Códigos cíclicos

Cruz Enrique Borges Hernández

12 de marzo de 2005

Índice

1. Motivación.

Durante los primeros años del siglo XX comienza el estudio de la transmisión de la información. Este campo estudia la resolución de tres problemas básico:

- Los códigos óptimos, que pretenden transmitir la mayor cantidad de información ocupando el menos espacio.
- Los códigos correctores, que buscan evitar la pérdida de información por problemas en la transmisión.
- Los códigos criptográficos, que intentan que la información que transmiten sólo sea leída por su destinatario.

El presente trabajo versa sobre los segundos, los códigos correctores de errores. La primera idea de código corrector puede ser transmitir varias veces el mismo mensaje esperando que alguno de ellos llegue correctamente a su destino. Esta idea es válida, pero entra dentro, casi, de las ideas probabilísticas (rozando incluso la fe). Además, es muy costosa, el mensaje aumenta excesivamente su tamaño con información *redundante*.

Los primeros códigos un poco más elaborados son los *códigos en bloque*. El método es bien simple, se basa en "bloques" de información que se transforman en otros "bloques" mediante una aplicación llamada *diccionario*.

Por ejemplo:

Diccionario:

0	0000
1	1001
2	1010
3	0011
4	1100
5	0101
6	0110
7	1111

Mensaje		Código
35724	$\Rightarrow$	00110101111110101100

El coste computacional de la de/codificación es excesivo. Se debe guardar el diccionario con el que se codifica la información, el cual tiende a ser relativamente grande (en este caso es mayor que el mensaje a codificar).

Surge entonces la necesidad de introducir alguna estructura algebraica en los códigos que nos permita "simplificar" los procesos de de/codificación. De aquí nacen los *códigos lineales* y dentro de ellos unos muy especiales, los *códigos cíclicos*.

2. Códigos lineales y códigos cíclicos.

2.1. Definiciones previas.

En esta subsección introduciremos una serie de conceptos básicos necesarios para entender el proceso de de/codificación con los códigos lineales.

Definición (Alfabeto)

Sea $\mathcal{A}$ un conjunto finito de símbolos entonces $\mathcal{A}$ es un alfabeto.

Definición (Código)

Sea $\mathcal{A}$, $\mathcal{B}$ dos alfabetos. Llamaremos código o diccionario a la aplicación inyectiva:

$$C : \bigcup_{k\in\mathbb{N}} \mathcal{A}^k \to \bigcup_{l\in\mathbb{N}} \mathcal{B}^l$$

Nota: usualmente también se denomina código a la imagen de la aplicación C.

Definición (Códigos en bloques)

$$\text{Si } C : \bigcup_{k\in\mathbb{N}} \mathcal{A}^k \to \mathcal{B}^n$$

es una aplicación inyectiva, entonces C es un código en bloque de longitud n.

Definición (Peso de Hamming)

Sea $x \in \mathcal{B}^n$ con $x = (x_1, \ldots, x_n)$ entonces:

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

es el peso de Hamming de x.

Definición (Distancia de Hamming)

Sea $x, y \in \mathcal{B}^n$ con $x = (x_1, \ldots, x_n)$, $y = (y_1, \ldots, y_n)$ entonces:

$$d(x, y) = |\{i \ / \ 1 \leq i \leq n \wedge x_i \neq y_i\}| = w(x - y)$$

es la distancia de Hamming entre x e y.

Nota 1: $w(x) = d(x, 0)$
Nota 2: Se puede comprobar w es una norma y que efectivamente d es la distancia inducida por la norma w.

Definición (Distancia mínima)

Sea c un código, llamaremos distancia mínima d del código C a:

$$d = d(C) = min\{d(x,y) \ / \ x,y \in C \ \wedge x \neq y\} = min\{w(x) \ / \ x \in C \wedge x \neq 0\}$$

Definición (Código lineal)

Sea $\mathbb{K}_q$ un cuerpo de q elementos. Llamaremos código lineal de longitud k a un subespacio vectorial C de $\mathbb{K}_q^n$ con $dim\ C = k$.

Nota: llamaremos parámetros fundamentales de un código lineal C a $[n,k,d]$ donde: $k = dim\ C$ y d es la distancia mínima de C.

Definición (Matriz generadora)

Los códigos lineales, al ser subespacios, se pueden tomar como la imagen de una aplicación lineal inyectiva. Podemos considerar entonces la matriz G asociada a dicha aplicación lineal. Definiremos dicha matriz como la matriz generadora del código.

Nota: al ser la matriz generatriz la asociada a una aplicación lineal inyectiva tendrá rango máximo.

Definición (Matriz de control)

Llamaremos matriz de control H de un código lineal a toda matriz tal que $GH^t = 0$ donde G es una matriz generatriz del código.

Definición 1 (Código cíclico)

Sea $\mathbb{K}_q$ un cuerpo de q elementos y S un código lineal de longitud n. C es un código cíclico si y solo si $(c_0, \ldots, c_n - 1) \in C$ entonces $(c_{n-1}, c_0, \ldots, c_{n-2}) \in C$.

Es usual también usar la siguiente definición de códigos cíclicos.

Definición 2 (Código cíclico)

Sea $A = \mathbb{K}_q[x] \ / < x^n - 1 >$ y consideramos el isomorfismo

$$\begin{array}{ccc} \varphi : \mathbb{K}_q^n & \longrightarrow & A \\ (c_0, \ldots, c_{n-1}) & \rightsquigarrow & c_0 + \ldots + c_{n-1}x^{n-1} \end{array}$$

$C \subset A \cong \mathbb{K}_q^n$ es un código cíclico si y solo si C es un ideal de A.

Veamos ahora que ambas definiciones son equivalentes:

"$1 \Rightarrow 2$"

Sea C un código cíclico (que mediante φ lo identificamos como un subconjunto de A). Tenemos que ver C es un ideal de A, esto es: $\forall c \in C \; \forall a \in A \Rightarrow ca \in C$. Como C es subespacio lineal tenemos que es cerrado para el producto y la multiplicación por escalares, luego nos queda por ver que $cx \in C$:

$$\begin{aligned} cx &= x(c_0 + \ldots + c_{n-1}x^{n-1}) = c_0x + \ldots + c_{n-1}x^n = \\ &= c_{n-1} + \ldots + c_{n-2}x^{n-1} + c_{n-1}(x^n - 1) = c_{n-1} + \ldots + c_{n-2}x^{n-1} \in C \end{aligned}$$

Luego C es un ideal de A.

"$2 \Rightarrow 1$"

Sea $c \in C$, con C ideal de A. Tenemos que ver que C es cíclico, esto es que:

- C es subespacio lineal.
- $\forall c = (c_0, \ldots, c_{n-1}) \in C \Rightarrow (c_{n-1}, c_0, \ldots, c_{n-2}) \in C$.

La primera parte es trivial y se deduce simplemente de las propiedades de C como ideal de A.

En cuanto a la segunda parte:

Sea $c = (c_0, \ldots, c_{n-1}) \in C \Rightarrow xc \in C$ pues $x \in A$ y C es un ideal.

Ahora bien:

$$\begin{aligned} cx &= c_0x + \ldots + c_{n-1}x^n = c_{n-1} + \ldots + c_{n-2}x^{n-1} + c_{n-1}(x^n - 1) = \\ &= c_{n-1} + \ldots + c_{n-2}x^{n-1} = (c_{n-1}, c_0, \ldots, c_{n-2}) \end{aligned}$$

Con lo cual tenemos que C es cíclico.

Definición (Polinomio generador)

Se puede demostrar que todos los ideales de A son principales, esto es:

$\forall C$ ideal de A (con longitud n como código) $\exists |\; g(x)$ mónico $\in A$ y divisor de $x^n - 1$ / $< g(x) >= C$.

A dicho polinomio $g(x)$ lo denominaremos polinomio generador del código C.

Definición (Polinomio de control)

Sea C un código cíclico de longitud n sobre $\mathbb{K}_q$. Sea $g(x)$ su polinomio generador con $\delta(g(x)) = n - k$. Llamaremos polinomio de control de C al polinomio:

$$h(x) = \frac{x^n - 1}{g(x)} = h_0 + h_1 x + \ldots + h_k x^k$$

2.2. Codificación

Veamos ahora como es el proceso de codificación de los códigos lineales y cíclicos[1].

La codificación de un código lineal es bien simple, basta con multiplicar la matriz generadora G del código por el bloque a codificar, esto es:

Sea $m \in \mathbb{K}_q^n$ el bloque a codificar y sea G una matriz generadora de nuestro código C entonces el bloque codificado será $m^t G$.

Observamos que dar un código lineal en realidad no es más que dar una matriz generatriz G y que el proceso de codificación de un bloque exige la realización de nk sumas y productos (k es la dimensión del código).

En cuanto a la codificación de un código cíclico el proceso es exactamente el mismo, solo varia la forma de la matriz generatriz G que en este caso es tiene una forma particular. Veamos a continuación como es dicha forma.

Proposición

Sea C un código cíclico de longitud n sobre $\mathbb{K}_q$. Sea $g(x)$ su polinomio generador con $\delta(g(x)) = n - k$.

Entonces $dim(C) = k$ y una base de C es $\{g(x), xg(x), \ldots, x^{k-1}g(x)\}$

Demostración

Veamos que efectivamente $\{g(x), xg(x), \ldots, x^{k-1}g(x)\}$ es base (con lo cual obtenemos inmediatamente que $dim(C) = k$).

Sea $h(x) \in C$ tenemos entonces que $h(x) = f(x)g(x)$ con $\delta(f(x)) \leq k$. Si $f(x) = a_0 + \ldots + a_{k-1}x^{k-1}$ tenemos que $f(x)g(x) = a_0 g(x) + \ldots + a_{k-1}g(x)$ con lo cual obtenemos que $\{g(x), xg(x), \ldots, x^{k-1}g(x)\}$ es un sistema generador.

Veamos ahora que es libre.

[1] Más adelante analizaremos el proceso de decodificación.

Sea $b_0g(x)+\ldots+b_{n-1}x^{n-1}g(x)=0 \Rightarrow b(x)g(x)=0$ con $b(x)=b_0+\ldots+b_{n-1}x^{n-1}$. Como $g(x)\neq 0$ pues no es un código trivial y como $C\subset A$ que es dominio de integridad tenemos entonces que $b=0$ con lo cual $b_i=0\ \forall i$.

Obtenemos así que que es un sistema libre y por lo tanto base.

Como consecuencia directa obtenemos que una matriz generadora del código tendrá la siguiente forma:

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

donde g_i son los coeficientes del polinomio generatriz.

Obtenemos así la matriz generadora de un código cíclico. Observando dicha matriz vemos que el número de operaciones a realizar para codificar un bloque se ha reducido a k^2 sumas y productos.

Veamos ahora otra forma de generar los códigos cíclicos. Sabemos que si C es un código cíclico entonces $\exists\ g(x)$ mónico y divisor de x^n-1 tal que $C=<g(x)>$. Supongamos entonces que $x^n-1=f_1f_2\ldots f_k$ con $\delta(f_i)=1\ \wedge\ f(\alpha_i)=0$ entonces, es claro, que $g(x)=f_{i_1}\ldots f_{i_r}$. Luego $C=\{c(x)\in A\ /\ c(\alpha_i)=0\ \forall i\}$

A partir de esta idea podríamos considerar construir un código eligiendo las raíces de x^n-1 en vez del polinomio $g(x)$ lo cual suele resultar más fácil y da como resultado la aparición de los códigos **BCH** que nombraremos más adelante y que son los más usados actualmente.

2.3. Decodificación.

Vamos ahora a analizar el proceso de decodificación de códigos lineales y cíclicos.

Para decodificar un código lineal usaremos el llamado algoritmo del líder. Básicamente lo que hace es decodificar la palabra por la palabra más cercana (en sentido de Hamming) a ella en el código, pero para ello nos valdremos de las propiedades que tienen los códigos por tener la estructura algebraica que poseen.

El proceso de decodificación tiene tres pasos: detección de errores, corrección de errores y decodificación. Para resolver la primera parte nos apoyaremos en el siguiente concepto:

Definición (Síndrome)

Sea $y \in \mathbb{K}_q^n$ y sea C un código lineal de dimension k con matriz de control H. Llamaremos síndrome de y al vector $S(y) = Hy^t \in \mathbb{K}_p^{n-k}$

Nota: tenemos que observar aquí una pequeña diferencia entre el cálculo del síndrome de un código lineal y otro cíclico. Mientras con el código lineal desconocemos completamente como es la matriz H y debemos hallarla "a mano", en los códigos cíclicos presenta una estructura determinada y conocida a priori. Dicha matriz posee la forma siguiente:

$$H = \begin{pmatrix} 0 & \dots & 0 & h_k & \dots & h_1 & h_0 \\ 0 & \dots & h_k & h_{k-1} & \dots & h_0 & 0 \\ \vdots & \iddots & \vdots & \vdots & \iddots & \vdots & \vdots \\ h_k & \dots & h_1 & h_0 & \dots & 0 & 0 \end{pmatrix}$$

donde h_i son los coeficientes de un polinomio de control.

Veamos pues que es en realidad una matriz de control:

$GH^t = (i, j)$ donde cada elemento (i, j) resulta ser el coeficiente $x^{n-i-j+1}$ del polinomio $g(x)h(x) = x^n - 1$. Con lo cual obtenemos el vector nulo y por lo tanto obtenemos el resultado buscado.

Veamos el uso que le podemos dar a esta aplicación lineal:

Sea $x \in C$ el bloque enviado y supongamos que han habido errores en la transmisión. Recibimos entonces el vector $y = x + e$. El síndrome de y será: $S(y) = S(x) + S(e) = S(e)$ pues es claro que una palabra pertenece al código si y solo si su síndrome es cero. De esta forma resolvemos el primer paso eficientemente, pues vasta con hacer $(n - k)n$ sumas y productos.

Veamos ahora como resolver la siguiente parte. Para ello debemos vamos a clasificar los elementos del conjunto de llegada por su síndrome mediante la siguiente relación de equivalencia $xRy \Leftrightarrow S(x) = S(y)$. Es trivial que dicha relación es efectivamente de equivalencia con lo obtenemos una partición del código. Vamos a intentar tomar un elemento representante de cada clase.

Definición (Líder)

Llamaremos líder de una clase al único elemento de peso mínimo, si existe, en sentido de Hamming.

El siguiente teorema nos asegura en qué casos la clase de un elemento posee líder, con lo que, como veremos más adelante, el código es capaz de corregir errores.

Proposición

Cada clase posee posee a lo supo un elemento de peso mínimo menor o igual a t, donde t es la capacidad correctora del código.

Demostración

Supongamos que existe u, v elementos de la misma clase y ambos con peso menor o igual a t. Tenemos entonces:

$$S(u) = S(v) \Rightarrow S(u - v) = 0 \Rightarrow u - v \in C$$
$$w(u - v) \leq w(u) + w(v) \leq 2t < d(C) \Rightarrow u - v = 0 \Rightarrow u = v$$

Este teorema nos asegura cuando una clase tiene líder.

Veamos como aplicar este teorema a la corrección de errores. Hemos detectado que y posee errores (mediante el cálculo de su síndrome) y buscamos $x \in C$ tal que $d(x, y) = w(x - y)$ sea mínima. Como todos los vectores $x - y, x \in C$ pertenecen a la misma clase tenemos que $\text{mín}_{x \in C}\{w(y - x)\}$ se da cuando $x - y$ es el líder de la clase y por el teorema anterior sabemos que si se han cometido menos de t errores entonces, asumiendo que el error cometido es el líder de la clase, podemos corregir el error simplemente restando el líder e al bloque recibido y.

Para proceder mediante este método es necesario obtener la tabla de síndrome/líder que tendrá q^{n-k} filas y dos columnas: en una irá el síndrome de cada clase y en la otra el líder de esta si existiese.

El problema de este paso es la tabla que hay que guardar, pues para códigos de uso cotidiano, n suele ser un número bastante grande. Sin embargo el coste operacional es una búsqueda y una resta, luego es constante.

Nos falta simplemente decodificar y, el bloque recibido. Para ello simplemente nos basta con resolver el siguiente sistema lineal. Como sabemos que y es una palabra del código, el sistema $y = xG$ tendrá solución única x que será el bloque original sin codificar.

En el caso de los códigos cíclicos el proceso es el mismo, sólo varía en la forma de corregir el error debido a las particularidades de estos códigos, pues no va a ser necesaria guardar la tabla completa de síndrome/líder, aunque nos vamos a ver forzados a realizar alguna que otra operación más. Aparte de esto la matriz H de control posee la forma particular vista con anterioridad.

Nos aprovecharemos de la siguiente idea: si tenemos que el código es cíclico, basta con saber corregir los errores en una posición (usualmente la última) y realizar luego permutaciones del bloque para ir corrigiendo las distintas posiciones. Con lo cual ahora la tabla de síndrome/líder sólo contiene aquellos líderes en los que la última coordenada sea no nula (esto es, errónea). Recibido y calculamos su síndrome y si esta en la tabla corregimos el error en la última posición, realizamos una permutación a los elementos de y y repetimos el proceso hasta comprobar todas las posiciones de y.

Con este proceso hemos reducido la tabla síndrome/líder a costa de realizar n cálculos de síndrome más[2].

La ventaja clara entre los códigos cíclicos y los lineales esta en que la tabla a guardar en este ultimo caso excede las posibilidades razonables de almacenamiento, mientras que con los códigos cíclicos la cosa es mucho menos abultada.

2.4. Ejemplos

Veamos ahora un ejemplo de codificación de de un código lineal y de otro cíclico. Comencemos por el lineal:

Queremos codificar la palabra `HOLA` y enviarla a través de un canal con ruido del que sabemos que comete un error en cada bloque enviado. Para ello primero, mediante una biyección transformamos las letras en un código binario:

Diccionario $:= \varphi$

A	0000	E	0100	I	1000	M	1100
B	0001	F	0101	J	1001	N	1101
C	0010	G	0110	K	1010	Ñ	1110
D	0011	H	0111	L	1011	O	1111

Luego aplicamos un código lineal generado por la matriz G y con matriz de control H:

$$G = \begin{pmatrix} 1\,1\,0\,1\,0\,0\,0 \\ 1\,0\,1\,0\,1\,0\,0 \\ 0\,1\,1\,0\,0\,1\,0 \\ 1\,1\,1\,0\,0\,0\,1 \end{pmatrix} \qquad H = \begin{pmatrix} 1\,0\,0\,1\,1\,0\,1 \\ 0\,1\,0\,1\,0\,1\,1 \\ 0\,0\,1\,0\,1\,1\,1 \end{pmatrix}$$

[2]Existen métodos aún más eficaces que nos permiten corregir el error mediante el cálculo de un sólo síndrome.

Hasta aquí lo único que hemos hecho ha sido dar G más o menos de forma arbitraria y calculando su *nulidad* hemos sacado una matriz H de control. Ahora debemos comprobar que el código que hemos generado es capaz de corregir los errores producidos por el canal, para ellos hallamos la distancia entre todas las palabras del código. Obtenemos que $d = 3$ por lo que sabemos que es capaz de detectar dos errores y corregir al menos uno[3].

Calculamos la tabla síndrome/líder de nuestro código:

Síndrome	Líder
000	0000000
100	1000000
010	0100000
001	0010000
110	0001000
101	0000100
011	0000010
111	0000001

Procedemos a realizar la codificación. Tenemos entonces:

$$\texttt{HOLA} \xrightarrow{\varphi} 0111\ 1111\ 1011\ 0000 \xrightarrow{C} 0010111\ 1111111\ 0101011\ 0000000$$

Enviamos el código a través del canal y recibimos:

$$0011111\ 1111111\ 1101111\ 0000000$$

Procedemos por el algoritmo del síndrome/líder. Calculamos el síndrome de cada bloque recibido[4]:

a) $s(0011111) = 110 \rightarrow$ Líder $= 0001000 \rightarrow$ decodificamos $0011111 - 0001000 = 0010111 = 0111 = \texttt{H}$

[3]Tenemos un teorema que afirma:

Teorema

Sea C un código en bloque de longitud n y distancia mínima d. Entonces el código es capaz de: a) Detectar $t < d$ errores. b) Corregir t errores si $2t < d$.

[4]Aquí estamos haciendo una pequeña trampa, este código es *sistemático*, esto es, en la matriz generatriz G hay una identidad. Por lo que el proceso de decodificación es instantáneo, una vez corregido el error, el bloque decodificado serán las posiciones en las que se encuentra la identidad en la matriz G. En este caso son las 4 últimas columnas

b) $s(1111111) = 000 \rightarrow$ Líder $= 0000000 \rightarrow$ decodificamos $1111111 - 0000000 = 1111111 = 1111 =$ `O`

c) $s(1101111) = 001 \rightarrow$ Líder $= 0010000 \rightarrow$ decodificamos $1101111 - 0010000 = 1111111 = 1111 =$ `O`

d) $s(0000000) = 000 \rightarrow$ Líder $= 0000000 \rightarrow$ decodificamos $0000000 - 0000000 = 0000000 = 0000 =$ `A`

Obtenemos entonces `HOOA` como palabra decodificada.

Observamos como el algoritmo es capaz de corregir el error producido en a) detecta que se ha producido un error en C) pero lo decodifica mal debido a que se han producido dos errores (recordar que nuestro código C corrige un error y detecta dos).

Veamos ahora el mismo ejemplo pero con un código cíclico. Usaremos la misma transformación φ dada anteriormente.

Factorizamos x^7-1 mod 2 y obtenemos que $x^7-1 = (x+1)(x^3+x+1)(x^3+x^2+1)$ Tomamos $g(x) = x^3+x^2+1$ como polinomio generador de nuestro código. Obtenemos entonces:

$$G = \begin{pmatrix} 1\,1\,0\,1\,0\,0\,0 \\ 0\,1\,1\,0\,1\,0\,0 \\ 0\,0\,1\,1\,0\,1\,0 \\ 0\,0\,0\,1\,1\,0\,1 \end{pmatrix} \qquad H = \begin{pmatrix} 0\,1\,1\,1\,0\,0\,1 \\ 1\,1\,1\,0\,0\,1\,0 \\ 1\,0\,1\,1\,1\,0\,0 \end{pmatrix}$$

Hallamos la distancia mínima d de nuestro código y obtenemos que $d = 3$ con lo cual obtenemos que podemos detectar dos errores y corregir uno. Nuestra tabla síndrome/líder es ahora bastante simple:

Síndrome	Líder
100	0000001

Pues sabemos que nos basta conocer el error que se comete en la última posición y permutar los elementos para corregir los errores.

Procedemos ahora a realizar lo codificación como anteriormente:

$$\texttt{HOLA} \xrightarrow{\varphi} 0111\ 1111\ 1011\ 0000 \xrightarrow{C} 0100011\ 1001011\ 1111111\ 0000000$$

Enviamos el código a través del canal y recibimos:

0101011 1001011 1100011 0000000

Procedemos por el algoritmo del síndrome/líder para códigos cíclicos. Calculamos el síndrome de cada bloque recibido, corregimos, si es necesario, la ultima posición y realizamos una permutación sobre el bloque.

a) 0101011	b) 1001011	c) 1100011	d) 0000000
$s(0101011) = 101$ ✓	$s(1001011) = 000$ ✓	$s(1100011) = 011$ ✓	$s(0000000) = 000$ ✓
$s(1010101) = 001$ ✓	$s(1100101) = 000$ ✓	$s(1110001) = 110$ ✓	$s(0000000) = 000$ ✓
$s(1101010) = 010$ ✓	$s(1110010) = 000$ ✓	$s(1111000) = 111$ ✓	$s(0000000) = 000$ ✓
$s(0110101) = 100$ ♯	$s(0111001) = 000$ ✓	$s(0111100) = 101$ ✓	$s(0000000) = 000$ ✓
$s(0011010) = 000$ ✓	$s(1011100) = 000$ ✓	$s(0011110) = 001$ ✓	$s(0000000) = 000$ ✓
$s(0001101) = 000$ ✓	$s(0101110) = 000$ ✓	$s(0001111) = 010$ ✓	$s(0000000) = 000$ ✓
$s(1000110) = 000$ ✓	$s(0010111) = 000$ ✓	$s(1000111) = 100$ ♯	$s(0000000) = 000$ ✓
0100011=H	1001011=O	0100011=H	0000000=A

Obtenemos entonces HOHA como palabra decodificada.

Vemos en la tabla como el algoritmo es capaz decodificar correctamente a), b) y d) no así c) al haberse producido dos errores.

3. Conclusiones.

Una vez vista la motivación por la que surgen estos códigos y visto muy por encima la forma de de/codificar de los códigos cíclicos y lineales, vamos, por último, a analizar críticamente el proceso de creación de un código y ver si con los códigos cíclicos y lineales estudiados en este trabajo tenemos resuelto, de forma eficiente, nuestro problema.

El problema es el siguiente, queremos transmitir información a través de un canal. Sabemos de antemano que durante el proceso de transmisión se produce un cierto porcentaje de errores que debemos evitar. Para ello debemos construir un código corrector con capacidad correctora suficiente para corregir dicha tasa de errores.

El problema se presenta cuando descubrimos que no conocemos de antemano la capacidad correctora de un código hasta que no hemos calculado su distancia mínima. Ahora bien, calcular dicha distancia es computacionalmente muy costoso, por lo que sería interesante obtener un método que nos permitiera conocer, a priori, la distancia mínima que nos va a ofrecer antes de generarlo.

Esta claro que en este contexto nuestros códigos lineales y cíclicos no son la solución deseada a nuestro problema. Es por eso que surgen los códigos **BCH**, que son un tipo particular de códigos cíclicos, construidos a partir del método de selección de raíces, que nos permiten averiguar a priori la capacidad correctora de nuestro código. Con esta técnica es viable, conocida la tasa de errores a la que nos queremos ajustar, construir de forma eficiente un código tal que sea capaz de corregir dicha tasa de errores.

Referencias

[1] Munuera Gómez. Juan y Tena. Juan. *"Codificación de la información"* . Secretariado de publicaciones e intercambio científico (1997).