UNIVERSIDAD AUTÓNOMA METROPOLITANA

El algoritmo de Berlekamp-Massey y decodificación de códigos BCH sobre el anillo $\mathbb{Z}_{p^s}$

Presenta
Rocío Meza Moreno

Asesor de tesis:
Dr. Horacio Tapia Recillas

Unidad Iztapalapa
Departamento de Matemáticas

México, D.F.
4 de mayo de 2007

Índice general

Agradecimientos

With a little help from my friends...
-Lennon & McCartney-

Son muchas las personas a las que debo agradecer e intentar nombrarlas a todas sería muy difícil, además, siendo una lista tan grande correría el riesgo de cometer algún olvido. Por lo tanto, decidí agradecer en general a todos aquellos que, de alguna u otra forma, son parte de este logro tan importante en mi vida. Por otro lado, no puedo (ni quiero) dejar de agradecer particular y muy especialmente a las siguientes personas.

A mi familia, por ser esa roca que me sostiene, sin ellos no estaría siquiera escribiendo estas líneas porque simplemente no sería quien soy. A mis padres, Moisés y Belén, por el amor y el apoyo incondicional que siempre me han brindado. A mis hermanos, Moisés y Aarón, por haber crecido conmigo y por haber compartido tantas cosas; por todas esas veces que estuvieron ahí para ayudarme. A mi hijo Deco por ser mi principal motivación y mi gran alegría; porque verlo crecer a diario es una experiencia maravillosa, invaluable.

A mis sinodales por haberse tomado la molestia de leer este trabajo. Al Dr. Horacio Tapia Recillas por haber sido mi guía en este proceso. A la Dra. Ma. José Arroyo Paniagua y al Dr. Francisco García Ugalde por sus comentarios y por su tiempo. Agradezco también a Conacyt por el apoyo recibido a través de una beca económica.

Finalmente, quiero agradecer a quienes estuvieron conmigo a lo largo de este trayecto y que con su ayuda hicieron posible la culminación de este trabajo. Va a sonar trillado pero debo decirlo porque es cierto, sin ustedes no lo hubiera logrado. Alejandro, Ismael, Oziel, Victor: gracias por su apoyo, por su paciencia, por sus consejos, por su ayuda, por compartir conmigo sus conocimientos y su experiencia, pero sobre todo gracias por su amistad.

Introducción

Esencialmente un código es una combinación de signos que representan algo dentro de un sistema establecido. Por ejemplo, las letras del alfabeto constituyen un código que representa sonidos por medio de símbolos gráficos. El lenguaje es un código que representa objetos e ideas por medio de palabras. La llamada clave Morse es un código que representa letras y números por medio de pulsaciones largas y cortas.

Son muchos los códigos que forman parte de la vida cotidiana aún cuando algunos de ellos pasan desapercibidos. Por ejemplo, es de todos conocido que prácticamente cualquier producto que hay en el mercado cuenta con un código de barras, que no es más que una manera de representar gráficamente y de forma que pueda ser leído por una máquina, el número de identificación de un artículo. Los libros cuentan también con un código de identificación que consiste en un número conocido como ISBN (International Standard Book Number), que identifica de manera única un libro y cuyos dígitos, además, dan información sobre el idioma en que ésta escrito y a qué editorial pertenece. El servicio postal estadounidense creó una forma de codificar números postales que representa dígitos como arreglos de barras verticales cuyo fin es ordenar y enviar el correo en forma más eficiente. Las tarjetas de crédito y los cheques cuentan también con un número que es un código de identificación. Todos estos códigos incluyen un dígito de control que es calculado a partir de los demás y cuyo objetivo es detectar cuando se está dando un número inválido, lo cual puede ocurrir, por ejemplo, debido a un error de tecleado, o bien, en el caso de los códigos de barras, por un error de lectura que puede deberse a una impresión dañada o borrosa del código correspondiente. En ese sentido, estos códigos son detectores de errores. Sin embargo, dada la importancia de las comunicaciones y de la transmisión confiable de información se hace necesaria la construcción de códigos capaces no sólo de detectar errores ocurridos en la transmisión, sino también de corregirlos. Es así que en 1948 nace la teoría de códigos con el famoso resultado de Claude Shannon que establece que dado un canal de información con cierta capacidad (dada en bits por segundo) es posible diseñar un esquema de codificación

cuya probabilidad de dar una decodificación incorrecta sea arbitrariamente pequeña. Sin embargo, la demostración de este resultado dada por Shannon no es constructiva, sino que usa un enfoque esencialmente probabilístico. El propósito de la teoría de códigos es, pues, dar métodos para la construcción de códigos detectores-correctores de errores de manera que la transmisión de información a través de canales ruidosos sea confiable.

La Figura 1 muestra el esquema general de un canal de información, que puede ser por ejemplo, una red de computadoras incluido el internet, circuitos telefónicos, cintas magnéticas, transmisión satelital, etc. El emisor envía un mensaje que consiste de una sucesión de símbolos tomados de un cierto alfabeto, sin embargo, durante la transmisión es común que el mensaje adquiera errores y el receptor no reciba el mensaje correcto. Una idea básica en teoría de códigos detectores-correctores de errores es enviar el mensaje que se quiere transmitir junto con cierta información redundante, es decir, se extiende la sucesión de símbolos del mensaje a una sucesión más grande en alguna forma que permita al receptor detectar y algunas veces corregir los errores que puedan ocurrir. Esto se hace por medio de un *esquema de codificación*, que consta de un conjunto de palabras llamado *código* y una función que asigna una palabra del código a cada mensaje. Se envía entonces la palabra codificada a través del canal. Una vez que se recibe la información, se intenta detectar y corregir los errores que pudieron ocurrir en la transmisión, para después recobrar el mensaje original a partir de la palabra del código corregida. A este proceso se le llama *decodificación*. Finalmente, el receptor obtiene un estimado del mensaje enviado.

Un problema importante es la construcción de códigos que corrijan la mayor cantidad de errores posible minimizando la cantidad de información redundante y que al mismo tiempo reduzcan la probabilidad de obtener una decodificación incorrecta.

Debido a la propia naturaleza de algunos canales de información los códigos inicialmente fueron construidos a partir del alfabeto binario, sin embargo, son también de gran importancia y han sido ampliamente estudiados diversos tipos de códigos cuyo alfabeto es un campo finito. Además, en los últimos años han tomado importancia los códigos sobre anillos finitos entre los cuales se encuentran $\mathbb{Z}_{p^s}$, $\mathbb{Z}_m$, anillos de Galois, anillos de cadena, etc. Varios grupos de investigadores se han dedicado al estudio de estos códigos debido a que tienen importantes

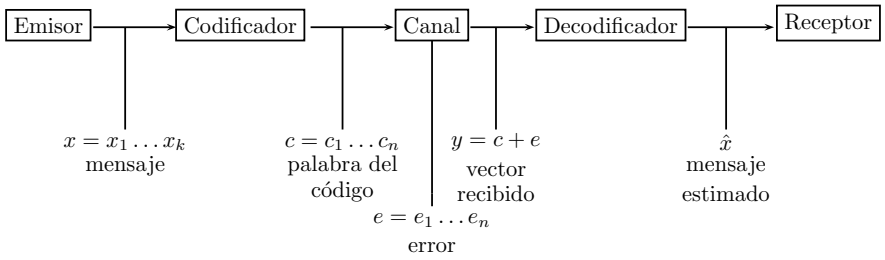

FIGURA 1. Estructura de un canal de información

aplicaciones, por ejemplo, en la generación de sucesiones utilizadas en telefonía celular.

El primer capítulo de este trabajo está dedicado a dar una breve introducción a los códigos lineales sobre un campo finito con especial énfasis en la familia de códigos llamados cíclicos los cuales tienen una amplia gama de aplicaciones debido a que pueden implementarse de manera muy eficiente. Entre los códigos cíclicos se encuentran los códigos BCH y de Reed-Solomon, estos últimos son usados en la corrección de errores de lectura en los discos compactos y DVD.

En el segundo capítulo, se introduce un importante algoritmo que permite decodificar en forma muy eficiente códigos BCH sobre campos finitos conocido como el algoritmo de Berlekamp-Massey, el cual, permitió la aplicación por primera vez de los códigos de Reed-Solomon.

Finalmente, el tercer capítulo extiende las ideas de los dos primeros a códigos cuyo alfabeto es el anillo $\mathbb{Z}_{p^s}$. El objetivo es dar un algoritmo para decodificar códigos BCH sobre este anillo. Para ello se describen brevemente los anillos de Galois y algunas de sus propiedades. Después se presentan los códigos cíclicos, más particularmente los BCH, sobre $\mathbb{Z}_{p^s}$ usando la construcción presentada por P. Shankar en [**22**]. Esta construcción conserva las ideas de la definición de códigos BCH sobre campos finitos y por tanto permite dar un algoritmo de decodificación con el mismo enfoque. El algoritmo presentado se basa principalmente en una generalización del algoritmo de Berlekamp-Massey sobre anillos de Galois que utiliza las ideas del algoritmo dado por J.A. Reeds y N.J.A. Sloane en [**19**] que originalmente fué pensado para anillos $\mathbb{Z}_m$.

Todos los algoritmos que se incluyen en el presente trabajo son ilustrados con ejemplos en los cuales, para llevar a cabo los cálculos correspondientes, se hizo uso del paquete computacional Mathematica 5.1.

Este trabajo supone algunos conceptos básicos de distintas áreas, principalmente campos finitos y álgebra conmutativa que pueden consultarse por ejemplo en [**13**] y en [**1**] respectivamente.

Capítulo 1

Códigos cíclicos sobre campos finitos

1.1. Introducción a los códigos lineales

En términos sencillos, un código es un conjunto de cadenas de elementos en un alfabeto dado. En particular nos interesan los códigos con entradas en algún campo finito. Más formalmente, consideremos el espacio vectorial de todas las n-adas sobre un campo finito $\mathbb{F}_q$, denotado por $\mathbb{F}_q^n$. Diremos que un código de longitud n sobre $\mathbb{F}_q$ es cualquier subconjunto de $\mathbb{F}_q^n$ y a los vectores del código los llamaremos *palabras.*

Con el fin de obtener una estructura algebraica más rica suele imponerse la condición de linealidad, lo cual lleva a la siguiente definición.

Definición 1.1.1. *Un código lineal $\mathcal{C}$ de longitud n sobre $\mathbb{F}_q$ es un subespacio vectorial de $\mathbb{F}_q^n$. Si $\mathcal{C}$ tiene dimensión k decimos que es un $[n,k]_q$-código lineal.*

En vista de que un código lineal $\mathcal{C}$ es un espacio vectorial, puede darse en términos de una matriz cuyos renglones forman una base para $\mathcal{C}$.

Definición 1.1.2. *Sea $\mathcal{C}$ un $[n,k]$-código lineal sobre $\mathbb{F}_q$. Decimos que G es una matriz generadora del código si es una matriz de $k \times n$ cuyos renglones forman una base para $\mathcal{C}$.*

Si $\mathcal{C}$ es un $[n,k]_q$-código con matriz generadora G entonces sus palabras son las posibles q^k combinaciones lineales de los renglones de G, de modo que el código está dado por

$$\mathcal{C} = \{\overline{x}G \mid \overline{x} \in \mathbb{F}_q^k\}$$

O bien, pensando a la matriz G como una transformación lineal de $\mathbb{F}_q^k$ en $\mathbb{F}_q^n$,

$$\mathcal{C} = \mathrm{Im}(G)$$

Un código lineal puede tener muchas matrices generadoras. Para cada conjunto de k columnas linealmente independientes de la matriz

generadora, las correspondientes coordenadas forman lo que se llama un conjunto de *información* para $\mathcal{C}$. Las restantes $n-k$ coordenadas forman el llamado conjunto de *redundancia* de $\mathcal{C}$. Si las primeras k coordenadas forman un conjunto de información, entonces el código correspondiente tiene una única matriz generadora de la forma $[I_k|A]$ donde I_k es la matriz identidad de $k \times k$. Una matriz generadora tal se dice que está en forma *estándar*. Cuando un código lineal se dá por medio de una matriz generadora en su forma estándar se simplifican los procesos de codificación y decodificación pues en este caso el mensaje original aparece en las k primeras coordenadas de la palabra codificada. Dada una matriz generadora G, puede obtenerse una matriz generadora en forma estándar por medio de operaciones elementales.

Sea $\mathcal{C}$ un $[n,k]_q$-código lineal con matriz generadora $G = [I_k|A]$ y considérese la matriz $H = [-A^t|I_{n-k}]$ donde A^t es la transpuesta de A. Entonces,

$$HG^t = [-A^t|I_{n-k}] \begin{bmatrix} I_k \\ A^t \end{bmatrix} = -A^t + A^t = 0$$

y como H es de rango $n-k$ se tiene que

$$\overline{c} \in \mathcal{C} \quad \text{si y sólo si} \quad H\overline{c}^t = 0$$

o equivalentemente,

$$\mathcal{C} = \{\overline{c} \in \mathbb{F}_q^n \mid H\overline{c}^t = 0\} \tag{1}$$

es decir, pensando a H como transformación lineal

$$\mathcal{C} = \ker(H)$$

Una matriz H que satisface (1) se conoce como *matriz verificadora de paridad* del código $\mathcal{C}$.

Para un $[n,k]_q$-código $\mathcal{C}$ una matriz generadora es simplemente una matriz cuyos renglones son linealmente independientes y generan el código. Los renglones de una matriz verificadora de paridad H son también independientes y por tanto H es una matriz generadora de algún código, el cual, se conoce como código *dual* u *ortogonal* de $\mathcal{C}$ y es denotado por $\mathcal{C}^\perp$. Obsérvese que $\mathcal{C}^\perp$ es un $[n,n-k]$-código. Puede definirse el código dual por medio del producto interior pues de (1) se tiene que

$$\mathcal{C}^\perp = \{\overline{x} \in \mathbb{F}_q^n \mid \overline{x} \cdot \overline{c} = 0 \quad \forall \overline{c} \in \mathcal{C}\}$$

Un concepto importante de un código lineal y crucial para determinar su capacidad de corregir errores es la llamada *distancia mínima*: entre más grande es esta distancia, más errores se pueden corregir (ver Teorema 1.1.7). Para definirla, damos un par de conceptos previos.

Definición 1.1.3. *El peso de Hamming, $p(\overline{x})$, del vector $\overline{x} \in \mathbb{F}_q^n$ es el número de sus coordenadas distintas de cero.*

Definición 1.1.4. *La distancia de Hamming, $d(\overline{x}, \overline{y})$, entre dos vectores $\overline{x}, \overline{y} \in \mathbb{F}_q^n$ se define como el número de coordenadas en que difieren. Es decir, $d(\overline{x}, \overline{y}) = p(\overline{x} - \overline{y})$*

Es fácil ver que la distancia de Hamming determina una métrica en el espacio vectorial $\mathbb{F}_q^n$ ([**9**], pág. 8). Ésta no es la única forma de metrizar dicho espacio, sin embargo a lo largo del presente trabajo al hablar de distancia nos estaremos refiriendo a la distancia de Hamming.

Ahora sí definimos la distancia mínima de un código.

Definición 1.1.5. *La distancia mínima d de un código $\mathcal{C}$ es la menor distancia entre dos palabras distintas. Esto es,*

$$d = \text{mín}\{d(\overline{x}, \overline{y}) \mid \overline{x} \neq \overline{y},\ \overline{x}, \overline{y} \in \mathcal{C}\}$$

Como $d(\overline{x}, \overline{y}) = p(\overline{x} - \overline{y})$, la distancia mínima, d, de un código lineal coincide con el peso mínimo de las palabras no cero del código. Esto significa que para encontrar la distancia mínima de un código lineal, no es necesario comparar todas las parejas de palabras posibles, basta determinar el peso de cada palabra distinta de cero.

Cuando se conoce la distancia mínima d de un $[n, k]_q$-código lineal, nos referimos a él como un $[n, k, d]_q$ código. La distancia mínima de un código lineal puede obtenerse a partir de la matriz verificadora de paridad del código como lo muestra el siguiente resultado.

Teorema 1.1.6. *Sea $\mathcal{C}$ un $[n, k]_q$ código lineal con matriz verificadora de paridad H. Entonces $\mathcal{C}$ tiene distancia mínima d si y sólo si H tiene d columnas linealmente dependientes pero cualesquier $d - 1$ columnas son linealmente independientes.*

Demostración. Una palabra $\overline{c} \in \mathcal{C}$ tiene peso p si y sólo si $H\overline{c}^t = 0$ con $\overline{c}$ un vector de peso p y esto es posible si y sólo si existen p columnas de H que son linealmente dependientes. Por tanto, para que el código tenga una palabra de peso d deben existir d columnas de H linealmente dependientes, además d será el peso mínimo si no existe un número menor de columnas linealmente dependientes. □

El siguiente resultado es bien conocido, y dá la relación entre la distancia mínima y la capacidad correctora de un código. Para mayores detalles el lector puede consultar [**9**] y [**14**].

Teorema 1.1.7. *Un código lineal sobre el campo $\mathbb{F}_q$ con distancia mínima d es capaz de corregir hasta $\lfloor \frac{d-1}{2} \rfloor$ errores, donde $\lfloor x \rfloor$ es el mayor entero menor o igual que x.*

Daremos ahora una cota superior para la distancia mínima que en general no es muy buena pero lleva a la definición de la importante clase de códigos conocidos como MDS entre los cuales se encuentran los códigos de *Reed-Solomon* que veremos más adelante.

Teorema 1.1.8 (Cota Singleton)**.** *Si $\mathcal{C}$ es un $[n, k, d]_q$-código lineal, entonces $d \leq n - k + 1$.*

Demostración. Una palabra con sólo un símbolo de información distinto de cero tiene peso a lo más $n-k+1$, pues tendrá a lo más $n-k$ dígitos de redundancia distintos de cero más el dígito de información. □

Un código cuya distancia mínima satisface el teorema anterior con igualdad se llama *Distancia Máxima Separable* (Maximum Distance Separable, MDS) y satisface que ningún otro código de longitud n y distancia mínima d tiene más palabras que un código MDS con estos mismos parámetros.

1.1.1. Códigos de Hamming

Los códigos de Hamming son probablemente los más conocidos entre los códigos lineales detectores correctores de errores. Fueron descubiertos en forma independiente por Marcel Golay en 1949 y por Richard Hamming en 1950. Estos códigos corrigen un error y son de fácil codificación y decodificación. Para más información sobre estos códigos pueden consultarse [**9**], [**14**], y [**21**].

De acuerdo al Teorema 1.1.6 la distancia mínima de un $[n, k]_q$ código lineal con matriz verificadora de paridad H es el menor entero d para el cual existen d columnas linealmente dependientes en H. Por tanto, la matriz verificadora de paridad de un $[n, k, 3]_q$ lineal código (cuya capacidad correctora es de un error) tiene la propiedad de que ningún par de sus columnas es linealmente dependiente pero existen tres columnas que sí lo son. En el caso de códigos binarios ($q = 2$) esto se logra tomando las columnas de H distintas y no nulas.

Si el número de renglones de la matriz verificadora es r se tienen $2^r - 1$ posibles columnas que satisfacen la condición anterior, a saber los $2^r - 1$ vectores binarios no cero de longitud r (que corresponden a la representación binaria de los primeros $2^r - 1$ enteros). Los códigos binarios de Hamming usan todas estas posibles columnas obteniendo un código de longitud $n = 2^r - 1$.

Definición 1.1.9. *Un código binario de Hamming, $\mathcal{H}_r$, de longitud $n = 2^r - 1$ ($r \geq 2$) es un código cuya matriz verificadora de paridad H tiene por columnas a todos los vectores binarios no cero de longitud r, cada uno apareciendo una sola vez.*

Por construcción H es una matriz de rango r y por tanto la dimensión del código $\mathcal{H}_r$ es $k = n - r = 2^r - r - 1$. Además, de la discusión previa a la definición se sigue que $\mathcal{H}_r$ tiene distancia mínima $d = 3$. Por tanto un código binario de Hamming $\mathcal{H}_r$ es un $[2^r - 1, 2^r - r - 1, 3]_2$-código lineal y tiene capacidad correctora de un solo error (ver Teorema 1.1.7).

Obsérvese que la elección de la matriz verificadora de un código de Hamming no es única de modo que hay varios códigos de Hamming con parámetros dados. Sin embargo, cualquier matriz verificadora de un código binario de Hamming puede obtenerse de cualquier otra con los mismos parámetros simplemente permutando sus columnas. En este sentido, los códigos de Hamming $[2^r - 1, 2^r - r - 1, 3]_2$ son equivalentes.

Consideremos por ejemplo el código $\mathcal{H}_3$, conocido como el $[7, 4, 3]_2$-código de Hamming, dado por la matriz verificadora cuyas columnas pensadas como representaciones binarias de los enteros entre 1 y 7 aparecen en orden ascendente. Esto es,

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Para el código $\mathcal{H}_3$ con esta matriz verificadora la decodificación es muy sencilla: supóngase que se envía la palabra $\overline{c} \in \mathcal{H}_3$ pero que en la transmisión ocurre un solo error recibiéndose el vector $\overline{r} = \overline{c} + \overline{e}$, donde $\overline{e}$ es un vector con sólo una entrada distinta de cero, digamos en la i-ésima posición. Entonces,

$$H\overline{r}^t = H\overline{c}^t + H\overline{e}^t = H\overline{e}^t$$

pues $\overline{c}$ está en el código. Pero $H\overline{e}^t$ es la i-ésima columna de H escrita como renglón, el cual visto como dígito binario da la representación de la posición en que ocurrió el error.

Por ejemplo, digamos que el error es $\overline{e} = (0,0,0,1,0,0,0)$, entonces

$$H\overline{e}^t = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = (1,0,0)$$

y 100 es la representación binaria del 4.

Los códigos de Hamming pueden definirse en cualquier campo finito $\mathbb{F}_q$. En este caso la matriz verificadora de paridad H es una matriz de $m \times (q^r - 1)/(q-1)$ cuyas columnas son linealmente independientes por parejas. Una matriz con esta característica define un $[(q^r-1)/(q-1), (q^r-1)/(q-1) - r, 3]_q$ código. Aquí la equivalencia de los códigos con los mismos parámetros es en el sentido de que la matriz verificadora de un código puede obtenerse de cualquier otra (con los mismos parámetros) permutando las columnas y multiplicando algunas de ellas por escalares no cero.

1.2. Códigos cíclicos

Los códigos cíclicos son un tipo especial de códigos lineales que han sido muy estudiados debido a su eficiente y fácil implementación, lo cual, lleva a importantes aplicaciones prácticas. En esta sección se presentan este tipo de códigos y algunas de sus propiedades más importantes. Entre los códigos cíclicos se encuentra la importante familia de códigos BCH y los códigos de Reed-Solomon.

Definición 1.2.1. *Un código lineal $\mathcal{C}$ de longitud n sobre $\mathbb{F}_q$ es llamado cíclico si dado $(c_0, c_1, \ldots, c_{n-1}) \in \mathcal{C}$ también su corrimiento $(c_{n-1}, c_0, \ldots, c_{n-2})$ está en $\mathcal{C}$.*

Ejemplo 1.2.2. El código lineal $C = \{0000, 0101, 1010, 1111\}$ es cíclico.

Ejemplo 1.2.3. Sea $\mathcal{H}_3$ el código binario de Hamming $[7,4,3]_2$ con matriz verificadora de paridad

$$H = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

Entonces un vector $\overline{c} = (c_1, c_2, c_3, c_4, c_5, c_6, c_7)$ es una palabra del código si y sólo si satisface la relación

$$H\overline{c}^t = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix} \overline{c}^t = 0$$

es decir,

$$\begin{aligned} c_1 &= c_4 + c_6 + c_7 \\ c_2 &= c_4 + c_5 + c_6 \\ c_3 &= c_5 + c_6 + c_7 \end{aligned}$$

Sustituyendo los posibles valores para c_4, c_5, c_6 y c_7 podemos obtener todas las palabras del código que son,

$$\begin{aligned} \mathcal{H}_3 = \{&0000000, 1010001, 1110010, 0100011, 0110100, 1100101, \\ &1000110, 0010111, 1101000, 0111001, 0011010, 1001011, \\ &1011100, 0001101, 0101110, 1111111\} \end{aligned}$$

Obsérvese que para cada palabra de $\mathcal{H}_3$, su corrimiento cíclico también es parte del código y por lo tanto $\mathcal{H}_3$ es cíclico.

Sea $\mathbb{F}_q$ un campo con $q = p^r$ elementos (p primo y r entero positivo) y sea $\mathcal{R}_n = \mathbb{F}_q[x]/\langle x^n - 1\rangle$ el anillo cociente de $\mathbb{F}_q[x]$ módulo el ideal $\langle x^n - 1\rangle$ generado por el polinomio $x^n - 1$. Las operaciones en $\mathcal{R}_n$ son la suma usual de polinomios y la multiplicación usual seguida de una reducción módulo $x^n - 1$ y sus elementos pueden representarse como polinomios de grado menor que n. La función,

$$\begin{aligned} \varphi : \mathbb{F}_q^n &\longrightarrow \mathcal{R}_n \\ (a_0, a_1, \ldots, a_{n-1}) &\longrightarrow a_0 + a_1 x + \cdots + a_{n-1}x^{n-1} \end{aligned}$$

es un isomorfismo de espacios vectoriales sobre $\mathbb{F}_q$ por lo que podemos pensar a los elementos de $\mathcal{R}_n$ como polinomios y también como vectores. Lo cual se hará indistintamente a lo largo de este trabajo.

Ahora, si multiplicamos $c(x) = c_0 + c_1 x + \cdots + c_{n-1}x^{n-1}$ por x en $\mathcal{R}_n$ obtenemos

$$\begin{aligned} x \cdot c(x) &= c_0 x + c_1 x^2 + \cdots + c_{n-1}x^n \\ &= c_{n-1} + c_0 x + \cdots + c_{n-2}x^{n-1} \end{aligned}$$

pues en $\mathcal{R}_n$, $x^n = 1$. Pero este polinomio corresponde al vector $(c_{n-1}, c_0, \ldots, c_{n-2})$ así que la multiplicación por x corresponde al corrimiento cíclico.

De lo anterior se desprende que podemos pensar a un código cíclico como un ideal del anillo $\mathcal{R}_n$. Más formalmente se tiene la siguiente,

Proposición 1.2.4. *Sea φ el isomorfismo definido antes. Entonces un código lineal $\mathcal{C} \subseteq \mathbb{F}_q^n$ es cíclico si y sólo si $\varphi(\mathcal{C})$ es un ideal de $\mathcal{R}_n$.*

Demostración. Sea $\mathcal{C}$ un código cíclico sobre $\mathbb{F}_q^n$ y $c_0 + c_1 x + \cdots + c_{n-1}x^{n-1} \in \varphi(\mathcal{C})$. Entonces, el correspondiente vector $\overline{c} = (c_0, c_1, \ldots, c_{n-1})$ está en el código y por tanto su corrimiento cíclico $(c_{n-1}, c_0, \ldots, c_{n-2})$ también está en $\mathcal{C}$ de modo que $x\varphi(\overline{c}) = c_{n-1} + c_0 x + \cdots + c_{n-2}x^{n-1} \in \varphi(\mathcal{C})$. De lo anterior se sigue que $\varphi(\mathcal{C})$ es cerrado bajo multiplicaciones por elementos de $\mathcal{R}_n$, además $\varphi(\mathcal{C})$ es un subgrupo aditivo de $\mathcal{R}_n$ y por lo tanto es un ideal de $\mathcal{R}_n$. Recíprocamente, si $\varphi(\mathcal{C})$ es un ideal de $\mathcal{R}_n$, dada una palabra $\overline{c} = (c_0, c_1, \ldots, c_{n-1}) \in \mathcal{C}$ se tiene que $x\varphi(\overline{c}) = c_{n-1} + c_0 x + \cdots + c_{n-2}x^{n-1} \in \varphi(\mathcal{C})$ y por tanto el corrimiento cíclico de $\overline{c}$ también está en $\mathcal{C}$. □

A continuación veremos algunas propiedades importantes de los códigos cíclicos sobre $\mathbb{F}_q$, para ello se usará el siguiente resultado para anillos conmutativos.

Lema 1.2.5. *Sea R un anillo conmutativo con unidad y sea I un ideal de R. Entonces los ideales de R/I son de la forma J/I con J un ideal de R tal que $J \supseteq I$.*

Teorema 1.2.6. *Sea $\mathcal{C}$ un código cíclico de longitud n sobre $\mathbb{F}_q$. Entonces,*

1. *Existe un único polinomio mónico $g(x)$ de grado mínimo en $\mathcal{C}$. Este polinomio es un generador de $\mathcal{C}$, es decir, $\mathcal{C} = \langle g(x) \rangle$ y es conocido como el polinomio generador del código $\mathcal{C}$.*
2. *$g(x)$ divide a $x^n - 1$ en $\mathbb{F}_q[x]$.*
3. *Sea r el grado de $g(x)$. Todo $c(x) \in \mathcal{C}$ puede escribirse en forma única como $c(x) = f(x)g(x)$ en $\mathbb{F}_q[x]$, donde $f(x) \in \mathbb{F}_q[x]$ es de grado menor que $n - r$. La dimensión de $\mathcal{C}$ es $n - r$. Esto significa que el mensaje $f(x)$ se codifica como $f(x)g(x)$.*
4. *Si $g(x) = g_0 + g_1 x + \cdots + g_r x^r$ entonces*

$$G = \begin{bmatrix} g_0 & g_1 & g_2 & \cdots & g_r & 0 & \cdots & 0 \\ 0 & g_0 & g_1 & \cdots & g_{r-1} & g_r & \cdots & 0 \\ \vdots & \ddots & & & & & \ddots & \vdots \\ 0 & & g_0 & \cdots & & & \cdots & g_r \end{bmatrix}$$

es una matriz generadora para $\mathcal{C}$.

DEMOSTRACIÓN. Si $\mathcal{C}$ es un ideal de $\mathcal{R}_n = \mathbb{F}_q[x]/\langle x^n - 1\rangle$, del Lema 1.2.5 existe un ideal J de $\mathbb{F}_q[x]$ tal que $J \supseteq \langle x^n - 1\rangle$ y

$$\mathcal{C} = J/\langle x^n - 1\rangle.$$

Es sabido que $\mathbb{F}_q[x]$ es un anillo de ideales principales y que un generador de J es el único polinomio mónico de grado mínimo en J ([**6**]). Sea $g(x)$ dicho polinomio. Entonces

$$J = \langle g(x)\rangle \quad \text{en } \mathbb{F}_q[x].$$

Es claro que $g(x)$, pensado como elemento de $\mathcal{R}_n$, es también un generador del ideal $\mathcal{C} = J/\langle x^n - 1\rangle$. Además, como $J \supseteq \langle x^n - 1\rangle$ entonces $x^n - 1 \in J = \langle g(x)\rangle$ y por tanto

$$g(x) \mid x^n - 1 \quad \text{en } \mathbb{F}_q[x].$$

Esto prueba (1) y (2). Ahora, como $g(x)$ genera el ideal $\mathcal{C}$, todo $c(x) \in \mathcal{C}, \text{gr}(c(x)) < n$, puede escribirse como $h(x)g(x)$ en $\mathcal{R}_n$. Luego, pensando a $c(x)$ como elemento de $\mathbb{F}_q[x]$ tenemos

$$\begin{aligned} c(x) &= h(x)g(x) + k(x)(x^n - 1) \\ &= h(x)g(x) + k(x)q(x)g(x) \\ &= f(x)g(x) \end{aligned}$$

donde $f(x) = h(x) + k(x)q(x)$ y $\text{gr}(f(x)) < n - r$. Ésto significa que los elementos del código son múltiplos de $g(x)$ por polinomios de grado menor que $n - r$ con las operaciones de $\mathbb{F}_q[x]$. Ahora, $g(x)$, $xg(x)$,..., $x^{n-r-1}g(x)$ son $n - r$ polinomios linealmente independientes en $\mathbb{F}_q[x]$ que generan al código y como su grado no sobrepasa $n-1$ siguen siendo linealmente independientes en $\mathcal{R}_n$, así que la dimensión de $\mathcal{C}$ es $n - r$ y queda probado (3). Finalmente, los vectores correspondientes a los polinomios $g(x), xg(x), \ldots, x^{n-r-1}g(x)$ son los renglones de la matriz generadora. □

Sea $g(x)$ el polinomio generador de un código cíclico $\mathcal{C}$. Como $g(x)$ divide a $x^n - 1$ entonces

$$h(x) = \frac{x^n - 1}{g(x)}$$

es un polinomio, digamos $h(x) = \sum_{i=0}^{k} h_i x^i$ de grado k, llamado *polinomio verificador* de $\mathcal{C}$. La razón de este nombre es la siguiente. Si $c(x) = \sum_{i=0}^{n-1} c_i x^i = f(x)g(x)$ es cualquier palabra del código $\mathcal{C}$ entonces, en $\mathcal{R}_n$,

$$\begin{aligned} c(x)h(x) &= (\sum_{i=0}^{n-1} c_i x^i)(\sum_{i=0}^{k} h_i x^i) \\ &= f(x)g(x)h(x) \\ &= f(x)(x^n - 1) \\ &= 0 \end{aligned}$$

En particular, deben anularse en este producto los coeficientes correspondientes a x^k, x^{k+1},...,x^{n-1}, esto es

$$\begin{aligned} c_0 h_k + c_1 h_{k-1} + \cdots + c_k h_0 &= 0 \\ c_1 h_k + c_2 h_{k-1} + \cdots + c_{k+1} h_0 &= 0 \\ &\vdots \\ c_{n-k-1} h_k + c_{n-k} h_{k-1} + \cdots + c_{n-1} h_0 &= 0 \end{aligned}$$

Considérese la siguiente matriz de tamaño $(n-k) \times (n)$

$$H = \begin{bmatrix} h_k & h_{k-1} & \cdots & h_0 & 0 & \cdots & 0 \\ 0 & h_k & \cdots & h_1 & h_0 & \cdots & 0 \\ \vdots & \vdots & \ddots & & & & \\ 0 & 0 & \cdots & h_k & h_{k-1} & \cdots & h_0 \end{bmatrix} \tag{2}$$

Las ecuaciones anteriores dicen que si $c(x)$ está en el código $\mathcal{C}$ entonces el vector correspondiente $\overline{c}$ satisface que $H\overline{c}^t = 0$. Por otro lado, como $k = \mathrm{gr}(h(x)) = n - \mathrm{gr}(g(x)) = \dim \mathcal{C}$ y los renglones de H son linealmente independientes, la condición $H\overline{c}^t = 0$ es suficiente para que $c(x) \in \mathcal{C}$. Por lo tanto, H es una matriz verificadora de paridad para el código $\mathcal{C}$.

1.2.1. Factorización de $x^n - 1$

Como el polinomio generador de un código cíclico de longitud n sobre $\mathbb{F}_q$ es un divisor de $x^n - 1$ estamos interesados en encontrar los factores irreducibles de este polinomio sobre $\mathbb{F}_q$.

Ya que en $\mathbb{F}_q[x]$, $x^{mq} - 1 = (x^m - 1)^q$, basta considerar el caso en que $(n, q) = 1$.

Definición 1.2.7. *Al menor entero positivo m tal que n divide a $q^m - 1$ se le llama el orden de q módulo n y se denota por $o_n(q)$, es decir, $q^{o_n(q)} \equiv 1 \mod n$.*

Sea $\mathbb{Z}_n$ el anillo de enteros módulo n. Obsérvese que $o_n(q)$ es el orden multiplicativo de q en este anillo.

Si $m = o_n(q)$, entonces $x^n - 1$ divide a $x^{q^m-1} - 1$ pero no divide a $x^{q^i-1} - 1$ para $0 < i < m$. Por otro lado, $\mathbb{F}_{q^m}$ es el campo de descomposición de $x^{q^m} - x = x(x^{q^m-1} - 1)$, es decir, es el menor campo que contiene a todas sus raíces, por tanto $\mathbb{F}_{q^m}$ es también el menor campo que contiene a las raíces de $x^n - 1$, llamadas raíces n-ésimas de la unidad. Ahora, como n y q son primos relativos, $x^n - 1$ y su derivada nx^{n-1} son también primos relativos, lo cual implica que $x^n - 1$ tiene n raíces distintas en $\mathbb{F}_{q^m}$.

Sea $G = \langle g \rangle$ el grupo multiplicativo generado por q en $\mathbb{Z}_n$ y considérese la acción de G sobre $\mathbb{Z}_n$ dada por:

$$\begin{aligned} * : G \times \mathbb{Z}_n &\to \mathbb{Z}_n \\ (q^i, k) &\to q^i * k = q^i k \end{aligned}$$

Para $0 \leq i < n$ denotaremos por C_i a la órbita de i bajo la acción de G. Es decir, $\mathcal{C}_i = \{i, iq, iq^2, \ldots, iq^{m_i-1}\}$ donde m_i es el menor entero positivo tal que $iq^{m_i} \equiv i \mod n$. A C_i se le conoce como la *clase ciclotómica para q módulo n que contiene al entero i.*

Por consiguiente se tiene una partición $\{C_i\}$ de $\mathbb{Z}_n$ dada por las clases ciclotómicas. En particular, $\mathbb{Z}_n = \bigcup_i C_i$ donde i corre sobre un conjunto de representantes de las clases ciclotómicas módulo n. Por ejemplo, para $n = 9$ y $q = 2$ las distintas clases ciclotómicas son:

$$\begin{aligned} C_0 &= \{0\} \\ C_1 &= \{1, 2, 4, 8, 7, 5\} \\ C_3 &= \{3, 6\} \end{aligned}$$

En general, la cardinalidad de cada órbita bajo la acción de un grupo G es un divisor de la cardinalidad de éste. Por tanto, la cardinalidad de cada clase ciclotómica C_i divide a $m = o_n(q)$.

Ahora, $\mathbb{F}_{q^m}^*$ es un grupo cíclico de orden $q^m - 1$ y como n divide a $q^m - 1$ entonces $\mathbb{F}_{q^m}^*$ tiene un subgrupo cíclico G de orden n formado por las raíces n-ésimas de la unidad. Sea α un generador de este subgrupo, $G = \langle \alpha \rangle$. A α se le llama raíz n-ésima primitiva de la unidad. Recuérdese que el polinomio mínimo (o irreducible) sobre $\mathbb{F}_q$ de un elemento $\theta \in \mathbb{F}_{q^m}$ es el polinomio mónico de menor grado en $\mathbb{F}_q[x]$ que tiene a θ como raíz. Es sabido que el polinomio mínimo de α^i sobre $\mathbb{F}_q$ está dado por:

$$\begin{aligned} irr(\alpha^i, \mathbb{F}_q) &= [x - \alpha^i][x - (\alpha^i)^q][x - (\alpha^i)^{q^2}] \cdots [x - (\alpha^i)^{q^{m_i - 1}}] \\ &= (x - \alpha^i)(x - \alpha^{iq})(x - \alpha^{iq^2}) \cdots (x - \alpha^{iq^{m_i - 1}}) \\ &= \prod_{j \in C_i} (x - \alpha^j) \end{aligned} \tag{3}$$

donde m_i es el menor entero positivo tal que $iq^{m_i} \equiv i \mod n$. Para mayores detalles sobre el polinomio mínimo el lector puede consultar [**21**], capítulo 7 pp. 297-300 y [**13**], capítulo 3 pp. 51-54 y 96. De este modo podemos escribir a $x^n - 1$ como producto de factores irreducibles sobre $\mathbb{F}_q$ como sigue:

$$x^n - 1 = \prod_i irr(\alpha^i, \mathbb{F}_q) = \prod_{j=0}^{n-1} (x - \alpha^j) \tag{4}$$

donde i corre sobre un conjunto de representantes de las clases ciclotómicas módulo n y la aritmética es la de $\mathbb{F}_{q^m}$.

Ejemplo 1.2.8. Para $n = 7$ y $q = 2$ se tiene que $m = o_n(q) = 3$ así que las raíces de $x^7 - 1$ están todas en $\mathbb{F}_8$ y las clases ciclotómicas para q módulo n son

$$\begin{aligned} C_0 &= \{0\} \\ C_1 &= \{1, 2, 4\} \\ C_3 &= \{3, 6, 5\} \end{aligned}$$

Además, se tiene que $\mathbb{F}_8 = \mathbb{F}_2[x]/\langle x^3 + x + 1 \rangle$ y si α es una raíz de $x^3 + x + 1$, genera $\mathbb{F}_8^*$ y por tanto es también raíz séptima primitiva de la unidad. Sea $m_i(x) = irr(\alpha^i, \mathbb{F}_2)$, entonces los polinomios irreducibles sobre $\mathbb{F}_2$ son

$$\begin{aligned} m_o(x) &= x - 1 = x + 1 \\ m_1(x) &= (x - \alpha)(x - \alpha^2)(x - \alpha^4) = x^3 + x + 1 \\ m_3(x) &= (x - \alpha^3)(x - \alpha^6)(x - \alpha^5) = x^3 + x^2 + 1 \end{aligned}$$

Por lo tanto, $x^7 - 1 = (x + 1)(x^3 + x + 1)(x^3 + x^2 + 1)$.

Esto nos permite describir todos los códigos cíclicos binarios de longitud 7 con polinomio generador $g(x)$ que se dan en la Tabla 1.

	$g_i(x)$	dim
$\mathcal{C}_0$	$m_0(x)m_1(x)m_3(x) = x^7 - 1$	0
$\mathcal{C}_1$	$m_1(x)m_3(x) = x^6 + x^5 + \cdots + 1$	1
$\mathcal{C}_2$	$m_0(x)m_1(x) = x^4 + x^3 + x^2 + 1$	3
$\mathcal{C}_3$	$m_0(x)m_3(x) = x^4 + x^2 + x + 1$	3
$\mathcal{C}_4$	$m_1(x) = x^3 + x + 1$	4
$\mathcal{C}_5$	$m_3(x) = x^3 + x^2 + 1$	4
$\mathcal{C}_6$	1	7

Tabla 1. Códigos cíclicos binarios de longitud $n = 7$.

Consideremos en particular el código $\mathcal{C}$ generado por $g_4(x) = x^3 + x + 1$. Del Teorema 1.2.6, una matriz generadora para $\mathcal{C}$ es

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

Para dar una matriz verificadora de paridad consideramos el polinomio verificador dado por

$$\begin{aligned} h(x) &= (x^7 - 1)/(x^3 + x + 1) \\ &= x^4 + x^2 + x + 1 \end{aligned}$$

así que de (2), una matriz verificadora para $\mathcal{C}$ es

$$H = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

Nótese que las columnas de H son todos los vectores no cero de $\mathbb{F}_2^3$, luego $\mathcal{C}$ es un $\mathcal{H}_3$, o bien un $[7, 4]$ código de Hamming (ver Sección 1.1.1).

En general, puede decirse cuántos códigos cíclicos de longitud n hay en $\mathcal{R}_n$ pues de (4) se ve que $x^n - 1$ tiene tantos factores irreducibles como clases ciclotómicas módulo n. Sea k el número de estas clases. Para construir un código cíclico de longitud n debe darse su polinomio generador de modo que sea producto de algunos de estos factores irreducibles, por tanto hay 2^k formas de elegir el polinomio generador. Es decir, el número de códigos cíclicos de longitud n es 2^k donde k es el

número de clases ciclotómicas para q módulo n. Más aún, la dimensión de estos códigos cíclicos en $\mathcal{R}_n$ son las posibles sumas de las cardinalidades de estas clases.

1.2.2. Los ceros de un código cíclico y la cota BCH

En la sección anterior se mostró que si $m = o_n(q)$ y $\alpha \in \mathbb{F}_{q^m}$ es una raíz n-ésima primitiva de la unidad, entonces podemos factorizar a $x^n - 1$ sobre $\mathbb{F}_{q^m}$ como

$$x^n - 1 = \prod_{j=0}^{n-1}(x - \alpha^j)$$

más aun,

$$x^n - 1 = \prod_{i} irr(\alpha^i, \mathbb{F}_q)$$

es la factorización de $x^n - 1$ como producto de irreducibles sobre $\mathbb{F}_q$, donde i corre sobre un conjunto de representantes de las clases ciclotómicas módulo n.

Consideremos ahora un código cíclico $\mathcal{C}$ en $\mathcal{R}_n$ con polinomio generador $g(x)$. Entonces podemos escribir

$$g(x) = \prod_{i} \prod_{j \in C_i} (x - \alpha^j) = \prod_{i} irr(\alpha^i, \mathbb{F}_q)$$

donde i toma valores en algún subconjunto de representantes de las clases ciclotómicas C_i módulo n.

Sea $T = \bigcup_i C_i$ la unión de estas clases ciclotómicas. A T se le llama *conjunto de definición* de $\mathcal{C}$ y a las raíces de la unidad $Z(\mathcal{C}) = \{\alpha^i | i \in T\}$ se les llama *ceros del código* $\mathcal{C}$. Obsérvese que $|T| = \text{gr}(g(x))$.

De lo anterior se concluye que $c(x) \in \mathcal{C}$ si y sólo si $c(\alpha^i) = 0$ para cada $i \in T$ y que la dimensión de $\mathcal{C}$ es $n - |T|$.

Por ejemplo para el código binario $\mathcal{C}$ de longitud $n = 7$ con polinomio generador $g(x) = x^4 + x^3 + x^2 + 1$ construido a partir de la raíz n-ésima primitiva α como se vió en el Ejemplo 1.2.8, el conjunto de definición de $\mathcal{C}$ es $T = C_0 \bigcup C_1 = \{0, 1, 2, 4\}$ y sus ceros son $Z(\mathcal{C}) = \{\alpha^0, \alpha, \alpha^2, \alpha^4\}$.

Obsérvese que en general, T depende de la raíz n-ésima primitiva elegida. Pues para este mismo ejemplo α^3 es también raíz séptima primitiva y respecto a ella, el conjunto de definición del código es $T = \{0, 3, 5, 6\}$, porque

$$\begin{aligned} g(x) &= (x - (\alpha^3)^0)(x - (\alpha^3)^3)(x - (\alpha^3)^5)(x - (\alpha^3)^6) \\ &= (x - \alpha^0)(x - \alpha^2)(x - \alpha)(x - \alpha^4) \end{aligned}$$

Vamos a decir que el conjunto de definición T contiene s elementos consecutivos si existe un conjunto $\{b, b+1, \ldots, b+s-1\}$ de enteros consecutivos módulo n tal que $\{b, b+1, \ldots, b+s-1\} \subseteq T$.

Para cualquier código es esencial el determinar su distancia mínima para conocer su capacidad correctora. A continuación se presenta un importante resultado que da una cota inferior para la distancia mínima de códigos cíclicos con un cierto número de elementos consecutivos en su conjunto de definición.

Teorema 1.2.9 (cota BCH)**.** *Sea $\mathcal{C}$ un código cíclico de longitud n sobre $\mathbb{F}_q$ con conjunto de definición T tal que contiene $\delta - 1$ elementos consecutivos. Entonces, $\mathcal{C}$ tiene distancia mínima por lo menos δ.*

Demostración. Por hipótesis, los ceros de $\mathcal{C}$ incluyen $\alpha^b, \alpha^{b+1}, \ldots, \alpha^{b+\delta-2}$, luego si $c \in \mathcal{C}$ es una palabra no cero de peso $w < \delta$ se cumple que $c(\alpha^b) = c(\alpha^{b+1}) = \cdots = c(\alpha^{b+\delta-2}) = 0$. Sea

$$c(x) = \sum_{j=1}^{w} c_{i_j} x^{i_j} \in \mathcal{C}. \tag{5}$$

Si

$$M = \begin{bmatrix} \alpha^{i_1 b} & \alpha^{i_2 b} & \cdots & \alpha^{i_w b} \\ \alpha^{i_1 (b+1)} & \alpha^{i_2 (b+1)} & \cdots & \alpha^{i_w (b+1)} \\ & & \cdots & \\ \alpha^{i_1 (b+w-1)} & \alpha^{i_2 (b+w-1)} & \cdots & \alpha^{i_w (b+w-1)} \end{bmatrix}$$

y $u = (c_{i_1}, c_{i_2}, \ldots, c_{i_w})$, entonces $Mu^t = 0$ y como $u \neq 0$ se tiene que $\det(M) = 0$. Pero por otro lado, $\det(M) = (\alpha^{i_1 b} \alpha^{i_2 b} \cdots \alpha^{i_w b}) \det V$, donde V es la matriz de Vandermonde:

$$V = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \alpha^{i_1} & \alpha^{i_2} & \cdots & \alpha^{i_w} \\ & & \cdots & \\ \alpha^{i_1 (w-1)} & \alpha^{i_2 (w-1)} & \cdots & \alpha^{i_w (w-1)} \end{bmatrix}$$

y como los α^{i_j} son distintos, $\det(V) \neq 0$ lo cual contradice el hecho de que $\det(M) = 0$. Por tanto, el peso de c es por lo menos δ. $\square$

1.3. Códigos BCH

Los códigos BCH fueron descubiertos en el caso binario por A. Hocquenghem en 1959 e independientemente por R.C. Bose y D.K. Ray Chaudhuri en 1960. Más adelante en 1961 serían generalizados a cualquier campo finito por D.C Gorenstein y N. Zierler. Este tipo de códigos están diseñados para aprovechar la cota BCH. Para mayores detalles se puede consultar [**3**], [**9**], [**14**], [**21**].

Se desea construir códigos cíclicos de longitud n con distancia mínima y dimensión tan grandes como sea posible. Una gran distancia mínima se logra tomando un conjunto de definición T conteniendo un buen número de elementos consecutivos. Y como la dimensión del código es $n-|T|$ se busca que $|T|$ sea lo menor posible. Por tanto, si se quiere que el código tenga distancia mínima por lo menos δ debe escogerse T tan pequeño como se pueda y conteniendo $\delta - 1$ elementos consecutivos.

Definición 1.3.1. *Sean n, b y δ enteros tales que $2 \leq \delta \leq n$ y $b \geq 0$. Un código BCH $\mathcal{C}$ sobre $\mathbb{F}_q$ de longitud n es un código cíclico cuyo conjunto de definición es: $T = C_b \bigcup C_{b+1} \bigcup \cdots \bigcup C_{b+\delta-2}$, donde C_i es la clase ciclotómica para q módulo n que contiene al entero i. Al entero δ se le conoce como la distancia diseñada del código $\mathcal{C}$.*

Obsérvese que el polinomio generador de un código así definido es

$$g(x) = LCM(irr(\alpha^b, \mathbb{F}_q), irr(\alpha^{b+1}, \mathbb{F}_q), \ldots, irr(\alpha^{b+\delta-2}, \mathbb{F}_q))$$

y que por la cota BCH tiene distancia mínima por lo menos δ. Además, cualquier código cíclico es un código BCH con distancia diseñada $\delta = 2$.

Ejemplo 1.3.2. Veamos algunos códigos BCH sobre $\mathbb{F}_3$ de longitud $n = 16$. Se tiene que $o_{16}(3) = 4$ de modo que $x^{16} - 1$ tiene todas sus raíces en $\mathbb{F}_{81}$. Las distintas clases ciclotómicas para 3 módulo 16 son:

$$\begin{aligned} C_0 &= \{0\} \\ C_1 &= \{1, 3, 9, 11\} \\ C_2 &= \{2, 6\} \\ C_4 &= \{4, 12\} \\ C_5 &= \{5, 7, 13, 15\} \\ C_8 &= \{8\} \\ C_{10} &= \{10, 14\} \end{aligned}$$

Ahora, un polinomio primitivo sobre $\mathbb{F}_3$ es $x^4 + x^3 + 2$, esto significa que si $\alpha \in \mathbb{F}_{81}$ es una raíz de este polinomio entonces $\mathbb{F}_{81}^* = \langle \alpha \rangle$.

Además $\beta = \alpha^5$ es una raíz n-ésima primitiva de la unidad pues su orden es $80/(5, 80) = 16$.

Si $m_i(x) = irr(\beta^i, \mathbb{F}_3)$, los polinomios irreducibles sobre $\mathbb{F}_3$ son

$$\begin{aligned} m_0(x) &= x - 1 = x + 2 \\ m_1(x) &= (x-\beta)(x-\beta^3)(x-\beta^9)(x-\beta^{11}) = x^4 + 2x^2 + 2 \\ m_2(x) &= (x-\beta^2)(x-\beta^6) = x^2 + 2x^2 + 2 \\ m_4(x) &= (x-\beta^4)(x-\beta^{12}) = x^2 + 1 \\ m_5(x) &= (x-\beta^5)(x-\beta^7)(x-\beta^{13})(x-\beta^{15}) = x^4 + x^2 + 2 \\ m_8(x) &= (x-\beta^8) = x + 1 \\ m_{10}(x) &= (x-\beta^{10})(x-\beta^{14}) = x^2 + x + 2 \end{aligned}$$

La Tabla 1.3.2 muestra algunos códigos BCH tomando $b = 1$. Recuérdese que códigos así construidos tienen distancia mínima por lo menos δ.

	δ	dim	T	$g(x)$
$\mathcal{C}_1$	2	12	C_1	$x^4 + 2x^2 + 2$
$\mathcal{C}_2$	3,4	10	$C_1 \bigcup C_2$	$x^6 + 2x^5 + x^4 + x^3 + x + 1$
$\mathcal{C}_3$	5	8	$C_1 \bigcup C_2 \bigcup C_4$	$x^8 + 2x^7 + 2x^6 + x^4 + 2x^3$ $+x^2 + x + 1$
$\mathcal{C}_4$	6,7,8	4	$C_1 \bigcup C_2 \bigcup C_4 \bigcup C_5$	$x^{12} + 2x^{11} + 2x^9 + 2x^8 + x^4$ $+2x^3 + 2x + 2$
$\mathcal{C}_5$	9,10	3	$C_1 \bigcup C_2 \bigcup C_4 \bigcup C_5 \bigcup C_8$	$x^{13} + 2x^{11} + 2x^{10} + x^9 + 2x^8$ $+x^5 + 2x^3 + 2x^2 + x + 2$
$\mathcal{C}_6$	11,...,16	1	$C_1 \bigcup C_2 \bigcup C_4 \bigcup C_5 \bigcup C_8 \bigcup C_{10}$	$x^{15} + x^{14} + x^{13} + \cdots + x + 1$

Tabla 2. Algunos códigos BCH de longitud $n = 16$ sobre $\mathbb{F}_3$

Teorema 1.3.3. *Sea $\mathcal{C}$ un código BCH sobre $\mathbb{F}_q$ de longitud n y distancia diseñada δ. Entonces, dim $\mathcal{C} \geq n - m(\delta - 1)$, donde $m = o_n(q)$.*

Demostración. La cardinalidad de cada clase ciclotómica para q módulo n es un divisor de $o_n(q)$. Por otro lado, el conjunto de definición de un código BCH con distancia diseñada δ es la unión de a lo más $\delta - 1$ clases ciclotómicas distintas, cada una de las cuales tiene cardinalidad a lo más $o_n(q)$, por lo tanto, la dimensión del código es por lo menos $n - o_n(q)(\delta - 1)$ □

1.4. Códigos de Reed-Solomon

En 1960 Irving Reed y Gustave Solomon [**20**] presentaron un tipo de códigos que ahora llevan su nombre el cual puede verse como una clase particular de códigos BCH. Los códigos de Reed-Solomon son de gran importancia tanto teórica como práctica. Son muy útiles en la construcción de otros códigos y actualmente son usados en telecomunicaciones, transmisión satelital (por ejemplo, la NASA utilizó este tipo de códigos en sus misiones Galileo a Júpiter en 1989, Magallanes a Venus ese mismo año y Ulises al Sol en 1990), en discos CD y DVD, etc. Para una descripción más detallada de los códigos de Reed-Solomon pueden consultarse [**9**], [**14**] y [**21**].

Definición 1.4.1. *Un código de Reed-Solomon (RS) $\mathcal{C}$ sobre el campo finito $\mathbb{F}_q$ es un código BCH de longitud $n = q - 1$.*

Obsérvese que en este caso $o_n(q) = 1$ así que los factores irreducibles de $x^n - 1$ son todos lineales y las clases ciclotómicas módulo n tienen cardinalidad 1. Aún más, las raíces de $x^n - 1$ son los elementos no cero de $\mathbb{F}_q$ y una raíz n-ésima primitiva es, en este caso, también un elemento primitivo de $\mathbb{F}_q$. Luego si $\mathcal{C}$ tiene distancia diseñada δ, el conjunto de definición de $\mathcal{C}$ tiene $\delta - 1$ elementos y $T = \{b, b+1, \ldots, b+\delta-2\}$ para algún entero $b \geq 1$. En general, un $[n, k, d]_q$-código lineal satisface que $k \leq n - d + 1$ (Teorema 1.1.8), luego del Teorema 1.3.3 se tiene que $k \geq n-\delta+1 \geq n-d+1 \geq k$. Lo cual implica que $d = \delta$ y $k = n-d+1$. En conclusión se tiene el siguiente,

Teorema 1.4.2. *Sea $\mathcal{C}$ un código RS sobre $\mathbb{F}_q$ de longitud $n = q - 1$ y distancia diseñada δ, entonces*

1. *$\mathcal{C}$ tiene conjunto de definición $T = \{b, b+1, \ldots, b+\delta-2\}$ para algún entero $b \geq 1$.*
2. *$\mathcal{C}$ tiene distancia mínima $d = \delta$ y dimensión $k = n - d + 1$. Es decir, $\mathcal{C}$ es un código MDS (ver página 14).*

Damos ahora una forma alternativa de definir los códigos de Reed-Solomon que es la forma en que fueron presentados originalmente por sus autores. Para $k \geq 0$, sea $P_k = \mathbb{F}_q[x]/\langle x^k - 1\rangle$, el $\mathbb{F}_q$-espacio lineal de los polinomios sobre $\mathbb{F}_q$ de grado menor que k.

Sea $\Gamma = \mathbb{F}_q^*$ y α un elemento primitivo de $\mathbb{F}_q$. El mapeo evaluación, $\mathrm{ev}_\Gamma : P_k \to \mathbb{F}_q^{q-1}$ está dado por $\mathrm{ev}_\Gamma(f) = (f(1), f(\alpha), \ldots, f(\alpha^{q-2}))$.

Teorema 1.4.3. *Sea α un elemento primitivo de $\mathbb{F}_q$ y k un entero con $0 \leq k \leq n = q - 1$. Entonces,*

$$\mathcal{C} = Im(\mathrm{ev}_\Gamma) = \mathrm{ev}_\Gamma(P_k)$$

es el $[n, k, n-k+1]_q$ *código de Reed-Solomon con* $b = 1$ *sobre* $\mathbb{F}_q$.

DEMOSTRACIÓN. Claramente $\mathcal{C}$ es un código lineal sobre $\mathbb{F}_q$ pues P_k es un espacio vectorial sobre $\mathbb{F}_q$. Veremos que tiene dimensión k, para ello sea

$$\psi : P_k \to \mathcal{C}$$

el mapeo dado por $\psi(f) = (f(1), f(\alpha), f(\alpha^2), \ldots, f(\alpha^{q-2}))$. Si f_1 y f_2 son polinomios en P_k tales que sus imágenes son iguales, entonces su diferencia es cero, es decir que $f_1 - f_2$ es un polinomio de grado a lo más $k-1$ con $q-1 = n \geq k$ raíces, así que debe ser $f_1 = f_2$. Esto muestra que ψ es uno a uno y por lo tanto como P_k es de dimensión k, también lo es $\mathcal{C}$. Veremos ahora que si D es el $[n, k, n-k+1]_q$ RS código con $b = 1$ sobre $\mathbb{F}_q$, entonces $C \subseteq D$, lo cual completa la prueba. El código D tiene como conjunto de definición a $T = \{1, 2, \ldots, n-k\}$. Sea $c(x) = \sum_{j=0}^{n-1} c_j x^j \in \mathcal{C}$. Por definición de $\mathcal{C}$ existe $f(x) = \sum_{m=0}^{k-1} f_m x^m \in P_k$ tal que $c_j = f(\alpha^j)$ para $0 \leq j < n$. Para ver que $c(x) \in D$ debe mostrarse que $c(x)$ se anula en los ceros del código D. Si $i \in T$ entonces

$$\begin{aligned} c(\alpha^i) &= \sum_{j=0}^{n-1} c_j \alpha^{ij} = \sum_{j=0}^{n-1} (\sum_{m=0}^{k-1} f_m \alpha^{jm}) \alpha^{ij} \\ &= \sum_{m=0}^{k-1} f_m \sum_{j=0}^{n-1} \alpha^{(i+m)j} = \sum_{m=0}^{k-1} f_m \frac{\alpha^{(i+m)n} - 1}{\alpha^{i+m} - 1} \end{aligned}$$

pero $\alpha^{(i+m)n} = 1$ y $\alpha^{i+m} \neq 1$ puesto que $1 \leq i+m \leq n-1 = q-2$ y α es una raíz n-ésima primitiva de la unidad. Por tanto, $c(\alpha^i) = 0$ para $i \in T$ como se quería. □

Capítulo 2

Algoritmos de decodificación para códigos BCH sobre $\mathbb{F}_q$

En este capítulo se presentan dos métodos para decodificar códigos BCH. El primero es conocido como el algoritmo de Peterson, Gorenstein y Zierler. Fué originalmente desarrollado para códigos binarios por W.W Peterson [**18**] en 1960 y generalizado en 1961 para códigos BCH no binarios por D.C. Gorenstein y N. Zierler [**8**]. El segundo método, conocido como el algoritmo de Berlekamp-Massey, fué desarrollado por E.R. Berlekamp [**4**] en 1967. En 1969, J.L. Massey [**17**] muestra que el algoritmo dado por Berlekamp resuelve el problema de encontrar la menor recurrencia lineal que genera una sucesión finita dada.

2.1. Algoritmo de Peterson, Gorenstein y Zierler

Sea $\mathcal{C}$ un código BCH sobre $\mathbb{F}_q$ construido a partir de la raíz n-ésima primitiva $\alpha \in \mathbb{F}_{q^m}$ y cuyo polinomio generador es

$$g(x) = LCM(irr(\alpha^b, \mathbb{F}_q), irr(\alpha^{b+1}, \mathbb{F}_q), \ldots, irr(\alpha^{b+2t-1}, \mathbb{F}_q))$$

Por la cota BCH (Teorema 1.2.9), este código tiene distancia mínima por lo menos $2t+1$ así que es capaz de corregir por lo menos t errores (Teorema 1.1.7).

Supóngase que se envía la palabra $c(x) \in \mathcal{C}$ pero que se recibe $r(x) = c(x) + e(x)$, donde $e(x)$ es el polinomio error con a lo más t coeficientes distintos de cero. Supóngase además que ocurrieron exactamente v errores, con $0 \leq v \leq t$ en las posiciones $i_1, i_2, \ldots, i_v$. Podemos escribir entonces $e(x) = e_{i_1}x^{i_1} + e_{i_2}x^{i_2} + \cdots + e_{i_v}x^{i_v}$ donde $e_{i_l} \in \mathbb{F}_q$ es la magnitud del l-ésimo error.

Para $r(x)$ y $j = 1, 2, \ldots, 2t$ se definen los síndromes como:

$$\begin{aligned} S_j &= r(\alpha^{b+j-1}) \\ &= c(\alpha^{b+j-1}) + e(\alpha^{b+j-1}) \\ &= e(\alpha^{b+j-1}) \end{aligned}$$

Obsérvese que si el polinomio error es nulo, los síndromes son todos cero. Con el fin de simplificar la notación hacemos $Y_l = e_{i_l}$, $X_l = \alpha^{i_l}$ para $l = 1, 2, \ldots, v$ donde i_l es la posición del l-ésimo error. Las Y_l son las magnitudes de error y las X_l son los números de localización de error que junto con el número v de errores ocurridos son valores desconocidos. Obsérvese además que las X_l son todas distintas porque α es un elemento de orden n.

Se tiene pues el siguiente sistema no lineal de $2t$ ecuaciones en las v localizaciones y las v magnitudes de error:

$$\begin{aligned} S_1 &= Y_1 X_1^b + \cdots + Y_v X_v^b \\ S_2 &= Y_1 X_1^{b+1} + \cdots + Y_v X_v^{b+1} \\ &\vdots \\ S_{2t} &= Y_1 X_1^{b+2t-1} + \cdots + Y_v X_v^{b+2t-1} \end{aligned} \tag{6}$$

Obsérvese que en el sistema (6) los únicos valores conocidos son los síndromes y para calcular éstos sólo se requiere conocer el polinomio recibido $r(x)$ y los ceros del código correspondiente. Además, debido a la forma en que se definieron los síndromes, este sistema tiene por lo menos una solución. La idea es encontrar las Y_l y las X_l a partir de los síndromes. Para ello, se definen algunas variables intermedias que pueden calcularse a partir de los síndromes y de las cuales se puedan determinar las localizaciones de error. Considérese el siguiente polinomio en $\mathbb{F}_q[x]$,

$$\Lambda(x) = \Lambda_v x^v + \Lambda_{v-1} x^{v-1} + \cdots + \Lambda_1 x + 1 \tag{7}$$

llamado *polinomio localizador* de errores y definido como el polinomio cuyas raíces son las inversas, X_l^{-1}, de las localizaciones de error, para $l = 1, 2, \ldots, v$. Esto es,

$$\Lambda(x) = (1 - X_1 x)(1 - X_2 x) \cdots (1 - X_v x)$$

Si se conocieran los coeficientes de $\Lambda(x)$ podríamos calcular sus raíces para obtener así las localizaciones de error. Veamos pues la

relación entre los coeficientes de $\Lambda(x)$ y los síndromes conocidos. Multiplicando el polinomio localizador dado en (7) por $Y_l X_l^{b+j+v-1}$ y haciendo $x = X_l^{-1}$ se obtiene

$$0 = Y_l X_l^{b+j+v-1}(1 + \Lambda_1 X_l^{-1} + \Lambda_2 X_l^{-2} + \cdots + \Lambda_v X_l^{-v})$$

o bien

$$Y_l(X_l^{b+j+v-1} + \Lambda_1 X_l^{b+j+v-2} + \cdots + \Lambda_v X_l^{b+j-1}) = 0$$

la cual es una ecuación válida para cada l y para cada j. Sumando estas ecuaciones para $l = 1, \ldots, v$, se tiene:

$$\sum_{l=1}^{v} Y_l X_l^{b+j+v-1} + \Lambda_1 \sum_{l=1}^{v} Y_l X_l^{b+j+v-2} + \cdots + \Lambda_v \sum_{l=1}^{v} Y_l X_l^{b+j-1} = 0$$

Esto es,

$$S_{j+v} + \Lambda_1 S_{j+v-1} + \cdots + \Lambda_v S_j = 0$$

Y como $v \leq t$, si tomamos $1 \leq j \leq v$ los subíndices indican síndromes conocidos. En conclusión tenemos las siguientes ecuaciones que relacionan los síndromes con los coeficientes del polinomio localizador de errores $\Lambda(x)$:

$$\Lambda_1 S_{j+v-1} + \Lambda_2 S_{j+v-2} + \cdots + \Lambda_v S_j = -S_{j+v} \quad \text{para } j = 1, \ldots, v \tag{8}$$

En forma matricial,

$$\begin{bmatrix} S_1 & S_2 & S_3 & \cdots & S_{v-1} & S_v \\ S_2 & S_3 & S_4 & \cdots & S_v & S_{v+1} \\ S_3 & S_4 & S_5 & \cdots & S_{v+1} & S_{v+2} \\ \vdots & & & & & \\ S_v & S_{v+1} & S_{v+2} & \cdots & S_{2v-2} & S_{2v-1} \end{bmatrix} \begin{bmatrix} \Lambda_v \\ \Lambda_{v-1} \\ \Lambda_{v-2} \\ \vdots \\ \Lambda_1 \end{bmatrix} = \begin{bmatrix} -S_{v+1} \\ -S_{v+2} \\ -S_{v+3} \\ \vdots \\ -S_{2v} \end{bmatrix} \tag{9}$$

Así pues, el problema de encontrar los coeficientes del polinomio localizador de errores se reduce a resolver un sistema de v ecuaciones lineales en las v incógnitas $\Lambda_1, \Lambda_2, \ldots, \Lambda_v$. Sin embargo, se tiene la dificultad de que v es un valor desconocido, a ese respecto el siguiente resultado es de utilidad.

Teorema 2.1.1. *La matriz de síndromes,*

$$M_\mu = \begin{bmatrix} S_1 & S_2 & \cdots & S_\mu \\ S_2 & S_3 & \cdots & S_{\mu+1} \\ \vdots & & & \\ S_\mu & S_{\mu+1} & \cdots & S_{2\mu-1} \end{bmatrix}$$

es invertible si $\mu = v$ donde v es el número de errores que ocurrieron. La matriz es singular si $\mu > v$.

Demostración. Sea $X_\mu = 0$ para $\mu > v$. De las ecuaciones (6) se ve que $M_\mu = A_\mu B_\mu A_\mu^t$ donde,

$$A_\mu = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ X_1 & X_2 & \cdots & X_\mu \\ \vdots & & & \\ X_1^{v-1} & X_2^{v-1} & \cdots & X_\mu^{\mu-1} \end{bmatrix}$$

y

$$B_\mu = \begin{bmatrix} Y_1X_1 & 0 & \cdots & 0 \\ 0 & Y_2X_2 & \cdots & 0 \\ \vdots & & \ddots & \\ 0 & 0 & \cdots & Y_\mu X_\mu \end{bmatrix}$$

con las X_l y las Y_l definidas como antes.

Por tanto, $\det(M_\mu) = \det(A_\mu)\det(B_\mu)\det(A_\mu)$. Si $\mu > v$, entonces $\det(B_\mu) = 0$ y M_μ es singular. Por otro lado como A_μ es una matriz de Vandermonde, tiene determinante distinto de cero siempre y cuando sus columnas sean distintas y no nulas, lo cual ocurre si $\mu = v$. Además cuando $\mu = v$ también se tiene que $\det(B_\mu) \neq 0$. Por tanto si $\mu = v$, M_μ es invertible. □

El teorema anterior es la base para un algoritmo decodificador de códigos BCH con las características mencionadas anteriormente. Primero debe encontrarse el valor correcto de v de la siguiente manera:

Probar con $v = t$ y calcular $\det(M_v)$. Si es distinto de cero, éste es el valor correcto de v pero si $\det(M_v) = 0$, se debe disminuir el valor de v a $v = t - 1$ para probar de nuevo. Mientras se obtenga un determinante cero se continúa reduciendo en uno el valor de v hasta obtener $\det(M_v) \neq 0$; cuando ésto ocurre se ha encontrado el valor verdadero de v.

El siguiente paso es invertir M_v y calcular el polinomio localizador de errores, $\Lambda(x)$, resolviendo el sistema (9).

Finalmente deben determinarse los ceros de $\Lambda(x)$ para encontrar las localizaciones de error. Si el código es binario automáticamente se tienen también las magnitudes de los errores, pues todas ellas son 1 en las posiciones en que hubo error. Si no, se retoman las ecuaciones que definen los síndromes dadas en (6). Como en este punto ya se conocen los valores de X_l se tienen $2t$ ecuaciones lineales en v variables. Las primeras v ecuaciones pueden resolverse si la matriz

$$A = \begin{bmatrix} X_1^b & X_2^b & \cdots & X_v^b \\ X_1^{b+1} & X_2^{b+1} & \cdots & X_v^{b+1} \\ \vdots & & & \\ X_1^{b+v-1} & X_2^{b+v-1} & \cdots & X_v^{b+v-1} \end{bmatrix}$$

es invertible, es decir si

$$\det(A) = (X_1^b X_2^b \cdots X_v^b) \det \begin{bmatrix} 1 & 1 & \cdots & 1 \\ X_1 & X_2 & \cdots & X_v \\ \vdots & & & \\ X_1^{v-1} & X_2^{v-1} & \cdots & X_v^{v-1} \end{bmatrix} \neq 0$$

Si hay v errores, los X_l son no cero y distintos. Por tanto en este caso $\det(A) \neq 0$ y el sistema (6) tiene solución.

En síntesis el algoritmo de Peterson-Gorenstein-Zierler para decodificar códigos BCH es el siguiente:

1. Calcular los síndromes $S_j = r(\alpha^{b+j-1})$ para $j = 1, 2, \ldots, 2t$.
2. Hacer $v = t$ y calcular $\det(M_v)$. Mientras $\det(M_v) = 0$ hacer $v \to v - 1$.
3. Calcular la inversa de la matriz M_v y resolver (9) para obtener $\Lambda(x)$.
4. Encontrar las raíces de $\Lambda(x)$. Invertir estos valores para obtener las localizaciones de error X_l.
5. Resolver las primeras v ecuaciones de (6) para determinar las magnitudes de error Y_l.

2.1.1. Ejemplos

Ejemplo 2.1.2. Consideremos el $[15, 7, 5]$ código binario BCH con polinomio generador

$$\begin{aligned} g(x) &= LCM(irr(\alpha, \mathbb{F}_2), irr(\alpha^2, \mathbb{F}_2), \ldots, irr(\alpha^6, \mathbb{F}_2)) \\ &= x^{10} + x^8 + x^5 + x^4 + x^2 + x + 1 \end{aligned}$$

Donde

$$\begin{aligned} irr(\alpha, \mathbb{F}_2) &= irr(\alpha^2, \mathbb{F}_2) = irr(\alpha^4, \mathbb{F}_2) = x^4 + x + 1 \\ irr(\alpha^3, \mathbb{F}_2) &= irr(\alpha^6, \mathbb{F}_2) = x^4 + x^3 + x^2 + x + 1 \\ irr(\alpha^5, \mathbb{F}_2) &= x^2 + x + 1 \end{aligned}$$

La aritmética es la de $\mathbb{F}_{2^4} = \mathbb{F}_2[y]/\langle y^4 + y + 1\rangle$ y α es un elemento primitivo de $\mathbb{F}_{2^4}$ el cual es raíz de $y^4 + y + 1$. En este caso $t = 3$. Supóngase que se envía la palabra $c(x) = x^{12}+x^{11}+x^9+x^8+x^7+x^2+1$ y que en su lugar se recibe $r(x) = x^{14} + x^{13} + x^{12} + x^{11} + x^9 + x^8 + x^7 + x^3 + x^2 + 1$. Para decodificar, primero se calculan los síndromes:

$$\begin{aligned} S_1 &= r(\alpha) = \alpha^6 \\ S_2 &= r(\alpha^2) = \alpha^{12} \\ S_3 &= r(\alpha^3) = \alpha^{12} \\ S_4 &= r(\alpha^4) = \alpha^9 \\ S_5 &= r(\alpha^5) = 0 \\ S_6 &= r(\alpha^6) = \alpha^9 \end{aligned}$$

A continuación se busca el número de errores ocurridos. En principio se prueba con $v = 3$. En este caso se tiene que:

$$M = \begin{bmatrix} S_1 & S_2 & S_3 \\ S_2 & S_3 & S_4 \\ S_3 & S_4 & S_5 \end{bmatrix} = \begin{bmatrix} \alpha^6 & \alpha^{12} & \alpha^{12} \\ \alpha^{12} & \alpha^{12} & \alpha^9 \\ \alpha^{12} & \alpha^9 & 0 \end{bmatrix}$$

Entonces $\det(M) = \alpha^5 \neq 0$ y por lo tanto ocurrieron $v = 3$ errores. Ahora debe invertise la matriz M para resolver el sistema (9) obteniendo,

$$\begin{aligned} \begin{bmatrix} \Lambda_3 \\ \Lambda_2 \\ \Lambda_1 \end{bmatrix} = M^{-1} \begin{bmatrix} -S_4 \\ -S_5 \\ -S_6 \end{bmatrix} &= \begin{bmatrix} \alpha^{13} & \alpha & 1 \\ \alpha & \alpha^4 & \alpha^2 \\ 1 & \alpha^2 & \alpha^{11} \end{bmatrix} \begin{bmatrix} -\alpha^9 \\ 0 \\ -\alpha^9 \end{bmatrix} \\ &= \begin{bmatrix} 1 \\ \alpha^{14} \\ \alpha^6 \end{bmatrix} \end{aligned}$$

Por lo tanto, el polinomio localizador de errores es:

$$\begin{aligned} \Lambda(x) &= x^3 + \alpha^{14}x^2 + \alpha^6 x + 1 \\ &= (x - \alpha)(x - \alpha^2)(x - \alpha^{12}) \\ &= (1 - \alpha^{14}x)(1 - \alpha^{13}x)(1 - \alpha^3 x) \end{aligned}$$

Por consiguiente las localizaciones de error son $X_1 = \alpha^{14}$, $X_2 = \alpha^{13}$ y $X_1 = \alpha^3$ lo cual indica que los errores ocurrieron en las posiciones 14, 13 y 3. Como el código es binario el polinomio de error es: $e(x) = x^{14} + x^{13} + x^3$ y la palabra enviada es,

$$\begin{aligned} c(x) &= r(x) - e(x) \\ &= x^{12} + x^{11} + x^9 + x^8 + x^7 + x^2 + 1 \end{aligned}$$

Ejemplo 2.1.3. Consideremos el código de Reed-Solomon de longitud $n = 10$ sobre $\mathbb{F}_{11}$ con polinomio generador

$$\begin{aligned} g(x) &= \prod_{i=1}^{8}(x - \alpha^i) \\ &= x^8 + 7x^7 + 10x^6 + 6x^5 + 7x^7 + 7x^4 + 3x^3 + 8x^2 + 5x + 9 \end{aligned}$$

En este caso $\alpha = 2$ es un generador de $\mathbb{F}_{11}^*$ y es raíz décima primitiva de la unidad. Este código corrige por lo menos $t = 4$ errores y tiene dimensión 2.

Supongamos que se envía la palabra cero pero en su lugar se recibe

$$\overline{r} = (0, 0, 0, 8, 0, 0, 0, 5, 0, 0)$$

el polinomio correspondiente es $r(x) = 8x^3 + 5x^7$ y los síndromes son:

$$\begin{array}{ll} S_1 = r(\alpha) = 0 & S_5 = r(\alpha^5) = 9 \\ S_2 = r(\alpha^2) = 9 & S_6 = r(\alpha^6) = 0 \\ S_3 = r(\alpha^3) = 3 & S_7 = r(\alpha^7) = 2 \\ S_4 = r(\alpha^4) = 3 & S_8 = r(\alpha^8) = 8 \end{array}$$

Buscamos el número de errores v ocurrido para lo cual probamos primero con $v = 4$. La matriz M es,

$$M = \begin{bmatrix} 0 & 9 & 3 & 3 \\ 9 & 3 & 3 & 9 \\ 3 & 3 & 9 & 0 \\ 3 & 9 & 0 & 2 \end{bmatrix}$$

cuyo determinante es $\det(M) = 0$ en $\mathbb{F}_{11}$ por lo que ahora tomamos $v = 3$ y probamos de nuevo. En este caso M es,

$$M = \begin{bmatrix} 0 & 9 & 3 \\ 9 & 3 & 3 \\ 3 & 3 & 9 \end{bmatrix}$$

con $\det(M) = 0$. Nuevamente disminuimos el valor de v, quedando $v = 2$ y ahora,

$$M = \begin{bmatrix} 0 & 9 \\ 9 & 3 \end{bmatrix}$$

con $\det(M) = 7$ lo cual indica que ocurrieron $v = 2$ errores. A continuación se obtiene la inversa de la matriz M para resolver el sistema (9):

$$\begin{bmatrix} \Lambda_2 \\ \Lambda_1 \end{bmatrix} = M^{-1} \begin{bmatrix} -S_3 \\ -S_4 \end{bmatrix} = \begin{bmatrix} 2 & 5 \\ 5 & 0 \end{bmatrix} \begin{bmatrix} -3 \\ -3 \end{bmatrix} = \begin{bmatrix} 1 \\ 7 \end{bmatrix}$$

De aquí que el polinomio localizador de errores es,

$$\begin{aligned} \Lambda(x) &= x^2 + 7x + 1 \\ &= (x - 8)(x - 7) \\ &= (1 - 7x)(1 - 8x) \end{aligned}$$

y las localizaciones de error son $X_1 = 7 = \alpha^7 =$ y $X_2 = 8 = \alpha^3$. Así que los errores ocurrieron en las posiciones 7 y 3. Finalmente, para hallar sus magnitudes resolvemos el sistema (6):

$$\begin{aligned} S_1 &= Y_1 X_1 + Y_2 X_2 \\ S_2 &= Y_1 X_1^2 + Y_2 X_2^2 \end{aligned}$$

o bien, matricialmente:

$$\begin{bmatrix} 0 \\ 9 \end{bmatrix} = \begin{bmatrix} 7 & 8 \\ 5 & 9 \end{bmatrix} \begin{bmatrix} Y_1 \\ Y_2 \end{bmatrix}$$

Para resolver invertimos la matriz correspondiente obteniendo:

$$\begin{bmatrix} Y_1 \\ Y_2 \end{bmatrix} = \begin{bmatrix} 9 & 3 \\ 6 & 7 \end{bmatrix} \begin{bmatrix} 0 \\ 9 \end{bmatrix} = \begin{bmatrix} 5 \\ 8 \end{bmatrix}$$

En conclusión, el polinomio de error es $e(x) = 5x^7 + 8x^3$ y por tanto la palabra enviada es

$$c(x) = r(x) - e(x) = 0$$

Observación 2.1.4. En los ejemplos anteriores la raíz n-ésima primitiva con la que se construye el código BCH es también un elemento primitivo del campo correspondiente $\mathbb{F}_{q^m}$ lo cual simplifica el trabajo. Ésto siempre ocurre en el caso de los códigos Reed-Solomon (Sección 1.4) pero en general no es así (ver Ejemplo 1.3.2). Sin embargo, dado un elemento primitivo de $\mathbb{F}_{q^m}$, digamos α, pueden obtenerse

las raíces n-ésimas primitivas de la unidad de la siguiente manera. Como $n \mid q^m - 1$ podemos escribir $q^m - 1 = nk$ y entonces el orden de α^i es

$$ord(\alpha^i) = \frac{q^m - 1}{(q^m - 1, i)} = \frac{nk}{(nk, i)}$$

por lo que α^i es una raíz n-ésima primitiva de la unidad si y sólo si $(nk, i) = k$. Lo cual ocurre solamente si $i = ku$ con u primo relativo a n. De lo anterior se sigue que las raíces n-ésimas primitivas de la unidad en $\mathbb{F}_{q^m}$ son

$$\{\alpha^i \mid i = \frac{q^m - 1}{n} u, u < n, (u, n) = 1\}$$

En particular, $\alpha^{\frac{q^m-1}{n}}$ es una raíz n-ésima primitiva de la unidad.

2.2. Cálculo de las localizaciones de los errores: el algoritmo de Berlekamp-Massey

Muchos de los cálculos requeridos para decodificar códigos BCH usando el algoritmo descrito en la sección anterior se centran en la solución del sistema (9). A continuación se presentará un algoritmo que permite resolver este sistema de una forma muy eficiente, el cual fué presentado por E.R. Berlekamp [**4**].

Supóngase que se conocen los valores $\Lambda_1, \Lambda_2, \ldots, \Lambda_v$, entonces el primer renglón de (9) define a S_{v+1} en términos de $S_1, \ldots, S_v$; el segundo define a S_{v+2} en términos de $S_2, \ldots, S_{v+1}$; y así sucesivamente. Este proceso está dado por la ecuación:

$$S_j = -\sum_{i=1}^{v} \Lambda_i S_{j-i} \quad \text{para} \quad j = v+1, \ldots, 2v \tag{10}$$

Para valores de Λ_i fijos, esta ecuación define una recurrencia lineal para los síndromes en términos de los v primeros. Expliquemos esto.

Una recurrencia lineal de longitud L consta de coeficientes Λ_1, Λ_2,...,Λ_L y valores iniciales S_1, S_2,...,S_L con los cuales se construyen S_{L+1}, S_{L+2},$\cdots$ por medio de la relación:

$$S_j = -\sum_{i=1}^{L} \Lambda_i S_{j-i} \quad j = L+1, L+2 \ldots \tag{11}$$

Tanto los valores iniciales como los coeficientes de la recurrencia se toman en el mismo campo, ya sea finito o infinito. No existe restricción para que el valor Λ_L sea distinto de cero.

Decimos que una recurrencia lineal genera una sucesión finita S_1, S_2,. . . ,S_n cuando sus valores coinciden con las n primeras salidas dadas por (11) para alguna inicialización adecuada. Obsérvese que si $L \geq n$ la recurrencia siempre genera la sucesión. Y si $L < n$, la recurrencia genera la sucesión si y sólo si

$$S_j = -\sum_{i=1}^{L} \Lambda_i S_{j-i} \quad j = L+1, L+2, \ldots, n \tag{12}$$

Recuérdese que los síndromes satisfacen (10), donde los Λ_i son los coeficientes del polinomio localizador de errores que es desconocido, por tanto, el objetivo es encontrar una recurrencia que, con valores iniciales adecuados, genere la sucesión de síndromes. Para ello, es útil considerar el polinomio cuyos coeficientes son los mismos que los de la recurrencia, es decir

$$\Lambda(x) = \Lambda_L x^L + \Lambda_{L-1} x^{L-1} + \cdots + \Lambda_1 x + 1$$

al cual, llamaremos *polinomio de conexión*. Es importante notar que como no se pide que $\Lambda_L \neq 0$ entonces $\mathrm{gr}(\Lambda(x)) \leq L$. De ahora en adelante denotaremos a una recurrencia lineal de longitud L por la pareja $(L, \Lambda(x))$, donde $\Lambda(x)$ es el polinomio de conexión.

Para encontrar una recurrencia que genere a S_1, S_2,. . . ,S_n se efectúa un proceso inductivo. En cada paso r, empezando con $r = 1$, se construye una recurrencia lineal $(L_r, \Lambda^{(r)}(x))$ de longitud mínima que genere los primeros r valores. Al inicio de la iteración r se tendrá una lista de recurrencias previas:

$$\begin{gathered}(L_1, \Lambda^{(1)}(x)),\\ (L_2, \Lambda^{(2)}(x)),\\ \vdots\\ (L_{r-1}, \Lambda^{(r-1)}(x))\end{gathered}$$

La idea principal del algoritmo de Berlekamp-Massey es encontrar la manera de calcular una nueva recurrencia de longitud mínima $(L_r, \Lambda^{(r)}(x))$ que genere la sucesión $S_1, \ldots, S_{r-1}, S_r$. La forma de hacerlo es usando la recurrencia más reciente y de ser necesario modificando

su longitud y polinomio de conexión. Para ello en la iteración r se calcula la siguiente salida de la $(r-1)$-ésima recurrencia, es decir,

$$\hat{S}_r = -\sum_{j=1}^{L_{r-1}} \Lambda_j^{(r-1)} S_{r-j}$$

y se resta $\hat{S}_r$ al valor deseado, S_r, para obtener la cantidad Δ_r conocida como la r-ésima discrepancia:

$$\begin{aligned} \Delta_r &= S_r - \hat{S}_r \\ &= S_r + \sum_{j=1}^{L_{r-1}} \Lambda_j^{(r-1)} S_{r-j} \\ &= \sum_{j=0}^{L_{r-1}} \Lambda_j^{(r-1)} S_{r-j} \end{aligned}$$

Si Δ_r es cero, no es necesario modificar la recurrencia pues en este caso a genera S_1, S_2,...,S_{r-1}, S_r y basta tomar $(L_r, \Lambda^{(r)}(x)) = (L_{r-1}, \Lambda^{(r-1)}(x))$. Si Δ_r no es cero, se construye el siguiente polinomio en la forma:

$$\Lambda^{(r)}(x) = \Lambda^{(r-1)}(x) + Ax^l \Lambda^{(m-1)}(x)$$

donde A es un elemento del campo en el que se está trabajando (que puede ser cualquiera pero en este contexto es $\mathbb{F}_{q^m}$), l es un entero y $\Lambda^{(m-1)}(x)$ es uno de los polinomios que se construyeron en las iteraciones anteriores. Con este nuevo polinomio recalculamos la discrepancia obteniendo:

$$\begin{aligned} \Delta'_r &= \sum_{j=0}^{L_r} \Lambda_j^{(r)} S_{r-j} \\ &= \sum_{j=0}^{L_{r-1}} \Lambda_j^{(r-1)} S_{r-j} + A \sum_{j=0}^{L_{m-1}} \Lambda_j^{(m-1)} S_{r-j-l} \end{aligned}$$

Si elegimos a m como un entero menor que r con $\Delta_m \neq 0$, $l = r - m$ y $A = -\Delta_m^{-1}\Delta_r$, entonces,

$$\Delta'_r = \Delta_r - \frac{\Delta_r}{\Delta_m}\Delta_m = 0$$

así que la recurrencia correspondiente al nuevo polinomio generará la sucesión S_1, ..., S_{r-1}, S_r. Sin embargo, se desea además que la recurrencia sea de longitud mínima. Vamos a ver que tomando m como la

más reciente iteración para la cual $L_m > L_{m-1}$, se obtiene una recurrencia de longitud mínima en cada paso. Para ello usaremos el siguiente lema:

Lema 2.2.1. *Supóngase que* $(L_{r-1}, \Lambda^{(r-1)}(x))$ *es una recurrencia lineal de longitud mínima que genera* S_1, S_2, ..., S_{r-1} *pero que no genera* S_1, S_2, ..., S_{r-1}, S_r *y que* $(L_r, \Lambda^{(r)}(x))$ *es una recurrencia que genera* S_1, S_2, ..., S_{r-1}, S_r. *Entonces*

$$L_r \geq \text{máx}\{L_{r-1}, r - L_{r-1}\}$$

Demostración. Vamos a ver que $L_r \geq L_{r-1}$ y $L_r \geq r - L_{r-1}$. La primera desigualdad es obvia porque si una recurrencia genera una sucesión, debe generar cualquier porción inicial de ella. La segunda desigualdad es clara si $L_{r-1} \geq r$. Supongamos pues que $L_{r-1} < r$ y que la segunda desigualdad no se cumple, es decir

$$L_r \leq r - 1 - L_{r-1} \tag{13}$$

Sea $c(x) = \Lambda^{(r-1)}(x)$, $b(x) = \Lambda^{(r)}(x)$, $L = L_{r-1}$ y $L' = L_r$. De la relación (13), se tiene que $r \geq L + L' + 1$, $L < r$, y por las hipótesis del lema,

$$S_r \neq -\sum_{i=1}^{L} c_i S_{r-i}, \tag{14}$$

$$S_j = -\sum_{i=1}^{L} c_i S_{j-i} \quad \text{para } j = L+1, \ldots, r-1 \tag{15}$$

y

$$S_j = -\sum_{k=1}^{L'} b_k S_{j-k} \quad \text{para } j = L'+1, \ldots, r \tag{16}$$

donde los c_i's son los coeficientes de $c(x)$ y los $b'_k s$ los de $b(x)$.

Sustituyendo (15) en (16) se obtiene,

$$S_r = -\sum_{k=1}^{L'} b_k S_{r-k} = \sum_{k=1}^{L'} b_k \sum_{i=1}^{L} c_i S_{r-k-i} \tag{17}$$

lo cual es válido porque $r - k$ corre de $r - 1$ hasta $r - L'$ que está en el rango $L + 1, \ldots, r - 1$ por la hipótesis de que $r \geq L + L' + 1$.

Por otro lado, de (14)

$$S_r \neq -\sum_{i=1}^{L} c_i S_{r-i} = \sum_{i=1}^{L} c_i \sum_{k=1}^{L'} b_k S_{r-i-k} \tag{18}$$

donde la expansión para S_{r-i} es válida porque $r-i$ corre de $r-1$ hasta $r-L$ lo cual está en el rango $L'+1, \ldots, r-1$ otra vez porque $r \geq L+L'+1$. Los sumandos de la derecha en las ecuaciones (17) y (18) pueden intercambiarse de modo que coincidan, con lo cual se obtiene que $S_r \neq S_r$ y ésto es una contradicción. □

Como consecuencia inmediata del lema anterior se tiene el siguiente,

Corolario 2.2.2. *Bajo las hipótesis del Lema 2.2.1 si se cumple la igualdad,*

$$L_r = \text{máx}\{L_{r-1}, r - L_{r-1}\}$$

entonces $(L_r, \Lambda^{(r)}(x))$ *es de longitud mínima.*

Debido a este corolario, para que en el r-ésimo paso la longitud L_r sea mínima puede tomar los siguientes dos valores, L_{r-1} o bien $r - L_{r-1}$. Por otro lado, ya se vió que la forma adecuada de tomar $\Lambda^{(r)}(x)$ es

$$\Lambda^{(r)}(x) = \Lambda^{(r-1)}(x) - \Delta_m^{-1} \Delta_r x^{r-m} \Lambda^{(m-1)} \tag{19}$$

donde m corresponde a la más reciente iteración para la cual $L_m > L_{m-1}$. Debido al Lema 2.2.1, si $\Lambda_r \neq 0$, la condición $2L_{r-1} \leq r-1$ es necesaria y suficiente para que haya un cambio en la longitud. Esto permite escribir el proceso en la siguiente forma.

Comenzando con $\Lambda^{(0)}(x) = 1$, $B^{(0)}(x) = 1$, $L_0 = 0$, en el paso r, $r = 1, \ldots, 2t$ tómese:

$$L_r = \delta_r (r - L_{r-1}) + (1 - \delta_r) L_{r-1}$$

$$\begin{bmatrix} \Lambda^{(r)}(x) \\ B^{(r)}(x) \end{bmatrix} = \begin{bmatrix} 1 & -\Delta_r x \\ \Delta_r^{-1} \delta_r & (1-\delta_r)x \end{bmatrix} \begin{bmatrix} \Lambda^{(r-1)}(x) \\ B^{(r-1)}(x) \end{bmatrix}$$

donde $\delta_r = \begin{cases} 1 & \text{si } \Delta_r \neq 0 \text{ y } 2L_{r-1} \leq r-1 \\ 0 & \text{en otro caso} \end{cases}$.

El polinomio $B^{(r)}(x)$ es un auxiliar que permite calcular $\Lambda^{(r)}(x)$ como está dado en (19) pues al ser m la más reciente iteración donde ocurrió un cambio de longitud, resulta que $B^{(m)}(x) = \Delta_m^{-1} \Lambda^{(m-1)}(x)$ y en las iteraciones correspondientes a $i = m, \ldots, r-1$ se toma $B^{(i)}(x) = xB^{(i-1)}(x)$.

Es importante notar que en el desarrollo anterior aparece el término Δ_r^{-1} lo cual puede no tener sentido si Δ_r es cero, pero ésto ocurre sólo cuando $\delta_r = 0$ y en este caso el término $\Delta_r^{-1}\delta_r$ es considerado igual a cero.

A continuación se muestra que el proceso descrito, en efecto, arroja el resultado deseado. Esta demostración fué dada por J.L. Massey en [**17**] poco después de que E.R. Berlekamp presentara su algoritmo.

Teorema 2.2.3 (Algoritmo de Berlekamp-Massey)**.** *Sea $\mathbb{F}$ un campo y S_1, S_2, ..., S_n elementos de $\mathbb{F}$ dados. Bajo las condiciones iniciales $\Lambda^{(0)}(x) = 1$, $B^{(0)}(x) = 1$ y $L_0 = 0$, para $r = 1, 2, \ldots, n$, úsese el siguiente conjunto de ecuaciones recursivas para calcular $\Lambda^{(n)}(x)$:*

1. $\Delta_r = \sum_{j=0}^{L_{r-1}} \Lambda_j^{(r-1)} S_{r-j}$
2. $L_r = \delta_r(r - L_{r-1}) + (1 - \delta_r)L_{r-1}$
3. $\begin{bmatrix} \Lambda^{(r)}(x) \\ B^{(r)}(x) \end{bmatrix} = \begin{bmatrix} 1 & -\Delta_r x \\ \Delta_r^{-1}\delta_r & (1-\delta_r)x \end{bmatrix} \begin{bmatrix} \Lambda^{(r-1)}(x) \\ B^{(r-1)}(x) \end{bmatrix}$

Donde $\delta_r = 1$ si $\Delta_r \neq 0$ y $2L_{r-1} \leq r - 1$; y $\delta_r = 0$ en otro caso. Entonces, $(L_n, \Lambda^{(n)}(x))$ es una recurrencia de longitud mínima que genera la sucesión S_1, S_2, ..., S_n, es decir, $\Lambda_0^{(n)} = 1$ y

$$S_r + \sum_{j=1}^{L_{r-1}} \Lambda_j^{(n)} S_{r-j} = 0 \quad \textit{para} \quad r = L_n + 1, \ldots, n$$

Demostración. Si se encuentra una recurrencia que genere la sucesión deseada y cuya longitud satisfaga la igualdad del Corolario 2.2.2, entonces debe ser de longitud mínima. La demostración será por inducción: veremos que la construcción dada por el algoritmo para la r-ésima recurrencia satisface la igualdad de dicho corolario suponiendo que se han construido iterativamente del mismo modo recurrencias tales para todo $k \leq r - 1$.

Para $k = 1, \ldots, r - 1$, sea $(L_k, \Lambda^{(k)}(x))$ una recurrencia de longitud mínima que genera $S_1, \ldots, S_k$. Supongamos por la hipótesis de inducción que $L_k = \text{máx}\{L_{k-1}, k - L_{k-1}\}$ siempre que $\Lambda^{(k)} \neq \Lambda^{(k-1)}$. Claramente esto es cierto para $k = 1$ porque si $\Lambda^{(1)} \neq \Lambda^{(0)}$ se tiene que $L_0 = 0$ y $L_1 = 1$.

Sea m el valor de k en la iteración más reciente que requirió un cambio en la longitud. Es decir, al final de la iteración $r - 1$, m es el entero tal que

$$L_{r-1} = L_m > L_{m-1}$$

Entonces,

$$\begin{aligned} S_j + \sum_{i=1}^{L_{r-1}} \Lambda_i^{(r-1)} S_{j-i} &= \sum_{i=0}^{L_{r-1}} \Lambda_i^{(r-1)} S_{j-i} \\ &= \begin{cases} 0 & \text{si } j = L_{r-1}+1, \ldots, r-1 \\ \Delta_r & \text{si } j = r \end{cases} \end{aligned} \tag{20}$$

Si $\Delta_r = 0$, la recurrencia $(L_{r-1}, \Lambda^{(r-1)}(x))$ también genera los primeros r valores y en este caso, $L_r = L_{r-1}$ y $\Lambda^{(r)}(x) = \Lambda^{(r-1)}(x)$.

Si $\Delta_r \neq 0$, se construye una nueva recurrencia. Recuérdese que el último cambio en la longitud ocurrió en $k = m$. Entonces

$$S_j + \sum_{i=1}^{L_{m-1}} \Lambda_i^{(m-1)} S_{j-i} = \begin{cases} 0 & \text{si } j = L_{m-1}+1, \ldots, m-1 \\ \Delta_m \neq 0 & \text{si } j = m \end{cases} \tag{21}$$

y por la hipótesis de inducción,

$$\begin{aligned} L_{r-1} = L_m &= \text{máx}\{L_{m-1}, m - L_{m-1}\} \\ &= m - L_{m-1} \end{aligned}$$

Por construcción, el nuevo polinomio es

$$\Lambda^{(r)}(x) = \Lambda^{(r-1)}(x) - \Delta_r \Delta_m^{-1} x^{r-m} \Lambda^{(m-1)}(x)$$

Sea $L_r = \text{gr}(\Lambda^{(r)}(x))$, entonces como

$$\text{gr}(\Lambda^{(r-1)}(x)) \leq L_{r-1} \quad \text{y} \quad \text{gr}(x^{r-m}\Lambda^{(m-1)}(x)) \leq r - m + L_{m-1}$$

se cumple que

$$\begin{aligned} \text{gr}(\Lambda^{(r)}(x)) &\leq \text{máx}\{L_{r-1}, r - m + L_{m-1}\} \\ &\leq \text{máx}\{L_{r-1}, r - L_{r-1}\} \end{aligned}$$

Esto significa que $\Lambda^{(r)}(x)$ es un polinomio válido para una recurrencia de longitud L_r, donde

$$L_r = \text{máx}[L_{r-1}, r - L_{r-1}]$$

Veamos que $(L_r, \Lambda^{(r)}(x))$ genera la sucesión deseada. Para ésto, calculamos la discrepancia correspondiente usando (20) y (21),

$$\begin{aligned} S_j - (-\sum_{i=1}^{L_r} \Lambda_i^{(r)} S_{j-i}) &= S_j + \sum_{i=1}^{L_{r-1}} \Lambda_i^{(r-1)} S_{j-i} \\ &- \Delta_r \Delta_m^{-1} [S_{j-r+m} + \sum_{i=1}^{L_{m-1}} S_{j-r+m-i}] \\ &= \begin{cases} 0 & \text{si } j = L_r + 1, \ldots, \\ & \qquad r-1 \\ \Delta_r - \Delta_r \Delta_m^{-1} \Delta_m = 0 & \text{si } j = r \end{cases} \end{aligned}$$

Por tanto, la nueva recurrencia $(L_r, \Lambda^{(r)}(x))$ genera $S_1, \ldots, S_r$ y como L_r satisface el Corolario 2.2.2, es de longitud mínima. En particular, se tiene que $(L_n, \Lambda^{(n)}(x))$ genera $S_1, \ldots, S_n$ y es de longitud mínima. □

Obsérvese que cuando se aplica el algoritmo a la sucesión de síndromes S_1, S_2, $\ldots$, S_{2t}, la actualización de la matriz requiere a lo más $2t$ multiplicaciones por iteración, y el cálculo de Δ_r no sobrepasa las t multiplicaciones por iteración. Como hay $2t$ pasos, se tienen a lo más $6t^2$ multiplicaciones. Por tanto el uso de este algoritmo en la resolución del sistema (9) es mejor que usar la inversión matricial que requiere del orden de t^3 operaciones.

2.2.1. Ejemplos

Ejemplo 2.2.4. Consideremos el $[15, 7, 5]$ código binario BCH dado en el Ejemplo 2.1.2. Supóngase que se recibe el polinomio:

$$r(x) = x^{14} + x^{13} + x^{12} + x^{11} + x^9 + x^8 + x^7 + x^3 + x^2 + 1$$

Usaremos el algoritmo de Berlekamp-Massey para encontrar el polinomio localizador de errores.

Los síndromes son:

$$S_1 = \alpha^6, \quad S_2 = \alpha^{12}, \quad S_3 = \alpha^{12}, \quad S_4 = \alpha^9, \quad S_5 = 0, \quad S_6 = \alpha^9$$

Veamos con detalle los dos primeros pasos del algoritmo siguiendo el Teorema 2.2.3. Los valores iniciales son:

$$\Lambda^{(0)}(x) = 1, \quad B^{(0)}(x) = 1 \quad \text{y} \quad L_0 = 0 \tag{22}$$

En el primer paso se toma $r = 1$ y se calcula la discrepancia:

$$\Delta_1 = S_1 = \alpha^6$$

Como $\Delta_1 \neq 0$ y además se cumple la condición $2L_0 \leq r - 1 = 0$ entonces $\delta_1 = 1$, por lo que hacemos

$$L_1 = r - L_0 = 1, \tag{23}$$

y usando los valores iniciales (22) calculamos

$$\begin{aligned} \Lambda^{(1)}(x) &= 6\Lambda^{(0)}(x) - \Delta_1 x B^{(0)}(x) \\ &= 1 + \alpha^6 x \end{aligned} \tag{24}$$

y

$$\begin{aligned} B^{(1)}(x) &= \Delta_1^{-1}\Lambda^{(0)}(x) \\ &= (\alpha^6)^{-1} \\ &= \alpha^9 \end{aligned} \tag{25}$$

En el siguiente paso se toma $r = 2$ y en seguida se calcula la segunda discrepancia

$$\begin{aligned} \Delta_2 &= S_2 + \alpha^6 S_1 \\ &= \alpha^{12} + \alpha^6\alpha^6 \\ &= 0 \end{aligned}$$

Como $\Delta_2 = 0$, en este caso $\delta_2 = 0$ y por tanto

$$L_2 = L_1 = 1 \tag{26}$$

y usando (24) y (25) calculamos

$$\Lambda^{(2)}(x) = \Lambda^{(1)}(x) = 1 + \alpha^6 x \tag{27}$$

y

$$\begin{aligned} B^{(2)}(x) &= xB^{(1)} \\ &= \alpha^9 x \end{aligned}$$

La Tabla 1 muestra los resultados completos para este ejemplo. Del último renglón de la tabla se desprende que el polinomio localizador de errores es $\Lambda(x) = 1+\alpha^6 x+\alpha^{14}x^2+x^3$, el cual coincide con el resultado obtenido en el Ejemplo 2.1.2.

r	Δ_r	$B^{(r)}(x)$	$\Lambda^{(r)}(x)$	L_r
0	$-$	1	1	0
1	α^6	α^9	$1+\alpha^6 x$	1
2	0	$\alpha^9 x$	$1+\alpha^6 x$	1
3	α^{10}	$\alpha^5+\alpha^{11}x$	$1+\alpha^6 x+\alpha^4 x^2$	2
4	0	$\alpha^5 x+\alpha^{11}x^2$	$1+\alpha^6 x+\alpha^4 x^2$	2
5	α^4	$\alpha^{11}+\alpha^2 x+x^2$	$1+\alpha^6 x+\alpha^{14}x^2+x^3$	3
6	0	$\alpha^{11}x+\alpha^2 x^2+x^3$	$1+\alpha^6 x+\alpha^{14}x^2+x^3$	3

Tabla 1. Resultados para el Ejemplo 2.2.4

EJEMPLO 2.2.5. Consideremos nuevamente el código Reed-Solomon de longitud $n = 10$ sobre $\mathbb{F}_{11}$ dado en el Ejemplo 2.1.3, el cual corrige $t = 4$ errores. Supóngase se envía la palabra cero pero se recibe el vector

$$\overline{r} = (0, 5, 0, 0, 0, 8, 0, 10, 3, 0)$$

Que corresponde al polinomio $r(x) = 3x^8 + 10x^7 + 8x^5 + 5x$. Buscamos el polinomio localizador de errores. Los síndromes son,

$$\begin{array}{ll} S_1 = r(\alpha) = 4 & S_5 = r(\alpha^5) = 2 \\ S_2 = r(\alpha^2) = 6 & S_6 = r(\alpha^6) = 3 \\ S_3 = r(\alpha^3) = 1 & S_7 = r(\alpha^7) = 4 \\ S_4 = r(\alpha^4) = 9 & S_8 = r(\alpha^8) = 7 \end{array}$$

Con el algoritmo de Berlekamp-Massey se obtienen los resultados que se muestran en la Tabla 2 cuyo último renglón muestra que el polinomio localizador de errores está dado por,

$$\begin{aligned} \Lambda(x) &= 1 + 7x^2 + 10x^3 + 2x^4 \\ &= (x-4)(x-8)(x-10)(x-6) \\ &= (1-3x)(1-7x)(1-10x)(1-2x) \end{aligned}$$

Por tanto las localizaciones de error son, $X_1 = 3 = \alpha^8$, $X_2 = 7 = \alpha^7$, $X_3 = 10 = \alpha^5$ y $X_4 = 2 = \alpha$, lo cual dice que los errores ocurrieron en las posiciones 8, 7, 5 y 1.

El algoritmo de Berlekamp-Massey permite encontrar el polinomio localizador de errores $\Lambda(x)$ pero una vez obtenido éste es necesario factorizarlo para así determinar las localizaciones de error. La forma usual de hacer esto es por medio de la llamada búsqueda de Chien que consiste en verificar uno a uno si α^j es raíz de $\Lambda(x)$. El algoritmo produce correctamente el polinomio localizador de errores siempre y cuando el número de errores ocurridos no supere la capacidad correctora, t, del

r	Δ_r	$B^{(r)}(x)$	$\Lambda^{(r)}(x)$	L_r
0	$-$	1	1	0
1	4	3	$1+7x$	1
2	1	$3x$	$1+4x$	1
3	3	$4+5x$	$1+4x+2x^2$	2
4	3	$4x+5x^2$	$1+3x+9x^2$	2
5	5	$9+5x+4x^2$	$1+3x+8x^3$	3
6	6	$9x+5x^2+4x^3$	$1+4x+3x^2+6x^3$	3
7	10	$10+7x+8x^2+5x^3$	$1+4x+x^2+4x^4$	4
8	7	$10x+7x^2+8x^3+5x^4$	$1+7x^2+10x^3+2x^4$	4

Tabla 2. Resultados para el Ejemplo 2.2.5

código. De no ser así, el algoritmo podría fallar produciendo un polinomio que no cumpla con los requerimientos de polinomio localizador de errores o bien produciendo un polinomio localizador legítimo pero incorrecto (y llevar así a una decodificación equivocada). El primer caso puede detectarse cuando el número de raíces distintas de $\Lambda(x)$ en $\mathbb{F}_{q^m}$ es diferente de L. Ambas posibilidades se muestran a continuación.

Ejemplo 2.2.6. Considérese nuevamente el $[15, 7, 5]$ código binario BCH de los Ejemplos 2.1.2 y 2.2.4. Supóngase que se envía la palabra $c(x) = x^{12}+x^{11}+x^9+x^8+x^7+x^2+1$, pero que durante la transmisión se induce el error $e(x) = x^{14} + x^{13} + x^3 + x^2$. De modo que se recibe $r(x) = x^{14} + x^{13} + x^{12} + x^{11} + x^9 + x^8 + x^7 + x^3 + 1$. Obsérvese que el error cometido excede la capacidad correctora del código que es $t = 3$. Con el algoritmo de Berlekamp-Massey se obtiene el polinomio

$$\begin{aligned}\Lambda(x) &= \alpha^{14}x^3 + \alpha^3 x + 1 \\ &= (1-\alpha^0 x)(1-\alpha^6 x)(1-\alpha^8 x)\end{aligned}$$

El cual lleva a la conclusión equivocada de que el error es $e(x) = x^8 + x^6 + 1$.

Supóngase ahora que el error ocurrido es $e(x) = x^{14}+x^{13}+x^3+x$, es decir, se recibe el polinomio $r(x) = x^{14} + x^{13} + x^{12} + x^{11} + x^9 + x^8 + x^7 + x^3 + x^2 + x + 1$. En este caso el algoritmo de Berlekamp-Massey da el polinomio

$$\Lambda(x) = \alpha^{11}x^3 + \alpha^4 x^2 + \alpha^{11}x + 1$$

que no tiene raíces en $\mathbb{F}_{16}$ lo que indica que ocurrieron más de 3 errores.

2.3. Cálculo de las magnitudes de los errores: el algoritmo de Forney

El algoritmo de Peterson, Gorenstein y Zierler descrito en la Sección 2.1 requiere la inversión de dos matrices lo cual representa un considerable trabajo computacional. La primera de estas inversiones es usada en la determinación del polinomio localizador de errores y puede evitarse usando el algoritmo de Berlekamp-Massey. La segunda es usada para determinar la magnitud de los errores (lo cual es necesario en códigos no binarios) y una forma de evitarla es por medio del algoritmo de Forney, el cual, se describirá a continuación.

Consideremos nuevamente el polinomio localizador de errores

$$\Lambda(x) = \Lambda_v x^v + \Lambda_{v-1} x^{v-1} + \cdots + \Lambda_1 x + 1 \tag{28}$$

que se definió como el polinomio cuyos ceros son las localizaciones de error inversas:

$$\Lambda(x) = (1 - X_1 x)(1 - X_2 x) \cdots (1 - X_v x)$$

Recuérdese que $X_l = \alpha^{i_l}$ y $Y_l = e_{i_l}$ son las localizaciones y las magnitudes de error respectivamente, donde i_l es la posición del l-ésimo error.

Definimos ahora el *polinomio de síndromes*,

$$S(x) = \sum_{j=1}^{2t} S_j x^{j-1} = \sum_{i=1}^{v} Y_i X_i^{b+j-1} x^{j-1} \tag{29}$$

y el polinomio *evaluador de errores* $\Omega(x)$, en la forma

$$\Omega(x) = S(x)\Lambda(x) \qquad \text{mod } x^{2t}$$

Este polinomio está relacionado las localizaciones y las magnitudes de error como sigue

$$\begin{aligned}
\Omega(x) &= [\sum_{j=1}^{2t} \sum_{i=1}^{v} Y_i X_i^{b+j-1} x^{j-1}][\prod_{l=1}^{v}(1 - X_l x)] \qquad \text{mod } x^{2t} \\
&= [\sum_{i=1}^{v} Y_i X_i^b \sum_{j=1}^{2t} (X_i x)^{j-1}][\prod_{l=1}^{v}(1 - X_l x)] \qquad \text{mod } x^{2t} \\
&= \sum_{i=1}^{v} Y_i X_i^b [(1 - X_i x) \sum_{j=1}^{2t} (X_i x)^{j-1}] \prod_{l \neq i}(1 - X_l x) \qquad \text{mod } x^{2t}
\end{aligned}$$

Pero el término entre corchetes es una factorización de $(1-(X_i x)^{2t})$ que módulo x^{2t} es 1, por lo que se obtiene

$$\Omega(x) = \sum_{i=1}^{v} Y_i X_i^b \prod_{l \neq i} (1 - X_l x) \tag{30}$$

Esta expresión permite calcular las magnitudes de error de una forma más simple que la inversión matricial antes descrita.

Teorema 2.3.1 (Algoritmo de Forney)**.** *Las magnitudes de error están dadas por*

$$Y_l = \frac{\Omega(X_l^{-1})}{X_l^b \prod_{j \neq l}(1 - X_j X_l^{-1})} = -\frac{\Omega(X_l^{-1})}{X_l^{b-1} \Lambda'(X_l^{-1})}$$

Demostración. Evaluando (30) en X_l^{-1} se obtiene

$$\Omega(X_l^{-1}) = Y_l X_l^b \prod_{j \neq l} (1 - X_j X_l^{-1})$$

y por otro lado, la derivada de $\Lambda(x)$ es

$$\Lambda'(x) = -\sum_{i=1}^{v} X_i \prod_{j \neq l} (1 - X_j x)$$

luego

$$\Lambda'(X_l^{-1}) = -X_l \prod_{j \neq l} (1 - X_j X_l^{-1})$$

de donde se sigue el resultado. □

Ejemplo 2.3.2. Considérese nuevamente el Ejemplo 2.2.5 donde se presenta un código sobre $\mathbb{F}_{11}$. Ya se obtuvieron el polinomio localizador de errores

$$\Lambda(x) = 1 + 7x^2 + 10x^3 + 2x^4$$

y las correspondientes localizaciones de error $X_1 = \alpha^8 = 3$, $X_2 = \alpha^7 = 7$, $X_3 = \alpha^5 = 10$, $X_4 = \alpha = 2$. El polinomio de síndromes está dado por

$$S(x) = 4 + 6x + x^2 + 9x^3 + 2x^4 + 3x^5 + 4x^6 + 7x^7$$

por tanto,

$$\begin{aligned} \Omega(x) &= S(x)\Lambda(x) \mod x^8 \\ &= 4 + 6x + 7x^2 + 3x^3 + 7x^8 + 7x^9 + x^{10} + 3x^{11} \mod x^8 \\ &= 4 + 6x + 7x^2 + 3x^3 \end{aligned}$$

además, $\Lambda'(x) = 3x + 8x^2 + 8x^3$. Para encontrar las magnitudes de error usamos la relación dada en el Teorema 2.3.1 con $b = 1$, obteniendo,

$$Y_1 = -\frac{\Omega(X_1^{-1})}{\Lambda'(X_1^{-1})} = -\frac{\Omega(4)}{\Lambda'(4)} = -\frac{2}{3} = 3$$
$$Y_2 = -\frac{\Omega(X_2^{-1})}{\Lambda'(X_2^{-1})} = -\frac{\Omega(8)}{\Lambda'(8)} = -\frac{1}{1} = 10$$
$$Y_3 = -\frac{\Omega(X_3^{-1})}{\Lambda'(X_3^{-1})} = -\frac{\Omega(10)}{\Lambda'(10)} = -\frac{2}{8} = 8$$
$$Y_4 = -\frac{\Omega(X_4^{-1})}{\Lambda'(X_4^{-1})} = -\frac{\Omega(6)}{\Lambda'(6)} = -\frac{5}{10} = 5$$

Estos errores corresponden a las posiciones 8, 7, 5 y 1 respectivamente así que el polinomio de error es $e(x) = 3x^8 + 10x^7 + 8x^5 + 5x$. Por lo tanto, la palabra enviada es $c(x) = r(x) - e(x) = 0$.

Capítulo 3

Códigos BCH sobre $\mathbb{Z}_{p^s}$ y decodificación

En este capítulo se introducen los códigos cíclicos sobre el anillo $\mathbb{Z}_{p^s}$, el anillo de enteros módulo p^s, donde p es un primo y s un entero positivo. Para ello, se definen en la Sección 3.1 los anillos de Galois que son las extensiones de $\mathbb{Z}_{p^s}$ utilizadas en la construcción de estos códigos y se dan algunas de sus propiedades más importantes. La Sección 3.2 dá una construcción de códigos cíclicos y BHC sobre anillos $\mathbb{Z}_{p^s}$ manteniendo las ideas centrales de su definición en campos finitos. Finalmente, en la Sección 3.3 se presenta un algoritmo para decodificar códigos BCH sobre $\mathbb{Z}_{p^s}$ basado principalmente en una extensión del algoritmo de Berlekamp-Massey en anillos de Galois.

3.1. Introducción a los anillos de Galois

Las ideas presentadas a continuación pueden verse en forma más detallada en [**2**], [**15**] y [**24**].

Sea p un número primo, s un entero positivo y $\mathbb{Z}_{p^s}$ el anillo de enteros módulo p^s. Claramente los divisores de cero de este anillo son los múltiplos de p. Además $\mathbb{Z}_{p^s}$ es un anillo local con ideal maximal $\langle p \rangle$ y campo residual $\mathbb{Z}_{p^s}/\langle p \rangle \simeq \mathbb{F}_p$.

Considérese el mapeo reducción dado por

$$\begin{aligned} \mu : \mathbb{Z}_{p^s} &\rightarrow \mathbb{F}_p \\ m &\rightarrow \mu(m) = m \mod p \end{aligned} \tag{31}$$

El núcleo de este homomorfismo es el ideal $\langle p \rangle$. El mapeo reducción puede extenderse a un mapeo entre los anillos de polinomios $\mathbb{Z}_{p^s}[x]$ y $\mathbb{F}_p[x]$ en la forma

$$\begin{aligned} \mu : \mathbb{Z}_{p^s}[x] &\rightarrow \mathbb{F}_p[x] \\ a_0 + a_1 x + \cdots + a_n x^n &\rightarrow \mu(a_0) + \mu(a_1)x + \cdots + \mu(a_n)x^n \end{aligned} \tag{32}$$

Es fácil ver que esta extensión es un homomorfismo de $\mathbb{Z}_{p^s}[x]$ sobre $\mathbb{F}_p[x]$ cuyo núcleo es el ideal $\langle p\rangle$ que en este contexto está dado por $\langle p\rangle = p\mathbb{Z}_{p^s}[x]$

Sea $f(x) \in \mathbb{Z}_{p^s}[x]$ un polinomio mónico de grado $m \geq 1$. Si su reducción, $\mu(f)(x)$, es irreducible sobre $\mathbb{F}_p[x]$, $f(x)$ es llamado *polinomio básico irreducible* en $\mathbb{Z}_{p^s}[x]$.

Es sabido que para un entero $m \geq 1$ existe un polinomio mónico básico irreducible de grado m sobre $\mathbb{Z}_{p^s}$ y que divide a $x^{p^m-1} - 1$ en $\mathbb{Z}_{p^s}[x]$ (Teorema 13.9, [**24**]).

Estamos ahora en condiciones de dar la siguiente definición.

Definición 3.1.1. *Sea $h(y) \in \mathbb{Z}_{p^s}[y]$ un polinomio mónico básico irreducible de grado m. El anillo de Galois se define como*

$$GR(p^s, m) = \mathbb{Z}_{p^s}[y]/\langle h(y)\rangle$$

Obsérvese que $GR(p, m) \simeq \mathbb{F}_{p^m}$ y $GR(p^s, 1) \simeq \mathbb{Z}_{p^s}$.

El anillo de Galois $GR(p^s, m)$ es un anillo local con ideal maximal $\langle p\rangle = pGR(p^s, m)$ que nuevamente consta de los divisores de cero y en este caso el campo residual es $GR(p^s, m)/\langle p\rangle \simeq \mathbb{F}_{p^m}$.

Además, $GR(p^s, m)$ es un anillo de cadena cuyos ideales son

$$\{0\} = \langle p^s\rangle \subset \langle p^{s-1}\rangle \subset \cdots \subset \langle p^2\rangle \subset \langle p\rangle \subset \langle p^0\rangle = GR(p^s, m).$$

Ejemplo 3.1.2. En $\mathbb{Z}_4[y]$ el polinomio $h(y) = y^2 + y + 1$ es mónico básico irreducible y por lo tanto el anillo de Galois $GR(2^2, 2)$ está dado por el anillo de clases residuales $\mathbb{Z}_4[y]/\langle y^2 + y + 1\rangle$. Es decir,

$$GR(2^2, 2) = \{a_1 y + a_0 \mid a_1, a_0 \in \mathbb{Z}_4\}$$

donde las operaciones se efectúan módulo el polinomio $y^2 + y + 1$. Además,

$$|GR(2^2, 2)| = (2^2)^2 = 16.$$

En general, si $h(y)$ es un polinomio mónico básico irreducible en $\mathbb{Z}_{p^s}[y]$ de grado m, el anillo de Galois $GR(p^s, m)$ está dado por

$$GR(p^s, m) = \{a_{m-1}y^{m-1} + \cdots + a_1 y + a_0 \mid a_{m-1}, \ldots, a_1, a_0 \in \mathbb{Z}_p^s\}$$

con las operaciones módulo $h(y)$ y su cardinalidad es

$$|GR(p^s, m)| = (p^s)^m = p^{sm}.$$

El siguiente resultado será de vital importancia en la sección 3.3.1.

Teorema 3.1.3. *Sea $r \in GR(p^s, m)$ distinto de cero. Entonces r puede escribirse como $r = up^t$ con u unidad y $0 \leq t \leq s-1$, donde el entero t es único y u es único módulo $\langle p^{s-t} \rangle$.*

Demostración. Si r es unidad, $t = 0$ satisface el teorema. Supongamos pues que r no es unidad, entonces debe ser divisor de cero y por tanto un múltiplo de p. Sea t la máxima potencia de p que divide a r, entonces podemos escribir $r = up^t$ para una unidad u. Claramente esta elección de t es única y satisface $0 < t \leq s-1$. Ahora, supongamos que $r = u_1 p^t$ con u_1 unidad. Tenemos que $up^t - u_1 p^t = 0$ en $\mathbb{Z}_{p^s}[y]$ pero esto significa que los coeficientes de $up^t - u_1 p^t$ son todos cero en $\mathbb{Z}_{p^s}$ de modo que $up^t - u_1 p^t = q(y)p^s$ para algún polinomio $q(y) \in \mathbb{Z}_{p^s}[y]$. Por lo tanto,

$$[u - u_1 - q(y)p^{s-t}]p^t = 0 \tag{33}$$

Si $u - u_1 - q(y)p^{s-t} = 0$, entonces u y u_1 son congruentes módulo $\langle p^{s-t} \rangle$ como se quería. Si $u - u_1 - q(y)p^{s-t} \neq 0$, definimos k como la mayor potencia de p que divide a $u - u_1 - q(y)p^{s-t}$, entonces podemos escribir

$$u - u_1 - q(y)p^{s-t} = \theta p^k \quad \text{con } \theta \text{ unidad.}$$

Así que de (33), $\theta p^k p^t = 0$ pero como $t < s$ y θ es unidad, debe cumplirse que

$$k \geq s - t$$

Sea $n = k - s + t$, entonces

$$u - u_1 - q(y)p^{s-t} = \theta p^{s-t+n}$$

es decir,

$$u - u_1 = [q(y) + \theta p^n]p^{s-t}$$

como se quería.

□

Se tiene el siguiente resultado que describe completamente el grupo de unidades de $GR(p^s, m)$ (Teorema 14.11, [**24**]).

Teorema 3.1.4. *Sea $R = GR(p^s, m)$ y R^* su grupo de unidades. Entonces,*

$$R^* = G_1 \times G_2$$

donde G_1 es un grupo cíclico de orden $p^m - 1$ y G_2 es un grupo de orden $p^{(s-1)m}$ tal que

1. *Si p es impar o si $p = 2$ y $s \leq 2$, entonces G_2 es un producto directo de m grupos cíclicos cada uno de orden p^{s-1}.*
2. *Si $p = 2$ y $s \geq 3$, entonces G_2 es un producto directo de un grupo cíclico de orden 2, un grupo cíclico de orden 2^{s-2} y $m-1$ grupos cíclicos cada uno de orden 2^{s-1}.*

3.2. Códigos cíclicos y BCH sobre $\mathbb{Z}_{p^s}$

En esta sección se presenta una forma de construir códigos cíclicos y, en particular códigos BCH, utilizando las ideas centrales de la construcción de este tipo de códigos sobre campos finitos.

Definición 3.2.1. *Un $\mathbb{Z}_{p^s}$-código lineal de longitud n es un $\mathbb{Z}_{p^s}$-submódulo $\mathcal{C}$ de $\mathbb{Z}_{p^s}^n$. Se dice que $\mathcal{C}$ es cíclico si siempre que $(c_0, c_1, \ldots, c_{n-1}) \in \mathcal{C}$ también su corrimiento $(c_{n-1}, c_0, \ldots, c_{n-2})$ está en $\mathcal{C}$.*

La distancia de Hamming y la distancia mínima para un código sobre $\mathbb{Z}_{p^s}$ se definen en la misma forma que para códigos sobre campos finitos. Además, la demostración del teorema 1.1.7 se basa en la linealidad del código y por tanto el resultado es extensivo a códigos sobre $\mathbb{Z}_{p^s}$, es decir, un $\mathbb{Z}_{p^s}$-código lineal con distancia mínima d puede corregir por lo menos $\lfloor \frac{d-1}{2} \rfloor$ errores.

Ejemplo 3.2.2. En $\mathbb{Z}_4$ un código cíclico de longitud $n = 3$ es

$$\begin{aligned}\mathcal{C} = \{&(002), (020), (200), (110), (101), (011), (220), (202), (022),\\ &(330), (303), (033), (013), (130), (301), (031), (310), (103),\\ &(112), (121), (211), (332), (323), (233), (123), (231), (312),\\ &(132), (321), (213), (222), (000)\}\end{aligned}$$

En esta sección $\mathcal{R}_n$ denotará el anillo $\mathbb{Z}_{p^s}[x]/\langle x^n - 1\rangle$ y sus elementos serán representados como polinomios con coeficientes en $\mathbb{Z}_{p^s}$ de grado menor que n. Además, identificaremos cada polinomio $c_0 + c_1 x + \cdots + c_{n-1}x^{n-1} \in \mathcal{R}_n$ con el vector $(c_0, c_1, \ldots, c_{n-1}) \in \mathbb{Z}_{p^s}^n$. Con estas ideas puede verse, en forma análoga al caso de campos finitos (Proposición 1.2.4), que los códigos cíclicos de longitud n sobre $\mathbb{Z}_{p^s}$ corresponden a los ideales de $\mathcal{R}_n$. El Lema 1.2.5 implica que si $g(x) \in \mathbb{Z}_{p^s}[x]$ es un

polinomio mónico que divide a x^n-1, entonces genera un código cíclico $\mathcal{C}$ sobre $\mathbb{Z}_{p^s}$, $g(x)$ es conocido como el polinomio generador del código $\mathcal{C}$. Sin embargo, Kanwar y López-Permouth muestran en [**12**] que, aún cuando los ideales de $\mathcal{R}_n$ son principales, no todos son generados por divisores de x^n-1. En este trabajo se consideran solamente códigos cíclicos cuyo polinomio generador divide a x^n-1.

La construcción de códigos cíclicos de longitud n sobre $\mathbb{Z}_{p^s}$ dada por P. Shankar en [**22**] se centra en la factorización de x^n-1 sobre el grupo de unidades del anillo de Galois $GR(p^s,m)$, donde $(n,p)=1$ y $n \mid p^m-1$. Para ver como se lleva a cabo ésto daremos algunos resultados que permiten extender las ideas utilizadas en la construcción de códigos cíclicos sobre campos finitos a este tipo de anillos.

Sea $\mathcal{R}=GR(p^s,m)$ y $\mathcal{R}^*$ su grupo de unidades. Como $\mathcal{R}^*$ es un grupo multiplicativo conmutativo, puede descomponerse como producto de grupos cíclicos. El subgrupo que nos interesa es un grupo cíclico cuyos elementos sean todas las raíces de x^n-1, el cual se denotará por G_n. Una vez identificado este grupo, construir códigos cíclicos se reduce a elegir algunos elementos de este grupo para que sean raíces del polinomio generador, $g(x)$, de modo que éste resulte divisor de x^n-1.

El Teorema 3.1.4 garantiza la existencia de un subgrupo cíclico G_1 de orden p^m-1 (este orden es primo relativo con p) del grupo de unidades $\mathcal{R}^*$. Entonces si $n \mid p^m-1$, debe haber un subgrupo cíclico de orden n de G_1 y por lo tanto de $\mathcal{R}^*$. Los siguientes resultados, dados por Shankar en [**22**], servirán en la construcción de los códigos cíclicos antes mencionada. En su demostración se requerirá un resultado que es consecuencia directa del Teorema XV.1 dado en [**15**].

Lema 3.2.3. *Sea $f(x)$ un polinomio que no es divisor de cero en $\mathcal{R}[x]$ y supóngase que su reducción $\mu(f)$ tiene una raíz simple β en $\mathbb{F}_{p^m}$. Entonces f tiene una y sólo una raíz $\alpha \in \mathcal{R}$ tal que $\mu(\alpha)=\beta$.*

Teorema 3.2.4. *Supóngase que α genera un subgrupo de orden n en $\mathcal{R}^*$, con $(n,p)=1$. Entonces el polinomio x^n-1 puede factorizarse como*

$$x^n-1=(x-\alpha)(x-\alpha^2)\cdots(x-\alpha^n)$$

si y sólo si $\mu(\alpha)$ tiene orden n en $\mathbb{F}_{p^m}^*$.

Demostración. Supóngase que x^n-1 se factoriza como

$$x^n-1=(x-\alpha)(x-\alpha^2)\cdots(x-\alpha^n)$$

entonces se satisface la relación,

$$x^n - 1 = (x - \mu(\alpha))(x - \mu(\alpha)^2) \cdots (x - \mu(\alpha)^n) \quad \text{en } \mathbb{F}_{p^m}$$

pero como $(n, p) = 1$, el polinomio $x^n - 1$ no tiene raíces múltiples sobre $\mathbb{F}_{p^m}$, lo cual implica que $\mu(\alpha)$ tiene orden n.

Supóngase ahora que $\mu(\alpha)$ es de orden n en $\mathbb{F}_{p^m}^*$, entonces podemos escribir

$$x^n - 1 = (x - \mu(\alpha))(x - \mu(\alpha)^2) \cdots (x - \mu(\alpha)^n)$$

en $\mathbb{F}_{p^m}$.

Sea $\overline{F} = \{\mu(\alpha), \mu(\alpha)^2, \ldots, \mu(\alpha)^n\}$. Como $x^n - 1$ no tiene raíces múltiples en $\mathbb{F}_{p^m}$ entonces por el Lema 3.2.3, a cada $\mu(\alpha)^i$ le corresponde un único elemento, digamos $\alpha_i \in \mathcal{R}^*$ tal que

$$\mu(\alpha_i) = \mu(\alpha)^i$$

y además,

$$x^n - 1 = (x - \alpha_1)(x - \alpha_2) \cdots (x - \alpha_n) \tag{34}$$

en $\mathcal{R}[x]$.

Por otro lado, como α es de orden n en $\mathcal{R}^*$, entonces los elementos α,α^2,$\cdots$,α^n son n raíces distintas de $x^n - 1$. Pero el conjunto$F = \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ consta de todas y cada una de las raíces de $x^n - 1$ sobre $\mathcal{R}^*$. En efecto, si hubiera una raíz de $x^n - 1$, digamos β, que no está en F entonces $\mu(\beta)$ sería raíz de $x^n - 1$ sobre $\mathbb{F}_{p^m}$ y por tanto $\mu(\beta) \in \overline{F}$, es decir, se tendría que

$$\mu(\beta) = \mu(\alpha)^i \quad \text{para algún } i \in \{1, 2, \ldots, n\}$$

y como $\beta \neq \alpha_i$ para todo i se contradice la unicidad del Lema 3.2.3. Por lo tanto, $F = \{\alpha, \alpha^2, \ldots, \alpha^n\}$ y de (34) se obtiene la factorización deseada

$$x^n - 1 = (x - \alpha)(x - \alpha^2) \cdots (x - \alpha^n).$$

□

Una consecuencia inmediata del teorema anterior es el siguiente resultado.

Corolario 3.2.5. *Sea α un elemento de orden n en $\mathcal{R}^*$, donde $(n, p) = 1$. Entonces un polinomio $k(x)$ con coeficientes en $\mathbb{Z}_{p^s}$ que divide a $x^n - 1$ puede factorizarse sobre G_n como*

$$k(x) = (x - \alpha^{i_1})(x - \alpha^{i_2}) \cdots (x - \alpha^{i_k})$$

si y sólo si $\mu(k(x))$ *puede factorizarse sobre* $\mathbb{F}_{p^m}^*$ *como*

$$\mu(k(x)) = (x - \mu(\alpha)^{i_1})(x - \mu(\alpha)^{i_2}) \cdots (x - \mu(\alpha)^{i_k}).$$

El siguiente resultado es útil para determinar un generador del subgrupo G_n.

Lema 3.2.6. *Sea* $\alpha \in \mathcal{R}$ *tal que* $\mu(\alpha)$ *genera un subgrupo cíclico de orden* n *en* $\mathbb{F}_{p^m}^*$. *Entonces* α *genera un subgrupo cíclico de orden* nd *en* $\mathcal{R}^*$, *donde* d *es un entero mayor o igual que* 1 *y* α^d *genera el subgrupo cíclico* G_n *en* $\mathcal{R}^*$

Demostración. Es claro que α debe ser una unidad de $\mathcal{R}$. Sea k el orden de α, entonces $\alpha^k = 1$ en $\mathcal{R}^*$ y por tanto, $\mu(\alpha)^k = 1$ en $\mathbb{F}_{p^m}$. Así que el orden de $\mu(\alpha)$, que es n, divide a k lo cual significa que α es de orden $k = nd$ para algún entero $d \geq 1$. Más aún, el orden de α^d es $nd/(nd, d) = n$ y por tanto, α^d genera G_n como se quería. □

El siguiente resultado será útil más adelante.

Proposición 3.2.7. *Sea* α *un generador de* G_n. *Entonces el elemento* $\alpha^{l_1} - \alpha^{l_2}$ *es una unidad en* $\mathcal{R}$ *si* $0 \leq l_1, l_2 \leq n - 1$ *y* $l_1 \neq l_2$

Demostración. Supóngase que $\alpha^{l_1} - \alpha^{l_2}$ es un divisor de cero en $\mathcal{R}$, entonces

$$\alpha^{l_1} - \alpha^{l_2} = p \cdot f(y)$$

donde $f(y) \in \mathcal{R}$. Por lo tanto,

$$\mu(\alpha^{l_1} - \alpha^{l_2}) = 0$$

Pero esto implica que $\mu(\alpha)^{l_1} = \mu(\alpha)^{l_2}$, lo cual no es posible cuando $l_1 \neq l_2$ y $0 \leq l_1, l_2 \leq n - 1$ pues del Teorema 3.2.4, $\mu(\alpha)$ es de orden n en $\mathbb{F}_{q^m}$. Por tanto, $\alpha^{l_1} - \alpha^{l_2}$ es una unidad en $\mathcal{R}$. □

Los resultados anteriores permiten dar una factorización de $x^n - 1$ sobre $\mathbb{Z}_{p^s}$. Para ver ésto primero damos la siguiente:

Definición 3.2.8. *Sea* α *un generador de* G_n. *Al polinomio mónico de menor grado con coeficientes en* $\mathbb{Z}_{p^s}$ *que se anula en* α^i *lo llamaremos el polinomio mínimo de* α^i *sobre* $\mathbb{Z}_{p^s}$ *y será denotado por* $M_i(x)$.

Este polinomio es el análogo al polinomio mínimo o irreducible $m_i(x)$ sobre un campo $\mathbb{F}_q$ mencionado el la Sección 1.2.1.

Del Colorario 3.2.5 y de la expresión (3) para el polinomio mínimo $m_i(x)$ sobre un campo se sigue que

$$M_i(x) = (x - \alpha^i)(x - \alpha^{ip})(x - \alpha^{ip^2}) \cdots (x - \alpha^{ip^{m_i - 1}}) \tag{35}$$

donde m_i es el menor entero positivo tal que $ip^{m_i} \equiv i \mod n$.

Los exponentes que aparecen en las raíces del polinomio $M_i(x)$ son los elementos de C_i, la clase ciclotómica para p módulo n que contiene al entero i (ver página 21). Es decir,

$$M_i(x) = \prod_{j \in C_i} (x - \alpha^j) \tag{36}$$

De lo anterior se sigue que podemos factorizar a $x^n - 1$ sobre $\mathbb{Z}_{p^s}$ en la forma:

$$x^n - 1 = \prod_j M_i(x)$$

donde j corre sobre un conjunto de representantes de las clases ciclotómicas para p módulo n.

EJEMPLO 3.2.9. Factorizaremos $x^8 - 1$ sobre $\mathbb{Z}_9$. Para ello trabajamos en un anillo de Galois adecuado. Como $8 \mid 3^2 - 1$, entonces el grupo de unidades del anillo de Galois $GR(3^2, 2) = \mathbb{Z}_9[y]/\langle y^2 + y + 2\rangle$ tiene un subgrupo cíclico de orden 8, denotado por G_8. Trabajaremos, pues, en el anillo $\mathcal{R} = \mathbb{Z}_9[x]/\langle y^2 + y + 2\rangle$. Es necesario obtener un generador del grupo cíclico G_8, para ello, usamos el Lema 3.2.6.

Sea $\beta = y \in \mathcal{R}$. Como el polinomio $y^2 + y + 2$ es primitivo sobre $\mathbb{F}_3$ ([**21**], pág. 463), entonces $\mu(\beta) = y$ tiene orden 8 en $\mathbb{F}_{3^2}^*$. Por otro lado, puede verificarse que β tiene orden 24 en $\mathcal{R}^*$, por lo tanto, del Lema 3.2.6, un generador de G_8 es

$$\alpha = \beta^3 = 8y + 2$$

Además, las distintas clases ciclotómicas para 3 módulo 8 son

$$\begin{aligned} C_0 &= \{0\} \\ C_1 &= \{1, 3\} \\ C_2 &= \{2, 6\} \\ C_4 &= \{4\} \\ C_5 &= \{5, 7\} \end{aligned}$$

y de (36), los polinomios mínimos sobre $\mathbb{Z}_9$ son

$$\begin{aligned} M_0(x) &= x - \alpha^0 = x + 8 \\ M_1(x) = M_3(x) &= (x - \alpha)(x - \alpha^3) = x^2 + 4x + 8 \\ M_2(x) = M_6(x) &= (x - \alpha^2)(x - \alpha^6) = x^2 + 1 \\ M_4(x) &= (x - \alpha^4) = x + 1 \\ M_5(x) = M_7(x) &= (x - \alpha^5)(x - \alpha^7) = x^2 + 5x + 8 \end{aligned}$$

Por lo tanto, en $\mathbb{Z}_9[x]$ se tiene

$$x^8 - 1 = (x+1)(x+8)(x^2+1)(x^2+4x+8)(x^2+5x+8)$$

El desarrollo anterior muestra que la construcción de un código cíclico de longitud n sobre $\mathbb{Z}_{p^s}$, donde $n \mid p^m - 1$, cuyo polinomio generador es un divisor de $x^n - 1$ en $\mathbb{Z}_{p^s}[x]$ es completamente análoga a la construcción vista sobre campos finitos por lo que también puede darse en términos de las raíces en G_n de su polinomio generador $g(x) \in \mathbb{Z}_{p^s}[x]$, en la forma siguiente. Sea α un elemento primitivo de G_n y

$$T = \{i \mid g(\alpha^i) = 0\} = \{i_1, i_2, \ldots, i_j\}.$$

A T se le llama *conjunto de definición* del código cíclico generado por $g(x)$ y el polinomio generador del código está dado por

$$g(x) = \prod_{i \in T}(x - \alpha^i) = LCM(M_{i_1}(x), M_{i_2}(x), \ldots, M_{i_j}(x))$$

donde $M_{i_k}(x)$ es el polinomio mínimo de α^{i_k} y el código correspondiente $\mathcal{C}$ está formado por las palabras $c(x)$ que satisfacen,

$$c(x) = q(x)g(x) \mod x^n - 1$$

donde $q(x) \in \mathbb{Z}_{p^s}[x]$.

Aún más, el polinomio

$$\mu(g(x)) = \prod_{i \in T}(x - \mu(\alpha^i)) = LCM(m_{i_1}(x), m_{i_2}(x), \ldots, m_{i_j}(x))$$

genera un código cíclico sobre $\mathbb{F}_p$, donde $m_{i_k}(x)$ es el polinomio mínimo de $\mu(\alpha^{i_k})$ sobre $\mathbb{F}_p$.

Ejemplo 3.2.10. Daremos algunos códigos cíclicos de longitud $n = 8$ sobre $\mathbb{Z}_9$. En el Ejemplo 3.2.9 se vió que $\alpha = 8y + 2$ genera un grupo cíclico de orden 8 en el grupo de unidades del anillo de Galois $GR(3^2, 2) = \mathbb{Z}_9[y]/\langle y^2+y+2\rangle$ y se obtuvieron los 5 distintos polinomios mínimos sobre $\mathbb{Z}_9$.

Por tanto, con la construcción descrita anteriormente pueden darse $2^5 = 32$ códigos cíclicos de longitud $n = 8$ sobre $\mathbb{Z}_9$. En la Tabla 1 se muestra el polinomio generador $g(x)$ y el conjunto de definición T respecto a la raíz octava primitiva α para algunos de estos códigos. Recuérdese que $M_i(x)$ es el polinomio mínimo de α^i sobre $\mathbb{Z}_9$.

Para el caso de códigos BCH se tiene la siguiente definición.

	$g(x)$	T
$\mathcal{C}_1$	$M_0(x)M_1(x)M_2(x)M_4(x)M_5(x) = x^8 + 8$	$\{0,1,2,3,4,5,6,7\}$
$\mathcal{C}_2$	$M_0(x)M_1(x)M_2(x)M_4(x) = x^6 + 4x^5 + 8x^4 + 8x^2 + 5x + 1$	$\{0,1,2,3,4,6\}$
$\mathcal{C}_3$	$M_0(x)M_2(x)M_4(x)M_5(x) = x^6 + 5x^5 + 8x^4 + 8x^2 + 4x + 1$	$\{0,2,4,5,6,7\}$
$\mathcal{C}_4$	$M_2(x)M_4(x)M_5(x) = x^5 + 6x^4 + 5x^3 + 5x^2 + 4x + 8$	$\{2,4,5,6,7\}$
$\mathcal{C}_5$	$M_1(x)M_2(x)M_4(x) = x^5 + 5x^4 + 4x^3 + 4x^2 + 3x + 8$	$\{1,2,3,4,6\}$
$\mathcal{C}_6$	$M_1(x)M_2(x) = x^4 + 4x^3 + 4x + 8$	$\{1,2,3,6\}$
$\mathcal{C}_7$	$M_1(x)M_5(x) = x^4 + 1$	$\{1,3,5,7\}$
$\mathcal{C}_8$	$M_4(x)M_5(x) = x^3 + 6x^2 + 4x + 8$	$\{4,5,7\}$
$\mathcal{C}_9$	$M_5(x) = x^2 + 5x + 8$	$\{5,7\}$
$\mathcal{C}_{10}$	$M_4(x) = x + 1$	$\{4\}$

Tabla 1. Algunos códigos cíclicos de longitud $n = 8$ sobre $\mathbb{Z}_9$.

Definición 3.2.11. *Sea α un elemento primitivo de G_n. Un código BCH sobre $\mathbb{Z}_{p^s}$ con distancia diseñada δ es un código cíclico de longitud n cuyo conjunto de definición contiene a $\{b, b+1, \ldots, b+\delta-2\}$ módulo n para algún entero $b \geq 0$ y $2 \leq \delta \leq n$. De modo que su polinomio generador es*

$$g(x) = LCM(M_b(x), M_{b+1}(x), \ldots, M_{b+\delta-2}(x))$$

donde $M_i(x)$ es el polinomio mínimo de α^i sobre $\mathbb{Z}_{p^s}$.

Obsérvese que todo código cíclico es trivialmente un código BCH tomando como distancia diseñada $\delta = 2$.

Los códigos recién definidos satisfacen la cota BCH, como lo muestra el siguiente teorema.

Teorema 3.2.12. *Un código BCH sobre $\mathbb{Z}_{p^s}$ con distancia diseñada δ tiene distancia mínima por lo menos δ.*

DEMOSTRACIÓN. Sea $\mathcal{C}$ el código BCH sobre $\mathbb{Z}_{p^s}$ con distancia diseñada δ generado por el polinomio $g(x) \in \mathbb{Z}_{p^s}[x]$. Por el Corolario 3.2.5, $\mu(g(x))$ genera un código BCH con distancia diseñada δ sobre $\mathbb{F}_p$, digamos $\overline{\mathcal{C}}$, el cual por la cota BCH (Teorema 1.2.9) tiene distancia mínima por lo menos δ. Supóngase que existe una palabra $c(x) \in \mathcal{C}$ de peso menor que δ, entonces $\mu(c(x))$ es una palabra de peso menor que δ en $\overline{\mathcal{C}}$ lo cual es una contradicción. Por lo tanto también $\mathcal{C}$ debe tener distancia mínima por lo menos δ. □

EJEMPLO 3.2.13. Algunos de los códigos cíclicos de longitud $n = 8$ sobre $\mathbb{Z}_9$ dados en el Ejemplo 3.2.10 son códigos BCH no triviales. Estos códigos se muestran en la Tabla 2 y debido al Teorema 3.2.12 tienen distancia mínima por lo menos δ. Se está tomando $b = 1$.

	$g(x)$	T	δ
$\mathcal{C}_1$	$x^8 + 8$	$\{0, 1, 2, 3, 4, 5, 6, 7\}$	9
$\mathcal{C}_2$	$x^6 + 4x^5 + 8x^4 + 8x^2 + 5x + 1$	$\{0, 1, 2, 3, 4, 6\}$	5
$\mathcal{C}_3$	$x^6 + 5x^5 + 8x^4 + 8x^2 + 4x + 1$	$\{0, 2, 4, 5, 6, 7\}$	4
$\mathcal{C}_4$	$x^5 + 6x^4 + 5x^3 + 5x^2 + 4x + 8$	$\{2, 4, 5, 6, 7\}$	4
$\mathcal{C}_5$	$x^5 + 5x^4 + 4x^3 + 4x^2 + 3x + 8$	$\{1, 2, 3, 4, 6\}$	5
$\mathcal{C}_6$	$x^4 + 4x^3 + 4x + 8$	$\{1, 2, 3, 6\}$	4
$\mathcal{C}_8$	$x^3 + 6x^2 + 4x + 8$	$\{4, 5, 7\}$	3

Tabla 2. Algunos códigos cíclicos de longitud $n = 8$ sobre $\mathbb{Z}_9$.

3.3. Decodificación de códigos BCH sobre $\mathbb{Z}_{p^s}$

Sea $\mathcal{C}$ un código BCH sobre $\mathbb{Z}_{p^s}$ construido a partir del elemento α primitivo en G_n con polinomio generador

$$g(x) = LCM(M_b(x), M_{b+1}(x), \ldots, M_{b+2t-1}(x))$$

Por el Teorema 3.2.12, este código tiene distancia mínima por lo menos $2t + 1$ y por lo tanto puede corregir por lo menos t errores. Por simplicidad consideraremos el caso en que $b = 1$, los ajustes necesarios para el caso general pueden hacerse de manera similar a la Sección 2.1.

Supóngase que se envía la palabra $c(x) \in \mathcal{C}$ y que en su lugar se recibe $r(x) = c(x) + e(x)$, donde $e(x) \in \mathbb{Z}_{p^s}[x]$ es el polinomio error con a lo más t coeficientes distintos de cero. Supóngase además que ocurrieron v errores en las posiciones $i_1, i_2, \ldots, i_v$, donde $0 \leq v \leq t$.

Podemos escribir entonces $e(x) = e_{i_1}x^{i_1} + e_{i_2}x^{i_2} + \cdots + e_{i_v}x^{i_v}$ donde e_{i_l} es la magnitud del l-ésimo error.

De igual forma que en el caso de campos, los síndromes están dados por,

$$S_j = r(\alpha^j) = e(\alpha^j) \quad \text{para } j = 1, 2, \ldots, 2t$$

Siguiendo el proceso visto en la Sección 2.1 definimos las magnitudes de error Y_l y las localizaciones de error X_l, como $Y_l = e_{i_l}$ y $X_l = \alpha^{i_l}$ respectivamente para $l = 1, 2, \ldots, v$ donde i_l es la posición del l-ésimo error. Obsérvese que las magnitudes de error son elementos en $\mathbb{Z}_{p^s}$. Nuevamente se tiene el siguiente sistema de ecuaciones que relaciona los síndromes con las magnitudes y localizaciones de error.

$$\begin{aligned} S_1 &= Y_1X_1 + \cdots + Y_vX_v \\ S_2 &= Y_1X_1^2 + \cdots + Y_vX_v^2 \\ &\vdots \\ S_{2t} &= Y_1X_1^{2t} + \cdots + Y_vX_v^{2t} \end{aligned} \tag{37}$$

Definimos el polinomio localizador de errores como

$$\begin{aligned} \Lambda(x) &= (1 - X_1x)(1 - X_2x)\cdots(1 - X_vx) \\ &= \Lambda_vx^v + \Lambda_{v-1}x^{v-1} + \cdots + \Lambda_1x + 1 \end{aligned} \tag{38}$$

que es un polinomio cuyos coeficientes están en el anillo de Galois $\mathcal{R} = GR(p^s, m)$

De igual manera que en el caso de campos (Sección 2.1), los coeficientes de $\Lambda(x)$ satisfacen las siguientes relaciones

$$\Lambda_1S_{j+v-1} + \Lambda_2S_{j+v-2} + \cdots + \Lambda_vS_j = -S_{j+v} \quad \text{para } j = 1, \ldots, v \tag{39}$$

El algoritmo iterativo visto en la Sección 2.2 requería en algunas actualizaciones el inverso multiplicativo de ciertos elementos, sin embargo, en este caso tanto los coeficientes de $\Lambda(x)$ como los síndromes están por definición en el anillo de Galois $\mathcal{R} = GR(p^s, m)$ donde no todos los elementos son invertibles. A continuación se verá una forma de generalizar el proceso iterativo descrito en la Sección 2.2 a este tipo de anillos, utilizando el hecho de que se trata de un anillo local.

3.3.1. El algoritmo de Berlekamp-Massey en anillos de Galois

El algoritmo de Berlekamp-Massey resuelve el siguiente problema: dada una sucesión $S_0, S_1, \ldots, S_{n-1}$ de elementos en un campo, determinar la recurrencia lineal de menor longitud que la genera. Veremos una generalización de este algoritmo que permite resolver este mismo problema en anillos de Galois, siguiendo las ideas dadas por J.A. Reeds y N.J.A Sloane en [**17**].

Sea $\mathcal{R} = GR(p^s, m)$ el anillo de Galois $\mathbb{Z}_{p^s}[y]/\langle h(y)\rangle$, donde $h(y)$ es un polinomio mónico básico irreducible de grado m sobre $\mathbb{Z}_{p^s}$ y sea $\mathcal{R}^*$ su grupo de unidades. Al igual que en la Sección 2.2 vamos a decir que la sucesión S_0, S_1,..., $S_{n-1} \in \mathcal{R}$ es generada por una recurrencia lineal de longitud L si existen elementos $a_0 = 1, a_1, \ldots, a_L \in \mathcal{R}$ tales que

$$\sum_{i=0}^{L} a_i S_{j-i} = 0 \quad \text{para } j = L, \ldots, n-1 \tag{40}$$

Sean $a(x) = a_0 + a_1 x + \cdots + a_L x^L$ y $S(x) = S_0 + S_1 x + \cdots + S_{n-1} x^{n-1}$, entonces la condición (40) es equivalente a

$$\begin{aligned} S(x)a(x) &\equiv b(x) \mod x^n \\ a(0) &= 1 \end{aligned} \tag{41}$$

Para algún polinomio $b(x) \in \mathcal{R}[x]$ de grado $\leq L-1$. La longitud de la recurrencia satisface $L \geq \text{máx}\{\text{gr}(a(x)), 1 + \text{gr}(b(x))\}$ y como se busca una recurrencia de longitud mínima puede suponerse que $L = \text{máx}\{\text{gr}(a(x)), 1 + \text{gr}(b(x))\}$. Sea $A = (a(x), b(x))$ y $L(A) = \text{máx}\{\text{gr}(a(x)), 1 + \text{gr}(b(x))\}$. Por convención se tomará $\text{gr}(0) = -\infty$.

Con la notación establecida, dados $S_0, S_1, \ldots, S_{n-1} \in \mathcal{R}$ lo que se busca es una recurrencia lineal $A = (a(x), b(x))$ de longitud mínima $L = L(A)$ que satisfaga (41). La idea principal es considerar no sólo esta condición si no que para cada $\eta = 0, 1, \ldots, s-1$ se buscarán parejas $A_\eta = (a_\eta(x), b_\eta(x))$ tales que

$$\begin{aligned} S(x)a_\eta(x) &\equiv b_\eta(x) \mod x^n \\ a_\eta(0) &= p^\eta \end{aligned} \tag{42}$$

con $L(A_\eta) = L_\eta$, mínima.

El algoritmo es nuevamente un proceso iterativo cuyo objetivo es calcular para todo $0 \leq k \leq n$ y $0 \leq \eta < s$ parejas $A_\eta^{(k)} = (a_\eta^{(k)}(x), b_\eta^{(k)}(x))$

que satisfacen

$$\begin{aligned} S(x)a_\eta^{(k)}(x) &\equiv b_\eta^{(k)}(x) \mod x^k \\ a_\eta^{(k)}(0) &= p^\eta \end{aligned} \tag{43}$$

con $L(A_\eta^{(k)})$ mínima.

Sea $p^{u_{\eta k}}$ la mayor potencia de p que divide al coeficiente de x^k en el polinomio $S(x)a_\eta^{(k)}(x) - b_\eta^{(k)}(x)$. Si este coeficiente es cero se toma $u_{\eta k} = s$. Más adelante se verá que en el k-ésimo paso se satisface la siguiente propiedad para todo $0 \leq r < k$,

(P_r): $\forall \quad 0 \leq g < e$ se tiene que $L(A_g^{(r+1)}) = L(A_g^{(r)})$ o bien existe un entero $h = f(g, r)$ tal que

$$\begin{aligned} g + u_{hr} &< e \\ L(A_g^{(r+1)}) &= r + 1 - L(A_h^{(r)}) \quad \text{y} \\ L(A_g^{(r+1)}) &> L(A_g^{(r)}) \end{aligned}$$

Estas condiciones son análogas a las del Lema 2.2.1 y el Corolario 2.2.2 en el algoritmo de Berlekamp-Massey para campos visto en la Sección 2.2.

Con esto, el algoritmo calcula $A_\eta^{(k+1)}$ y $f(\eta, k)$ para $0 \leq \eta < s$ de manera que satisfagan la propiedad (P_k). Las cantidades $L(A_\eta^{(k)})$ también satisfacen que

$$L(A_{\eta+1}^{(k)}) \leq L(A_\eta^{(k)}) \leq L(A_\eta^{(k+1)})$$

Recuérdese que el objetivo es el siguiente: dados $S_0, S_1, \ldots, S_{n-1} \in \mathcal{R} = GR(p^s, m)$ encontrar una pareja $A = (a(x), b(x))$ tal que

$$\begin{aligned} S(x)a(x) &\equiv b(x) \mod x^n \\ a(0) &= 1 \end{aligned}$$

cuya longitud $L = L(A) = \text{máx}\{\text{gr}(a(x)), 1 + \text{gr}(b(x))\}$ sea mínima.

A continuación se describe el algoritmo que determina la pareja $A = (a(x), b(x))$ con las condiciones deseadas. En el paso inicial se toma $k = 0$ y para cada $\eta = 0, 1, \ldots, s-1$ se define

$$\begin{aligned} a_\eta^{(0)}(x) &= p^\eta, \quad b_\eta^{(0)}(x) = 0, \\ a_\eta^{(1)}(x) &= p^\eta, \quad b_\eta^{(1)}(x) = p^\eta S_0, \\ A_\eta^{(i)} &= (a_\eta^{(i)}(x), b_\eta^{(i)}(x)) \quad \text{para } i = 0, 1 \end{aligned}$$

Se define también $\theta_{\eta 0}p^{u_{\eta 0}}$ como el término constante en $S(x)a_\eta^{(0)}(x) - b_\eta^{(0)}(x)$ donde $\theta_{\eta 0} \in \mathcal{R}^*$. Si dicho término es cero se toma $\theta_{\eta 0} = 1$ y $u_{\eta 0} = s$. Finalmente se define $f(\eta, 0) = 0$.

El siguiente paso se realiza para $k = 1, 2, \ldots, n-1$ y produce $A_\eta^{(k+1)}$. Para cada $\eta = 0, 1, \ldots, s-1$ se calcula $\theta_{\eta k}p^{u_{\eta k}}$ como el coeficiente de la potencia x^k en $S(x)a_\eta^{(k)}(x) - b_\eta^{(k)}(x)$ de manera que $\theta_{\eta k} \in \mathcal{R}^*$. En virtud del Teorema 3.1.3 esto siempre puede hacerse (aunque no de manera única). Si este coeficiente es cero se toma $\theta_{\eta k} = 1$ y $u_{\eta k} = s$. El término $\theta_{\eta k}p^{u_{\eta k}}$ corresponde a la discrepancia en el algoritmo para campos. Se tienen los siguientes casos:

Caso 1: Si $u_{\eta k} = s$ se toma $A_\eta^{(k+1)} = A_\eta^{(k)}$

Caso 2: Si $u_{\eta k} < s$, se definen $g = s - 1 - u_{\eta k}$ y $f(\eta, k) = g$. Se consideran entonces los subcasos:

Caso 2a: Si $L(A_g^{(k)}) = 0$ hacer $A_\eta^{(k+1)} = (a_\eta^{(k)}(x), b_\eta^{(k)}(x) + \theta_{\eta k}p^{u_{\eta k}}x^k)$

Caso 2b: Si $L(A_g^{(k)}) > 0$ se toma $0 \leq r < k$ tal que $L(A_g^{(r)}) < L(A_g^{(r+1)}) = L(A_g^{(k)})$, es decir, r es la más reciente iteración en la que hubo un cambio en la longitud $L(A_g^{(0)})$, $L(A_g^{(1)})$,...,$L(A_g^{(k)})$ y $h = f(g, r)$. En este caso se toma

$$
\begin{aligned}
a_\eta^{(k+1)}(x) &= a_\eta^{(k)}(x) - \theta_{\eta k}\theta_{hr}^{-1}p^{u_{\eta k}-u_{hr}}x^{k-r}a_h^{(r)}(x) \\
b_\eta^{(k+1)}(x) &= b_\eta^{(k)}(x) - \theta_{\eta k}\theta_{hr}^{-1}p^{u_{\eta k}-u_{hr}}x^{k-r}b_h^{(r)}(x) \quad \text{y} \\
A_\eta^{(k+1)} &= (a_\eta^{(k+1)}, b_\eta^{(k+1)})
\end{aligned} \tag{44}
$$

Aquí concluye el paso k. Al final del paso $n-1$ el algoritmo termina y la pareja deseada $A = (a(x), b(x))$ está dada por

$$A_0^{(n)} = (a_0^{(n)}(x), b_0^{(n)}(x))$$

Obsérvese que tiene sentido escribir $p^{u_{\eta k}-u_{hr}}$ en (44) pues por la elección de r y de la propiedad (P_r) se sigue que $L(A_g^{(k)}) = L(A_g^{(r+1)}) = r+1-L(A_h^{(r)})$ donde $h = f(g, r)$ y $g + u_{hr} < e$. Y como $g = e-1-u_{\eta k}$, se concluye que $u_{hr} \leq u_{\eta k}$. Así pues, una potencia anterior de p puede usarse para eliminar la potencia de p en la discrepancia actual y con la elección de $A_\eta^{(k+1)}$ dada en (44) se logra que

$$
\begin{aligned}
S(x)a_\eta^{(k+1)}(x) &\equiv b_\eta^{(k+1)}(x) \mod x^{k+1} \\
a_\eta^{(k+1)}(0) &= p^\eta
\end{aligned}
$$

Para probar que el algoritmo funciona se requiere, al igual que en el caso de campos, un lema previo. Denotaremos por $P_\eta^{(k)}$ al conjunto de parejas $(a(x), b(x))$ que satisfacen

$$\begin{aligned} S(x)a(x) &\equiv b(x) \mod x^k \\ a(0) &= p^\eta \end{aligned}$$

y por $Q_\eta^{(k)}$ con $0 \leq \eta \leq s$, al conjunto

$$Q_\eta^{(k)} = \{(a(x), b(x)) \mid S(x)a(x) \equiv b(x) + \theta p^\eta x^k \mod x^{k+1} \quad \text{para algún} \quad \theta \in \mathcal{R}^*\}$$

Obsérvese que por definición, la discrepancia en el paso k satisface

$$S(x)a_\eta^{(k)}(x) \equiv b_\eta^{(k)}(x) + \theta_{\eta k} p^{u_{\eta k}} x^k \mod x^{k+1} \tag{45}$$

y por lo tanto $A_\eta^{(k)} \in P_\eta^{(k)} \bigcap Q_{u_{\eta k}}^{(k)}$

Lema 3.3.1. *Si $(a(x), b(x)) \in P_\eta^{(k)}$ y $(c(x), d(x)) \in Q_u^{(k-1)}$ donde η y u satisfacen la relación $\eta + u < s$ entonces*

$$L(a(x), b(x)) + L(c(x), d(x)) \geq k$$

Demostración. Por definición de $P_\eta^{(k)}$ y $Q_u^{(k-1)} \mod x^k$ se tiene que:

$$\begin{aligned} S(x)a(x) &\equiv b(x) \mod x^k \quad \text{y} \\ S(x)c(x) &\equiv d(x) + \theta p^u x^{k-1} \mod x^k \end{aligned}$$

para algún $\theta \in \mathcal{R}^*$, así que

$$\begin{aligned} S(x)a(x)c(x) &\equiv b(x)c(x) \mod x^k \quad \text{y} \\ S(x)a(x)c(x) &\equiv a(x)d(x) + \theta p^u x^{k-1} a(x) \mod x^k \end{aligned}$$

Por tanto,

$$\begin{aligned} b(x)c(x) - a(x)d(x) &\equiv \theta p^u x^{k-1} a(x) \mod x^k \\ &\equiv \theta p^u x^{k-1} a(0) \mod x^k \\ &= \theta p^{\eta+u} x^{k-1} \\ &\neq 0 \quad \text{pues} \quad \eta + u < s \end{aligned}$$

Ésto significa que el grado de $b(x)c(x) - a(x)d(x)$ es por lo menos $k-1$ pero por otro lado,

$$\begin{aligned} \mathrm{gr}(b(x)c(x) - a(x)b(x)) &\leq \text{máx}\{\mathrm{gr}(b(x)c(x)), \mathrm{gr}(a(x)b(x))\} \\ &\leq L(a(x), b(x)) + L(c(x), d(x)) - 1 \end{aligned}$$

lo cual completa la prueba. □

Diremos que la pareja $(a(x), b(x)) \in P_\eta^{(k)}$ tiene longitud mínima en $P_\eta^{(k)}$ si

$$L(a(x), b(x)) \leq L(c(x), d(x)) \quad \forall (c(x), d(x)) \in P_\eta^{(k)}$$

El Lema 3.3.1 permite dar condiciones para que una pareja sea de longitud mínima lo cual se establece en el siguiente,

Corolario 3.3.2. *Bajo las hipótesis del Lema 3.3.1, si se cumple la igualdad*

$$L(a(x), b(x)) + L(c(x), d(x)) = k$$

entonces $(a(x), b(x))$ *es de longitud mínima en* $P_\eta^{(k)}$

Ahora sí estamos en condiciones de demostrar que el algoritmo descrito en efecto funciona.

Teorema 3.3.3. *Para todo* $k = 0, 1, \ldots, n$ *y* $\eta = 0, 1, \ldots, s-1$ *la pareja* $A_\eta^{(k)}$ *dada por el algoritmo descrito anteriormente tiene longitud mínima en* $P_\eta^{(k)}$

Demostración. La prueba es por inducción sobre k. La hipótesis de inducción es que, al inicio del paso k se cumplen las propiedades P_0, P_1, P_{k-1} y que $A_g^{(r)}$ tiene longitud mínima en $P_g^{(r)}$ para $0 \leq r \leq k$ y $0 \leq g \leq s$. En el k-ésimo paso se lleva a cabo lo necesario para construir $A_\eta^{(k+1)}$. Debe probarse pues, que al final del paso k se cumple la propiedad (P_k), es decir,

(P_k): $\forall \quad 0 \leq \eta < s$ se tiene que $L(A_\eta^{(k+1)}) = L(A_\eta^{(k)})$ o bien existe un entero g con

$$\begin{aligned} \eta + u_{gk} &< s \\ L(A_\eta^{(k+1)}) &= k + 1 - L(A_g^{(k)}) \quad \text{y} \\ L(A_\eta^{(k+1)}) &> L(A_\eta^{(k)}) \end{aligned}$$

y que $A_\eta^{(k+1)}$ tiene longitud mínima en $P_\eta^{(k+1)}$ para $0 \leq \eta < s$.

Probar P_0 y la minimalidad de $A_\eta^{(0)}$ y $A_\eta^{(1)}$ es inmediato. Supóngase pues que se está en el paso k y que ocurre el caso 1. En estas condiciones se toma $A_\eta^{(k+1)} = A_\eta^{(k)}$ con lo cual se satisface P_k además $A_\eta^{(k+1)}$ tiene longitud mínima por hipótesis de inducción.

Supongamos ahora que ocurre el caso 2a. Veremos primero que se satisface la propiedad (P_k). Podemos suponer que $L(A_\eta^{(k+1)}) \neq L(A_\eta^{(k)})$. En este caso se tiene que

$$L(A_g^{(k)}) = 0 \quad \text{y} \quad A_g^{(k)} = (p^g, 0)$$

y por construcción la discrepancia satisface (45), esto es

$$S(x)p^g \equiv \theta_{gk} p^{u_{gk}} x^k \mod x^{k+1}$$

Ésto implica que $p^{s-g} = p^{s-(s-1-u_{\eta k})} = p^{1+u_{\eta k}}$ divide a los $S_0, \ldots, S_{k-1}$ y que $S_k = \theta p^{u_{gk}-g}$ para algún $\theta \in \mathcal{R}^*$

Sea $S_i = p^{1+u_{\eta k}} S_i'$ para $i < k$. Como $\theta_{\eta k} p^{u_{\eta k}}$ satisface (45) tenemos

$$S(x)a_\eta^{(k)}(x) \equiv b_\eta^{(k)}(x) + \theta_{\eta k} p^{u_{\eta k}} x^k \mod x^{k+1}$$

así

$$[p^{1+u_{\eta k}}(S_0' + \cdots + S_{k-1}' x^{k-1}) + \theta p^{u_{gk}-g} x^k + \cdots][p^\eta + \cdots] \equiv b_\eta^{(k)}(x) + \theta_{\eta k} p^{u_{\eta k}} x^k \mod x^{k+1}$$

Igualando los coeficientes de x^k se obtiene que

$$\gamma p^{1+u_{\eta k}} + \theta p^{u_{gk}-g+\eta} = \theta_{\eta k} p^{u_{\eta k}} \quad \text{para algún} \quad \gamma \in \mathcal{R}$$

lo cual implica que:

$$\gamma p^{1+u_{\eta k}} + \theta p^{u_{gk}+\eta-s+1+u_{\eta k}} = \theta_{\eta k} p^{u_{\eta k}}$$

Como $\theta_{\eta k}$ es unidad, p no divide a $p^{u_{gk}+\eta-s+1}$ por lo que

$$\eta + u_{gk} = s - 1 < s$$

Ahora, $\text{gr}(b_\eta^{(k)}(x)) \leq k-1$ y en el caso bajo consideración la siguiente pareja es:

$$A_\eta^{(k+1)} = (a_\eta^{(k)}(x), b_\eta^{(k)}(x) + \theta_{\eta k} p^{u_{\eta k}} x^k)$$

Así

$$L(A_\eta^{(k+1)}) = \text{máx}\{\text{gr}(a_\eta^{(k)}), k+1\}$$

y por la suposición de que $L(A_\eta^{(k+1)}) \neq L(A_\eta^{(k)})$ se concluye que

$$\begin{aligned} L(A_\eta^{(k+1)}) &= k+1 \\ &= k+1-L(A_g^{(k)}) \end{aligned}$$

y que $L(A_\eta^{(k+1)}) > L(A_\eta^{(k)})$. Tenemos pues que $\eta + u_{gk} < s$ y $L(A_\eta^{(k+1)}) + L(A_g^{(k)}) = k+1$ lo cual junto con el Corolario 3.3.2 implica la minimalidad de $A_\eta^{(k+1)}$ en $P_\eta^{(k+1)}$.

Resta ver el caso 2b. Para ver la propiedad (P_k), nuevamente podemos suponer que se cumple $L(A_\eta^{(k+1)}) \neq L(A_\eta^{(k)})$ y así

$$k - r + L(A_h^{(r)}) > L(A_\eta^{(k)})$$

además por la elección de r y la hipótesis de inducción,

$$L(A_g^{(k)}) = L(A_\eta^{(r+1)}) = r + 1 - L(A_h^{(r)}) \tag{46}$$

entonces

$$k + 1 > L(A_\eta^{(k)}) + L(A_g^{(k)}) \tag{47}$$

Considérese el polinomio

$$\begin{aligned} q(x) &= a_\eta^{(k)}(x)[S(x)a_g^{(k)}(x) - b_g^{(k)}(x)] - a_g^{(k)}(x)[S(x)a_\eta^{(k)}(x) - b_\eta^{(k)}(x)] \\ &= a_g^{(k)}(x)b_\eta^{(k)} - a_\eta^{(k)}b_g^{(k)}(x) \end{aligned}$$

De (47) resulta que

$$\begin{aligned} \mathrm{gr}(q(x)) &\leq \max\{\mathrm{gr}(a_g^{(k)}(x)b_\eta^{(k)}(x)), \mathrm{gr}(a_\eta^{(k)}(x)b_g^{(k)}(x))\} \\ &\leq L(A_\eta^{(k)}) + L(A_g^{(k)}) - 1 \\ &< k \end{aligned}$$

Por otro lado, por definición, el polinomio $q(x)$ se expresa como

$$q(x) = (p^\eta + \cdots)(\theta_{gk}p^{u_{gk}}x^k + \cdots) - (p^g + \cdots)(\theta_{\eta k}p^{u_{\eta k}}x^k + \cdots)$$

que contiene sólo términos de grado mayor o igual que k. De lo anterior se desprende que $q(x) \equiv 0$ pero el coeficiente de x^k es

$$\theta_{gk}p^{u_{gk}+\eta} - \theta_{\eta k}p^{u_{\eta k}+g} = 0$$

y como θ_{gk} y $\theta_{\eta k}$ son unidades, en virtud del Teorema 3.1.3, debe ser

$$\begin{aligned} u_{gk} + \eta &= u_{\eta k} + g \\ &= u_{\eta k} + s - 1 - u_{\eta k} \\ &= s - 1 \end{aligned}$$

Por lo tanto, $\eta + u_{gk} < s$. Ahora, por la forma en que este caso en se toma $A_\eta^{(k+1)}$, de (46) y (47) se tiene que

$$\begin{aligned} L(A_\eta^{(k+1)}) &\leq \max\{L(A_\eta^{(k)}), k - r + L(A_h^{(r)})\} \\ &= \max\{L(A_\eta^{(k)}), k + 1 - L(A_g^{(k)})\} \\ &= k + 1 - L(A_g^{(k)}) \end{aligned}$$

Además, como $A_\eta^{(k+1)} \in P_\eta^{(k+1)}$, $A_g^{(k)} \in Q_{u_{gk}}^{(k)}$ y $\eta + u_{gk} < s$ se satisfacen las hipótesis del Lema 3.3.1 así que

$$L(A_\eta^{(k+1)}) + L(A_g^{(k)}) \geq k+1$$

con lo cual se tiene la igualdad:

$$L(A_\eta^{(k+1)}) = k+1 - L(A_g^{(k)})$$

y de (46) tiene que $L(A_\eta^{(k+1)}) > L(A_\eta^{(k)})$.

Finalmente, el Lema 3.3.2 garantiza la minimalidad de $L(A_\eta^{(k+1)})$. □

A continuación se darán algunos ejemplos que ilustren el algoritmo descrito anteriormente.

EJEMPLO 3.3.4. Considérese el anillo de Galois $\mathcal{R} = GR(2^2, 2) = \mathbb{Z}_4[y]/\langle y^2 + y + 1\rangle$. En $\mathcal{R}$ se busca la menor recurrencia que genere la sucesión $S_0 = 2$, $S_1 = y+1$, $S_2 = 2y+2$, $S_3 = 2y$, $S_4 = 3y+2$. Aquí $p = 2$ y $s = 2$.

Veamos con detalle los primeros pasos del algoritmo de Berlekamp-Massey en anillos de Galois como está dado en la página 67. Se tiene que

$$S(x) = 2 + (y+1)x + (2y+2)x^2 + (2y)x^3 + (3y+2)x^4$$

En el paso inicial se toma $k = 0$ y se definen

$$A_0^{(0)} = (1,0), \quad A_1^{(0)} = (2,0) \tag{48}$$

$$A_0^{(1)} = (1,2), \quad A_1^{(1)} = (2,0) \tag{49}$$

Se define entonces $\theta_{\eta 0} p^{u_{\eta 0}}$ como el término constante en $S(x)a_\eta^{(0)} - b_\eta^{(0)}$, para $\eta = 0, 1$. En este caso resulta

$$\theta_{00} p^{u_{00}} = 2 = 1 \cdot 2^1$$
$$\theta_{10} p^{u_{10}} = 0 = 1 \cdot 2^2$$

Por tanto,

$$\theta_{00} = 1, \quad u_{00} = 1 \tag{50}$$

$$\theta_{10} = 1, \quad u_{10} = 2 \tag{51}$$

Finalmente hacemos

$$f(\eta, 0) = 0 \quad \text{para} \quad \eta = 0, 1 \tag{52}$$

y con ésto concluye la inicialización.

Para el siguiente paso tomamos $k = 1$ y para $\eta = 0, 1$ se calcula $\theta_{\eta 1} p^{u_{\eta 1}}$ como el coeficiente de x en $S(x)a_\eta^{(1)} - b_\eta^{(1)}$, obteniendo

$$\theta_{01} p^{u_{01}} = y + 1 = (y+1) \cdot 2^0$$
$$\theta_{11} p^{u_{11}} = 2y + 2 = (y+1) \cdot 2^1$$

así que

$$\theta_{01} = y + 1, \quad u_{01} = 0 \tag{53}$$
$$\theta_{11} = y + 1, \quad u_{11} = 1 \tag{54}$$

Ahora, como $u_{01} = 0 < 2$, estamos en el caso 2 por lo que se define

$$g = 2 - 1 - u_{01} = 1 \quad \text{y}$$

$$f(0, 1) = 1 \tag{55}$$

De (49) se ve que $L(A_g^{(1)}) = L(A_1^{(1)}) = 0$, y de acuerdo al paso 2a se toma

$$a_0^{(2)}(x) = a_0^{(1)}(x) \quad \text{y} \quad b_0^{(2)}(x) = b_0^{(1)}(x) + \theta_{01} p^{u_{01}} x^1$$

por lo tanto, de (49) y (53), se obtiene

$$A_0^{(2)} = (1, 2 + (y+1)x) \tag{56}$$

Por otro lado de (54), $u_{11} = 1 < 2$ y nuevamente se está en el caso 2 de modo que se define

$$g = 2 - 1 - u_{11} = 0 \quad \text{y}$$

$$f(1, 1) = 0 \tag{57}$$

además, de (49), $L(A_g^{(1)}) = L(A_0^{(1)}) = 1 > 0$ por lo que ahora se considera el caso 2b y se toma r como la más reciente iteración para la cual $L(A_g^{(r)}) < L(A_g^{(r+1)}) = L(A_g^{(k)})$. Como $L(A_0^{(0)}) = 0$ y $L(A_0^{(1)}) = 1$ (ver (48) y (49)), entonces $r = 0$. Además, se toma $h = f(g, r) = f(0, 0) = 0$. Finalmente hacemos

$$\begin{aligned} a_1^{(2)}(x) &= a_1^{(1)}(x) - \theta_{11}\theta_{00}^{-1} p^{u_{11}-u_{00}} x \cdot a_0^{(0)}(x) \\ &= 2 - (y+1) \cdot 1^{-1} p^{1-1} x \cdot 1 \\ &= 2 + (3y+3)x \end{aligned}$$

y

$$\begin{aligned} b_1^{(2)}(x) &= b_1^{(1)}(x) - \theta_{11}\theta_{00}^{-1}p^{u_{11}-u_{00}}x \cdot b_0^{(0)}(x) \\ &= 0 \end{aligned}$$

Por lo tanto,

$$A_1^{(2)} = (2 + (3y+3)x, 0) \tag{58}$$

Estos resultados y los correspondientes a las restantes iteraciones se muestran en la Tabla 3 que dá las parejas $(a_\eta^{(k)}(x), b_\eta^{(k)}(x))$ y en la Tabla 4 que dá los correspondientes valores de L, $\theta_{\eta k}$, $u_{\eta k}$ y $f(\eta, k)$, donde $L = L(a_\eta^{(k)}(x), b_\eta^{(k)}(x))$.

k	$\eta = 0$	$\eta = 1$
0	$(1, 0)$	$(2, 0)$
1	$(1, 2)$	$(2, 0)$
2	$(1, 2 + (y+1)x)$	$(2 + (3y+3)x, 0)$
3	$(1 + 2x, 2 + (y+1)x)$	$(2, (2y+2)x)$
4	$(1 + 2x + (2y+2)x^2, 2 + (y+1)x)$	$(2, (2y+2)x)$
5	$(1 + 2x + 2yx^2 + (3y+1)x^3, 2 + (y+1)x)$	$(2 + (2y+2)x^3, (2y+2)x)$

Tabla 3. $(a_\eta^{(k)}(x), b_\eta^{(k)}(x))$ para el Ejemplo 3.3.4.

k	$\eta = 0$	$\eta = 1$
0	$(0, 1, 1, 0)$	$(0, 1, 2, 0)$
1	$(1, y+1, 0, 1)$	$(0, y+1, 1, 0)$
2	$(2, y+1, 1, 0)$	$(1, 3y, 0, 1)$
3	$(2, y, 1, 0)$	$(2, 1, 2, -)$
4	$(2, 3y+2, 0, 1)$	$(2, y, 1, 0)$
5	$(3, -, -, -)$	$(3, -, -, -)$

Tabla 4. $(L,\ \theta_{\eta k},\ u_{\eta k},\ f(\eta, k))$ para el Ejemplo 3.3.4.

Del último renglón de la Tabla 3 se obtiene que la recurrencia buscada es:

$$S_j + 2S_{j-1} + 2yS_{j-2} + (3y+1)S_{j-3} = 0 \quad \text{para} \quad j \geq 3$$

EJEMPLO 3.3.5. Considérese el anillo de Galois $\mathcal{R} = GR(3^2, 4) = \mathbb{Z}_9[y]/\langle y^4 + y^3 + 2\rangle$. Usaremos el algoritmo de Berlekamp-Massey en $\mathcal{R}$ para encontrar la menor recurrencia lineal que genere la sucesión $S_0 = 4y^3 + 5y^2 + 3y + 2$, $S_1 = 3y^3 + 6y^2 + 4$, $S_2 = 5y^3 + 4y^2 + 6y + 1$, $S_3 = 5$. En este caso $p = 3$ y $s = 2$.

Los resultados obtenidos se muestran en la Tabla 5 donde aparecen las parejas $(a_\eta^{(k)}(x), b_\eta^{(k)}(x))$, y en la Tabla 6, que da los valores de L, $\theta_{\eta k}$, $u_{\eta k}$ y $f(\eta, k)$, con $L = L(a_\eta^{(k)}(x), b_\eta^{(k)}(x))$.

k	$\eta = 0$
0	$(1, 0)$
1	$(1, 4y^3 + 5y^2 + 3y + 2)$
2	$(1 + (4y^3 + 5y^2 + 3y + 5)x, 4y^3 + 5y^2 + 3y + 2)$
3	$(1 + (4y^3 + 5y^2 + 3y + 5)x + (3y^3 + 6y^2)x^2, 4y^3 + 5y^2 + 3y + 2)$
4	$(1 + (6y^3 + 5)x + (5y^3 + y^2 + 3y + 8)x^2, (4y^3 + 5y^2 + 3y + 2) + (5y^3 + y^2 + 3y + 8)x)$
k	$\eta = 1$
0	$(3, 0)$
1	$(3, 3y^3 + 6y^2 + 6)$
2	$(3 + (3y^3 + 6y^2 + 6)x, 3y^3 + 6y^2 + 6)$
3	$(3 + (3y^3 + 6y^2 + 6)x, 3y^3 + 6y^2 + 6)$
4	$(3 + (3y^3 + 6y^2 + 6)x, 3y^3 + 6y^2 + 6)$

Tabla 5. $(a_\eta^{(k)}(x), b_\eta^{(k)}(x))$ para el Ejemplo 3.3.5.

k	$\eta = 0$	$\eta = 1$
0	$(0, 4y^3 + 5y^2 + 3y + 2, 0, 0)$	$(0, 4y^3 + 5y^2 + 3y + 2, 1, 0)$
1	$(1, 3y^3 + 6y^2 + 4, 0, 1)$	$(1, 1, 1, 0)$
2	$(1, 2y^3 + y^2 + 2, 1, 0)$	$(1, 1, 2, -)$
3	$(2, 2, 1, 0)$	$(1, 1, 2, -)$
4	$(2, -, -, -)$	$(1, -, -, -)$

Tabla 6. $(L,\ \theta_{\eta k},\ u_{\eta k},\ f(\eta, k))$ para el Ejemplo 3.3.5.

La recurrencia buscada está dada por los coeficientes del polinomio $a_0^{(4)}(x)$ que aparece en la Tabla 5:

$$S_j + (6y^3 + 5)S_{j-1} + (5y^3 + y^2 + 3y + 8)S_{j-2} = 0 \quad \text{para} \quad j \geq 2$$

3.3.2. Cálculo de las localizaciones de los errores

Calcular las localizaciones de error cuando se trabaja en anillos $\mathbb{Z}_{p^s}$ requiere un paso más que sobre campos pues en el anillo de Galois $\mathcal{R}$ la solución del sistema (39) en general no es única y el polinomio cuyos coeficientes generan la sucesión de síndromes, obtenido por medio del algoritmo de Berlekamp-Massey para anillos de Galois puede no ser el polinomio localizador de errores buscado. Veremos que es posible determinar las localizaciones de error a partir de las raíces del polinomio encontrado con el algoritmo de Berlekamp-Massey visto en la sección anterior. Se siguen las ideas descritas en [**11**].

Sea

$$\begin{aligned}\Lambda_0(x) &= \widehat{\Lambda}_v x^v + \widehat{\Lambda}_{v-1}x^{v-1} + \cdots + \widehat{\Lambda}_1 x + 1 \\ &= (1 - Z_1 x)(1 - Z_2 x)\cdots(1 - Z_v x)\end{aligned}$$

la factorización sobre $\mathcal{R}$ del polinomio obtenido con el algoritmo de Berlekamp-Massey. Entonces el polinomio recíproco de $\Lambda_0(x)$ dado por $x^v + \widehat{\Lambda}_1 x^{v-1} + \cdots + \widehat{\Lambda}_v$ puede factorizarse como

$$\Lambda_0(x) = (x - Z_1)(x - Z_2)\cdots(x - Z_v)$$

El siguiente resultado es útil para encontrar las v localizaciones de error.

Teorema 3.3.6. *Sea $\widehat{\Lambda}(x) \in \mathcal{R}[x]$ un polinomio con v raíces distintas sobre $\mathcal{R}$, digamos*

$$\begin{aligned}\widehat{\Lambda}(x) &= x^v + \widehat{\Lambda}_1 x^{v-1} + \cdots + \widehat{\Lambda}_{v-1}x + \widehat{\Lambda}_v \\ &= (x - Z_1)(x - Z_2)\cdots(x - Z_v)\end{aligned}$$

tal que sus coeficientes $\widehat{\Lambda}_i$ satisfacen la recurrencia (39). Entonces

$$Y_i P_i = 0$$

donde $P_i = \widehat{\Lambda}(X_i)$ para $1 \leq i \leq v$ con Y_i las magnitudes y X_i las localizaciones de error.

Demostración. Multiplicando el polinomio $\Lambda(x)$ por $Y_i X_i^j$ se obtiene,

$$Y_i X_i^j(x^v + \widehat{\Lambda}_1 x^{v-1} + \cdots + \widehat{\Lambda}_{v-1}x + \widehat{\Lambda}_v) = Y_i X_i^j (x - Z_1)(x - Z_2)\cdots(x - Z_v)$$

para $i \leq j \leq v$.

Evaluando en X_i y sumando para $1 \leq i \leq v$,

$$S_{j+v} + \widehat{\Lambda}_1 S_{j+v-1} + \cdots + \widehat{\Lambda}_{v-1} S_{j+1} + \widehat{\Lambda}_v S_j = \sum_{i=1}^{v} Y_i X_i^j P_i$$

Pero como los $\widehat{\Lambda}_i$ satisfacen (39), el lado izquierdo de esta ecuación se anula para $1 \leq j \leq v$ y por tanto

$$\sum_{i=1}^{v} Y_i X_i^j P_i = 0 \quad \text{para} \quad 1 \leq j \leq v$$

Matricialmente se tiene,

$$\begin{bmatrix} X_1 & X_2 & \cdots & X_v \\ X_1^2 & X_2^2 & \cdots & X_v^2 \\ \vdots & & & \\ X_1^v & X_2^v & \cdots & X_v^v \end{bmatrix} \begin{bmatrix} Y_1 P_1 \\ Y_2 P_2 \\ \vdots \\ Y_v P_v \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

el cual es un sistema homogéneo de ecuaciones lineales sobre el anillo $\mathcal{R}$ en las variables Y_iP_i cuya matriz correspondiente es cuadrada y tiene por determinante
(59)

$$(X_1 \cdots X_v) \det \begin{bmatrix} 1 & 1 & \cdots & 1 \\ X_1 & X_2 & \cdots & X_v \\ \vdots & & & \\ X_1^{v-1} & X_2^{v-1} & \cdots & X_v^{v-1} \end{bmatrix} = (X_1 \cdots X_v) \prod_{i>j} (X_i - X_j)$$

porque esta última matriz es de Vandermonde. Ahora, en virtud de la Proposición 3.2.7, este valor es una unidad de $\mathcal{R}$ pues recuérdese que los X_i son de la forma α^l para algún entero l entre 0 y $n-1$, donde α es un generador de G_n (Sección 3.2). Por lo tanto, dicha matriz es invertible (para más detalles consultar [**16**]) de modo que el sistema tiene solución única, en este caso, la trivial. Por lo tanto, $Y_iP_i = 0$ para $1 \leq i \leq v$ como se quería. □

Obsérvese que el polinomio $\widehat{\Lambda}(x)$ dado como el recíproco del polinomio $\Lambda_0(x)$ obtenido con el algoritmo de Berlekamp-Massey, es mónico y sus coeficientes satisfacen (39), luego si tiene v raíces distintas, cumple con las condiciones del teorema y puede usarse para encontrar las localizaciones de error a partir de sus raíces en la forma siguiente.

Como las magnitudes de error son distintas de cero, del teorema anterior se concluye que cada producto $P_i = (X_i - Z_1)(X_i - Z_2) \cdots (X_i - Z_v)$ es un divisor de cero en $\mathcal{R}$ por lo que cada P_i tiene por lo menos un factor $(X_i - Z_l)$ que es divisor de cero en $\mathcal{R}$. Más aún, si para P_i el factor $(X_i - Z_{l_1})$ es divisor de cero y para P_k el factor $(X_k - Z_{l_2})$ es divisor de cero, entonces $l_1 \neq l_2$ para $i \neq k$. Pues de lo contrario, si $l_1 = l_2$, se tendría que $X_i - X_k$ es también divisor de cero lo cual no es posible para $i \neq k$ (Proposición 3.2.7, Sección 3.2). De lo anterior se desprende que a cada Z_i corresponde una única localización de error X_i tal que $X_i - Z_i$ es divisor de cero.

Con estas ideas se tiene el siguiente procedimiento para encontrar las localizaciones de error:

1. Calcular las raíces en $\mathcal{R}$ de $\widehat{\Lambda}(x)$, el polinomio recíproco del polinomio $\Lambda_0(x)$ que se obtiene por medio del algoritmo de Berlekamp-Massey. Digamos, $Z_1, Z_2, \ldots, Z_v$.
2. De entre las localizaciones de error, $X_0 = \alpha^0$, $X_1 = \alpha$, …, $X_{n-1} = \alpha^{n-1}$ seleccionar X_i tal que

$$X_i - Z_i \in \langle p \rangle$$

Los elementos así encontrados son las localizaciones de error buscadas.

3.3.3. Cálculo de las magnitudes de los errores

Una vez determinadas las localizaciones de error sólo resta encontrar sus magnitudes. Para ello pueden aplicarse las ideas vistas en la Sección 2.3, es decir, las magnitudes de error Y_1, Y_2,...,Y_v están dadas por

$$Y_l = \frac{X_l^{-1}\Omega(X_l^{-1})}{\prod_{j\neq l}(1 - X_j X_l^{-1})} = -\frac{\Omega(X_l^{-1})}{\Lambda'(X_l^{-1})} \tag{60}$$

donde, $\Lambda(x) = \Lambda_v x^v + \Lambda_{v-1}x^{v-1} + \cdots + \Lambda_1 x + 1$ es el polinomio localizador de errores, $S(x) = \sum_{j=1}^{2t} S_j x^{j-1}$ y $\Omega(x) = S(x)\Lambda(x) \mod x^{2t}$.

El detalle importante es probar que esta división tiene sentido, es decir, debe probarse que el producto que aparece en el denominador de la relación (60) es una unidad de $\mathcal{R}$. En efecto, si algún factor $(1 - X_j X_l^{-1})$ fuera divisor de cero, también lo sería $(X_l - X_j)$ que es de la forma $\alpha^{i_l} - \alpha^{i_j}$ pero de la Proposición 3.2.7 esta última expresión no puede ser divisor de cero para $l \neq j$.

3.3.4. Ejemplos

En esta sección se darán algunos ejemplos que ilustren todo el proceso de decodificación para códigos BCH sobre $\mathbb{Z}_{p^s}$

Ejemplo 3.3.7. Sea $\mathcal{C}$ un código BCH de longitud $n = 8$ sobre $\mathbb{Z}_9$ con polinomio generador

$$g(x) = x^5 + 5x^4 + 4x^3 + 4x^2 + 3x + 8$$

La aritmética es la de $\mathcal{R} = \mathbb{Z}_9[y]/\langle y^2 + y + 2\rangle$ y $\alpha = 8y + 2$ es un elemento primitivo de G_8. Este código corresponde al código $\mathcal{C}_5$ dado en el Ejemplo 3.2.13 en donde se ve que:

$$\begin{aligned} g(x) &= LCM(M_1(x), M_2(x), M_3, M_4(x)) \\ &= (x-\alpha)(x-\alpha^3)(x-\alpha^2)(x-\alpha^6)(x-\alpha^4) \end{aligned}$$

donde $M_i(x)$ es el polinomio mínimo de α^i, de aquí que el código es capaz de corregir por lo menos $t = 2$ errores. Supóngase que se envía la palabra cero

$$\overline{c} = (0, 0, 0, 0, 0, 0, 0, 0)$$

y que se recibe el vector

$$\overline{r} = (0, 3, 0, 0, 0, 0, 6, 0)$$

Entonces el vector de error está dado por,

$$\overline{e} = (0, 3, 0, 0, 0, 0, 6, 0)$$

y el polinomio correspondiente al vector recibido es $r(x) = 6x^6 + 3x$.

Para decodificar primero se calculan los síndromes que son,

$$\begin{aligned} S_1 &= r(\alpha) = 3 \\ S_2 &= r(\alpha^2) = 3y \\ S_3 &= r(\alpha^3) = 3 \\ S_4 &= r(\alpha^4) = 3 \end{aligned}$$

A continuación se busca una recurrencia de longitud mínima en $\mathcal{R}$ que genere los elementos $S_0 = 3, S_1 = 3y, S_2 = 3, S_3 = 3$. Para ello usamos el algoritmo de Berlekamp-Massey para anillos de Galois con $p = 3$ y $s = 2$. Los resultados se muestran en la Tabla 7. Veamos cómo se obtuvieron algunos de ellos.

Primeramente se toma $k = 0$ y se definen

$$A_0^{(0)} = (1, 0), \quad A_1^{(0)} = (3, 0) \tag{61}$$

$$A_0^{(1)} = (1, 3), \quad A_1^{(1)} = (3, 0) \tag{62}$$

En seguida, para $\eta = 0, 1$, se define $\theta_{\eta 0} p^{u_{\eta 0}}$ como el término constante en $S(x)a_\eta^{(0)} - b_\eta^{(0)}$. En este caso resulta

$$\begin{aligned} \theta_{00} p^{u_{00}} &= 3 = 1 \cdot 3^1 \\ \theta_{10} p^{u_{10}} &= 0 = 1 \cdot 3^2 \end{aligned}$$

Por tanto,

$$\theta_{00} = 1, \quad u_{00} = 1 \tag{63}$$

$$\theta_{10} = 1, \quad u_{10} = 2 \tag{64}$$

Finalmente hacemos

$$f(\eta, 0) = 0 \quad \text{para} \quad \eta = 0, 1 \tag{65}$$

con lo cual termina la inicialización.

En el siguiente paso tomamos $k = 1$ y para $\eta = 0, 1$ se calcula $\theta_{\eta 1} p^{u_{\eta 1}}$ como el coeficiente de x en $S(x)a_\eta^{(1)} - b_\eta^{(1)}$, obteniendo

$$\begin{aligned} \theta_{01} p^{u_{01}} &= 3y = y \cdot 3^1 \\ \theta_{11} p^{u_{11}} &= 0 \end{aligned}$$

así que

$$\theta_{01} = y, \quad u_{01} = 1 \tag{66}$$

$$\theta_{11} = 1, \quad u_{11} = 2 \tag{67}$$

Puesto que $u_{01} = 1 < 2$, estamos en el caso 2 por lo que se define

$$g = 2 - 1 - u_{01} = 0 \quad \text{y}$$

$$f(0, 1) = 0 \tag{68}$$

De (62) se ve que $L(A_g^{(1)}) = L(A_0^{(1)}) = 1 \neq 0$, por lo que se considera el caso 2b y se toma r como la más reciente iteración para la cual $L(A_g^{(r)}) < L(A_g^{(r+1)}) = L(A_g^{(k)})$. Como $L(A_0^{(0)}) = 0$ y $L(A_0^{(1)}) = 1$ (ver (61) y (62)), entonces $r = 0$. Además, se toma $h = f(g, r) = f(0, 0) = 0$. Finalmente hacemos

$$\begin{aligned} a_0^{(2)}(x) &= a_0^{(1)}(x) - \theta_{01}\theta_{00}^{-1} p^{u_{01}-u_{00}} x \cdot a_0^{(0)}(x) \\ &= 1 - (y) \cdot 1^{-1} p^{1-1} x \cdot 1 \\ &= 1 + (8y)x \end{aligned}$$

y

$$\begin{aligned} b_0^{(2)}(x) &= b_0^{(1)}(x) - \theta_{01}\theta_{00}^{-1} p^{u_{01}-u_{00}} x \cdot b_0^{(0)}(x) \\ &= 3 - 0 = 3 \end{aligned}$$

Por lo tanto,

$$A_0^{(2)} = (1 + (8y)x, 3) \tag{69}$$

Por otro lado, de (67), $u_{11} = 2$ así que de acuerdo al algoritmo se está en el caso 1 por lo que se toma

$$A_1^{(2)} = A_1^{(1)} = (3, 0) \tag{70}$$

k	$\eta = 0$	$\eta = 1$
0	$(1, 0)$	$(3, 0)$
1	$(1, 3)$	$(3, 0)$
2	$(1 + (8y)x, 3)$	$(3, 0)$
3	$(1 + (8y)x + (8y)x^2, 3)$	$(3, 0)$
4	$(1 + (8y)x + (8y)x^2, 3)$	$(3, 0)$

Tabla 7. $(a_\eta^{(k)}(x), b_\eta^{(k)}(x))$ para el Ejemplo 3.3.7.

Del último renglón de la Tabla 7 se obtiene el polinomio $\Lambda_0(x) = 1+(8y)x+(8y)x^2$ cuyo recíproco y sus raíces son: $\widehat{\Lambda}(x) = x^2+(8y)x+(8y)$, $Z_1 = 8y + 2$ y $Z_2 = 2y + 7$.

De entre los valores $\alpha^0 = 1, \alpha = 8y+2, \alpha^2 = 4y+2, \alpha^3 = y+3, \alpha^4 = 8, \alpha^5 = y + 7, \alpha^6 = 5y + 7, \alpha^7 = 8y + 6$, se tiene que $X_1 = \alpha$ y $X_2 = \alpha^6$ son tales que $X_1 - Z_1 = 0 \in \langle 3 \rangle$ y $X_2 - Z_2 = 3y \in \langle 3 \rangle$. Por lo tanto, X_1 y X_2 son las localizaciones de error e indican que ocurrieron dos errores en las posiciones 1 y 6.

Finalmente, para determinar las magnitudes de los errores usamos las ecuaciones dadas en (60) con

$$\begin{aligned} \Lambda(x) &= (1 - \alpha x)(1 - \alpha^6 x) \quad \text{y} \\ \Omega(x) &= 3 \end{aligned}$$

obteniendo

$$\begin{aligned} Y_1 &= \frac{-\Omega(\alpha^7)}{\Lambda'(\alpha^7)} = \frac{-3}{6y + 5} = 3 \\ Y_2 &= \frac{-\Omega(\alpha^2)}{\Lambda'(\alpha^2)} = \frac{-3}{3y + 4} = 6 \end{aligned}$$

Por lo tanto, el polinomio de error es $e(x) = 3x + 6x^6$ y la palabra enviada es, en efecto, $c(x) = r(x) - e(x) = 0$.

Ejemplo 3.3.8. Considérese el anillo de Galois $\mathcal{R} = \mathbb{Z}_9[y]/\langle y^4 + y^3 + 2 \rangle$ y sea $\alpha = 8y^3 + 2y^2 + 5y + 5$. Puede verificarse que α es un generador de las raíces décimosextas de la unidad G_{16} sobre $\mathcal{R}$ así que podemos usar este elemento para construir códigos cíclicos de longitud

$n = 16$ sobre $\mathbb{Z}_9$. Los distintos polinomios mínimos sobre $\mathbb{Z}_9$ son:

$$\begin{aligned}
M_0(x) &= x - \alpha^0 = x + 8 \\
M_1(x) &= (x-\alpha)(x-\alpha^3)(x-\alpha^9)(x-\alpha^{11}) = x^4 + 5x^2 + 8 \\
M_2(x) &= (x-\alpha^2)(x-\alpha^6) = x^2 + 5x + 8 \\
M_4(x) &= (x-\alpha^4)(x-\alpha^{12}) = x^2 + 1 \\
M_5(x) &= (x-\alpha^5)(x-\alpha^7)(x-\alpha^{13})(x-\alpha^{15}) = x^4 + 4x^2 + 8 \\
M_8(x) &= x - \alpha^8 = x + 1 \\
M_{10}(x) &= (x-\alpha^{10})(x-\alpha^{14}) = x^2 + 4x + 8
\end{aligned}$$

Sea $\mathcal{C}$ el código BCH generado por,

$$\begin{aligned}
g(x) &= LCM(M_1(x), M_2(x), M_3, M_4(x)) \\
&= M_1(x)M_2(x)M_4(x) \\
&= x^8 + 5x^7 + 5x^6 + 3x^5 + 7x^4 + 2x^3 + 4x^2 + 4x + 1
\end{aligned}$$

Este código tiene capacidad correctora $t = 2$. Supóngase que se envía la palabra cero

$$\overline{c} = (0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)$$

y que se recibe el vector

$$\overline{r} = (0,0,0,0,2,0,0,0,0,0,6,0,0,0,0,0)$$

cuyo polinomio correspondiente es,

$$r(x) = 6x^{10} + 2x^4$$

En el proceso de decodificación primeramente se calculan los síndromes, obteniendo,

$$\begin{aligned}
S_1 &= r(\alpha) = 4y^3 + 5y^2 + 3y + 2 \\
S_2 &= r(\alpha^2) = 3y^3 + 6y^2 + 4 \\
S_3 &= r(\alpha^3) = 5y^3 + 4y^2 + 6y + 1 \\
S_4 &= r(\alpha^4) = 5
\end{aligned}$$

El siguiente paso es obtener la recurrencia de longitud mínima en $\mathcal{R}$ que los genere para lo cual usamos el algoritmo de Berlekamp-Massey sobre anillos de Galois. Éstos cálculos se efectuaron en el Ejemplo 3.3.5, la Tabla 5 correspondiente muestra el polinomio cuyos coeficientes dan la recurrencia deseada que es,

$$\Lambda_0(x) = 1 + (6y^3 + 5)x + (5y^3 + y^2 + 3y + 8)x^2$$

El correspondiente polinomio recíproco es, entonces

$$\widehat{\Lambda}(x) = x^2 + (6y^3 + 5)x + (5y^3 + y^2 + 3y + 8)$$

Las raíces de $\widehat{\Lambda}(x)$ fueron obtenidas por búsqueda exhaustiva y son $Z_1 = 4y^3 + 8y^2 + 3y + 3$ y $Z_2 = 8y^3 + y^2 + 6y + 1$.

De entre los valores

$$\begin{array}{ll} \alpha^0 = 1 & \alpha^8 = 8\ \alpha^8 = 8 \\ \alpha = 8y^3 + 2y^2 + 5y + 5 & \alpha^9 = y^3 + 7y^2 + 4y + 4 \\ \alpha^2 = 2y^3 + 7y^2 + 6y & \alpha^{10} = 7y^3 + 2y^2 + 3y \\ \alpha^3 = 2y^3 + 6y^2 + 5y + 7 & \alpha^{11} = 7y^3 + 3y^2 + 4y + 2 \\ \alpha^4 = 8y^3 + y^2 + 6y + 1 & \alpha^{12} = y^3 + 8y^2 + 3y + 8 \\ \alpha^5 = 7y^3 + 8y^2 + 7y + 6 & \alpha^{13} = 2y^3 + y^2 + 2y + 3 \\ \alpha^6 = 7y^3 + 2y^2 + 3y + 4 & \alpha^{14} = 2y^3 + 7y^2 + 6y + 5 \\ \alpha^7 = 3y^3 + 2y^2 + 6y + 4 & \alpha^{15} = 6y^3 + 7y^2 + 3y + 5 \end{array}$$

vemos que,

$$\alpha^{10} - Z_1 = 3y^3 + 3y^2 + 6 \in \langle 3 \rangle$$

y

$$\alpha^4 - Z_2 = 0 \in \langle 3 \rangle$$

por lo tanto las localizaciones de error son $X_1 = \alpha^{10}$ y $X_2 = \alpha^4$. Se concluye pues que los errores ocurrieron en las posiciones 10 y 4. Para determinar sus magnitudes usamos las ecuaciones (60) con,

$$\begin{aligned} \Lambda(x) &= (1 - \alpha^{10}x)(1 - \alpha^4 x) \quad \text{y} \\ \Omega(x) &= (2y^3 + 7y^2 + 6y + 5)x + (4y^3 + 5y^2 + 3y + 2) \end{aligned}$$

y obtenemos

$$\begin{aligned} Y_1 &= \frac{-\Omega(\alpha^6)}{\Lambda'(\alpha^6)} = \frac{-(3y^3 + 6y^2 + 3)}{y^3 + 8y^2 + 3y + 1} = 6 \\ Y_2 &= \frac{-\Omega(\alpha^{12})}{\Lambda'(\alpha^{12})} = \frac{-(2y^3 + 7y^2 + 6y + 2)}{8y^3 + y^2 + 6y + 8} = 2 \end{aligned}$$

De lo anterior se concluye que el polinomio de error es $e(x) = 6x^{10} + 2x^4$ y por tanto la palabra enviada es $c(x) = r(x) - e(x) = 0$.

En los dos ejemplos anteriores la solución del sistema (39) no es única. Los polinomios $\Lambda_0(x)$ y el verdadero polinomio localizador de errores $\Lambda(x)$ son tales que sus coeficientes la satisfacen. Veremos ahora una condición necesaria y suficiente para que la solución de dicho sistema sea única, en cuyo caso, las localizaciones de error no requieren del cálculo adicional pues el algoritmo de Berlekamp-Massey directamente da el polinomio localizador de errores el cual corresponde a la única solución.

Proposición 3.3.9. *El sistema*

$$\Lambda_1 S_{j+v-1} + \Lambda_2 S_{j+v-2} + \cdots + \Lambda_v S_j = -S_{j+v} \quad \textit{para } j = 1, \ldots, v$$

donde los S_i son los síndromes, tiene solución única si y sólo si las magnitudes de los errores Y_l, $1 \leq l \leq v$ son unidades en $\mathbb{Z}_{p^s}$.

DEMOSTRACIÓN. Consideremos el sistema en su forma matricial,

$$\begin{bmatrix} S_1 & S_2 & \cdots & S_v \\ S_2 & S_3 & \cdots & S_{v+1} \\ \vdots & & & \\ S_v & S_{v+1} & \cdots & S_{2v-1} \end{bmatrix} \begin{bmatrix} \Lambda_v \\ \Lambda_{v-1} \\ \vdots \\ \Lambda_1 \end{bmatrix} = \begin{bmatrix} -S_{v+1} \\ -S_{v+2} \\ \vdots \\ -S_{2v} \end{bmatrix} \tag{71}$$

Al igual que en la Sección 2.1 podemos escribir a la matriz del sistema (71) como $M = ABA^t$, donde $A_{ij} = X_j^{i-1}$ y $B_{ij} = X_i^b Y_i \delta_{ij}$, con $\delta_{ij} = \begin{cases} 1 & \text{si } i = j \\ 0 & \text{si } i \neq j \end{cases}$.

La matriz A es la misma que aparece en la ecuación (59) y ya se vió que su determinante es una unidad. Por otro lado, la matriz B tiene determinante

$$\det(B) = \prod_{i=1}^{v} Y_i X_i$$

que es un divisor de cero si y sólo si algún Y_i es divisor de cero. Por lo tanto, $\det(M)$ es una unidad si y sólo si todos los Y_l, $1 \leq l \leq v$ son unidades y en este caso la solución de (39) es única. □

Conclusiones

Este trabajo presenta una construcción de códigos cíclicos y en particular códigos BCH sobre el anillo $\mathbb{Z}_{p^s}$ que es completamente análoga a la forma usual de dar este tipo de códigos sobre campos finitos. La idea es dar los códigos cíclicos en términos de las raíces de su polinomio generador para lo cual se requiere factorizar el polinomio $x^n - 1$ sobre una extensión adecuada de $\mathbb{Z}_{p^s}$, que resulta ser el anillo de Galois $GR(p^s, m)$.

Se describe, además, un algoritmo para decodificar códigos BCH sobre $\mathbb{Z}_{p^s}$ que extiende las ideas de uno de los algoritmos más importantes y eficientes para la decodificación de códigos BCH sobre un campo finito: el Algoritmo de Berlekamp-Massey. Dicho algoritmo resuelve el problema de encontrar la menor recurrencia lineal que genera una sucesión dada sobre cualquier campo y al ser aplicado a la sucesión de síndromes permite encontrar el polinomio localizador de errores. El algoritmo aquí presentado se basa principalmente en las ideas de J. A. Reeds y N. J. A. Sloane quienes en [**19**] dan un algoritmo cuyo objetivo es encontrar la menor recurrencia lineal que genera una sucesión dada en $\mathbb{Z}_m$, en este trabajo se utilizan estas ideas para resolver el mismo problema pero ahora sobre el anillo de Galois $GR(p^s, m)$. Cuando se aplica este algoritmo a la sucesión de síndromes para un código sobre $\mathbb{Z}_{p^s}$, se obtiene un polinomio que no necesariamente es el polinomio localizador de errores, sin embargo, usando la técnica presentada por J. C. Interlando, R. Palazzo y M. Elia en [**11**] el polinomio localizador de errores correcto puede obtenerse a partir de éste.

Cabe mencionar que existen otras generalizaciones del algoritmo de Berlekamp-Massey tanto para otro tipo de códigos, como lo es el algoritmo de Sudan-Guruswami utilizado en la decodificación de códigos geométrico algebraicos [**23**], como a otras estructuras, caso del algoritmo modificado de Berlekamp-Massey para anillos conmutativos con unidad dado en [**11**]. Existen también generalizaciones del algoritmo que permiten generar sucesiones múltiples [**7**], lo cual tiene aplicación

en decodificación de códigos cíclicos, además, esta idea ha sido extendida en [**10**] al caso de sucesiones múltiples sobre anillos conmutativos con unidad, lo cual permite decodificar códigos cíclicos sobre $\mathbb{Z}_{p^s}$.

Es importante destacar que en el algoritmo presentado es necesario llevar a cabo divisiones en el anillo de Galois $GR(p^s, m)$ por lo tanto, para ésta y de igual manera para otras aplicaciones son de especial interés algoritmos que permitan calcular en forma eficiente los recíprocos para las unidades en este tipo de anillos. Sin embargo, a pesar de que se han diseñado muchos algoritmos para el cálculo de recíprocos sobre campos finitos o anillos de enteros, no se ha puesto gran atención en algoritmos para anillos de Galois. En [**5**], por ejemplo, se describen tres diferentes métodos para calcular el recíproco de una unidad en $GR(p^s, m)$, pero aún es mucho el trabajo que hay hacer en estas direcciones.

Referencias

[1] M. F. Atiyah and I. G. McDonald, *Introduction to Commutative Algebra*, Addison-Wesley, 1969.

[2] G. Bini and F. Flamini, *Finite Commutative Rings and their Applications*, Kluwer Academic Publishers, Boston, 2002.

[3] Blahut, Richard E., *Theory and Practice of Error Control Codes*, Addison-Wesley, NY, 1984.

[4] Berlekamp, Elwyn R., *Algebraic Coding Theory*, McGraw-Hill, NY, 1968.

[5] M. Elia, J.C. Interlando and R. Palazzo, Jr., *Computing the Reciprocal of Units in Finite Galois Rings*, Journal of Discrete Mathemathical Sciences and Cryptography, Vol 3/4, 2000.

[6] Fraleigh, Jonh B., *A First Course in Abstrac Algebra*, Addison-Wesley, USA, 2003.

[7] G. L. Feng and K. K. Tzeng, *A Generalization of the Berlekamp-Massey Algorithm for Multisequence Shift-Register synthesis with Applications to Decoding Cyclic Codes*, IEEE Transactions on Information Theory, Vol. 37, No. 5, pp. 1274-1287, Septiembre, 1991.

[8] D. C. Gorenstein and N. Zierler, *A Class of Error-Correcting Codes in p^m Symbols*, Journal of the Society for Industrial and Applied Mathematics, Vol. 9, No. 2, pp. 207-214, Junio 1961.

[9] W.C. Huffman and V. Pless, *Fundamentals of Error-Correcting Codes*, Cambridge University Press, UK, 2003.

[10] J.C. Interlando and R. Palazzo, Jr., *Multisequence Generation and Decoding of Cyclic Codes over Z_q*, IEEE International Symposium on Information Theory, p. 277, Septiembre, 1995.

[11] J.C. Interlando, R. Palazzo, Jr. and M. Elia, *On the Decoding of Reed-Solomon and BCH Codes over Integer Residue Rings*, IEEE Transactions on Information Theory, Vol. 43, No. 3, pp. 1013-1021, Mayo 1997.

[12] P. Kanwar and S. R. López-Permouth, *Cyclic Codes over the Integers Modulo p^m*, Finite Fields and their Applications, Vol. 3, pp. 334-352, 1997.

[13] R. Lidl and H. Niederreiter, *Finite Fields*, Encyclopedia of Mathematics and Its Applications. Cambridge University Press, UK, 1997.

[14] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*, North Holland, Amsterdam, 1977.

[15] McDonald, Bernard R., *Finite Rings With Identity*, Marcel Dekker Inc., NY, 1974.

[16] McDonald, Bernard R., *Linear Algebra Over Commutative Rings*, Marcel Dekker Inc., NY, 1984.

[17] J.L. Massey, *Shift-Register Synthesis and BCH Decoding*, IEEE Transactions on Information Theory, Vol. IT-15, No. 1, pp. 122-127, Enero 1969.

[18] Peterson, W. W., *Encoding and Error-Correction Procedures for the Bose-Chaudhuri Codes*, IRE Transactions on Information Theory, Vol. IT-6, pp. 459-470, 1960.

[19] J. A. Reeds and N. J. A. Sloane, *Shift-Registers Synthesis (Modulo m)*, SIAM J. Computing, Vol. 14, pp. 505-513, 1985.

[20] I. S. Reed and G. Solomon, *Polynomial Codes over Certain Finite Fields*, Journal of the Society of Industrial and Applied Mathematics, Vol. 8, No. 2, pp. 300-304, 1960.

[21] Roman, Steven, *Coding and Information Theory*, Springer-Verlag, New York, 1992.

[22] Shankar, Priti, *On BCH Codes Over Arbitrary Integer Rings*, IEEE Transactions on Information Theory, Vol IT-25, No. 4, pp. 480-483, Julio 1979.

[23] V. Guruswami and M. Sudan, *Improved Decoding of Reed-Solomon and Algebraic-Geometry Codes*, IEEE Transactions on Information Theory, Vol. 45, No. 6, pp. 1757-1767, Septiembre 1999.

[24] Wan, Zhe-Xian, *Lectures on Finite Fields and Galois Rings*, World Scientific, USA, 2003.

UNIVERSIDAD AUTÓNOMA METROPOLITANA

El algoritmo de Berlekamp-Massey y decodificación de códigos BCH sobre el anillo $\mathbb{Z}_{p^s}$

Presenta
L.M.A. Rocío Meza Moreno

Asesor de tésis:
Dr. Horacio Tapia Recillas

Unidad Iztapalapa
Departamento de Matemáticas

México, D.F.
4 de mayo de 2007