Un recorrido por los códigos: de BCH a MCELIECE

Alejandro Muñoz Azaustre

Un recorrido por los códigos: de BCH a MCELIECE

Alejandro Muñoz Azaustre

Memoria presentada como parte de los requisitos para la obtención del título de Grado en Matemáticas por la Universidad de Sevilla.

Tutorizada por

Prof. Tutor Alberto Castaño Domínguez

Agradecimientos

Quisiera empezar este trabajo, con las clásicas palabras y receptores, agradeciendo a mis padres el apoyo que me han dado para poder seguir estudiando en otra universidad y terminar mi grado de una vez por todas. A mis amigas, Sandra, Lucía, Violeta, Melody, Mario (que no se ofenda nadie, la lista tiene que ser de rango conmensurable); que han estado empujando de mí para no tirar la toalla, algunas de ellas también leerán estas palabras y trabajo.

Y no me olvido de mi tutor, Alberto, al que el primer día que llegué a la clase de TCYC en el curso pasado sin conocerlo de nada ni conocer a nadie, le pregunté si estaría disponible y dispuesto a tutorizarme un trabajo de fin de grado; y aquí estamos. Al igual que él conmigo, yo también he estado bastante cómodo trabajando bajo su supervisión y agradeciendo infinitamente, tanto sus lecturas y aportes, como la libertad que me ha dejado en la redacción del tema y en general, la libertad que me ha dejado en todo lo que rodea al trabajo; nos hemos puesto bastante de acuerdo en todo y ha aceptado de buen grado mis ideas mejorándolas con las suyas.

Para finalizar, aunque esto es un agradecimiento de vida, no quiero olvidarme de recordar aquí a mi abuela Manoli, que se fue en noviembre de este curso. Gran parte de esta memoria ha sido escrita en los muchos trayectos que he hecho en el tren entre Málaga y Sevilla a lo largo del curso, de entre esos trayectos, los que también hice la semana que estuvo hospitalizada. «Abuela, aquí estoy cumpliendo lo que me hiciste prometerte en tu última voluntad».

Simplemente, gracias.

Índice general

Resumen

En este trabajo se realiza un estudio de los códigos desde una perspectiva algebraica pasando de las definiciones más esenciales a un estudio en profundidad de una clase concreta de códigos, los códigos BCH. También estudiamos un caso particular de ellos que son interesantes, los códigos Reed-Solomon; finalizando este trabajo con la generalización de los códigos BCH que son los códigos de Goppa. Además, vemos una utilidad de los últimos códigos que es el criptosistema de McEliece.

Abstract

In this project we do an approach to Coding theory by an algebraic view. We will start, from the beginning, by giving the most essential definitions in order to achieve enough knowledge to be able to understand one of the most important family of codes, BCH codes. Furthermore, we will study in depth a particular case of them, Reed-Solomon codes. Lastly, we will see a generalisation of BCH codes, Goppa codes in their classic version and how they are used in the McEliece Cryptosystem.

Introducción

La Teoría de Códigos es una rama de las matemáticas bastante reciente, se considera que comenzó con los estudios de Claude E. Shannon quien publicó un artículo en dos partes [Sha48a, Sha48b] en 1948, aunque se disputa el puesto de "padre" de la Teoría de Códigos con su compañero de trabajo, Richard W. Hamming, pues se dice que este escribió su artículo [Ham50] en 1947 aunque no fue publicado hasta 1950. Dejando de lado quién fue el primero, ambos iniciaron la Teoría de Códigos aunque con perspectivas un poco distintas, Shannon aportó una perspectiva estocástica y existencial; frente a la perspectiva de Hamming constructiva y combinatoria. Tampoco debemos olvidarnos de Marcel J. E. Golay en cuyo artículo [Gol49], en 1949, definió dos códigos lineales dando los coeficientes de su matriz generatriz.

La mayoría de los trabajos realizados al principio de la década de 1950 introducen bastantes conceptos que servirán de base para las distintas especialidades que se desarrollaron a partir de entonces, tales como la construcción de los códigos, descodificación en bloque, la estructura del peso, cotas para la distancia, algoritmos de descodificación... Mientras que la mayoría de los artículos escritos a partir de 1956 se pueden clasificar fácilmente dentro de una de estas especialidades, los anteriores a este año no se pueden clasificar fácilmente.

David E. Muller [Mul54] e Irving S. Reed [Ree54] no solo construyeron una clase de códigos (los códigos Reed-Muller), también asentaron las bases del estudio de la Teoría de Códigos desde una perspectiva algebraica.

David Slepian en [Sle56] expuso los fundamentos matemáticos de la Teoría de Códigos Lineales. Un libro muy interesante que recomiendo al lector más curioso es [Ber74], donde Berlekamp incluye una pequeña introducción histórica que me ha sido de gran ayuda para redactar esta parte y cuyo contenido son los artículos más relevantes de la Teoría de Códigos ordenados por aparición.

Actualmente, esta teoría es la base de todas nuestras comunicaciones digitales, por ello es tan importante el estudio de códigos capaces de mejorar los ya existentes; aunque nos deberíamos preguntar qué significa mejorar un código. Mejorar un código no es más que saber escoger el adecuado para cada situación, ya hemos visto en la asignatura de Teoría de Códigos y Criptografía las distintas ventajas e inconvenientes sobre el uso de códigos lineales y códigos cíclicos. Por ejemplo, los códigos lineales son sencillos sin apenas cálculos (solo hay que trabajar con vectores y matrices) pero su inconveniente es el espacio que hay que usar para almacenar datos, con parámetros pequeños son manejables incluso en papel pero en cuanto se aumenta la longitud, rápidamente se vuelven poco prácticos teniendo que guardar gran cantidad de datos (tablas de síndromes, matrices generatrices, etc.). Los códigos cíclicos pueden recoger mucha información en poco espacio, estamos hablando de polinomios definidos sobre cuerpos finitos; la gran ventaja es que aquí podemos aumentar los parámetros sin mayor problema (solo requerimos un polinomio generador que cumpla ciertos requisitos) pero en cambio todos los cálculos son necesarios hacerlos con un ordenador a poco que aumente el tamaño del código, es relativamente sencillo saber si un polinomio es irreducible o no sobre un cuerpo pero en cambio factorizarlo en irreducibles requiere de herramientas de cálculo.

Hay que tener cuidado también con la fina línea que separa la teoría de códigos y la criptografía, muchas veces confundidas y mezcladas inconscientemente. Los códigos se centran en el envío de la información a través de un canal y los posibles errores (ruido) que puedan ocurrir desde que el emisor la envía hasta que la recibe el receptor. La criptografía se centra en evitar que el mensaje enviado sea leído fácilmente por alguien que lo intercepte a mitad de camino, en ningún momento preocupa si el mensaje ha llegado correctamente; es decir, sin errores.

Es más, hasta Shannon en [Sha49] realiza un estudio sobre criptografía, el artículo aparecía originalmente en un informe secreto llamado "A Mathematical Theory of Cryptography" fechado a 1 de septiembre de 1945; es decir, previo a la publicación de sus artículos de códigos. Por lo tanto, es evidente la preocupación tanto en la seguridad de la transmisión como en los errores que se produzcan en el canal.

Pocos años después, como ya veremos en este trabajo, la familia de códigos Goppa llevaron a ser la base del criptosistema de McEliece. Aquí desaparece la delgada línea fusionando ambas teorías y dejando un problema al que actualmente se le busca solución: ¿es el sistema de McEliece suficientemente seguro frente a los ordenadores cuánticos?

Este trabajo está estructurado en un capítulo inicial donde fijamos las notaciones y definiciones ya vistas en la asignatura de TCYC, además de añadir conceptos necesarios para el desarrollo de los capítulos siguientes que no han sido vistos en la asignatura.

Un segundo capítulo en el que tratamos de forma detallada los códigos BCH y los códigos Reed-Solomon. Explotamos en profundidad las propiedades de los códigos cíclicos que, bajo ciertos parámetros, son realmente útiles y generan muchos resultados que se aplican en la actualidad.

El último capítulo que presentamos aquí, es un inicio para un posible trabajo que continúe con el estudio que llevo estos años. Tratamos los códigos de Goppa que son la base que sustenta el sistema criptográfico de McEliece. Nos quedaremos en los códigos clásicos de Goppa que son los que podemos estudiar desde una perspectiva algebraica en consonancia con este trabajo, sin olvidar mencionar que los códigos de Goppa son también códigos algebraico-geométricos y estos pueden ser estudiados a partir de la geometría algebraica, una perspectiva totalmente distinta a la de este trabajo.

Para el contenido del trabajo he usado fundamentalmente:

- Capítulo 1: [MA18] (se puede consultar pinchando aquí) y las notas de clase del curso 2021/2022 de TCYC. En mi anterior trabajo de fin de grado se puede encontrar una gran variedad de referencias básicas en el campo de la teoría de códigos lineales y cíclicos, además de los resultados más importantes. Por ello, y por lo visto en las clases de TCYC, en este nuevo trabajo hemos pasado por alto muchas de las demostraciones por no aportar nada al grueso del contenido.
- Capítulo 2: [MS77] y [MT97] han sido las referencias fundamentales. Ocasionalmente, he consultado [McE04] para algunas referencias sobre códigos cíclicos y BCH. Se puede observar que [MT97] es una versión más moderna y traducida de muchos de los resultados expuestos en [MS77] para la parte de códigos BCH. Por otro lado, la única referencia que ha tratado de forma general los códigos Reed-Solomon ha sido [MS77]. Este capítulo absorbe el grueso del trabajo que era nuestro objetivo real, desarrollar la teoría de una familia de códigos correctores de errores a partir de propiedades algebraicas.
- Capítulo 3: [Sin19] es un artículo que encontramos bastante correcto en tanto a que, resumidamente, nos lleva a definir el criptosistema de McEliece usando los códigos de Goppa sin entrar en grandes detalles. Estos detalles pueden tener la envergadura de otro trabajo de fin de grado nuevo, como algunos que he visto en internet de otras universidades. Nos limitamos a exponer conceptos que sirvan de antesala a otro proyecto.

Estas no son las únicas referencias con las que he trabajado, puntualmente a lo largo del trabajo he ido refiriéndome a distintos artículos en los que consultar las notas originales de los autores cuyos resultados estudiamos. En ocasiones, estas consultas han sido de gran ayuda puesto que algunas traducciones han ido perdiendo contenido y han dado lugar a dudas matemáticas.

La intención de este trabajo ha sido desarrollar el tema que tratamos de forma constructiva y que cualquier estudiante (con unos conocimientos mínimos, pero alcanzables a nivel de 4º curso de Grado) sea capaz de seguir, entender y aprender algo nuevo que complemente a la asignatura de TCYC.

Alejandro Muñoz Azaustre

1 Códigos lineales y cíclicos

Este capítulo es un recordatorio sobre los conceptos y resultados vistos en la asignatura de *Teoría de Códigos y Criptografía* del tercer curso del Grado en Matemáticas con el que asentar la notación y los resultados necesarios para así poder construir el resto del trabajo. En este capítulo usaré las referencias [MA18, Teoría de Códigos], que es mi anterior TFG realizado, y las notas de clase de la asignatura de TCYC; por ello, pasaremos por alto muchas de las demostraciones de los resultados.

1.1 Conceptos previos

En esta sección vamos a tratar las definiciones y resultados más básicos sobre los que se sustenta cualquier familia de códigos.

Definición 1.1. Se define un alfabeto $\mathcal{A}$ como un conjunto de símbolos sin significado individual. Es decir, pueden ser números, letras o cualquier tipo de carácter a los que les podamos aplicar estos conceptos.

Definición 1.2. Un código $\mathcal{C}$ sobre un alfabeto $\mathcal{A}$ es un conjunto de secuencias de elementos de $\mathcal{A}$. A los elementos de $\mathcal{C}$ se les llama palabras código. Un código en bloque de n caracteres no es más que un subconjunto de $\mathcal{A}^n$, es decir, un conjunto de palabras de exactamente n letras en el alfabeto $\mathcal{A}$.

Definición 1.3. Dado un código $\mathcal{C} \subset \mathcal{A}^n$, llamamos:

- Longitud del código al número n.
- Tamaño del código al número $|\mathcal{C}|$.

Ahora pasaremos a definir una métrica que será la que usemos durante todo el trabajo, ésta es la asociada a la siguiente distancia:

Definición 1.4. Definimos la métrica o distancia de Hamming en $\mathcal{A}^n$ como:

$$\begin{array}{rccc} d & : & \mathcal{A}^n \times \mathcal{A}^n & \longrightarrow & \mathbb{R}^+ \\ & & (x,y) & \mapsto & |\{i \,/\, x_i \neq y_i,\ i = 1,\ldots,n\}| \end{array} .$$

Es decir, el número de componentes en las que difieren las dos palabras cuya distancia queramos medir.

Para ver que es una distancia basta comprobar las tres propiedades de la definición. Es claro que $d(x,x) = 0$ ya que no hay ninguna componente distinta entre las dos palabras, que es positiva también se ve rápidamente ya que estamos contando componentes en las que difieren las dos y la propiedad más laboriosa (pero no por ello imposible) es la triangular.

Definición 1.5. Sea $|\mathcal{A}| = q$. Dado un código $\mathcal{C} \subset \mathcal{A}^n$, llamamos:

- Distancia mínima del código al número

$$d(\mathcal{C}) = \min\{d(x,y) \,/\, \forall\, x,y \in \mathcal{C}\}.$$

- Distancia mínima relativa del código al número

$$\delta(\mathcal{C}) = \frac{d(\mathcal{C})}{n}.$$

- Tasa de transmisión de información del código al número

$$R(\mathcal{C}) = \frac{\log_q(|\mathcal{C}|)}{n}.$$

- Redundancia del código al número

$$n - \log_q(|\mathcal{C}|).$$

Normalmente diremos que $\mathcal{C}$ es un código tipo (n,m,d) para decir que es un código de longitud n, tamaño m y distancia mínima d. Las demás constantes relacionadas con el código, las que se suelen denotar por parámetros del código, se pueden calcular a partir de estas tres (conociendo $\mathcal{A}$).

Definición 1.6. Para cada $x \in \mathcal{C}$ se define la función peso $w(x)$ como el número de componentes no nulas de x, es decir,

$$w(x) = |\{i \,/\, x_i \neq 0,\ 1 \leq i \leq n\}|.$$

Claramente, definiremos el peso mínimo de un código $\mathcal{C}$ como $\min\limits_{x \neq 0} w(x)$.

Esta última definición nos lleva a una relación con la distancia de Hamming bastante útil en algunas ocasiones, que veremos en la próxima sección.

Definición 1.7. [MS77, Pág. 11] Se define un canal simétrico $q-$ario como un canal que tiene q símbolos de entrada y q símbolos de salida, con una probabilidad $1-p > \frac{1}{q}$ de que no ocurra ningún error en la transimisión donde cada uno de los $q-1$ posibles errores son equiprobables.

Para corregir errores hemos de establecer las condiciones del canal con la probabilidad de que se produzca un error. Normalmente se trabaja con canales simétricos.

- La probabilidad $p < 1 - \frac{1}{q}$ de que ocurra un error es independiente de la posición del símbolo en el que se produzca.
- Los errores son independientes, es decir, un error en una partición del mensaje no afecta a otras posiciones.

Muchos códigos usan el método de corrección del vecino más próximo. Este método lo que hace es, una vez recibida una palabra, si esta no está en el código elige la palabra código más cercana a la recibida. Es un método que maximiza la probabilidad de corregir errores cuando el canal es simétrico.

Teorema 1.8. *Sea $\mathcal{C}$ un código y sea d su distancia mínima. Entonces:*

1. *Puede detectar hasta s errores de cualquier palabra código si $d(\mathcal{C}) \geq s+1$.*
2. *Puede corregir hasta t errores de cualquier palabra código si $d(\mathcal{C}) \geq 2t+1$.*

Corolario 1.9. *Si un código $\mathcal{C}$ tiene distancia mínima d entonces $\mathcal{C}$ puede usarse para:*

1. *Detectar hasta $d-1$ errores.*
2. *Corregir hasta $\left\lfloor \frac{d-1}{2} \right\rfloor$ errores en cualquier palabra código.*

Definición 1.10. Un código $\mathcal{C}$ se dice que es perfecto si existe t tal que para cada $x \in \mathcal{A}^n$ hay una única palabra código $c \in \mathcal{C}$ tal que $\mathrm{d}(x,c) \leq t$.

Y con esto terminamos el primer apartado de definiciones que valen para cualquier familia de códigos. Ahora pasaremos a la sección de códigos lineales, donde nuestro alfabeto pasará a ser un cuerpo finito.

1.2 Códigos lineales

En esta sección nos vamos a centrar en códigos cuyo alfabeto es $\mathcal{A} = \mathbb{F}_q$, con $q = p^m$ siendo p un número primo. Consideraremos $\mathcal{A}^n = \mathbb{F}_q^n$ como espacio vectorial, donde notaremos sus vectores como filas.

Definición 1.11. Un código lineal $\mathcal{C}$ de longitud n sobre el alfabeto $\mathcal{A} = \mathbb{F}_q$ es un subespacio vectorial de $\mathbb{F}_q^n$.

Observación 1.12. Un código lineal $\mathcal{C} \subset \mathbb{F}_q^n$ tiene, por ser subespacio vectorial, una dimensión que notaremos por $\dim(\mathcal{C}) = k$. Esta definición es clara ya que partimos de $\mathcal{C}$ un código tipo $\left(n, q^k, d\right)$ con $R(\mathcal{C}) = \frac{k}{n}$ y redundancia $n - k$; es decir, de las n coordenadas que tiene cada palabra de $\mathcal{C}$, k contienen información y esto evidencia que la dimensión sea k.

La notación de $\left(n, q^k, d\right)$ se referirá a códigos en general, cuando nos refiramos a lineales cambiaremos los paréntesis por corchetes y si desde el principio conocemos o fijamos el alfabeto, relajaremos la notación por $[n, k, d]$; en caso de no requerir (o desconocer a priori) la distancia mínima, escribiremos $[n, k]$. Aunque aquí, [Hil86, Pág. 47] tiene a bien advertirnos que no todos los $\left(n, q^k, d\right)$ −códigos son un $[n, k, d]$ −código; hemos suprimido la hipótesis y concepto de lineal, por eso falla.

Lema 1.13. *Si $x,\ y \in \mathcal{A}^n$, entonces*

$$d(x, y) = w(x - y)$$

donde d denota la distancia de Hamming y w el peso.

Demostración. El vector $x - y$ tiene componentes no nulas justamente donde x e y difieren, aplicando la definición de d obtenemos la prueba. ∎

Teorema 1.14. *Sea $\mathcal{C}$ un código lineal tipo $\left(n, q^k, d\right)$, entonces*

$$d = \min_{0 \neq x \in \mathcal{C}} w(x).$$

Ya hemos definido un código lineal como un subespacio vectorial, como las palabras del código son vectores, podemos encontrar una base del espacio vectorial.

Definición 1.15. Sea $\{g_1, \ldots, g_k\}$ una base de $\mathcal{C}$. Esta base consiste en k vectores linealmente independientes, recordemos que $\dim(\mathcal{C}) = k$ y las palabras (o vectores) $g_i \in \mathcal{C}$ tienen longitud n.

Definición 1.16. Sea la matriz G de la siguiente forma

$$G = \begin{pmatrix} g_1 \\ \vdots \\ g_k \end{pmatrix} = \begin{pmatrix} g_{1,1} & \cdots & g_{1,n} \\ \vdots & \ddots & \vdots \\ g_{k,1} & \cdots & g_{k,n} \end{pmatrix} \in \mathcal{M}_{k\times n}\left(\mathbb{F}_q\right)$$

Como todas las palabras código $x \in \mathcal{C}$ se pueden generar a partir de G, a esta matriz se le llama la matriz generatriz del código lineal $\mathcal{C}$.

Recordemos que $n \geq k$, con lo que llegamos a la siguiente definición bastante útil en la práctica.

Definición 1.17. Sea G una matriz generatriz de un $[n, k]$ −código. Entonces actuando sobre G con operaciones elementales podemos transformarla en la forma estándar

$$G = (I_k|A)$$

donde I_k es la matriz identidad de dimensión k y A es una matriz $k \times (n - k)$.

Definición 1.18. Una matriz de control H para un $[n, k]$ −código $\mathcal{C}$ es una matriz $(n - k) \times n$ que satisface $G \cdot H^t = 0$, donde 0 describe la matriz donde todas sus componentes son 0. Por tanto, un código lineal se puede definir también a partir de su matriz de control

$$\mathcal{C} = \left\{x \in \mathcal{A}^n \,/\, x\, H^t = 0\right\},$$

de esta forma queda completamente determinado.

Definición 1.19. Se definen las ecuaciones de control como aquellas que resultan del sistema $x \cdot H^t = 0$, siendo $x \in \mathbb{F}_q^n$ un vector de coordenadas.

Teorema 1.20. *Sea $G = (I_k|A)$ una matriz generatriz en forma estándar de un $[n, k]$ −código, entonces una matriz de control de $\mathcal{C}$ es $H = (-A^t|I_{n-k})$.*

Observación 1.21. Hago aquí una observación, que seguro no es necesaria para alguien que vea evidente lo siguiente, hemos dicho que las matrices G y H son «una» y no «la» matriz ya que cualquier operación elemental sobre ellas nos devuelven otras matrices que generan un código equivalente al anterior. Y por tanto, un código equivalente no es más que el mismo código lineal de partida en el que hemos realizado una permutación en las posiciones del código o hemos multiplicado los símbolos de ciertas posiciones fijadas por un escalar no nulo.

Ahora paso a definir un concepto que nos resultará útil tener a mano, el concepto de código dual, que es análogo a la definición del ortogonal de un espacio vectorial sobre $\mathbb{R}$. Para ello, necesitamos definir previamente el producto escalar que vamos a utilizar:

Definición 1.22. Se define el producto escalar de dos vectores $u, v \in \mathbb{F}_q^n$ como el escalar (perteneciente a $\mathbb{F}_q$) que resulta de la forma bilineal, simétrica:

$$u \cdot v = u_1 v_1 + \ldots + u_n v_n$$

En el caso en el que $u \cdot v = 0$ diremos que son ortogonales.

Nótese que lo denominamos producto escalar por la operación que hacemos, y la similitud al usual en $\mathbb{R}$, pero es importante resaltar que en $\mathbb{F}_q$ no tenemos asegurado que sea no degenerado.

Definición 1.23. Dado un $[n, k]$ −código $\mathcal{C}$, se define el código dual de $\mathcal{C}$ y se denota por $\mathcal{C}^\perp$ al conjunto de los vectores de $\mathbb{F}_q^n$ que son ortogonales a cualquier palabra código de $\mathcal{C}$, es decir:

$$\mathcal{C}^\perp = \left\{ v \in \mathbb{F}_q^n \,/\, v \cdot u = 0 \ \ \forall\, u \in \mathcal{C} \right\}$$

Lema 1.24. *Supongamos $\mathcal{C}$ un $[n, k]$ −código con matriz generatriz G. Entonces un vector $v \in \mathbb{F}_q^n$ pertenece a $\mathcal{C}^\perp$ si y solo si v es ortogonal a cada fila de G; es decir,*

$$v \in \mathcal{C}^\perp \iff v\, G^t = 0$$

Demostración. ($\Leftarrow$) Esta implicación es trivial pues las filas de G son palabras código.
($\Rightarrow$) Supongamos que las filas de G son $g_1, g_2, ..., g_k$ y que $v \cdot g_i = 0$ para cada i. Si u es otra palabra código cualquiera de $\mathcal{C}$, entonces $u = \sum_{i=1}^{k} \lambda_i g_i$ con $\lambda_i \in \mathbb{F}_q$, luego:

$$v \cdot u = \sum_{i=1}^{k} \lambda_i (v \cdot g_i) = \sum_{i=1}^{k} \lambda_i 0 = 0$$

Por tanto, v es ortogonal a cualquier palabra código de $\mathcal{C}$ y también lo es en $\mathcal{C}^\perp$. ∎

Teorema 1.25. *Sea $\mathcal{C}$ un $[n, k]$ −código sobre $\mathbb{F}_q$. Entonces el código dual $\mathcal{C}^\perp$ de $\mathcal{C}$ es un $[n, n-k]$ −código lineal.*

Demostración. Veamos en primer lugar que $\mathcal{C}^\perp$ es un código lineal.

Sean $v_1, v_2 \in \mathcal{C}^\perp$ y $\lambda, \mu \in \mathbb{F}_q$. Entonces, para cada $u \in \mathcal{C}$:

$$(\lambda v_1 + \mu v_2) \cdot u = \lambda(v_1 \cdot u) + \mu(v_2 \cdot u) = \lambda \cdot 0 + \mu \cdot 0 = 0$$

En consecuencia $\lambda v_1 + \mu v_2 \in \mathcal{C}^\perp$ y queda probado que $\mathcal{C}^\perp$ es lineal.

Veamos ahora que $\mathcal{C}^\perp$ tiene dimensión $n - k$.

$\mathcal{C}^\perp$ es un subconjunto de $\mathbb{F}_q^n$ cuyas ecuaciones implícitas vienen dadas por G. Por tanto, es un subespacio vectorial de dimensión $n - k$ en virtud del Lema 1.24.

Corolario 1.26. *Una matriz de control H para un $[n, k]$ −código $\mathcal{C}$ es una matriz generatriz de $\mathcal{C}^\perp$.*

Teorema 1.27. *Para cualquier $[n, k]$ −código $\mathcal{C}$ se tiene que $(\mathcal{C}^\perp)^\perp = \mathcal{C}$.*

Demostración. Claramente $\mathcal{C} \subset (\mathcal{C}^\perp)^\perp$ puesto que cada vector de $\mathcal{C}$ es ortogonal a cualquier vector en $\mathcal{C}^\perp$. Además, $\dim((\mathcal{C}^\perp)^\perp) = n - (n - k) = k = \dim(\mathcal{C})$, por tanto nos queda que $\mathcal{C} = (\mathcal{C}^\perp)^\perp$.

Definición 1.28. Un $[n, k]$ −código $\mathcal{C}$ que satisface $\mathcal{C}^\perp = \mathcal{C}$ se dice que es autodual.

Proposición 1.29. *La distancia mínima de $\mathcal{C}$, d, es la menor cantidad de columnas de H que necesitamos para formar un conjunto linealmente dependiente.*

La estructura extra de espacio vectorial que nos proporcionan los códigos lineales nos permite una descodificación sistemática siguiendo el principio de distancia mínima. Supongamos que tenemos un código lineal $\mathcal{C} \subset \mathbb{F}_q^n$ con matriz de control H y que hemos enviado una palabra $x \in \mathcal{C}$, pero recibimos

$$y = x + e,$$

donde e es el error en la transmisión.

Definición 1.30. Supongamos que H es la matriz de control de un $[n, k]$ −código $\mathcal{C}$. Entonces para cualquier vector $y \in \mathcal{A}^n$, al vector fila de longitud $n - k$

$$S(y) = y \cdot H^t$$

se le llama síndrome de y.

Observación 1.31. Algunas propiedades del síndrome son:

- $S(z) = 0 \iff z \in \mathcal{C}$
- Si $y = x + e$, entonces $S(y) = S(e)$.
- El síndrome de un vector es una combinación lineal de las columnas de H correspondientes a los lugares en los se ha cometido un error de transmisión.

En realidad, podemos ver que el síndrome no es más que la aplicación lineal

$$S : \mathbb{F}_q^n \longrightarrow \mathbb{F}_q^{n-k}$$

definida por la matriz H, que verifica $\ker(S) = \mathcal{C}$. Esto debería darnos una pista sobre la próxima observación, ya que vamos a poder definir clases de equivalencia a partir de la aplicación.

Observación 1.32. Dado un código lineal $\mathcal{C} \subset \mathbb{F}_q^n$, de tamaño q^k, consideramos el espacio $\mathbb{F}_q^n/\mathcal{C}$, que es un espacio vectorial de dimensión $n - k$. Sus elementos son las clases

$$x + \mathcal{C} = \{x + u \,|\, u \in \mathcal{C}\}$$

cada una de ellas formadas por q^k elementos. Dos vectores u, v están en la misma clase si y solo si

$$u - v \in \mathcal{C} \iff S(u) = S(v).$$

Por tanto, en los elementos de una clase aparecen los vectores que provienen de cometer el mismo error de transmisión.

Definición 1.33. Diremos que un elemento u es líder de una clase $\mathbb{F}_q^n/\mathcal{C}$ cuando sea el único elemento de peso mínimo.

Corolario 1.34. *Hay una correspondencia biyectiva entre clases y síndromes.*

Teorema 1.35. *Sea $\mathcal{C}$ un $[n, k]-$código. El código puede corregir t errores si y solo si todas las palabras de peso t son también líderes de la clase correspondiente.*

La demostración del teorema no nos aporta nada en este trabajo, es meramente aplicar las definiciones y propiedades expuestas.

Para terminar la sección, vamos a detallar un ejemplo de uso de un código lineal bastante sencillo:

Ejemplo 1.36. En este ejemplo, basado en el ejemplo de [MP15, Pág. 12], vamos a aplicar a un caso real los códigos. Supongamos que tenemos un robot en otro planeta, está claro que tenemos que tener un código con capacidad de corregir errores puesto que nos arriesgamos a perder un aparato muy costoso. El robot tendrá cuatro instrucciones para desplazarse, estas las codificaremos de la siguiente forma:

$$\mathcal{C}_0 = \begin{cases} \uparrow & \mapsto 00, \\ \rightarrow & \mapsto 01, \\ \downarrow & \mapsto 11, \\ \leftarrow & \mapsto 10. \end{cases}$$

Podemos considerar como alfabeto $\mathbb{F}_2$ y vamos a añadirle redundancia para corregir errores de forma automática, consideremos pues que las palabras tengan longitud 5, entonces nuestro código será el siguiente:

$$\mathcal{C} = \begin{cases} \uparrow & \mapsto 00\underline{000}, \\ \rightarrow & \mapsto 01\underline{101}, \\ \downarrow & \mapsto 11\underline{011}, \\ \leftarrow & \mapsto 10\underline{110}. \end{cases}$$

Tenemos la matriz generatriz:

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

Y la matriz de control H:

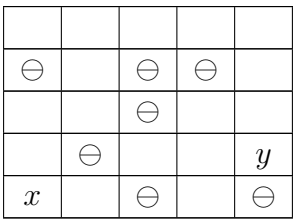
$$H = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \end{pmatrix}.$$

$\mathcal{C} = \{00000, 01101, 11011, 10110\} \subset \mathbb{F}_2^5$, $|\mathcal{C}| = 2^k = 4$ tenemos que $k = 2$, por tanto $\mathcal{C}$ es un $(5, 2)-$código lineal. Claramente se ve que la distancia mínima de Hamming de las palabras es $d = 3$. Por el Corolario 1.9 el código detecta hasta 2 errores y puede corregir 1 error. Para calcular los síndromes, en primer lugar necesitamos conocer el

número de clases que tenemos, a saber son $q^{n-k} = 2^3 = 8$. Estas clases son:

Vectores					Síndrome
00000	00000	11011	10110	01101	$S(00000) = 000$,
10000	10000	01011	00110	11101	$S(10000) = 110$,
01000	01000	10011	11110	00101	$S(01000) = 101$,
00100	00100	11111	10010	01001	$S(00100) = 100$,
00010	00010	11001	10100	01111	$S(00010) = 010$,
00001	00001	11010	10111	01100	$S(00001) = 001$,
11000	11000	00011	01110	10101	$S(11000) = 011$,
11100	10001	00111	01010	11100	$S(10001) = 111$.

Y de aquí, supongamos que queremos que el robot se mueva desde el punto x al punto y:



Por tanto, el camino que debe seguir el robot es:

$$\uparrow\uparrow\rightarrow\uparrow\uparrow\rightarrow\rightarrow\rightarrow\downarrow\downarrow\downarrow$$

El mensaje que le transmitiremos es:

$$00000\,00000\,01101\,00000\,00000\,01101\,01101\,01101\,11011\,11011\,11011$$

He separado levemente cada palabra enviada para que sea más fácil su lectura, el emisor envía la cadena completa y el receptor la dividirá en palabras de longitud 5. Supongamos ahora que el robot recibe en la última palabra $w = 10011$.

Calculamos $S(10011) = 101$ y buscamos al líder de la clase con ese síndrome, que es 01000. Ahora calculamos $v = w - 01000 = 10011 - 01000 = 11011$, por tanto, hemos corregido el error correctamente y el robot llegará al punto y.

1.3 Códigos Cíclicos

Los códigos cíclicos son un caso particular de los códigos lineales, ya que en los cíclicos trabajamos con polinomios y los definiremos sobre anillos que serán de la forma $\mathbb{F}_q[X]/\langle X^n - 1\rangle$.

Definición 1.37. Un código cíclico es un código lineal $\mathcal{C} \subset \mathbb{F}_q^n$ que verifica la siguiente propiedad:

$$(x_1, \ldots, x_n) \in \mathcal{C} \Longrightarrow (x_n, x_1, \ldots, x_{n-1}) \in \mathcal{C}.$$

Es decir, si un vector pertenece al código, cualquier permutación cíclica de sus componentes también pertenece al código.

Nota 1.38. Por razones que nos serán evidentes al final de la sección, cambiaremos la notación habitual de los elementos de $\mathbb{F}_q^n$ por la siguiente:

$$\mathbf{x} = \overrightarrow{x} = \underline{x} = (x_0, x_1, \ldots, x_{n-1})\,.$$

He escrito el vector x de las diversas formas en las que se puede encontrar escrito en las referencias, en general si no hay lugar a dudas, obviaré cualquier notación de vector por simplificar la lectura y escritura; cuando sea necesario marcaré las diferencias.

Nota 1.39. Hemos de suponer que $\text{mcd}(n, q) = 1$ si queremos poder aplicar las propiedades más potentes de códigos cíclicos.

La interpretación de los códigos cíclicos se hace en dos contextos distintos, que normalmente se usan de manera indiferenciada:

- La interpretación en el espacio vectorial de los polinomios:
  Identificamos $\mathbb{F}_q^n$ con $\mathbb{F}_q[X]_{\leq n-1}$, el conjunto de los polinomios sobre $\mathbb{F}_q$ de grado menor o igual que $n-1$. Esta identificación es explícitamente:

$$u = (u_0, u_1, \ldots, u_{n-1}) \longleftrightarrow u_0 + u_1 X + \ldots + u_{n-1} X^{n-1}.$$

- La interpretación en el anillo cociente de polinomios:
  Alternativamente, identificamos $\mathbb{F}_q^n$ con el anillo $\mathbb{F}_q[X]/\langle X^n - 1\rangle$, usando las congruencias habituales que no son más que polinomios de grado menor o igual que $n-1$.

Huelga decir que ambas identificaciones son un isomorfismo de espacios vectoriales.

Todos estos resultados nos llevan a la siguiente proposición:

Proposición 1.40. *Para un código cíclico $\mathcal{C}$ es equivalente:*

1. $u(X) \in \mathcal{C}$.
2. $X \cdot u(X) \mod X^n - 1 \in \mathcal{C}$.

Consideraremos trabajar en el anillo cociente $\mathbb{F}_q[X]/\langle X^n-1\rangle$, esto nos aporta una gran ventaja ya que al ser $X^n \equiv 1 \mod (X^n - 1)$, podemos reducir cualquier polinomio módulo X^n-1 simplemente reemplazando X^n por 1, X^{n+1} por X y así sucesivamente.

El siguiente teorema caracteriza algebraicamente los códigos cíclicos:

Teorema 1.41. *Un código $\mathcal{C} \subset \mathbb{F}_q[X]/\langle X^n - 1\rangle$ es un código cíclico si y solo si $\mathcal{C}$ satisface las siguientes condiciones:*

1. *Si $a(X), b(X) \in \mathcal{C}$ entonces $a(X) + b(X) \in \mathcal{C}$.*
2. *Si $a(X) \in \mathcal{C}$ y $r(X) \in \mathbb{F}_q[X]/\langle X^n - 1\rangle$ entonces $r(X) \cdot a(X) \in \mathcal{C}$.*

Es decir, los códigos cíclicos son precisamente los ideales del anillo $\mathbb{F}_q[X]/\langle X^n - 1\rangle$.

Veamos ahora una forma sencilla de construir ejemplos de códigos cíclicos:

Notación 1.42. Sea $f(X)$ un polinomio cualquiera en $\mathbb{F}_q[X]/\langle X^n - 1\rangle$ y notemos por $\langle f(X)\rangle$ el ideal generado por $f(X)$, es decir,

$$\langle f(X)\rangle = \{r(X)f(X) \,|\, r(X) \in \mathbb{F}_q[X]/\langle X^n - 1\rangle\}\,.$$

Definición 1.43. Para cualquier polinomio $f(X) \in \mathbb{F}_q[X]/\langle X^n - 1\rangle$, el conjunto $\langle f(X)\rangle$ es un código cíclico. Este conjunto se llama el código generado por $f(X)$.

Ahora veremos que la forma anterior de construir fácilmente códigos cíclicos es esencialmente la única forma, es decir, cualquier código cíclico se genera mediante un polinomio. Dicho de otra forma más algebraica, cualquier ideal en $\mathbb{F}_q[X]/\langle X^n - 1\rangle$ es un ideal principal.

Teorema 1.44. *Sea $\mathcal{C}$ un código cíclico no nulo en $\mathbb{F}_q[X]/\langle X^n - 1\rangle$. Entonces:*

1. *Existe un único polinomio mónico $g(X)$ de grado mínimo en $\mathcal{C}$.*
2. *$\mathcal{C} = \langle g(X)\rangle$.*
3. *$g(X)$ es un factor de $X^n - 1$.*

Demostración.

1. Supongamos que $g(X)$ y $h(X)$ son polinomios mónicos en $\mathcal{C}$ de grado mínimo. Entonces, $g(X) - h(X) \in \mathcal{C}$ y tiene grado menor que $g(X)$ y $h(X)$. Esto nos deja una contradicción ya que si $g(X) \neq h(X)$, existe un escalar cuyo producto por $g(X) - h(X)$ consigue que sea mónico, que esté en $\mathcal{C}$ y de un grado inferior a $g(X)$. Por tanto, $g(X) = h(X)$.
2. Supongamos $a(X) \in \mathcal{C}$. Aplicando el algoritmo de la división en $\mathbb{F}_q[X]$, $a(X) = q(X)g(X)+r(X)$, con $\deg(r(X)) < \deg(g(X))$. Pero $r(X) = a(X)-q(X)g(X) \in \mathcal{C}$, por las propiedades vistas en el Teorema 1.41. Usando que $\deg(g(X))$ es el mínimo (de los $\neq 0$), no nos queda otra que $r(X) = 0$ y por consiguiente $a(X) \in \langle g(X) \rangle$.
3. Aplicando el algoritmo de la división de nuevo, $X^n - 1 = q(X)g(X) + r(X)$ donde $\deg(r(X)) < \deg(g(X))$. Es decir, $r(X) \equiv -q(X)g(X) \mod (X^n - 1)$ y por tanto, $r(X) \in \langle g(X) \rangle$. Usando de nuevo que $\deg(g(X))$ es mínimo, tenemos que $r(X) = 0$ y en consecuencia, $g(X)$ es un factor de $X^n - 1$.

Tras estos resultados, podemos realizar la siguiente observación:

Dado que en $\mathbb{F}_q[X]/\langle X^n-1 \rangle$ todos los ideales son principales, bajo las mismas hipótesis del teorema, queda probado que podemos identificar el código $\mathcal{C}$ como $\mathcal{C} = \langle g(X) \rangle$. Dicho lo cual, nos lleva a poder definir el concepto de base de $\mathcal{C}$ como espacio vectorial, esta la obtenemos en la demostración del siguiente teorema:

Teorema 1.45. *Supongamos que $\mathcal{C}$ es un código cíclico con polinomio generador*

$$g(X) = g_0 + g_1 X + \ldots + g_{n-k} X^{n-k}$$

de grado $n - k$. Entonces $\dim(\mathcal{C}) = k$ y una matriz generatriz para $\mathcal{C}$ es:

$$G = \begin{pmatrix} g_0 & g_1 & g_2 & \cdots & g_{n-k} & 0 & 0 & \cdots & 0 \\ 0 & g_0 & g_1 & g_2 & \cdots & g_{n-k} & 0 & \cdots & 0 \\ 0 & 0 & g_0 & g_1 & g_2 & \cdots & g_{n-k} & & \vdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & 0 & g_0 & g_1 & g_2 & \cdots & g_{n-k} \end{pmatrix}$$

Corolario 1.46. *Sea $\mathcal{C} \subset \mathbb{F}_q[X]/\langle X^n - 1 \rangle$ un código cíclico con $\deg(g(X)) = n - k$. Entonces*

$$\mathcal{B} = \{g(X), Xg(X), \ldots, X^{k-1}g(X)\}$$

es una base de $\mathcal{C}$ como espacio vectorial.

La matriz generatriz dada en el Teorema 1.45 no está dada en forma estándar. La forma usual de escribir una matriz generatriz G en forma estándar para un código lineal no es la apropiada para códigos cíclicos. No obstante, hay una forma natural de matriz de control para un código cíclico. Esta forma está íntimamente relacionada con el «polinomio de control».

Definición 1.47. Sea $\mathcal{C}$ un $(n,k)-$código cíclico con polinomio generador $g(X)$. Por el Teorema 1.44, $g(X)$ es un factor de X^n-1 y por tanto

$$X^n - 1 = g(X) \cdot h(X)$$

para algún polinomio $h(X)$. Como $g(X)$ es mónico, $h(X)$ también lo es. Por el Teorema 1.45, $\deg(g(X)) = n-k$, luego $\deg(h(X)) = k$. El polinomio $h(X)$ es conocido como el polinomio de control de $\mathcal{C}$.

Teorema 1.48. *Sea $\mathcal{C}$ un código cíclico en $\mathbb{F}_q[X]/\langle X^n-1\rangle$ con polinomio generador $g(X)$ y polinomio de control $h(X)$. Entonces $c(X) \in \mathbb{F}_q[X]/\langle X^n-1\rangle$ es una palabra código de $\mathcal{C}$ si y solo si $c(X)\cdot h(X) = 0$.*

El teorema que acabamos de ver y que $\dim(\langle h(X)\rangle) = n-k = \dim(\mathcal{C}^\perp)$ nos podrían llevar a pensar que $h(X)$ genera el código dual $\mathcal{C}^\perp$. En general esto no es así. La clave está en que el producto de $c(X)$ con $h(X)$ sea cero en $\mathbb{F}_q[X]/\langle X^n-1\rangle$, pero esto no significa que como vectores de $\mathbb{F}_q^n$ sean ortogonales. No obstante, vamos a ver ahora que la condición $c(X)h(X) = 0$ en $\mathbb{F}_q[X]/\langle X^n-1\rangle$ nos lleva a algunas relaciones de ortogonalidad escogiendo de una forma natural la matriz de control.

Teorema 1.49. *Sea $\mathcal{C}$ un $(n,k)-$código cíclico con polinomio de control*

$$h(x) = h_0 + h_1 X + \ldots + h_k X^k.$$

Entonces:

1. *Una matriz de control para $\mathcal{C}$ es:*

$$H = \begin{pmatrix} h_k & h_{k-1} & \cdots & \cdots & h_0 & 0 & 0 & \cdots & 0 \\ 0 & h_k & h_{k-1} & \cdots & \cdots & h_0 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & & & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & & & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & h_k & k_{k-1} & \cdots & \cdots & h_0 \end{pmatrix}.$$

2. *$\mathcal{C}^{\perp}$ es un código cíclico generado por el polinomio:*

$$\overline{h}(X) = h_k + h_{k-1}X + ... + h_0X^k.$$

Demostración.

1. Se ha visto en TCYC.
2. Si conseguimos probar que $\overline{h}(X)$ es un factor de $X^n - 1$, aplicando el Teorema 1.44, obtendremos que $\langle \overline{h}(X) \rangle$ es un código cíclico cuya matriz generatriz es H y consecuentemente $\langle \overline{h}(X) \rangle = \mathcal{C}^{\perp}$. Si nos fijamos en que $\overline{h}(X) = X^k h(X^{-1})$ y usando que $h(X^{-1})g(X^{-1}) = (X^{-1})^n - 1$, tenemos que $X^k h(X^{-1})X^{n-k}g(X^{-1}) = X^n(X^{-n} - 1) = 1 - X^n$. Luego, en efecto, $\overline{h}(X)$ es un factor de $X^n - 1$.

Nota 1.50.

- Al polinomio $\overline{h}(X) = X^k h(X^{-1}) = h_k + h_{k-1}X + ... + h_0X^k$ habitualmente se le dice el polinomio recíproco de $h(X)$, sus coeficientes son los de $h(X)$ en orden inverso.
- Podemos considerar $\overline{h}(X)$ como el polinomio generador de $\mathcal{C}^{\perp}$, aunque estrictamente hablando en los casos no binarios, hay que multiplicarlo por el escalar h_0^{-1} para hacerlo mónico.
- El polinomio $h(X^{-1}) = X^{n-k}\overline{h}(X)$ es un elemento de $\mathcal{C}^{\perp}$.

Voy a finalizar la sección con un ejemplo de código cíclico:

Ejemplo 1.51 ([Hil86, Pág. 150])**.** Encontrar todos los códigos cíclicos ternarios de longitud 4 y escribir una matriz generatriz para cada uno de ellos:

Empezamos factorizando $x^4 - 1$ en factores irreducibles sobre $\mathbb{F}_3[x]$:

$$x^4 - 1 = (x - 1)(x^3 + x^2 + x + 1) = (x - 1)(x + 1)(x^2 + 1)$$

Así que hay $2^3 = 8$ divisores de $x^4 - 1$ en $\mathbb{F}_3[x]$, cada uno de los cuales generan un código cíclico. Por el Teorema 1.44, estos son los únicos códigos cíclicos ternarios de longitud 4. Los códigos están descritos en la siguiente tabla a partir de sus polinomios generadores y las correspondientes matrices generatrices dadas por el Teorema 1.45.

Polinomio Generador	Matriz Generatriz
1	I_4
$x-1$	$\begin{pmatrix} 2 & 1 & 0 & 0 \\ 0 & 2 & 1 & 0 \\ 0 & 0 & 2 & 1 \end{pmatrix}$
$x+1$	$\begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}$
x^2+1	$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}$
$(x-1)(x+1) = x^2-1$	$\begin{pmatrix} 2 & 0 & 1 & 0 \\ 0 & 2 & 0 & 1 \end{pmatrix}$
$(x-1)(x^2+1) = x^3+2x^2+x+2$	$\begin{pmatrix} 2 & 1 & 2 & 1 \end{pmatrix}$
$(x+1)(x^2+1) = x^3+x^2+x+1$	$\begin{pmatrix} 1 & 1 & 1 & 1 \end{pmatrix}$
$x^4-1=0$	$\begin{pmatrix} 0 & 0 & 0 & 0 \end{pmatrix}$

2 Códigos BCH y de Reed-Solomon

En este capítulo veremos códigos capaces de corregir múltiples errores aprovechando las ventajas que nos aporta trabajar con códigos cíclicos, siendo la familia más importante los códigos BCH. Finalizaremos el capítulo con un estudio en detalle de una subclase muy importante de estos códigos, los códigos Reed-Solomon.

En general, me basaré en el capítulo 11 de [MT97] para definir los conceptos y propiedades de los códigos BCH, además de para darle estructura al contenido del capítulo.

No obstante, para estos códigos necesitamos recurrir a resultados de cuerpos finitos para los que he trabajado a la par con [MT97] y [MS77] como he referenciado en cada caso.

2.1 Códigos BCH

Los códigos Bose-Chauduri-Hocquenghem (BCH) fueron descubiertos por Alexis Hocquenghem (1959) e independientemente por Dwijendra Kumar Ray-Chaudhuri y Raj Chandra Bose [BRC60]. Estos códigos tomaron el nombre a partir de las iniciales de los apellidos de sus tres descubridores.

Los códigos BCH son códigos que pueden corregir errores múltiples a partir de sus propiedades algebraicas, estas propiedades están fundamentadas en los anillos cociente con un polinomio primitivo; son un tipo de códigos cíclicos cuya estructura y propiedades han sido estudiadas profundamente desde que se descubrieron, llevando a usarlos en bastantes situaciones por su fiabilidad, como en transmisiones de comunicaciones con satélites, por ejemplo.

2.1.1 Conceptos previos

Vamos a empezar la sección definiendo una propiedad de los códigos cíclicos que no ha sido definida en el capítulo anterior ya que ahora es cuando vamos a poder ver su utilidad.

Definición 2.1 ([MT97, Cap. 10 §3]). Sea $X^n - 1 = f_1(X)f_2(X) \cdot \ldots \cdot f_m(X)$ la descomposición de $X^n - 1$ en factores irreducibles y sea α_i una raíz de $f_i(X)$. Se tiene que

$$\mathcal{C}_i := \langle f_i(X) \rangle = \{c(X) \in \mathbb{F}_q[X]/\langle X^n - 1 \rangle \mid c(\alpha_i) = 0\}.$$

Es decir, estamos caracterizando el código $\mathcal{C}_i$ a partir de un polinomio del código que cumple que es divisible por $X - \alpha_i$.

En general, para el código cíclico $\mathcal{C}$ generado por $g(X) = f_{i_1} \cdot f_{i_2} \cdot \ldots \cdot f_{i_r}$, se tendrá

$$\mathcal{C} = \langle g(X) \rangle = \{c(X) \mid c(\alpha_{i_1}) = c(\alpha_{i_2}) = \cdots = c(\alpha_{i_r}) = 0\},$$

con lo que tenemos que los códigos cíclicos pueden definirse como conjuntos de polinomios con ciertas raíces $n-$ésimas de la unidad como ceros.

Tras esta definición, si nos ponemos las gafas de teoría de Galois, podemos realizar las siguientes equivalencias de conceptos moviéndonos de códigos cíclicos a cuerpos finitos:

Teorema 2.2 ([MS77, Ch. 7 §5]). *Denotemos por m al menor entero positivo tal que n divide a $q^m - 1$; es decir, m es el orden multiplicativo de q módulo n.*

Hay n elementos distintos $\alpha_0, \alpha_1, \ldots, \alpha_{n-1}$ en $\mathbb{F}_{q^m}$ (las $n-$ésimas raíces de la unidad) tales que

$$X^n - 1 = \prod_{i=0}^{n-1} (X - \alpha_i).$$

Por tanto, a $\mathbb{F}_{q^m}$ se le denomina el cuerpo de descomposición de $X^n - 1$.

Corolario 2.3 ([MS77, Ch.4, Cor. 3]). *Todo elemento $\beta \in \mathbb{F}_q$ de orden $q = p^m$ cumple la identidad:*

$$\beta^{p^m} = \beta,$$

equivalentemente cumple que es una solución de la ecuación

$$X^{p^m} = X.$$

Por tanto

$$X^{p^m} - X = \prod_{\beta \in \mathbb{F}_q} (X - \beta).$$

El corolario anterior es una adaptación del pequeño teorema de Fermat, gracias a este podemos realizar la siguiente observación:

Observación 2.4. Todo elemento $\beta \in \mathbb{F}_q$ ($q = p^m$) cumple la ecuación

$$x^q - x = 0.$$

Este polinomio tiene todos sus coeficientes en $\mathbb{F}_p$ y es mónico, pero β puede cumplir también una ecuación con grado menor.

Esta observación nos lleva a la siguiente definición:

Definición 2.5 ([MS77, Ch.4 §3]). Sea $\beta \in \mathbb{F}_q$, siendo $q = p^m$. El polinomio mínimo sobre $\mathbb{F}_p$ de β es el polinomio mónico de grado mínimo, notémoslo por $M(X)$, con coeficientes en $\mathbb{F}_p$ tal que

$$M(\beta) = 0.$$

Este polinomio cumple las siguientes propiedades:

1. $M(X)$ es irreducible.
2. Si $f(X)$ es un polinomio cualquiera (con coeficientes en $\mathbb{F}_p$) tal que $f(\beta) = 0$, entonces $M(X)|f(X)$.
3. $M(X)|X^{p^m} - X$.

   Demostración. Inmediato a partir de la propiedad anterior y de la observación 2.4. ∎

4. $\deg M(X) \leq m$.

   Demostración. $\mathbb{F}_{p^m}$ es un espacio vectorial de dimensión m sobre $\mathbb{F}_p$. Por tanto, existen $m+1$ elementos $1, \beta, \ldots, \beta^m$ que son linealmente dependientes, es decir, existen $\lambda_i \in \mathbb{F}_p$ (al menos alguno no nulo) tales que

   $$\sum_{i=0}^{m} \lambda_i \beta^i = 0.$$

   Se tiene pues:

   $$p(X) = \sum_{i=0}^{m} \lambda_i X^i$$

   que es un polinomio de grado menor o igual que m que tiene raíz por raíz a β. Luego $\deg M(X) \leq m$. ∎

5. El polinomio mínimo de un elemento primitivo de $\mathbb{F}_{p^m}$ tiene grado m. A este polinomio se le llama polinomio primitivo.

   Demostración. Sea β un elemento primitivo de $\mathbb{F}_{p^m}$ con polinomio mínimo $M(X)$ de grado d. Si usamos $M(X)$ para generar un cuerpo $\mathbb{F}$ de orden p^d, entonces β también pertenecerá a $\mathbb{F}$ y por tanto, cualquier elemento de $\mathbb{F}_{p^m}$; con lo que tenemos que $d \geq m$ y aplicando la propiedad anterior, se obtiene que $d = m$. ∎

6. β y β^p tienen el mismo polinomio mínimo, en este caso se dirá que son conjugados.

   Demostración. Inmediata a partir del pequeño teorema de Fermat y la definición de polinomio mínimo. ∎

Definición 2.6. La operación de multiplicar por p divide los enteros positivos mód $p^m - 1$ en conjuntos llamados clases ciclotómicas mód $p^m - 1$.

Las clases ciclotómicas que contienen al entero $s > 0$ consisten en

$$C_s = \left\{s, ps, p^2 s, p^3 s, \ldots, p^{m_s - 1} s\right\},$$

donde m_s es el menor natural tal que

$$p^{m_s} \cdot s \equiv s \mod p^m - 1.$$

En general, tomaremos como representante de la clase C_s al menor entero s de cada clase.

Tenemos pues que las clases ciclotómicas están definidas mediante una relación de equivalencia y por tanto son clases de equivalencia. Esto nos lleva a tener la siguiente propiedad de forma natural:

Observación 2.7. Notemos que a las potencias de α pueden pasarles dos cosas:

- Que sean conjugadas y por tanto tienen el mismo polinomio mínimo, con lo que los exponentes pertenecen a la misma clase ciclotómica.
- Que no sean conjugadas, generando sus exponentes distintas clases ciclotómicas.

Con lo que concluimos que las clases ciclotómicas son disjuntas.

Observación. En general en este trabajo, salvo que se indique lo contrario, notaremos por α a un elemento primitivo de $\mathbb{F}_q$.

Para finalizar, se tiene la siguiente propiedad de los polinomios mínimos:

7. Si $i \in C_s$, entonces
$$M^{(i)}(X) = \prod_{j \in C_s} (X - \alpha^j).$$
Es más,
$$X^{p^m-1} - 1 = \prod_s M^{(s)}(X),$$
donde s es un índice que se mueve sobre los representantes de las clases ciclotómicas mód $p^m - 1$.

Si nos quitamos las gafas de teoría de Galois y volvemos al punto en el que estábamos con los códigos cíclicos, no es complicado ver lo siguiente:

Corolario 2.8 ([MS77, Ch. 7 §5]). *El polinomio mínimo de α^s es*
$$M^{(s)}(X) = \prod_{i \in C_s} (X - \alpha^i).$$

Este es un polinomio mónico con coeficientes en $\mathbb{F}_q$, es de grado mínimo y tal que tiene a α^s como una raíz.

También se tiene
$$X^n - 1 = \prod_s M^{(s)}(X)$$
donde s varía en el conjunto de los representantes de las clases ciclotómicas mód n. Esta es, por tanto, la factorización de $X^n - 1$ en factores irreducibles sobre $\mathbb{F}_q$.

Tras estos resultados, podemos pasar a reescribir la definición 2.1 de la siguiente forma:

Definición 2.9 ([MS77, Ch. 7 §5]). Sea $\mathcal{C}$ un código cíclico con polinomio generador $g(X)$. Como $g(X)$ es un factor de $X^n - 1$ en $\mathbb{F}_q$, se tiene
$$g(X) = \prod_{j \in J} (X - \alpha^j),$$
donde $j \in J$ implica que $qj \mod n \in J$. Por tanto, J es una unión de clases ciclotómicas.

Los ceros del código son las $n-$ésimas raíces de la unidad $\{\alpha^j \mid j \in J\}$. Claramente, las otras raíces $n-$ésimas de la unidad son los ceros del polinomio $h(X)$ que resulta de $X^n - 1 = g(X)h(X)$.

2.1.2 Definición y parámetros

Teorema 2.10 (Cota BCH [MS77, Ch. 7, Th. 8])**.** Sea $\mathcal{C}$ un código cíclico con polinomio generador $g(X) \in \mathbb{F}_q[X]$, α elemento primitivo de $\mathbb{F}_q$ y tal que para ciertos naturales $b \geq 0$ y $\delta \geq 1$,

$$g(\alpha^b) = g(\alpha^{b+1}) = \ldots = g(\alpha^{b+\delta-2}) = 0.$$

Es decir, el código tiene una cadena de $\delta - 1$ potencias consecutivas de α como ceros. Entonces se tiene que la distancia mínima del código es al menos δ.

Demostración. Sea $c(X)$ un polinomio en $\mathcal{C}$ y $c = (c_0, c_1, \ldots, c_{n-1})$ la palabra asociada al polinomio $c(X)$, entonces

$$c(\alpha^b) = c(\alpha^{b+1}) = \ldots = c(\alpha^{b+\delta-2}) = 0,$$

por tanto, $c \cdot (H')^t = 0$ donde

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

Nótese que H' no necesita ser la matriz de control de $\mathcal{C}$ completa, porque no está definida sobre $\mathbb{F}_q$. La idea de la demostración es probar que cualesquiera $\delta - 1$ columnas de H' son linealmente independientes sobre $\mathbb{F}_{q^m}$. Supongamos que c tiene peso $w \leq \delta - 1$, es decir, $c_i \neq 0$ si y solo si $i \in \{a_1, a_2, \ldots, a_w\}$. Entonces, que $c \cdot (H')^t = 0$ implica

$$\begin{pmatrix} c_{a_1} & \cdots & c_{a_w} \end{pmatrix} \begin{pmatrix} \alpha^{a_1} & \cdots & \alpha^{a_w(b+w-1)} \\ \alpha^{a_1(b+1)} & \cdots & \alpha^{a_w(b+w-1)} \\ \vdots & \cdots & \vdots \\ \alpha^{a_1(b+w-1)} & \cdots & \alpha^{a_w(b+w-1)} \end{pmatrix}^t = 0.$$

Por tanto, el determinante de la matriz de la derecha es nulo, pero este determinante es igual a

$$\alpha^{(a_1+\ldots+a_w)b} \begin{vmatrix} 1 & \cdots & 1 \\ \alpha^{a_1} & \cdots & \alpha^{a_w} \\ \vdots & \cdots & \vdots \\ \alpha^{a_1(w-1)} & \cdots & \alpha^{a_w(w-1)} \end{vmatrix},$$

que es un determinante de Vandermonde y en consecuencia, no nulo; alcanzando una contradicción. ∎

Observación. En el teorema que acabamos de ver, tradicionalmente se considera $\delta \geq 1$, pero si $\delta = 1$ el código deja de sernos útil, es el valor mínimo. Por ello, normalmente se considera $\delta \geq 2$, aunque escribamos la cota con la definición habitual.

Corolario 2.11. *Un código cíclico de longitud n con ceros $\alpha^b, \alpha^{b+r}, \alpha^{b+2r}, \ldots, \alpha^{b+(\delta-2)r}$, donde r y n son primos relativos, tiene distancia mínima al menos δ.*

Demostración. Análoga a la del teorema anterior aunque (al menos a priori) el enunciado del corolario no sea consecuencia del enunciado del teorema. ∎

Definición 2.12. Un código cíclico de longitud n sobre $\mathbb{F}_q$ generado por $g(X)$ es un código BCH con distancia prevista δ, si para un entero $b \geq 0$,

$$g(X) = \text{mcm}\left\{M^{(b)}(X), M^{(b+1)}(X), \ldots, M^{(b+\delta-2)}(X)\right\}.$$

Es decir, $g(X)$ es el polinomio mónico de menor grado sobre $\mathbb{F}_q$ que tiene por ceros a $\alpha^b, \alpha^{b+1}, \ldots, \alpha^{b+\delta-2}$. Consecuentemente,

$$c \text{ está en el código} \iff c(\alpha^b) = c(\alpha^{b+1}) = \ldots = c(\alpha^{b+\delta-2}) = 0.$$

Por tanto, el código tiene una cadena de $\delta - 1$ potencias consecutivas de α como ceros. A partir del teorema 2.10 concluimos que la distancia mínima del código es mayor o igual que la distancia prevista δ. La última caracterización nos deja entrever, además, que la matriz de control del código es

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

donde cada entrada es reemplazada por la fila de m elementos correspondientes a cada α^s como vectores sobre $\mathbb{F}_q$.

La dimensión del código es, como en todos los códigos cíclicos, $n - \deg(g(X))$.

Teorema 2.13 ([MS77, Ch. 9, Th. 1])**.**

1. *Un código BCH sobre $\mathbb{F}_q$ de longitud n y distancia prevista δ tiene distancia mínima $d \geq \delta$, y dimensión $k \geq n - m(\delta - 1)$.*
2. *Mejora de la cota (caso particular):*
   Dado un código BCH sobre $\mathbb{F}_{2^m}$ de longitud n y distancia prevista δ impar con $\delta = 2t + 1$, entonces $d \geq \delta$ y $k \geq n - mt > n - m(\delta - 1)$.

3. *Mejora de la cota (caso general):*
   Dado un código BCH sobre $\mathbb{F}_{p^m}$ de longitud n y distancia prevista $\delta = pt + 1$, entonces $d \geq \delta$ y $k \geq n - m(p-1)t$.

Demostración. 1. Se deduce a partir de la prueba del Teorema 2.10. Concretamente, la matriz H, al representarla en $\mathbb{F}_p$, tiene $m(\delta - 1)$ columnas. Alguna de ellas serán linealmente dependientes, pero entonces sabemos que $n - k$, que es el rango de H, debe ser menor que $m(\delta - 1)$.

El apartado 3. es un refinamiento del primero cuando $\delta = pt + 1$ para cierto t. Por el primero se tiene que $n - k \leq mpt$; de hecho, hay que ver que realmente podemos sustituir p por $(p-1)$. En [MS77, Ch. 7 §6], tenemos que en característica p, $M^{(i)}(X) = M^{(pi)}(X)$, así que si tenemos pt raíces, que vengan de $ptM^{(j)}(X)$, t de ellos nos van a dar las mismas raíces, que a su vez nos van a proporcionar las mt columnas iguales a otras en H; con lo que, realmente $n - k$ no puede pasar de $mpt - mt = m(p-1)t$, y así tenemos que $k \geq n - m(p-1)t$. ∎

Hay ciertos valores para los parámetros que son muy importantes ya que forman clases concretas de los códigos BCH:

- Si $b = 1$ se dice que es un código BCH en sentido restringido (*narrow sense*).
- Si $n = q^m - 1$ diremos que es un código BCH primitivo, ya que α es un elemento primitivo de $\mathbb{F}_{q^m}$.
- Si $n = q - 1$ ($m = 1$) son los conocidos como códigos Reed-Solomon.

Visto que, en este caso, los códigos se construyen a partir de los parámetros, nos podemos plantear si nos vale que n sea todo lo grande que queramos o si por el contrario, existen unas cotas para los parámetros. Podemos encontrar la respuesta a esta pregunta en el artículo [LW67] y en [MS77, Ch. 9 §5]; donde se estudia qué pasa con los códigos BCH para valores de n grande (se estudian las cotas asintóticamente). Hay una zona óptima de valores en la que se mueven n, k y d, como vimos en TCYC, con las cotas de Plotkin y Gilbert-Varshamov. A pesar de esto, sigue siendo una familia de códigos bastante buena para valores razonablemente grandes.

2.1.3 Descodificación de los Códigos BCH

En esta sección «por simplicidad y generalidad» vamos a trabajar con [MT97, Cap. 11 §2].

El interés de los códigos BCH radica, de una parte, en la posibilidad de elegir a priori la capacidad correctora deseada (determinada por δ) y de otra en la existencia de un

algoritmo efectivo de descodificación. Dado que, en general, no es factible conocer la auténtica distancia de un código BCH, en la práctica se utiliza δ como un sustituto de la misma.

Sea $\mathcal{C}$ el código BCH sobre $\mathbb{F}_q$ de longitud n y distancia prevista $\delta = 2t + 1$. Sea α un raíz primitiva $n-$ésima de la unidad. Sin pérdida de generalidad, por simplificar la notación, vamos a descodificar el código determinado por las raíces $\alpha, \ldots, \alpha^{\delta-1}$, y con matriz de control sobre $\mathbb{F}_q$

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

Supongamos enviada una palabra $c \in \mathcal{C}$ y recibido un vector $y = c + e$ con $w(e) = r \leq t$. Sean $0 \leq i_1 < \ldots < i_r \leq n - 1$, las posiciones en las que han ocurrido errores y $e_{i_1}, \ldots, e_{i_r}$ las coordenadas del vector error e en esas posiciones (el resto de componentes son ceros).

El primer paso en la descodificación consiste en calcular el síndrome del vector recibido

$$S = S(y) \overset{(1.31)}{=} S(e) = yH^t = (s_0, \ldots, s_{\delta-2})$$

que, recurriendo a la notación polinómica, podemos escribir

$$S(X) = s_0 + \ldots + s_{\delta-2}X^{\delta-2}.$$

Obsérvese que para cada $h = 0, \ldots, \delta - 2$,

$$s_h = y(\alpha^{h+1}) = e(\alpha^{h+1}) = \sum_{j=1}^{r} e_{i_j}(\alpha^{h+1})^{i_j} = \sum_{j=1}^{r} e_{i_j}(\alpha^{i_j})^{h+1}.$$

Por si algún lector se ha despistado, recordemos que los vectores y y e son polinomios, es decir, $y(X)$ y $e(X)$. Por tanto, lo anterior tiene sentido ya que estamos evaluando dichos polinomios en los α^{h+1}.

En ambas referencias [MS77, MT97] acuerdan simplificar un poco la notación de s_h para no arrastrar tantos subíndices. Consideremos pues

$$\eta_j = \alpha^{i_j} \quad \varepsilon_j = e_{i_j}$$

para $j = 1, \ldots, r$; con lo que

$$s_h = \varepsilon_1 \eta_1^{h+1} + \ldots + \varepsilon_r \eta_r^{h+1}.$$

Los η_j son llamados localizadores del error y los ε_j valores del error (siempre con respecto al vector recibido y). Por supuesto, conocer estos $2r$ números equivale a conocer el error.

Si $S(X) = 0$ entonces $y \in \mathcal{C}$, con lo que el mensaje recibido es dado por válido y no hace falta descodificación. Por tanto, supondremos que $S(X) \neq 0$.

Definición 2.14. Llamaremos polinomio localizador de errores al polinomio

$$L(X) = (1 - \eta_1 X) \cdot \ldots \cdot (1 - \eta_r X)\,.$$

Llamaremos polinomio evaluador de errores al polinomio

$$E(X) = \sum_{j=1}^{r} \varepsilon_j \prod_{i \neq j} (1 - \eta_i X)\,.$$

Los polinomios $L(X)$ y $E(X)$ son de grados r y $r-1$ respectivamente, cuyo conocimiento implica conocer los η_j y ε_j.

Proposición 2.15. *En las condiciones anteriores,*

1. *Si $\rho_1, \ldots, \rho_r$ son las raíces de $L(X)$, entonces sus inversos $\rho_1^{-1}, \ldots, \rho_r^{-1}$ son los localizadores del error.*
2. *Conocidos $\eta_1, \ldots, \eta_r$, los valores del error son*

$$\varepsilon_j = \frac{-\eta_j E\left(\eta_j^{-1}\right)}{L'\left(\eta_j^{-1}\right)},$$

siendo $L'(X)$ la derivada (formal) de $L(X)$.

Demostración.

1. Inmediato.
2. Partamos derivando $L(X)$,

$$L'(X) = \sum_{h=1}^{r} (-\eta_h) \prod_{i \neq h} (1 - \eta_h X)\,,$$

η_j^{-1} es raíz de todos los sumandos de $L'(X)$ excepto del $j-$ésimo. Por tanto

$$\varepsilon_j L'\left(\eta_j^{-1}\right) = (-\eta_j)\, \varepsilon_j \prod_{i \neq j} \left(1 - \eta_i \eta_j^{-1}\right) = -\eta_j E\left(\eta_j^{-1}\right)$$

de donde podemos terminar deduciendo el resultado despejando ε_j.

El algoritmo de descodificación que estudiamos no nos proporciona directamente el vector e, sino los polinomios $L(X)$ y $E(X)$. A partir de estos, podemos deducir los η_j y ε_j y, finalmente, las posiciones y valores del error. Para determinar estos $L(X)$ y $E(X)$ recurrimos al polinomio síndrome.

Podemos escribir el polinomio $E(X)$ en términos de serie de potencias

$$E(X) = \sum_{j=1}^{r} \varepsilon_j \frac{L(X)}{1-\eta_j X} = L(X) \sum_{j=1}^{r} \varepsilon_j \sum_{i=0}^{\infty} \eta_j^i X^i = \\ = L(X) \sum_{i=0}^{\infty} \left(\sum_{j=1}^{r} \varepsilon_j \eta_j^i \right) X^i = L(X) \sum_{i=0}^{\infty} e\left(\alpha^i\right) X^i.$$

Teorema 2.16 (Ecuación clave). *Los polinomios $L(X)$, $E(X)$ y $S(X)$ están relacionados mediante la ecuación*

$$E(X) \equiv L(X)S(X) \text{ mód } X^{\delta-1}.$$

Demostración. Por lo visto justo antes del teorema y la definición de s_h,

$$E(X) \equiv L(X) \sum_{i=0}^{\delta-2} e\left(\alpha^{i+1}\right) X^i = L(X) \sum_{i=0}^{\delta-2} s_i X^i = L(X)S(X) \text{ mód } X^{\delta-1}$$

como se quería probar.

Es decir, tras este teorema concluimos que el polinomio $E(X)$ se puede obtener inmediatamente tras calcular $L(X)$ y $S(X)$. El proceso de descodificación quedará completo en cuanto seamos capaces de determinar $L(X)$ y $E(X)$, o simplemente $L(X)$ a tenor de lo visto en el teorema previo. Ahora vamos a pasar a estudiar dos métodos alternativos para realizar este cálculo.

Método euclídeo

Este método se basa en el algoritmo de Euclides en $\mathbb{F}_q[X]$. Para esto, nuestra referencia nos recomienda enunciar unos resultados adicionales sobre $L(X)$ y $E(X)$ que nos serán de ayuda.

Lema 2.17. *mcd* $(L(X), E(X)) = 1$.

Demostración. Inmediato por no tener raíces en común. ∎

Proposición 2.18. *Si* $\widetilde{L}(X)$ *y* $\widetilde{E}(X)$ *son dos polinomios que verifican*

1. $\deg\left(\widetilde{L}(X)\right) \leq t$, $\deg\left(\widetilde{E}(X)\right) < t$; *y*
2. $\widetilde{E}(X) \equiv S(X)\widetilde{L}(X)\ \left(\textit{mód}\, X^{\delta-1}\right)$,

entonces existe un polinomio $\lambda(X)$ *tal que* $\widetilde{L}(X) = \lambda(X)L(X)$ *y* $\widetilde{E}(X) = \lambda(X)E(X)$.

Demostración. De la congruencia (2) y la ecuación clave, podemos deducir que

$$E(X)\widetilde{L}(X) \equiv L(X)\widetilde{E}(X)\ \left(\text{mód}\, X^{\delta-1}\right).$$

Como $\deg\left(E(X)\widetilde{L}(X)\right) < \delta - 1$ y $\deg\left(L(X)\widetilde{E}(X)\right) < \delta - 1$, esta congruencia implica que $E(X)\widetilde{L}(X) = L(X)\widetilde{E}(X)$. Siendo los polinomios $L(X)$ y $E(X)$ primos relativos, no queda otra alternativa que $L(X)|\widetilde{L}(X)$, con lo que existe $\lambda(X)$ tal que $\widetilde{L}(X) = \lambda(X)L(X)$, con lo que tenemos que $\widetilde{E}(X) = \lambda(X)E(X)$ por la igualdad previa. ∎

Podemos concluir tras este resultado que para determinar $L(X)$ y $E(X)$ nos basta con encontrar dos polinomios $\widetilde{L}(X)$ y $\widetilde{E}(X)$ que verifiquen las condiciones de la proposición anterior de manera que, salvo multiplicación por elementos de $\mathbb{F}_q$,

$$L(X) = \frac{\widetilde{L}(X)}{\text{mcd}\left(\widetilde{L}(X), \widetilde{E}(X)\right)} \text{ y } E(X) = \frac{\widetilde{E}(X)}{\text{mcd}\left(\widetilde{L}(X), \widetilde{E}(X)\right)}.$$

A partir de aquí, utilizaremos fuertemente los resultados obtenidos y las notaciones definidas en el Apéndice A.

El proceso de cálculo es el siguiente, tomamos como polinomios iniciales para el algoritmo de Euclides, $f_0(X) = X^{\delta-1}$ y $f_1(X) = S(X)$. Sea j el menor índice para el que $\deg\left(f_j(X)\right) < t$ (es decir, $\deg\left(f_{j-1}(X)\right) \geq t$). Establezcamos lo siguiente:

$$\widetilde{L}(X) = u_j(X) \text{ y } \widetilde{E}(X) = (-1)^{j+1} f_j(X).$$

Donde u_j y f_j quedan definidos como en el apéndice A.

Proposición 2.19. *Si $r \leq t$, los polinomios $\widetilde{L}(X)$ y $\widetilde{E}(X)$ verifican las condiciones* (1) *y* (2) *de la proposición 2.18.*

Demostración. Por definición $\deg \widetilde{E} < t$. Para probar las otras propiedades haremos uso de los resultados establecidos en A. En primer lugar, según la proposición A.3

$$\deg u_j(X) = \deg f_0(X) - \deg f_{j-1}(X)$$

luego, por ser $\deg f_0(X) = \delta - 1$ y $\deg f_{j-1}(X) \geq t$, se tiene que $\deg u_j(X) \leq t$.

Para probar (2), según la proposición A.2,

$$(-1)^{j+1} f_j(X) = (-1)^{j+1}(-1)^j \left(v_j(X)X^{\delta-1} - u_j(X)S(X)\right)$$

de donde

$$(-1)^{j+1} f_j(X) - u_j(X)S(X) = -v_j(X)X^{\delta-1} \equiv 0 \text{ mód } X^{\delta-1}$$

y se tiene (2). ∎

Aún más, los polinomios $\widetilde{L}(X)$, $\widetilde{E}(X)$ verifican la propiedad siguiente:

Proposición 2.20. *mcd* $\left(\widetilde{L}(X), \widetilde{E}(X)\right) = 1$.

Demostración. Como sabemos, $\widetilde{L}(X) = \lambda(X)L(X)$ y $\widetilde{E}(X) = \lambda(X)E(X)$, siendo $\lambda(X) = \text{mcd}\left(\widetilde{L}(X), \widetilde{E}(X)\right)$. Debemos probar que $\lambda(X) \in \mathbb{F}_q$. Para ello, como según A.3 (2.), $\text{mcd}\,(u_j(X), v_j(X)) = 1$, es suficiente probar que $\lambda(X)$ divide a ambos polinomios u_j y v_j. Esto es claro para $u_j(X)$ ya que

$$\lambda(X)L(X) = \widetilde{L}(X) = u_j(X).$$

En cuanto a $v_j(X)$, en la demostración de 2.19 se probó que

$$v_j(X)X^{\delta-1} = \widetilde{L}(X)S(X) - \widetilde{E}(X) = \lambda(X)\left(L(X)S(X) - E(X)\right).$$

Por otra parte, como $E(X) \equiv L(X)S(X)$ mód $X^{\delta-1}$, existe un polinomio $\mu(X)$ tal que $E(X) = L(X)S(X) + \mu(X)X^{\delta-1}$. Sustituyendo este valor en la igualdad anterior

$$v_j(X)X^{\delta-1} = -\lambda(X)\mu(X)X^{\delta-1}$$

luego $v_j(X) = -\lambda(X)\mu(X)$ y $\lambda(X)|v_j(X)$. ∎

Llegados a este punto,

$$L(X) = \lambda u_j(X) \text{ y } E(X) = (-1)^{j+1}\lambda f_j(X)$$

y falta únicamente determinar la constante $\lambda \in \mathbb{F}_q$. Como el auténtico polinomio localizador verifica que $L(0) = 1$,

$$1 = L(0) = \lambda u_j(0).$$

En definitiva, hemos probado el siguiente resultado:

Teorema 2.21. *Si $r \leq t$, los polinomios*

$$L(X) = \frac{u_j(X)}{u_j(0)} \quad E(X) = \frac{(-1)^{j+1} f_j(X)}{u_j(0)}$$

son los auténticos polinomios localizador y evaluador de errores.

Para terminar, vamos a poner los distintos pasos de la descodificación en forma de algoritmo.

Algoritmo 2.22. *Recibido un vector y:*

1. *Determinar su síndrome $S(X) = s_1 + \ldots + s_{\delta-1}X^{\delta-2}$, con $s_i = u(\alpha_i)$.*
2. *Aplicar el algoritmo de Euclides modificado a los polinomios $f_0(X) = X^{\delta-1}$ y $f_1(X) = S(X)$, hasta obtener un índice j tal que $\deg f_j(X) < t$.*
3. *Computar los polinomios localizador y evaluador de errores*

$$L(X) = \frac{u_j(X)}{u_j(0)} \quad E(X) = \frac{(-1)^{j+1} f_j(X)}{u_j(0)}.$$

4. *(Método de Chien) [Chi64] Encontrar las raíces de $L(X)$ evaluando este polinomio en todos los $1, \alpha, \ldots, \alpha^{n-1}$. Si $\alpha^{h_1}, \ldots, \alpha^{h_r}$ son tales raíces, entonces sus inversos son los localizadores del error, $\eta_1 = \alpha^{n-h_1}, \ldots, \eta_r = \alpha^{n-h_r}$. El valor del error asociado al localizador η_j es*

$$\varepsilon_j = \frac{-\alpha^{n-h_j} E(\alpha^{h_j})}{L'(\alpha^{h_j})}.$$

5. *Descodificar, y si es necesario corregir, el mensaje recibido:*
   para $i = 0, \ldots, n-1$, la $i-$ésima cordenada del mensaje descodificado es

$$\begin{cases} y_i & \text{si } i \neq n - h_1, \ldots, n - h_r; \\ y_i - \varepsilon_j & \text{si } i = n - h_j. \end{cases}$$

Método de Berlekamp-Massey

Este método es una variante del anterior en tanto que sustituye el algoritmo de Euclides por la teoría de sucesiones recurrentes. Este método determina únicamente el polinomio $L(X)$, deduciéndose $E(X)$ a partir de él y el polinomio síndrome usando la ecuación clave.

Supongamos, al igual que antes, que hemos emitido c y recibimos $y = c+e$. Sea $S(y) = (s_0, \ldots, s_{\delta-2})$ el síndrome del vector recibido. Recordemos que $\eta_j = \alpha^{i_j},\ \varepsilon_j = e_{i_j}$ para $j = 1, \ldots, r$ y que $s_h = \varepsilon_1\eta_1^{h+1} + \ldots + \varepsilon_r\eta_r^{h+1}$.

Consideremos el polinomio

$$B(X) = \prod_{i=1}^{r} (X - \eta_i)\,.$$

Podemos escribir este polinomio en función de los polinomios simétricos elementales $\sigma_1, \ldots, \sigma_r$ en los η_i:

$$B(X) = X^r - \sigma_1 X^{r-1} + \ldots + (-1)^r\sigma_r.$$

Por la construcción de $B(X)$ tenemos que sus raíces son $\eta_1, \ldots, \eta_r$, con lo que obtenemos las siguientes r ecuaciones

$$\eta_i^r - \sigma_1\eta_i^{r-1} + \ldots + (-1)^r\sigma_r = 0,\ \ i = 1, \ldots, r.$$

Multiplicando la ecuación $i-$ésima por $\varepsilon_i\eta_i^j$, j fijo $1 \leq j \leq r$, y sumando todas las relaciones así obtenidas, se tiene

$$s_{j+r-1} - \sigma_1 s_{j+r-2} + \ldots + (-1)^r\sigma_r s_{j-1} = 0.$$

Repitiendo el proceso para cada $j = 1, \ldots, r$, finalmente se obtiene el sistema lineal de r ecuaciones en las r incógnitas $l_i = (-1)^i\sigma_i$:

$$A := \left\{ \begin{array}{ccccccccc} s_r & + & s_{r-1}l_1 & + & \cdots & + & s_0 l_r & = & 0 \\ s_{r+1} & + & s_r l_1 & + & \cdots & + & s_1 l_r & = & 0 \\ \vdots & & \vdots & & & & \vdots & & \vdots \\ s_{2r-1} & + & s_{2r}l_1 & + & \cdots & + & s_{r-1}l_r & = & 0 \end{array} \right.$$

Lema 2.23. *El sistema anterior tiene solución única en los l_i si y sólo si (como estamos suponiendo) el error e tiene justamente peso r.*

Demostración. La matriz de coeficientes del sistema,

$$C = \begin{pmatrix} s_0 & s_1 & \cdots & s_{r-1} \\ s_1 & s_2 & \cdots & s_r \\ \vdots & \vdots & & \vdots \\ s_{r-1} & s_r & \cdots & s_{2r} \end{pmatrix}$$

se factoriza en la forma $C = VDV^t$, siendo

$$V = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ \eta_1 & \eta_2 & \cdots & \eta_r \\ \vdots & \vdots & & \vdots \\ \eta_1^{r-1} & \eta_2^{r-1} & \cdots & \eta_r^{r-1} \end{pmatrix}, \quad D = \begin{pmatrix} \varepsilon_1\eta_1 & 0 & \cdots & 0 \\ 0 & \varepsilon_2\eta_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \varepsilon_r\eta_r \end{pmatrix}.$$

Dado que V es una matriz de Vandermonde y D una matriz diagonal, el que ambas sean no singulares equivale a la condición enunciada. ∎

Una vez conocidos los $l_1, \ldots, l_r$ (y el número de errores r), el polinomio localizador de errores se deduce inmediatamente.

Proposición 2.24. *El polinomio localizador de errores es*

$$L(X) = 1 + l_1X + \cdots + l_rX^r.$$

Demostración. Es suficiente probar que el polinomio, así obtenido, verifica que $L(\eta_i^{-1}) = 0$ para $i = 1, \ldots, r$. Ahora bien

$$\eta_i^r L(\eta_i^{-1}) = \eta_i^r + l_1\eta_i^{r-1} + \ldots + l_r = B(\eta_i) = 0$$

por definición de $B(X)$. ∎

Como el número de errores de la palabra recibida es desconocido, sería necesario calcularlo. Haciendo uso del lema 2.23 basta determinar el máximo r tal que el sistema A tenga solución única. Una vez hecho esto, basta resolver tal sistema para obtener los l_i.

Esta teoría sobre el papel funciona muy bien, pero computacionalmente es bastante compleja de implementar, por ello han sido propuestos diversos algoritmos alternativos. Ahora vamos a pasar a estudiar el elaborado por Berlekamp-Massey.

Este algoritmo, realmente, calcula el polinomio mínimo de una sucesión en recurrencia de orden r, supuesto que conocemos $2r$ términos de dicha sucesión. Notemos que el

sistema A muestra que los s_i forman realmente una sucesión recurrente de orden r, con ecuación de recurrencia

$$s_r + s_{r-1}l_1 + \ldots + s_0 l_r = 0$$

y que realmente se conocen los $2r$ términos de la sucesión ya que $r \leq t = \left\lfloor \delta - \frac{1}{2} \right\rfloor$, y $s_0, \ldots, s_{\delta-2}$ se deducen del síndrome. Además, el lema 2.23 garantiza que la anterior ecuación de recurrencia tiene orden mínimo. Esta característica de los códigos BCH es la que permite aplicar el método de Berlekamp-Massey. Pasemos ya a describir el algoritmo.

A partir del síndrome del vector recibido

$$S(X) = s_0 + s_1 X + \ldots + s_{\delta-2} X^{\delta-2}$$

se construyen, de manera recursiva, las cuatro sucesiones:

$$\{L_j(X)\}_{j=0}^{2t}, \ \{E_j(X)\}_{j=0}^{2t}, \ \{m_j\}_{j=0}^{2t}, \ \{b_j\}_{j=0}^{2t}.$$

Para ello, tomamos los valores iniciales:

$$L_0(X) = 1, \ E_0(X) = X, \ m_0 = 0, \ b_0 = s_0,$$

y, supuesto construidos los términos $j-$ésimos, se definen

$$L_{j+1}(X) = L_j(X) - b_j E_j(X).$$

$$E_{j+1}(X) = \begin{cases} b_j^{-1} X L_j(X) & \text{si } b_j \neq 0 \text{ y } m_j \geq 0 \\ X E_j(X) & \text{en otro caso} \end{cases}.$$

$$m_{j+1} = \begin{cases} -m_j & \text{si } b_j \neq 0 \text{ y } m_j \geq 0 \\ m_j + 1 & \text{en otro caso} \end{cases}.$$

$$b_{j+1} = \text{coeficiente de } X^{j+1} \text{ en } L_{j+1}(X)S(X).$$

Una vez construidas estas sucesiones, sea $s = \left\lfloor t + \frac{1}{2} - \frac{m_{2t}}{2} \right\rfloor$. El polinomio

$$m(X) = X^s L_{2t}\left(\frac{1}{X}\right)$$

resulta ser el polinomio mínimo de la sucesión recurrente, y tiene por coeficientes precisamente los l_i buscados. Una vez conocidos dichos l_i, se construye el polinomio localizador de errores $L(X)$ como en la proposición 2.24 y, a partir de él y de $S(X)$, el polinomio evaluador de errores $E(X)$, como ya se comentó tras el teorema 2.16. El cálculo de las posiciones y valores del error, y la descodificación del mensaje recibido se lleva a cabo como en la versión euclídea.

2.2 Códigos de Reed-Solomon

Para esta sección vamos a utilizar el capítulo 10 de [MS77].

En la sección anterior mencionamos que los códigos BCH tales que $n = q - 1$ eran conocidos como códigos Reed-Solomon. Este valor de n genera una familia concreta y distinguible por cumplir ciertas propiedades que veremos a lo largo de la sección.

Estos códigos son apropiados para construir otros códigos, ya sean solos (por ejemplo realizando una extensión a códigos binarios) o en combinación con otros códigos, como los códigos concatenados.

Definición 2.25. Un código Reed-Solomon (o RS) sobre $\mathbb{F}_q$ es un código BCH de longitud $N = q - 1$, con lo que evidencia que q jamás puede ser 2. Es decir, la longitud es el número de elementos no nulos en el espacio de definición. Utilizaremos N, K y D para denotar la longitud, dimensión y distancia mínima; distinguiendo con mayúsculas de los parámetros habituales que utilizaremos más adelante para los códigos binarios.

En este caso, al ser $X^{q-1} - 1 = \prod_{\beta \in \mathbb{F}_q^*} (X - \beta)$, tenemos que el polinomio mínimo de α^i es simplemente $M^{(i)}(X) = X - \alpha^i$. Por tanto, el código RS de longitud $q - 1$ y distancia prevista δ tiene polinomio generador

$$g(X) = \left(X - \alpha^b\right)\left(X - \alpha^{b+1}\right) \cdot \ldots \cdot \left(X - \alpha^{b+\delta-2}\right).$$

Normalmente, pero no siempre, $b = 1$.

Ejemplo 2.26. Consideremos $\mathbb{F}_{2^2} = \{0, 1, \alpha, \alpha^2\}$ con la relación $\alpha^2 + \alpha + 1 = 0$. Un código RS sobre $\mathbb{F}_4$ con $N = 3$, $\delta = 2$ y $b = 2$ tiene como polinomio generador $g(X) = X - \alpha^2$. Las 4^2 palabras del código son:

$$\begin{array}{cccc} 000 & 1\alpha 0 & \alpha^2 0\alpha & \alpha^2\alpha 1 \\ 01\alpha & \alpha\alpha^2 0 & 10\alpha^2 & 111 \\ 0\alpha\alpha^2 & \alpha^2 10 & 1\alpha^2\alpha & \alpha\alpha\alpha \\ 0\alpha^2 1 & \alpha 01 & \alpha 1\alpha^2 & \alpha^2\alpha^2\alpha^2. \end{array}$$

Hemos incluido este ejemplo porque conforme vayamos avanzando, iremos ampliándolo con las propiedades que iremos definiendo.

La dimensión de un código RS es $K = N - \deg g(X) = N - \delta + 1$. La distancia mínima es D, que por la cota BCH es al menos $\delta = N - K + 1$; y por la cota de Singleton[1]

[1] **Cota de Singleton**: Si $\mathcal{C}$ es un $[n, k, d]$ código, entonces $n - k \geq d - 1$. *Recordatorio de un resultado visto en TCYC.*

tenemos que no puede ser mayor a ese valor, con lo que

$$D = N - K + 1,$$

y esto significa que los códigos RS son códigos MDS (Máxima Distancia de Separación), es decir, alcanzan la cota de Singleton con igualdad.

Los códigos Reed-Solomon son importantes por diversas razones:

- Son códigos que se usan de forma natural cuando requerimos un código de longitud inferior que el cardinal del cuerpo. Por ser MDS, tenemos que tienen el mayor valor posible para la distancia mínima.
- Son adecuados para construir otros códigos, como veremos. Por ejemplo, se pueden extender a códigos binarios con una distancia mínima bastante alta.
- Son de gran utilidad para corregir errores en ráfagas.

2.2.1 Códigos Reed-Solomon extendidos

Si añadimos un dígito de control a un código, no siempre se incrementa la distancia mínima, no obstante tenemos el siguiente teorema:

Teorema 2.27. *Sea $\mathcal{C}$ el $[N = q - 1, K, D]$ código RS con polinomio generador*

$$g(X) = (X - \alpha)\left(X - \alpha^2\right) \cdot \ldots \cdot \left(X - \alpha^{D-1}\right).$$

Entonces extendiendo cada palabra código $c = c_0 c_1 \ldots c_{N-1}$ de $\mathcal{C}$ añadiendo un dígito de control

$$c_N = -\sum_{i=0}^{N-1} c_i$$

se produce un $[N + 1, K, D + 1]$ código.

Demostración. Supongamos que $c \in \mathcal{C}$ tiene peso D. El peso mínimo aumenta a $D+1$ ya que

$$c(1) = -c_N = \sum_{i=0}^{N-1} c_i \neq 0.$$

Pero $c(X) = a(X)g(X)$ para algún polinomio $a(X)$, luego $c(1) = a(1)g(1)$. $g(1) \neq 0$ porque 1 no es raíz de $g(X)$. Es más, $a(1) \neq 0$ o bien $c(X)$ es múltiplo de $(X - 1)\, g(X)$ y tiene peso mayor o igual que $D + 1$ por la cota BCH.

Ejemplo 2.28. Continuando el ejemplo 2.26, el teorema anterior nos da el código $[4, 2, 3]$, quedando de la siguiente forma:

$$\begin{array}{cccc} 0000 & 1\alpha 0\alpha^2 & \alpha^2 0\alpha 1 & \alpha^2\alpha 10 \\ 01\alpha\alpha^2 & \alpha\alpha^2 01 & 10\alpha^2\alpha & 1111 \\ 0\alpha\alpha^2 1 & \alpha^2 10\alpha & 1\alpha^2\alpha 0 & \alpha\alpha\alpha\alpha \\ 0\alpha^2 1\alpha & \alpha 01\alpha^2 & \alpha 1\alpha^2 0 & \alpha^2\alpha^2\alpha^2\alpha^2. \end{array}$$

2.2.2 Extendiendo códigos sobre $\mathbb{F}_{2^m}$ a códigos binarios

Ahora vamos a extender los códigos sobre $\mathbb{F}_{2^m}$ a códigos binarios. $\mathbb{F}_q$, donde $q = p^m$, es un $\mathbb{F}_p-$espacio vectorial de dimensión m . Por tanto, un código RS $[N, K, D]$ sobre $\mathbb{F}_q$ pasa a ser un código $[n = mN, k = mK, d \geq D]$ sobre $\mathbb{F}_p$. En el caso en el que $q = 2^m$ los códigos binarios obtenidos por este procedimiento (y otros derivados) usualmente tienen valores altos de distancia mínima, como veremos a continuación.

Sea $\xi_1, \ldots, \xi_m$ una base de $\mathbb{F}_{2^m}$ sobre $\mathbb{F}_2$. Entonces, si $\beta = \sum_{i=1}^{m} b_i\xi_i$ es un elemento cualquiera de $\mathbb{F}_{2^m}$, $b_i \in \mathbb{F}_2$, asignamos β a $b_1, b_2, \ldots, b_m$. Esta aplicación envía códigos lineales en códigos lineales, pero los códigos cíclicos no necesariamente son asignados a otro código cíclico.

Ejemplo 2.29. Usando la base $\{1, \alpha\}$ de $\mathbb{F}_4$ sobre $\mathbb{F}_2$, podemos realizar la siguiente asignación:

$$\begin{array}{ccc} 0 & \mapsto & 00, \\ 1 & \mapsto & 10, \\ \alpha & \mapsto & 01, \\ \alpha^2 & \mapsto & 11. \end{array}$$

Entonces, el código RS $[3, 2, 2]$ definido sobre $\mathbb{F}_4$ del ejemplo 2.26, se transforma en el código binario $[6, 4, 2]$ recogido en la siguiente tabla:

$$\begin{array}{cccc} 000000 & 100100 & 110001 & 110110 \\ 001001 & 011100 & 100011 & 101010 \\ 000111 & 111000 & 101101 & 010101 \\ 001110 & 010010 & 011011 & 111111 \end{array}.$$

Ejemplo 2.30. Sea $c = (c_0, c_1, \ldots, c_{N-1})$ una palabra del código RS $[N, K, D]$ sobre $\mathbb{F}_{2^m}$. Si reemplazamos cada c_i por su correspondiente $m-$tupla binaria y añadimos

un dígito de control a cada $m-$tupla, obtenemos un código binario con los siguientes parámetros:

$$n = (m+1)(2^m - 1), \quad k = mK, \quad d \geq 2D = 2\left(2^m - K\right),$$

para cualquier $K = 1, \ldots, 2^m - 2$. El factor 2 antes de D se debe a la adicción del dígito de control. Por cada coordenada distinta de 0 en el código original, vamos a obtener dos distintas de 0 en el nuevo. La misma construcción aplicada al código RS extendido nos da códigos binarios de tipo

$$\left[(m+1)2^m, mK, d \geq 2\left(2^m - K + 1\right)\right],$$

para $K = 1, \ldots, 2^m - 1$.

Es decir, por ejemplo, de los códigos $[15, 10, 6]$ y $[16, 10, 7]$ sobre $\mathbb{F}_{2^4}$ obtenemos los códigos binarios $[75, 40, 12]$ y $[80, 40, 14]$.

Ejemplo 2.31. [MS77, Ex. 5.3] Usando la base $\{1, \alpha, \alpha^6\}$ de $\mathbb{F}_{2^3}$ sobre $\mathbb{F}_2$, donde α es raíz del polinomio $X^3 + X + 1$, consideramos la aplicación

$$\begin{array}{lll} 0 \mapsto 000, & 1 \mapsto 100, & \alpha \mapsto 010, \\ \alpha^2 \mapsto 101, & \alpha^3 \mapsto 110, & \alpha^4 \mapsto 111, \\ \alpha^5 \mapsto 011, & \alpha^6 \mapsto 001. & \end{array}$$

Consideremos ahora el código RS $[7, 5, 3]$ sobre $\mathbb{F}_{2^3}$ con polinomio generador

$$g_1(X) = \left(X + \alpha^5\right)\left(X + \alpha^6\right) = \alpha^4 + \alpha X + X^2.$$

Sorprendentemente, este código se puede asociar con el código binario BCH $[21, 15, 3]$ con polinomio generador

$$g_2(Y) = M^{(1)}(Y) = 1 + Y + Y^2 + Y^4 + Y^6.$$

Tenemos $g_1(X)$ que podemos asociar al siguiente vector

$$111,\ 010,\ 100,\ 000,\ 000,\ 000,\ 000$$

que está también asociado $g_2(Y)$. Al igual ocurre con $\alpha g_1(X)$ que es enviado a $Y g_2(Y)$, $\alpha^2 g_1(X)$ a $Y^2 g_2(Y)$, $X g_1(X)$ a $Y^3 g_2(Y)$, etc.

Este es el único ejemplo (en palabras de MacWilliams y Sloane) no trivial conocido de un código cíclico que es enviado mediante este procedimiento a otro código cíclico.

Los códigos del ejemplo 2.30 son tan buenos que es interesante estudiar su comportamiento para valores grandes de m.

Teorema 2.32. *El código RS $[N = 2^m - 1, K, D]$ con ceros $\alpha, \alpha^2, \ldots, \alpha^{D-1}$ contiene al código binario primitivo BCH de longitud N y distancia prevista D. De igual forma los códigos RS extendidos contienen a los códigos BCH extendidos*[2].

Demostración. Si c pertenece al código binario BCH, c es un vector binario con $c(\alpha) = \ldots = c(\alpha^{D-1}) = 0$ y, por tanto, también pertenece al código RS. ∎

Teorema 2.33. *Los códigos binarios obtenidos a partir de códigos Reed-Solomon y con parámetros*

$$n = (m+1)(2^m - 1), \quad k = mK, \quad d \geq 2D = 2\left(2^m - k\right),$$

para cualquier $K = 1, \ldots, 2^m - 2$,

o a partir del código RS extendido con parámetros

$$\left[(m+1)2^m, mK, d \geq 2\left(2^m - K + 1\right)\right],$$

para $K = 1, \ldots, 2^m - 1$,

son asintóticamente malos. Esto es, no contienen una familia infinita de códigos tales que tanto la tasa de información como la distancia relativa tiendan a valores positivos.

Demostración. Dado que este resultado no lo utilizamos a lo largo del trabajo, la demostración se puede consultar en [MS77, Ch. 10, Th. 3]. ∎

No obstante, usando una construcción ligeramente más compleja, es posible obtener códigos binarios asintóticamente buenos a partir de códigos RS. Esto se puede consultar en la sección 11 de nuestra referencia y son los denominados códigos Justesen, también recomiendo consultar el artículo original [Jus72] en el que se basa y amplía dicha sección. Nosotros no vamos a entrar en este punto puesto que perderíamos el objetivo de este trabajo, pero sí al menos hacemos mención a este hecho que marca una diferencia con respecto a los códigos BCH y de ahí la importancia de distinguir a esta familia de códigos. Encomiendo también al lector más curioso darle una lectura al artículo [BJ74] en el que Berlekamp y Justesen discuten la existencia de algunos códigos cíclicos que asintóticamente son buenos.

Otro hecho importante que caracteriza a los códigos RS es su utilidad para corregir muchos errores en ráfagas.

[2] El concepto de código BCH extendido es el mismo que el de los códigos RS extendidos definidos en el teorema 2.27.

Definición 2.34. Decimos que un vector $x \in \mathbb{F}_{q^m}$ es una ráfaga si todas sus componentes no nulas son consecutivas. Estas componentes se pueden separar en bloques de longitud b a lo largo de todo el vector, son fácilmente identificables ya que el bloque comienza y termina con las únicas componentes no nulas de x.

Los códigos binarios obtenidos a partir de códigos RS son particularmente útiles para corregir muchas ráfagas. Una ráfaga binaria de longitud b puede afectar a lo sumo r símbolos de $\mathbb{F}_{2^m}$ adyacentes, donde r viene dado por

$$(r-2)m + 2 \leq b \leq (r-1)m + 1.$$

Por tanto, si D es mucho mayor que r, podremos corregir muchas ráfagas. Esta desigualdad es por la expresión de los elementos de $\mathbb{F}_{2^m}$ como vectores de $(\mathbb{F}_2)^m$: $r-2$ $m-$tuplas deben verse afectadas de lleno, que corresponderán a $b-2$ elementos de $\mathbb{F}_{2^m}$, y dos elementos más al menos se verán afectados, correspondientes a la última componente del primer vector/elemento de la ráfaga, y la primera del último. Si cambiamos m por $m+1$, obtenemos $(r-1)m+2$, así que por la misma razón la cota superior debe ser $(r-1)m+1$.

2.2.3 Codificación de códigos Reed-Solomon

Por ser los códigos Reed-Solomon cíclicos, estos pueden ser codificados por cualquiera de los métodos descritos previamente. No obstante, el siguiente método (fue el que originalmente propusieron Reed y Solomon) es bastante simple y presenta ventajas prácticas.

Antes de empezar con la codificación, necesitamos definir el siguiente concepto:

Definición 2.35 (Polinomio de Mattson-Solomon)**.** El polinomio de Mattson-Solomon de $a(X)$ es la transformación lineal de $a(X)$ definida por

$$A(Z) = MS(a(X)) = \sum_{j=0}^{n-1} a\left(\alpha^{-j}\right) Z^j.$$

La transformación inversa de Mattson-Solomon o transformación de Fourier viene dada por

$$a(X) = MS^{-1}(A(Z)) = \frac{1}{n}\sum_{i=0}^{n-1} A(\alpha^i) X^i.$$

Sea $u = (u_0, u_1, \ldots, u_{K-1})$, $u_i \in \mathbb{F}_q$ el mensaje de símbolos a codificar, y sea

$$u(Z) = \sum_{i=0}^{K-1} u_i Z^i.$$

Entonces la palabra código correspondiente a u será el vector c cuyo polinomio de Mattson-Solomon es $Nu(Z)$, donde $N = q - 1$. Por tanto,

$$c = \left(u(1), u(\alpha), \ldots, u(\alpha^{N-1})\right). \tag{2.1}$$

O, en el caso de los códigos extendidos con un dígito de control

$$c = \left(u(0), u(1), u(\alpha), \ldots, u(\alpha^{N-1})\right).$$

Podemos ver que, en efecto, c está en el código RS comprobando que

$$c(X) = \sum_{i=0}^{N-1} c_i X^i$$

tiene por ceros a $\alpha, \alpha^2, \ldots, \alpha^{D-1}$. De hecho, el polinomio MS de c, $Nu(Z)$, es igual a

$$\sum_{j=0}^{N-1} A_{-j} z^j \quad \text{donde } A_{-j} = c(\alpha^{-j})$$

Por tanto, si igualamos los coeficientes de ambos polinomios y recordando que $N = -1$ en $\mathbb{F}_q$, tenemos que

$$c(1) = -u_0, c(\alpha^{-1}) = -u_1, \ldots, c(\alpha^{-K+1}) = -u_{K-1}$$

y $c(\alpha^j) = 0$ para $1 \leq j \leq N - K = D - 1$. Con lo que obtenemos que c está en el código RS y este es un método para codificar el código. Nótese no obstante que este codificador no es sistemático.

Definición 2.36. Sea $\mathcal{C}$ un $\left(n, M = q^k, d\right)$ código (lineal o no lineal) sobre $\mathbb{F}_q$ con un codificador que envía el mensaje $u_0, \ldots, u_{k-1}$ a las palabras código $c_0, \ldots, c_{n-1}$. El codificador se dirá que es sistemático si existen coordenadas $i_0, \ldots, i_{k-1}$ tales que $u_0 = c_0, \ldots, u_{k-1} = c_{k-1}$, es decir, si el mensaje no cambia en la palabra código.

Ejemplo 2.37. Por ejemplo, el codificador que realiza la siguiente acción

mensaje		palabra código
00	$\mapsto$	000
01	$\mapsto$	010
10	$\mapsto$	101
11	$\mapsto$	111

es sistemático (con $i_0 = 0$ y $i_1 = 1$), mientras que el mismo código con codificador

mensaje		palabra código
00	$\mapsto$	000
01	$\mapsto$	111
10	$\mapsto$	101
11	$\mapsto$	010

no lo es. Un código que no sea sistemático revuelve el mensaje. A pesar de conseguir calcular el vector error y la palabra código sea recuperada, en un código no sistemático es necesario realizar varios cálculos más para recuperar el mensaje original.

2.2.4 Códigos Reed-Solomon generalizados

Una clase ligeramente más general de códigos que los RS se obtienen a partir de la ecuación 2.1 si se reemplaza por la siguiente

$$c = \left(v_1 u(1), v_2 u(\alpha), \ldots, v_N u(\alpha^{N-1})\right),$$

donde los v_i son elementos no nulos de $\mathbb{F}_q$. La ecuación 2.1 es el caso en el que todos los $v_i = 1$. Esto nos sugiere la existencia de más generalizaciones del código.

Definición 2.38. Sea $\alpha = (\alpha_1, \ldots, \alpha_N)$ donde $\alpha_i \in \mathbb{F}_{q^m}$ todos distintos y sea $v = (v_1, \ldots, v_N)$ donde $v_i \in \mathbb{F}_{q^m}$, no nulos pero no necesariamente distintos. Entonces el código RS generalizado, denotado por $\mathrm{GRS}_K(\alpha, v)$, consiste en todos los vectores

$$(v_1 F(\alpha_1), v_2 F(\alpha_2), \ldots, v_N F(\alpha_N))$$

donde $F(Z)$ varía entre todos los polinomios de grado menor que K con coeficientes en $\mathbb{F}_{q^m}$. Este es un $[N, K]$ código sobre $\mathbb{F}_{q^m}$. Ya que F tiene a lo sumo $K - 1$ ceros, la distancia mínima es al menos $N - K + 1$, por lo que es igual a $N - K + 1$. Tenemos que es un código MDS.

Teorema 2.39. *El dual de $GRS_K(\alpha, v)$ es $GRS_{N-K}(\alpha, v')$ para algún v'.*

Demostración. Supongamos que $K = N-1$, y sea $\mathcal{D}$ el código dual de $\mathrm{GRS}_{N-K}(\alpha, v)$. Entonces $\mathcal{D}$ tiene dimensión 1 y consiste en todos los múltiplos por un escalar de algún vector fijo $v' = (v'_1, \ldots, v'_N)$. Tenemos que demostrar que todos los $v'_i \neq 0$, para ello

tenemos que v' satisface:

$$\begin{pmatrix} 1 & \cdots & 1 \\ \alpha_1 & \cdots & \alpha_N \\ \vdots & \ddots & \vdots \\ \alpha_1^{N-2} & \cdots & \alpha_N^{N-2} \end{pmatrix} \begin{pmatrix} v_1 v_1' \\ v_2 v_2' \\ \vdots \\ v_N v_N' \end{pmatrix} = 0 \tag{2.2}$$

Si ocurre que algún $v_i' = 0$, entonces obtenemos un conjunto de ecuaciones cuya matriz de coeficientes es una matriz de Vandemonde y cuadrada, por lo que tenemos que todos los $v_i v_i' = 0$ y por tanto se anulan todos los v_i', lo que es imposible.

Entonces el código $\mathrm{GRS}_K(\alpha, v)$ es el dual de $\mathrm{GRS}_{N-K}(\alpha, v')$, para todo $K < N-1$, ya que

$$\sum_{i=1}^{N} \left(\alpha_i^s v_i\right)\left(\alpha_i^t v_i'\right) = \sum_{i=1}^{N} \alpha_i^{s+t} v_i v_i' = 0$$

para $s \leq K-1$, $t \leq N-K-1$, por 2.2. ∎

2.2.5 Descodificando códigos RS

Al ser los códigos RS casos especiales de los códigos BCH, estos pueden ser descodificados por los métodos vistos en 2.1.3. Es interesante comentar el método original de descodificación de Reed y Solomon por su valor teórico, aunque en la práctica no sea útil.

Supongamos que enviamos la palabra código 2.1 y que ocurre un vector error $e = (e_0, \ldots, e_{N-1})$, y recibimos $y = (y_0, \ldots, y_{N-1})$. Por tanto el descodificador conoce

$$\begin{aligned} y_0 &= e_0 + u_0 + u_1 + u_2 + \ldots + u_{K-1}, \\ y_1 &= e_1 + u_0 + \alpha u_1 + \ldots + \alpha^{K-1} u_{K-1}, \\ &\vdots \\ y_{N-1} &= e_{N-1} + u_0 + \alpha^{N-1} u_1 + \alpha^{2(N-1)} u_2 + \ldots + \alpha^{(K-1)(N-1)} u_{K-1}. \end{aligned} \tag{2.3}$$

Si no hay errores, $e = 0$, y cualesquiera K de estas N ecuaciones pueden ser resueltas para calcular el mensaje $u = (u_0, \ldots, u_{K-1})$, ya que la matriz de coeficientes es una matriz de Vandermonde. Por tanto, hay $\binom{N}{K}$ determinaciones o candidatos a ser el u correcto.

Si hay muchos errores, algún conjunto de K ecuaciones nos dará un u incorrecto. Pero no podemos recibir muchos candidatos a u incorrectos.

Teorema 2.40. *Si ocurren w errores, un u incorrecto recibirá a lo sumo $\binom{w+K-1}{K}$ candidatos. El u correcto recibirá al menos $\binom{N-w}{K}$ candidatos.*

Demostración. Por ser las ecuaciones del sistema 2.3 independientes, cualesquiera K de ellas tienen exactamente una solución u. Para obtener más de un candidato, u debe ser solución de más de K ecuaciones. Un u incorrecto puede ser solución de, a lo sumo, $w + K - 1$ ecuaciones, consistiendo en w ecuaciones erróneas y $K - 1$ correctas. (Porque si u es la solución de K ecuaciones correctas, entonces u es correcto.) Por tanto, un u erróneo puede ser solución de a lo sumo $\binom{w+K-1}{K}$ conjuntos de K ecuaciones. Claramente hay $\binom{N-w}{K}$ conjuntos de ecuaciones correctas con las que obtenemos el u correcto. ∎

Y con este último teorema concluimos el capítulo central de este trabajo, donde hemos podido estudiar en profundidad dos familias muy especiales de códigos cíclicos de gran utilidad en el campo de las comunicaciones a larga distancia.

3 Códigos de Goppa y Criptosistemas basados en códigos

El objetivo de este capítulo final es exponer una clase de códigos que son la base del criptosistema de McEliece, siendo aquí donde unimos el interés tanto en que la información llegue de forma correcta como cifrada. Intentaremos no ser muy detallados en los resultados, ya que si profundizamos en este tema alcanzaríamos material suficiente para realizar otro trabajo más de fin de grado, pero es interesante tratarlo aunque sea por encima para en un futuro poder continuarlo en otro posible trabajo.

Para ello utilizaremos como referencia el artículo [Sin19] que resume los contenidos de los artículos más relevantes y nos lleva de forma ordenada al cometido de este capítulo.

3.1 Códigos de Goppa clásicos

Los códigos de Goppa reciben este nombre debido a su descubridor, Valeri Denísovich Goppa (nacido en 1939 en la R. S. F. S. de Rusia). Se pueden consultar los textos originales, que están escritos en ruso, en los artículos [Gop70, Gop71]; aunque gracias a Berlekamp [Ber73] también tenemos una traducción al inglés que, a modo de resumen, nos expone el contenido de dichos artículos.

Estos códigos fueron descubiertos en 1970 a partir de la relación entre la geometría algebraica y los códigos (hay un amplio campo de estudio sobre los códigos algebraico-geométricos que dan para otro trabajo). Poco después de los trabajos de Goppa se descubrió que estos códigos son también una subclase muy interesante de códigos alternantes[1], definidos por H. J. Helgert en 1974 [Hel74].

[1] Un código alternante se define como la restricción lineal del dual de un código Reed-Solomon Generalizado a $\mathbb{F}_q$.

Los códigos de Goppa tienen un algoritmo eficiente de descodificación gracias a N. Patterson [Pat75] que aporta una mejora particular al algoritmo de Berlekamp visto en el capítulo anterior.

Definición 3.1. Sean $g(X) = g_0 + g_1X + g_2X^2 + \ldots + g_tX^t \in \mathbb{F}_{q^m}[X]$ y $L = \{\alpha_1, \alpha_2, \ldots, \alpha_n\} \subseteq \mathbb{F}_{q^m}$ tales que $g(\alpha_i) \neq 0$, para todo $\alpha_i \in L$.
Entonces al código definido por

$$\Gamma(L, g(X)) := \left\{ c = (c_1, c_2, \ldots, c_n) \in \mathbb{F}_q^n \ : \ \sum_{i=1}^{n} \frac{c_i}{X - \alpha_i} \equiv 0 \text{ mód } g(X) \right\}$$

se le conoce como código de Goppa con parámetros $g(X)$ y L.

Para cada $1 \leq i \leq n$, que $g(\alpha_i) \neq 0$ equivale a que mcd $(X - \alpha_i, g(X)) = 1$, con lo que la fracción $\frac{1}{X-\alpha_i}$ se calcula en $\mathbb{F}_{q^m}[X]/\langle g(X)\rangle$ de la forma siguiente:
Bajo las condiciones anteriores se puede comprobar fácilmente que:

1. El inverso multiplicativo de $(X - \alpha_i)$ en el anillo cociente $\mathbb{F}_{q^m}[X]/\langle g(X)\rangle$ existe.
2. El valor de $(X - \alpha_i)^{-1}$ en $\mathbb{F}_{q^m}[X]/\langle g(X)\rangle$ es $-\left(\frac{g(X)-g(\alpha_i)}{X-\alpha_i}\right) g(\alpha_i)^{-1}$.

Un vector $c \in \Gamma(L, g)$ si y solo si $\sum_{i=1}^{n} c_i \left(\frac{g(X)-g(\alpha_i)}{X-\alpha_i}\right) g(\alpha_i)^{-1} \equiv 0$ mód $g(X)$.

A partir de estas propiedades obtenemos la siguiente proposición:

Proposición 3.2. *Un vector $c \in \Gamma(L, g)$ si y solo si $\sum_{i=1}^{n} c_i \left(\frac{g(X)-g(\alpha_i)}{X-\alpha_i}\right) g(\alpha_i)^{-1} = 0$ como polinomio en $\mathbb{F}_{q^m}[X]$. Esta suma no puede ser un múltiplo de g no nulo ya que su grado es estrictamente menor.*

Esta proposición, a ojo de buen algebrista y a estas alturas del trabajo, nos está caracterizando una matriz de control para los códigos de Goppa:

Corolario 3.3. *Una matriz de control sobre $\mathbb{F}_{q^m}$ para un código de Goppa $\Gamma(L, g(X))$ es:*

$$H = \begin{pmatrix} g_t g(\alpha_1)^{-1} & \cdots & g_t g(\alpha_n)^{-1} \\ (g_t\alpha_1 + g_{t-1})g(\alpha_1)^{-1} & \cdots & (g_t\alpha_n + g_{t-1})g(\alpha_n)^{-1} \\ \vdots & \ddots & \vdots \\ (g_t\alpha_1^{t-1} + \ldots + g_1)g(\alpha_1)^{-1} & \cdots & (g_t\alpha_n^{t-1} + \ldots + g_1)g(\alpha_n)^{-1} \end{pmatrix}.$$

Esta matriz se puede separar en el producto de tres matrices como $H = CXY$, donde:

$$C = \begin{pmatrix} g_t & 0 & \cdots & 0 \\ g_{t-1} & g_t & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ g_1 & g_2 & \cdots & g_t \end{pmatrix}, \; X = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_n \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{t-1} & \alpha_2^{t-1} & \cdots & \alpha_n^{t-1} \end{pmatrix}$$

$$Y = \begin{pmatrix} g(\alpha_1)^{-1} & 0 & \cdots & 0 \\ 0 & g(\alpha_2)^{-1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & g(\alpha_n)^{-1} \end{pmatrix}$$

Por tanto, tenemos que $c \in \Gamma(L, g(X))$ si y solo si $cH^t = 0 \iff c(CXY)^t = 0 \iff cY^tX^tC^t = 0$. Por ser C invertible, nos queda la equivalencia:

$$c \in \Gamma(L, g(X)) \iff c(XY)^t = 0.$$

Nota 3.4. Podemos considerar la matriz XY sobre $\mathbb{F}_{q^m}$ como una matriz de control del código $\Gamma(L, g(X))$. La matriz tiene la siguiente forma:

$$XY = \begin{pmatrix} g(\alpha_1)^{-1} & g(\alpha_2)^{-1} & \cdots & g(\alpha_n)^{-1} \\ \alpha_1 g(\alpha_1)^{-1} & \alpha_2 g(\alpha_2)^{-1} & \cdots & \alpha_n g(\alpha_n)^{-1} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{t-1} g(\alpha_1)^{-1} & \alpha_2^{t-1} g(\alpha_2)^{-1} & \cdots & \alpha_n^{t-1} g(\alpha_n)^{-1} \end{pmatrix}.$$

Nota 3.5. Nótese que los elementos de $\mathbb{F}_{q^m}$ se pueden ver como vectores de longitud m sobre $\mathbb{F}_q$ (ambos cuerpos son isomorfos como espacios vectoriales), con lo que tenemos una matriz de control sobre $\mathbb{F}_q$ para el código $\Gamma(L, g(X))$ de dimensión $mt \times n$, con al menos t columnas cualesquiera linealmente independientes sobre $\mathbb{F}_q$. Por tanto, la distancia de Hamming de un código de Goppa cumple $d(\Gamma(L, g(X))) \geq t + 1$.

Ocurre también que la matriz XY definida sobre $\mathbb{F}_q$ a lo sumo tiene mt filas linealmente independientes; lo que implica que $\text{rg}(XY) \leq mt$ y que $\text{Null}(XY) \geq n - mt$, dicho de otra forma, $\dim_{\mathbb{F}_q} \Gamma(L, g(X)) \geq n - mt$.

Los códigos de Goppa no dejan de ser códigos cíclicos, con lo que tanto la codificación como la descodificación y la corrección de errores mediante síndromes se realizan de la forma habitual. El esquema que tenemos se lo debemos a Sugiyama, Kasahara, Hirasawa y Namekawa en el artículo [SKHN75], en otras referencias, simplemente lo denominan como método de Sugiyama.

Vamos a pasar ahora a definir qué forma tiene el polinomio síndrome para estos códigos, para ello, necesitamos recordar la siguiente definición que aparece páginas atrás:

Sea $y = (y_1, y_2, \ldots, y_n)$ el vector recibido con r errores, donde exigimos que $d \geq 2r+1$ para asegurar la capacidad de corregir el máximo de errores. Sea $L = \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$,

$$y = (y_1, y_2, \ldots, y_n) = (c_1, c_2, \ldots, c_n) + (e_1, e_2, \ldots, e_n)\,,$$

con $e_i \neq 0$ en exactamente r posiciones. Necesitamos:

- Localizar las posiciones de los errores, definamos $B = \{i \,:\, e_i \neq 0 \text{ para } 1 \leq i \leq n\}$.
- Encontrar los valores de los e_i tales que $i \in B$.

Para esto recurrimos a los siguientes polinomios:

Definición 3.6. Polinomio localizador de errores $L(X)$ y polinomio evaluador de errores $E(X)$:

$$L(X) = \prod_{i \in B}(X - \alpha_i),$$

$$E(X) = \sum_{i \in B} e_i \prod_{j \in B, j \neq i} (X - \alpha_j).$$

Definición 3.7. Se define el síndrome del vector recibido y como $S_y(X)$, donde:

$$S_y(X) := \sum_{i=1}^{n} \frac{y_i}{X - \alpha_i} = \sum_{i=1}^{n} \frac{c_i}{X - \alpha_i} + \sum_{i \in B} \frac{e_i}{X - \alpha_i} = \sum_{i \in B} \frac{e_i}{X - \alpha_i} \text{ mód } g(X).$$

La siguiente proposición viene en la referencia que estamos trabajando, es interesante recordar estas propiedades porque se han mencionado varias páginas atrás en la descodificación de los códigos BCH; recordemos que los Goppa son una generalización de estos códigos.

Proposición 3.8. *Sea e el vector error con peso r $\left(r \leq \left\lfloor \frac{t}{2} \right\rfloor\right)$. Sean $L(X)$, $E(X)$ y $S_y(X)$ como los hemos descrito previamente. Entonces se cumple:*

1. $\deg(L(X)) = r$.
2. $\deg(E(X)) \leq r - 1$.
3. *mcd* $(L(X), E(X)) = 1$.
4. $e_k = \frac{E(\alpha_k)}{L'(\alpha_k)}$, *donde* $k \in B$ *y* L' *representa la derivada de* L.
5. $L(X)S_y(X) \equiv E(X)$ *mód* $g(X)$.

Con esta proposición, tenemos los ingredientes para el esquema de detección y corrección de errores del código, que es el mismo que el ya visto en el capítulo anterior para los códigos BCH.

Pero en el caso de que el código Goppa sea binario, tenemos un esquema alternativo gracias a Patterson. Este algoritmo calcula el síndrome del vector recibido $S_y(X)$ y después resuelve la ecuación clave $L(X)S_y(X) \equiv E(X)$ mód $g(X)$ con $E(X) = L'(X)$, utilizando fuertemente que el código es binario. El polinomio localizador de errores se puede separar en potencias pares e impares de X tal que $L(X) = a^2(X) + Xb^2(X)$, por estar definidos en un cuerpo de característica 2.

El algoritmo de Patterson se puede describir de la forma siguiente:

Parámetros de entrada: El vector recibido y y el código binario de Goppa $\Gamma(L, g)$.

1. Calcular el síndrome $S_y(X)$, que pertenece a $\mathbb{F}_{2^m}[X]/\langle g(X)\rangle$.
2. Calcular $T(X) = S_y(X)^{-1}$ mód $g(X)$.
3. Calcular $P(X) = \sqrt{T(X) + X}$ mód $g(X)$.
4. Calcular $u(X)$ y $v(X)$ con $u(X) = v(X)S_y(X)$ mód $g(X)$.
5. Calcular el polinomio localizador $L(X) = [u(X)]^2 + X[v(X)]^2$.
6. Encontrar las raíces de $L(X)$.
7. Encontrar las posiciones con errores, es decir, el vector error e.

Parámetro de salida: el vector error e.

Ahora vamos a ver un ejemplo bastante completo sobre cómo trabajar con un código de Goppa.

Ejemplo 3.9. Sea la extensión $\mathbb{F}_9 = \mathbb{F}_3[X]/\langle X^2 + 2X + 2\rangle$ sobre $\mathbb{F}_3$ y sea α una de las raíces del polinomio $X^2 + 2X + 2$. Pasemos ahora a listar todos los elementos generados por las potencias de α:

$$\begin{array}{rcccccl}
0 & = & & & & = & (0,0)\,; \\
1 & = & 1 & & & = & (1,0)\,; \\
\alpha & = & & & \alpha & = & (0,1)\,; \\
\alpha^2 & = & 1 & + & \alpha & = & (1,1)\,; \\
\alpha^3 & = & 1 & + & 2\alpha & = & (1,2)\,; \\
\alpha^4 & = & 2 & & & = & (2,0)\,; \\
\alpha^5 & = & & & 2\alpha & = & (0,2)\,; \\
\alpha^6 & = & 2 & + & 2\alpha & = & (2,2)\,; \\
\alpha^7 & = & 2 & + & \alpha & = & (2,1)\,.
\end{array}$$

Consideremos el código de Goppa $\Gamma(L, g(X))$ con los parámetros:

$$g(X) = X(X - \alpha^7) = X^2 + \alpha^3 X,$$

$$L = \left\{\alpha^i \,:\, 0 \leq i \leq 6\right\}.$$

Entonces, tenemos que la matriz de control sobre $\mathbb{F}_9$ es:

$$H = \begin{pmatrix} \alpha^7 & \alpha^2 & \alpha^6 & \alpha^7 & \alpha & \alpha & \alpha^2 \\ 1 & \alpha^4 & \alpha & \alpha^3 & \alpha^6 & \alpha^7 & \alpha^2 \end{pmatrix}.$$

Equivalentemente, H sobre $\mathbb{F}_3$ tiene la forma:

$$H^* = \begin{pmatrix} 2 & 1 & 2 & 2 & 0 & 0 & 1 \\ 1 & 1 & 2 & 1 & 1 & 1 & 1 \\ 1 & 2 & 0 & 1 & 2 & 2 & 1 \\ 0 & 0 & 1 & 2 & 2 & 1 & 1 \end{pmatrix}.$$

Ahora, haciendo uso que el $\ker(H^*)$ produce la matriz generatriz del código, tenemos:

$$G = \begin{pmatrix} 1 & 0 & 0 & 2 & 2 & 1 & 0 \\ 0 & 1 & 0 & 2 & 2 & 0 & 1 \\ 0 & 0 & 1 & 2 & 2 & 0 & 0 \end{pmatrix}.$$

Los parámetros de este código son $[7, 3, d \geq 3]$. Ahora, supongamos que enviamos el mensaje $m = (0, 0, 0)$ (por no complicarnos con los cálculos). Nuestro primer paso es codificar el mensaje, a saber, hacemos $m \cdot G = (0, 0, 0, 0, 0, 0, 0)$.

Supongamos que recibimos el vector $y = (0, 0, 0, 0, 0, 0, 2)$, claramente a golpe de vista observamos que se ha producido un error. Nuestro descodificador automáticamente detectará que el error se ha producido ($y \cdot H^t = (2, 2, 2, 2) \neq 0$) y será capaz de corregirlo ya que la distancia mínima es al menos 3. El objetivo que tenemos ahora es encontrar el vector error, para ello calculamos:

1. Síndrome:

$$S_y(X) = \sum_{i=0}^{6} \frac{y_i}{X - \alpha_i} = \frac{2}{X - \alpha^6} \equiv 1 + (\alpha + 1)X \text{ mód } g(X).$$

2. Sustituyendo $L(X) = l_0 + X$ y luego calculando $L(X)S_y(X)$ mód $X^2 + \alpha^3 X$ tenemos:

$$L(X)S_y(X) = (l_0 + X)\,(1 + (\alpha + 1)X) = l_0 + (1 + (\alpha + 1)l_0)\,X + (\alpha + 1)X^2 \equiv$$

$$\equiv l_0 + (1 + \alpha l_0 + l_0 + \alpha)\,X = l_0 + ((1 + \alpha) + (1 + \alpha)l_0)\,X$$

Por tanto, para $E(X) = e_0$, tenemos el sistema de ecuaciones por igualación de los términos de los polinomios:

$$\begin{cases} e_0 &= l_0 \\ 0 &= (1 + \alpha) + (1 + \alpha)l_0 \end{cases}.$$

La solución del sistema es $l_0 = 2 = \alpha^4$, $e_0 = \alpha^4$. Con lo que tenemos:

$$L(X) = X + \alpha^4, \quad E(X) = \alpha^4.$$

3. La raíz de $L(X)$ es $\alpha^8 = 1 = \alpha_6$, por tanto, el conjunto de las posiciones de los errores es:
$$B = \{i \, : \, L(\alpha_i) = 0\} = \{7\}\,.$$
4. La componente errónea es $e_7 = \frac{\alpha^4}{1} = \alpha^4 = 2$.
5. La palabra código enviada es:

$$c = y - e = (0,0,0,0,0,0,2) - (0,0,0,0,0,0,2) = (0,0,0,0,0,0,0)$$

Y podemos descodificar el mensaje por el método habitual, obteniendo $m = (0,0,0)$. Hemos tenido que revisar los cálculos en sage, posiblemente los cambios en métodos predefinidos de sage han condicionado el ejemplo, con lo que podríamos considerar que hemos hecho un ejemplo nuevo a partir del dado por el artículo.

3.2 Criptografía basada en códigos

El nacimiento de la criptografía basada en códigos se debe al trabajo de Robert J. McEliece en 1978 [McE78]. McEliece es el primero en implementar el uso de códigos binarios de Goppa para desarrollar el criptosistema de clave pública basado en códigos. Hay muchas razones por las que los códigos de Goppa fueron la primera elección para el criptosistema de McEliece. En el artículo referido previamente nos explica, en palabras suyas:

«Usamos el hecho de que los códigos de Goppa, en general, tienen un algoritmo de descodificación rápido mientras que los códigos lineales, en general, carecen de ello. Construimos un criptosistema de clave pública que aparenta ser seguro a la par que permite tasas de información realmente rápidas. Este tipo de criptosistemas es ideal para usarlos en redes de comunicación multiusuario, como los previstos por la NASA para la distribución de datos adquiridos en el espacio.»

En dicho artículo, también nos hace referencia a otro [BMvT78], en el que demuestran que el problema general de descodificar códigos lineales es $NP-$completo; y en este hecho se basa la seguridad del criptosistema tomando los parámetros valores suficientemente grandes.

Podemos dar otro motivo de la elección de los códigos de Goppa y es que son fáciles de generar pero complicados de encontrar, dicho de otra forma, cualquier polinomio irreducible sobre $\mathbb{F}_{2^m}$ puede usarse para construir un código de Goppa pero las matrices generatrices de dichos códigos son casi aleatorias (no siguen un patrón fácil de hallar).

Para cualquier longitud n fijada, hay muchos códigos Goppa distintos. Aunque el número exacto de códigos Goppa, dada la longitud n del código y el grado del polinomio generador t, no es conocido, Ryan y Fitzpatrick [RF03] encontraron una forma de calcular cotas superiores, que son exactas para algunos valores pequeños de los parámetros. Por ejemplo, para un código de Goppa de longitud 128 capaz de corregir al menos 10 errores existen alrededor de $1{,}04 \times 10^{15}$ códigos distintos, mientras que si mantenemos la longitud y queremos corregir al menos 15 errores, existen alrededor de $2{,}38 \times 10^{25}$. De hecho, el número de códigos de Goppa crece exponencialmente con la longitud del código y el grado del polinomio generador. Por estos detalles los códigos de Goppa siguen siendo la familia principal de códigos usados en el criptosistema de McEliece.

Esencialmente, existen dos tipos de criptosistemas basados en códigos sobre cuya estructura se basan todos los demás criptosistemas de esta clase. El primer sistema es el Criptosistema de McEliece, en el que la matriz generatriz del código de Goppa se oculta permutando y mezclando las entradas de la matriz, y la hacemos pública. El texto cifrado se genera mediante la codificación del mensaje con la matriz disponible en clave pública y aplicándole XOR con algún error de peso pequeño, dependiendo de los parámetros del código Goppa.

El segundo sistema es el criptosistema Niederreiter[2], en el que el mensaje es un vector aleatorio con un error de peso pequeño. La clave pública se convierte en una matriz de control de un código generalizado de Reed Solomon (GRS) a la que permutamos y mezclamos las entradas.

Sidelnikov y Shestakov [SS92] demostraron que los códigos GRS propuestos por Niederreiter eran una mala opción en su criptosistema, en cambio se ha demostrado que los códigos de Goppa funcionan bien. Buscando artículos al respecto, hay también algunos que bajo ciertos parámetros demuestran que los códigos GRS funcionan bien, pero no dan la amplia generalidad que nos brindan los de Goppa.

[2]Esta va a ser la breve mención que hagamos al criptosistema de Niederreiter, está bien por usar otros códigos estudiados en el trabajo, pero por extensión del mismo no vamos a entrar en detalles. Para más detalles se puede consultar la referencia de este capítulo, [Sin19].

3.2.1 Problemas difíciles en la teoría de códigos

En general, los problemas que plantean la teoría de códigos y en los que se fundamenta la seguridad de los criptosistemas basados en códigos son los siguientes:

1. Problema de descodificación general: Dado un $[n, k]-$ código $\mathcal{C}$ sobre $\mathbb{F}_q$, un natural t_0 y un vector $c \in \mathbb{F}_q^n$, encontrar una palabra código $x \in \mathcal{C}$ tal que $d(x, c) \leq t_0$.
2. Problema de descodificación mediante síndromes: Dada una matriz H y un vector s, ambos sobre $\mathbb{F}_q$, y un natural t_0; encontrar un vector $x \in \mathbb{F}_q^n$ con peso $w(x) = t_0$ tal que $x \cdot H^t = s$.
   Se demostró que estos problemas son $NP-$completos en 1978 en [BMvT78] para códigos binarios, y en 1997 Alexander Barg en [Bar97] (este artículo viene también recogido en [PHB98]) lo demostró para códigos sobre cualquier cuerpo finito.
3. Descodificación mediante síndromes con parámetro de Goppa: Dada una matriz binaria H de tamaño $2^m \times r$ y un síndrome s, decidir si existe una palabra código de peso r/m tal que $x \cdot H^t = s$.
   Este problema es también NP, la prueba de este hecho se puede encontrar en [AFS05].
4. Distinción de códigos de Goppa: Dada una matriz H $r \times n$, decidir cuándo H es la matriz de control de un código Goppa.
   En 2013, [FGUO+13] demostraron que los códigos binarios de Goppa con «tasas de información altas» se pueden distinguir de los códigos lineales aleatorios.

3.2.2 Descodificación de conjuntos de información

Un atacante que intercepte un mensaje encriptado y tiene dos opciones para recuperar el mensaje original m.

- Encontrar el código secreto, es decir, encontrar la matriz generatriz G a partir de la matriz $\hat{G}$ que está mezclada y permutada.
- Descodificar y sin saber un algoritmo eficiente de descodificación para la matriz $\hat{G}$.

Los ataques del primer tipo son conocidos como ataques estructurales.

Definición 3.10. Sean G una matriz generatriz de un $[n, k]$ −código, I un subconjunto de $\{1, \ldots, n\}$ y G_I la submatriz $k \times k$ de G definida por las columnas de G cuyos subíndices pertenezcan a I. Si G_I es invertible, entonces I es un conjunto de información.

Una definición equivalente a partir de la matriz de control:

Definición 3.11. Usando una matriz de control H, un subconjunto I de $\{1, \ldots, n\}$ es un IS (conjunto de información) si y solo si la submatriz formada por las columnas de índices $\{1, \ldots, n\} \setminus I$ es no singular.

La descripción en términos de la matriz de control, aunque menos intuitiva, favorece la explicación de cómo los algoritmos de descodificación de conjuntos de información (ISD en inglés, como notaremos de aquí en adelante) funcionan. Este algoritmo induce un ataque genérico contra todos los criptosistemas basados en códigos, independientemente de nuestro esquema actual. El algoritmo básico de ISD fue dado por Prange [Pra62] con mejoras de Leon [Leo88], Lee-Brickell [LB88], Stern [Ste89] y Canteaut-Chabaud [CC98].

Un atacante no conoce el código secreto y esto le lleva a tener que descodificar un código aparentemente aleatorio sin ninguna estructura evidente. Los algoritmos más conocidos que no explotan ninguna estructura de código se basan en la ISD.

La idea es encontrar un conjunto de coordenadas de un vector confuso que estén libres de errores (es decir, un conjunto de información, como se definió anteriormente) y de tal manera que la restricción de la matriz generatriz del código en estas posiciones sea invertible. Luego, el mensaje original se puede calcular multiplicando el vector cifrado por el inverso de la submatriz.

3.3 Criptosistema de McEliece

La criptografía de clave pública reciente se basa en gran medida en problemas de teoría de números, como la factorización o el cálculo del logaritmo discreto. Estos sistemas constituyen una excelente opción en muchas aplicaciones, y su seguridad está bien definida y comprendida. Sin embargo, uno de los principales inconvenientes es que serán vulnerables una vez que los ordenadores cuánticos de un tamaño adecuado estén disponibles. Entonces hay una gran necesidad de sistemas alternativos que resistan a los atacantes equipados con tecnología cuántica.

Con el desarrollo de ordenadores cuánticos, el riesgo para la criptografía actual está aumentando. El próximo escenario para el mundo criptográfico se basa en los criptosistemas postcuánticos, o podemos decir criptosistemas resistentes a los procesadores cuánticos. Los sistemas de cifrado basados en la teoría de la códigos son un tipo de criptosistemas que son capaces de resistir la computación cuántica, y esto proporciona un área esperanzadora en la criptografía postcuántica.

La versión original del criptosistema McEliece dada por Robert J. McEliece [McE78], basado en códigos binarios de Goppa en el año 1978, se describe de la siguiente manera. Los valores de $\mathbf{n}$, $\mathbf{k}$ y $\mathbf{t}$ son parámetros disponibles públicamente, pero L, g, P y S son secretos generados al azar. Entonces, este criptosistema de clave pública funciona con los siguientes pasos:

Paso 1: En primer lugar, Alice genera un par de claves públicas y privadas en función de los valores disponibles públicamente. Durante esto:

1. Alice selecciona un $[\mathbf{n}, \mathbf{k}]$ −código binario de Goppa, con su matriz generatriz G de dimensión $\mathbf{k} \times \mathbf{n}$, capaz de corregir $\mathbf{t}$ errores.
2. Después, selecciona una matriz S binaria, no singular, de dimensión $\mathbf{k} \times \mathbf{k}$ y una matriz permutación[3] P de dimensión $\mathbf{n} \times \mathbf{n}$.
3. Hace el cálculo de la matriz $\mathbf{k} \times \mathbf{n}$, $\hat{\mathbf{G}} = S \cdot G \cdot P$.
4. Publica su clave pública: $\left(\hat{\mathbf{G}}, \mathbf{t}\right)$.
5. Mantiene privada la clave: (S, G, P).

Paso 2: Supongamos que Bob tiene que enviar un mensaje cifrado a Alice:

1. Bob tiene un mensaje binario de texto plano m de longitud k.
2. Carga la clave pública de Alice: $\left(\hat{\mathbf{G}}, \mathbf{t}\right)$.
3. Genera un vector aleatorio z de $n-$bits con peso $w(z) = t$.
4. Bob calcula el texto cifrado $c = m \cdot \hat{\mathbf{G}} + z$ y lo envía a Alice.

Paso 3: Supongamos que Alice ha recibido el texto cifrado c. Ella descifra el texto cifrado recibido como:

1. Alice calcula P^{-1} usando su clave privada.
2. Multiplicamos el texto recibido por P^{-1}:

$$c \cdot P^{-1} = m \cdot S \cdot G + z \cdot P^{-1}$$

3. Finalmente, utiliza el algoritmo de descodificación (algoritmo de Patterson) de los códigos Goppa para determinar el valor de m.

[3] Una matriz de permutación es la matriz cuadrada con todos sus elementos $n \times n$ iguales a 0, excepto uno cualquiera por cada fila y columna, el cual debe ser igual a 1.

3.3.1 Ataque de reenvío de mensajes o mensajes relacionados

Hay varios tipos de ataques conocidos a la descodificación de los conjuntos de información. Nosotros vamos a desarrollar el ataque de reenvío de mensajes.

Supongamos que el remitente cifró un mensaje m dos veces y se generan dos textos cifrados:

$$\begin{cases} c_1 &= m \cdot S \cdot G \cdot P + e_1 \\ c_2 &= m \cdot S \cdot G \cdot P + e_2 \end{cases},$$

donde $e_1 \neq e_2$. Esto se llama condición de reenvío de mensajes. En este caso, es fácil para el criptoanalista recuperar m del sistema anterior. Como el mismo mensaje se cifra dos veces, decimos que la profundidad de reenvío es 2 en este caso. Sea $c_j(i)$ la $i-$ésima coordenada de c_j, entonces definimos

$$L_0 \coloneqq \{i \in \{1, 2, \ldots, n\} \,:\, c_1(i) + c_2(i) = e_1(i) + e_2(i) = 0\},$$

$$L_1 \coloneqq \{i \in \{1, 2, \ldots, n\} \,:\, c_1(i) + c_2(i) = e_1(i) + e_2(i) = 1\}.$$

- Si $l \in L_0$, entonces o bien $e_1(l) = 0 = e_2(l)$ o bien $e_1(l) = 1 = e_2(l)$. Asumiendo que la elección de ambos vectores es independiente, tenemos la probabilidad:

  $$P\left(e_1(l) = 1 = e_2(l)\right) = \left(\frac{t}{n}\right)^2.$$

  Para el caso de los parámetros originales de McEliece, tenemos:

  $$\left(\frac{50}{1024}\right)^2 \approx 0{,}0024.$$

  Por tanto, cuando consideramos $l \in L_0$, el caso más significativo es cuando $e_1(l) = 0 = e_2(l)$; equivalentemente, ni $c_1(l)$ ni $c_2(l)$ tienen errores.
- Si $l \in L_1$, equivale a que o bien $c_1(l)$ o bien $c_2(l)$ contiene algún error.

Ahora, en esta variante, nuestro objetivo es aproximar la probabilidad de adivinar k columnas independientes de las indexadas por L_0. Sea p_m la probabilidad de que e_1 y e_2 tengan exactamente m coordenadas no nulas en común. Entonces:

$$p_m = P\left(|\{i \,:\, e_1(i) = 1\} \cap \{i \,:\, e_2(i) = 1\}| = m\right) = \frac{\binom{t}{m}\binom{n-k}{t-m}}{\binom{n}{t}}.$$

Por tanto, el cardinal esperado de L_1 es:

$$E\left(|L_1|\right) = \sum_{m=0}^{t} (2t - 2m) p_m,$$

ya que para cada i que $e_1(i) = 1 = e_2(i)$ reduce $|L_1|$ en dos.

Para los parámetros originales del criptosistema de McEliece, este es aproximadamente $95{,}1$.

Por ejemplo, supongamos $|L_1| = 94$. Entonces $|L_0| = 1024 - 94 = 930$, de los cuales $|L_0| \times 0{,}0024 \approx 3$ son erróneos. Tenemos la probabilidad de adivinar entre 5413 columnas no separadas de aquellas indexadas por L_0, es

$$\frac{\binom{927}{524}}{\binom{930}{524}} \approx 0{,}0828.$$

Por tanto, el criptoanalista espera el éxito con solo 12 intentos, con un coste de $12 \times 524^3 \approx 10^{10}$ operaciones. Es un gasto muy bajo en comparación con otros ataques conocidos y compromete la seguridad del sistema.

La existencia de este y otros métodos de ataque provoca que haya que aumentar los tamaños de las claves para garantizar la seguridad. Bernstein, Lange y Peters [BLP08] dan las longitudes de código recomendadas y su correspondiente seguridad, como se describe a continuación:

Longitud n del código	Peso t del vector error	Seguridad (en bits)
512	21	33,0
1024	38	57,9
2048	69	103,5
4096	127	187,9
8192	234	344,6
16384	434	637,4

3.3.2 Asignación de claves

- La matriz $\hat{\mathbf{G}}$ de tamaño $\mathbf{k} \times \mathbf{n}$. (Pública)
- Las matrices S de tamaño $\mathbf{k} \times \mathbf{k}$ y P de tamaño $\mathbf{n} \times \mathbf{n}$. (Privadas)
- El polinomio de Goppa $g(X)$ sobre $\mathbb{F}_{2^m}$ de grado t. (Privado)
- El conjunto $L = \{\alpha_1, \alpha_2, \ldots, \alpha_n\} \subseteq \mathbb{F}_{2^m}$. (Privado)

Los parámetros dados en la construcción original eran:

$$\begin{array}{lcl} \mathbf{n} & : & 1024 = 2^{10} \\ \mathbf{t} & : & 50 \\ m & : & 10 \\ \mathbf{k} & : & n - mt = 524 \end{array}$$

Para estos parámetros, el tamaño de la clave pública es:

$$kn = 524 \times 1024 = 536576 \text{ bits} \approx 66 \text{ KB}$$

Este sería el peso del archivo que contiene la matriz de clave pública $\hat{G}$.

El tamaño de la clave privada es:

$$\left(k^2 + n^2\right) + (t \times n) + (n \times m) = (274576 + 1048576) + 500 + 10240 = 1333892 \text{ bits} \approx 162{,}8 \text{ KB}$$

Donde contamos el peso del producto $S \cdot P$, la evaluación del polinomio $g(X)$ en las todas las $\alpha_i \in L$ y el tamaño del conjunto L.

Estos cálculos evidencian que la seguridad del criptosistema reside en que las claves tienen tamaños razonablemente grandes y no son abarcables en tiempo polinomial.

Como el tamaño de la clave en este esquema es muy grande, Niederreiter propuso una variante del criptosistema McEliece, en el que se utilizan códigos duales de Reed-Solomon generalizados (GRS). Esto escapa al objetivo del trabajo, así que no lo comentaremos.

3.3.3 McEliece clásico: criptografía conservadora basada en códigos

Empecemos por asentar un par de conceptos necesarios para comprender el funcionamiento del McEliece clásico.

- Un esquema de cifrado de clave pública (PKE, en inglés) es un esquema con claves públicas y privadas, en el que podemos cifrar un mensaje utilizando la clave pública y descifrar utilizando la clave privada.
- Un método de encapsulación de claves (KEM, en inglés) es un esquema con claves públicas y privadas, donde podemos usar la clave pública para crear un texto cifrado (encapsulación) que contenga una clave simétrica elegida al azar. Podemos descifrar el texto cifrado usando la clave privada.

El McEliece clásico es un mecanismo de encapsulación de claves, que establece una clave simétrica para dos usuarios finales. Este KEM, descrito por un equipo de investigación (se puede consultar la definición del criptosistema en el documento de octubre de 2022) también es candidato en la cuarta ronda (a mayo de 2023) para la competición del NIST de estandarización global del sistema criptográfico post-cuántico. Está diseñado para proporcionar seguridad a un nivel muy alto, incluso contra ordenadores cuánticos. La descodificación del conjunto de información es la estrategia de ataque más efectiva que se conoce. Sin embargo, a pesar de este y otros algoritmos relacionados, los ataques de recuperación de claves son mucho más lentos que la descodificación del conjunto de información.

La fuerza está contigo, joven Skywalker, pero aún no eres un Jedi.

Darth Vader

A | Algoritmo de Euclides en $\mathbb{F}_q[X]$

Este apéndice ha sido extraído íntegramente de [MT97, Cap. 9 §4].

Se ha visto a lo largo de cualquier curso de estructuras algebraicas que el anillo de polinomios $\mathbb{F}_q[X]$ y $\mathbb{Z}$ tienen ciertas propiedades en común, la que nos concierne en este apéndice es la existencia de una división euclídea o con resto:

Dados $f_0(X), f_1(X) \in \mathbb{F}_q[X]$ con $\deg f_0(X) \geq \deg f_1(X)$, existen polinomios $q(X)$, $r(X)$ tales que $\deg r(X) < \deg f_1(X)$ y

$$f_0(X) = f_1(X)q(X) + r(X).$$

Otra importante consecuencia es el algoritmo de Euclides para el cálculo del máximo común divisor de dos polinomios. El proceso es del todo análogo al bien conocido de $\mathbb{Z}$. No obstante, como será utilizado en la descodificación de los códigos BCH vamos a estudiarlo con cierto detalle. Antes de comenzar, recordemos que la noción de mcd está definida salvo producto por unidades del anillo, que en nuestro caso son los elementos no nulos de $\mathbb{F}_q$.

Dados $f_0(X), f_1(X) \in \mathbb{F}_q[X]$ con $\deg f_0(X) \geq \deg f_1(X)$, para determinar $m(X) = \text{mcd}\,(f_0(X), f_1(X))$, realizamos sucesivamente las divisiones

$$\begin{aligned} f_0(X) &= f_1(X)q_1(X) + f_2(X) \\ f_1(X) &= f_2(X)q_2(X) + f_3(X) \\ f_2(X) &= f_3(X)q_3(X) + f_4(X) \\ &\vdots \end{aligned}$$

Como $\deg f_1(X) > \deg f_2(X) > \cdots$, al cabo de un número finito de pasos obtendremos un resto $f_k(X) = 0$.

Proposición A.1. *Con las notaciones anteriores, si $f_k(X) = 0$, entonces*

$$mcd\,(f_0(X), f_1(X)) = f_{k-1}(X).$$

Demostración. Si $f_k(X) = 0$, entonces $f_{k-1}(X)|f_{k-2}(X)$, e iteradamente

$$f_{k-1}(X)|f_{k-3}(X), \ldots, f_{k-1}(X)|f_1(X),\ f_{k-1}(X)|f_0(X);$$

en consecuencia,

$$f_{k-1}(X)|\text{mcd}\,(f_0(X), f_1(X)) = m(X).$$

Recíprocamente, como $m(X)|f_0(X)$ y $m(X)|f_1(X)$, se verifica que

$$m(X)|f_2(X) = f_0(X) - f_1(X)q_1(X),$$

e iteradamente

$$m(X)|f_3(X), \ldots, m(X)|f_{k-1}(X).$$

De estos dos resultados $m(X)|f_{k-1}(X)$ y $f_{k-1}(X)|m(X)$, y por ser $\mathbb{F}_q[X]$ un DFU, deducimos que salvo producto por una unidad de $\mathbb{F}_q$, $m(X) = f_{k-1}(X)$. ∎

Es conocido que $m(X) = \text{mcd}\,(f_0(X), f_1(X))$ puede escribirse como combinación lineal de $f_0(X),\ f_1(X)$ con coeficientes en $\mathbb{F}_q[X]$, no es más que la identidad de Bézout aplicada a polinomios. Una ligera modificación del algoritmo de Euclides permite determinar estos coeficientes. En concreto, se definen los polinomios $u_i(X),\ v_i(X)$ mediante

$$\begin{array}{ll} u_0(X) = 0 & u_1(X) = 1; \\ v_0(X) = 1 & v_1(X) = 0; \end{array}$$

e, iteradamente, para $1 < i < k$

$$\begin{array}{rcl} u_{i+1}(X) & = & u_i(X)q_i(X) + u_{i-1}(X); \\ v_{i+1}(X) & = & v_i(X)q_i(X) + v_{i-1}(X). \end{array}$$

Proposición A.2. *Con las notaciones anteriores, para cada $i = 0, \ldots, k-1$, se verifica que*

$$f_i(X) = (-1)^i\,(f_0(X)v_i(X) - f_1(X)u_i(X))\,.$$

Demostración. Simplemente por inducción sobre i. ∎

Para terminar, vamos a establecer otras dos propiedades de estos polinomios $u_i(X)$, $v_i(X)$. Estas propiedades serán utilizadas en el capítulo de códigos BCH.

Proposición A.3. *Los polinomios $u_i(X)$, $v_i(X)$ verifican que,*

1. *si $i \geq 1$, entonces $\deg u_i(X) = \deg f_0(X) - \deg f_{i-1}(X)$; y*
2. *$u_i(X)v_{i+1}(X) - v_i(X)u_{i+1}(X) = (-1)^{i+1}$.*

Demostración. Las dos demostraciones son fáciles y se realizan por inducción. Como ejemplo vamos a realizar la primera. El resultado es evidentemente cierto para $i = 1$. Supongamos que se cumple para un cierto $i = i_0 \in \mathbb{N}$

$$\deg u_{i_0}(X) = \deg f_0(X) - \deg f_{i_0-1}(X),$$

probarlo para $i_0 + 1$ es equivalente a probar que

$$\deg u_{i_0+1}(X) - \deg u_{i_0}(X) = \deg f_{i_0-1}(X) - \deg f_{i_0}(X).$$

Como el grado del término de la derecha coincide con $\deg q_{i_0}(X)$, el resultado se deduce de la definición de $u_{i_0+1}(X)$.

Bibliografía

[AFS05] D. Augot, M. Finiasz, and N. Sendrier, *A family of fast syndrome based cryptographic hash functions*, Progress in Cryptology – Mycrypt 2005, Springer Berlin Heidelberg, 2005, pp. 64–83.

[Bar97] A. M. Barg, *Complexity issues in coding theory*, Handbook of Coding Theory (V. Pless, W.C. Huffman, and R.A. Brualdi, eds.), vol. I, Elsevier Science, 1997, pp. 649 – 754.

[Ber73] E. R. Berlekamp, *Goppa codes*, IEEE Transactions on Information Theory **19** (1973), no. 5, 590–592.

[Ber74] E. R. Berlekamp, *Key papers in the development of coding theory*, IEEE Press - The Institute of Electrical and Electronics Engineers, Inc., 1974.

[BJ74] E. R. Berlekamp and J. Justesen, *Some long cyclic linear binary codes are not so bad*, IEEE Transactions on Information Theory **20** (1974), no. 3, 351–356.

[BLP08] D. J. Bernstein, T Lange, and C. Peterson, *Attacking and defending the McEliece cryptosystem*, Post-Quantum Cryptography, Springer Berlin Heidelberg, 2008, pp. 31–46.

[BMvT78] E. R. Berlekamp, R. McEliece, and H. van Tilborg, *On the inherent intractability of certain coding problems (corresp.)*, IEEE Transactions on Information Theory **24** (1978), no. 3, 384–386.

[BRC60] R.C. Bose and D.K. Ray-Chaudhuri, *On a class of error correcting binary group codes*, Information and Control **3** (1960), no. 1, 68–79.

[CC98] A. Canteaut and F. Chabaud, *A new algorithm for finding minimum-weight words in a linear code: application to McEliece's cryptosystem and to narrow-sense BCH codes of length 511*, IEEE Transactions on Information Theory **44** (1998), no. 1, 367–378.

[Chi64] R. Chien, *Cyclic decoding procedures for Bose-Chaudhuri-Hocquenghem codes*, IEEE Transactions on Information Theory **10** (1964), no. 4, 357–363.

[FGUO+13] J.-C. Faugere, V. Gauthier-Umana, A. Otmani, L. Perret, and J.-P. Tillich, *A distinguisher for high-rate McEliece cryptosystems*, IEEE Transactions on Information Theory **59** (2013), no. 10, 6830–6844.

[Gol49] M. J. E. Golay, *Notes on digital coding (correspondence)*, Proceedings of the IRE **37** (1949), no. 6, 657–657.

[Gop70] V. D. Goppa, *A new class of linear error-correcting codes*, Problemy Peredachi Informatsii **6** (1970), no. 3, 24–30.

[Gop71] _____, *Rationale Darstellung von Codes und (L,g)-Codes*, Problemy Peredachi Informatsii **7** (1971), no. 3, 41–49.

[Ham50] R. W. Hamming, *Error detecting and error correcting codes*, The Bell System Technical Journal **29** (1950), no. 2, 147–160.

[Hel74] H.J. Helgert, *Alternant codes*, Information and Control **26** (1974), no. 4, 369–380.

[Hil86] R. Hill, *A first course in coding theory*, Oxford University Press, 1986.

[Jus72] J. Justesen, *Class of constructive asymptotically good algebraic codes*, IEEE Transactions on Information Theory **18** (1972), no. 5, 652–656.

[LB88] P. J. Lee and E. F. Brickell, *An observation on the security of McEliece's public-key cryptosystem*, Lecture Notes in Computer Science, Springer Berlin Heidelberg, 1988, pp. 275–280.

[Leo88] J.S. Leon, *A probabilistic algorithm for computing minimum weights of large error-correcting codes*, IEEE Transactions on Information Theory **34** (1988), no. 5, 1354–1359.

[LW67] Shu Lin and E.J. Weldon, *Long BCH codes are bad*, Information and Control **11** (1967), no. 4, 445–451.

[MA18] A. Muñoz Azaustre, *Teoría de códigos*, Universidad de Málaga, 2018.

[McE78] R. J. McEliece, *A public-key cryptosystem based on algebraic coding theory*, Deep Space Network Progress Report **44** (1978), 114–116.

[McE04] R. J. McEliece, *The theory of information and coding*, Cambridge University Press, 2004.

[MP15] A. Martínez Peral, *Técnicas algebraicas en códigos correctores de errores.*, Universidad de Murcia, 2015.

[MS77] F. J. MacWilliams and N. J. A. Sloane, *The theory of error-correcting codes (north-holland mathematical library, vol. 16)*, North Holland Publishing Company, 1977.

[MT97] J. Munuera and J. Tena, *Codificación de la información*, Series de Manuales y Textos Universitarios, Universidad de Valladolid, 1997.

[Mul54] D. E. Muller, *Application of Boolean Algebra to Switching Circuit Design and to Error Detection*, Transactions of the I.R.E. Professional Group on Electronic Computers **EC-3** (1954), no. 3, 6–12.

[Pat75] N. Patterson, *The algebraic decoding of goppa codes*, IEEE Transactions on Information Theory **21** (1975), no. 2, 203–207.

[PHB98] V. Pless, W.C. Huffman, and R.A. Brualdi, *Handbook of coding theory*, Handbook of Coding Theory, no. v. 1, Elsevier Science, 1998.

[Pra62] E. Prange, *The use of information sets in decoding cyclic codes*, IEEE Transactions on Information Theory **8** (1962), no. 5, 5–9.

[Ree54] I. S. Reed, *A Class of Multiple-Error-Correcting Codes and the Decoding Scheme*, Transactions of the IRE Professional Group on Information Theory **4** (1954), no. 4, 38–49.

[RF03] J. Ryan and P. Fitzpatrick, *Counting irreducible Goppa codes*, Workshop on Coding and Cryptography, 03 2003.

[Sha48a] C. E. Shannon, *A Mathematical Theory of Communication*, Bell System Technical Journal **27** (1948), no. 3, 379–423.

[Sha48b] _____, *A Mathematical Theory of Communication*, Bell System Technical Journal **27** (1948), no. 4, 623–656.

[Sha49] C. E. Shannon, *Communication theory of secrecy systems*, Bell System Technical Journal **28** (1949), no. 4, 656–715.

[Sin19] H. Singh, *Code based Cryptography: Classic McEliece*, arXiv:1907.12754 [cs.CR], 2019.

[SKHN75] Y. Sugiyama, M. Kasahara, S. Hirasawa, and T. Namekawa, *A Method for Solving Key Equation for Decoding Goppa Codes*, Information and Control **27** (1975), no. 1, 87–99.

[Sle56] D. Slepian, *A Class of Binary Signaling Alphabets*, Bell System Technical Journal **35** (1956), no. 1, 203–234.

[SS92] V. M. Sidelnikov and S. O. Shestakov, *On insecurity of cryptosystems based*

on generalized reed-solomon codes, Discrete Mathematics and Applications **2** (1992), no. 4.

[Ste89] J. Stern, *A method for finding codewords of small weight*, Coding Theory and Applications, Springer-Verlag, 1989, pp. 106–113.