UNIVERSIDAD DEL VALLE
DE GUATEMALA

Facultad de Ciencias y Humanidades

UN SISTEMA CRIPTOGRÁFICO BASADO EN LA TEORÍA DE CÓDIGOS CORRECTORES DE ERRORES

Modelo de trabajo profesional presentado por
Gerson Omar Cordón Salvatierra
para optar al grado académico de
Licenciado en Matemática

Guatemala

1997

UN SISTEMA CRIPTOGRÁFICO BASADO EN LA TEORÍA DE CÓDIGOS CORRECTORES DE ERRORES

UNIVERSIDAD DEL VALLE
DE GUATEMALA

Facultad de Ciencias y Humanidades

UN SISTEMA CRIPTOGRÁFICO BASADO EN LA TEORÍA DE CÓDIGOS CORRECTORES DE ERRORES

Modelo de trabajo profesional presentado por
Gerson Omar Cordón Salvatierra
para optar al grado académico de
Licenciado en Matemática

Guatemala

1997

Vo. Bo. :

f)_____

Dr. Sergio López-Permouth, Asesor

Tribunal:

f)_____

Dr. Roberto Antonio Molina Cruz

f)_____

Dr. Juan Francisco Escamilla

f)_____

Dr. Leonel Morales Aldana

Fecha de aprobación: Guatemala, 3 de octubre de 1997.

DEDICATORIA

A mi primera familia nuclear:	Mi padre, Manuel Francisco; mi madre, María del Carmen; mis hermanas: Geraldina y Georgina; mi sobrina Yéraldin, Zoila y su hijo porque no han podido dejar de ayudarme, desde que nací
A mi segunda familia nuclear:	Mi esposa Kathleen y mi hijo Yerson Manuel Lenin porque no han dejado de apoyarme, desde que me conocen
A Dorval, Ricardo, Paulo y Francisco:	por ser amigos álgebro-abstractos
A los doctores Sergio, Roberto y Raúl	por ser mis formadores y asesores
A Julio, David, Víctor, Antonio y Hugo	por ser amigos analíticos
A Fredy y Héctor	por ser amigos álgebro-lineales

Y, a todo aquello que crean que también debo dedicarla.

PREFACIO

La teoría de códigos correctores de errores (o teoría de codificación) es la que permite que las imágenes de Marte enviadas por la sonda Pathfinder sean admiradas aquí en la Tierra tan fielmente como si se estuviera en el planeta rojo. Por otro lado, la criptografía es la que permite que se envíen mensajes secretos a través de la red pública Internet. Tan disímiles como puedan parecer las dos disciplinas anteriores, sin embargo con investigación de científicos en el mundo (mayormente europeos y asiáticos) han logrado un gran desarrollo individual e interrelacionado, en los últimos 25 años.

El objetivo del presente trabajo es exponer los códigos de Goppa, los cuales son el fundamento en la teoría de codificación que hace posible tener un criptosistema de clave pública, el de McEliece. La teoría de codificación es básicamente álgebra lineal, pero los códigos de Goppa se basan en Teoría de Campos de Galois y sería requerido un curso de álgebra abstracta en teoría de anillos y campos, pero este requisito es optativo. Sin embargo es posible seguir toda la línea general de pensamiento habiendo tomado un primer curso de álgebra lineal y un curso de cálculo que contenga series. Además se dan las nociones de criptografía necesarias en el primer capítulo y hay un apéndice sobre complejidad de algoritmos para el neófito en estas materias.

Es relevante reconocer que no encontrándose literatura en español ni traducida sobre los temas, se ha hecho libre traducción de la jerga pertinente usada, de la que puede culparse únicamente al autor. Sin embargo, se considera que en este aspecto es una contribución importante como obra de consulta en nuestra lengua nativa.

Respecto a la numeración de las proposiciones consideradas se ha utilizado el llamado sistema decimal, que consiste en asignar a cada proposición la identificación que le corresponde de acuerdo a su profundidad en las subdivisiones. Por ejemplo, existe una proposición designada como IV.C.1, que indica el capítulo IV, título C, subtítulo 1.

CONTENIDO

RESUMEN

El esquema de encripción pública de McEliece está basado en los códigos correctores de error. La idea detrás del esquema es seleccionar primero un código particular para el cual un algoritmo de descodificación eficiente se conoce. Dado que el problema de descodificar un código lineal es **NP-completo,** una descripción del código original puede servir como la clave privada, mientras que una descripción del código transformado sirve como clave pública.

El esquema de encripción de McEliece (cuando se usa con los códigos de Goppa) ha resistido criptoanálisis a la fecha. También es notable que ha sido el primer esquema de clave pública que usa aleatorización durante el proceso. Aunque muy eficiente, el esquema de McEliece ha recibido poca atención en la práctica debido a sus claves públicas muy grandes.

I. INTRODUCCIÓN

Al introducir el presente estudio se dice que dentro de las nociones de Criptografía, ésta es definida como la ciencia, o arte, de escribir secretamente. La palabra se deriva del griego, κριπτο(kripto), por escondido, y λραφια (grafía), algo que está escrito. La transmisión secreta de mensajes es uno de los problemas más antiguos del mundo. La Criptografía ha sido empleada desde la antigüedad por hombres de estado, embajadores y el ejército.

En 360 a.C., Aeneas Tacticus asignó un capítulo completo a mensajes secretos en su manual militar. Describe varios métodos los cuales, en estos tiempos, parecerían fuera de época. Uno de estos métodos, del cual estuvo particularmente orgulloso, consistía en perforar un hueso de oveja (suficientemente grande) con tantos agujeros como letras hubiesen en el alfabeto. Entonces tomaba una hebra y la insertaba a través de cada agujero correspondiente a la letra del mensaje. Para descifrar el mensaje, todo lo que tenía que hacerse era sacar la hebra y marcar los agujeros por los cuales pasaba.

En 400 a.C., Spartes usó un tipo de palo con forma de cono, conocido como "cuento del cielo", alrededor del cual enrollaban un pergamino. Entonces, escribían el mensaje verticalmente en las varias circunferencias en espiral, y lo desenrollaban, de tal forma que las letras quedaban mezcladas. Para reconstruir el mensaje, el receptor tenía que poseer un palo del mismo grosor alrededor del cual simplemente tenía que enrollar el pergamino.

La idea básica siempre es la misma: mezclar el mensaje para hacerlo ininteligible para cualquier extraño. Usualmente, el mensaje M es una secuencia finita de símbolos de algún alfabeto S. El mensaje encriptado se denota como e(M). La función de **desencriptamiento** se denota d, y verifica d(e(M)) = M. En general e y d dependen de un parámetro el cual es llamado la **clave** y se denota K. Entonces C=e(M,K) es llamado **criptograma** (o **texto cifrado**) y M = d(C,K) es llamado **texto plano** (o **texto limpio**).

Así un sistema criptográfico puede definirse como una tripleta <M, K, C>, donde M (resp. K) está constituido de símbolos de un alfabeto S_1 (resp. S_2). Más aún deben existir dos funciones e y d:

$$e : M \times K \rightarrow C \qquad d : C \times K \rightarrow M$$

las cuales cumplen que para cada $(m,k) \in M \times K$

$$d(e(m,k),k) = m.$$

Esto posee el siguiente problema. Si el receptor del mensaje no conoce la clave d entonces no puede recuperar M. Así el creador del sistema criptográfico tiene que distribuir la clave a todos los que quieran comunicarse con él. Pero esto no es seguro si hay un usuario deshonesto.

Más aún hay otro problema. Visualícese el desarrollo de una red de computadoras que consiste de mil usuarios donde cada par de usuarios requiere una clave separada para comunicarse entre sí. Sería instructivo pensar en el grafo completo de n vértices, representando los usuarios; con las $\frac{n(n+1)}{2}$ aristas correspondientes a las necesidades de intercambio de información. Así en la red de 1,000 usuarios, aproximadamente 500,000 claves deben intercambiarse de alguna forma que no sea la red.

Es posible considerar que una clave pueda ser dividida en dos partes, K=(E,D), tal que E es necesaria para el encriptamiento, mientras que D sería necesaria para el desencriptamiento.

Si fuera posible idear un criptosistema tal, entonces los siguientes beneficios serían conseguidos. Primero que todo, dado que la información necesaria para el encriptamiento no provee, a priori, a un atacante con suficiente información para desencriptar, entonces ya no hay una razón para tenerla secreta. Consecuentemente E puede ser hecha pública a todos los usuarios de la red. Un criptosistema concebido de tal forma es llamado un **criptosistema público** (o **PKC**).

Estos criptosistemas están basados en funciones de **una-vía**. Una definición no formal de estas funciones sigue.

Sean S y T dos conjuntos cualesquiera, una función de una-vía es una función $f : S \rightarrow T$ la cual cumple:

(1) $\forall x \in S$, f(x) puede ser "fácilmente" calculada

(2) No hay una forma "fácil" de recuperar x dado y = f(x), aún si un número grande de elementos de T son conocidos.

Obviamente, una forma de obtener x, conociendo f(x), es ir a través del conjunto S. Pero si la card(S) $\approx 2^n$, con n grande, entonces es imposible hacerlo en un tiempo finito.

Para cada usuario U_i, escójase un par (K_i,L_i) tal que cada mensaje M

$$d(e(M,K_i),L_i) = M$$

En un directorio público de sistema ancho, lístense todas las claves "públicas" K_i. L_i es la clave privada de cada usuario U_i. Si un usuario U_j quiere enviar un mensaje a U_i, procedería como sigue:

a) U_j toma la clave K_i de U_i

b) U_j envía $C = e(M,K_i)$.

Así, en lugar de tener que manejar la distribución secreta de $O(n^2)$ claves en una red de n usuarios, solo n claves son requeridas, y no necesitan ser distribuidas secretamente.

Para hacer seguro el sistema, e y d tienen que verificar cuatro restricciones fundamentales:

(1) C = e(M,K) puede ser fácilmente calculada con M y K,

(2) Es imposible calcular M sabiendo solo C,

(3) M puede ser fácilmente calculada con C y la clave secreta L.

En otras palabras, (1) y (2) implican que e es una función de una-vía. Pero (3) implica que existe una clave tal que e es invertible. Una función de una-vía con esta característica es llamada una **"función trampa"**.

(4) El par (K_i,L_i) puede ser fácilmente calculado.

En otras palabras, debe haber un gran número de pares (K,L) tal que sea imposible describir una lista exhaustiva. Desde la aparición de la Criptografía de clave pública, básicamente todos los protocolos prácticos han sido basados en problemas difíciles de teoría de números, y recaído en operaciones aritméticas con números grandes. Más aún, muchos de estos protocolos dependen (peligrosamente) de uno y sólo un problema: factorización. La intención del presente trabajo es presentar un PKC basado en códigos correctores de errores: el criptosistema de McEliece.

II. CÓDIGOS CORRECTORES DE ERRORES

El problema principal de la teoría de la codificación puede ser descrito de una forma simple. Supóngase que una cadena de datos fuente, digamos en la forma de bits (0's y 1's), está siendo transmitida a través de un canal de comunicación, tal como una línea telefónica. De tiempo en tiempo, distorsiones ocurren a través del canal, causando que algunos de los 0's se vuelvan 1's y viceversa. La pregunta es: ¿Cómo se puede decir cuando la fuente original de datos ha sido cambiada, y cuando lo ha hecho, como se pueden recuperar los datos originales?

A. GENERALIDADES

Definición II.A.1. Sea $p > 1$ un número primo. GF(p) es el campo finito que contiene p elementos, Z/pZ.

Definición II.A.2. Un código de longitud n, sobre A, es un subconjunto C de A^n. A es llamado ***alfabeto***, n la ***longitud*** del código C , y los elementos de C son llamados ***palabras-código***.

Ejemplo: El código:

$$C = \{ (0,0,0,1), (1,1,1,0), (0,1,0,1), (1,1,0,1), (1,1,1,1) \}$$

es un código de longitud 4 sobre el alfabeto $A = \{ 0,1 \}$.

Definición II.A.3. Un código C_1 es ***equivalente*** a un código C_2 si cada palabra de C_1 viene de una palabra permutada de C_2. La permutación siendo la misma para cada palabra.

Definición II.A.4. Sea A un alfabeto, A^n el conjunto de palabras de longitud n sobre A, $x=(x_1,...,x_n) \in A^n$ & $y=(y_1,...,y_n) \in A^n$. La ***distancia de Hamming*** entre x & y está dada por $d_H(x,y) = \#\{ i \mid x_i \neq y_i, 1 \leq i \leq n \}$.

Definición II.A.5. La ***distancia mínima*** de un código C sobre A es el entero d definido por

$$d = \min\{ d_H(x,y) \mid x,y \in C \ \& \ x \neq y \}.$$

Nota: Generalmente la búsqueda de la distancia mínima de un código es un problema NP-completo (ver apéndice A).

Sea $x = (x_1,...,x_n)$ una palabra-código de C , sobre un alfabeto A. Supóngase que esta palabra es enviada a través de un canal perturbado, sea x' la palabra recibida. El problema es decidir si x'=x, y si no, como recuperar la palabra x. La intención de la teoría de codificación es encontrar "buenos" códigos (sobre "buenos" alfabetos) permitiendo determinar propiedades concernientes a la detección de errores, corrección de errores, y la razón "poder de corrección/longitud del código".

B. CÓDIGOS CORRECTORES DE E ERRORES

Definición II.B.1. Un código C de longitud n, sobre un alfabeto A, es ***corrector de e errores*** si cuando más e (pero al menos uno) errores son hechos en una palabra-código, la palabra resultante no es una palabra-código. Es decir que para cada palabra x' de A^n, existe a lo más una palabra x de C tal que $d_H(x,x') \leq e$. Esto es equivalente a decir que las bolas de radio e alrededor de cada palabra-código son disjuntas.

Teorema II.B.2. Sea d la distancia mínima del código C . Si $d \geq 2e + 1$ entonces C es corrector de e errores.

Prueba: Tómese como hipótesis que C tiene por lo menos distancia mínima 2e + 1 y que una palabra w es recibida, junto con la información de que han ocurrido e errores. Si hubieran dos palabras-código u & v a una distancia e de w, entonces por la desigualdad triangular $d_H(u,v) \leq 2e$, que contradice la hipótesis. Por lo tanto hay una única palabra-código u a distancia a lo más e de w y puede deducirse que u debió haber sido transmitida. QED

Nota: Así si la distancia mínima de un código C es d, se dice corrector de $\lfloor \frac{d-1}{2} \rfloor$ errores.

Los códigos más interesantes son aquellos cuyo alfabeto A es un campo finito. Entonces, C tiene estructura de espacio vectorial. Estos códigos son llamados *códigos lineales.*

C. CÓDIGOS LINEALES

Definición II.C.1. Un ***código lineal*** de longitud n sobre GF(q) es un ***subespacio vectorial*** de $[GF(q)]^n$. Si C tiene dimensión k sobre $[GF(q)]^n$, se dice que C es un código (n,k), y si C tiene distancia mínima d, se dice que C es un código (n,k,d).

Proposición II.C.2. Si C es un código (n,k) sobre GF(q), entonces C tiene q^k palabras-código.

Prueba: Sea $\{ v_1,v_2,...,v_k \}$ una base para C , entonces toda palabra $w \in C$ puede ser escrita de manera única como

$$w = a_1v_1 + a_2v_2 + ... + a_kv_k$$

para $a_i \in GF(q)$. Como cada a_i puede ser alguno de los q elementos de GF(q), entonces el número total de dichas combinaciones lineales distintas de los v_i es q^k. QED

Definición II.C.3. Sea C un código lineal (n,k), el *peso* ω (x) de una palabra $x \in C$ es el número de posiciones no cero en x, i.e.

$$\omega (x) = d_H(x,0) = \#\{ i \mid x_i \neq 0, 1 \leq i \leq n \}$$

Proposición II.C.4. Si C es un código lineal entonces $\forall x,y \in C$

$$d_H(x,y)=\omega(x-y).$$

Prueba: Por definiciones de distancia de Hamming y peso se tiene

$$d_H(x,y)=\#\{ i \mid x_i \neq y_i, 1 \leq i \leq n \}=\#\{ i \mid x_i - yi \neq 0, 1 \leq i \leq n \}=\omega(x-y).$$ QED

Definición II.C.5. El peso (mínimo) $\omega_{min}(C)$ de un código lineal C es el peso mínimo de todas las palabras-código no cero en C .

1. Matriz generadora

Definición II.C.1.1. Sea C un código (n,k) sobre GF(q). Una matriz G_{kxn} cuyas filas forman una base para C es llamada una ***matriz generadora*** para C .

Definición II.C.1.2. Una matriz generadora de la forma $G=(I_k| M)$, con I_k la matriz identidad de tamaño k, se dice que está en ***forma típica***.

Proposición II.C.1.3. Todo código C es equivalente a un código C' cuya matriz generadora esté en forma típica.

Prueba: Sea M una matriz generadora para C. Es posible mediante operaciones elementales de fila convertir M en una matriz M' que esté en forma escalonada reducida. Las filas de M' son combinaciones lineales de las filas de M, las cuales son palabras-código. Así que las filas de M' son palabras-código. Ahora se permutan las columnas de M' para producir una matriz M'' tal que las primeras entradas no cero en la i-ésima fila estén en la i-ésima columna. Esto corresponde a permutar los símbolos de las palabras-código para producir un código C'. Claramente M'' empieza con una matriz identidad. Más aún M'' es una matriz generadora para C'. QED

2. Matriz de control

Definición II.C.2.1. Sean x & y dos elementos de $[GF(q)]^n$, el ***producto escalar*** de x & y se define como $< x,y > = x_1y_1 + ... + x_ny_n$.

Definición II.C.2.2. Sea C un código lineal (n,k) sobre GF(q). El conjunto

$C^{\perp} = \{ x \in [GF(q)]^n \mid < x,c >=0, \forall c \in C \}$

es llamado el ***código dual (u ortogonal)*** de C .

Proposición II.C.2.3. El código dual de un código (n,k) es un código (n,n-k).

Prueba: Recuérdese que dim[C] + dim[$C^{\perp}$]=n y por hipótesis, dim[C]=k, por lo tanto dim[$C^{\perp}$]= n-dim[C]=n-k. QED

Definición II.C.2.4. Una ***matriz de control (o de revisión de paridad)*** del código C, es una matriz generadora del código $C^{\perp}$.

Proposición II.C.2.5. Sea H una matriz de control del código C(n,k).

Sea $x = (x_1,...,x_n) \in [GF(q)]^n$ entonces:

$$x \in C \Leftrightarrow Hx^{\perp}=0$$

donde $x^{\perp}$ denota la transpuesta de la palabra x.

Prueba: Evidente, solamente notando que H es la matriz generadora de $C^{\perp}$ significando que sus filas son palabras-código del dual. Al hacer $Hx^{\perp}$, cada producto escalar se anula por ortogonalidad. Esta es la equivalencia que establece la proposición. QED

Definición II.C.2.6. Sea C un código lineal (n,k) sobre GF(q), sea $y = (y_1,...,y_n) \in [GF(q)]^n$, el ***síndrome*** de y es la palabra de $[GF(q)]^{n-k}$ definida por :

$$S = Hy^{\perp}.$$

Proposición II.C.2.7. Sea C un código lineal, y G su matriz generadora en forma típica, $G = (I_k \mid M)$. Entonces $H=(-M^T \mid I_{n-k})$ es una matriz de control del código C .

Prueba: Puede verse por cómputo directo que las filas de G y H son ortogonales entre sí. Revísese esto siendo $M=(m_{ij})$. Entonces la primera fila de G, $g_1=(1,0,...,0,m_{11},m_{12},...,m_{1(n-k)})$, y la primera fila de H, $h_1=(-m_{11},-m_{21},...,-m_{k1},1,0,...,0)$, así que $g_1 \bullet h_1=-m_{11}+m_{11}=0$. Claramente G tiene rango k y H tiene rango (n-k). QED

D. CÓDIGOS CÍCLICOS

Definición II.D.1. Un código C es ***cíclico*** si:

i) C es un código lineal

ii) $(x_1,...,x_n) \in C \Rightarrow (x_n,x_1,...,x_{n-1}) \in C$.

Sea $m=(m_0,m_1,...,m_{n-1})$ una palabra-código de un código cíclico C (n,k), e identifíquese el polinomio $m(x)=m_0+m_1x+...+m_{n-1}x^{n-1}$. Entonces C es cíclico si y solo si, para toda palabra-código identificada m, $x \bullet m(x)[\text{mod}(x^n-1)]$ corresponde a una palabra-código.

Definición II.D.2. La ***forma polinomial***, C (x), de un código cíclico C , es el conjunto $\{m(x) \mid m \in C\}$.

Proposición II.D.3. Sea g(x) un polinomio no cero mónico de grado mínimo, entonces g(x) *genera la* forma polinomial del código cíclico C en el sentido de que cualquier palabra-código $w(x) \in C$ puede ser escrita en la forma w(x)=f(x)g(x) para un polinomio adecuado f(x). Tal polinomio g(x) es llamado el ***generador*** del código.

Prueba: Asúmase que existe un w(x) que no puede escribirse así. Entonces por el algoritmo de la división w(x)=q(x)g(x)+r(x), con el grado de r(x) menor que el de g(x). Dado que w(x) y q(x)g(x) son palabras-código y C es lineal, entonces $r(x) \in C$ que contradeciría el grado mínimo de g(x). Así que un tal w(x) no existe. QED

Proposición II.D.4. Todo código cíclico no cero, de longitud n, sobre GF(q), tiene un único generador mónico de grado mínimo g(x) que es divisor de x^n-1.

Prueba: De lo contrario, podría escribirse (por el algoritmo euclidiano) $x^{n-1}=g(x)q(x)+r(x)$, donde el grado de r(x) es menor que el grado de g(x). Ahora $r(x)=-a(x)g(x)[\text{mod } x^{n-1}]$ y así r(x) está en $<g(x)>$. Esto sería una contradicción con la minimalidad del grado de g(x) a menos que r(x)=0 y por lo tanto g(x) divide a x^{n-1}. QED

Proposición II.D.5. La dimensión del código cíclico C , de longitud n, cuyo generador es g(x), es k = n-deg[g(x)].

Prueba: La dimensión se puede probar mostrando una base, pero adicionalmente se probará que si $g(x)=g_0+g_1x+...+g_{n-k}x^{n-k}$, entonces una matriz generadora para el código cíclico C es la matriz

$$\begin{pmatrix} g_0 & g_1 & \cdots & g_{n-k} & 0 & \cdots & 0 \\ 0 & g_0 & \cdots & g_{n-k-1} & g_{n-k} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & & & & g_{n-k} \end{pmatrix}.$$

Los vectores g(x), g(x)x, $g(x)x^2$,..., $g(x)x^{k-1}$ son linealmente independientes porque si no, habrían elementos a_i, $0 \le i \le k-1$, que $a_0g(x)+a_1g(x)+...+a_{k-1}g(x)x^{k-1}=(a_0+a_1x+...+a_{k-1}x^{k-1})g(x)=0$. Pero este producto tiene grado menor que n así que no puede ser $0[\text{mod } x^n-1]$ a menos que cada a_i sea 0. Para ver que estos vectores generan C , nótese que cualquier s(x) en C puede ser expresado como c(x)g(x) donde el grado de c(x) es menor o igual que k-1. Por lo tanto $c(x)g(x)=(c_0+c_1x+...+c_{k-1}x^{k-1})g(x)=c_0g(x)+c_1g(x)+...+c_{k-1}g(x)x^{k-1}$. De esto se sigue que una matriz generadora de C es la matriz cuyas filas son k (que es la dimensión del código) y la primera es g(x)x, la segunda es $g(x)x^2$,...,hasta $g(x)x^{k-1}$. QED

Los códigos cíclicos son los más usados en la práctica. Entre ellos, existen códigos particulares cuya estimación de la distancia mínima es conocida antes de la construcción del código. Este es el caso de los códigos BCH y los Reed-Solomon. Éste es el caso también de los códigos de Goppa aunque éstos generalmente no son cíclicos. Más aún, para estos tres códigos, existen algoritmos descodificadores de orden polinomial.

1. Códigos BCH (Bose, Chauduri, Hocquenghem)

Definición II.D.1.1. Sea C un código cíclico de longitud n sobre GF(q) y sea g(x) su polinomio generador. C es un código BCH de ***distancia asignada*** δ si δ es el entero más grande tal que

$$\exists b \in Z,\ g(\beta^b)=g(\beta^{b+1})=\ldots=g(\beta^{b+\delta-2})=0$$

donde β es una n-ésima raíz primitiva de la unidad† sobre $GF(q^m)$, m siendo el entero más pequeño tal que $n \mid q^m-1$.

Definición II.D.1.2. Cuando b es 1 el código es llamado BCH en ***sentido reducido.***

Teorema II.D.1.3. (cota BCH) Sea C un código BCH de distancia asignada δ, su distancia mínima verifica que

$$d \geq \delta.$$

Prueba: Sea la matriz de control del código

$$H=\begin{pmatrix} 1 & \beta^b & (\beta^b)^2 & \cdots & (\beta^b)^{n-1} \\ 1 & \beta^{b+1} & (\beta^{b+1})^2 & \cdots & (\beta^{b+1})^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \beta^{b+\delta-2} & (\beta^{b+\delta-2})^2 & \cdots & (\beta^{b+\delta-2})^{n-1} \end{pmatrix}.$$

Se muestra que ni d-1 ni menos columnas de H son linealmente dependientes sobre $GF(q^m)$. Para este propósito escójanse un conjunto de d-1 columnas encabezadas por los elementos $(\beta^b)^{i_1}$,..., $(\beta^b)^{i_{d-1}}$ y fórmese el determinante (d-1)×(d-1). Se factorizan los encabezados de columna y se obtiene:

---

† Significa que $\beta^n=1$ y $\beta^m \neq 1$ para $0<m<n$.

$$(\beta^b)^{(i_1+\ldots+i_{d-1})}\begin{vmatrix} 1 & 1 & \ldots & 1 \\ \beta^{i_1} & \beta^{i_2} & \ldots & \beta^{i_{d-1}} \\ \vdots & \vdots & \ddots & \vdots \\ \beta^{(d-2)i_1} & \beta^{(d-2)i_2} & \ldots & \beta^{(d-2)i_{d-1}} \end{vmatrix}.$$

Se ve que la matriz que consista de cualesquiera δ-1 columnas de H es un múltiplo de una matriz cuyo determinante es Vandermonde y no cero ya que las raíces de g(x) son todas diferentes. Por lo tanto cualesquiera δ-1 columnas de H son independientes sobre GF(q^m) y así sobre GF(q), lo que muestra que $d \geq \delta$. QED

2. Códigos Reed-Solomon generalizados

Definición II.D.2.1. Sea $\alpha=(\alpha_1,\ldots,\alpha_n)$ donde los α_i son elementos distintos de GF(q^m), y sea $v=(v_1,\ldots,v_n)$ donde los v_i son elementos no cero (pero no necesariamente distintos) de GF(q^m). Entonces el código de Reed-Solomon generalizado (n,k), $GRS_k(\alpha,v)$, es el conjunto de todos los vectores de la forma:

$$(v_1F(\alpha_1),\ldots,v_nF(\alpha_n))$$

donde F(x) es cualquier polinomio de grado menor que k, con coeficientes de GF(q^m).

La matriz de control de un código $GRS_k(\alpha,v)$ es:

$$H = \begin{pmatrix} \gamma_1 & \gamma_2 & \cdots & \gamma_n \\ \gamma_1\alpha_1 & \gamma_2\alpha_2 & \cdots & \gamma_n\alpha_n \\ \vdots & \vdots & \ddots & \vdots \\ \gamma_1\alpha^{n-k-1}{}_1 & \gamma_2\alpha^{n-k-1}{}_2 & \cdots & \gamma_n\alpha^{n-k-1}{}_n \end{pmatrix} =$$

$$= \begin{pmatrix} 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_n \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{n-k-1} & \alpha_2^{n-k-1} & \cdots & \alpha_n^{n-k-1} \end{pmatrix} \begin{pmatrix} \gamma_1 & 0 & \cdots & 0 \\ 0 & \gamma_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \gamma_n \end{pmatrix}$$

donde $\gamma=(\gamma_1,...,\gamma_n)$ ($\gamma_i \in GF(q^m)\setminus\{0\}$) verifica que $GRS_k(\alpha,v)^{\perp}= GRS_{n-k}(\alpha,\gamma)$. Es decir que esta matriz es generadora del código $GRS_{n-k}(\alpha,\gamma)$.

E. Un primer acercamiento a los códigos de Goppa

Sea C un código BCH (sentido reducido) de longitud n, y de distancia mínima d, sobre GF(q). Sea g(x) un generador del código, g tiene d raíces consecutivas. Más precisamente, se tiene

$$g(\beta)=g(\beta^2)=...=g(\beta^{d-1})=0$$

donde β es una n-ésima raíz primitiva de la unidad. Sea $c=(c_0,...,c_{n-1}) \in C$ y c(x) su forma polinomial. Siendo toda palabra-código un múltiplo del generador, se tiene

$\exists a(x)$ tal que $c(x)=a(x)g(x)$. Así

$c(\beta)=c(\beta^2)=...=c(\beta^{d-1})=0$ que da

$$\sum_{i=0}^{n-1} c_i (\beta^j)^i = 0, \quad 1 \le j \le d .$$

Proposición II.E.1. Sea C un código BCH en sentido reducido, de longitud n y distancia mínima d, existe un polinomio p tal que

$$\sum_{i=0}^{n-1} \frac{c_i}{z-\beta^{-i}} = \frac{z^{d-1}p(z)}{z^n-1}.$$

Prueba: Se tiene:

$$\frac{z^{n-1}}{z-\beta^{-i}} = \sum_{k=0}^{n-1} z^k (\beta^{-i})^{n-1-k} = \sum_{k=0}^{n-1} \beta^{i(k+1)} z^k$$

Considérese la sucesión $(u_k)_{k\in N}$ definida por $u_k=z^k(\beta^{-i})^{n-1-k}$ que es una progresión geométrica de razón $z\beta^i$. Así:

$$S_{n-1} = \frac{(1-z^n)\beta^i}{1-z\beta^i} = \frac{z^n-1}{z-\beta^{-i}}$$, dado que $\beta^n=1$, luego

$$S_{n-1} = \sum_{k=0}^{n-1} u_k = \sum_{k=0}^{n-1} z^k (\beta^{-i})^{n-1-k} = \sum_{k=0}^{n-1} z^k \beta^{i(k+1)}.$$

Sea $(c_0,...,c_{n-1})$ una palabra-código, entonces:

$$\sum_{i=0}^{n-1} c_i \frac{z^n - 1}{z - \beta^{-i}} = \sum_{i=0}^{n-1} c_i \sum_{k=0}^{n-1} z^k \beta^{i(k+1)} = \sum_{k=0}^{n-1} z^k \underbrace{\sum_{i=0}^{n-1} c_i (\beta^{k+1})^i}_{=0 \, para \, 0 \le k < d-1} =$$

$$= \sum_{k=d-1}^{n-1} z^k \sum_{i=0}^{n-1} c_i \beta^{i(k+1)} = a_{d-1} z^{d-1} + a_d z^d + ... + a_{n-1} z^{n-1} = z_{d-1} p(z). \quad \text{QED}$$

Proposición II.E.2. Sea g un polinomio tal que $g(\gamma) \neq 0$ entonces:

$$\frac{1}{z-\gamma} \equiv \frac{-1}{g(\gamma)} \left(\frac{g(z) - g(\gamma)}{z - \gamma} \right) [\text{mod } g(z)].$$

Prueba: Se busca un polinomio r tal que:

$$r(z)(z-\gamma) \equiv 1 [\text{mod } g(z)]$$

esto implica que existe un polinomio q tal que:

$$r(z)(z-\gamma) = q(z)g(z) + 1$$

que es lo mismo que decir:

$$(z-\gamma)r(z) + g(z)[-q(z)] = 1 \quad (**)$$

ahora $(z-\gamma)$ & $g(z)$ son primos entre sí, dado que $g(\gamma) \neq 0$, por tanto por el teorema de Bezout, γ es raíz de $q(z)$ y $\deg[q] < \deg[(z-\gamma)] = 1$. Esto implica que $\deg[q] = 0$, así que g es un polinomio constante. Tomando $z = \gamma$ y sustituyendo por z en (**), se tiene

$$q(\gamma) = \frac{-1}{g(\gamma)} \Rightarrow q = \frac{-1}{g(\gamma)} \text{ y por lo tanto}$$

$$r(z) = \frac{q(z)g(z) + 1}{z - \gamma} = \frac{-1}{g(\gamma)} \left(\frac{g(z) - g(\gamma)}{z - \gamma} \right). \quad \text{QED}$$

Ahora con esta proposición y con la anterior se puede caracterizar una palabra-código c por:

$$\sum_{i=0}^{n-1} \frac{c_i}{z-\beta^{-i}} \equiv 0[\text{mod } z^{d-1}].$$

Ciertamente:

$$\frac{1}{z-\beta^{-i}} \equiv -\beta^i \left(\frac{1-(\beta^i z)^{d-1}}{1-\beta^i z} \right) [\text{mod} z^{d-1}], \text{ ahora}$$

$$\frac{1-(\beta^i z)^{d-1}}{1-\beta^i z} = \sum_{j=0}^{d-2} (\beta^i z)^j, \text{ entonces}$$

$$\frac{1}{z-\beta^{-i}} \equiv -\sum_{j=0}^{d-2} z^j \beta^{i(j+1)} [\text{mod } z^{d-1}], \text{ luego}$$

$$z^{d-1} p(z) =^n \sum_{i=0}^{n-1} \frac{c_i}{z-\beta^{-i}} - \sum_{i=0}^{n-1} \frac{c_i}{z-\beta^{-i}},$$

$$z^n \sum_{i=0}^{n-1} \frac{c_i}{z-\beta^{-i}} \equiv 0[\text{mod } z^{d-1}],$$

$$z^{d-1} p(z) \equiv 0[\text{mod } z^{d-1}].$$

Por lo tanto:

$$\sum_{i=0}^{n-1} \frac{c_i}{z-\beta^{-i}} \equiv 0 \quad [\text{mod } z^{d-1}]..$$

En el próximo capítulo, que concierne a los códigos de Goppa, se verá que estos códigos pueden caracterizarse con una ecuación similar.

III. CÓDIGOS DE GOPPA

A. DEFINICIONES

Definición III.A.1. Sea $m \in N$, sea $g(z)$ un polinomio de grado t con coeficientes de $GF(q^m)$. Sea $L=\{\eta_1,...,\eta_n\} \subset GF(q^m)$, tal que $card(L)=n$ y $g(\eta_i)\neq 0$ para $1 \leq i \leq n$. El ***código de Goppa*** $\Gamma(L,g)$, de longitud n, está definido por el conjunto de palabras $c=(c_1,...,c_n)$ sobre el alfabeto $GF(q)$ para las cuales:

$$R_c(z) = \sum_{i=1}^{n} \frac{c_i}{z-\eta_i} \equiv 0 \quad [\text{mod } g(z)]..$$

Si $g(z)$ es irreducible entonces $\Gamma(L,g)$ es llamado ***irreducible***. Claramente los códigos de Goppa son lineales.

Ejemplo: Sea $g(z)=z^{d-1}$, $L=\{\beta^{-i} \mid 0 \leq i \leq n-1\}$ donde β es una n-ésima raíz primitiva de la unidad en $GF(q^m)$. Sea $c=(c_1,...,c_n)$, entonces $c \in \Gamma(L,g)$ si y solo si:

$$\sum_{i=1}^{n} \frac{c_i}{z-\beta^{-i+1}} \equiv 0[\text{mod } z^{d-1}].$$

Así, conforme al capítulo anterior, $\Gamma(L,g)$ es, en este caso, un código BCH de longitud n y distancia mínima d.

B. MATRIZ DE CONTROL

Conforme al capítulo previo y conviniendo que $h_i = \frac{1}{g(\eta_i)}$, se tiene:

$$R_c(z) \equiv \sum_{i=1}^{n} c_i \frac{g(z)-g(\eta_i)}{z-\eta_i} h_i [\text{mod } g(z)], \text{ por lo tanto:}$$

$$c \in \Gamma(L,g) \Leftrightarrow \sum_{i=1}^{n} c_i \frac{g(z)-g(\eta_i)}{z-\eta_i} h_i = 0, \text{ que puede ser escrito:}$$

$$\left(\frac{g(z)-g(\eta_1)}{z-\eta_1} h_1, \ldots, \frac{g(z)-g(\eta_n)}{z-\eta_n} h_n \right) (c_1, \ldots, c_n)^T = 0.$$

Supóngase que g es de grado r entonces $g(z)=\sum_{i=0}^{r} g_i z^i$. Por lo tanto:

$$g(z)-g(x)=\sum_{i=0}^{r} g_i (z^i - x^i)\text{. Ahora:}$$

$z^i-x^i=(z-x)(z^{i-1}+z^{i-2}x+...+zx^{i-2}+x^{i-1})$. Así:

$$\frac{z^i - x^i}{z-x} = \sum_{j=0}^{i-1} z^j x^{i-1-j}\text{ . Ahora:}$$

$$\frac{g(z)-g(x)}{z-x} = \sum_{i=0}^{r} g_i \sum_{j=0}^{i-1} z^j x^{i-1-j} =$$

$$= \sum_{i=0}^{r} g_i \sum_{j=0}^{r} z^j x^{i-1-j} \text{ al asumir que } x^{i-1-j}=0 \text{ para } j > i-1.$$

Sea λ= i-1-j, entonces:

$$\frac{g(z)-g(x)}{z-x} = \sum_{j=0}^{r} z^j \sum_{\lambda=0}^{r} g_{\lambda+j+1} x^{\lambda}$$

con $g_{\lambda+j+1}=0$ para $\lambda+j+1>r$.

Al determinar para cuáles coeficientes están estas sumas definidas, se nota primero que, $\frac{g(z)-g(x)}{z-x}$, es un polinomio de grado ≤ r-1. Así, la primera suma, está definida para $0 \le j \le r-1$. Más aún, los g_k son los coeficientes de g(z), por lo tanto la segunda suma, está definida para $0 \le \lambda+j+1 \le r$. Por lo que los coeficientes de z^i son:

$$\sum_{j=i+1}^{r} g_j x^{j-i-1}\text{ . Por lo tanto:}$$

$$\frac{g(z)-g(x)}{z-x} = \sum_{j=0}^{r-1} z^j \sum_{\lambda=j+1}^{r} g_\lambda x^{\lambda-j-1}$$

Así $c \in \Gamma(L,g)$ si y solo sí:

$$\sum_{i=1}^{n} c_i h_i \sum_{j=0}^{r-1} z^j \sum_{\lambda=j+1}^{r} g_\lambda \eta_i^{\lambda-j-1} = 0 \Leftrightarrow$$

$$\Leftrightarrow \sum_{j=0}^{r-1} z^j \sum_{i=1}^{n} c_i h_i \sum_{\lambda=j+1}^{r} g_\lambda \eta_i^{\lambda-j-1} = 0$$

a medida que este polinomio es cero, implica que:

$$\forall j=0,...,r-1 \sum_{i=1}^{n} c_i h_i \sum_{\lambda=j+1}^{r} g_\lambda \eta_i^{\lambda-j-1} = 0$$

que es igual que decir:

$$\forall j=0,..,r-1 \left(h_1 \sum_{\lambda=j+1}^{r} g_\lambda \eta_1^{\lambda-j-1}, \ldots, h_n \sum_{\lambda=j+1}^{r} g_\lambda \eta_n^{\lambda-j-1} \right) (c_1, \ldots, c_n)^T = 0$$

o escrito como producto de matrices, $Hc^T=0$ donde:

$$H=\begin{pmatrix} g_r h_1 & \cdots & g_r h_n \\ (g_{r-1}+\eta_1 g_r)h_1 & \cdots & (g_{r-1}+\eta_n g_r)h_n \\ \vdots & \ddots & \vdots \\ (g_1+\eta_1 g_2+\ldots+\eta_1^{r-1} g_r)h_1 & \cdots & (g_1+\eta_n g_2+\ldots+\eta_n^{r-1} g_r)h_n \end{pmatrix}$$

La matriz H obtenida es una matriz de control del código. Generalmente, otra matriz de control es considerada.

Proposición III.B.1. Sea $\Gamma(L,g)$ un código de Goppa de longitud n, con g de grado r, entonces:

$$\begin{pmatrix} h_1 & \cdots & h_n \\ \eta_1 h_1 & \cdots & \eta_n h_n \\ \vdots & \ddots & \vdots \\ \eta_1^{r-1} h_1 & \cdots & \eta_n^{r-1} h_n \end{pmatrix}$$

es una matriz de control del código.

Prueba: $H=\begin{pmatrix} g_r & 0 & \cdots & 0 \\ g_{r-1} & g_r & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ g_1 & g_2 & \cdots & g_r \end{pmatrix}\begin{pmatrix} h_1 & \cdots & h_n \\ \eta_1 h_1 & \cdots & \eta_n h_n \\ \vdots & \ddots & \vdots \\ \eta_1^{r-1} h_1 & \cdots & \eta_n^{r-1} h_n \end{pmatrix}=XY$

X es no singular, dado $\det(X)= g_r^r$ y $g_r \neq 0$, por lo tanto Y genera el mismo espacio que H.

QED

Nota: De ahora en adelante, se tomará como la matriz de control del código H=Y.

Ejemplo: Tómese $g(z)=z^2+z+1$, $L=GF(2^3)$, entonces $g(z)\neq 0\ \forall z\in GF(2^3)$. Una matriz de control del correspondiente código de Goppa es

$$\begin{pmatrix} \frac{1}{g(0)} & \frac{1}{g(1)} & \frac{1}{g(\alpha)} & \frac{1}{g(\alpha^2)} & \cdots & \frac{1}{g(\alpha^6)} \\ \frac{0}{g(0)} & \frac{1}{g(1)} & \frac{\alpha}{g(\alpha)} & \frac{\alpha^2}{g(\alpha^2)} & \cdots & \frac{\alpha^6}{g(\alpha^6)} \end{pmatrix} \text{ es decir}$$

$$H=\begin{pmatrix} 1 & 1 & \alpha^2 & \alpha^4 & \alpha^2 & \alpha & \alpha & \alpha^4 \\ 0 & 1 & \alpha^3 & \alpha^6 & \alpha^5 & \alpha^5 & \alpha^6 & \alpha^3 \end{pmatrix}=\begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \end{pmatrix}.$$

C. MATRIZ GENERADORA

Para los códigos de Goppa no hay fórmula explícita que dé la forma de la matriz generadora como la hay para los códigos cíclicos. La única forma de obtenerla es poniendo H en la forma $(I_{rxm} \mid M)$. Se aplica la eliminación de Gauss sobre H, luego se la diagonaliza. Dos casos son considerados:

1) la eliminación de Gauss termina, es posible diagonalizar la submatriz (rm×rm) y se obtiene la matriz generadora,

2) la eliminación de Gauss falla, las filas de H no son linealmente independientes, pero en este paso se tiene una submatriz (r'×n), r'<rm, cuyas filas forman una base del código dual (excepto permutaciones). Ésta es la matriz de control del código en el sentido estricto. Diagonalizándola, se obtiene la matriz generadora del código.

Nota: Para ambos casos, la eliminación de Gauss puede permutar algunas columnas de la matriz inicial. Así que para obtener la matriz G, se intercambian estas columnas al final del proceso.

Ejemplo: Se considera la matriz H descrita al final de la sección previa. La reducción de Gauss termina y queda:

$$\begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix}.$$

Las columnas 6 y 7 han sido permutadas durante el proceso. La diagonalización conduce a la siguiente matriz:

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix}.$$

El cómputo de la matriz $(M^T \mid I_2)$ da:

$$G'=\begin{pmatrix}1 & 1 & 1 & 1 & 0 & 0 & 1 & 0\\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1\end{pmatrix}$$

Ahora se intercambian la columnas 6 y 7, y se obtiene la siguiente matriz generadora:

$$G=\begin{pmatrix}1 & 1 & 1 & 1 & 0 & 1 & 0 & 0\\ 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1\end{pmatrix}$$

Las palabras-código del código de Goppa sobre GF(2), definido por el polinomio $g(z)=z^2+z+1$, y L=GF(2) son 00000000, 00111111, 11110100 y 11001011. La longitud del código es 8, su dimensión 2 y su distancia mínima 5.

En la siguiente sección, se verá como determinar una estimación de los parámetros del código sin generarlo.

D. PARÁMETROS DEL CÓDIGO

Proposición III.D.1. Sea $\Gamma(L,g)$ un código de Goppa de longitud n, con $L \subset GF(q^m)$ y tal que deg[g]=r. Sea k la dimensión del código y d su distancia mínima, entonces:

$$k \geq n-rm \qquad \& \qquad d \geq r+1.$$

Prueba: La matriz de control sobre GF(q) es obtenida por la sustitución de cada elemento de la matriz (r×n), con coeficientes en $GF(q^m)$, por un vector columna de longitud m. Así la matriz obtenida tiene rm filas, y su rango es a lo más rm, i.e. $\dim[\Gamma(L,g)^{\perp}] \leq rm$. Más aún $\dim[\Gamma(L,g)]=n-\dim[\Gamma(L,g)^{\perp}]$, por lo tanto $k \geq n-rm$. Nótese que

$$H=\begin{pmatrix}1 & 1 & \cdots & 1\\ \eta_1 & \eta_2 & \cdots & \eta_n\\ \vdots & \vdots & \ddots & \vdots\\ \eta_1^{r-1} & \eta_2^{r-1} & \cdots & \eta_n^{r-1}\end{pmatrix}\begin{pmatrix}h_1 & & & 0\\ & h_2 & & \\ & & \ddots & \\ 0 & & & h_n\end{pmatrix} = UV, \text{ donde } h_i=\frac{1}{g(\eta_i)}.$$

Sea c una palabra-código no cero tal que $\omega(c) \leq r$, entonces $c \in \Gamma(L,g) \Leftrightarrow H\, c^T = UVc^T = 0$. Ahora $Vc^T = (h_1c_1,...,h_nc_n)$, como $h_i \neq 0$ entonces $\omega(Vc^T) = \omega(c) \Rightarrow \omega(Vc^T) \leq r$. Sea $y = Vc^T$, se tiene que $Uy = 0$ y $\omega(y) \leq r$. Considérese la submatriz S, de U, cuyas columnas corresponden a las posiciones 1 en y. Se tiene que $\det(S) = 0$. Ahora obsérvese que cualesquiera r columnas de U forman una submatriz cuadrada K, que es una matriz de Vandermonde. Como todos los η_i son distintos, $\det(K) \neq 0$. Por lo tanto cualesquiera r columnas de U son linealmente independientes sobre GF(q). Esto conduce a una contradicción con $\det(S) = 0$. Así que $\forall c \in \Gamma(L,g)$, $\omega(c) \geq r+1$ y por lo tanto $d \geq r+1$. QED

Nota: Se puede ver aquí, la ventaja de los códigos de Goppa sobre los códigos BCH. Ciertamente, tan pronto como el polinomio de Goppa es determinado, una estimación de la distancia mínima del código es conocida. Para un código BCH, cuando el polinomio generador es conocido, se tiene que determinar el número de raíces consecutivas para tener una estimación de la distancia mínima.

E. CÓDIGOS DE GOPPA BINARIOS

Proposición III.E.1. Sea g(z) un polinomio con coeficientes en $GF(2^m)$ y sea $L \subset GF(2^m)$, la distancia mínima de un código de Goppa $\Gamma(L,g)$ verifica:

$$d \geq \deg[\varrho(z)]+1$$

con $\varrho(z)$ el polinomio de grado más pequeño que es cuadrado perfecto y divisible por g(z).

Prueba: Aquí ya se está trabajando con códigos binarios. Sea $c=(c_1,...,c_n)$ una palabra-código de peso w, esto es existen $l_1,...,l_w$ tal que $c_{l_1} = ... = c_{l_w} = 1$. Sea:

$$f_c(z) = \prod_{i=1}^{w} (z - \eta_{l_i}).$$

Sea f_c' la derivada de f_c, entonces:

$$f'_c(z) = \sum_{i=1}^{w} \prod_{\substack{j=1 \\ j \neq i}}^{w} (z - \eta_{l_j}).$$

Se tiene que:

$$R_c(z) = \sum_{i=1}^{n} \frac{c_i}{z - \eta_i} = \sum_{i=1}^{w} \frac{1}{z - \eta_{l_i}} = \frac{f_c'(z)}{f_c(z)}$$

es decir $R_c(z) f_c(z) = f_c'(z)$.

Más aún, dado que el polinomio $f_c(z)$ y $g(z)$ no tienen raíces comunes en ninguna extensión, son primos relativos, y así:

$$c \in \Gamma(L,g) \Leftrightarrow g(z) \mid R_c(z) \Leftrightarrow g(z) \mid f_c'(z).$$

Dado que q=2, el polinomio $f_c'(z)$ tiene solo potencias pares de z, y es un cuadrado perfecto.

Se muestra entonces que dado que $g(z) \mid f_c'(z) \Leftrightarrow \varrho(z) \mid f_c'(z)$ donde $\varrho(z)$ es el polinomio de grado más pequeño que es cuadrado perfecto y divisible por $g(z)$.

Si $\varrho(z) \mid f_c'(z)$ entonces, como $g(z) \mid \varrho(z)$, $g(z) \mid f_c'(z)$. Si $g(z) \mid f_c'(z)$ entonces, como $f_c'(z)$ es divisible por $g(z)$ y es cuadrado perfecto, $\varrho(z)$ divide a $f_c'(z)$. Por lo tanto:

$$g(z) \mid f_c'(z) \Leftrightarrow \varrho(z) \mid f_c'(z).$$

Conforme a la relación anterior:

$\varrho(z) \mid f_c'(z) \Leftrightarrow R_c(z) \equiv 0 [\text{mod } \varrho(z)]$. Así:

$R_c(z) \equiv 0 [\text{mod} g(z)] \Leftrightarrow R_c(z) \equiv 0 [\text{mod } \varrho(z)]$. Ahora:

$$c \in \Gamma(L,g) \Leftrightarrow c \in \Gamma(L,\varrho).$$

Por lo tanto $g(z)$ y $\varrho(z)$ generan el mismo código. Sea d_g la distancia mínima de $\Gamma(L,g)$ y d_ϱ la de $\Gamma(L,\varrho)$, entonces:

$$d_g = d_\varrho \geq \deg[\varrho(z)]+1. \text{ QED}$$

Si $g(z)$ es un polinomio irreducible sobre GF(2) o no tiene raíces múltiples en ninguna extensión de campo, entonces $\varrho(z)=g(z)^2$ y la distancia mínima de $\Gamma(L,g) \geq 2\deg[g(z)]+1$.

Así para estos polinomios particulares el poder de corrección se duplica. Esto es muy interesante para la Criptografía. Ciertamente, en Criptografía siempre se trata de trabajar en dimensiones grandes, dado que mientras más grande es un conjunto es menos fácil la búsqueda de un elemento particular. Más aún, para cifrar un mensaje, es natural agregarle ruido (o errores en este caso). La complejidad de búsqueda de la palabra "buena" crece exponencialmente con el número de errores. Así, para un código de Goppa definido por un polinomio irreducible, se puede agregar el doble de errores en comparación con el mismo código definido por algún otro polinomio y no se disminuye la dimensión del código.

Definición III.E.2. Un código de Goppa cuyo polinomio de Goppa es irreducible es llamado un ***código de Goppa irreducible***.

Definición III.E.3. Un código de Goppa binario cuyo polinomio de Goppa no tiene raíces múltiples es llamado un ***código de Goppa separable***.

Ejemplo: El código de Goppa definido en la sección D por $g(z)=z^2+z+1$ normalmente verifica que $d \geq 3$. La distancia mínima del código es 5. Ciertamente z^2+z+1 es irreducible sobre GF(2), así $d \geq 5$.

Nótese que la matriz de control (sobre la extensión del campo) luce como la de un código de Reed-Solomon generalizado. Ciertamente $\Gamma(L,g)$ es la restricción a GF(q) de

$GRS_{n-r}(\eta,v)$, donde $r=\deg[g(z)]$, $v=(v_1,\ldots,v_n)$ &

$$v_i = \frac{g(\eta_i)}{\prod_{\substack{j=1 \\ j \neq i}}^{n} (\eta_i - \eta_j)}.$$

El ***código binario de Hamming*** $H_2(r)$ es uno de los códigos lineales más populares. Puede ser descrito como un código cíclico. Sus parámetros son:

$$n=2^r-1, \qquad k=2^r-1-r, \qquad d=3.$$

Las 2^r-1 columnas de la matriz de control corresponden a todos los vectores no cero de $GF(2^r)$. Para finalizar este capítulo será descrito como un código de Goppa.

Sea $g(z)=\alpha^5 z$ donde α es un elemento primitivo de $GF(2^3)$‡. Sea $L=\{1, \alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6\} \subset GF(2^3)$ entonces $\forall z \in L$, $g(z) \neq 0$. Más aún $g(z)$ no tiene raíces múltiples. Así, se ha definido el código de Goppa $\Gamma(L,g)$ cuyos parámetros son:

$$n=7 \qquad k \geq 7-3=4 \qquad d \geq 3.$$

Una matriz de control del código es:

$$H=(\alpha^2\ \alpha\ 1\ \alpha^6\ \alpha^5\ \alpha^4\ \alpha^3)=\begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{pmatrix}.$$

Ésta es exactamente la matriz de control del código binario de Hamming de longitud 7.

‡ Significa que $\alpha^n \neq 1$ para $1 \leq n < 2^3-1$.

IV. ALGORITMOS DESCODIFICADORES

Este capítulo es un enfoque intuitivo a los algoritmos descodificadores. Se restringe a algoritmos simples. Muestra cómo se usan algoritmos para descodificar códigos BCH que también descodifican códigos de Goppa.

A. EL PROBLEMA PRINCIPAL

Sea C un código lineal binario, dado que:

- $x \in C$ es el "mensaje enviado"
- x es distorsionado a través de un canal ruidoso por un error $e \in [GF(q)]^n$
- $y=x+e$ es el "mensaje recibido" por el descodificador

El problema de descodificación es entonces encontrar x a partir de y. Así un algoritmo descodificador tiene que, empezando con un elemento de $[GF(q)]^n$, retornar una palabra-código (si esto es posible). Más aún, para un mismo elemento de $[GF(q)]^n$ la palabra retornada debe ser única, es decir que el algoritmo debe ser determinístico.

Definición IV.A.1. Sea C (n,k) un código lineal, cuya distancia mínima es d, un algoritmo descodificador es una función:

$$\gamma: [GF(q)]^n \rightarrow C \cup \{\infty\}$$

$$y \mapsto \gamma(y)$$

tal que $\forall x \in C$, $\gamma(x)=x$ & $\gamma(y)=\infty$ si y puede ser descodificado.

Definición IV.A.2. Sea e un entero positivo, γ es ***acotado*** por e, si $\forall y \in [GF(q)]^n$,

$\forall x \in C$, $d_H(x,y) < \frac{e}{2} \Rightarrow \gamma(y)=x$, además γ es ***estrictamente acotado*** por e, si $\gamma(x)=x \neq \infty \Rightarrow$

$d_H(x,y) < \frac{e}{2}$.

Definición IV.A.3. Sea γ un algoritmo descodificador, γ ***es dominado por*** la métrica d, si:

$$\forall y \in [GF(q)]^n, \forall x \in C , \gamma(y) \varepsilon C \Rightarrow d_H(\gamma(y),y) \leq d_H(x,y).$$

Todo código C acepta un algoritmo descodificador acotado por su distancia mínima, respecto a la distancia de Hamming, y que finaliza para toda entrada. Tal algoritmo es llamado de ***descodificación de máxima verosimilitud.***

B. NOTACIONES

Para hacer las cosas más fáciles, restrínjanse los códigos de Goppa tal que $0 \notin L$.

Cuando:

$0 \in L$, una leve modificación del algoritmo es necesaria. Sea $\Gamma(L,g)$ un código de Goppa de longitud $n=|L|$, corrector de t errores sobre GF(q). Sea a una palabra recibida conteniendo $0 \leq v \leq t$ errores. Estos errores ocurren en lugares ignorados $i_1,...,i_v$. La magnitud de un error es desconocida (excepto para el caso binario, donde es siempre 1). De hecho, no se conocen $i_1,...,i_v$, ni se conocen la magnitud de los errores. Ni siquiera se sabe el valor de v. A cada lugar i_j se asocia el elemento del campo $X_j=\eta_{i_j}$, de L. Sea Y_j la amplitud del error correspondiente al lugar X_j. Los X_j se conocen como ***lugares de error.*** Nótese que el número de lugares de error

de cada componente del patrón de errores, debe ser distinto porque todos los elementos de L son distintos.

Defínase el polinomio $\Lambda(z)=(1-zX_1)...(1-zX_v)$. Este polinomio es conocido como el polinomio ***localizador de errores*** y es definido como el polinomio con ceros en los lugares de error X_i^{-1} para i=1,...,v. Si este polinomio es conocido, se pueden encontrar sus ceros y con ellos los lugares de error.

C. EL SISTEMA CLAVE

Sea $\Gamma(L,g)$ un código de Goppa y sea a la palabra recibida. Compútese su síndrome $S=Ha^T$. Se tiene:

$$S_\mu = \sum_{j=1}^{n} \eta_j^{\mu-1} \gamma_j a_j, \; 1 \le \mu \le r$$

con $r=\deg[g(z)]$ y $\gamma_i=g(\eta_i)^{-1}$. Sea a' la palabra-código correspondiente con a (i.e. a sin el error e). Entonces:

$$\sum_{j=1}^{n} \eta_j^{\mu-1} \gamma_j a'_j = 0 \Rightarrow S_\mu = 0 = S_\mu = \sum_{j=1}^{n} \eta_j^{\mu-1} \gamma_j (a_j - a'_j).$$

Ahora $a_j-a_j'=0$ si $j\neq i_1,...,i_v$, o sino $a_{i_j} - a'_{i_j} = Y_j$ para j=1,...,v. Así

$$S_\mu = \sum_{j=1}^{v} \eta_{i_j}^{\mu-1} \gamma_{i_j} Y_j = \sum_{j=1}^{v} X_j^{\mu-1} \gamma_{i_j} Y_j.$$ Es decir:

$$\begin{array}{ccccccccc} S_1 & = & Y_1\gamma_{i_1} & + & Y_2\gamma_{i_2} & + & \cdots & + & Y_v\gamma_{i_v} \\ S_2 & = & X_1Y_1\gamma_{i_1} & + & X_2Y_2\gamma_{2i} & + & \cdots & + & X_vY_v\gamma_{i_v} \\ \vdots & & \vdots & & \vdots & & \ddots & & \vdots \\ S_r & = & X_1^{r-1}Y_1\gamma_{i_1} & + & X_2^{r-1}Y_2\gamma_{i_2} & + & \cdots & + & X_v^{r-1}Y_v\gamma_{i_v} \end{array}$$

. Nótese que:

$$\Lambda(z)=(1-zX_1)\ldots(1-zX_v)=\Lambda_v z^v+\Lambda_{v-1}z^{v-1}+\ldots+\Lambda_1 z+1.$$

Multiplíquense ambos lados del polinomio anterior por $X_j^{k+v-1}Y_j\gamma_{i_j}$ y sea $z=X_j^{-1}$. Se tiene:

$$0=X_j^{k+v-1}Y_j\gamma_{i_j}+\ldots+\Lambda_{v-1}X_j^{k}Y_j\gamma_{i_j}+\Lambda_v X_j^{k-1}Y_j\gamma_{i_j}.$$

Sumando estas ecuaciones desde j=1 hasta j=v

$$0=\sum_{j=1}^{v}X_j^{k+v-1}Y_j\gamma_{i_j}+\ldots+\Lambda_{v-1}\sum_{j=1}^{v}X_j^{k}Y_j\gamma_{i_j}+\Lambda_v\sum_{j=1}^{v}X_j^{k-1}Y_j\gamma_{i_j}=S_{k+v}+\ldots+\Lambda_{v-1}S_k+\Lambda_v S_k$$

Como $v \leq t$, los subíndices siempre especifican los síndromes conocidos si k está en el intervalo $1 \leq k \leq v$. Por lo tanto se tiene el conjunto de ecuaciones:

$$\Lambda_1 S_{k+v-1}+\Lambda_2 S_{k+v-1}+\ldots+\Lambda_v S_k=-S_{v+k} \text{ para } k=1,\ldots,v.$$

Éste es el conjunto de ecuaciones lineales relacionado con los síndromes de los coeficientes de $\Lambda(z)$. En forma matricial, queda:

$$\begin{pmatrix} S_1 & S_2 & S_3 & \cdots & S_{v-1} & S_v \\ S_2 & S_3 & S_4 & \cdots & S_v & S_{v+1} \\ S_3 & S_4 & S_5 & \cdots & S_{v+1} & S_{v+2} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ S_v & S_{v+1} & S_{v+2} & \cdots & S_{2v-2} & S_{2v-1} \end{pmatrix} \begin{pmatrix} \Lambda_v \\ \Lambda_{v-1} \\ \Lambda_{v-2} \\ \vdots \\ \Lambda_1 \end{pmatrix} = \begin{pmatrix} -S_{v+1} \\ -S_{v+2} \\ -S_{v+3} \\ \vdots \\ -S_{2v} \end{pmatrix}.$$

Este es el ***sistema clave*** de los códigos de Goppa. Éste es exactamente el mismo sistema que se obtiene cuando se computan los síndromes para un código BCH. Así, se puede usar el descodificador BCH para descodificar los códigos de Goppa. Nótese que cuando 0 $\in$L, entonces no se puede usar el mismo polinomio localizador de errores, dado que 0 no tiene inverso. El sistema clave puede resolverse si la matriz es no singular.

Teorema IV.C.1. Sea v el número de errores que ocurren. La matriz de síndromes

$$M = \begin{pmatrix} S_1 & S_2 & \cdots & S_\mu \\ S_2 & S_3 & \cdots & S_{\mu+1} \\ \vdots & \vdots & \ddots & \vdots \\ S_\mu & S_{\mu+1} & \cdots & S_{2\mu-1} \end{pmatrix}$$

Donde:

$S_k=\sum_{j=1}^{v} X_j^{k-1} Y_j \gamma_{i_j}$ y $M_{kl}=S_{k+l-1}$, es no singular si μ=v. La matriz es singular si μ > v.

Prueba: Sea $Y_\mu=0$ para μ > v (esto puede hacerse ya que Y_μ no está asociado ahora con una magnitud de error). Sea A la matriz de Vandermonde

$$A=\begin{pmatrix} 1 & 1 & \cdots & 1 \\ X_1 & X_2 & \cdots & X_\mu \\ \vdots & \vdots & \ddots & \vdots \\ X_1^{\mu-1} & X_2^{\mu-1} & \cdots & X_\mu^{\mu-1} \end{pmatrix}$$, y sea B la matriz diagonal

$$B=\begin{pmatrix} Y_1\gamma_{i_1} & & & 0 \\ & \ddots & & \\ & & \ddots & \\ 0 & & & Y_\mu\gamma_{i_\mu} \end{pmatrix}.$$

Entonces $(ABA^T)_{kl}=\sum_{j=1}^{\mu} (AB)_{kj} A'^{T}_{jl}$. Ahora $(AB)_{kj}=A_{kj}\times B_{jj}=X_j^{k-1}Y_j\gamma_{i_j}$. Así:

$$(ABA^T)_{kl} = \sum_{j=1}^{\mu} X_j^{k-1} Y_j \gamma_{i_j} X_j^{l-1} = \sum_{j=1}^{\mu} X_j^{k+l-2} Y_j \gamma_{i_j} = M_{kl}.$$

Por lo tanto $M=ABA^T$ y entonces $\det(M)=\det(A)\det(B)\det(A)$:

- si $\mu > v$ entonces $\det(B)=0$ dado que $Y_\mu=0$, así $\det(M)=0$
- si $\mu=v$, dado que $X_1,\ldots,X_\mu$ son todos distintos, $\det(A)\neq 0$ porque A es una matriz de Vandermonde. Más aún $\det(B)\neq 0$, por lo que $\det(M)\neq 0$. QED

El teorema provee la base para el algoritmo descodificador. Este es el descodificador de Peterson-Gorenstein-Zieler. Primero se encuentra el valor correcto de v como sigue. Como valor de prueba, se pone $\mu=t$ y se computa el determinante de M. Si es no cero, este es el valor correcto de v. De otra manera, se reduce el valor de prueba de μ en 1 y se repite. Cuando un determinante no cero se obtiene, el número actual de errores que han ocurrido es entonces conocido. Luego, se invierte M y se computa $\Lambda(z)$. Se encuentran sus ceros para inferir los lugares de error. Si el código es binario, los errores son conocidos. De otra manera computar:

$$\begin{pmatrix} Y_1 \\ \vdots \\ Y_\mu \end{pmatrix} = \begin{pmatrix} \gamma_{i_1} & \cdots & \gamma_{i_\mu} \\ \vdots & \ddots & \vdots \\ X_1^{\mu-1}\gamma_{i_1} & \cdots & X_\mu^{\mu-1}\gamma_{i_\mu} \end{pmatrix}^{-1} \begin{pmatrix} S_1 \\ \vdots \\ S_\mu \end{pmatrix}.$$

La complejidad de este algoritmo es cúbica, dado que hay dos inversas de matrices que computar. Otros dos algoritmos más poderosos existen, su complejidad es cuadrática. Estos son los algoritmos de Berlekamp-Massey y el de Euclides. La idea principal siempre es la misma: asociar cada lugar de error con un elemento del campo y computar el polinomio localizador de errores.

V. EL SISTEMA CRIPTOGRÁFICO DE MCELIECE

El criptosistema de McEliece es uno de los primeros de clave pública. Fue propuesto en 1978 y desde entonces nunca ha sido violado. Su seguridad depende de la complejidad de la descodificación de un código no estructurado. Aunque el algoritmo no es difícil de implementar, se necesita el conocimiento de la teoría de los códigos de Goppa.

El uso de los códigos de Goppa es una muy buena elección porque:

- forman una clase grande de códigos, así que es imposible hacer un lista exhaustiva de ellos
- un algoritmo polinomial de descodificación es conocido para ellos.

A. EL SISTEMA CRIPTOGRÁFICO

La idea es construir un código de Goppa y tomarlo como un código lineal general. Supóngase que el usuario U quiere recibir mensajes cifrados. Primero, U genera un código de Goppa $\Gamma(L_u,g_u)$ de parámetros (n_u,k_u,t_u) donde t_u es el poder de corrección del código. Para hacer esto U elige un polinomio irreducible, de grado t_u, sobre $GF(2^{m_u})$ (aquí $n_u=2^{m_u}$). Existe una probabilidad de $\frac{1}{t_u}$ de encontrar un polinomio irreducible, y existe un algoritmo rápido por Berlekamp para evaluar irreducibilidad. Por lo tanto es una elección razonable.

Ciertamente, el número exacto de polinomios irreducibles mónicos en GF(q)[X] ($q=2^m$) está dado por la fórmula:

$$N_q(t)=\frac{1}{t}\sum_{d|m}\mu\left(\frac{m}{d}\right)q^d,$$

donde μ es la siguiente función de Moebius,

$$\mu(n) = \begin{cases} 1 & \text{si } n = 1, \\ (-1)^k & \text{si } n \text{ es el producto de } k \text{ primos distintos}, \\ 0 & \text{si } n \text{ tiene un factor cuadrado} > 1. \end{cases}$$

Sea G_u una matriz generadora del código, mézclesela mediante la selección de una matriz $k_u \times k_u$ S_u no singular aleatoria densa y una matriz permutación P_u. Computar $\varrho_u = S_u G_u P_u$ que genera un código lineal con la misma razón de información y distancia mínima que el código generado por G_u.

ϱ_u, t_u son hechos públicos y S_u, P_u son mantenidos secretos por U. Para ***encriptar,*** supóngase que se quiere enviar un mensaje M al usuario U. Divídase M en m_i bloques de k_u bits. Para cada m_i envíese $c_i = m_i G_u + e_i$, donde e_i es un vector aleatorio de longitud n_i, peso menor o igual que t_u. Para ***desencriptar,*** U computa $c_i' = c_i P_u^{-1} = m_i S_u G_u + e_i P_u^{-1}$. Como $\omega(e_i P_u^{-1}) = \omega(e_i)$, y como $S_u G_u$ genera el mismo código que G_u, U aplica su algoritmo descodificador y encuentra $m_i S_u G_u$ del cual infiere fácilmente $m_i = m_i S_u S_u^{-1}$.

Aunque el algoritmo fue uno de los primeros algoritmos de clave pública, y no ha habido ataque criptoanalítico exitoso en su contra, nunca ha logrado una amplia aceptación por parte de la comunidad criptográfica. El esquema es de dos o tres órdenes de magnitud más rápido que el RSA[§], pero hay algunos problemas. La clave pública es enorme: 2^{19} bits de longitud. La expansión de datos es grande: el texto cifrado es el doble de largo que el original.

B. CRIPTOANÁLISIS

Es la disciplina que estudia métodos para violar criptosistemas.

---

§ Por Rivest, Shamir y Adleman, autores del criptosistema más usado en la actualidad.

1. Elemental. Primero, puede tratarse de reconstruir el código original. El valor n_u da la extensión de GF(2) donde el código fue construido. Pero el polinomio usado para generar GF(2^{m_u}) es desconocido. Luego el polinomio irreducible usado para generar el código también es desconocido. Finalmente la permutación P_u tiene que ser encontrada. Esto produce un factor de trabajo de cerca de:

$$\frac{2^{m_u}}{m_u} \times \frac{2^{m_u t_u}}{m_u t_u} \times n_u!$$

el cual para parámetros sugeridos da cerca de 2^{9267}.

Luego, podría tratarse de encontrar el error e, el cual produce un factor de trabajo de $\binom{n_u}{t_u}$ el cual es cerca de 2^{284}.

Finalmente, puede tratarse de encontrar la palabra-código cuya distancia de Hamming al texto cifrado es menor o igual que t_u. El factor de trabajo es 2^{k_u} que da 2^{512}.

2. Especializado. Aunque el problema de descodificar códigos no estructurados se cree actualmente irresoluble, existen algoritmos probabilísticos que pueden resolver este problema para algunos parámetros. Tienen un tiempo de cómputo que crece exponencialmente.

a. La idea principal. Los algoritmos propuestos tratan de encontrar una palabra de peso p en un código lineal binario aleatorio. Un algoritmo trabaja sobre la matriz generadora y el otro sobre la matriz de control.

Sea G(k,n) la matriz pública usada en el protocolo de McEliece. Sea e el poder de corrección del código C , generado por G. Todas las palabras de C tienen peso mayor o igual que 2e+1. Sea x=mG+e un texto cifrado, donde e es una palabra binaria de longitud n cuyo

peso es menor o igual que e. Considérese $G'=\begin{pmatrix} G \\ x \end{pmatrix}$. Entonces G' genera un código C' que contiene la palabra e. Más aún es la palabra de peso mínimo del código.

b. Rendimiento de los algoritmos. El estudio de estos algoritmos muestra que la búsqueda de una palabra de peso p en un código lineal binario se vuelve más compleja a medida que p está cerca de la cota de Gilbert-Varshamov. Esta cota provee una estimación teórica para el peso mínimo d de un código aleatorio (n,k), literalmente: $H_2\left(\frac{d}{n}\right) = 1 - \frac{k}{n}$ donde H_2 es la función entropía definida por: $H_2(x)=-x\log_2(x)-(1-x)\log_2(x)$. Cuando n es 2k, d es aproximadamente 0.11n y la complejidad para encontrar una palabra de peso 0.11n es cerca de $O(\Omega^k)$ donde $\Omega \cong 1.18$. Así tomando los parámetros sugeridos (n,k,t) para el protocolo estudiado se calcula un estimado del factor de trabajo, que significa que la constante bajo la notación O-grande es realmente muy grande, lo que finalmente garantiza la seguridad del esquema.

c. Ataque de Lee y Brickell. Este ataque es solamente válido para el algoritmo de McEliece y es el mejor conocido, aunque igualmente no es practicable. La idea principal es decir que el error e débilmente recupera la palabra mG. Ciertamente, para parámetros sugeridos por McEliece, la palabra e mezcla cerca del 5% de la palabra mG. Sea c=mG+e el texto cifrado, básicamente el ataque es:

- Tómense aleatoriamente k columnas de G. Llámese G_k a esta nueva matriz. Considérense las k posiciones correspondientes en c. Llámese a este nuevo vector c_k,
- Compútese $m'=c_kG_k^{-1}$. Si las k posiciones correspondientes en e son todas 0, entonces m'=m.

El factor de trabajo de este ataque es $k^3 \frac{\binom{n}{k}}{\binom{n-t}{k}}$ que para parámetros sugeridos da $2^{80.7}$.

Lee y Brickell proponen un recurso que reduce la complejidad de este ataque a 2^{69}.

VI. CONCLUSIONES

Los códigos de Goppa tienen ventaja sobre los códigos BCH, ya que al tener el polinomio de Goppa determinado, de inmediato se tiene una estimación de la distancia mínima del código, lo que no pasa para los polinomios generadores de BCH ya que aún hay que determinar el número de raíces consecutivas para determinar la distancia mínima del código.

Los códigos de Goppa pueden ser definidos con base en polinomios irreducibles, lo cual permite duplicar el número de errores que se pueden agregar a los mensajes cifrados que criptográficamente es más interesante.

Existen 3 algoritmos para descodificar las encripciones basadas en códigos de Goppa, con complejidades cúbica y cuadrática, lo cual es eficiente desde el punto de vista computacional.

El criptosistema de McEliece basado en códigos de Goppa tiene como ventajas: a) es un sistema elegante y fácil de comprender, b) su seguridad y resistencia al criptoanálisis está bien estudiada y documentada desde su descubrimiento en 1978 y c) su implementación se logra con altas velocidades de codificación / descodificación. Su principal desventaja es el tamaño de su clave pública lo que ha limitado su uso en la práctica.

VII. BIBLIOGRAFÍA

Brookshear, G. 1993. *Teoría de la computación: lenguajes formales, autómatas y complejidad.* Wilmington, Addison-Wesley Iberoamericana. 338 pp.

Fraleigh, J. 1987. *Álgebra abstracta.* Wilmington, Addison-Wesley Iberoamericana. 485 pp.

Ganley, M. 1993. *Cryptography and coding III.* Berkshire, Oxford University Press. 383 pp.

Garfinkel, S. 1995. *PGP: Pretty Good Privacy.* Sebastopol, O'Reilly & Associates, Inc. 393 pp.

Herstein, I. 1990. *Álgebra Moderna.* 2da. ed. México, D.F., Editorial Trillas, S.A. de C.V. 392 pp.

Hoffmann, D.; D. Leonard, C. Lindner, K. Phelps, C. Rodger y J. Wall. 1991. *Coding theory: the essentials.* New York, Marcel Dekker, Inc. 393 pp.

Lint, J. van y G. van der Geer. 1988. *Introduction to coding theory and algebraic geometry.* Birkhäuser, Basel. 83 pp.

Pless, V. 1982. *Introduction to the theory of error-correcting codes.* 2nd. ed. New York, John Wiley and Sons. 201 pp.

Pretzel, O. 1992. *Error-correcting codes and finite fields.* Oxford, Clarendon Press. 398 pp.

Poli, A. y Ll. Huguet. 1992. *Error correcting codes: theory and aplications.* París, Masson. 512 pp.

Robling, D. 1982. *Cryptography and data security.* Massachusetts, Addison-Wesley Publishing Company. 400 pp.

Sinkov, A. 1968. *Elementary cryptanalisis: a mathematical approach.* New York, Random House and The L.W. Singer Company. 189 pp.

Verón, P. 1994. *Cryptosystems based on error correcting codes.* Toulon et du Var, University Press. 28 pp.

Welsh, D. 1988. *Codes and cryptography.* Oxford, Clarendon Press. 257 pp.

VIII. APÉNDICE

A. COMPLEJIDAD DE ALGORITMOS

Complejidad de algoritmos La fortaleza de un criptosistema está determinada por la complejidad computacional de los algoritmos utilizados para resolverlo. La complejidad computacional de un algoritmo es medida por sus requerimientos de tiempo (T) y espacio (S), donde T y S son expresadas como funciones de n, y n caracteriza el tamaño de la entrada. Una función f(n) es expresada típicamente como de "orden de magnitud" de la forma O[g(n)] (llamada notación O-grande), donde f(n)=O[g(n)] significa que existen constantes c y n_0 tal que $f(n) \leq c|g(n)|$ para $n \geq n_0$. Como ejemplo, supóngase f(n)=17n+10. Entonces f(n)=O(n) porque $17n+10 \leq 18n$ para $n \geq 10$.

Medir los requerimientos de tiempo y espacio de un algoritmo por su rendimiento de orden de magnitud tiene la ventaja de ser independiente del sistema; así, no es necesario conocer las duraciones exactas de diferentes instrucciones o el número de bits usados para representar diferentes tipos de datos. Al mismo tiempo, nos permite ver como los requerimientos de tiempo y espacio crecen al ritmo del incremento del tamaño de la entrada. Por ejemplo, si $T=O(n^2)$, duplicando el tamaño de la entrada cuadruplica el tiempo de ejecución.

Se acostumbra clasificar los algoritmos por sus complejidades de tiempo o espacio. Un algoritmo es **polinomial** (más precisamente, de tiempo polinomial) si su tiempo de ejecución está dado por $T=O(n^t)$ para alguna constante t; es **constante** si t=0, **lineal** si t=1, **cuadrático** si t=2, y así en adelante. Es **exponencial** si $T=O(t^{h(n)})$ para t constante y h(n) polinomial.

Para n grande, la complejidad de un algoritmo puede hacer una enorme diferencia. Por ejemplo, considérese una máquina capaz de realizar una instrucción por microsegundo; esto es, 10^6 instrucciones por segundo, o 8.64×10^{10} instrucciones por día. Para un algoritmo

computacionalmente imposible en una máquina secuencial. Sin embargo, es concebible que una configuración con 1 millón de procesadores pudieran completar el cómputo en 10 días. Para $T=O(2^n)$ la ejecución del algoritmo se vuelve computacionalmente imposible aún si se tuvieran trillones de procesadores en paralelo.

Muchos criptosistemas pueden ser resueltos mediante la búsqueda exhaustiva en el espacio de claves, probando cada posible clave para saber si descifra el criptograma en un texto plano, conocido o con significado. Si $n=2^{H(K)}$ es el tamaño del espacio de claves, entonces el tiempo de ejecución de esta estrategia es $T=O(n)=O(2^{H(K)})$. Así, el tiempo es lineal en el número de claves, pero exponencial en la longitud de la clave.

Complejidad de problemas y NP-completitud La teoría de complejidad clasifica un problema de acuerdo al tiempo y espacio mínimos necesarios para resolver los casos más difíciles de un problema en una Máquina de Turing. Una Máquina de Turing determinística (TM por sus siglas en inglés) es una máquina de estado finito con una cinta infinita de lectura-escritura. Una TM es un modelo "realista" de computación en el cual problemas que son polinomialmente resolubles en una TM son también polinomialmente resolubles en sistemas reales y viceversa.

La clase **P** consiste de todos los problemas resolubles en tiempo polinomial. La clase **NP** consiste de todos los problemas resolubles en tiempo polinomial en una TM no determinística. Esto significa que si la máquina adivina la solución, puede revisar su correctitud en tiempo polinomial. Por supuesto, esto no "resuelve" el problema, porque no hay garantía de que la máquina adivine la respuesta correcta.

La clase NP contiene a la clase P porque cualquier problema polinomialmente resoluble en una TM determinística es polinomialmente resoluble es una no determinística. Si todos los problemas NP fueran polinomialmente resolubles en una TM determinística, se tendría P=NP. Aunque muchos problemas en NP parecen más "difíciles" que los problemas en P, aún nadie ha probado que $P \neq NP$.

Se ha mostrado que el problema de la satisfabilidad tiene la propiedad de que cualquier otro problema en NP puede ser reducido a él en tiempo polinomial. Esto significa que si el problema de la satisfabilidad es polinomialmente resoluble, entonces cualquier problema en NP es polinomialmente resoluble. Este conjunto de problemas equivalentes es llamado los problemas **Np-completos**, y tiene la propiedad de que si cualquiera de los problemas está en P, entonces todos los problemas NP están en P y P=NP. Así, los problemas NP-completos son los más "difíciles" en NP. Los algoritmos conocidos más rápidos para resolver sistemáticamente estos problemas tienen tiempo de solución para el peor caso de complejidad exponencial a una entrada de tamaño n.