165	227	229	172

201	154	202	94	75	199	203	89
147	53	149	188	150	143	151	178
53	147	55	63	59	82	61	186
106	39	110	126	118	164	122	117
212	78	220	252	236	73	244	234
169	156	185	249	217	146	233	213
83	57	115	243	179	37	211	171
166	114	230	231	103	74	167	87
77	228	205	207	206	148	79	174
154	201	155	159	157	41	158	93
87	167	91	209	95	176	11	246
174	79	182	163	190	97	222	237
93	158	109	71	125	194	189	219
186	61	218	142	250	133	123	183
117	122	181	29	245	11	246	111
234	244	107	58	235	22	237	222
213	233	214	116	215	44	219	189
171	211	173	232	175	88	183	123
127	12	253	48	247	192	223	3
254	24	251	96	239	129	191	6

Anexo VI. Nuevo algoritmo del epígrafe 3.3

VI.1 Pseudocódigo de la variante dos del nuevo algoritmo

INPUT: Polinomio primitivo definidor del campo $GF(2^8)$, el polinomio primitivo $g_1(x) = g_{10} + g_{11}x + g_{12}\,x^2 + g_{1\,3}x^3 + x^4 \in GF(2^8)[x]$. La tabla Prod [256][256] y Sum [256][256] del producto y la suma de todos los elementos del campo $GF(2^8)$, expresados como potencia del elemento primitivo.

OUTPUT: Matrices MDS A y A^{-1}

$$Fila_i = \{f_{i0}, f_{i1}, f_{i2}, f_{i3}\}, con\ f_{ij} \in GF(2^8)^*$$

1. Generar aleatoriamente los 4 elementos no nulos de la primera fila $f_{10}, f_{11}, f_{12}, f_{13}$.
2. for i = 2 to n do
3. $f_{i0} := g_{10} f_{(i-1)3}$;
4. $f_{i1} := f_{(i-1)0} + g_{11} f_{(i-1)3}$;
5. $f_{i2} := f_{(i-1)1} + g_{12} f_{(i-1)3}$;
6. $f_{i3} := f_{(i-1)2} + g_{13} f_{(i-1)3}$;
7. if $f_{ij} = 0$ for $algun\ i, j,\ i = \overline{2,4}, j = \overline{0,3}$, go to step 1, else go to step 10
8. end if
9. end for
10. for r =1 to K do
11. Calcular r_k
12. if $r_k = 0$ para algún k, go to step 1, else go to step 15
13. end if
14. end for

15. Calcular el polinomio $\tilde{f}_{10} + \tilde{f}_{11}x + \tilde{f}_{12}\,x^2 + \tilde{f}_{13}x^3$ inverso del polinomio $f_{10} + f_{11}x + f_{12}x^2 + f_{13}x^3$
16. if $\tilde{f}_{1i} = 0$, para algún $i = \overline{0,3}$, go to step 1 ,else go to step 17
17. Row 1:= $\{\tilde{f}_{10}, \tilde{f}_{11}, \tilde{f}_{12}\,, \tilde{f}_{13}\}$;
18. for i = 2 to n do
19. $\tilde{f}_{i0} := g_{10}f_{(i-1)3}$;
20. $\tilde{f}_{i1} := f_{(i-1)0} + g_{11}f_{(i-1)3}$;
21. $\tilde{f}_{i2} := f_{(i-1)1} + g_{12}f_{(i-1)3}$;
22. $\tilde{f}_{i3} := f_{(i-1)2} + \ g_{13}f_{(i-1)3}$;
23. if $\tilde{f}_{ij} = 0$ for $algún\ i\,, j,\ i = \overline{2,4}\,, j = \overline{0,3}$, go to step 1, else go to step 26
24. end if
25. end for
26. return A y A^{-1};

VI.2 Precomputo de los determinantes de las submatrices de orden dos.

$$r_1 = f_{11}f_{20} + f_{10}f_{21}, r_2 = f_{12}f_{20} + f_{10}f_{22}, r_3 = f_{13}f_{20} + f_{10}f_{23}, r_4 = f_{12}f_{21} + f_{11}f_{22},$$

$$r_5 = f_{13}f_{21} + f_{11}f_{23}, r_6 = f_{13}f_{22} + f_{12}f_{23}, r_7 = f_{11}f_{30} + f_{10}f_{31}, r_8 = f_{12}f_{30} + f_{10}f_{32},$$

$$r_9 = f_{13}f_{30} + f_{10}f_{33}, r_{10} = f_{12}f_{31} + f_{11}f_{32}, r_{11} = f_{13}f_{31} + f_{11}f_{33}, r_{12} = f_{13}f_{32} + f_{12}f_{33},$$

$$r_{13} = f_{11}f_{40} + f_{10}f_{41}, r_{14} = f_{12}f_{40} + f_{10}f_{42}, r_{15} = f_{13}f_{40} + f_{10}f_{43}, r_{16} = f_{12}f_{41} + f_{11}f_{42},$$

$$r_{17} = f_{13}f_{41} + f_{11}f_{43}, r_{18} = f_{13}f_{42} + f_{12}f_{43}, r_{19} = f_{21}f_{30} + f_{20}f_{31}, r_{20} = f_{22}f_{30} + f_{20}f_{32},$$

$$r_{21} = f_{23}f_{30} + f_{20}f_{33}, r_{22} = f_{22}f_{31} + f_{21}f_{32}, r_{23} = f_{23}f_{31} + f_{21}f_{33}, r_{24} = f_{23}f_{32} + f_{22}f_{33},$$

$$r_{25} = f_{21}f_{40} + f_{20}f_{41}, r_{26} = f_{22}f_{40} + f_{20}f_{42}, r_{27} = f_{23}f_{40} + f_{20}f_{43}, r_{28} = f_{22}f_{41} + f_{21}f_{42},$$

$$r_{29} = f_{23}f_{41} + f_{21}f_{43}, r_{30} = f_{23}f_{42} + f_{22}f_{43}, r_{31} = f_{31}f_{40} + f_{30}f_{41}, r_{32} = f_{32}f_{40} + f_{30}f_{42},$$

$$r_{33} = f_{33}f_{40} + f_{30}f_{43}, r_{34} = f_{32}f_{41} + f_{31}f_{42}, r_{35} = f_{33}f_{41} + f_{31}f_{43}, r_{36} = f_{33}f_{42} + f_{32}f_{43}$$

VI.3 Obtención de expresiones recursivas para el cálculo de los elementos de la matriz para la variante 1 del nuevo algoritmo

Denotando los elementos de la fila i por $f_{i\,j}$, $con\ i = \overline{1,4}$, $j = \overline{0,3}$

$$Fila_i = (f_{i0}, f_{i1}, f_{i2}, f_{i3})$$

Al calcular las filas se obtiene

$$Fila_1 = (f_{10}, f_{11}, f_{12}, f_{13}) = \left(b_{1,0}, b_{1,1}, b_{1,2}, b_{1,3}\right)$$

$$Fila_2 = \left(g_{10}b_{1,3}b_{2,0}, b_{1,0}b_{2,0} + g_{11}b_{1,3}b_{2,0}, b_{1,1}b_{2,0} + g_{12}b_{1,3}b_{2,0}, b_{1,2}b_{2,0} + g_{13}b_{1,3}b_{2,0}\right)$$

Extrayendo factor común

$$Fila_2 = \left(b_{2,0}\left(g_{10}b_{1,3}\right), b_{2,0}\left(b_{1,0} + g_{11}b_{1,3}\right), b_{2,0}\left(b_{1,1} + g_{12}b_{1,3}\right), b_{2,0}\left(b_{1,2} + g_{13}b_{1,3}\right)\right)$$

$$Fila_2 = \left(b_{2,0}(g_{10}f_{13}), b_{2,0}(f_{10} + g_{11}f_{13}), b_{2,0}(f_{11} + g_{12}f_{13}), b_{2,0}(f_{12} + g_{13}f_{13})\right)$$

$$\begin{aligned} Fila_3 = \big(&g_{10}b_{1,2}b_{2,0}b_{3,0} + g_{10}g_{13}b_{1,3}b_{2,0}b_{3\,,0}, g_{11}b_{1,2}b_{2,0}b_{3,0} + g_{10}b_{1,3}b_{2,0}b_{3,0} \\ &+ g_{11}g_{13}b_{1,3}b_{2,0}b_{3,0}, b_{1,0}b_{2,0}b_{3,0} + g_{12}b_{1,2}b_{2,0}b_{3,0} + g_{11}b_{1,3}b_{2,0}b_{3,0} \\ &+ g_{12}g_{13}b_{1,3}b_{2,0}b_{3,0}, b_{1,1}b_{2,0}b_{3,0} + g_{13}b_{1,2}b_{2,0}b_{3,0} + g_{12}b_{1,3}b_{2,0}b_{3,0} \\ &+ g_{13}^2b_{1,3}b_{2,0}b_{3,0}\big) \end{aligned}$$

Extrayendo factor común

$$\begin{aligned} Fila_3 = \Big(&b_{3,0}g_{10}b_{2,0}\left(b_{1,2} + g_{13}b_{1,3}\right), b_{3,0}\Big(b_{2,0}\left(g_{10}b_{1,3}\right) \\ &+ g_{11}b_{2,0}\left(b_{1,2} + g_{13}b_{1,3}\right)\Big), b_{3,0}\Big(b_{2,0}\left(b_{1,0} + g_{11}b_{1,3}\right) \\ &+ g_{12}b_{2,0}\left(b_{1,2} + g_{13}b_{1,3}\right)\Big), b_{3,0}\Big(b_{2,0}\left(b_{1,1} + g_{12}b_{1,3}\right) \\ &+ g_{13}b_{2,0}\left(b_{1,2} + g_{13}b_{1,3}\right)\Big)\Big) \end{aligned}$$

$$Fila_3 = \left(b_{3,0}g_{10}f_{23}, b_{3,0}(f_{20} + g_{11}f_{23}), b_{3,0}(f_{21} + g_{12}f_{23}), b_{3,0}(f_{22} + g_{13}f_{23})\right)$$

$$\begin{aligned} Fila_4 = (&g_{10}b_{1,1}b_{2,0}b_{3,0}b_{4,0} + g_{10}g_{13}b_{1,2}b_{2,0}b_{3,0}b_{4,0} + g_{10}g_{12}b_{1,3}b_{2,0}b_{3,0}b_{4,0} \\ &+ g_{10}g_{13}^2b_{1,3}b_{2,0}b_{3,0}b_{4,0}, g_{11}b_{1,1}b_{2,0}b_{3,0}b_{4,0} + g_{10}b_{1,2}b_{2,0}b_{3,0}b_{4,0} \\ &+ g_{11}g_{13}b_{1,2}b_{2,0}b_{3,0}b_{4,0} + g_{11}g_{12}b_{1,3}b_{2,0}b_{3,0}b_{4,0} + g_{10}g_{13}b_{1,3}b_{2,0}b_{3,0}b_{4,0} \\ &+ g_{11}g_{13}^2b_{1,3}b_{2,0}b_{3,0}b_{4,0}, g_{12}b_{1,1}b_{2,0}b_{3,0}b_{4,0} + g_{11}b_{1,2}b_{2,0}b_{3,0}b_{4,0} \\ &+ g_{12}g_{13}b_{1,2}b_{2,0}b_{3,0}b_{4,0} + g_{10}b_{1,3}b_{2,0}b_{3,0}b_{4,0} + g_{12}^2b_{1,3}b_{2,0}b_{3,0}b_{4,0} \\ &+ g_{11}g_{13}b_{1,3}b_{2,0}b_{3,0}b_{4,0} + g_{12}g_{13}^2b_{1,3}b_{2,0}b_{3,0}b_{4,0}, b_{1,0}b_{2,0}b_{3,0}b_{4,0} \\ &+ g_{13}b_{1,1}b_{2,0}b_{3,0}b_{4,0} + g_{12}b_{1,2}b_{2,0}b_{3,0}b_{4,0} + g_{13}^2b_{1,2}b_{2,0}b_{3,0}b_{4,0} \\ &+ g_{11}b_{1,3}b_{2,0}b_{3,0}b_{4,0} + g_{13}^3b_{1,3}b_{2,0}b_{3,0}b_{4,0}) \end{aligned}$$

Extrayendo factor común

$$\begin{aligned} Fila_4 = \Big(&b_{4,0}g_{10}\Big(b_{3,0}\Big(b_{2,0}(b_{1,1} + g_{12}b_{1,3}) \\ &+ g_{13}b_{2,0}(b_{1,2} + g_{13}b_{1,3})\Big)\Big), b_{4,0}\Big(b_{3,0}g_{10}b_{2,0}(b_{1,2} + g_{13}b_{1,3}) \\ &+ g_{11}\Big(b_{3,0}\Big(b_{2,0}(b_{1,1} + g_{12}b_{1,3}) + g_{13}b_{2,0}(b_{1,2} + g_{13}b_{1,3})\Big)\Big)\Big), \\ b_{4,0}\Big(&b_{3,0}\Big(b_{2,0}(g_{10}b_{1,3}) + g_{11}b_{2,0}(b_{1,2} + g_{13}b_{1,3})\Big) \\ &+ g_{12}b_{3,0}\Big(b_{2,0}(b_{1,1} + g_{12}b_{1,3}) \\ &+ g_{13}b_{2,0}(b_{1,2} + g_{13}b_{1,3})\Big)\Big), b_{4,0}\Big(b_{3,0}\Big(b_{2,0}(b_{1,0} + g_{11}b_{1,3}) \\ &+ g_{12}b_{2,0}(b_{1,2} + g_{13}b_{1,3})\Big) \\ &+ g_{13}b_{3,0}\Big(b_{2,0}(b_{1,1} + g_{12}b_{1,3}) + g_{13}b_{2,0}(b_{1,2} + g_{13}b_{1,3})\Big)\Big)\Big) \end{aligned}$$

$$Fila_4 = \left(b_{4,0}g_{10}f_{33}, b_{4,0}(f_{30} + g_{11}f_{33}), b_{4,0}(f_{31} + g_{12}f_{33}), b_{4,0}(\mathrm{f}_{32} + \mathrm{g}_{13}\mathrm{f}_{33})\right)$$

VI.4 Algoritmo para la determinación de la inversa de una matriz y sus variantes

Este algoritmo para el cálculo de la matriz inversa de una matriz no singular fue propuesto en (Freyre et al. 2015).

Entrada: Polinomio $x^8 + x^4 + x^3 + x^2 + 1$, primitivo definidor del campo $GF(2^8)$. Polinomios primitivos $g_1(x), g_2(x), g_3(x) \in GF(2^8)[x]$ seleccionados a priori y que cumplen que: $gr\big(g_1(x)\big) = 4,\ gr\big(g_2(x)\big) = 3,\ gr(g_3(x)) = 2)$.

La matriz aleatoria $M = \begin{pmatrix} \mathrm{b}_{1,0} & \mathrm{b}_{1,1} & \mathrm{b}_{1,2} & \mathrm{b}_{1,3} \\ \mathrm{c}_{2,0} & \mathrm{b}_{2,0} & \mathrm{b}_{2,1} & \mathrm{b}_{2,2} \\ \mathrm{c}_{3,0} & \mathrm{c}_{3,1} & \mathrm{b}_{3,0} & \mathrm{b}_{3,1} \\ \mathrm{c}_{4,0} & \mathrm{c}_{4,1} & \mathrm{c}_{4,2} & \mathrm{b}_{4,0} \end{pmatrix}$

Donde: $c_{i,j}\, y\ b_{k,t} \in GF(2^8), \big(b_{k,0}, b_{k,1}, \dots, b_{k,n-k}\big) \neq 0,$

$$k = \overline{1,4},\ \ t = \overline{0,3},\ \ i = \overline{2,4}\ y\ j = \overline{0,2},\ \mathrm{b}_{1,i} \neq 0, \forall\, i.$$

Salida: Matriz A^{-1}.

Cálculo de la fila i - ésima de A^{-1}.

Paso 1

Entrada: $(a_0, a_1, a_2, a_3) := i - \acute{e}simo\ vector\ can\acute{o}nico$

$$(a_0 + a_1x + a_2x^2 + a_3x^3)\big(b_{1,0} + b_{1,1}x + b_{1,2}x^2 + b_{1,3}x^3\big)^{-1}\ mod g_1(x)$$

$$\widehat{a_0} + \widehat{a_1}\,x + \widehat{a_2}x^2 + \widehat{a_3}x^3 := x^{i-1}\big(b_{1,0} + b_{1,1}x + b_{1,2}x^2 + b_{1,3}x^3\big)^{-1}\ mod g_1(x)$$

Salida: $(\widehat{a_0}, \widehat{a_1}, \widehat{a_2}, \widehat{a_3})$

Paso 2:

Entrada: $(a_0, a_1, a_2, a_3) := (\widehat{a_0}, \widehat{a_1}, \widehat{a_2}, \widehat{a_3})$

$$\widehat{a_1} + \widehat{a_2}x + \widehat{a_3}x^2 := (a_1 + a_2x + a_3x^2)\left(b_{2,0} + b_{2,1}x + b_{2,2}x^2\right)^{-1} mod g_2(x)$$

$$\widehat{a_0} := a_0 + c_{2,0}\,\widehat{a_1}$$

Salida: $(\widehat{a_0}, \widehat{a_1}, \widehat{a_2}, \widehat{a_3})$

Paso 3:

Entrada: $(a_0, a_1, a_2, a_3) := (\widehat{a_0}, \widehat{a_1}, \widehat{a_2}, \widehat{a_3})$

$$\widehat{a_2} + \widehat{a_3}x := (a_2 + a_3x)\left(b_{3,0} + b_{3,1}x\right)^{-1} mod g_3(x)$$

$$\widehat{a_0} := a_0 + c_{3,0}\,\widehat{a_2}\,;\ \widehat{a_1} := a_1 + c_{3,1}\,\widehat{a_2}$$

Salida: $(\widehat{a_0}, \widehat{a_1}, \widehat{a_2}, \widehat{a_3})$

Paso 4:

Entrada: $(a_0, a_1, a_2, a_3) := (\widehat{a_0}, \widehat{a_1}, \widehat{a_2}, \widehat{a_3})$

$$\widehat{a_3} := a_3\left(b_{4,0}\right)^{-1};\ \widehat{a_0} := a_0 + c_{4,0}\,\widehat{a_3}\,;\ \widehat{a_1} := a_1 + c_{4,1}\,\widehat{a_3}\,;\ \ \widehat{a_2} := a_2 + c_{4,2}\,\widehat{a_3}$$

Cuando la matriz M tiene la forma

$$\begin{pmatrix} b_{1,0} & b_{1,1} & b_{1,2} & b_{1,2} \\ 0 & b_{2,0} & 0 & 0 \\ 0 & 0 & b_{3,0} & 0 \\ 0 & 0 & 0 & b_{4,0} \end{pmatrix}$$

El algoritmo de cálculo de la inversa adopta la forma

Fila i - ésima de A^{-1}.

Paso 1

Entrada: $(a_0, a_1, a_2, a_3) := i - \acute{e}simo\ vector\ can\acute{o}nico$

$$(a_0 + a_1x + a_2x^2 + a_3x^3)\left(b_{1,0} + b_{1,1}x + b_{1,2}x^2 + b_{1,3}x^3\right)^{-1} mod g_1(x)$$

$$= x^{i-1}\left(b_{1,0} + b_{1,1}x + b_{1,2}x^2 + b_{1,3}x^3\right)^{-1} mod g_1(x) = \widehat{a_0} + \widehat{a_1}\,x + \widehat{a_2}x^2 + \widehat{a_3}x^3$$

Salida: $(\widehat{a_0}, \widehat{a_1}, \widehat{a_2}, \widehat{a_3})$

Paso 2:

Entrada: $(a_0, a_1, a_2, a_3) := (\widehat{a_0}, \widehat{a_1}, \widehat{a_2}, \widehat{a_3})$

$$\widehat{a_1} + \widehat{a_2}x + \widehat{a_3}x^2 := (a_1 + a_2x + a_3x^2)\left(b_{2,0}\right)^{-1} mod g_2(x)$$

$$\widehat{a_0} := a_0$$

Salida: $(\widehat{a_0}, \widehat{a_1}, \widehat{a_2}, \widehat{a_3})$

Paso 3:

Entrada: $(a_0, a_1, a_2, a_3) := (\widehat{a_0}, \widehat{a_1}, \widehat{a_2}, \widehat{a_3})$

$$\widehat{a_2} + \widehat{a_3}x := (a_2 + a_3x)\left(b_{3,0}\right)^{-1} mod g_3(x);\ \widehat{a_0} := a_0;\ \widehat{a_1} := a_1$$

Salida: $(\widehat{a_0}, \widehat{a_1}, \widehat{a_2}, \widehat{a_3})$

Paso 4:

Entrada: $(a_0, a_1, a_2, a_3) := (\widehat{a_0}, \widehat{a_1}, \widehat{a_2}, \widehat{a_3})$

$$\widehat{a_3} := a_3\left(b_{4,0}\right)^{-1}; \widehat{a_0} := a_0;\ \widehat{a_1} := a_1;\ \widehat{a_2} := a_2$$

Cuando la matriz M tiene la forma

$$\begin{pmatrix} b_{1,0} & b_{1,1} & b_{1,2} & b_{1,2} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

El algoritmo de cálculo de la inversa adopta la forma

Fila i - ésima de A^{-1}.

Paso 1

Entrada: $(a_0, a_1, a_2, a_3) := i - \acute{e}simo\ vector\ can\acute{o}nico$

$$(a_0 + a_1x + a_2x^2 + a_3x^3)\left(b_{1,0} + b_{1,1}x + b_{1,2}x^2 + b_{1,3}x^3\right)^{-1} mod g_1(x)$$

$$= x^{i-1}\left(b_{1,0} + b_{1,1}x + b_{1,2}x^2 + b_{1,3}x^3\right)^{-1} mod g_1(x) = \widehat{a_0} + \widehat{a_1}\,x + \widehat{a_2}x^2 + \widehat{a_3}x^3$$

Salida: $(\widehat{a_0}, \widehat{a_1}, \widehat{a_2}, \widehat{a_3})$

Paso 2:

Entrada: $(a_0, a_1, a_2, a_3) := (\widehat{a_0}, \widehat{a_1}, \widehat{a_2}, \widehat{a_3})$

$$\widehat{a_1} + \widehat{a_2}x + \widehat{a_3}x^2 := (a_1 + a_2x + a_3x^2)\ mod g_2(x)$$

$$(\widehat{a_1}, \widehat{a_2}, \widehat{a_3}) := (a_1, a_2, a_3)\ ;\ \widehat{a_0} := a_0$$

Salida: (a_0, a_1, a_2, a_3)

Paso 3:

Entrada: (a_0, a_1, a_2, a_3)

$$\widehat{a_2} + \widehat{a_3}x := (a_2 + a_3x)\ mod g_3(x)$$

$$(\widehat{a_2}, \widehat{a_3}) := (a_2, a_3);\ \widehat{a_0} := a_0;\ \widehat{a_1} := a_1$$

Salida: (a_0, a_1, a_2, a_3)

Paso 4:

Entrada: (a_0, a_1, a_2, a_3)

$$\widehat{a_3} := a_3; \widehat{a_0} := a_0;\ \widehat{a_1} := a_1;\ \widehat{a_2} := a_2$$

Salida: (a_0, a_1, a_2, a_3)

En resumen el cálculo de la i-ésima fila se reduce a un solo paso.

Entrada: $(a_0, a_1, a_2, a_3) := i - \acute{e}simo\ vector\ can\acute{o}nico$

$$x^{i-1}\left(b_{1,0} + b_{1,1}x + b_{1,2}x^2 + b_{1,3}x^3\right)^{-1}\ mod g_1(x) = \widehat{a_0} + \widehat{a_1}\,x + \widehat{a_2}x^2 + \widehat{a_3}x^3$$

Salida: $Fila_i := (\widehat{a_0}, \widehat{a_1}, \widehat{a_2}, \widehat{a_3})$

VI.5 Determinantes de las submatrices de orden dos del epígrafe 3.1.1

1. $\alpha^2 + \alpha^4 + \alpha^5 + \alpha^7 + (1 + \alpha + \alpha^4 + \alpha^7)b_{10} + \alpha^6 b_{10}^2$,
2. $1 + \alpha + \alpha^3 + \alpha^4 + \alpha^7 + (1 + \alpha^2 + \alpha^3 + \alpha^5 + \alpha^7)b_{10} + (\alpha + \alpha^3 + \alpha^5 + \alpha^7)b_{10}^2$,
3. $\alpha^2 + \alpha^4 + \alpha^5 + \alpha^7 + (1 + \alpha^2 + \alpha^4 + \alpha^5 + \alpha^7)b_{10} + (\alpha^3 + \alpha^5)b_{10}^2$,
4. $1 + \alpha^2 + \alpha^3 + \alpha^6 + (\alpha^2 + \alpha^3 + \alpha^4 + \alpha^5 + \alpha^6)b_{10}$,

5. $\alpha + \alpha^2 + \alpha^4 + \alpha^5 + \alpha^7 + (1 + \alpha + \alpha^2 + \alpha^4 + \alpha^5)b_{10}$,
6. $1 + \alpha^4 + \alpha^6 + \alpha^7 + (\alpha^4 + \alpha^5 + \alpha^7)b_{10}$,
7. $\alpha + \alpha^2 + \alpha^4 + \alpha^5 + (\alpha + \alpha^3 + \alpha^5)b_{10} + (1 + \alpha + \alpha^3 + \alpha^4)b_{10}^2$,
8. $1 + \alpha^6 + (1 + \alpha + \alpha^2 + \alpha^3 + \alpha^6)b_{10} + (1 + \alpha + \alpha^3 + \alpha^4 + \alpha^6)b_{10}^2$,
9. $\alpha + \alpha^2 + \alpha^4 + \alpha^5 + (\alpha + \alpha^4 + \alpha^5 + \alpha^6)b_{10} + (1 + \alpha^2)b_{10}^2$,
10. $\alpha + \alpha^2 + \alpha^6 + (\alpha^4 + \alpha^6)b_{10}$,
11. $1 + \alpha + \alpha^2 + \alpha^3 + \alpha^4 + \alpha^6 + (1 + \alpha^3 + \alpha^4 + \alpha^5)b_{10}$,
12. $\alpha + \alpha^2 + (1 + \alpha^3 + \alpha^7)b_{10}$,
13. $\alpha^2 + \alpha^7 + (1 + \alpha^2 + \alpha^3 + \alpha^5)b_{10} + (1 + \alpha + \alpha^2 + \alpha^3 + \alpha^4 + \alpha^5 + \alpha^7)b_{10}^2$,
14. $1 + \alpha + \alpha^4 + \alpha^6 + \alpha^7 + (1 + \alpha^2 + \alpha^3 + \alpha^4 + \alpha^5 + \alpha^6 + \alpha^7)b_{10} + (\alpha^3 + \alpha^4 + \alpha^7)b_{10}^2$,
15. $\alpha^2 + \alpha^7 + (1 + \alpha^3 + \alpha^5 + \alpha^6)b_{10} + (1 + \alpha^4 + \alpha^5 + \alpha^7)b_{10}^2$,
16. $1 + \alpha + \alpha^5 + \alpha^7 b_{10}$,
17. $\alpha + \alpha^2 + \alpha^4 + \alpha^7 + (\alpha + \alpha^3 + \alpha^5 + \alpha^6 + \alpha^7)b_{10}$,
18. $1 + \alpha + \alpha^2 + \alpha^3 + \alpha^6 + \alpha^7 + (\alpha + \alpha^2 + \alpha^4 + \alpha^6 + \alpha^7)b_{10}$,
19. $\alpha^3 + \alpha^5 + (1 + \alpha^2 + \alpha^3 + \alpha^6 + \alpha^7)b_{10} + (\alpha + \alpha^2 + \alpha^3)b_{10}^2$,
20. $1 + \alpha^3 + \alpha^4 + (1 + \alpha + \alpha^3 + \alpha^7)b_{10} + (\alpha^3 + \alpha^4 + \alpha^6 + \alpha^7)b_{10}^2$,
21. $\alpha + \alpha^7 + (1 + \alpha^2 + \alpha^5)b_{10} + (1 + \alpha + \alpha^2 + \alpha^4 + \alpha^7)b_{10}^2$,
22. $1 + \alpha + \alpha^2 + \alpha^3 + \alpha^4 + \alpha^5 + \alpha^6 + (\alpha + \alpha^3 + \alpha^4 + \alpha^5)b_{10} + (1 + \alpha + \alpha^3 + \alpha^5 + \alpha^6 + \alpha^7)b_{10}^2$,
23. $1 + \alpha^2 + \alpha^4 + \alpha^6 + \alpha^7 + (\alpha + \alpha^2 + \alpha^4 + \alpha^6)b_{10} + (\alpha + \alpha^6 + \alpha^7)b_{10}^2$,
24. $1 + \alpha^2 + \alpha^3 + \alpha^4 + \alpha^5 + \alpha^6 + \alpha^7 + (1 + \alpha^2 + \alpha^7)b_{10} + (1 + \alpha^2 + \alpha^4 + \alpha^5 + \alpha^6)b_{10}^2$,

25. $\alpha^2 + \alpha^6 + (\alpha^2 + \alpha^3 + \alpha^4 + \alpha^5 + \alpha^7)b_{10} + (1 + \alpha^2 + \alpha^3 + \alpha^6 + \alpha^7)b_{10}^2$,

26. $\alpha + \alpha^2 + \alpha^6 + \alpha^7 + (1 + \alpha + \alpha^3 + \alpha^4 + \alpha^6)b_{10} + (1 + \alpha + \alpha^2 + \alpha^3 + \alpha^5 + \alpha^7)b_{10}^2$,

27. $\alpha + \alpha^2 + \alpha^4 + \alpha^7 + (\alpha^2 + \alpha^3 + \alpha^5 + \alpha^6 + \alpha^7)b_{10} + (\alpha^5 + \alpha^6 + \alpha^7)b_{10}^2$,

28. $\alpha + \alpha^3 + \alpha^4 + \alpha^6 + (\alpha + \alpha^3)b_{10} + (\alpha^5 + \alpha^7)b_{10}^2$,

29. $1 + \alpha + \alpha^2 + \alpha^3 + \alpha^6 b_{10} + (\alpha^2 + \alpha^4 + \alpha^5 + \alpha^7)b_{10}^2$,

30. $1 + \alpha^4 + \alpha^5 + (\alpha^4 + \alpha^6)b_{10} + (1 + \alpha^2 + \alpha^4 + \alpha^6 + \alpha^7)b_{10}^2$,

31. $\alpha^3 + \alpha^5 + \alpha^6 + \alpha^7 + (\alpha + \alpha^2)b_{10} + (1 + \alpha + \alpha^2 + \alpha^3 + \alpha^4 + \alpha^7)b_{10}^2$,

32. $\alpha^3 + \alpha^6 + (1 + \alpha + \alpha^7)b_{10} + (1 + \alpha^3 + \alpha^4 + \alpha^5 + \alpha^6)b_{10}^2$,

33. $1 + \alpha + \alpha^7 + (1 + \alpha + \alpha^5 + \alpha^6)b_{10} + (1 + \alpha^3 + \alpha^6 + \alpha^7)b_{10}^2$,

34. $1 + \alpha^5 + (\alpha^2 + \alpha^4 + \alpha^6)b_{10} + (\alpha + \alpha^3 + \alpha^4 + \alpha^5 + \alpha^6 + \alpha^7)b_{10}^2$,

35. $1 + \alpha^2 + \alpha^5 + \alpha^6 + (1 + \alpha^2 + \alpha^6 + \alpha^7)b_{10} + (1 + \alpha^3 + \alpha^7)b_{10}^2$,

36. $\alpha + \alpha^2 + \alpha^3 + (1 + \alpha^5 + \alpha^6)b_{10} + (1 + \alpha^3 + \alpha^5 + \alpha^6)b_{10}^2$