Universidad de Valladolid

Facultad de Ciencias

Trabajo Fin de Máster

Máster Universitario en Investigación en Matemáticas

Random Network Coding y Corrección de Errores

Autor: José Ignacio Segovia Martín

***Tutor*: Dr. Diego Ruano Benito**

Índice general

Introducción

Este trabajo se enmarca dentro del área de los códigos de red, comúnmente conocidos por su nombre en inglés: Network Coding. Un nodo intermedio, en una red de datos tradicional, reenvía datos sin modificarlos. Sin embargo, si los nodos pueden procesar sus datos entrantes y generar versiones modificadas de ellos, el rendimiento puede aumentar [1]. Este proceso se denomina Network Coding y se puede modelar con herramientas algebraicas, como puede verse en [13] y en el primer capítulo de este trabajo.

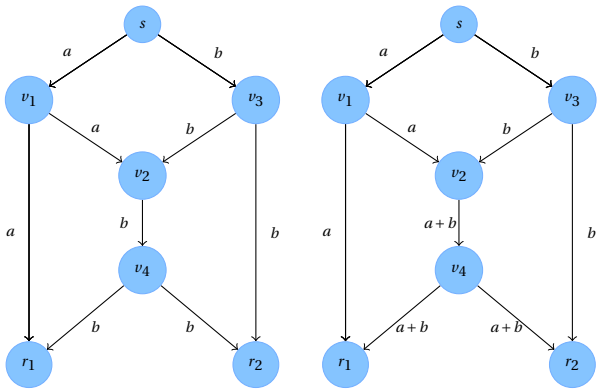

Figura 1: Red mariposa: (izquierda) sin network coding y (derecha) con network coding.

El ejemplo estándar para explicar este fenómeno es la red mariposa (ver Figura 1). El remitente s desea enviar los símbolos a y b a los receptores r_1 y r_2 a través de una red. Si enviamos la información de forma tradicional (routing), es imposible que ambos receptores obtengan ambos símbolos de información de una sola vez: el nodo v_3 tiene que elegir qué reenviar (en la Figura 1, el nodo v_3 envía b y r_2 no pueden obtener a). Sin embargo, si consideramos network coding, el nodo v_3 puede calcular $a+b$ y enviarlo. De esta forma ambos receptores pueden obtener los dos símbolos. Por ejemplo, r_2 ahora recibe $a+b$ y b, por lo que puede obtener a a partir de: $a = (a+b)-b$. En este sencillo ejemplo podemos considerar que la información es binaria y la suma es la operación XOR.

En [15], se mostró que, si los nodos intermedios emplean combinaciones lineales sobre un cuerpo finito, se puede alcanzar la capacidad máxima de transmisión de la red. La codificación de red que únicamente emplea combinaciones lineales se llama codificación de red lineal (linear network coding). La codificación de red lineal es muy útil en comunicaciones inalámbricas, pero también es útil en otras áreas, como la distribución de archivos digitales y el almacenamiento de datos distribuido.

Puede suceder que los vértices de la red estén dañados o que la comunicación entre dos nodos de la red se vea afectada por ruido". De esta forma, al transmitir símbolos por ellos, se producen errores, y debido a que los vértices de la red generan combinaciones lineales de los símbolos recibidos, estos errores se pueden transmitir por toda la red, haciendo que un único error en una arista o vértice haga la comunicación inútil. Por este motivo, nos conviene tener un código global junto con un algoritmo de decodificación, que nos permita corregir estos errores una vez transmitida la información.

En el Capítulo 1 se introducen los conceptos básicos sobre Network Coding, como el modelado de la red, mediante el grafo dirigido $G = (V, E)$, donde V es el conjunto de vértices que componen la red y E es el conjunto de aristas. A su vez, las operaciones que se realizan en los nodos las podemos modelizar a través de una matriz cuadrada F, cuyo tamaño será el número de aristas de la red, teniendo como entradas los coeficientes $f_{i,j}$ si existe un camino (i, j) en nuestro grafo, con i, j aristas. En caso contrario, la entrada de la matriz será 0. En términos de esta matriz, formulamos el polinomio de transferencia, que nos permite obtener la caracterización algebraica del éxito de la transmisión. En resumen, somos capaces de modelar el éxito en la comunicación en una red mediante un polinomio, obteniendo así una caracterización algebraica que nos dice que la comunicación será exitosa si el polinomio no se anula. Por último, estudiamos el algoritmo de Jaggi, que nos permite obtener unos coeficientes de codificación asociados a la red de forma que la comunicación sea satisfactoria. En este capítulo incluimos una serie de ejemplos sobre conceptos relacionados con las redes como el cálculo del polinomio de transferencia, junto con un ejemplo final en el que aplicamos el Algoritmo de Jaggi para la red de mariposa de la Figura 1. Para la realización de este capítulo se han tenido en cuenta, principalmente, las referencias [9], [11] y [13].

El Capítulo 2 estudiaremos lo que conocemos como Random Network Coding. Esto hace referencia a la obtención de los coeficientes de codificación asociados a la red de manera aleatoria. Esto supone que la comunicación de esta forma no siempre será exitosa, pues puede darse una combi-

nación de coeficientes de manera que no podamos llegar a una solución. La ventaja de elegir los coeficientes de codificación de manera aleatoria en lugar de obtenerlos a través del Algoritmo de Jaggi es que de esta forma es más dinámico, pues no necesitamos conocer la estructura de la red para tener una comunicación exitosa, además tampoco nos importa que la red cambie. Para este problema también estudiaremos cuál es la probabilidad de que esta elección nos lleve a una comunicación exitosa, introduciendo distintas cotas para esta probabilidad. Algunas de las cotas estudiadas en este capítulo se obtienen mediante el polinomio de tranferencia, a través de bases de Gröbner y la huella de un ideal (conjunto de monomios que no pertenecen al ideal generado por los monomios líderes). El resto de cotas tendrán un carácter más combinatorio, y están relacionadas con aspectos de la red como su sistema de flujo. Las cotas nos permiten elegir un cuerpo finito lo suficientemente grande para que la comunicación sea satisfactoria con una cierta probabilidad, pero suficientemente pequeño para que la implementación sea eficiente. A lo largo de este capítulo hemos construido varios ejemplos en los que podemos ver el cálculo de todas estas cotas. Para la realización de este capítulo se han tenido en cuenta las referencias [10] y [21].

En el Capítulo 3 estudiaremos como corregir los posibles errores producidos en la red durante la transmisión, pues un único error en una arista se puede propagar por toda la red, haciendo inservible la comunicación. En códigos clásicos, las palabras del código son vectores a los que les añadíamos una cierta redundancia, de forma que, cuando hay un cierto número de errores, podemos recuperar la palabra enviada. En nuestro caso, veremos un código para nuestra red como un conjunto de subespacios vectoriales. Ésto implica que cada palabra del código es un subespacio vectorial, de forma que enviamos un conjunto generador de este subespacio a través de la red. Si no se produjesen errores, el receptor obtendría otro sistema de generadores de este subespacio formada por combinaciones lineales de estos generadores, de forma que podría recuperar el espacio enviado. En caso de producirse errores, podriamos perder dimensión de este subespacio, recibiendo otro de dimensión menor, u obtener algún elemento que no formara parte de este subespacio, recibiendo uno de dimensión igual e incluso mayor. Estudiaremos pues las propiedades de estos códigos, junto con su decodificación. A lo largo del capítulo se han creado además diversos ejemplos que ayudan a la comprensión, junto con un ejemplo final en el que construimos todo el proceso de codificación y decodificación. Para la realización de este capítulo se ha tenido en cuenta la referencia [12].

Los principales algoritmos de este trabajo han sido programados en

Singular, en forma de procedimientos. Estos procedimientos han sido utilizados para los cálculos en los distintos ejemplos realizados en todo el trabajo. En el Capítulo 4 podemos encontrar todos estos procedimientos, junto con una explicación de su uso y funcionamiento, y un ejemplo en el que han sido aplicados.

Agradecer a mi tutor, Diego Ruano Benito por su trabajo de dirección realizado que me ha permitido finalizar el trabajo a tiempo a pesar de la situación ocurrida en este curso 2019/2020.

Capítulo 1

Introducción a Network Coding

1.1. Primeros ejemplos

Para introducir el concepto de Network Coding (Códigos de Red) comenzamos considerando el siguiente ejemplo.

Tenemos dos personas A,B que quieren enviarse un mensaje entre ellos, pero no pueden hacerlo directamente. Para ello mandan los respectivos mensajes a,b a un repetidor R. Si hacemos uso de la comunicación tradicional, el repetidor enviará primero el mensaje a y después el mensaje b, a ambos.

$$\begin{aligned}
&\mathrm{A} \xrightarrow{a} \mathrm{R} \qquad \mathrm{B}\\
&\mathrm{A} \qquad \mathrm{R} \xleftarrow{b} \mathrm{B}\\
&\mathrm{A} \xleftarrow{a} \mathrm{R} \xrightarrow{a} \mathrm{B}\\
&\mathrm{A} \xleftarrow{b} \mathrm{R} \xrightarrow{b} \mathrm{B}
\end{aligned}$$

Haciendo uso de Network Coding, el repetidor envía un único mensaje $a+b$ a ambos, de esta forma, A, que había enviado el mensaje a recibe $a+b$ y puede recuperar el mensaje $b = (a+b)-a$, del mismo modo B recibe $a+b$ y puede recuperar el mensaje $a = (a+b)-b$.

$$\begin{aligned}
&\mathrm{A} \xrightarrow{a} \mathrm{R} \qquad \mathrm{B}\\
&\mathrm{A} \qquad \mathrm{R} \xleftarrow{b} \mathrm{B}\\
&\mathrm{A} \xleftarrow{a+b} \mathrm{R} \xrightarrow{a+b} \mathrm{B}
\end{aligned}$$

Consideramos ahora un ejemplo más complejo, usado en multitud de artículos como [9], [10] y [13]. Tomamos el grafo $G = (V,E)$ que podemos ver en la Figura 1.1, donde V es el conjunto de vértices del grafo, numerados del 1 al $|E| = 9$ y E el conjunto de aristas. En este caso, el emisor s quiere enviar el conjunto de mensajes $\{a,b\}$ a los receptores r_1 y r_2.

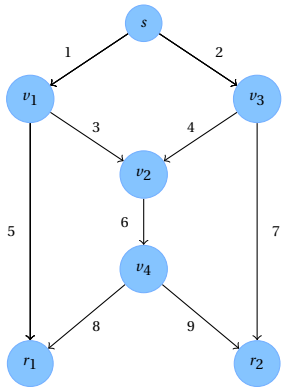

Figura 1.1: Red de Mariposa

***Definición* 1.1.** *Un camino simple es una secuencia, finita o infinita, de aristas que unen una secuencia de vértices y además son todos distintos. Un camino es dirigido si las aristas tienen todas la misma dirección.*

En este trabajo todos los caminos son dirigidos.

Si nos fijamos primero en r_1, encontramos un conjunto de dos caminos dirigidos disjuntos (sin ninguna arista en común) que parten de s y llegan a r_1,

$$\{(1,5),(2,4,6,8)\}. \tag{1.1}$$

Llamaremos a este conjunto flujo de tamaño dos, pues el conjunto tiene dos elementos. Si consideramos que el emisor s envía el mensaje a a través de la arista 1 y el mensaje b a través de la arista 2, el mensaje a se propaga hasta r_1 por el camino $(1,5)$ mientras que el mensaje b se propaga hasta r_1 por el camino $(2,4,6,8)$.

El flujo para r_2 es

$$\{(1,3,6,9),(2,7)\}. \tag{1.2}$$

De esta forma tendríamos dos soluciones parciales de nuestro problema que consiste en enviar los mensajes a, b, de manera simultánea, a los receptores r_1 y r_2. Si combinamos ambas soluciones vemos que si el emisor manda el mensaje a a través de 1 y el mensaje b a través de 2, por (1.1) la arista 6 envía el mensaje b mientras que por (1.2) dicha arista envía el mensaje a, como se puede ver en la Figura 1.2.

Para solventar este problema hacemos lo mismo que en ejemplo anterior, el vértice v_2 recibe el mensaje a por la arista 3 y el mensaje b por la arista 4, y envía la suma $a+b$ a través de la arista 6 que se transmite por las aristas 8 y 9 hasta r_1 y r_2 respectivamente, y otra vez $a = (a+b)-b$, $b = (a+b)-a$. De esta forma ambos receptores r_1 y r_2 pueden obtener el conjunto de mensajes $\{a,b\}$, como se puede ver en la Figura 1.3.

Esto es lo que conocemos como network coding, tenemos funciones de codificación que dada la información que entra por un vértice deciden

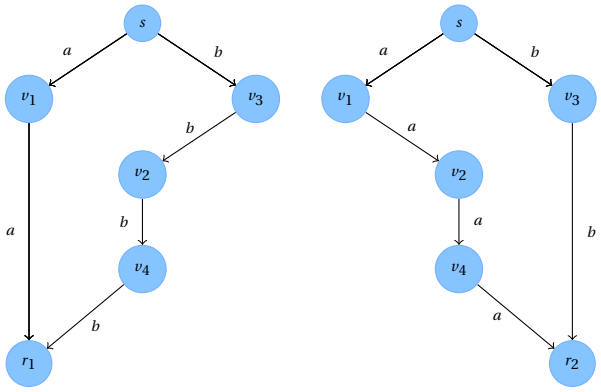

Figura 1.2: Flujo para r_1 (izquierda). Flujo para r_2 (derecha).

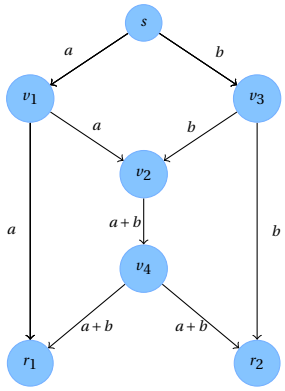

Figura 1.3: Problema con Network Coding.

que se envía a través de las aristas que parten de ese vértice, y funciones de decodificación en los receptores. El término network coding fue utilizado por primera vez hace dos décadas por Ashlswede, Cai, Li y Yeung en [1] y [22], aunque ahora mismo es un área que abarca temas que van desde la elección de coeficientes de codificación de manera aleatoria (random network coding), como veremos más adelante, hasta la correción de errores en redes o temas relacionados con criptografía.

1.2. El problema de Network Coding

Sea, al igual que antes, $G = (V, E)$ un grafo dirigido. Sea $S = \{s_1, \ldots, s_{|S|}\} \subseteq V$ el conjunto de emisores y $R = \{r_1, \ldots, r_{|R|}\} \subseteq V$ el conjunto de receptores, contenidos en el conjunto de vértices. Un vector mensaje es una variable $\overrightarrow{X} = (X_1, \ldots, X_h)$ cuyas componentes son los mensajes, y que puede tomar cualquier valor de $\mathbb{F}^h$, con $\mathbb{F}$ un cuerpo finito. Consideramos ahora la aplicación sobreyectiva $K : \{X_1, \ldots, X_h\} \to S$, también llamada función fuente. Diremos que el mensaje X_j se genera en s_j si $K(X_j) = s_j$. Definimos además el

$out(v)$ como el conjunto de aristas que salen del vértice v, y para una arista $e = (u, v)$ definimos $out(e) = out(v)$. De la misma forma definimos $in(v)$ e $in(e)$ como el conjunto de aristas que llegan a v y e respectivamente. En este trabajo asumiremos siempre que nuestro grafo dirigido G carece de ciclos, es decir, que podemos ordenar las aristas de forma que si existe un camino dirigido en el que se pasa antes por e_1 que por e_2, entonces $e_1 \prec e_2$, de esta forma podemos asumir que la transmisión no se produce de manera secuencial. A este orden definido de manera natural sobre las aristas le llamamos orden ancestral.

Para cada arista j tenemos una variable $Y(j)$ que toma valores en $\mathbb{F}$, nuestro cuerpo finito, definida a través de la función f_j de forma que

$$\begin{aligned} Y(j) &= f_j\left((Y(i)|i \in in(j)), (X_k|X_k \text{ se genera en } u)\right) \\ &= \sum_{i \in in(j)} f_{i,j} Y(i) + \sum_{\substack{X_i \in S \\ K(X_i) = u}} a_{i,j} X_i, \end{aligned} \tag{1.3}$$

con $f_{i,j}$, $a_{i,j} \in \mathbb{F}$.

Además, si la función f_j, a la que llamaremos función de codificación, carece de argumentos, $Y(j)$ toma el valor 0. También tenemos la función demanda D, que es una aplicación de R en el conjunto de subconjuntos ordenados no vacíos de $\{X_1, \ldots, X_h\}$. El receptor $r \in R$ demanda los mensajes $X_{i_1}, \ldots, X_{i_t}$ si tenemos que $D(r) = (X_{i_1}, \ldots, X_{i_t})$. Por tanto, para cada receptor tenemos asociadas $|D(r)|$ variables $Z_1^{(r)}, \ldots, Z_{|D(r)|}^{(r)}$ que toman valores en $\mathbb{F}$, para cada $Z_j^{(r)}$ tenemos una función $d_j^{(r)}$ de forma que

$$\begin{aligned} Z_j^{(r)} &= d_j^{(r)}\left((Y(i)|i \in in(r)), (X_k|K(X_k) = r)\right) \\ &= \sum_{i \in in(r)} b_{i,j}^{(r)} Y(i) + \sum_{\substack{X_i \in S \\ K(X_i) = r}} \widetilde{b}_{i,j}^{(r)} X_i, \end{aligned} \tag{1.4}$$

con $b_{i,j}^{(r)}$, $\widetilde{b}_{i,j}^{(r)} \in \mathbb{F}$.

Llamaremos problema de Network Coding a la red $G = (V, E)$ con emisores S, receptores R, vector de mensajes $\overrightarrow{X} = (X_1, \ldots, X_h)$, función fuente K y función de demanda D. Diremos que el problema tiene solución si existe un cuerpo finito distinto del trivial, es decir, distinto del cuerpo de un elemento, y un conjunto de funciones de codificación f_j y decodificación $d_j^{(r)}$ de forma que se cumpla

$$\left(Z_1^{(r)}, \ldots, Z_{|D(r)|}^{(r)}\right) = D(r), \qquad \forall r \in R.$$

Si tenemos que $D(r) = (X_1, \ldots, X_h) \quad \forall r \in R$, entonces estamos en el caso denominado multicast. Si tenemos un cuerpo finito $\mathbb{F}_q$ y las funciones de

codificación f_j y decodificación $d_j^{(r)}$ son lineales sobre $\mathbb{F}_q$ entonces hablamos de linear network coding (códigos lineales de red), en el que tenemos las variables $Y(j)$ y $Z_j^{(r)}$, donde $j = (u, v)$, vienen dadas por

$$Y(j) = \sum_{i \in in(j)} f_{i,j} Y(i) + \sum_{\substack{X_i \in S \\ K(X_i) = u}} a_{i,j} X_i, \tag{1.5a}$$

$$Z_j^{(r)} = \sum_{i \in in(r)} b_{i,j}^{(r)} Y(i) + \sum_{\substack{X_i \in S \\ K(X_i) = r}} \widetilde{b}_{i,j}^{(r)} X_i. \tag{1.5b}$$

Asumiremos en este trabajo que los conjuntos S y R son disjuntos, por lo que el último sumando de (1.4) y (1.5b) es nulo.

Por ejemplo, para el problema representado en la Figura 1.1, tenemos que $h = \#\{X_1, X_2\} = 2, |E| = 9$,

$$\begin{aligned} A &= \begin{bmatrix} a_{11} & a_{12} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ a_{21} & a_{22} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \end{aligned}$$

$$\begin{aligned} F &= \begin{bmatrix} 0 & 0 & f_{13} & 0 & f_{15} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & f_{24} & 0 & 0 & f_{27} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & f_{36} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & f_{46} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & f_{68} & f_{69} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \end{aligned}$$

$$B^{(r_1)} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ b_{51}^{(r_1)} & b_{52}^{(r_1)} \\ 0 & 0 \\ 0 & 0 \\ b_{81}^{(r_1)} & b_{82}^{(r_1)} \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}, \qquad B^{(r_2)} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ b_{71}^{(r_2)} & b_{72}^{(r_2)} \\ 0 & 0 \\ b_{91}^{(r_2)} & b_{92}^{(r_2)} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \\ 1 & 1 \end{bmatrix}.$$

Dado un problema de Network Coding, sea $r \in R$ y $D(r) = (X_{i_1}, \ldots, X_{i_t})$. Un flujo de r es un conjunto de t caminos disjuntos de S en r, donde para cada emisor $s \in S$ se originan $v^{(r)}(s)$ de esos caminos en s. Entendemos por $j \in F$ que j es una arista que forma parte F. Para poder resolver un problema de Network Coding necesitaremos que exista un flujo para todo receptor $r \in R$, además, en el caso de multicast, la existencia de un flujo para cada receptor es una condición suficiente para que se pueda resolver, como veremos más adelante en el Lema 1.2.

1.3. Linear network coding for multicast

Sea A una matriz de tamaño $h \times |E|$ que toma el valor $a_{i,j}$ en la entrada (i, j) si $j \in out(K(X_i))$ y 0 en caso contrario. Sea F una matriz de tamaño $|E| \times |E|$ que toma el valor $f_{i,j}$ en la entrada (i, j) si $j \in out(i)$ y 0 en caso contrario. Para cada $r \in R$ definimos la matriz $B^{(r)}$ de tamaño $|E| \times h$ que toma el valor $b_{i,j}^{(r)}$ en la entrada (i, j) si $i \in in(r)$ y 0 en caso contrario. Un par (A, F) o una upla $(A, F, B^{(r_1)}, \ldots, B^{(r_{|R|})})$ es lo que llamamos un código lineal de red.

Para cualquier n entero no negativo, la matriz F^n guarda información de los caminos de longitud n [21], es decir, los caminos compuestos por n aristas, de forma que la entrada (i, j) de la matriz F^n es

$$\sum_{\substack{(i = j_0, j_1, \ldots, j_n = j) \\ \text{es un camino} \\ \text{en G}}} f_{i=j_0, j_1} f_{j_1, j_2} \cdots f_{j_{n-1}, j_n = j}. \tag{1.9}$$

Como el grafo dirigido es acíclico, $F^N = 0$ para un cierto N lo suficientemente grande, por lo que la matriz $(I + F + \ldots + F^{N-1})$ guarda información de todos los posibles caminos en nuestro grafo. Si asignamos valores a los $a_{i,j}$ y los $f_{i,j}$, tenemos que

$$(Y(1), \ldots, Y(|E|)) = (X_1, \ldots, X_h) A (I + F + \ldots + F^{N-1}). \tag{1.10}$$

Además, si damos valores a los $b_{i,j}^{(r)}$ entonces

$$(Y(1),\ldots,Y(|E|))B^{(r)} = (X_1,\ldots,X_h)A(I+F+\ldots+F^{N-1})B^{(r)}, \tag{1.11}$$

que es la salida de la función de decodificación para cada receptor r.

Llamaremos matriz de transferencia para el receptor r a la matriz

$$M^{(r)} = A(I+F+\ldots+F^{N-1})B^{(r)}.$$

Haciendo uso de la igualdad $(I-F)(I+F+\ldots+F^{N-1}) = I$ (teniendo en cuenta que $F^N = 0$), obtenemos

$$M^{(r)} = A(I-F)^{-1}B^{(r)}. \tag{1.12}$$

La aplicación que nos lleva $[X_1,\ldots,X_h]$ en $[Z_1^{(r)},\ldots,Z_h^{(r)}]$ para el receptor r viene dada por la matriz de transferencia $M^{(r)}$.

Para que la codificación y la decodificación sea exitosa, en lugar de pedir que la matriz $M^{(r)}$ sea la matriz identidad I, pediremos que $M^{(r)}$ tenga rango máximo para todo receptor r, es decir,

$$\prod_{r\in R} \det M^{(r)} \neq 0, \tag{1.13}$$

pues tenemos que, dado un problema de network coding, si algún network code $(A,F,B^{(r_1)},\ldots,B^{(r_{|R|})})$ en el cuerpo $\mathbb{F}_q$ satisface que $M^{(r)}$ tiene rango máximo h para cada receptor r_i, con $i = 1,\ldots,|R|$, entonces $\tilde{B}^{(r_i)} = B^{(r_i)}{M^{(r_i)}}^{-1}$ cumple que $A(I-F)^{-1}\tilde{B}^{(r_i)} = I$, y $(A,F,\tilde{B}^{(r_1)},\ldots,\tilde{B}^{(r_{|R|})})$ es una solución de nuestro problema.

De esta forma, dados valores para los $a_{i,j}$, $f_{i,j}$ y $b_{i,j}^{(r)}$ de forma que se cumpla (1.13), podemos cambiar los valores de $b_{i,j}^{(r)}$ para obtener $M^{(r)} = \cdots = M^{|R|} = I$.

***Lema* 1.1.** *Dada la matriz de transferencia $M = A(I-F)^{-1}B^{(r)}$ de nuestra red para un receptor r, tenemos que*

$$\det M^{(r)} = (-1)^{h(|E|+1)}\det E^{(r)}$$

donde $E^{(r)}$ es la matriz de Edmond para r,

$$E^{(r)} = \begin{bmatrix} A & 0 \\ I-F & B^{(r)} \end{bmatrix}.$$

Demostración. Tenemos que

$$\begin{bmatrix} I & -A(I-F)^{-1} \\ 0 & I \end{bmatrix}\begin{bmatrix} A & 0 \\ I-F & B^{(r)} \end{bmatrix} = \begin{bmatrix} 0 & -A(I-F)^{-1}B^{(r)} \\ I-F & B^{(r)} \end{bmatrix}.$$

Como

$$\det \begin{bmatrix} I & -A(I-F)^{-1} \\ 0 & I \end{bmatrix} = 1,$$

se tiene que

$$\det \begin{bmatrix} A & 0 \\ I-F & B^{(r)} \end{bmatrix} = \det \begin{bmatrix} 0 & -A(I-F)^{-1}B^{(r)} \\ I-F & B^{(r)} \end{bmatrix}.$$

Desarrollando la expresión de la derecha,

$$\begin{aligned} \det \begin{bmatrix} 0 & -A(I-F)^{-1}B^{(r)} \\ I-F & B^{(r)} \end{bmatrix} &= (-1)^{h|E|} \det \begin{bmatrix} -A(I-F)^{-1}B^{(r)} & 0 \\ B^{(r)} & I-F \end{bmatrix} \\ &= (-1)^{h|E|} \det(-A(I-F)^{-1}B^{(r)}) \det(I-F) \\ &= (-1)^{h(|E|+1)} \det(A(I-F)^{-1}B^{(r)}) \det(I-F) \\ &= (-1)^{h(|E|+1)} \det(A(I-F)^{-1}B^{(r)}), \end{aligned}$$

pues $\det(I-F) = 1$ por ser F una matriz triangular superior con ceros en la diagonal. □

La ventaja que supone la matriz de Edmond respecto de la matriz de transferencia es que no es necesario calcular el N para el cual $F^N = 0$, además es claro que $\prod_{r\in R} \det M^{(r)} \neq 0 \iff \prod_{r\in R} \det E^{(r)} \neq 0$.

***Lema* 1.2.** *Un problema de network coding con h mensajes se puede resolver si y solo si cada receptor r tiene asociado un conjunto H_r de h aristas que llegan al receptor r para las que*

$$\sum_{\text{caminos disjuntos}} |A_{\{l_{1,1},\ldots,l_{h,1}\}}| \prod_{j=1}^{h} g(\varepsilon_j) \neq 0, \tag{1.14}$$

donde los caminos disjuntos son de la forma $\varepsilon_1 = (l_{1,1},\ldots,l_{1,n_1})$, …, $\varepsilon_h = (l_{h,1},\ldots,l_{h,n_h})$ tal que $l_{i,1}$ parte de S_i, $l_{i,n_h} \in H_r$; $A_{\{l_{1,1},\ldots,l_{h,1}\}}$ es la submatriz de A que consiste en las aristas $\{l_{1,1},\ldots,l_{h,1}\}$; y

$$g(\varepsilon) = \begin{cases} f_{e_1,e_2} f_{e_2,e_3} \cdots f_{e_{k-1},e_k}, & \text{si } k > 1 \\ 1, & \text{si } k = 1 \end{cases}$$

es el producto de los coeficientes de codificación correspondientes al camino $\varepsilon = (e_1,\ldots,e_k)$. La suma es sobre todos los flujos desde donde se generan los mensajes hasta las aristas en H_r, siendo cada uno de ellos un conjunto de h caminos disjuntos que parte de los emisores y llega a una arista distinta en H_r.

Demostración. Asumimos primero que existe una cierta numeración de las aristas de nuestra red. Para $1 \le \eta' \le \eta \le |\varepsilon|$, sea $\theta_{\eta',h}$ el conjunto de todos los conjuntos de enteros $\{e_1, e_2, \ldots, e_k\}$ tal que $\eta' \le e_1 < e_2 < \cdots < e_k = \eta$. Sea $H = \{\eta_1, \ldots, \eta_h\}$, donde $1 \le \eta_1 < \cdots < \eta_h \le |\varepsilon|$.

Sea $\mathbf{a}_\eta$ y $\mathbf{c}_\eta$ la η-ésima columna de A y $A(I-F)^{-1}$ respectivamente. Por como están definidas las matrices A y $(I_F)^{-1} = I + F + F^2 + \cdots$, podemos definir $\mathbf{c}_\eta$ de manera recursiva como sigue:

$$\mathbf{c}_1 = \mathbf{a}_1, \tag{1.15a}$$

$$\mathbf{c}_\eta = \sum_{i=1}^{\eta-1} \mathbf{c}_i f_{i,\eta} + \mathbf{a}_\eta, \quad \eta = 2, 3, \ldots, |\varepsilon|. \tag{1.15b}$$

Expandimos ahora el determinante de $A(I-F)^{-1}$ por la η-ésima columna usando (1.15b), y obtenemos

$$\begin{aligned} |A(I-F)^{-1}_H| &= \begin{vmatrix} | & & | \\ \mathbf{c}_{\eta_1} & \cdots & \mathbf{c}_{\eta_h} \\ | & & | \end{vmatrix} \\ &= \sum_{\substack{\{i\,:\,1 \le i < \eta_h,\\ i \ne \eta_1, \ldots, \eta_{h-1}\}}} \begin{vmatrix} | & & | & | \\ \mathbf{c}_{\eta_1} & \cdots & \mathbf{c}_{\eta_{h-1}} & \mathbf{c}_i \\ | & & | & | \end{vmatrix} f_{i,\eta_h} \\ &\quad + \begin{vmatrix} | & & | & | \\ \mathbf{c}_{\eta_1} & \cdots & \mathbf{c}_{\eta_{h-1}} & \mathbf{a}_{\eta_h} \\ | & & | & | \end{vmatrix}. \end{aligned}$$

Seguimos de manera recursiva, expandiendo de esta forma el determinante en su columna $\mathbf{c}_\eta$ cuyo índice η es el mayor, usando (1.15b) para $\eta > 1$ y (1.15a) para $\eta = 1$. Cada vez que hacemos este proceso, obtenemos el determinante de $A(I-F)^{-1}_H$ como combinación lineal de determinantes. Cada determinante no nulo corresponde una matriz compuesta por columnas de $\{\mathbf{a}_{k_1}, \ldots, \mathbf{a}_{k_s}, \mathbf{c}_{k_{s+1}}, \mathbf{c}_{k_h}\}$ tal que $k_i \ne k_j$ para todo $i \ne j$, y $\min(k_1, \ldots, k_s) > \max(k_{s+1}, \ldots, k_h)$. El coeficiente de cada término es un producto de términos $f_{i,\eta}$ tal que $\eta > k_{s+1}, \ldots, k_h$, y es de la forma $\prod_{j=1}^{h} g(\varepsilon_j)$ donde $\varepsilon_j \in S_{k_{j'},\eta_j}$ para algún $j' \in [1,h]$ y $\varepsilon_i \cap \varepsilon_j = \emptyset$ para todo $i \ne j$. Por inducción tenemos estas propiedades para todos los determinantes no nulos en la expresión expandida.

La inducción termina cuando la expresión es una combinación lineal

de determinantes de la forma $|\mathbf{a}_{l_1} \cdots \mathbf{a}_{l_h}|$. Entonces tenemos que

$$|A(I-F)_H^{-1}| = \sum_{\substack{\{(\eta'_1,\ldots,\eta'_h):\\ 1 \le \eta'_j \le \eta_j,\\ \eta'_i \ne \eta'_j, \forall\, i \ne j\}}} \begin{vmatrix} | & & | \\ \mathbf{a}_{\eta'_1} & \cdots & \mathbf{a}_{\eta'_h} \\ | & & | \end{vmatrix} \sum_{\substack{\{(\varepsilon_1,\ldots,\varepsilon_h):\\ \varepsilon_j \in S_{\eta'_j,\eta_j},\\ \varepsilon_i \cap \varepsilon_j 0 \emptyset\\ \forall\, i \ne j\}}} \prod_{j=1}^{h} g(\varepsilon_j).$$

Teniendo en cuenta que cada conjunto $\varepsilon = \{e_1, e_2, \ldots, e_k\}$ tal que $g(\varepsilon) \neq 0$ se corresponde a un camino que consiste en las aristas $e_1, e_2, \ldots, e_k$; que la condición $\varepsilon_j \cap \varepsilon_k = \emptyset$ para todo $j \neq k$, $1 \leq j, k \leq h$ implica que los caminos son disjuntos y que el determinante $|\mathbf{a}_{\eta'_1} \cdots \mathbf{a}_{\eta'_h}|$ es no nulo si las aristas $\mathbf{a}_{\eta'_j}$ transmiten h combinaciones lineales independientes de los mensajes. □

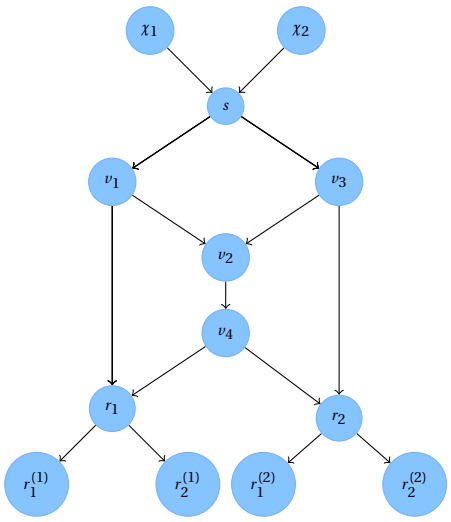

Figura 1.4: Red de mariposa extendida

El determinante de la matriz de transferencia tiene un significado topológico. Para ello consideramos la extensión del grafo $G = (V, E)$, como podemos observar en la Figura 1.4. Añadimos h vértices $\chi_1, \ldots, \chi_h$ a V, después, para cada $i = 1, \ldots, h$ añadimos una arista de χ_i al emisor donde se genera X_i. Además, para cada receptor $r \in R$ añadimos h vértices $r^{(1)}, \ldots, r^{(h)}$, y

por último añadimos una arista de r a cada $r^{(i)}$. Definimos ahora para cada receptor r un flujo compuesto por caminos disjuntos de $\{\chi_1,\dots,\chi_h\}$ en $\{r^{(1)},\dots,r^{(h)}\}$.

De esta forma un camino en el flujo extendido se corresponde con un producto de un único a_{ij}, varios f_{ij} y un único b_{ij}, luego por (1.9) la posición (i,j) de la matriz $M^{(r)}$ contiene la información de todos los caminos de χ_i en $r^{(j)}$. En el caso de la red de la Figura 1.1,

$$M^{(r_1)} = \begin{bmatrix} a_{1,1}f_{1,5}b_{5,1}^{(r_1)} + [a_{1,1}(f_{1,3}f_{3,6}f_{6,8}) + a_{1,2}(f_{2,4}f_{4,6}f_{6,8})]b_{8,1}^{(r_1)} & a_{1,1}f_{1,5}b_{5,2}^{(r_1)} + [a_{1,1}(f_{1,3}f_{3,6}f_{6,8}) + a_{1,2}(f_{2,4}f_{4,6}f_{6,8})]b_{8,2}^{(r_1)} \\ a_{2,1}f_{1,5}b_{5,1}^{(r_1)} + [a_{2,1}(f_{1,3}f_{3,6}f_{6,8}) + a_{2,2}(f_{2,4}f_{4,6}f_{6,8})]b_{8,1}^{(r_1)} & a_{2,1}f_{1,5}b_{5,2}^{(r_1)} + [a_{2,1}(f_{1,3}f_{3,6}f_{6,8}) + a_{2,2}(f_{2,4}f_{4,6}f_{6,8})]b_{8,2}^{(r_1)} \end{bmatrix},$$

$$M^{(r_2)} = \begin{bmatrix} a_{1,2}f_{2,7}b_{7,1}^{(r_2)} + [a_{1,1}(f_{1,3}f_{3,6}f_{6,9}) + a_{1,2}(f_{2,4}f_{4,6}f_{6,9})]b_{9,1}^{(r_2)} & a_{1,2}f_{2,7}b_{7,2}^{(r_2)} + [a_{1,1}(f_{1,3}f_{3,6}f_{6,9}) + a_{1,2}(f_{2,4}f_{4,6}f_{6,9})]b_{9,2}^{(r_2)} \\ a_{2,2}f_{2,7}b_{7,1}^{(r_2)} + [a_{2,1}(f_{1,3}f_{3,6}f_{6,9}) + a_{2,2}(f_{2,4}f_{4,6}f_{6,9})]b_{9,1}^{(r_2)} & a_{2,2}f_{2,7}b_{7,2}^{(r_2)} + [a_{2,1}(f_{1,3}f_{3,6}f_{6,9}) + a_{2,2}(f_{2,4}f_{4,6}f_{6,9})]b_{9,2}^{(r_2)} \end{bmatrix}.$$

Tomando $a_{1,1} = a_{2,2} = 1$, y $a_{2,1} = a_{1,2} = 0$, tenemos que

$$\det M^{(r_1)} \det M^{(r_2)} = \left(f_{1,3}f_{1,5}f_{2,4}f_{2,7}f_{3,6}f_{4,6}f_{6,8}f_{6,9}\right) \det \tilde{B}^{(r_1)} \det \tilde{B}^{(r_2)},$$

donde $\tilde{B}^{(r_i)}$ es una matriz formada por las filas no nulas de $B^{(r_i)}$.

Hay una correspondencia entre los flujos a $r^{(1)},\dots,r^{(h)}$ en el grafo extendido y las expresiones monomiales que aparecen en $\det M^{(r)}$. Estas expresiones aparecen con coeficientes 1 ó -1. Considerando ahora el producto $\prod_{r\in R} \det M^{(r)}$, los coeficientes pasan a ser miembro de $\mathbb{F}_p$, donde p es la característica del cuerpo $\mathbb{F}_q$.

Tomamos ahora la red junto con su red expandida, que se muestra en la Figura 1.5. En este problema tenemos que $h = \#\{X_1, X_2, X_3\} = 3$, $|E| = 18$,

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ a_{21} & a_{22} & a_{23} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ a_{31} & a_{32} & a_{33} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$
$$= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$B^{(r_1)} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ b_{4,1}^{(r_1)} & b_{4,2}^{(r_1)} & b_{4,3}^{(r_1)} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ b_{12,1}^{(r_1)} & b_{12,2}^{(r_1)} & b_{12,3}^{(r_1)} \\ b_{13,1}^{(r_1)} & b_{13,2}^{(r_1)} & b_{13,3}^{(r_1)} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B^{(r_2)} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ b_{5,1}^{(r_1)} & b_{5,2}^{(r_1)} & b_{5,3}^{(r_1)} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ b_{14,1}^{(r_1)} & b_{14,2}^{(r_1)} & b_{14,3}^{(r_1)} \\ b_{15,1}^{(r_1)} & b_{15,2}^{(r_1)} & b_{15,3}^{(r_1)} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix},$$

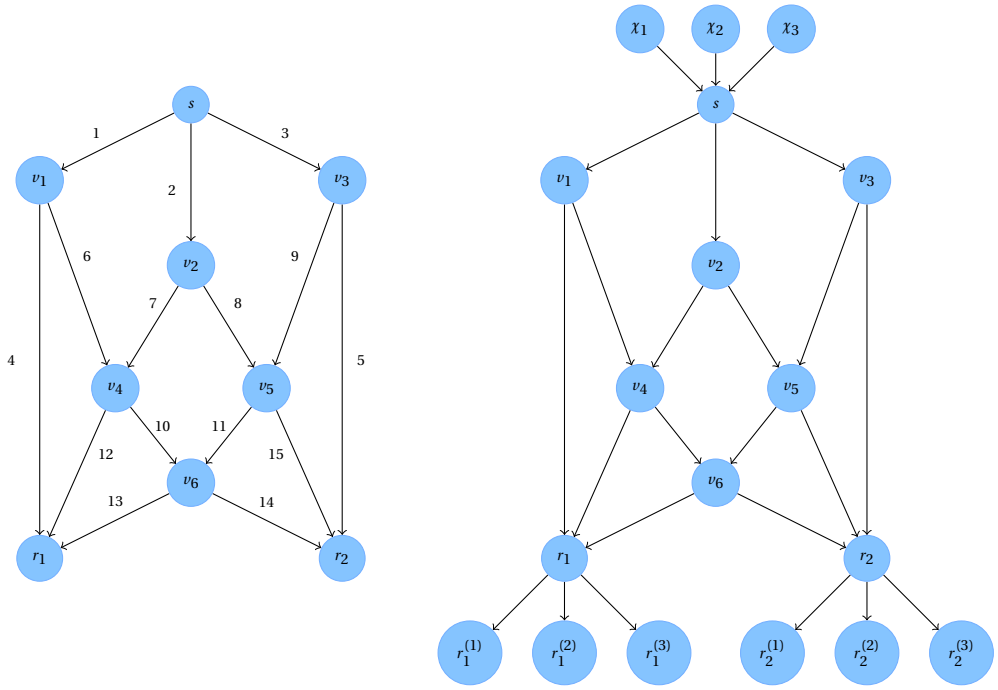

Figura 1.5: Ejemplo de red junto con su red expandida.

$$F = \begin{bmatrix}
0 & 0 & 0 & f_{1,4} & 0 & f_{1,6} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & f_{2,7} & f_{2,8} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & f_{3,5} & 0 & 0 & 0 & f_{3,9} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & f_{6,10} & 0 & f_{6,12} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & f_{7,10} & 0 & f_{7,12} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & f_{8,11} & 0 & 0 & 0 & f_{8,15} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & f_{9,11} & 0 & 0 & 0 & f_{9,15} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & f_{10,13} & f_{10,14} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & f_{11,13} & f_{11,14} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}$$

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

Podemos considerar los coeficientes $a_{i,j}$, $f_{i,j}$ y $b_{i,j}^{(r)}$ como variables cuando son desconocidos. El polinomio $\prod_{r\in R} \det M^{(r)}$ es lo que llamamos el polinomio de transferencia. El polinomio de transferencia nos permite ver cuando un problema de network coding se puede resolver. Además, como veremos en el capítulo siguiente, es útil a la hora la utilización de random network coding.

***Proposición* 1.1.** *El polinomio de transferencia es no nulo si y solo si existe un flujo de tamaño h para cada receptor.*

***Lema* 1.3.** *Consideramos un problema de network coding en el que ν aristas j tienen coeficientes $a_{i,j}$ y/o $f_{l,j}$ y/o $b^{(r)}_{k,j}$. Entonces el polinomio asociado al determinante de la matriz de Edmond*

$$\begin{bmatrix} A & 0 \\ I-F & B^{(r)} \end{bmatrix}$$

tiene grado máximo ν en las variables $\{a_{i,j}, f_{l,j}, b^{(r)}_{k,j}\}$ y es lineal en cada una de esas variables.

Demostración. Cada variable $\{a_{i,j}, f_{l,j}, b^{(r)}_{k,j}\}$ aparece en una única columna de la matriz de Edmond y, además, únicamente las ν columnas que se corresponden con las arisas que transmiten combinaciones aleatorias contienen las variables $\{a_{i,j}, f_{l,j}, b^{(r)}_{k,j}\}$.

El determinante se puede escribir como suma de productos de $|R|+|E|$ entradas de la matriz, una de cada fila y columna. Cada producto es lineal en cada variable $\{a_{i,j}, f_{l,j}, b^{(r)}_{k,j}\}$, y tiene grado máximo ν en esas variables. □

Denotamos como $\mathcal{F} = \{F_1, \ldots, F_{|R|}\}$ donde F_i es el flujo al receptor r_i para $i = 1, \ldots, |R|$. En el polinomio de transferencia, los $a_{i,j}$ y los $f_{i,j}$ aparecen a lo sumo con potencia $|R|$ por el Lema 1.3 y teniendo en cuenta que el polinomio de transferencia se obtiene como el producto de $|R|$ determinantes, mientras que los $b^{(r)}_{i,j}$ aparecen con potencia 1, como mucho, pues solo aparecen en uno de los |R| determinantes.

1.4. Variedades, huella de un ideal y otros resultados sobre polinomios

En esta sección veremos algunas definiciones y resultados del Algebra Conmutativa que podemos encontrar en [7] (ver también [8]) y que nos serán útiles para obtener resultados relacionados con network coding. Los conceptos más básicos como Bases de Gröbner y otros se darán por conocidos, pues han sido estudiados en diversas asignaturas de la carrera y máster.

***Definición* 1.2.** *Sea $\mathbb{K}$ un cuerpo, y sean $f_1, \ldots, f_s$ polinomios en $\mathbb{K}[x_1, \ldots, x_n]$. Entonces*

$$V(f_1, \ldots, f_s) = \{(a_1, \ldots, a_n) \in \mathbb{K}^n : f_i(a_1, \ldots, a_n) = 0, \forall i \in [1, s]\}. \tag{1.20}$$

$V(f_1,\dots,f_s)$ es lo que conocemos como variedad afín definida por $f_1,\dots,f_s$. Es decir, una variedad afín $V(f_1,\dots,f_s)\subset\mathbb{K}^n$ es el conjunto de todas las soluciones del sistema de ecuaciones $f_1(x_1,\dots,x_n)=\cdots=f_s(x_1,\dots,x_n)=0$.

***Definición* 1.3.** *Sea $\prec$ un orden monomial , $\mathbb{F}$ un cuerpo cualquiera y consideramos un ideal $I\subseteq\mathbb{F}[X_1,\dots,X_m]$. El "footprint", o huella, de I respecto del orden monomial $\prec$ es el conjunto de monomios que no se pueden encontrar como monomio líder de ningún polinomio del ideal. Lo denotamos por $\Delta_\prec(I)$, que son los generadores como espacio vectorial sobre $\mathbb{F}$ de $\mathbb{F}[X_1,\dots,X_m]/I$ (consultar [8]).*

***Teorema* 1.1.** *Sea $I\subset\mathbb{C}[x_1,\dots,x_n]$ un ideal tal que su variedad $V=V(I)$ es un conjunto finito.*

El número de puntos en V es como mucho $\dim(\mathbb{C}[x_1,\dots,x_n]/I)$*, entendiendo por "dim" la dimensión como espacio vectorial sobre $\mathbb{C}$.*

Demostración. Primero veremos que dados puntos $p_1,\dots,p_m\in\mathbb{C}^n$ todos ellos distintos, hay un polinomio $f_1\in\mathbb{C}[x_1,\dots,x_n]$ con $f_1(p_1)=1$ y $f_1(p_2)=\cdots=f_1(p_m)=0$. Para ello vemos que si $a\neq b\in\mathbb{C}^n$, entonces tienen que ser distintos en una cierta coordenada, digamos la j-ésima, y por tanto $g=(x_j-b_j)/(a_j-b_j)$ satisface que $g(a)=1$ y $g(b)=0$. Aplicando esta observación a cada par $p_1\neq p_i$, $i\geq 2$, obtenemos los polinomios g_i con $g_i(p_1)=1$ y $g_i(p_i)=0$ para $i\geq 2$, obteniendo así $f_1=g_2\cdot g_3\cdots g_m$.

Podemos aplicar este mismo argumento para obtener los polinomios $f_1,\dots,f_m$ con $f_i(p_i)=1$ y $f_i(p_j)=0$ para $i\neq j$.

Ahora, para probar el teorema, suponemos que $V=\{p_1,\dots,p_m\}$, donde los p_i son distintos. De esta forma podemos obtener los $f_1,\dots,f_m$ al igual que antes. Ahora, si podemos probar que $[f_1],\dots,[f_m]\in\mathbb{C}[x_1,\dots,x_n]/I$ son linealmente independientes tenemos que

$$m\leq\dim(\mathbb{C}[x_1,\dots,x_n]/I), \tag{1.21}$$

y la proposición estaría probada.

Para probar la independencia lineal suponemos que $\sum_{i=1}^m a_i[f_i]=[0]$ en $\mathbb{C}[x_1,\dots,x_n]/I$, donde los $a_i\in\mathbb{C}$. Esto nos dice que $g=\sum_{i=1}^m a_i f_i\in I$, es decir, que g se anula en todos los puntos de la variedad $V=\{p_1,\dots,p_m\}$. Entonces, para $1\leq j\leq m$, tenemos que

$$0=g(p_j)=\sum_{i=1}^m a_i f_i(p_j)=0+a_j f_j(p_j)=a_j,$$

y, por tanto, son linealmente independientes. □

***Proposición* 1.2.** *Si $I\subseteq\mathbb{F}[X_1,\dots,X_m]$ tiene dimensión finita, entonces la variedad $V_{\overline{\mathbb{F}}}(I)$, donde $\overline{\mathbb{F}}$ es la clausura algebraica de $\mathbb{F}$, tiene dimensión $|\Delta_\prec(I)|$ como mucho.*

La diferencia entre el Teorema 1.1 y la Proposición 1.2 es que para el Teorema 1.1 trabajamos con el cuerpo $\mathbb{C}$ y pedimos que la variedad del ideal $V = V(I)$ sea un conjunto finito, mientras que en la Proposición 1.2 estamos trabajando con un cuerpo $\mathbb{F}$ cualquiera al que le pedimos que sea finito.

***Corolario* 1.1.** *Sea $F(X_1,\dots,X_m) \in \mathbb{F}[X_1,\dots,X_m]$ un polinomio no nulo, donde $\mathbf{F}$ es un cuerpo que contiene a $\mathbf{F}_q$ y fijamos un orden monomial cualquiera $\prec$. Suponemos que el monomio líder de nuestro polinomio F es $X_1^{i_1}\cdots X_m^{i_m}$, con $0 \le i_1,\dots,i_m < q$. Entonces el número de elementos de $F(X_1,\dots,X_m)$ no nulos en $\mathbb{F}_q^m$ es al menos $(q-i_1)\cdots(q-i_m)$.*

Demostración. Lo primero a tener en cuenta es que podemos reemplazar nuestro ideal $I = \langle F(X_1,\dots,X_m)\rangle$ por otro ideal que tenga la misma variedad, como es el ideal $I' = \langle F(X_1,\dots,X_m), X_1^q - X_1,\dots,X_m^q - X_m\rangle$, de esta forma nos aseguramos que la huella es finita. Aplicando ahora la Proposición (1.2), tenemos que el número de ceros del polinomio $F(X_1,\dots,X_m)$ en $\mathbb{F}_q^m$ es como mucho $|\Delta_\prec(I')|$. Además, si tenemos en cuenta que

$$|\Delta_\prec(I')| \subseteq \left\{ X_1^{s_1}\cdots X_m^{s_m} \mid \begin{array}{c} 0 \le s_t < q \text{ para } t = 1,\dots,m \\ i_t \le s_t \text{ no se cumple para todo } t = 1,\dots,m \end{array} \right\},$$

tenemos la acotación buscada. □

***Teorema* 1.2.** *Diremos que un problema de multicast network coding se puede resolver si y solo si el polinomio de transferencia correspondiente es no nulo. Cuando el problema de network coding se puede resolver entonces el problema de linear network coding funciona para todo cuerpo $\mathbb{F}_q$ con $q \ge |R|$.*

Demostración. Lo primero que necesitamos es que exista un sistema de flujo de tamaño h para que podamos tener multicast. Haciendo uso de la Proposición (1.1) la segunda implicación está probada. Para probar la primera implicación comenzamos suponiendo que existe un sistema de flujo de tamaño h. Entonces, por la Proposición 1.1 tenemos que el polinomio de transferencia es no nulo y que no hay ninguna variable elevada a una potencia mayor que $|R|$, entonces, por el Corolario 1.1 existe una solución no nula para el problema de linear network coding si el número de elementos del cuerpo es mayor que $|R|$. □

***Teorema* 1.3.** *Un problema de multicast network coding se puede resolver si y solo si el grafo contiene un sistema de flujo de tamaño h.*

Tenemos que todos los coeficientes en nuestro polinomio de transferencia pertenecen a $\mathbb{F}_p$, donde p es la característica del cuerpo. Buscamos la menor potencia m de p para la que los coeficientes se puedan elegir en

$\mathbb{F}_{p^m}$ de forma que la comunicación sea satisfactoria. De esta forma, para exponentes mayores de p, podemos hacer el polinomio de transferencia módulo $X^{p^m} - X$, donde X recorre todos los $a_{i,j}$, $f_{i,j}$, sabíamos que los $b_{i,j}^{(r)}$ aparecen todos con potencia 1 luego con ellos no hay problema. De esta forma, reemplazamos todos los X^{p^m} por X, entonces el problema se podrá resolver si y solo el polinomio que resulta es no nulo.

1.5. El algoritmo de Jaggi

Dado un problema de multicast network coding podemos encontrar fácilmente un sistema de flujo, como podemos ver en el capítulo 26 de [6]. Además, dado un sistema de flujo de tamaño h, existen algoritmos de baja complejidad (tiempo polinomial) que nos permiten encontrar una solución al problema en el cuerpo $\mathbb{F}_q$, para cualquier $q \geq |R|$. Uno de ellos es el algoritmo de Jaggi.

***Definición* 1.4.** *Dados los coeficientes $a_{i,j}$, $f_{i,j}$ para un problema de network coding, para cada arista j definimos el vector de codificación global como $c_g(j) = (c_1, \ldots, c_h)$ si tenemos que $Y(j) = c_1 X_1 + \cdots + c_h X_h$.*

Entonces tenemos, para cada receptor r, que si los vectores globales de codificación de $\text{in}(r)$ generan todo $\mathbb{F}_q^h$, entonces los coeficientes de codificación $a_{i,j}$, $f_{i,j}$ son parte de la solución del problema de network coding.

Lo único que nos faltaría es elegir los $b_{i,j}^{(r)}$, los cuales podemos calcular fácilmente a través del algebra lineal.

Para comenzar con el algoritmo es necesario modificar nuestro problema de network coding. Comenzamos tomando un nuevo vértice s', entonces, para cada emisor $s_i \in S$, añadimos $v(s_i)$ aristas de s' en s_i, donde s_i es el número de mensajes que se crean en el vértice s_i. En total se añaden h nuevas aristas, donde h es el número de mensajes que tenemos. Ahora asumimos que s' genera los h mensajes de nuestro problema, y sea $\mathcal{F} = (F_1, \ldots, F_{|R|})$ un sistema de flujo, donde F_i es el flujo que va al receptor r_i. Para utilzar este algoritmo necesitamos el problema de network coding modificado definido anteriormente.

Para inicializar el algoritmo, tenemos que:

- Si (i, j) no forma parte de ningún camino en nuestro sistema de flujo, entonces tomamos $f_{i,j} = 0$.
- Elegimos las funciones de codificación en el punto s' de forma que los vectores de codificación global de las h nuevas aristas sean

$$(1, 0, \ldots, 0), (0, 1, 0, \ldots, 0), \ldots, (0, \ldots, 0, 1) \tag{1.22}$$

(no es necesario guardar la información de las funciones de codificación en s' puesto que no forman parte de nuestro problema original, y no son relevantes).

Comenzamos inicializando $C_1 = \cdots = C_{|R|}$ como el conjunto ordenado de las nuevas h aristas, y $B_1, \ldots, B_{|R|}$ el conjunto ordenado correspondiente de vectores de codificación global, que se corresponde con (1.22).

La parte principal de nuestro algoritmo consiste en ir modificando los conjuntos ordenados $C_1, \ldots, C_{|R|}$, y los coeficientes de codificación relacionados con las modificaciones son elegidos de forma que para $l = 1, \ldots, |R|$, se tengan las siguientes dos propiedades:

- C_l constituye un corte en F_l, es decir, que los caminos contenidos en C_l forman parte de caminos existentes en F_l.
- El conjunto ordenador de vectores de codificación global correpondiente a C_l genera todo $\mathbb{F}_q^h$.

Las modificaciones las vamos haciendo de forma que, al final de un número finito de pasos, se cumple que $C_l = \text{in}(r_l) \cap F_l$ para $l = 1, \ldots, |R|$. Es decir, por la segunda propiedad, que para cada receptor se puedan calcular los $b_{i,j}^{(r)}$ mediante técnicas de algebra lineal. Esto va a suceder siempre pues el flujo a cada receptor es finito, además, como el conjunto de vectores de codificación global genera todo $\mathbb{F}_h^q$, tenemos un sistema

$$\begin{pmatrix} Y(l_1) \\ \vdots \\ Y(l_h) \end{pmatrix} = \begin{bmatrix} c_g(l_1) \\ \vdots \\ c_g(l_h) \end{bmatrix} \begin{pmatrix} X_1 \\ \vdots \\ X_h \end{pmatrix}, \tag{1.23}$$

para cada receptor r, con $in(r) = \{l_1, \ldots, l_h\}$. Es un sistema lineal en el que las incógnitas son $X_1, \ldots, X_h$ y, por álgebra lineal, sabemos que el sistema se puede resolver si efectivamente el conjunto de vectores de codificación global general todo el espacio.

El conjunto de aristas $j \in \mathcal{F}$ se recorre siguiendo el orden ancestral. Dada una arista j, nos interesan únicamente los receptores r_l para los que $j \in F_l$. Entonces añadimos j al conjunto ordenado C_l y eliminamos la arista anterior en F_l (por la que hemos llegado a j), es decir, eliminamos la única arista $i \in (C_l \cap \text{in}(j))$ para la cual (i, j) forma parte de un camino en F_l. Sean $l_1, \ldots, l_k$ los valores de l para los cuales tenemos que modificar C_l, y sean $i_1, \ldots, i_k$ (pudiera darse el caso que $i_s = i_t$ para $s \neq t$) las aristas que quitamos de los conjuntos $C_{l_1}, \ldots, C_{l_k}$. Lo que necesitamos saber ahora es cómo elegir los coeficientes $f_{i_1,j}, \ldots, f_{i_k,j}$ de forma que todos los conjuntos de vectores de codificación global $B_{l_1}, \ldots, B_{l_k}$ asociados a $C_{l_1}, \ldots, C_{l_k}$ generan todo $\mathbb{F}_q^h$. La probabilidad de éxito eligiendo los coeficientes de codificación de manera aleatoria es no nula, como garantiza el siguiente lema.

***Lema* 1.4.** *Dada una base* $\{\vec{b}_1,\dots,\vec{b}_h\}$ *de* $\mathbb{F}_q^h$ *y* $\vec{c}\in\mathbb{F}_q^h$*, existe una única elección de* $a\in\mathbb{F}_q$ *tal que*

$$\vec{c}+a\vec{b}_h\in \mathrm{Span}_{\mathbb{F}_q}\{\vec{b}_1,\dots,\vec{b}_{h-1}\}. \tag{1.24}$$

Demostración. Si escribimos $\vec{c}$ como combinación lineal $\vec{c}=c_1b_1+\cdots+c_nb_n$, solo existe un $a\in\mathbb{F}_q$ para el que se da (1.24), y ese es $a=-c_h$.

□

Aplicamos ahora el Lema 1.4 para cada i_t con la base B_{l_t}, con $\vec{b}_h=c_g(i_t)$, y con

$$\vec{c}=\sum_{i\in\{i_1,\dots,i_k\}\setminus\{i_t\}} f_{l,j}c_g(l).$$

Sea $k'=|\{i_1,\dots,i_k\}|=|(\mathrm{in}(j)\cap\mathcal{F})|$. La probabilidad de que la elección de los coeficientes, en un paso del algoritmo, haga que podamos obtener una solución es, al menos,

$$\frac{q^{k'}-kq^{k'-1}+(k-1)}{q^{k'}}=1-\frac{k}{q}+\frac{k-1}{q^{k'}}. \tag{1.25}$$

Esta probabilidad la obtenemos asumiendo que si la elección de los coeficientes de codificación no nos proporciona una solución para r_{l_t} si que la proporciona para $r_{l_1},\dots,r_{l_{t-1}},r_{l_{t+1}},\dots,r_k$. De esta forma contamos la elección $f_{i_1,j}=\cdots=f_{i_k,j}=0$ k veces.

La expresión (1.25) es positiva si tenemos que $q\geq k$, y como $k\leq|R|$, hemos probado que para un problema de network coding que se pueda resolver $\mathbb{F}_q$ es suficiente si $q\geq|R|$.

A continuación vemos escrito el algoritmo de Jaggi, que también podemos ver en [11],escrito en pseudocódigo.

***Algoritmo* 1.1.**

Input: *Un problema (acíclico) de multicast netowork coding que se pueda resolver, con G=(V,E). Un cuerpo $\mathbb{F}_q$ con $q \geq |R|$. Un orden ancestral definido sobre E.*

Output: *Coeficientes de codificación $a_{i,j}$, $f_{i,j}$.*

Inicialización

Sea $V' := V \cup \{s'\}$ y $E' := E \cup \{e_1, \ldots, e_h\}$, donde las nuevas aristas son las descritas anteriormente;

Encontrar un sistema de flujo $\mathcal{F} = \{F_1, \ldots, F_{|R|}\}$ en $G' := (V', E')$ de s' a cada receptor;

for $l = 1, \ldots, |R|$

$C_l := (e_1, \ldots, e_h)$;

$B_l := ((1,0,\ldots,0),(0,1,0,\ldots,0),\ldots,(0,\ldots,0,1))$;

endfor

Actualización

for $j \in E$ *por el orden ancestral*

Sean $(r_{l_1}, \ldots, r_{l_k})$ los receptores con j en $\mathcal{F}$;

Sean $(i_1, \ldots, i_k)$ los predecesores de j en $\mathcal{F}$;

if $e \notin \mathcal{F}$ ***then***

$f_{p,e} = 0$ *para todo $p \in in(e)$, de este modo, $c_g(e) = 0$;*

else if $(i_1, \ldots, i_k) = i_1$ ***then***

$f_{p,e} = \mathbb{1}[p = i_1]$, *donde i_1 es el único predecesor de e en $\mathcal{F}$, de este modo, $c_g(e) = c_g(i_1)$;*

else

Elegir los coeficientes de codificación aleatoriamente tal que $c_g(e) \notin span\{B_r - c_g(i_v)\}$ para todo r_{l_v};

else

endfor

for $r_{l_v} \in (r_{l_1}, \ldots, r_{l_k})$

$C_{l_v} := (C_{l_v} \setminus \{i_v\}) \cup \{j\}$;

$B_{l_v} := c_g(C_{l_v})$;

endfor

devolver *todos los valores $f_{i,j}$.*

Para finalizar este capítulo, vamos a considerar un par de ejemplos para ver el funcionamiento del Algoritmo 1.1.

***Ejemplo* 1.1.** *Consideramos la red dada por la Figura 1.1. Primero comenzamos considerando la red ampliada. Para ello, consideramos un nuevo vértice e' en el que se generaran ambos mensajes X_1 y X_2, y consideramos aristas que unen s' a s. De esta forma tenemos que*

$$V' = \{s, v_1, v_2, v_3, v_4, r_1, r_2\} \cup \{s'\},$$
$$E' = \{1,2,3,4,5,6,7,8,9\} \cup \{e_1, e_2\}.$$

Para este ejemplo, el sistema de flujo viene dado por $\mathcal{F} = \{F_1, F_2\}$ *con*

$$F_1 = (e_1, 1, 5), (e_2, 2, 4, 6, 8),$$
$$F_2 = \{(e_1, 1, 3, 6, 9), (e_2, 2, 7)\},$$

donde F_1, F_2 *son el flujo asociado a* r_1 *y* r_2 *respectivamente en el problema ampliado. Comenzamos tomando los valores* $a_{1,1} = a_{2,2} = 1$, $a_{1,2} = a_{2,1} = 0$ *y* $f_{i,j} = 0$ *para los caminos* (i, j) *que no estén en* $\mathcal{F}$. *Además, como las aristas* $3, 4, 5, 7, 8, 9$ *son utilizadas por un único receptor, el algoritmo nos dice que* $f_{1,3} = f_{1,5} = f_{2,4} = f_{2,7} = f_{6,8} = f_{6,9} = 1$.

Inicialmente tenemos que $C_{r_1} = C_{r_2} = (e_1, e_2)$ *y* $B_{r_1} = B_{r_2} = ((1,0),(0,1))$. *Las actualizaciones siguiendo el orden ancestral en las aristas, de forma que vamos teniendo*

$$C_{r_1} = (e_1, e_2) \to (1, e_2) \to (1, 2) \to (1, 4) \to (5, 4) \to (5, 6) \to (5, 8),$$
$$C_{r_2} = (e_1, e_2) \to (1, e_2) \to (1, 2) \to (3, 2) \to (6, 2) \to (6, 7) \to (9, 7),$$

donde las aristas 5 y 8 son las aristas de $in(r_1) \cap F_1$ *y las aristas 9 y 7 son las aristas de* $in(r_2) \cap F_2$. *En este ejemplo, los únicos coeficientes a elegir aleatoriamente son* $f_{3,6}$ *y* $f_{4,6}$. *Tomando ambos coeficientes igual a 1 llegamos a una solución, pues, como podemos ver en la Figura 1.6, de esta forma ambos receptores reciben un conjunto de vectores que generan todo el espacio.*

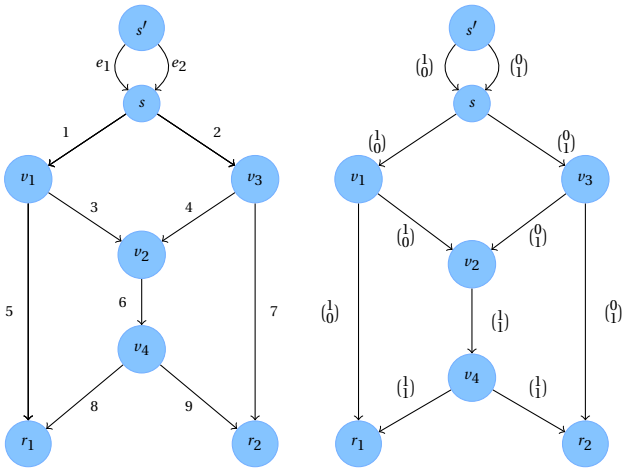

Figura 1.6: Vectores de codificación global sobre la red de mariposa.

Efectivamente vemos que a los receptores les llega $((1,0),(1,1))$ *y* $((1,1),(0,1))$, *que forman una base en* $\mathbb{F}_q^2$.

***Ejemplo* 1.2.** *En este segundo ejemplo, consideramos la Figura 1.7. Para calcular los coeficientes de codificación asociados a esta red hemos hecho uso del procedimiento Jaggi que podemos encontrar en la Sección 4.2 del Capítulo 4, donde el sistema de flujo $\mathcal{F}$ elegido es*

$$F_{r_1} = \{(1,3,5),(2,4,6)\},$$
$$F_{r_2} = \{(1,3,7),(2,4,8)\}.$$

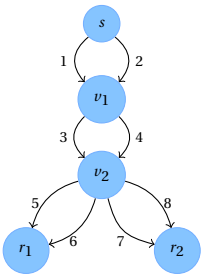

Figura 1.7: Red para el segundo ejemplo del Algoritmo de Jaggi

El algoritmo nos devuelve los coeficientes $f_{i,j}$ que podemos encontrar en la Sección 4.2 del Capítulo 4, y que vistos en forma de matriz, son

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

.

En caso de hacer uso del Algoritmo para una red en la que no se pueda obtener un conjunto de coeficientes de codificación que nos lleven a una comunicación satisfactoria, el Algoritmo entra en bucle a la hora de elegir los coeficientes de codificación, pues da igual cuales eliga que nunca va a conseguir que generen todo el espacio. Esto se podría solucionar haciendo que cuando llegue a un número de iteraciones, si el Algoritmo de Jaggi no ha encontrado una solución, devuelva error.

Capítulo 2

Random Network Coding

En este capítulo, cuando hablamos de random network coding hacemos referencia a la elección, a priori, de un subconjunto del conjunto de coeficientes de codificación y la posterior elección del resto de los coeficientes de manera aleatoria como en [10]. En otras referencias, como en [12], el uso del término random network coding hace referencia a otros temas.

Comenzamos considerando un problema de random network coding que se pueda resolver. Entonces, por el Corolario (1.1), sabemos que, para un q lo suficientemente grande, podemos elegir los coeficientes de codificación y decodificación de manera aleatoria tal que el problema se resuelva con alta probabilidad. Podemos elegir a priori $f_{i,j} = 1$ si tenemos que $in(j) = \{i\}$, y el resto de coeficientes los elegimos de manera aleatoria. De esta forma, dados unos ciertos coeficientes, los receptores pueden aprender, si es posible, como decodificar el mensaje.

Para ello, enviamos a través de la red un sistema de vectores (mensajes) de la forma

$$\overrightarrow{X} = (1,0,\ldots,0), \overrightarrow{X} = (0,1,0,\ldots,0),\ldots, \overrightarrow{X} = (0,\ldots,0,1).$$

De esta forma, el receptor solo tiene que ver qué información le llega por cada una de las aristas. La lista ordenada de información que le llega a cada arista de la red es lo que conocemos como vector de codificación global para esa arista. De esta forma, podemos elegir los coeficientes $b_{i,j}^{(r)}$ de forma que se resuelva el problema de random network coding si y solo si cada receptor recibe un vector de codificación global, a través de los vértices que llegan a él, que genere todo el espacio $\mathbb{F}_q^h$.

Al igual que sucede con el Algoritmo 1.1 (Algoritmo de Jaggi), necesitamos que la lista de vectores que llegan a los receptores genere todo el espacio $\mathbb{F}_q^h$, pues, de esta forma, a la hora de calcular los $b_{i,j}^{(r)}$, tenemos un sistema de rango máximo que podemos resolver, y que nos permite que ca-

da receptor pueda conocer los h mensajes que se envian por la red.

A continuación procederemos a hablar de la probabilidad P_{exito}, es decir, de la probabilidad de que los receptores puedan descifrar los mensajes tras haber elegido aleatoriamente los coeficientes de codificación.

2.1. Aproximación algebraica

Comenzamos considerando un problema de network coding que se pueda resolver en el que ya hemos elegido una serie de coeficientes de codificación a priori, de tal manera que formen parte de alguna solución de nuestro problema.

***Proposición* 2.1.** *Sea ν el número de coeficientes de codificación elegidos aleatoriamente en $\mathbb{F}_q$. Entonces,*

$$P_{\text{exito}} \geq P_{\text{Ho0}} := \left(\frac{q-|R|}{q}\right)^{\nu}, \qquad \text{para } q \geq |R|. \tag{2.1}$$

La notación usada P_{Ho0} y P_{Ho1}, P_{Ho2}, que utilizaremos más adelante, hace referencia a la notación empleada por los autores de [10].

Demostración. Consideramos el polinomio de transferencia $\prod_{r\in R} \det M^{(r)}$ en las variables $a_{i,j}$, $f_{i,j}$ y $b_{i,j}^{(r)}$. Damos valor a los coeficientes de codificación elegidos a priori y veamos los $b_{i,j}^{(r)}$ como constantes. Es decir, consideramos el polinomio de transferencia como un polinomio sobre $\mathbb{F}_q(Y_1,\ldots,Y_w)$, donde $Y_1,\ldots,Y_w$ se corresponden con los $b_{i,j}^{(r)}$. El polinomio escrito de esta manera es lo que llamamos polinomio de transferencia a priori. Tiene ν variables, y por el Lema 1.3 cada una tiene a lo sumo potencia $|R|$, luego, por el Corolario 1.1, podemos encontrar al menos $(q-|R|)^{\nu}$ de las q^{ν} posibles elecciones de coeficientes aleatorios de forma que el polinomio de transferencia tome valores no nulos. □

Ahora bien, si fijamos un orden monomial y sabemos que el monomio líder del polinomio de transferencia a priori es $X_1^{i_1}\cdots X_m^{i_m}$, entonces podemos deducir que

$$P_{\text{exito}} \geq P_{\text{FP2}} := \prod_{j=1}^{m} \frac{q-i_j}{q}. \tag{2.2}$$

Además, si no queremos saber cual es el término líder del polinomio de transferencia a priori, podemos aplicar una cota más débil

$$P_{\text{exito}} \geq P_{\text{FP1}} := \text{mín}\left\{ \prod_{j=1}^{m}\left(\frac{q-i_j}{q}\right) \middle| X_1^{i_1}\cdots X_m^{i_m} \begin{array}{l}\text{es un monomio}\\ \text{en el polinomio de transferencia}\end{array} \right\}. \tag{2.3}$$

La notación P_{FP1} y P_{FP2} hace referencia a la Footprint Bound (Proposición 1.2) y es obvio que $P_{\mathrm{FP1}} \leq P_{\mathrm{FP2}}$. El estimador P_{Ho0} no es muy bueno, pero tiene la ventaja de que es muy fácil de calcular, mientras que los estimativos P_{FP1} y P_{FP2} son mucho mejores aunque obtenerlos requiere más trabajo. A continuación veremos como podemos mejorar la cota de P_{Ho0} sin tener que complicar mucho los cálculos. Para ello, enunciamos primero el siguiente lema.

***Lema* 2.1.** *Sea $\mathbb{F}$ un cuerpo que contiene a $\mathbb{F}_q$. Sea $F(X_1,\ldots,X_m) \in \mathbb{F}[X_1,\ldots,X_m]$ un polinomio no nulo tal que todos los monomios $X_1^{j_1}\cdots X_m^{j_m}$ en el soporte de F satisfacen*

1. *$j_1,\ldots,j_m \leq d$, donde d es un entero tal que $d \leq q$,*
2. *$j_1+\cdots+j_m \leq dN$ para algún entero N fijado con $N \leq m$.*

Tomamos los valores $X_1,\ldots,X_m$ de manera aleatoria en $\mathbb{F}_q$, entonces la probabilidad de que $F(X_1,\ldots,X_m)=0$ es como mucho

$$1-\left(\frac{q-d}{q}\right)^N.$$

Demostración. Comenzamos tomando una variable X_1 en F y sea d_1 el mayor exponente de X_1 en F. Expresamos ahora F como $F = X_1^{d_1}F_1 + R_1$, donde F_1 es un polinomio de grado $dN-d_1$ a lo sumo que no contiene la variable X_1, y R_1 es un polinomio en el cual el mayor exponente de X_1 es menor que d_1. Haciendo uso del Principio de la decisión diferida (consular [18]), la probabilidad de que $F=0$ no se ve afectada si damos un valor a X_1 después de fijar todos los demás coeficientes. Si para alguna elección de los otros coeficientes, $F_1 \neq 0$, entonces F es un polinmio en $\mathbb{F}[X_1]$ de grado d_1.

Por el Teorema de Schwartz-Zippel (consultar [18]), tenemos que $P[F = 0|F_1 \neq 0] \leq d_1/q$. Entonces,

$$\begin{aligned} P[F=0] &\leq \frac{d_1}{q}P[F_1\neq 0] + P[FP_1 = 0] \\ &= \frac{d_1}{q}(1-P[F_1=0]) + P[F_1=0] \\ &= P[F_1=0]\left(1-\frac{d_1}{q}\right)+\frac{d_1}{q}. \end{aligned} \tag{2.4}$$

Ahora queremos acotar $P[F_1=0]$. Para ello elegimos una variable X_2 en F_1 siendo d_2 el mayor grado de X_2 en F_1. Al igual que antes, expresamos F_1 de la forma $F_1 = X_2^{d_2}F_2 + R_2$, donde F_2 es un polinomio de grado $dN-d_1-d_2$ que no contiene las variables X_1, X_2, y R_2 es un polinomio en el que el mayor exponente de X_2 es menor que d_2. Repetimos el razonamiento

tomando variables X_i, y definiendo d_i y F_i para $i = 3, 4, \ldots$ hasta llegar a un cierto $i = k$ para el cual F_k es constante y $P[F_k = 0] = 0$.

Tenemos en cuenta que $1 \leq d_i \leq d < q$ para todo i, y $\sum_{i=1}^{k} d_i \leq dN$, luego $k \leq dN$, pues suponiendo que todos los d_i toman el valor mínimo, tenemos que $\sum_{i=1}^{k} 1 = k \leq dN$.

Si aplicamos de nuevo el teorema de Schwartz-Zippel, tenemos que para $k' = 1, 2, \ldots, k$

$$P[F_{k'} = 0] \leq P[F_{k'+1} = 0]\left(1 - \frac{d_{k'+1}}{q}\right) + \frac{d_{k'+1}}{q}.$$

Combinando todas estas desigualdades podemos ver, por inducción, que

$$P[F = 0] \leq \frac{\sum_{i=1}^{k} d_i}{q} - \frac{\sum_{i \neq j} d_i d_j}{q^2} + \cdots + (-1)^{k-1} \frac{\prod_{i=1}^{k} d_i}{q^k}. \tag{2.5}$$

Entonces lo que tenemos es el siguiente problema de optimización:

$$\begin{aligned} &\text{Maximizar } f = \frac{\sum_{i=1}^{dN} d_i}{q} - \frac{\sum_{i \neq j} d_i d_j}{q^2} + \cdots + (-1)^{dN-1} \frac{\prod_{i=1}^{dN} d_i}{q^{dN}} \\ &\text{sujeto a } 0 \leq d_i \leq d < q \ \forall i \in [1, dN], \sum_{i=1}^{dN} d_i \leq dN, \text{ y } d_i \text{ entero,} \end{aligned} \tag{2.6}$$

cuyo máximo es una cota superior de $P[F = 0]$.

Comenzamos considerando el problema obtenido al rebajar la condición de que los d_i sean enteros. Sea $\mathbf{d}^* = \{d_1^*, \ldots, d_{dN}^*\}$ una solución óptima. Para cualquier conjunto θ_l formado por l enteros distintos en $[1, dN]$, sea

$$f = 1 - \frac{\sum_{i \in \theta_l} d_i}{q} + \frac{\sum_{i \in \theta_l, i \neq j} d_i d_j}{q^2} + \cdots + (-1)^l \frac{\prod_{i \in \theta_l} d_i}{q^l}.$$

Por inducción ahora sobre l, podemos ver que $0 < f_{\theta_l} < 1$ para cualquier conjunto θ_l de l enteros distintos de $[1, dN]$. Si $\sum_{i=1}^{dN} d_i^* < dN$, entonces tiene que haber algún $d_i^* < d$, y existe una solución $\mathbf{d} = \{d_1, \ldots, d_{dN}\}$ tal que $d_i = d_i^* + \epsilon$, con $\epsilon > 0$, y $d_l = d_l^*$ para $l \neq i$, que satisface

$$f(\mathbf{d}) - f(\mathbf{d}^*) = \frac{\epsilon}{q}\left(1 - \frac{\sum_{l \neq i} d_l^*}{q} + \cdots + (-1)^{dN-1} \frac{\prod_{l \neq i} d_l^*}{q^{dN-1}}\right). \tag{2.7}$$

Esta expresión es positiva, luego contradice la hipótesis de que $\mathbf{d}^*$ es óptimo, luego $\sum_{i=1}^{dN} d_i^* = dN$. Supongamos ahora que $0 < d_i^* < d$ para algún d_i^*, entonces existe algún d_j^* tal que $0 < d_j^* < d$, pues si $d_j^* = 0$ o $d_j^* = d$ para todo j, entonces $\sum_{i=1}^{dN} d_i^* \neq dN$. Asumimos, sin pérdida de generalidad, que $0 < d_i^* \leq d_j^* < d$, entonces existe un vector $\mathbf{d} = \{d_1, \ldots, d_{dN}\}$ tal que

$d_i = d_i^* - \epsilon$, $d_j = d_j^* + \epsilon$, $\epsilon > 0$, y $d_l = d_l^*$ para todo $l \neq i, j$, que cumple

$$f(\mathbf{d}) - f(\mathbf{d}^*) = -\left(\frac{(d_i^* - d_j^*)\epsilon - \epsilon^2}{q^2}\right)\left(1 - \frac{\sum_{l\neq i,j} d_l^*}{q} + \cdots + (-1)^{dN-2}\frac{\prod_{l\neq i,j} d_l^*}{q^{dN-2}}\right), \tag{2.8}$$

que es positivo de nuevo, lo que contradice la hipótesis de que $\mathbf{d}^*$ era minimal.

Por lo tanto, $\sum_{i=1}^{dN} d_i^* = dN$, con $d_i^* = 0$ o $d_i^* = d$. Luego exactamente N de las d_i^* variables toman el valor d. Como la solución óptima es una solución entera, es solución del problema del problema (2.6). La solución es

$$f = N\frac{d}{q} - \binom{N}{2}\frac{d^2}{q^2} + \cdots + (-1)^{N-1}\frac{d^N}{q^N} = 1 - \left(1 - \frac{d}{q}\right)^N. \tag{2.9}$$

□

***Teorema* 2.1.** *Consideramos un problema de random network coding en el que alguno (o todos) de los coeficientes $\{a_{i,j}, f_{l,j}\}$ se eligen aleatoriamente del cuerpo finito $\mathbb{F}_q$, con $q > d$, y el resto de los coeficientes vienen fijados a priori. Si los coeficientes de codificación fijados a priori forman parte de alguna solución, entonces la probabilidad de que los coeficientes elegidos aleatoriamente nos proporcionen la solución es*

$$P_{\text{exito}} \geq P_{\text{Ho2}} := \left(\frac{q - |R|}{q}\right)^{\nu'}, \tag{2.10}$$

donde ν' es el número máximo de aristas con coeficientes aleatorios asociados a un flujo para cualquiera de los receptores. El número de coeficientes de codificación en el monomio asociado al flujo es ν'.

Además, como el número de aristas en un flujo es como mucho el número de aristas en nuestra red, tenemos una cota más pequeña,

$$P_{\text{exito}} \geq P_{\text{Ho1}} := \left(\frac{q - |R|}{q}\right)^{\nu''}, \tag{2.11}$$

donde ν'' es el número de aristas $j \in E$ tal que alguno de los $a_{i,j}$ o $f_{l,j}$ se han elegido de manera aleatoria.

Demostración. Sabemos que, por el Lema 1.2, un network code (una elección de coeficientes $a_{i,j}$ y $f_{l,j}$) es válido si, para cada receptor r, tenemos una combinación lineal de productos $a_{x_1,l_1}\cdots a_{x_h,l_h} f_{i_1,l_{h+1}}\cdots f_{i_{\nu_r},l_{h+\nu_r}}$, donde $\{l_1, \ldots, l_{h+\nu_b}\}$ forma un flujo a r, que es no nula. El producto de las correspondientes expresiones para d receptores tiene grado menor o igual que $d\nu'$, donde $\nu' = \max_r \nu_r$, y el mayor exponente de cualquiera de las variables es como mucho d. Por el Lema 2.1 llegamos al resultado.

La cota para P_{Ho1} es trivial pues es claro que $P_{\text{Ho1}} \leq P_{\text{Ho2}} \leq P_{\text{exito}}$. □

***Teorema* 2.2.** *La probabilidad P_{exito} satisface*

$$P_{\text{Ho0}} \leq P_{\text{Ho1}} \leq P_{\text{Ho2}} \leq P_{\text{FP1}} \leq P_{\text{FP2}} \leq P_{\text{exito}}. \tag{2.12}$$

Demostración. Queremos ver que $P_{\text{Ho2}} \leq P_{\text{FP1}}$, pues el resto de las desigualdades son triviales por como las hemos ido definiendo.

Recordamos que

$$P_{\text{Ho2}} = \left(\frac{q-|R|}{q}\right)^{\nu'},$$

$$P_{\text{FP1}} = \text{mín}\left\{ \prod_{j=1}^{m} \left(\frac{q-i_j}{q}\right) \middle| X_1^{i_1} \cdots X_m^{i_m} \begin{array}{l} \text{es un monomio} \\ \text{en el polinomio de transferencia} \end{array} \right\}.$$

Asumimos que $q \geq |R|$ pues sino la cota dada por P_{Ho2} sería trivial, y obviamente tendríamos la desigualdad. Tenemos en cuenta que $i_j \leq |R| \leq q$ para todo j pues todas las variables tienen como grado máximo el número de receptores por el Lema 1.3. Además, tenemos que $\sum_j i_j \leq |R|\nu'$ con la definición de ν' dada en la página anterior.

Queremos estudiar ahora mín$\prod_{j=1}^{m}\left(\frac{q-i_j}{q}\right)$ sobre todos los $i_1,\ldots,i_m$ tal que $i_j \leq |R|$ para todo j y tal que $\sum_j i_j \leq |R|\nu'$ donde $\nu' \leq m$. Entonces, si queremos llegar al mínimo, tenemos que $\sum_j i_j = |R|\nu'$ y asumimos, sin pérdida de generalidad, que $i_1 \leq i_2 \leq \cdots \leq i_m$. Además, si tenemos en cuenta que para $i \leq i'$,

$$(q-i)(q-i') > (q-(i+1))(q-(i'-1)), \tag{2.14}$$

tendríamos que $i_1 = \cdots = i_{\nu'} = |R|$ y $i_{\nu'+1} = \cdots = i_m = 0$, en cuyo caso $P_{\text{Ho2}} = P_{\text{FP1}}$. □

***Ejemplo* 2.1.** *Tomamos ahora como ejemplo la red de la Figura 1.5. En este ejemplo teníamos un emisor s y dos receptores r_1 y r_2. Suponemos que los tres mensajes se generan en s, y como tenemos que para cada receptor existe un flujo de tamaño 3, el problema de network coding se puede resolver. Elegimos a priori los coeficientes $a_{1,1} = a_{2,2} = a_{3,3} = 1$ y $a_{1,2} = a_{1,3} = a_{2,3} = a_{2,1} = a_{3,1} = a_{3,2} = 0$.*

Elegimos también a priori los coeficientes

$$f_{1,4} = f_{1,6} = f_{2,7} = f_{2,8} = f_{3,5} = f_{3,9} = 1$$
$$f_{6,12} = f_{7,10} = f_{8,11} = f_{9,15} = f_{10,13} = f_{10,14} = 0,$$

(los coeficientes que tomamos iguales a 0 es porque no forman parte de ningún flujo de nuestro sistema de flujo) y el resto de los coeficientes los elegiremos de manera arbitraria. El polinomio de transferencia a priori es

$$F = (abcdef)Q, \tag{2.15}$$

donde

$$a = f_{6,10} \quad b = f_{7,12} \quad c = f_{8,15}$$
$$d = f_{9,11} \quad e = f_{10,14} \quad f = f_{11,13}$$

y $Q = \det\tilde{B}^{(r_1)}\det\tilde{B}^{(r_2)}$. *Las matrices* $\tilde{B}^{(r_1)}$ *y* $\tilde{B}^{(r_2)}$ *consisten en las filas no nulas de* $B^{(r_1)}$ *y* $B^{(r_2)}$ *respectivamente. Si nos restringimos ahora a cuerpos* $\mathbb{F}_q$ *de tamaño al menos 4 podemos aplicar las cotas que estudiamos anteriormente. Considerando el único monomio en F, tenemos que*

$$P_{\text{FP1}} = \frac{(q-1)^6}{q^6}. \tag{2.16}$$

En este ejemplo no es necesario considerar un orden para las variables, pues al tener un único monomio,

$$P_{\text{FP2}} = P_{\text{FP1}} = \frac{(q-1)^6}{q^6}. \tag{2.17}$$

Además,

$$P_{\text{Ho2}} = \frac{(q-2)^3}{q^3}. \tag{2.18}$$

Vemos que P_{FP2} *es mayor que* P_{Ho2}, *obviamente lo mismo sucede para* P_{FP1}. *En la Tabla 2.1 vemos los valores de* P_{FP2}, P_{FP1} *y* P_{Ho2} *para distintas elecciones de q.Estás probabilidades tienden hacia 1 cuando hacemos crecer q.*

Tabla 2.1: Estimativos de la probabilidad de éxito

q	4	8	16	32	64
P_{FP2}	0,178	0,449	0,679	0,827	0,910
P_{FP1}	0,178	0,449	0,679	0,827	0,910
P_{Ho2}	0,125	0,421	0,670	0,824	0,909

Ahora vamos a considerar el cuerpo $\mathbb{F}_2$. *En este cuerpo no podemos tomar las cotas* P_{Ho0}, P_{Ho1} *y* P_{Ho2}, *pues no tenemos que* $q > |R|$.

Primero tomamos el polinomio de transferencia a priori módulo $(a^2 - a, \ldots, f^2 - f)$, *que en este caso es el mismo polinomio. Si aplicamos la cota* P_{FP2}, *tenemos que la probabilidad de éxito es al menos* 2^{-6}, *que es una cota muy baja.*

***Ejemplo* 2.2.** *Consideramos ahora la red que se muestra en la Figura 2.1 (obtenida de [9]). En esta red, el emisor s envía los mensajes* X_1, X_2 *hasta los receptores* r_1, r_2. *De nuevo aplicaremos network coding. Este ejemplo los cuerpos con los que trabajaremos serán cuerpos de característica 2. Elegimos a priori los coeficientes* $a_{1,1} = a_{2,2} = 1$ *y* $a_{1,2} = a_{2,1} = 0$.

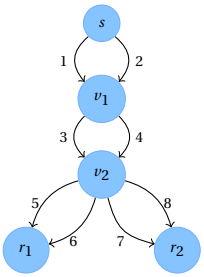

Figura 2.1: Red para el Ejemplo 2.2.

En este ejemplo, no tomamos ningún $f_{i,j}$ a priori. Por lo tanto nuestro polinomio de transferencia es

$$\begin{aligned} F =& (-2abcdegjl + 2abcdehjk + 2abcdfgil - 2abcdfhik \\ &+ b^2c^2egjl - b^2c^2ehjk - b^2c^2fgil + b^2c^2fhik \\ &+ a^2d^2egjl - a^2d^2ehjk - a^2d^2fgil + a^2d^2fhik)Q, \end{aligned} \tag{2.19}$$

pero como trabajamos, para este ejemplo, con cuerpos con característica 2, tenemos que

$$\begin{aligned} F =& (b^2c^2egjl - b^2c^2ehjk - b^2c^2fgil + b^2c^2fhik \\ &+ a^2d^2egjl - a^2d^2ehjk - a^2d^2fgil + a^2d^2fhik)Q, \end{aligned} \tag{2.20}$$

donde

$$\begin{array}{lllll} a = f_{1,3} & b = f_{1,4} & c = f_{2,3} & d = f_{2,4} & e = f_{3,5} \quad f = f_{3,6} \\ g = f_{3,7} & h = f_{3,8} & i = f_{4,5} & j = f_{4,6} & k = f_{4,7} \quad l = f_{4,8} \end{array}$$

y $Q = \det\tilde{B}^{(r_1)}\det\tilde{B}^{(r_2)}$. Las matrices $\tilde{B}^{(r_1)}$ y $\tilde{B}^{(r_2)}$ consisten en las filas no nulas de $B^{(r_1)}$ y $B^{(r_2)}$ respectivamente. Si nos restrigimos ahora a cuerpos $\mathbb{F}_q$ de tamaño al menos 4 podemos aplicar las cotas que estudiamos anteriormente.

Considerando como orden el orden lexicográfico $\prec$ con

$$l \prec k \prec j \prec i \prec h \prec g \prec f \prec e \prec d \prec c \prec b \prec a,$$

tenemos que el monomio líder de F es a^2d^2egjl, y por tanto

$$P_{\mathrm{FP2}} = \frac{(q-1)^4}{q^4}\frac{(q-2)^2}{q^2}. \tag{2.21}$$

En este ejemplo, considerando todos los monomios, tenemos que

$$P_{\mathrm{FP1}} = \frac{(q-1)^4}{q^4}\frac{(q-2)^2}{q^2}. \tag{2.22}$$

Tabla 2.2: Estimativos de la probabilidad de éxito

q	4	8	16	32	64
P_{FP2}	$0{,}791 \times 10^{-1}$	0,330	0,591	0,774	0,881
P_{FP1}	$0{,}791 \times 10^{-1}$	0,330	0,591	0,774	0,881
P_{Ho2}	$0{,}625 \times 10^{-1}$	0,316	0,586	0,772	0,880

Además,

$$P_{\text{Ho2}} = \frac{(q-2)^4}{q^4}. \tag{2.23}$$

Vemos que P_{FP2} es mayor que P_{Ho2}, y lo mismo sucede para P_{FP1}. En la Tabla 2.2 vemos los valores de P_{FP2}, P_{FP1} y P_{Ho2} para distintas elecciones de q.

Ahora vamos a considerar el cuerpo $\mathbb{F}_2$. En este cuerpo no podemos tomar las cotas P_{Ho0}, P_{Ho1} y P_{Ho2}, pues no tenemos que $q > |R|$.

Primero tomamos el polinomio de transferencia a priori módulo $(a^2 - a, \ldots, l^2 - l)$, obteniendo así

$$\begin{aligned}\hat{F} =& (bcegjl - bcehjk - bcfgil + bcfhik \\ &+ adegjl - adehjk - adfgil + adfhik)Q,\end{aligned} \tag{2.24}$$

De nuevo, haciendo uso de la probabilidad P_{FP2}, tenemos que la probabilidad de éxito es al menos 2^{-6}. Como sucedía en el ejemplo anterior, es muy baja.

Está claro que para poder obtener las cotas P_{FP1} y P_{FP2} es necesario calcular el polinomio de transferencia, lo que requiere un alto coste operativo cuando hay varios sistemas de flujo para la red, pues si hay uno único no requiere ningún esfuerzo. Esto nos lleva a cuestionarnos si, dado un sistema de flujo de tamaño h para nuestra red, existe un orden monomial para el cual el monomio correspondiente a este sistema de flujo sea el monomio líder, pues de esta forma obtendríamos automáticamente la cota P_{FP2} sin la necesidad de calcular el polinomio de transferencia.

Esto no tiene por qué suceder, pues como veíamos en el Ejemplo 2.2, existen monomios correspondientes a sistemas de flujo que se cancelan en el cálculo de $\prod_{r\in R} \det M^{(r)}$.

Además, también puede pasar que no exista ningún orden monomial para el cual el monomio asociado al sistema de flujo elegido sea el monomio líder de nuestro polinomio de transferencia. Para ilustrar esto mejor volvemos al Ejemplo 2.2.

En este caso supondremos que la característica de $\mathbb{F}_q$ es distinta de 2,

luego el polinomio de transferencia a priori contiene a los monomios

$$a^2 d^f hik, \tag{2.25a}$$
$$b^2 c^2 fhik, \tag{2.25b}$$
$$abcdfhik. \tag{2.25c}$$

Sabemos que un orden monomial $\prec$ es un buen orden si para toda terna monomios A, B, C, se tiene que si $A \prec B$, entonces $AC \prec BC$. Sea $\prec$ un orden monomial, entonces se cumple que

$$bcfhik \prec adfhik \tag{2.26a}$$
$$o$$
$$bcfhik \succ adfhik. \tag{2.26b}$$

Asumimos primero que nos encontramos en el caso (2.26a). Si multiplicamos ambos lados de (2.26a) por ad tendríamos (2.25c) en el lado izquierdo y (2.25a) en el derecho, luego (2.25c) no puede ser el monomio líder del polinomio de transferencia.

Supongamos ahora que tenemos (2.26b). Si multiplicamos ambos lados de (2.26b) por bc tendríamos (2.25b) en el lado izquierdo y (2.25c) en el derecho. Luego (2.25c) nunca va a ser el monomio líder del polinomio de transferencia a priori.

2.2. Cotas de flujo

En esta sección, para obtener una cota de la probabilidad de éxito, haremos uso del Algoritmo de Jaggi (Algoritmo 1.1). Previamente lo modifaremos, pues, en este caso, en lugar de encontrar un flujo para la red, asumiremos que nos viene ya dado. Además, para cada paso, consideramos la primera elección de coeficientes de codificación aleatorios de forma que si nos proporcionan una solución el algoritmo sigue funcionando. En caso contrario, el algoritmo para y devuelve fallo. Si el algoritmo no falla concluye devolviendo éxito. Éste es el Algoritmo 2.1 (consultar [21]), donde q es el tamaño del cuerpo.

En el Algoritmo 2.1 hacemos uso de nueva notación como $R_{\mathcal{F}}(e)$, que es el conjunto de receptores que usan e en $\mathcal{F}$, es decir,

$$R_{\mathcal{F}}(e) = \{r \in R : \text{ hay un camino de } e \text{ a } r \text{ en } \mathcal{F}\}. \tag{2.27}$$

También, dado un sistema de flujo $\mathcal{F}$ y una arista $e \in \mathcal{F}$, denotamos por $f^r_{\leftarrow}(e)$ a la arista anterior a e en el camino que forma parte del flujo F_r.

***Algoritmo* 2.1.**

Input: *Un problema de multicast netowork coding que se pueda resolver. Un sistema de flujo $\mathcal{F}$ para el problema.*

Output: *Éxito, junto con los coeficientes de codificación, si la elección de coeficientes de codificación nos proporciona una solución y error en caso contrario.*

Inicialización

Sea $V' := V \cup \{s'\}$ *y* $E' := E \cup \{e_1, \ldots, e_h\}$.

for $r \in R$

$C_r := \{e_1, \ldots, e_h\}$;

$B_r := c_g(C_r)$;

endfor

Elección aleatoria de los vectores de codificación global y comprobación de si la elección nos lleva a una solución.

for $e \in \mathcal{F}$ *en el orden ancestral*

if $e \notin \mathcal{F}$ ***then***

$f_{p,e} = 0$ *para todo* $p \in in(e)$;

$c_g(e) = 0$;

elseif $|R_{\mathcal{F}}| = 1$ ***then***

$f_{p,e} = 1$ *si* $p \in f^r_{\leftarrow}(e)$, *0 si no;*

$c_g(e) = c_g(f^r_{\leftarrow}(e))$;

else

Elegimos los coeficientes de codificación aleatoriamente;

if $c_g(e) \in \text{span}\{B_r - c_g(f^r_{\leftarrow}(e))\}$ *para todo* $r \in R_{\mathcal{F}}(e)$ ***then***

return *error;*

end

end

for $r \in R_{\mathcal{F}}(e)$

$C_r := C_r - f^r_{\leftarrow}(e) + e$;

$B_r := c_g(C_r)$;

endfor

endfor

return *éxito junto con los coeficientes de codificación*

Este Algoritmo es una modificación del Algoritmo 1.1, pues éste Algoritmo no calcula el sistema de flujo, sino que lo damos nosotros mismos como input. La parte de inicialización coincide, y los coeficientes a priori son obtenidos de la misma manera. La otra diferencia que supone este algoritmo respecto del Algoritmo 1.1 es que en el Algoritmo 1.1 tomabamos los coeficientes de codificación aleatoriamente hasta que éstos nos llevaran a una solución, mientras que en Algoritmo 2.1 tomamos una única elección de coeficientes de codificación aleatoriamente, y únicamente los devolvemos si nos llevan a una solución; en caso contrario, devolvemos error.

El siguiente lema que utilizaremos para probar las cotas relacionadas con el Algoritmo 2.1 es una formalización del Lema 1.4 en el marco del random network coding.

***Lema* 2.2.** *Tenemos un problema de network coding for multicast, sea $\mathcal{F}$ un sistema de flujo y sea e una arista de la red. Asumimos que corremos el algoritmo de Jaggi modificado (Algoritmo 2.1) y sea $P_{\text{error}}(e)$ la probabilidad de que el algoritmo devuelva fallo cuando llega a la arista e suponiendo que no ha devuelto error en las aristas anteriores. Entonces,*

$$P_{\text{error}}(e) \leq \frac{|R_{\mathcal{F}}(e)|q^{|in'(e)|} - |R_{\mathcal{F}}(e)| + 1}{q^{|in'(e)|}} \leq \frac{|R_{\mathcal{F}}(e)|}{q}, \tag{2.28}$$

donde $|in'(e)|$ es $|in(e)|$ junto con el número de mensajes generados en s, con $e = (s, v)$.

Demostración. Por el Lema 1.4 sabemos que dados todos los coeficientes de codificación excepto $f_{f^r_{\leftarrow}(e),e}$, hay, a lo sumo, una elección de $f_{f^r_{\leftarrow}(e),e}$ que no nos lleva a una solución.

Para establecer la cota, tenemos en cuenta que, en el peor de los casos, para todos los coeficientes de codificación , hay a lo sumo $|R_{\mathcal{F}}(e)|q^{|in'(e)|-1}$ elecciones que no nos llevan a una solución. Como tenemos un total de $q^{|in'(e)|}$ posibles elecciones, tenemos la cota buscada.

Para la segunda cota, hay que tener en cuenta que en realidad hay menos elecciones que no nos llevan a una solución, pues estamos contando la elección de que todos los coeficientes de codificación sean 0 un total de $|R_{\mathcal{F}}(e)|$ veces en lugar de una única vez, llegando así a la cota más fina. □

Haciendo uso de la cota obtenida en el lema anterior, obtenemos las probabilidades de éxito de la elección de coeficientes de manera aleatoria para nuestro problema de random network coding.

***Proposición* 2.2.** *Para un problema de random network coding for multicast en el que los coeficientes de codificación se eligen aleatoriamente en $\mathbb{F}_q$, tenemos que*

$$P_{\text{exito}} \geq \max_{\mathcal{F}} \left\{ \prod_{e \in \mathcal{F}} \frac{q - |R_{\mathcal{F}}(e)|}{q} \right\}, \tag{2.29}$$

donde $\mathcal{F}$ es un sistema de flujo.

Demostración. La probabilidad de éxito es al menos la probabilidad de que el algoritmo de Jaggi modificado (Algoritmo 2.1) devuelva éxito para cualquier sistema de flujo $\mathcal{F}$. Por lo tanto,

$$P_{\text{exito}} \geq \max_{\mathcal{F}} \left\{ P(\text{Alg 2.1 devuelva éxito}) \right\}. \tag{2.30}$$

Además, sabemos que el Algoritmo 2.1 devuelve éxito si y solo si para toda arista $e \in \mathcal{F}$, para todo receptor $r \in R_{\mathcal{F}}(e)$, la dimensión del espacio generado por B_r es h.

Por lo tanto, tenemos que

$$P(\text{Alg. 2.1 devuelva éxito}) = \prod_{e\in\mathcal{F}} \prod_{r\in R_{\mathcal{F}}(e)} P\left[c_q(e) \notin \text{span}B_r^e | \text{span}B_r = \mathbb{F}^h\right], \tag{2.31}$$

donde $B_r^e = B_r - c_g(f_{\leftarrow}^r(e))$. Haciendo ahora uso del Lema 2.2, tenemos que

$$P(\text{Alg. 2.1 devuelva éxito}) \geq \prod_{e\in\mathcal{F}} \frac{q - |R_{\mathcal{F}}(e)|}{q}, \tag{2.32}$$

y de esta forma tenemos el resultado que queríamos. □

Ahora haremos uso de la cota más fina obtenida en el Lema 2.2 para obtener lo siguiente.

***Proposición* 2.3.** *Para un problema de network coding for multicast en el que elegimos los coeficientes de codificación de manera aleatoria en el cuerpo finito $\mathbb{F}_q$, la probabilidad de éxito es al menos*

$$P_{\text{exito}} \geq \max_{\mathcal{F}} \left\{ \prod_{e\in\mathcal{F}} \left(1 - \frac{|R_{\mathcal{F}}(e)|}{q} + \frac{|R_{\mathcal{F}}(e)| - 1}{q^{|in'(e)|}}\right)\right\}. \tag{2.33}$$

A través de estas dos proposiciones hemos definido dos nuevas cotas para la probabilidad de éxito de nuestro problema de random network coding dado un cierto sistema de flujo $\mathcal{F}$:

$$P_{\text{exito}} \geq P_{\text{FB1}} := \prod_{e\in\mathcal{F}} \frac{q - |R_{\mathcal{F}}(e)|}{q}, \tag{2.34}$$

$$P_{\text{exito}} \geq P_{\text{FB2}} := \prod_{e\in\mathcal{F}} \left(\frac{q - |R_{\mathcal{F}}(e)|}{q} + \frac{|R_{\mathcal{F}}(e)| - 1}{q^{|in'(e)|}}\right). \tag{2.35}$$

Nuestro objetivo ahora es mejorar las cotas obtenidas en (2.34) y (2.35).

Para ello, consideramos dos receptores $r_1, r_2 \in R_{\mathcal{F}}(e)$ y $f_{\leftarrow}^{r_1}(e) \in C_{r_1}$, $f_{\leftarrow}^{r_2}(e) \in C_{r_2}$. Asumimos que $\text{span}B_{r_1}^e = \text{span}B_{r_2}^e$, entonces tenemos que $c_g(e) \notin \text{span}B_{r_1}^e$ si y solo si $c_g(e) \notin \text{span}B_{r_2}^e$. Podríamos utilizar $|\{\text{span}B_r^e : r \in R_{\mathcal{F}}(e)\}|$, donde recordamos que $B_r^e = B_r - c_g(f_{\leftarrow}^r(e))$, pero es más práctico considerar $|\{C_r^e : r \in R_{\mathcal{F}}(e)\}|$, donde tenemos que $C_r^e = C_r - f_{\leftarrow}^r(e)$. Además, es obvio que $|\{C_r^e : r \in R_{\mathcal{F}}(e)\}| \geq |\{\text{span}B_r^e : r \in R_{\mathcal{F}}(e)\}|$. Usando la notación $r_{\mathcal{F}}(e) = |\{C_r^e : r \in R_{\mathcal{F}}(e)\}|$, tenemos la siguiente proposición:

***Proposición* 2.4.** *Dado un problema de network coding, sea $\mathcal{F}$ un sistema de flujo y sea e una arista cualquiera de la red. Si ejecutamos el algoritmo de Jaggi modificado (Algoritmo 2.1), y sea $P_{\text{exito}}(e)$ la probabilidad de que el algoritmo no devuelva error cuando realiza una actualización, condicionada a que no devolvió error anteriormente, tenemos que*

$$P_{\text{exito}}(e) \geq \frac{q - r_{\mathcal{F}}(e)}{q}. \tag{2.36}$$

Ahora, haciendo uso de la idea de la Proposición 2.3, sustituyendo $|R_{\mathcal{F}}(e)|$ por $r_{\mathcal{F}}(e)$, obtenemos las dos siguientes nuevas cotas:

$$P_{\text{exito}} \geq P_{\text{FB3}} := \prod_{e \in \mathcal{F}} \left(\frac{q - r_{\mathcal{F}}}{q} \right), \tag{2.37}$$

$$P_{\text{exito}} \geq P_{\text{FB4}} := \prod_{e \in \mathcal{F}} \left(\frac{q - r_{\mathcal{F}}}{q} - \frac{r_{\mathcal{F}} - 1}{q^{|in'(e)|}} \right). \tag{2.38}$$

Es claro que, por construcción,

$$P_{\text{exito}} \geq P_{\text{FB4}} \geq P_{\text{FB3}} \geq P_{\text{FB2}} \geq P_{\text{FB1}}. \tag{2.39}$$

Hay que tener en cuenta además que $r_{\mathcal{F}}$ no solo depende del sistema de flujo elegido sino también del orden ancestral en el que vamos recorriendo los distintos vértices de la red.

Para ilustrar esto, veremos un ejemplo en el que a partir de la misma red podemos obtener dos cotas de la probabilidad de éxito distintas en función del orden ancestral elegido sobre las aristas.

***Ejemplo* 2.3.** *Sea s el emisor que envía dos mensajes X_1, X_2 a través de la red de la Figura 2.2, y sean r_1, r_2 los dos receptores. Vamos a considerar dos órdenes de aristas distintos y buscaremos cotas para la probabilidad de éxito haciendo uso de la Proposición 2.4.*

El sistema de flujo asociado a la Figura 2.2 es $\mathcal{F} = \{F_{r_1}, F_{r_2}\}$, con

$$F_{r_1} = \left\{(e_1, e_3), (e_2, e', e_5)\right\}, \quad F_{r_2} = \left\{(e_1, e_4), (e_2, e', e_6)\right\}.$$

Consideramos el primer orden dado por

$$e_1 \prec e_2 \prec e' \prec e_3 \prec e_4 \prec e_5 \prec e_6,$$

tenemos que, inicialmente, $C_{r_1} = C_{r_2} = \{e_1, e_2\}$, y asumimos que $\dim(spanB_{r_1})$ $= \dim(spanB_{r_2}) = 2$. *Siguiendo ahora el orden establecido, reemplazamos e_2 por e' y tenemos que $r_{\mathcal{F}}(e') = 1$, luego $1 - 1/q$ es la cota de la probabilidad de que $c_g(e')$ sea linealmente independiente en $B_{r_1}^{e_1}$ y $B_{r_2}^{e_1}$. Los dos siguientes pasos son ahora reemplazar e_1 por e_3 y e_4 respectivamente. En ambos casos*

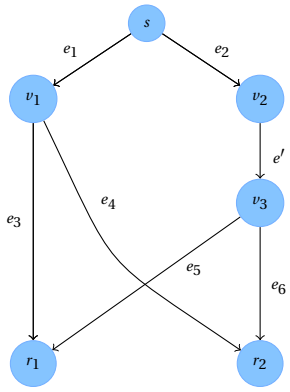

Figura 2.2: Red en la que la probabilidad de éxito varía en función del orden elegido.

tenemos que $r_{\mathcal{F}}(e_3) = r_{\mathcal{F}}(e_4) = 1$, *luego de nuevo obtenemos la cota* $1 - 1/q$. *Por último reemplazamos* e' *por* e_5 *y* e_6 *en dos pasos. De nuevo tenemos que* $r_{\mathcal{F}}(e_5) = r_{\mathcal{F}}(e_6) = 1$, *por lo que ambos pasos nos llevan a la cota* $1 - 1/q$. *Por lo tanto, en total obtenemos la cota* $(1 - 1/q)^5$.

En este ejemplo estamos trabajando sobre la red original y no la red ampliada como haríamos para ejecutar el algoritmo 2.1, pero nos interesa únicamente ver como llegamos a dos cotas distintas.

Ahora consideramos un segundo orden en las aristas dado por

$$e_1 \prec e_2 \prec e_3 \prec e_4 \prec e' \prec e_5 \prec e_6.$$

Al igual que antes, comenzamos con $C_{r_1} = C_{r_2} = \{e_1, e_2\}$, *y, de nuevo, asumimos que* $\dim(spanB_{r_1}) = \dim(spanB_{r_2}) = 2$. *Primero reemplazamos* e_1 *por* e_3 *y* e_4 *en dos pasos. Como* $r_{\mathcal{F}}(e_3) = r_{\mathcal{F}}(e_4) = 1$, *ambos pasos nos llevan a la cota* $1 - 1/q$. *Tras estos dos pasos, tenemos que* $C_{r_1} = \{e_3, e_2\}$, $C_{r_2} = \{e_4, e_2\}$, *entonces, a la hora de reemplazar* e_2 *por* e', *tenemos* $r_{\mathcal{F}}(e') = |\{C_{r_1} - e_2, C_{r_2} - e_2\}| = |\{\{e_3\}, \{e_4\}\}| = 2$, *luego tenemos la cota* $1 - 2/q$. *Por último, reemplazamos* e' *por* e_5 *y* e_6 *en dos pasos. Como* $r_{\mathcal{F}}(e_5) = r_{\mathcal{F}}(e_6) = 1$, *tenemos que la cota viene dada por* $1 - 1/q$ *en ambos pasos. Luego con este orden de aristas, obtenemos la cota* $(1 - 1/q)^4(1 - 2/q)$. *Con lo que efectivamente las probabilidades dadas por* (2.37) *y* (2.38) *dependen del orden de aristas elegido.*

Por último, para concluir con esta sección, vamos a calcular sobre un ejemplo todas las cotas estudiadas en la misma.

***Ejemplo* 2.4.** *Consideramos la Figura 2.3, que consiste en la red ampliada de la Figura 1.5.*

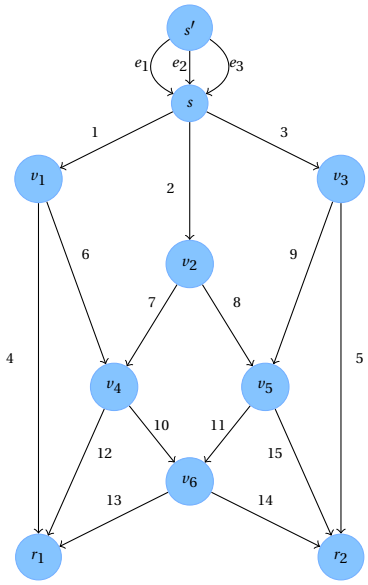

Figura 2.3: Red modificada.

En ella, tenemos que $\mathcal{F} = \{F_{r_1}, F_{r_2}\}$, *con*

$$F_{r_1} = \{(e_1,1,4),(e_2,2,7,12),(e_3,3,9,11,13)\},$$
$$F_{r_2} = \{(e_1,1,6,10,14),(e_2,2,8,15),(e_3,3,5)\}.$$

Para este ejemplo, suponemos que todos los coeficientes de codificación son elegidos aleatoriamente, y consideramos el orden dado por

$$1 \prec 2 \prec 3 \prec 6 \prec 7 \prec 8 \prec 9 \prec 10 \prec 11 \prec 4 \prec 5 \prec 12 \prec 15 \prec 13 \prec 14.$$

Vamos calculando para cada arista e $|R_{\mathcal{F}}(e)|$, $|in'(e)|$ *y* $r_{\mathcal{F}}(e)$. *Inicialmente tenemos que* $C_{r_1} = C_{r_2} = \{e_1, e_2, e_3\}$.

Ahora vamos recorriendo las distintas aristas para cada uno de los flujos que componen el sistema de flujos. Reemplazamos e_1 *por 1, de esta forma tenemos que* $|R_{\mathcal{F}}(1)| = 2$, $|in'(1)| = 3$ *y* $r_{\mathcal{F}}(1) = 1$. *Lo mismo sucede al reemplazar* e_2 *por 2 y* e_3 *por 3. Tras estos tres pasos, tenemos que* $C_{r_1} = C_{r_2} = \{1,2,3\}$. *A continuación, reemplazamos la arista 1 por la arista 6, en este caso tenemos que* $|R_{\mathcal{F}}(6)| = 1$, $|in'(6)| = 1$ *y* $r_{\mathcal{F}}(6) = |\{C_r - f^r_{\leftarrow}(e) : r \in R_{\mathcal{F}}(6)\}| = 1$. *Exactamente lo mismo ocurre cuando realizamos las sustituciones 2 por 7 y 8(en dos pasos) y 3 por 9. Ahora lo que tenemos es* $C_{r_1} = \{1,7,9\}$ *y* $C_{r_2} = \{6,8,3\}$. *Los siguientes dos pasos serán sustituir 6 por 10 y 9 por 11. En ambos pasos tenemos que* $|R_{\mathcal{F}}(10)| = |R_{\mathcal{F}}(11)| = 1$, $|in'(10)| = |in'(11)| = 2$ *y* $r_{\mathcal{F}}(10) = |\{\{8,3\}\}| = 1$, $r_{\mathcal{F}}(11) = |\{\{1,7\}\}| = 1$.

A continuación, siguiendo el orden de aristas establecido, sustituimos 1 por 4. Tenemos que $|R_{\mathcal{F}}(4)| = 1$, $|in'(4)| = 1$ *y* $r_{\mathcal{F}}(4) = 1$, *lo mismo sucede al sustituir 3 por 5 por como es la red.*

Si sustituimos ahora 7 por 12 y 8 por 15 en dos pasos, tenemos que $|R_{\mathcal{F}}(12)| = |R_{\mathcal{F}}(15)| = 1$, $|in'(12)| = |in'(15)| = 2$ *y* $r_{\mathcal{F}}(12) = r_{\mathcal{F}}(15) = 1$. *Finalmente, concluimos sustituyendo 11 por 13 y 10 por 14, donde tenemos que* $|R_{\mathcal{F}}(13)| = |R_{\mathcal{F}}(14)| = 1$, $|in'(13)| = |in'(14)| = 2$ *y* $r_{\mathcal{F}}(13) = r_{\mathcal{F}}(14) = 1$. *Por lo tanto*

$$P_{FB1} = \left(\frac{q-1}{q}\right)^{12}\left(\frac{q-2}{q}\right)^{3},$$
$$P_{FB2} = \left(\frac{q-1}{q}\right)^{12}\left(\frac{q-2}{q}+\frac{1}{q^3}\right)^{3},$$
$$P_{FB3} = P_{FB4} = \left(\frac{q-1}{q}\right)^{15}.$$

Para finalizar el ejemplo, construimos una tabla con distintos valores de q y vemos que valores van tomando las distintas probabilidades.

Tabla 2.3: Estimativos de la probabilidad de éxito

q	4	8	16	32	64
P_{FB1}	$0{,}4\times10^{-2}$	$0{,}85\times10^{-1}$	0,3088	0,5629	0,753
P_{FB2}	$0{,}43\times10^{-2}$	$0{,}856\times10^{-1}$	0,3091	0,5630	0,753
P_{FB3}	$0{,}134\times10^{-1}$	0,1349	0,3798	0,6211	0,7896
P_{FB4}	$0{,}134\times10^{-1}$	0,1349	0,3798	0,6211	0,7896

2.2.1. Comparación entre las cotas de flujo obtenidas y las cotas P_{Ho} y P_{FP}

En esta subsección vamos a comparar las nuevas cotas obtenidas con las cotas obtenidas en las secciones anteriores. En concreto compararemos P_{FB1} con P_{Ho2} y P_{FP2}, además de dar un ejemplo en el que $P_{\text{FP2}} < P_{\text{FB2}}$.

***Proposición* 2.5.** *Para un problema de network coding que se pueda resolver, tenemos que* $P_{Ho2} \leq P_{FB1}$.

Demostración. Comenzamos fijando una arista e y renombramos los coeficientes de codificación $a_{j,e}$ y $f_{j',e}$ en $X_{i_1},\ldots,X_{i_m}$. Sea k_t el número de veces que aparece la variable X_{i_t} en el monomio dado por el sistema de flujo $\mathcal{F}$. Por definición, tenemos que

$$\sum_{t=1}^{m} k_t = |R_{\mathcal{F}}(e)|. \tag{2.42}$$

Identificamos el monomio $X_{i_1}^{k_1}\cdots X_{i_m}^{k_m}$ con $\mathbf{X}_e^{|R_{\mathcal{F}}(e)|}$. De esta forma, el monomio que se corresponde con el sistema de flujo está definido como un

monomio en las variables $\mathbf{X}_e$ con $e \in \mathcal{F}$. Si consideramos un polinomio de transferencia en el que su monomio líder sea $\prod_{e\in\mathcal{F}} \mathbf{X}_e^{|R_{\mathcal{F}}(e)|}$, entonces, haciendo uso del Corolario 1.1, tenemos la cota

$$\prod_{e\in\mathcal{F}} \left(\frac{q - |R_{\mathcal{F}}(e)|}{q} \right) = P_{\text{FB1}}.$$

Además, si tomamos el número ν' definido en la página 33 y haciendo uso de la demostración del Teorema 2.2, tenemos que $P_{\text{Ho2}} \leq P_{\text{FB1}}$. □

Ahora queremos ver que sucede con P_{FB1} y P_{FP2}.

Dada una arista e de la red, renombramos los coeficientes de codificación $a_{u,e}$ y $f_{u',e}$ (para todo u, u') como $X_1, \ldots, X_k$. Sea i_j la potencia de X_j en el monomio de $\prod_{r\in R} \det E^{(r)}$ (el polinomio de transferencia) que se corresponde con el sistema de flujo $\mathcal{F}$. De nuevo, por construcción, tenemos que $\sum_{j=1}^{k} i_j = |R_{\mathcal{F}}(e)|$. Por inducción es sencillo ver que

$$\frac{q - |R_{\mathcal{F}}(e)|}{q} \leq \prod_{j=1}^{k} \frac{q - i_j}{q}. \tag{2.43}$$

Entonces, tomando el producto a ambos lados de (2.43) para cada arista $e \in \mathcal{F}$, el lado izquierdo de la desigualdad nos conduce a P_{FB1} y el lado derecho a P_{FP2} si existe un orden monomial para el cual el monomio elegido es el monomio líder del polinomio de transferencia.

Parece a priori que la cota dada por P_{FP2} es en general mejor que las cotas obtenidas a través del flujo en la sección anterior, aunque el problema sigue residiendo en si es posible o no que un monomio del polinomio de transferencia sea el monomio líder para un orden de aristas $\prec$ concreto.

Como vimos en el Ejemplo 2.2, esto no es siempre posible. Además, no está claro si el resto de cotas de flujo nos dan una mejor estimación de la probabilidad de éxito que P_{FP2}.

A continuación veremos un ejemplo en el que P_{FB2} es mejor que la estimación obtenida por P_{FP2}.

***Ejemplo* 2.5.** *Consideramos la red dada por la Figura 2.4. Sean $f_{1,3}$, $f_{2,3}$ los únicos coeficientes de codificación obtenidos de manera aleatoria y asumimos que los vectores de codificación global en $out(s)$ son linealmente independientes cuando consideramos los pares* $(0,1)$ *y* $(0,2)$.

Los dos sistemas de flujo de la red son

$$\mathcal{F}_1 = \{\{(0,4),(1,3,7)\},\{(0,6),(1,3,5)\}\},$$
$$\mathcal{F}_2 = \{\{(0,4),(2,3,7)\},\{(0,6),(2,3,5)\}\}.$$

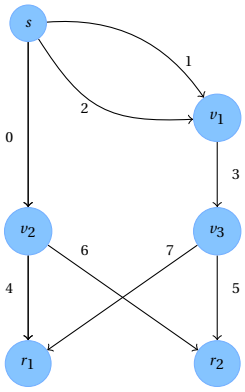

Figura 2.4: Red en la que P_{FB2} es mayor que P_{FP2}.

Luego el polinomio de transferencia asociado a la red tendrá los monomios $f^2_{1,3}$ *y* $f^2_{2,3}$*, según el orden monomial elegido uno u otro será el monomio líder (no es relevante).*

Entonces,

$$P_{FP2} = P_{Ho2} = P_{FB1} = \frac{q-2}{q}, \tag{2.45}$$

pues para cualquiera de los dos sistemas de flujo, tenemos que $R_{\mathcal{F}}(3) = 2$ *y solo tenemos un coeficiente que calcular para nuestro sistema de flujo. Además, como* $|in'(3)| = 2$*, tenemos que* $P_{FB2} = \frac{q-2}{q} + \frac{1}{q^2}$*. Por último,* $r_{\mathcal{F}}(3) = 1$*, pues para ambos sistemas de flujo,* $C^3_{r_1} = C^3_{r_2} = \{0\}$*. Por lo tanto* $P_{FB3} = P_{FB4} = \frac{q-1}{q}$*.*

En efecto, en este ejemplo

$$P_{FP2} < P_{FB2} < P_{FB3} = P_{FB4}.$$

2.3. Obtención de cotas a través de los vértices del flujo

En esta sección, haciendo uso de una idea similar basada en cómo hemos obtenido las cotas de flujo, probaremos una cota de la probabilidad de éxito, que podemos encontrar en [2].

En [2] no se tiene en cuenta el caso en el que $out(r) \neq \emptyset$ para un receptor r. Además, los problemas de network coding planteados tienen un único receptor, cosa que a nosotros no nos importa.

En la sección anterior ibamos recorriendo las aristas de la red de forma que los pasos del algoritmo actualizaban la arista $f^r_{\leftarrow}(e)$ por e y trataban de acotar la probabilidad de que $c_g(e) \in \text{span}B^e_r$ para, de esta forma, obtener cotas de la probabilidad de éxito. En este caso, el proceso lo modificamos

pues vamos a trabajar sobre los vértices en lugar de trabajar sobre las aristas. Para determinar un orden ancestral sobre los vértices, asumimos un orden sobre las aristas, pues de esta forma automáticamente nos dará un orden en los vértices. Tenemos que $v < v'$ si existe un camino de v a v'. Más formalmente, si tenemos un cierto orden

$$(v_{i_1}, v_{j_1}) < \cdots < (v_{i_{|E|}}, v_{j_{|E|}})$$

para las aristas, el orden para los vértices V vendrá determinado por el orden de las aristas de forma que elegimos la primera vez que aparece cada vértice v_{i_t} en ese orden.

Ahora vamos a introducir un poco de notación para esta parte. Sea $v \in V$, definimos

$$\begin{aligned} R_{\mathcal{F}}(v) &= \cup_{e \in in(v)} R_{\mathcal{F}}(e), \\ f^r_{\leftarrow}(v) &= \{e \in in(v) : r \in R_{\mathcal{F}}(e)\}, \\ f^r_{\rightarrow}(v) &= \{e \in out(v) : r \in R_{\mathcal{F}}(e)\}. \end{aligned}$$

Con estás definiciones, es claro que $|f^r_{\leftarrow}(v)| = |f^r_{\rightarrow}(v)|$, pues por cada arista $e_i \in f^r_{\leftarrow}(v)$, existe una arista $e'_i \in f^r_{\rightarrow}(v)$ tal que si $e_i \neq e_j$, entonces $e'_i \neq e'_j$. Esto se debe a que en el flujo F_r asociado al receptor r, los caminos que lo conforman son disjuntos.

***Algoritmo* 2.2.**

Input: *Un problema de multicast network coding que se pueda resolver. Un sistema de flujo $\mathcal{F}$ para el problema. Un conjunto de coeficientes de codificación para la red.*

Output: *Éxito si la elección de coeficientes de codificación nos proporciona una solución y error en caso contrario.*

Inicialización

Consideramos el problema con la red ampliada, extendemos el sistema de flujo acorde a la red ampliada y cambiamos los coeficientes de codificación acorde a esto.

Sea $V' := V \cup \{s'\}$ *y* $E' := E \cup \{e_1, \ldots, e_h\}$.

for $r \in R$

$C_r := \{e_1, \ldots, e_h\}$;

$B_r := c_g(C_r)$;

endfor

Comprobación de los vectores de codificación global.

for $v \in V$ *en el orden ancestral*

Calcular $c_g(e)$ *para todo* $e \in out(v)$ *basado en los coeficientes de codificación;*

for $r \in R_{\mathcal{F}}(v)$

$C_r := C_r - f^r_{\leftarrow}(v) + f^r_{\rightarrow}(v)$;
$B_r := c_g(C_r)$;
endfor
if *dim span*$B_r \neq h$ *para algún* $r \in R_{\mathcal{F}}(v)$ **then**
return *error*
end
endfor
return *éxito*

La versión alternativa del algoritmo de Jaggi [21] con la que trabajaremos en esta sección la podemos ver en el Algoritmo 2.2. El algoritmo, dado un conjunto de coeficientes de codificación para un sistema de flujo, comprueba si se llega a una solución. En caso de que sea así, devuelve éxito, y, en caso contrario, devuelve error. De nuevo, C_r es un corte en el flujo F_r de $\mathcal{F}$ asociado al receptor r y B_r es el conjunto de vectores de codificación global de las aristas en C_r.

Al contrario que en los algoritmos anteriores, en los que en cada paso se actualizaba una única arista de C_r, al estar trabajando sobre los vértices de la red, en cada paso se pueden modificar multiples aristas en C_r.

La probabilidad que nos interesa calcular es

$$P(v) := P\left[\begin{array}{l}\dim \text{span}B_r = h\ \forall r \in R_{\mathcal{F}}(v) \text{ después de la modificación} \\ \dim \text{span}B_r = h\ \forall r \in R_{\mathcal{F}}(v) \text{ antes de la modificación}\end{array}\right]. \tag{2.47}$$

Para ello, primero comenzamos considerando un único receptor, es decir,

$$P(v) := P\left[\begin{array}{l}\dim \text{span}B_r = h \text{ después de la modificación} \\ \dim \text{span}B_r = h \text{ antes de la modificación}\end{array}\right]. \tag{2.48}$$

Antes vimos que $|f^r_{\leftarrow}(v)| = |f^r_{\rightarrow}(v)|$. Sea $n = |f^r_{\leftarrow}(v)| = |f^r_{\rightarrow}(v)|$, entonces tenemos la siguiente cota:

$$P_r(v) \geq \prod_{i=0}^{n-1}\left(\frac{q^n - q^i}{q^n}\right). \tag{2.49}$$

A continuación veremos como probar esta nueva cota.

Sea B^D_r el conjunto de vectores de codificación global de las aristas en $C^D_r = C_r \setminus D$, es decir, $B^D_r = \cap_{e\in D\cup C_r} B^e_r$, donde $B^e_r = B_r - c_g(e)$ al igual que lo definimos anteriormente. Además, sea $e \in out(v)$, definimos

$$x_r(e) = \sum_{p\in in(v)-f^r_{\leftarrow}(v)} f_{p,e}c_g(p) \tag{2.50}$$

como la contribución de $c_g(e)$al vector de codificación global, donde e son las aristas que llegan a v que no forman parte de $F_r \in \mathcal{F}$.

Fijamos un receptor $r \in R_{\mathcal{F}}(r)$ y sea i_1 una arista que sale de nuestro vértice v. Sea $f_{\leftarrow}^r(v) = \{p_{k+1}, \dots, p_h\}$, sea $B_r^{\{p_{k+1},\dots,p_h\}} = \{b_1, \dots, b_k\}$, y sea $c_g\left(f_{\leftarrow}^r(v)\right) = \{b_{k+1,\dots,b_h}\}$. Como B_r es una base de dimensión h del espacio, podemos escribir

$$x_r(i_1) = x_1 b_1 + \cdots + x_k b_k + x_{k+1} b_{k+1} + \cdots + x_h b_h \tag{2.51}$$

para unos ciertos x_j. Al igual que sucedía en el Lema 1.4, existe una única elección de coeficientes de codificación de $f_{\leftarrow}^r(v)$ en i_1 tal que $c_g(i_1) \in \text{span}B_r^{f_{\leftarrow}^r(v)}$; $f_{p_{k+1},i_1} = -x_{k+1}, \dots, f_{p_h,i_1} = -x_h$.

Como hay q^n elecciones, donde $n = h - k$ y solo una de ellas nos lleva a que $c_g(i_1) \in \text{span}B_r^{f_{\leftarrow}^r(v)}$, tenemos que la probabilidad de que $c_g(i_1) \notin \text{span}B_r^{f_{\leftarrow}^r(v)}$ es $\frac{q^n-1}{q}$.

Ahora, fijado $c_g(i_1)$ tal que $c_g(i_1) \notin \text{span}B_r^{f_{\leftarrow}^r(v)}$, queremos calcular la probabilidad de que

$$c_g(i_2) \notin \text{span}\left(B_r^{f_{\leftarrow}^r(v)} \cup \{c_g(i_1)\}\right), \tag{2.52}$$

o, de manera más general, asumiendo que $\mathcal{B} = B_r^{f_{\leftarrow}^r(v)} \cup \{c_g(i_1), \dots, c_g(i_t)\}$ es linealmente independiente, queremos calcular la probabilidad de que

$$c_g(i_{t+1}) \notin \text{span}\mathcal{B}. \tag{2.53}$$

***Lema* 2.3.** *Sean k, h, t naturales con $1 \leq k < h$ y $k + t < h$. Sea $\{\boldsymbol{b}_1, \dots, \boldsymbol{b}_h\}$ una base de $\mathbb{F}_q^h$ y sean $\boldsymbol{b}'_{k+1}, \dots, \boldsymbol{b}'_{k+t}$ tal que*

$$V = span\{\boldsymbol{b}_1, \dots, \boldsymbol{b}_k, \boldsymbol{b}'_{k+1}, \dots, \boldsymbol{b}'_{k+t}\} \tag{2.54}$$

tiene dimensión $k+t$. Dado $\boldsymbol{c} \in \mathbb{F}_q^h$, el número de posibles elecciones de $(a_{k+1}, \dots, a_h)$, $a_i \in \mathbb{F}_q$ tal que

$$\boldsymbol{c} + a_{k+1}\boldsymbol{b}_{k+1} + \cdots + a_h \boldsymbol{b}_h \in V \tag{2.55}$$

es q^t.

Asumimos que el Algoritmo 2.2 funciona correctamente hasta el paso en el que visitamos el vértice w, es decir, para cada receptor r, B_r genera todo $\mathbb{F}_q^h$. Consideramos para un receptor en particular los conjuntos $f_{\leftarrow}^r(w)$ y $f_{\rightarrow}^r(w)$. Si $w = r$ no hay que modificar nada y tendríamos que $f_{\leftarrow}^r(w) = f_{\rightarrow}^r(w) = \emptyset$. Asumimos que $f_{\rightarrow}^r(w) \neq \emptyset$, entonces, por el Lema 2.3, la probabilidad de que los coeficientes de codificación relacionados con w hagan que B_r genere todo $\mathbb{F}_q^h$ vienen dadas por

$$P_r(v) \geq \prod_{i=1}^{|f_{\rightarrow}^r(w)|} \frac{q^{|f_{\rightarrow}^r(w)|} - q^{i-1}}{q^{|f_{\rightarrow}^r(w)|}} = \prod_{i=0}^{n-1} \frac{q^n - q^i}{q^n}. \tag{2.56}$$

Por [2], tenemos una cota más sencilla de calcular:

$$P_r(v) \geq \prod_{i=0}^{n-1} \frac{q^n - q^i}{q^n} = \prod_{i=0}^{n-1} \left(1 - \frac{1}{q^{n-i}}\right) \geq 1 - \frac{1}{q-1}, \tag{2.57}$$

donde la última desigualdad se prueba fácilmente por inducción, teniendo en cuenta que $\left(1 - \frac{1}{q}\right) \geq \left(1 - \frac{1}{q-1}\right)$ y $\prod_{i=0}^{n-1} \left(1 - \frac{1}{q^{n+1-i}}\right) \geq \prod_{i=0}^{n-1} \left(1 - \frac{1}{q^{n-i}}\right)$ para $q \geq 2$.

Además, si hacemos tender q a infinito, la diferencia entre el lado izquierdo y el derecho de la desigualdad tiende a 0. Vemos que la probabilidad de error viene acotada por $1/(1-q)$. Además, en el peor de los casos, esta probabilidad de error se aplica a todos los receptores, luego tenemos que

$$\begin{aligned} P(v) &\geq 1 - \sum_{r \in R_{\mathcal{F}}(v)} \left(1 - \prod_{i=0}^{|f^r_{\rightarrow}(v)|-1} \frac{q^{|f^r_{\rightarrow}(v)|} - q^i}{q^{|f^r_{\rightarrow}(v)|}}\right) \\ &> 1 - \frac{|R_{\mathcal{F}}(v)|}{q-1}. \end{aligned} \tag{2.58}$$

Como esto lo tenemos sobre cada vértice, tenemos

$$P_{\text{exito}} > \prod_{v \in V_I} \left(1 - \frac{|R_{\mathcal{F}}(v)|}{q-1}\right) \geq \left(1 - \frac{|R_{\mathcal{F}}(v)|}{q-1}\right)^{|V_I \cap \mathcal{F}|} \geq \left(1 - \frac{|R_{\mathcal{F}}(v)|}{q-1}\right)^{|V_I|}, \tag{2.59}$$

donde $V_I = \{v \in V : out(v) \neq \emptyset\}$; o, de manera similar,

$$P_{\text{exito}} \geq P_{\text{FB'}} := \prod_{v \in V_I} \left(1 - \sum_{r \in R_{\mathcal{F}}(v)} \left(1 - \prod_{i=0}^{|f^r_{\rightarrow}(v)|-1} \frac{q^{|f^r_{\rightarrow}(v)|} - q^i}{q^{|f^r_{\rightarrow}(v)|}}\right)\right). \tag{2.60}$$

Hay que tener en cuenta que si un vértice v no está en el sistema de flujo, $n(r) = 0$ y no hay ningún r tal que $r \in R_{\mathcal{F}}(v)$, luego podemos sumar sobre todo el conjunto de vértices. Además, la cota menos estricta obtenida en (2.59) es muy similar a la cota obtenida en [2], por lo que la denotamos como

$$P_{\text{Balli}} := \left(1 - \frac{|R|}{q-1}\right)^{|V_I|}. \tag{2.61}$$

La diferencia es que en [2] únicamente se consideran redes con un solo emisor y tampoco se tiene en cuenta la posibilidad de que un receptor tenga aristas que salgan de él.

2.3.1. Comparación de P_{Balli} con el resto de cotas

A lo largo del capítulo hemos ido definiendo diversidad de cotas: las cotas P_{FP1} y P_{FP2}, relacionadas con los monomios del polinomio de transferencia; las cotas de flujo P_{FB1}, P_{FB2}, P_{FB3} y P_{FB4} [9]; la cota P_{Balli} [2] y la cota $P_{\text{FB'}}$ que acabamos de definir; y las cotas P_{Ho0}, P_{Ho1} y P_{Ho2} [10].

Ya sabemos que, por el Teorema 2.2, $P_{\text{Ho2}} \leq P_{\text{FP1}} \leq P_{\text{FP2}}$. Además, $P_{\text{FB1}} \leq P_{\text{FB2}} \leq P_{\text{FB3}} \leq P_{\text{FB4}}$ por como han sido construidas. En el siguiente ejemplo veremos como P_{Balli} y $P_{\text{FB'}}$ son cotas que no se pueden comparar con P_{FB1}.

***Ejemplo* 2.6.** *De nuevo, volvemos a la red representada en la Figura 2.3. Teníamos que*

$$P_{FB1} = \left(\frac{q-1}{q}\right)^{12}\left(\frac{q-2}{q}\right)^{3},$$
$$P_{FB2} = \left(\frac{q-1}{q}\right)^{12}\left(\frac{q-2}{q}+\frac{1}{q^3}\right)^{3},$$
$$P_{FB3} = P_{FB4} = \left(\frac{q-1}{q}\right)^{15}.$$

Procedemos ahora a calcular P_{Balli}. Sabiendo que

$$V_I = \{v \in V : out(v) \neq \emptyset\} = \{s, v_1, v_2, v_3, v_4, v_5, v_6\}$$

tenemos que $P_{Balli} = \left(1-\frac{2}{q-1}\right)^7$. Ahora, para calcular $P_{FB'}$,

$$\begin{aligned}
P_{FB'} = &\left(1-\sum_{r\in R_{\mathcal{F}}(s)}\left(1-\prod_{i=0}^{|f^r_{\rightarrow}(s)|-1}\frac{q^{|f^r_{\rightarrow}(s)|}-q^i}{q^{|f^r_{\rightarrow}(s)|}}\right)\right)\\
&\left(1-\sum_{r\in R_{\mathcal{F}}(v_1)}\left(1-\prod_{i=0}^{|f^r_{\rightarrow}(v_1)|-1}\frac{q^{|f^r_{\rightarrow}(v_1)|}-q^i}{q^{|f^r_{\rightarrow}(v_1)|}}\right)\right)^3\\
&\left(1-\sum_{r\in R_{\mathcal{F}}(v_4)}\left(1-\prod_{i=0}^{|f^r_{\rightarrow}(v_4)|-1}\frac{q^{|f^r_{\rightarrow}(v_4)|}-q^i}{q^{|f^r_{\rightarrow}(v_4)|}}\right)\right)^2\\
&\left(1-\sum_{r\in R_{\mathcal{F}}(v_6)}\left(1-\prod_{i=0}^{|f^r_{\rightarrow}(v_6)|-1}\frac{q^{|f^r_{\rightarrow}(v_6)|}-q^i}{q^{|f^r_{\rightarrow}(v_6)|}}\right)\right)\\
= &\left(1-2\left(1-\frac{q^3-q^0}{q^3}\frac{q^3-q^1}{q^3}\frac{q^3-q^2}{q^3}\right)\right)\left(1-2\left(1-\frac{q-1}{q}\right)\right)^6,
\end{aligned}$$

pues tenemos que $|f^r_{\rightarrow}(s)| = 3$ para todo r y $|f^r_{\rightarrow}(v_i)| = 1$ para $i = 1,\ldots,6$. Tomando ahora cuerpos con distinto número de elementos, construimos la siguiente tabla:

Tabla 2.4: Estimativos de la probabilidad de éxito

q	2	3	4	8	16	32	64
$P_{\text{FB'}}$	0	$0{,}1938 \times 10^{-3}$	$0{,}6 \times 10^{-2}$	0,128	0,389	0,635	0,800
P_{Balli}	0	0	$0{,}457 \times 10^{-3}$	$0{,}95 \times 10^{-1}$	0,367	0,627	0,798
P_{FB1}	0	$0{,}286 \times 10^{-3}$	$0{,}4 \times 10^{-2}$	$0{,}85 \times 10^{-1}$	0,3088	0,5629	0,753
P_{FB2}	$0{,}477 \times 10^{-6}$	$0{,}392 \times 10^{-3}$	$0{,}43 \times 10^{-2}$	$0{,}856 \times 10^{-1}$	0,3091	0,5630	0,753

Efectivamente, para $q = 2$, las tres cotas son triviales; para $q = 3$, la cota P_{Balli} sigue siendo trivial, mientras que $P_{FB1} > P_{FB'}$; para $q = 4$, tenemos que $P_{FB'} > P_{FB1} > P_{Balli}$ mientras que para $q > 8$, tenemos que $P_{FB'} > P_{Balli} > P_{FB1}$. Por lo tanto, no son comparables. En [21], tenemos un ejemplo en el que para $q > 4$ tenemos ya esta última cadena de desigualdades.

En cuanto a las cotas obtenidas en este capítulo, todas ellas son aceptables, como hemos visto en los distintos ejemplos, pues tienden a 1 cuando hacemos crecer q. Las probabilidades P_{Ho0}, P_{Ho1} y P_{Ho2}, aunque son las menos fiables, son las más sencillas de obtener, y como $P_{\text{Ho2}} \geq P_{\text{Ho1}} \geq P_{\text{Ho0}}$, podemos obtener P_{Ho2}, y si es lo suficientemente buena podríamos llegar a omitir el cálculo de mejores cotas. Para redes en las que haya un único sistema de flujo, el polinomio de transferencia estará formado por un único monomio, y por tanto será el monomio líder sin importar el orden. Además, este monomio será muy sencillo de calcular, pues estará formado por el producto de todos los coeficientes de codificación $f_{i,j}$, con (i, j) un camino, formado por las aristas i y j, que forma parte del flujo. Por lo tanto, para este tipo de redes, será extremadamente sencillo obtener la cota P_{PF2}. En cuanto a las cotas tipo P_{FB}, necesitamos un sistema de flujo para calcularlas. Las dos primeras, P_{FB1} y P_{FB2}, si bien son peores que las dos siguientes, son sencillas de calcular, pues únicamente necesitamos conocer el número de receptores que tienen la arista e en su flujo y el número $|in'(e)|$, para cada arista del grafo. Las cotas P_{FB3} y P_{FB4} son algo más complejas de calcular, y también dependen del sistema de flujo elegido, por lo que, visto en los ejemplos que, en general, las cotas P_{FB1} y P_{FB2} son lo suficientemente buenas, no merece la pena tanto su cálculo, salvo para casos en los que las otras cotas nos den probabilidades inferiores a las deseadas.

Por último, en cuanto a las cotas obtenidas en la última sección, ya vimos que no son comparables con cotas como por ejemplo P_{FB1}. Además, la cota $P_{\text{FB'}}$, puede llegar a ser algo tediosa de calcular, y a priori, solo estaremos interesados en conocerla para redes en las que el número de vértices sea bastante menor que el número de aristas, pues esta cota trabaja sobre los vértices de la red, mientras que el resto de cotas lo hacen sobre las aristas. Finalmente, la cota de Balli P_{Balli}, aunque tampoco es comparable con el resto de cotas, es sencilla de calcular, ya que solo necesitamos conocer el número de receptores y el de vértices tal que salgan de ellos al menos una arista.

En la práctica, elegiremos un cuerpo lo suficientemente grande, pero preferentemente no muy grande y usualmente con un número de elementos que sea una potencia de 2 o un cuerpo donde se pueda tener una implementación optimizada de su aritmética, y elegiremos un conjunto de coeficientes aleatorios sobre ese cuerpo. Si estos coeficientes nos llevan a una solución, ya tenemos resuelto el problema, y, en caso contrario, elegiremos de nuevo los coeficientes aleatoriamente, ya que, como hemos ido viendo a lo largo de este capítulo, las probabilidades de éxito para nuestra elección son bastante altas, por lo que acabaremos consiguiendo una elección que nos lleve a una solución.

Capítulo 3

Corrección de errores en Network Coding

En el capítulo anterior, vimos cómo transimitir a través de una red un conjunto de mensajes, en forma de vectores, hasta una serie de receptores. El emisor enviaba una base a través de la red, y los receptores eran capaces de decodificar los mensajes si el conjunto de receptores que recibían eran otra base de ese mismo espacio, pues eso significaba que, resolviendo un sistema lineal de ecuaciones, podían recuperar la información enviada. El problema es que, si durante la transmisión, se introduce un vector que no forma del espacio generado por los vectores transmitidos, al irse formando combinaciones lineales de estos vectores a lo largo de la red, la comunicación se volvería completamente inútil. Este suceso es a lo que llamaremos errores. También podría pasar que se perdiera alguno de los mensajes durante la transmisión, es decir, que no llegara al receptor, haciendo que éste no reciba la cantidad suficiente de información como para poder decodificarla. Ésto es lo que llamaremos borrones. Los errores y borrones los podemos ver, desde el punto de vista de la criptografía, como un enemigo saboteando nuestra comunicación, interfiriendo en un nodo y haciendo que se pierdan paquetes (borrón) o introduciendo paquetes sin sentido (error).

En este capítulo, veremos cómo tratar con estos dos problemas. Para ello, consideraremos la información un espacio vectorial. De esta forma, el emisor envía un conjunto generador del espacio (o base). Los receptores obtienen un conjunto de elementos que generan otro espacio vectorial, de forma que si no se producen errores ni borrones, éste coincidirá con el enviado, y sino, será un espacio vectorial distinto.

3.1. Canales de transmisión

En está sección introducimos el concepto de canal de transmisión de una manera similar a como definíamos la transmisión para random lineal network coding, donde tanto el emisor como el receptor desconocen la estructura de la red por la que se está enviando la información.

Comenzamos considerando un problema de network coding para un único emisor y un único receptor. La comunicación entre ambos individuos sucede en una serie de "generaciones". En cada generación, el emisor manda una serie de paquetes de longitud fija, que podemos ver como vectores fila de longitud N sobre un cuerpo finito $\mathbb{F}_q$, a través de los nodos intermedios que componen la red. Cada uno de los nodos genera una combinación lineal aleatoria, sobre $\mathbb{F}_q$, de los paquetes que le llegan y la transmite a los siguientes nodos.

De esta forma, el receptor obtiene una serie de paquetes generados arbitrariamente e intenta averiguar cuales eran los paquetes originales que se habían enviado por la red. En este caso no asumimos que la red sea acíclica o que sea no secuencial.

Al igual que sucede con la forma clásica de codificar la información, la información la comenzamos viendo como vectores de longitud k a los que les añadimos una cierta redundancia para convertirlos en vectores de longitud N.

Sean $\{p_1,\dots,p_M\}$, con $p_i \in \mathbb{F}_q^N$, el conjunto de vectores que se envían por la red. Si ésta no contiene errores, el receptor obtiene paquetes y_j, $j = 1,2,\dots,L$, con

$$y_j = \sum_{i=1}^{M} h_{j,i} p_i, \tag{3.1}$$

como sucedía en la página 10 del Capítulo 1, donde los $h_{j,i} \in \mathbb{F}_q$ son elegidos aleatoriamente. A priori L no está fijado, y el receptor recoge todos los paquetes que puede. De cualquier forma, las propiedades de la red como, por ejemplo, el mínimo corte entre el emisor y el receptor, pueden influenciar en la distribución de los $h_{j,i}$ [13], de manera que llegue un punto en el que obtener más información no sea beneficioso.

Si consideramos que se añaden T paquetes con errores, denotados por e_t, con $t = 1,\dots,T$, en vez de tener (3.1), obtendríamos

$$y_j = \sum_{i=1}^{M} h_{j,i} p_i + \sum_{t=1}^{T} g_{j,t} e_t, \tag{3.2}$$

donde $g_{j,t} \in \mathbb{F}_q$.

El hecho de que se puedan producir en cualquier punto de la red hace que se puedan propagar. De hecho, si $g_{j,1} \neq 0$ para todo j, un solo error en

una arista puede corromper toda la red. En forma matricial, tenemos que

$$y = Hp + Ge, \tag{3.3}$$

donde H y G son matrices aleatorias de tamaño $L \times M$ y $L \times T$ respectivamente, p es una matriz de tamaño $M \times N$ cuyas filas son los vectores transmitidos por la red, y es una matriz $L \times N$ cuyas filas son los vectores recibidos que contienen errores y e es una matriz $T \times N$ cuyas filas son los errores introducidos en la red. Además, aunque H y G sean matrices aleatorias, la topología de la red impone ciertas condiciones sobre éstas, como vimos en el primer capítulo.

La cuestión es qué sucede con p, visto como una matriz, al premultiplicarlo por la matriz aleatoria H. Como vemos en (3.1), lo que se producen son combinaciones lineales de las filas de p, por lo que Hp será una matriz que puede tener el mismo rango que p o menor. En este caso, si vemos p como un espacio vectorial, Hp es un subespacio de p cuya dimensión es el rango de la matriz Hp.

Sea W un espacio vectorial de dimensión N sobre $\mathbb{F}_q$, para un N fijo. Todos los paquetes transmitidos entre el emisor y el receptor serán vectores de W. Por lo tanto, el conjunto de estos paquetes será un conjunto de vectores de W, que generarán un subespacio vectorial de W. Buscaremos entonces definir un modelo en función de los subespacios de W que se generan a través de los paquetes p_i.

Denotamos por $\mathcal{P}(W)$ al conjunto de todos los subespacios de W. Denotamos por $\dim(V)$ la dimensión de un elemento $V \in \mathcal{P}(W)$, es decir, la dimensión del subespacio. Dados dos subespacios $U, V \in \mathcal{P}(W)$, definimos $U + V = \{u + v : u \in U, v \in V\}$, es decir, el mínimo subespacio que contiene a ambos. Además, si $U \cap V = \{0\}$, la suma la denotamos como suma directa $U \oplus V$, con $\dim(u \oplus V) = \dim(U) + \dim(V)$. Podemos escribir V como $V = (U \cap V) \oplus V'$, donde V' es un subespacio isomorfo al espacio cociente $V/(U \cap V)$. En ese caso, $U + V = U + (U \cap V) \oplus V' = U \oplus V'$.

***Definición* 3.1.** *Definimos el operador de correción H_k, para $k \geq 0$, como un operador estocástico sobre los subespacios de W, de forma que, si $dim(V) > k$, $H_k(V)$ devuelve un subespacio aleatorio de V de dimensión k, y, en caso contrario, $H_k(V)$ devuelve V.*

El operador H_k es un simulador de lo que sucede en la red bajo la hipótesis de que el espacio recibido tiene dimensión k.

Además, dados dos subespacios U, V de W, podemos siempre escribir U como $U = H_k(V) \oplus E$ para algún subespacio E de W, asumiendo que $k = \dim(U \cap V)$ y tal que $H_k(U) = U \cap V$. Esto nos permite dar la siguiente definición para el canal de transmisión mencionado anteriormente.

***Definición* 3.2.** *Un canal de transmisión C asociado a un espacio ambiente W es un canal con alfabeto de entrada y salida* $\mathcal{P}(W)$*. El canal de entrada V y de salida U están relacionados por la expresión*

$$U = H_k(V) \oplus E, \tag{3.4}$$

donde $k = dim(U \cap V)$*, y E es el espacio de error. Decimos que la red de transmisión comete* $\rho = dim(V) - k$ *borrones y* $t = dim(E)$ *errores al transformar V en U.*

Esta expresión nos dice en qué se transforma V, el espacio vectorial generado por los paquetes enviados a través de la red, al llegar al receptor. Al borrarse paquetes durante la transmisión, el espacio V se transforma en un espacio vectorial, $H_k(V)$, introducido en la Definición 3.1, de dimensión igual o menor. Si a ésto le añadimos que, al producirse errores, el espacio se transforma en uno más grande, tenemos la expresión(3.4).

Para la expresión (3.4), tenemos que $E \cap V = \{0\}$ y, por tanto, la elección de E depende de V. De no ser así, $U = H_k(V) + E$ para un espacio de error arbitrario, y entonces podríamos descomponer E como $E = (E \cap V) \oplus E'$. Por lo tanto, $U = H_k(V) + (E \cap V) \oplus E' = H_{k'}(V) \oplus E'$ para algún $k' \geq k$, por lo que que tengan intersección no trivial sería beneficioso.

En las siguientes secciones veremos como construir códigos en $\mathcal{P}(W)$. Estos códigos son lo que conocemos como códigos de longitud uno, pues para transmitir una palabra del código necesitamos un uso del canal de transmisión.

3.2. Códigos en canales de transmisión

Ahora que hemos definido que sucede con una red en la que se producen errores y borrones, nuestra intención es construir códigos para estos canales que corrijan errores y borrones. Para construir los códigos, debemos primero dar una métrica, con la correspondiente función de distancia, para los espacios definidos en la sección previa.

3.2.1. Métrica en $\mathcal{P}(W)$

Definimos la función $d : \mathcal{P}(W) \times \mathcal{P}(W) \longrightarrow \mathbb{Z}_+$ como

$$d(A,B) := \dim(A+B) - \dim(A \cap B). \tag{3.5}$$

Además, como sabemos que, para espacios vectoriales, $\dim(A+B) = \dim(A) + \dim(B) - \dim(A \cap B)$, podemos reescribir (3.5) como

$$\begin{aligned} d(A,B) &= \dim(A) + \dim(B) - 2\dim(A \cap B) \\ &= 2\dim(A+B) - \dim(A) - \dim(B), \end{aligned} \tag{3.6}$$

conocida como subspace distance, o distancia entre subespacios.

En estudios posteriores se mejoró esta métrica para códigos de dimensión no constante, definiendo la distancia entre dos subespacios como

$$d(A,B) = dim(A+B) - \text{mín}\{dim(A), dim(B)\},$$

denotada como injection distance.

***Lema* 3.1.** *La función $d(A,B) = dim(A+B) - dim(A \cap B)$ es una métrica para el espacio $\mathcal{P}(W)$.*

Demostración. Para ver que es una métrica, dados los subespacios $A, B, C \in \mathcal{P}(W)$, tenemos que ver que se cumple

1. $d(A,B) \geq 0$ y $d(A,B) = 0$ si y solo si $A = B$.
2. $d(A,B) = d(B,A)$.
3. $d(A,B) \leq d(A,C) + d(C,B)$.

La primera y la segunda condición son triviales por como hemos definido la función distancia en (3.5) y (3.6). Para la tercera condición, tenemos

$$\begin{aligned} d(A,B) - d(A,C) - d(C,B) &= 2\left(\dim(A \cap C) + \dim(B \cap C) - \dim(C) - \dim(A \cap B)\right) \\ &= \dim(A \cap C + B \cap C) - \dim(C) \\ &\quad + \dim(A \cap B \cap C) - \dim(A \cap B) \\ &\leq 0, \end{aligned}$$

pues sabemos que $A \cap C + B \cap C \subseteq C$ y $A \cap B \cap C \subseteq A \cap B$. □

Dada una base de W, como éste es un espacio vectorial de dimensión N sobre $\mathbb{F}_q$, podemos representar los elementos de W como N-uplas con coordenadas en $\mathbb{F}_q$ respecto de la base. Como producto interno tomamos el producto usual, dado por

$$(u,v) = \sum_{i=1}^{N} u_i v_i \quad \text{para } u = (u_1, \ldots, u_N), v = (V_1, \ldots, v_N). \tag{3.8}$$

Si U es un subespacio de dimensión k de W, definimos el espacio ortogonal a U como

$$U^{\perp} = \left\{ v \in W : (u,v) = 0 \text{ para todo } u \in U \right\}, \tag{3.9}$$

que es un subespacio de dimensión $N-k$. Además, dados dos subespacios U y V de W, tenemos que $(U^{\perp})^{\perp} = U$, $(U+V)^{\perp} = U^{\perp} \cap V^{\perp}$ y $(U \cap V)^{\perp} = U^{\perp} + V^{\perp}$. Por último,

$$\begin{aligned} d(U^{\perp}, V^{\perp}) &= \dim(U^{\perp} + V^{\perp}) - \dim(U^{\perp} \cap V^{\perp}) \\ &= \dim((U \cap V)^{\perp}) - \dim((U+V)^{\perp}) \\ &= N - \dim(U \cap V) - (N - \dim(U+V)) \\ &= d(U,V). \end{aligned} \tag{3.10}$$

3.2.2. Códigos y códigos de dimensión constante

Sea W un espacio vectorial de dimensión N sobre $\mathbb{F}_q$. Un código para un canal de transmisión con espacio ambiente W es un subconjunto de $\mathcal{P}(W)$, es decir, una colección de subespacios de W. En este caso, las palabras del código son subespacios de W, por eso en la subsección anterior hemos dado las definiciones de distancia entre dos subespacios.

Al igual que sucede con los códigos clásicos, necesitamos definir la distancia mínima y el tamaño del código para definirlo. Como las palabras del código son subespacios, también necesitamos conocer la dimensión de las palabras de éste (más adelante veremos que nos interesa trabajar con códigos cuyas palabras tengan todas la misma dimensión).

El tamaño de un código C lo denotamos por $|C|$, su distacia mínima por

$$D(C) = \min_{X,Y \in C : X \neq Y} d(X,Y), \tag{3.11}$$

y la dimensión máxima de una palabra del código C como

$$l(C) = \max_{X \in C} \dim(X). \tag{3.12}$$

Nos referiremos a un código cuyas palabras tengan todas la misma dimensión como un código de dimensión constante. Diremos que un código C para un canal de transmisión con espacio ambiente N-dimensional sobre $\mathbb{F}_q$ es de tipo $[N, l(C), \log_q |C|, D(C)]$ en analogía con los códigos clásicos $[n,k,d]$.

El código dual del código C es el código $C^\perp = \{U^\perp : U \in C\}$, donde tenemos, por (3.9), que $D(C^\perp) = D(C)$. Si C es un código de dimensión constante de tipo $[N,l,M,D]$, entonces $C^\perp$ es un código de dimensión constante de tipo $[N, N-l, M, D]$. Está claro que ambos códigos tienen el mismo espacio ambiente, y mismo número de elementos, y además, por construcción, la dimensión de los elementos de $C^\perp$ es $N-l$, pues la dimensión de los elementos de C es l. El hecho de que ambos códigos tengan la misma distancia mínima es una consecuencia de (3.10).

***Definición* 3.3.** *Dado C un código de tipo $[N, l, \log_q |C|, D]$, su peso normalizado λ, su tasa de transmisión R y su distancia mínima normalizada δ vienen definidas por*

$$\lambda = \frac{l}{N}, \quad R = \frac{\log_q |C|}{Nl}, \quad \delta = \frac{D}{2l}. \tag{3.13}$$

En las definiciones dadas, $\lambda \in [0,1]$. Además, si el código C es un código de dimensión constante, $\lambda \in [0, 1/2]$, pues si el código C es tal que $l > N/2$, tomamos el dual $C^\perp$ con $l < N/2$, que tiene mismas propiedades en cuanto

a distancia que C por (3.10). La distancia mínima normalizada $\delta \in [0,1]$, alcanzando el valor 1 si los espacios tienen intersección trivial, pues $D = 2l - \dim(X \cap Y)$, para $X, Y \in C$. Por último, la tasa de transmisión $R \in [0,1]$, y además tiende a 1 solo si λ tiende a 0 (como veremos más adelante). Nos interesa que este parámetro sea lo mayor posible. La diferencia $Nl - \log_q |C|$ es lo que conocemos como redundancia del código.

El problema a la hora de construir códigos para el canal de transmisión definido en (3.4) reside en calcular las uplas $[\lambda, R, \delta]$ cuando la dimensión N se va haciendo arbitrariamente grande. Puede parecer un poco irreal, pues asumimos que la red puede trabajar con paquetes arbitrariamente grandes, pero expresaremos los resultados para una longitud finita N. Los códigos que consideraremos son los mencionados anteriormente, aquellos que inducen un único uso del canal de transmisión, pues como vimos anteriormente, cada vez que hacemos uso del canal de transmisión enviamos una palabra del código, es decir, un subespacio de W.

Un decodificador de distancia mínima para un código C es aquel que dada una salida U del canal de transmisión , devuelve la palabra más cercana del código $V \in C$, es decir, una palabra del código $V \in C$ que satisface que para todo $V' \in C$, $d(U,V) \leq d(U,V')$. Como las palabras del código son en realidad subespacios, para un código de dimensión constante la palabra más cercana será aquella que tenga mayor intersección con U.

En el siguiente teorema, veremos, al igual que sucede con los códigos clásicos, bajo qué condiciones podemos corregir los errores y borrones que se producen en el canal.

***Teorema* 3.1.** *Dado un código que usamos para la transmisión por un canal de transferencia. Sea $V \in C$ el subespacio transmitido, y sea*

$$U = H_k(V) \oplus E$$

el recibido, con $t = dim(E)$. Sea $\rho = (l(C) - k)_+$, con $(l(C) - k)_+ := \max(0, l(C) - k)$, el número máximo de borrones que induce el canal. Si tenemos que

$$2(t + \rho) < D(C), \tag{3.14}$$

entonces el decodificador de distancia mínima de C obtiene V a partir de U.

Recordamos que, en los códigos clásicos, pedíamos que $2t + \rho < D(C)$, pues decíamos que dos borrones equivalían a un error (pues tenemos que encontrarlo y corregirlo).

Demostración. Sea $V' = H_k(V)$, haciendo uso de la desigualdad triangular, tenemos que $d(V,U) \leq d(V,V') + d(V',U)$. Además, como $V' \subseteq V$, tenemos

que $V' \cap V = V'$ y $V' + V = V$, y como $U = V' \oplus E$, tenemos que $V' \subseteq U$ y por lo tanto, $V' \cap U = V'$, $V' + U = U$. Con todo eso,

$$d(V,U) \leq d(V,V') + d(V',U) \leq \rho + t.$$

Consideramos ahora $T \neq V$ otra palabra del código C, entonces

$$D(C) \leq d(V,T) \leq d(V,U) + d(U,T),$$

por lo que

$$d(U,T) \geq D(C) - d(V,U) \geq D(C) - (\rho + t).$$

Ahora, haciendo uso de (3.14) y teniendo en cuenta todo lo anterior,

$$d(U,T) > \rho + t \geq d(U,V),$$

luego $d(U,T) > d(U,V)$ y por tanto un decodificador de distancia mínima nos devuelve V. Recordamos que un decodificador de distancia mínima nos devuelve la palabra del código más cercana a U, es decir, para los códigos con los que estamos trabajando, es el subespacio que tenga mayor intersección con U. □

El siguiente corolario es una consecuencia del teorema anterior, cuando estamos en el caso de que o sólo se producen borrones o sólo se producen errores.

***Corolario* 3.1.** *Sea C un código que usamos en una transmisión a través de un canal de transferencia en el que mandamos $V \in C$. Si recibimos*

$$U = H_{dim(W)}(V) \oplus E = V \oplus E \tag{3.15}$$

y $2t < D(C)$, con $t = dim(E)$, entonces un decodificador de distancia mínima para C nos devuelve V. Además, si recibimos

$$U = H_k(V) \oplus \{0\} = H_k(V) \tag{3.16}$$

y $2\rho < D(C)$, con $\rho = (l(C) - k)_+$, de nuevo un decodificador de distancia mínima para C nos devuelve V.

La primera parte del corolario anterior nos dice que si no hay borrones, un decodificador de distancia mínima corrige errores hasta dimensión

$$t \leq \left\lfloor \frac{D(C) - 1}{2} \right\rfloor. \tag{3.17}$$

Para los códigos clásicos teníamos que si $2t + \rho < D(C)$, un decodificador de distancia mínima nos devuelve V a partir de U [19]. Si no hay borrones,

la desigualdad anterior nos dice que un decodificador de distancia mínima corrige errores hasta dimensión

$$t \leq \left\lfloor \frac{D(C)-1}{2} \right\rfloor,$$

al igual que sucede con los códigos que acabamos de construir.

En cuanto a network coding, nos interesa elegir códigos en los que cada palabra tenga la misma dimensión, pues saber la dimensión de la palabra puede ser usado por el decodificador. Cuando hablamos de códigos de dimensión constante nos restringimos a códigos de tipo $[N,l,M,D]$, con $l \leq N-l$, pues si tenemos un código C de tipo $[N,l,M,D]$ con $l > N-l$, podemos reemplazarlo por su código dual $C^{\perp}$ que como vimos en(3.9), mantiene todas las propiedades de distancia.

***Definición* 3.4.** *Definimos la grassmanniana como $\mathcal{P}(W,l)$, es decir, el conjunto de todos los subespacios de W de dimensión l. El grafo de Grassmann $G_{W,l}$ es un grafo cuyo conjunto de vértices es $\mathcal{P}(W,l)$, y una arista une dos vértices U, V del grafo si y solo si $d(U,V)=2$, con la distancia definida anteriormente.*

Con la definición de grassmanniana dada y el código como lo hemos definido, podemos establecer una biyección entre los elementos del grassmanniano $\mathcal{P}(W,l)$ y las palabras (subespacios) de nuestro código de dimensión fija l. Además, con la definición de distancia dada en (3.6), que una arista una a dos elementos U, V del grafo de Grassmann equivale a decir que $\dim(U \cap V) = l-1$, es decir, que se encuentran a distancia mínima.

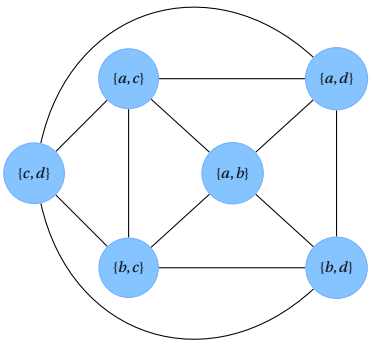

Figura 3.1: Grafo de Grassmann $G_{W,2}$ donde W es el espacio formado por cuatro elementos $\{a,b,c,d\}$ independientes.

Recordamos que un grafo con distancia regular es aquel en el que todos los vértices tienen el mismo número de aristas que salen de él y un grafo

con distancia regular es un grafo regular tal que para cada par de vertices v, w, el número de vértices a distancia j de v y a distancia k de w depende únicamente de j, k y $d(v,w)$.

No entraremos en detalle con este tipo de grafos (consultar [5]), pero tienen la propiedad de ser un grafo con distancia regular. Las palabras de los códigos de dimensión constante las podemos ver de esta forma como vértices del grafo de Grassmann $G_{W,l}$.

En la Figura 3.1 podemos ver un ejemplo del grafo de Grassmann $G_{W,2}$, donde W es un espacio formado por 4 elementos independientes, que denotamos como $\{a,b,c,d\}$. Cada vértice del grafo representa un subespacio de W de dimensión 2, y trazamos una arista entre dos vértices si los subespacios correspondientes a cada vértice están a distancia 2.

En cuanto a network coding lineal, asumimos que el receptor sabe que se ha transimitido una palabra del código $V \in \mathcal{P}(W,l)$. De esta forma, puede elegir recolectar paquetes de información hasta que con ellos genere un subespacio de dimensión l. Esta situación se corresponde con un canal de transmisión de la forma $U = H_{l-t(E)}(V) \oplus E$, con $t(E)$ borrones y $t(E)$ errores. Por el Teorema 3.1, podemos corregir errores hasta dimensión $\left\lfloor \frac{D(C)-1}{4} \right\rfloor$.

Para finalizar esta sección, daremos un par de ejemplos de códigos en $\mathcal{P}(W,l)$

***Ejemplo* 3.1.** *Consideramos W el espacio vectorial formado por N-uplas sobre $\mathbb{F}_q$. Consideramos un conjunto $C \in \mathcal{P}(W,l)$ de espacios U_i, $i = 1,2,\ldots,|C|$, con matrices generadoras $G(U_i) = (I|A_i)$, donde I es la matriz identidad de tamaño $l \times l$ y A_i son todas las matrices posibles de tamaño $l \times (N-l)$ sobre $\mathbb{F}_q$. Cada $G(U_i)$ genera un espacio distinto, y la intersección de dos de ellos tendrá como mucho dimensión $l-1$. Por lo tanto, la distancia mínima del código será*

$$D(C) = l + l - 2(l-1) = 2.$$

Por lo que estamos ante un código de dimensión constante de tipo $[N,l,l(N-l),2]$ con peso normalizado $\lambda = l/N$, tasa de transmisión $R = 1-\lambda$ y distancia normalizada $\delta = \frac{D(C)}{2l} = \frac{1}{l} = \frac{1}{\lambda N}$.

Este ejemplo es un código que, aunque es el que se comenzó usando para random linear network coding por ser un código de dimensión constante y distancia mínima la más pequeña posible para ser un código de dimensión constante, no es el código más óptimo para una distancia D dada, como veremos en el siguiente ejemplo.

***Ejemplo* 3.2.** *Consideramos de nuevo el espacio vectorial W definido en el Ejemplo 3.1, pero esta vez el código que elegiremos será $C' = \mathcal{P}(W,l)$, que, de nuevo, es un código de dimensión constante $[N,l,\log_q|\mathcal{P}(W,l)|,2]$ mayor*

que el código C del Ejemplo 3.1. Además, en la sección siguiente veremos que $|\mathcal{P}(W,l)|$ *es igual al coeficiente de Gauss* $\begin{bmatrix} N \\ l \end{bmatrix}_q$.

Por último, definimos $C'' = \cup_{i=1}^{l} \mathcal{P}(W,i)$, que ya no es un código de dimesión constante. Tenemos que, obviamente, C'' es un código más grande pero, además, tiene distancia mínima $D = 1$ (tomamos $U, U' \in C''$ con $\dim U = \dim U' + 1$ y $U' \subset U$, entonces $d(U,U') = 1$). El principal problema de este código es que, al no ser un código de dimensión constante, el receptor no tiene la información de cuál es la dimensión de la palabra enviada por el canal de transferencia, luego tiene que determinar cuándo se acaba de transmitir. Por el contrario, en los dos ejemplos anteriores, esa información venía ya dada, pues la dimensión del espacio venía fijada de antemano.

3.3. Cotas para códigos

Como hemos visto en la sección anterior, nos interesa construir códigos de dimensión constante. Por ello, necesitamos conocer el número de subespacios de una cierta dimensión l de un espacio vectorial W, pues nos dará el número máximo de palabras que puede tener el código. Ese número viene definido a través de los llamados coeficientes de Gauss.

Consideramos paquetes en $\mathcal{P}(W,l)$ con W un espacio vectorial de dimensión N sobre $\mathbb{F}_q$. Definimos el q-ésimo coeficiente de Gauss para los enteros no negativos l, n con $l \leq n$, como

$$\begin{aligned} \begin{bmatrix} n \\ l \end{bmatrix}_q &:= \frac{(q^n-1)(q^{n-1}-1)\cdots(q^{n-l+1}-1)}{(q^l-1)(q^{l-1}-1)\cdots(q-1)} \\ &= \prod_{i=0}^{l-1} \frac{q^{n-i}-1}{q^{l-i}-1}, \end{aligned} \tag{3.18}$$

donde el producto cuando $l = 0$ lo interpretamos como 1.

Como podemos ver en el capítulo 24 de [17], los coeficientes de Gauss $\begin{bmatrix} n \\ l \end{bmatrix}_q$ nos definen el número de subespacios de dimensión l que hay en un espacio vectorial de dimensión n sobre $\mathbb{F}_q$.

Para $q > 1$, en el siguiente lema podemos ver el comportamiento asintótico de $\begin{bmatrix} n \\ l \end{bmatrix}_q$, lo que nos será útil de cara a aproximar el número de palabras del código.

***Lema* 3.2.** *El coeficiente de Gauss* $\begin{bmatrix} n \\ l \end{bmatrix}_q$ *satisface*

$$1 < q^{-l(n-l)} \begin{bmatrix} n \\ l \end{bmatrix}_q < 4 \tag{3.19}$$

para $0 < l < n$. *Por lo tanto, podemos escribir* $\begin{bmatrix} n \\ \lambda n \end{bmatrix}_q = O\left(q^{n^2\lambda(1-\lambda)}\right)$, *con* $0 < \lambda < 1$.

Demostración. Podemos interpretar $q^{l(n-l)}$ como el número de subespacios de dimensión n en $\mathbb{F}_q^n$ que se producen como el espacio generado por las filas de una matriz de la forma $[I|A]$, donde I es una matriz de tamaño $l \times l$ y A es una matriz $l \times (n-l)$ arbitraria sobre $\mathbb{F}_q$ (como sucedía en el Ejemplo 3.1 para $N = n$). Como $l > 0$, este conjunto no contiene todos los subespacios de dimensión l de $\mathbb{F}_q^n$ y de esta forma obtenemos el lado izquierdo de la desigualdad,

$$q^{l(n-l)} < \begin{bmatrix} n \\ l \end{bmatrix}_q.$$

Para probar el lado derecho, vemos que

$$\begin{aligned} \begin{bmatrix} n \\ l \end{bmatrix}_q &= q^{l(n-l)} \frac{(1-q^{-n})(1-q^{-n+1})\cdots(1-q^{-n+l-1})}{(1-q^{-l})(1-q^{-l+1})\cdots(1-q^{-1})} \\ &< q^{l(n-l)} \frac{1}{(1-q^{-l})(1-q^{-l+1})\cdots(1-q^{1})} \\ &< q^{l(n-l)} \prod_{j=1}^{\infty} \frac{1}{(1-q^{-j})}. \end{aligned}$$

Por el teorema 15.2 de [17], la función $f(x) = \prod_{j=1}^{\infty} \frac{1}{1-x^j}$ es la función generadora de las particiones de enteros, que crece con x. En este caso, nosotros estamos interesados en conocer el comportamiento de la función $f(1/q)$ para $q \geq 2$. Entonces,

$$\prod_{j=1}^{\infty} \frac{1}{1-q^{-j}} \leq \prod_{j=1}^{\infty} \frac{1}{1-2^{-j}} = 1/Q_0 < 4, \tag{3.20}$$

donde $Q_0 \approx 0{,}288788095$ es una constante probabilística combinatoria definida en [4] que nos proporciona la probabilidad de que una matriz cuadrada binaria sobre $\mathbb{F}_2$ sea no singular. □

***Definición* 3.5.** *Sea* X *un conjunto finito. Un esquema asociativo con* d *clases es un par* $(X, \mathcal{R})$ *tal que*

1. $\mathcal{R} = \{R_0, R_1, \ldots, R_d\}$ *es una partición de* $X \times X$*;*
2. $R_0 = \Delta := \{(x,x) | x \in X\}$*;*
3. $R_i = R_i^{\perp}$*, es decir,* $(x,y) \in R_i \Rightarrow (y,x) \in R_i$ *para todo* $i \in \{0,\ldots,d\}$*;*
4. *si* $(x,y) \in R_k$*, el número de elementos* $z \in X$ *tal que* $(x,z) \in R_i$*,* $(z,y) \in R_j$ *es una constante* p_{ij}^k*.*

Volviendo al grafo de Grassmann, sabemos por [5] que es un sistema asociativo, con $(U,V) \in R_i \Leftrightarrow \dim(U \cap V) = l - i$ donde $U, V \in \mathcal{P}(W,l)$. Daremos la definición de esfera en términos del grafo de Grassmann. De esta forma, podemos dar las cotas de Hamming y Gilbert-Varshamov, al igual que hacemos con los códigos clásicos.

***Definición* 3.6.** *Sea* W *un espacio vectorial de dimensión* N *y sea* $\mathcal{P}(W,l)$ *el conjunto de subespacios de* W *de dimensión* l*, la esfera* $S(V,l,t)$ *de radio* t *centrada en* $V \in \mathcal{P}(W,l)$ *se define como el conjunto de subespacios* U *con* $d(U,V) \leq 2t$*, es decir,*

$$S(V,l,t) = \left\{U \in \mathcal{P}(W,l) : d(U,V) \leq 2t\right\}. \tag{3.21}$$

Esta definición de esfera viene dada en términos de la distancia en el grafo de Grassmann, donde recordamos que dos vértices $U, V \in \mathcal{P}(W,l)$ están unidos por una arista si $d(U,V) = 2$. De esta forma, dado V un subespacio de W de dimensión l, diremos que $U \in S(V,l,t)$ si, en el grafo de Grassmann, podemos llegar desde V a U, vistos como vértices, a través de un camino de t aristas o menos. El siguiente teorema nos dice cuál es el número de elementos (espacios) que están en la esfera.

***Teorema* 3.2.** *El número de espacios en* $S(V,l,t)$ *no depende de* V *y es igual a*

$$|S(V,l,t)| = \sum_{i=0}^{t} q^{i^2} \begin{bmatrix} l \\ i \end{bmatrix} \begin{bmatrix} N-l \\ i \end{bmatrix} \tag{3.22}$$

para $t \leq l$*.*

Demostración. Primero vemos cual es el número de espacios U cuya intersección con V genera un subespacio de dimensión $l-i$, es decir, el número de espacios a distancia $2i$ de U. Podemos elegir el subespacio de dimensión $l-i$ en la intersección de $\begin{bmatrix} l \\ l-i \end{bmatrix} = \begin{bmatrix} l \\ i \end{bmatrix}$ formas distintas. Una vez elegido este subespacio, lo podemos completar para tener un subespacio de dimensión l en un total de

$$\frac{(q^N - q^l)\cdots(q^N - q^{l+i-1})}{(q^l - q^{l-i})\cdots(q^l - q^{l-1})} = q^{i^2} \begin{bmatrix} N-l \\ i \end{bmatrix}$$

formas. Luego el número de espacios a distancia $2i$ alrededor de V es $q^{i^2}\begin{bmatrix} N-l \\ i \end{bmatrix}\begin{bmatrix} l \\ i \end{bmatrix}$. Como nos interesa conocer el número de espacios a distancia menor o igual que $2t$, obtenemos la expresión

$$|S(V,l,t)| = \sum_{i=0}^{t} q^{i^2}\begin{bmatrix} l \\ i \end{bmatrix}\begin{bmatrix} N-l \\ i \end{bmatrix}.$$

Además, como $\mathcal{P}(W,l)$ es un grafo con distancia regular, la elección no depende de V [5]. □

Como nota, $|S(V,l,t)| = |S(V,N-l,t)|$, cosa que ya sabíamos por (3.9).

Ahora que ya hemos visto la definición de esfera en el contexto del grafo de Grassmann y sabemos calcular el número de elementos que hay en ella, podemos dar la cota de Hamming (o sphere-packing bound) y la de Gilbert-Varshamov (o sphere-covering bound) para el código construido sobre la grassmanniana. Las cotas que obtendremos nos permiten diferenciar entre un buen código, uno malo y un código imposible.

***Teorema* 3.3.** *Sea C un conjunto de espacios en $\mathcal{P}(W,l)$ tal que $D(C) \geq 2t$, y sea $s = \left\lfloor \frac{t-1}{2} \right\rfloor$. El número de elementos (subespacios de dimensión l de W) que hay en C está acotado superiormente por la expresión*

$$\begin{aligned} |C| \leq \frac{|\mathcal{P}(W,l)|}{|S(V,l,s)|} &= \frac{\begin{bmatrix} N \\ l \end{bmatrix}}{|S(V,l,s)|} \\ &< \frac{\begin{bmatrix} N \\ l \end{bmatrix}}{q^{s^2}\begin{bmatrix} N-l \\ s \end{bmatrix}\begin{bmatrix} l \\ s \end{bmatrix}} < 4q^{(l-s)(N-s-l)}, \end{aligned} \tag{3.23}$$

también llamada cota de Hamming o sphere-packing bound.

Por el contrario, existe un código C' con $D(C') \geq 2t$ tal que

$$\begin{aligned} |C| \geq \frac{|\mathcal{P}(W,l)|}{|S(V,l,t-1)|} &= \frac{\begin{bmatrix} N \\ l \end{bmatrix}}{|S(V,l,t-1)|} \\ &> \frac{\begin{bmatrix} N \\ l \end{bmatrix}}{(t-1)q^{(t-1)^2}\begin{bmatrix} l \\ t \end{bmatrix}\begin{bmatrix} N-l \\ t-1 \end{bmatrix}} > \frac{1}{16t}q^{(l-t+1)(N-t-l+1)}, \end{aligned} \tag{3.24}$$

lo que conocemos como cota de Gilbert-Varshamov o sphere-covering bound.

Demostración. Si tenemos en cuenta la expresión obtenida en (3.22) junto con (3.19) y la sección 2.2 de [19] llegamos al resultado deseado. □

De nuevo, estas cotas son simétricas para l y $N-l$ como consecuencia de (3.9). Vamos ahora a expresar las cotas obtenidas en el Teorema 3.3 en términos de los parámetros normalizados.

***Corolario* 3.2.** *Sea C una colección de espacios en $\mathcal{P}(W,l)$ con distancia mínima normalizada $\delta = \frac{D(C)}{2l}$. La tasa de transmisión R de C viene acotada por*

$$R \leq (1-\delta/2)(1-\lambda(1+\delta/2)) + o(1), \tag{3.25}$$

donde $o(1)$ tiende a cero cuando N crece. Por el contrario, existe un código C' con distancia normalizada δ tal que la tasa de transmisión R viene acotada por

$$R \geq (1-\delta)(1-\lambda(\delta+1)) + o(1), \tag{3.26}$$

donde $o(1)$ tiende a cero cuando N crece.

Demostración. Tomamos a ambos lados de (3.23) y (3.24) logaritmo en base q, y teniendo en cuenta que $\log_q |C| = RNl$, y las definiciones dadas en (3.13), llegamos a las cotas dadas en (3.25) y (3.26). □

En la gráfica izquierda de la Figura 3.2, hemos representado las cotas obtenidas en el Corolario 3.2 para un valor de λ fijo igual a $\lambda = 0{,}2$. En ellas vemos como varía la tasa de transmisión R en función de la distancia normalizada δ. Estas cotas nos dicen que un código malo es un código cuya tasa de transmisión R en función de la distancia normalizada δ esté por debajo de la línea azul, uno bueno es aquel que se encuentre entre la línea azul y roja y uno imposible es aquel que se encuentre por encima de la línea roja.

La cota superior, dada por la expresión (3.25) no parece muy buena, sobre todo teniendo en cuenta que $\delta \in [0,1]$. Ésta es la motivación para calcular la cota superior de Singleton. Además, en [12] podemos ver representada esta misma gráfica para un valor de $\lambda = 0{,}25$.

Nuestro objetivo ahora es calcular la cota superior de Singleton. Para ello, primero comenzamos definiendo una operación de recortado para códigos. Suponemos que C es una colección de subespacios en $\mathcal{P}(W,l)$, donde W tiene dimensión N. Sea W' un subespacio de W de dimensión $N-1$. Obtenemos un código recortado C' a partir de C tomando, en lugar de $V \in C$, $V' = H_{l-1}(V \cap W')$, es decir, reemplazamos V por $V \cap W'$ si $V \cap W'$ tiene dimensión $l-1$ y sino reemplazamos V por un subespacio de V de dimensión $l-1$. Denotaremos a este código como $C' = C|_{W'}$. Esta operación es similar a la que definíamos en la primera sección del capítulo, denotada por $H_k(V)$, cuando "perdíamos"dimensión del espacio vectorial V en la transmisión.

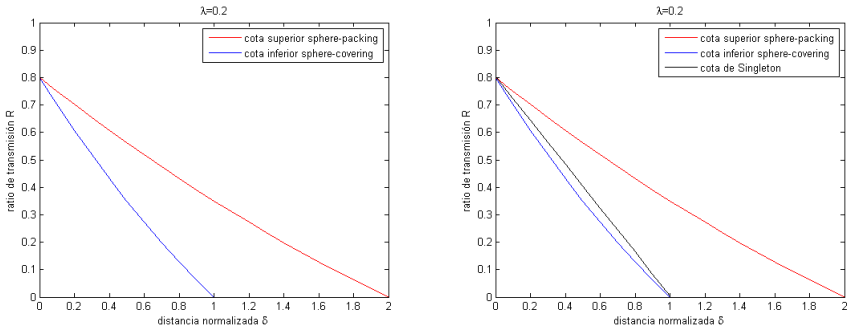

Figura 3.2: Izquierda: Cotas normalizadas de sphere-packing y sphere-covering en función de δ para $\lambda = 0{,}2$. Derecha: Cotas normalizadas de sphere-packing y sphere-covering, junto con la cota normalizada de Singleton ($N = 1000$) en función de δ para $\lambda = 0{,}2$.

***Teorema* 3.4.** *Si $C \subseteq \mathcal{P}(W,l)$ es un código de tipo $[N,l,\log_q|C|,D]$ con $D>2$ y W' es un subespacio de W de dimensión $N-1$, entonces $C' = C|_{W'}$ es un código de tipo $[N-1,l-1,\log_q|C|,D']$ con $D' \geq D-2$.*

Demostración. Comenzaremos viendo que $D' \geq D-2$. Sean U, V dos palabras del código C y supongamos que $U' = H_{l-1}(U \cap W')$ y $V' = H_{l-1}(V \cap W')$ son las correspondientes palabras en C'. Como $U' \subseteq U$ y $V' \subseteq V$, tenemos que $U' \cap V' \subseteq U \cap V$, luego

$$2\dim(U' \cap V') \leq 2\dim(U \cap V) = d(U) + d(V) - d(U,V) \leq 2l - D.$$

Entonces tenemos que

$$d(U',V') = \dim(U') + \dim(V') - 2\dim(U' \cap V') \geq 2(l-1) + D - 2l = D-2.$$

Como $D>2$, la distancia $d(U',V') > 0$, es decir, U' y V' son distintos, por lo que podemos identificar U con U' y V con V', luego C y C' tienen el mismo número de palabras. De esta forma, podemos establecer una biyección entre los elementos de C y C', es decir, una biyección entre $\mathcal{P}(W,l)$ y $\mathcal{P}(W',l-1)$. □

Una vez estudiado como recortar un código C para los códigos con los que estamos trabajando (códigos de dimensión constante cuyas palabras son subespacios), podemos dar la cota superior de Singleton, que mejora notablemente la cota de Hamming.

***Teorema* 3.5.** *Un código $C \subseteq \mathcal{P}(W,l)$ del tipo $[N,l,\log_q|C|,D]$ satisface que*

$$|C| \leq \left[\begin{array}{c} N-(D-2)/2 \\ \max\{l, N-l\} \end{array} \right]_q, \tag{3.27}$$

también conocida como cota de Singleton.

Demostración. La demostración es similar a la de la cota de Singleton para códigos clásicos. Si recortamos C un total de $(D-2)/2$ veces, obtenemos un código $C' = C|_{W'}$ de tipo $[N-(D-2)/2, l-(D-2)/2, \log_q |C|, D']$ cuyas palabras tienen dimensión $l-(D-2)/2$ y con $D' \geq 2$. El código no puede tener más palabras que el correspondiente grassmanniano, pues existe una biyección entre las palabras del código y los elementos del grassmanniano, que tiene

$$\left[\begin{array}{c} N-(D-2)/2 \\ l-(D-2)/2 \end{array} \right]_q = \left[\begin{array}{c} N-(D-2)/2 \\ N-l \end{array} \right]_q \tag{3.28}$$

palabras. Repitiendo el mismo argumento sobre el código dual $C^{\perp}$, que sabemos, por (3.10), que conserva las propiedades de distancia, tenemos

$$\left[\begin{array}{c} N-(D-2)/2 \\ l \end{array} \right]_q . \tag{3.29}$$

Entonces, (3.28)<(3.29) si y solo si $l < N-l$, obteniendo de esta forma la cota deseada. □

De nuevo, al igual que hicimos para las cotas de Hamming y Gilbert-Varshamov, normalizaremos la cota obtenida en el Teorema 3.5 en términos de los parámetros normalizados descritos en (3.13). Consideramos el caso en el que $l \leq N-l$, es decir, $\lambda \leq 1/2$.

***Corolario* 3.3.** *Sea C una colección de espacios en $\mathcal{P}(W,l)$, con $l \leq dim(W)/2$ y distancia mínima normalizada $\delta = \frac{D(C)}{2l}$. La tasa de transmisión está acotada superiormente por*

$$R \leq (1-\delta)(1-\lambda) + \frac{1}{\lambda N}(1-\lambda+o(1)). \tag{3.30}$$

Demostración. Tomamos la cota obtenida en (3.27) y aplicamos $\log_q$ a ambos lados de la desigualdad para llegar a la expresión buscada. □

Al igual que hicimos con las cotas de Hamming y Gilbert-Varshamov, en la gráfica de la derecha de la Figura 3.2 hemos representado, junto con las dos cotas obtenidas anteriormente, la cota superior de Singleton cuando N crece (la hemos representado para $N = 1000$). Vemos como, efectivamente, la cota superior de Singleton es mucho mejor que la cota superior de Hamming.

Para concluir con esta sección, haremos una tabla comparativa entre los códigos Reed-Solomon tradicionales y los códigos asociados a la grassmanniana estudiados en este capítulo. Construimos para ello la Tabla 3.1. En el lado izquierdo de ésta vemos algunas de las propiedades de los códigos Reed-Solomon clásicos, y, en el lado derecho, veremos estas propiedades para los códigos de dimensión constante asociados a la grassmnniana estudiados en este capítulo.

Tabla 3.1: Tabla comparativa.

Códigos clásicos	**Códigos asociados a la grassmanniana**
códigos $\mathcal{C} \subset \mathbb{F}_q^n$;	códigos $C \subset \mathcal{P}(W,l)$ o grassmanniana,
métrica de Hamming: $d(x,y) = \lvert\{i : x_i \neq y_i\}\rvert$;	métrica del subespacio: $d(U,V) = \dim(U+V) - \dim(U \cap V)$;
distancia mínima $D(\mathcal{C}) = \min\{d(x,y) : x,y \in \mathcal{C}\}$;	distancia mínima $D(C) = \min\{d(U,V) : U,V \in C\}$;
código de parámetros $[n,k,d]_q$;	código de parámetros $[N,l,\log_q \lvert C\rvert, D(C)]$;
código dual $\mathcal{C}^\perp$ $[n,n-k,d']_q$;	código dual $C^\perp$ $[N,N-l,\log_q \lvert C\rvert, D(C)]$;
podemos corregir t errores y ρ borrones si $2t+\rho < D(\mathcal{C})$;	podemos corregir t errores y ρ borrones si $2(t+\rho) < D(C)$;
un decodificador de distancia mínima corrige t errores, con $t \leq \left\lfloor \frac{D(C)-1}{2} \right\rfloor$;	un decodificador de distancia mínima corrige t errores, con $t \leq \left\lfloor \frac{D(C)-1}{2} \right\rfloor$;

3.4. Construcción de un código tipo Reed-Solomon y el algoritmo de decodificación

Para acabar el capítulo, en esta sección construiremos un código, para nuestra métrica, que pueda corregir los errores y borrones de nuestro canal de transmisión descrito en la primera sección de este capítulo.

Para ello comenzamos definiendo el concepto de polinomio linealizado. Sea $\mathbb{F}_q$ un cuerpo y $\mathbb{F} = \mathbb{F}_{q^m}$ un cuerpo extensión. Como podemos ver en el capítulo 11 de [3], decimos que un polinomio $L(x)$ es un polinomio linealizado sobre $\mathbb{F}$ si lo podemos escribir como

$$L(x) = \sum_{i=0}^{d} a_i x^{q^i}, \tag{3.31}$$

con coeficientes $a_i \in \mathbb{F}$ para $i = 0,\ldots,d$. Decimos que $L_1(x) \equiv L_2(x)$ si $L_1(x) - L_2(x) \equiv 0$, donde 0 representa al polinomio cuyos coeficientes son todos

nulos. Por comodidad para la notación, usaremos $x^{[i]}$ para denotar x^{q^i}. De esta forma,

$$L(x) = \sum_{i=0}^{d} a_i x^{[i]}. \tag{3.32}$$

Damos ahora dos operaciones sobre los polinomios linealizados. Sea $L_1(x) = \sum_{i=0}^{d} a_i x^{[i]}$, $L_2(x) = \sum_{i=0}^{d'} b_i x^{[i]}$ dos polinomios linealizados. Si suponemos que $d \leq d'$, tenemos que

$$\alpha L_1(x) + \beta L_2(x) = \sum_{i=0}^{d} \underbrace{(\alpha a_i + \beta b_i)}_{c_i} x^{[i]} + \sum_{i=d+1}^{d'} \underbrace{\beta b_i}_{c_i} x^{[i]} = \sum_{i=0}^{d'} c_i x^{[i]}$$

es un polinomio linealizado, con $\alpha, \beta \in F$. Esto no sucede con el producto, pues $q^{[i]} + q^{[j]} = q^{[i]+[j]}$ no tiene por qué ser igual a un cierto $q^{[k]}$. Sin embargo, la composición $L_1(L_2(x))$, que denotamos como $L_1(x) \otimes L_2(x)$, de dos polinomios linealizados si es un polinomio linealizado sobre $\mathbb{F}$.

Si $L_1(x) = \sum_{i\geq 0} a_i x^{[i]}$ y $L_2(x) = \sum_{j\geq 0} b_j x^{[j]}$, entonces

$$\begin{aligned} L_1(x) \otimes L_2(x) = L_1(L_2(x)) &= \sum_{i\geq 0} a_i (L_2(x))^{[i]} \\ &= \sum_{i\geq 0} a_i \left(\sum_{j\geq 0} b_j x^{[j]} \right)^{[i]} \\ &= \sum_{i\geq 0} \sum_{j\geq 0} a_i b_j^{[i]} x^{[i]+[j]} = \sum_{k\geq 0} c_k x^{[k]}, \end{aligned}$$

con $c_k = \sum_{i=0}^{l} a_i b_{k-i}^{[i]}$. Esta operación no es conmutativa en general.

Si $L_1(x)$ tiene grado q^{d_1} y $L_2(x)$ tiene grado q^{d_2}, entonces $L_1(x) \otimes L_2(x)$ y $L_2(x) \otimes L_1(x)$ tienen ambos grado $q^{d_1+d_2}$.

De esta forma, con las operaciones $+$, $\otimes$ definidas, el conjunto de los polinomios linealizados forman un anillo no conmutativo con elemento unidad. Definimos la norma de un elemento del anillo como su grado. Al igual que hemos dado la operación de "multiplicación"$\otimes$ por la derecha y la izquierda como la composición, podemos definir la operación de división, también a través de la composición y la suma. Sean $a(x)$, $b(x)$, existen polinomios linealizados únicos $q_L(x)$, $q_R(x)$, $r_L(x)$ y $r_R(x)$ tal que

$$a(x) = q_L(x) \otimes b(x) + r_L(x) = q_R(x) \otimes b(x) + r_R(x), \tag{3.33}$$

con $r_L(x) \equiv 0$ ó $\deg(r_L(x)) < \deg(b(x))$, y $r_R(x) \equiv 0$ ó $\deg(r_R(x)) < \deg(b(x))$.

A continuación describiremos el algoritmo que nos permite calcular $q_R(x)$ y $r_R(x)$ para nuestros polinomios linealizados $a(x)$, $b(x)$. Usamos en él la notación $lc(a(x))$ para referirnos al coeficiente del término líder de $a(x)$ (el término de mayor grado).

Algoritmo* 3.1.** ***Algoritmo RDiv

Input: *un par* $a(x)$, $b(x)$ *de polinomios linealizados sobre* $\mathbb{F} = \mathbb{F}_{q^m}$, $b(x) \equiv 0$

Output: *un par* $q_R(x)$, $r_R(x)$ *de polinomios linealizados sobre* $\mathbb{F}_{q^m}$ *que verifican* (3.33).

if $deg(a(x)) < deg(b(x))$ ***then***

  return $(0, a(x))$

else

  $d = deg(a(x))$, $e = deg(b(x))$, $a_d = lc(a(x))$, $b_e = lc(b(x))$;

  $t(x) = (a_d / b_e^{[d-e]}) x^{[d-e]}$

  return $(t(x), 0) +$ ***RDiv***$(a(x) - b(x) \otimes t(x), b(x))$

endif

El algoritmo **RDiv** nos devuelve dos polinomios $q(x)$ y $r(x)$ tal que $a(x) = b(x) \otimes q(x) + r(x)$ con $r(x) \equiv 0$ ó $\deg(r(x)) < \deg(b(x))$.

***Ejemplo* 3.3.** *Sea* $a(x) = x^{[4]} + x^{[3]} + x^{[1]} + x^{[0]}$, $b(x) = x^{[2]} + x^{[1]} + x^{[0]}$ *sobre* $\mathbb{F}_2$. *Como* $deg(a(x)) = 4 \geq deg(b(x)) = 2$ *calculamos* $t_1(x) = x^{[2]}$. *Ahora* $a_1(x) = a(x) - b(x) \otimes t(x) = x^{[2]} + x^{[1]} + x^{[0]}$. *De nuevo* $deg(a_1(x)) = 2 \geq deg(b(x)) = 2$ *por lo que pasamos ahora a calcular* $t_2(x) = x^{[0]}$, *obteniendo así* $a_2(x) = a_1(x) - b(x)\, t_1(x) = 0$, *por lo que hemos acabado y tenemos que* $a(x) = b(x) \otimes t(x) + r(x)$ *con* $t(x) = x^{[2]} + x^{[0]}$ *y* $r(x) = 0$.

De manera muy similar, podemos construir el Algoritmo **LDiv**, que realiza la división por la izquierda, cambiando las dos últimas líneas del Algoritmo **RDiv** por

$$t(x) = (a_d / b_e^{[d-e]}) x^{[d-e]},$$
$$\textbf{return}(t(x), 0) + \textbf{LDiv}(a(x) - t(x) \otimes b(x), b(x)).$$

El nombre de polinomio linealizado no es casual, pues viene dado por la siguiente propiedad. Sea $L(x)$ un polinomio linealizado sobre $\mathbb{F}$ y sea K una extensión de $\mathbb{F}$. Entonces podemos ver K como un espacio vectorial sobre $\mathbb{F}_q$. La aplicación que lleva $\beta \in K$ en $L(\beta) \in K$ es **lineal** respecto de $\mathbb{F}_q$, es decir, para todo $\beta_1, \beta_2 \in K$ y para todo $\lambda_1, \lambda_2 \in \mathbb{F}_q$,

$$L(\lambda_1 \beta_1 + \lambda_2 \beta_2) = \lambda_1 L(\beta_1) + \lambda_2 L(\beta_2). \tag{3.34}$$

Ahora suponemos que elegimos K de forma que incluye todos los ceros de $L(x)$. Estos ceros se corresponden con el núcleo de $L(x)$ visto como una aplicación lineal, luego forman un espacio vectorial sobre $\mathbb{F}_q$. Si $L(x)$ tiene grado q^d, el espacio vectorial tiene dimensión d a lo sumo, siendo menor si $L(x)$ tiene raices repetidas.

Si V es un subespacio de dimensión n de K,

$$L(x) = \prod_{\beta \in V} (x - \beta) \tag{3.35}$$

es un polinomio mónico linealizado sobre K (ver el Teorema 3.52 de [16]).

***Lema* 3.3.** *Sea d un entero positivo y sean $f(x)$, $g(x)$ dos polinomios linealizados sobre $\mathbb{F}$ de grado menor que q^d. Si $\alpha_1, \ldots, \alpha_d$ son los puntos linealmente independientes de K tal que $f(\alpha_i) = g(\alpha_i)$ para $i = 1, \ldots, d$, entonces $f(x) \equiv g(x)$.*

Demostración. Esta demostración utilizará un resultado clásico conocido sobre polinomios, que nos dice que si un polinomio tiene más raíces que su grado, éste solo puede ser el polinomio idénticamente nulo.

Definimos $h(x) = f(x) - g(x)$, entonces, $\alpha_1, \ldots, \alpha_d$ son ceros de $h(x)$, y por tanto también lo serán las q^d combinaciones lineales de estos elementos, por lo que tiene q^d ceros distintos, pero como el grado de $h(x)$ es menor que q^d, $h(x) \equiv 0$. □

3.4.1. Construcción del código

Comenzamos dando un ejemplo sobre la construcción de un código RS (Reed-Solomon) clásico.

***Ejemplo* 3.4.** *Consideramos un código estándar RS con $n = q - 1$. Sea $q = 4$, entonces $n = 3$ y tomamos $k = 2$. Como los códigos RS son MDS [19], $d = n - k + 1 = 2$, luego tenemos un código RS (3,2,2). Vamos a trabajar sobre $\mathbb{F}_4$. A continuación vemos las tablas de suma y multiplicación para este cuerpo.*

Tabla 3.2: Tablas de suma y multiplicación en $\mathbb{F}_4$

$+$	0	1	α	α^2
0	0	1	α	α^2
1	1	0	α^2	α
α	α	α^2	0	1
α^2	α^2	α	1	0

$\times$	0	1	α	α^2
0	0	0	0	0
1	0	1	α	α^2
α	0	α	α^2	1
α^2	0	α^2	1	α

Tomando vectores de longitud $k = 2$, la siguiente aplicación nos permite obtener las palabras de tamaño n.

$$RS[3,2,2] : f(z) = f_0 + f_1 z \longmapsto (f(\alpha^0), f(\alpha^1), f(\alpha^2)),$$

donde $\alpha \in \mathbb{F}_4$ es un elemento primitivo, $f(z) \in \mathbb{F}_4[z]$ y $deg f(z) < 2$. Es decir,

$$(f_0, f_1) \longmapsto (f_0 + f_1\alpha^0, f_0 + f_1 + \alpha^1, f_0 + f_1\alpha^2).$$

Como f_0 y f_1 pueden tomar 4 valores cada uno, obtenemos así las 16 palabras del código.

$$
\begin{array}{ll}
(0,0) \mapsto (0,0,0) & (\alpha,0) \mapsto (\alpha,\alpha,\alpha) \\
(0,1) \mapsto (1,\alpha,\alpha^2) & (\alpha,1) \mapsto (\alpha^2,0,1) \\
(0,\alpha) \mapsto (\alpha,\alpha^2,1) & (\alpha,\alpha) \mapsto (0,1,\alpha^2) \\
(0,\alpha^2) \mapsto (\alpha^2,1,\alpha) & (\alpha,\alpha^2) \mapsto (1,\alpha^2,0) \\
(1,0) \mapsto (1,1,1) & (\alpha^2,0) \mapsto (\alpha^2,\alpha^2,\alpha^2) \\
(1,1) \mapsto (0,\alpha^2,\alpha) & (\alpha^2,1) \mapsto (\alpha,1,0) \\
(1,\alpha) \mapsto (\alpha^2,\alpha,0) & (\alpha^2,\alpha) \mapsto (1,0,\alpha) \\
(1,\alpha^2) \mapsto (\alpha,0,\alpha^2) & (\alpha^2,\alpha^2) \mapsto (0,\alpha,1)
\end{array}
$$

Al igual que con los códigos RS clásicos, donde obtenemos las componentes de las palabras evaluando un polinomio mensaje, en los códigos asociados a canales de transmisión, donde las palabras son espacios vectoriales, obtendremos una base del espacio vectorial transmitido al evaluar un polinomio mensaje linealizado.

Sea $\mathbb{F}_q$ un cuerpo finito, y sea $\mathbb{F} = \mathbb{F}_{q^m}$ una extensión finita de $\mathbb{F}_q$. Vemos $\mathbb{F}$ como un espacio vectorial de dimensión n sobre $\mathbb{F}_q$. Sea $A = \{\alpha_1, \ldots, \alpha_l\} \subset \mathbb{F}$ un conjunto de elementos linealmente independientes en el espacio vectorial, estos elementos generan un subespacio vectorial $\langle A \rangle \subseteq \mathbb{F}$ sobre $\mathbb{F}_q$ de dimensión l, con $l \leq m$. Tomaremos ahora como espacio ambiente la suma directa $W = \langle A \rangle \oplus \mathbb{F} = \left\{(\alpha, \beta) : \alpha \in \langle A \rangle, \beta \in \mathbb{F}\right\}$, un espacio vectorial de dimensión $l + m$ sobre $\mathbb{F}_q$.

Sea $u = (u_0, u_1, \ldots, u_{k-1}) \in \mathbb{F}^k$ un vector con k símbolos sobre $\mathbb{F}$, o mk símbolos sobre $\mathbb{F}_q$. Sea $\mathbb{F}^k[x]$ el conjunto de polinomios linealizados sobre $\mathbb{F}$ de grado q^{k-1} a lo sumo. Sea $f \in \mathbb{F}^k[x]$, dado por

$$f(x) = \sum_{i=0}^{k-1} u_i x^{[i]}, \tag{3.36}$$

un polinomio linealizado con coeficientes u y sea $\beta_i = f(\alpha_i)$. Cada par (α_i, β_i), con $i = 1 \ldots, l$, lo podemos ver como un vector de W. Además, como $\{\alpha_1, \ldots, \alpha_l\}$ un conjunto linealmente independiente, también lo será $\{(\alpha_1, \beta_1), \ldots, (\alpha_l, \beta_l)\}$. Este conjunto generará un subespacio V de dimensión l de W. Definimos ev_A la aplicación que lleva $f(x) \in \mathbb{F}^k[x]$ en el espacio $V \in \mathcal{P}(W, |A|)$.

***Lema* 3.4.** *Si $|A| \geq k$, la aplicación*

$$ev_A : \mathbb{F}^k[x] \longrightarrow \mathcal{P}(W, |A|)$$

es inyectiva.

Demostración. Suponemos $|A| \geq k$ y $ev_A(f) = ev_A(g)$ para unas ciertas $f(x), g(x) \in \mathbb{F}^k[x]$. Sea $h(x) = f(x) - g(x)$. Entonces $h(\alpha_i) = 0$ para $i = 1, \ldots, l$, por lo que al ser $h(x)$ un polinomio linealizado, $h(x) = 0$ para todo $x \in \langle A \rangle$. Luego $h(x)$ tiene al menos $q^{|A|} \geq q^k$ ceros, pero como $h(x)$ tiene grado máximo q^{k-1}, por el Lema 3.3, $h(x) \equiv 0$, luego $f(x) \equiv g(x)$. □

A partir de ahora asumiremos que $l \geq k$. El Lema 3.4 implica que si tenemos que $l \geq k$, la imagen de $\mathbb{F}^k[x]$ por ev_A es un código $C \subseteq \mathcal{P}(W, l)$ con q^{mk} palabras, que serán espacios vectoriales.

Teníamos que si $\{\alpha_1, \ldots, \alpha_r\}$ era un conjunto linealmente independiente y $\beta_i = f(\alpha_i)$ para $i = 1, \ldots, r$, entonces $\{(\alpha_1, \beta_1), \ldots, (\alpha_r, \beta_r)\}$ era un conjunto linealmente independiente. Veremos ahora que esto también sucede al revés.

***Lema* 3.5.** *Si $\{(\alpha_1, \beta_1), \ldots, (\alpha_r, \beta_r)\} \subseteq W$ es una colección de r elementos linealmente independientes tal que $\beta_i = f(\alpha_i)$ para un cierto polinomio linealizado f sobre $\mathbb{F}$, entonces $\{\alpha_1, \ldots, \alpha_r\}$ es un conjunto linealmente independiente.*

Demostración. Suponemos que para unos ciertos $\gamma_1, \ldots, \gamma_r \in \mathbb{F}_q$ tenemos que $\sum_{i=1}^r \gamma_i \alpha_i = 0$. En W tendríamos que

$$\sum_{i=1}^r \gamma_i(\alpha_i, \beta_i) = \left(\sum_{i=1}^r \gamma_i \alpha_i, \sum_{i=1}^r \gamma_i \beta_i \right) = \left(0, \sum_{i=1}^r \gamma_i f(\alpha_i) \right)$$
$$= \left(0, f\left(\sum_{i=1}^r \gamma_i \alpha_i \right) \right) = (0, f(0)) = (0, 0).$$

Como los (α_i, β_i) son linealmente independientes entre ellos, ésto solo sucede si $\gamma_1 = \cdots = \gamma_r = 0$. □

El siguiente teorema nos dice qué tipo de código C es el generado por la imagen de $\mathbb{F}^k[x]$ a través de ev_A.

***Teorema* 3.6.** *Sea C la imagen de $\mathbb{F}^k[x]$ por ev_A, con $l = |A| \geq k$. Entonces C es un código de tipo $[l + m, l, mk, 2(l - k + 1)]$.*

Demostración. Solo necesitamos probar la distancia mínima, pues el resto es trivial por como hemos definido el código. Sean $f(x)$, $g(x)$ dos elementos de $\mathbb{F}^k[x]$ distintos, y sea $U = ev_A(f)$, $V = ev_A(g)$. Suponemos que $\dim(U \cap V) = r$, por lo que existen r elementos linealmente independientes $(\alpha'_1, \beta'_1), \ldots, (\alpha'_r, \beta'_r)$ tal que $f(\alpha'_i) = g(\alpha'_i) = \beta'_i$. Haciendo uso del Lema 3.5, $\alpha'_1, \ldots, \alpha'_r$ son linealmente independientes y, por tanto, generan un espacio B de dimensión r tal que $f(b) - g(b) = 0$ para todo $b \in B$.

Si $r \geq k$, entonces $f(x)$ y $g(x)$ serían dos polinomios linealizados de grado menor que q^k que coinciden en al menos k puntos linealmente independientes, y, por el Lema 3.3, $f(x) \equiv g(x)$. Como hemos dicho que son distintos, tenemos que $r \leq k-1$. Luego,

$$d(U,V) = \dim(U) + \dim(V) - 2\dim(U \cap V) = 2l - 2r \geq 2(l-k+1). \quad (3.37)$$

□

Aplicando la cota de Singleton al código descrito en el Teorema 3.6, tenemos que

$$|C| = \begin{bmatrix} N-(D-2)/2 \\ l-(D-2)/2 \end{bmatrix}_q = \begin{bmatrix} m+k \\ k \end{bmatrix}_q < 4q^{mk}, \quad (3.38)$$

donde la última desigualdad la obtenemos teniendo en cuenta (3.19). Esto implicaría que un código que alcance la cota de Singleton tendría 4 veces más palabras que el código C que acabamos de definir. Pero entonces, para un N lo suficientemente grande, la diferencia sería irrelevante. En términos de la tasa de transmisión, tenemos que

$$R = (1-\lambda)\left(1-\delta+\frac{1}{\lambda N}\right), \quad (3.39)$$

que tiene el mismo comportamiento que la cota de Singleton cuando N tiende a infinito, por lo que estos códigos tipo Reed-Solomon están cerca de alcanzar la cota de Singleton.

3.4.2. Decodificación

Suponemos que una palabra V del código C, es decir, un subespacio, se transmite por un canal de transmisión y recibimos un subespacio U de W de dimensión $(l-\rho+t)$, con $\dim(U \cap V) = l-\rho$. En este caso, tenemos ρ borrones, un error de dimensión t y $d(U,V) = \rho + t$. Nuestro objetivo es recuperar V de U si $\rho + t < D/2 = l-k+1$. Para ello, procedemos a describir un algoritmo de decodificación de distancia mínima Sudan-style "list-1", que podemos encontrar en la sección 9.3 de [20].

Sea $r = l-\rho+t$ la dimensión del espacio recibido, y sea (x_i, y_i), con $i = 1,\ldots,r$, una base de U. Suponemos que podemos construir un polinomio en dos variables $Q(x,y)$ tal que

$$Q(x,y) = Q_x(x) + Q_y(y) \text{ tal que } Q(x_i,y_i) = 0 \text{ para todo } i = 1,\ldots,r, \quad (3.40)$$

donde $Q_x(x)$ es un polinomio linealizado sobre $\mathbb{F}_{q^m}$ de grado $q^{\tau-1}$ a lo sumo, y $Q_y(y)$ es un polinomio linealizado sobre $\mathbb{F}_{q^m}$ de grado $q^{\tau-k}$ a lo sumo.

Elegimos $Q(x,y)$ de forma que interpole en una base de U, y como $Q(x,y)$ es un polinomio linealizado, $Q(x,y)=0$ para todo $(x,y)\in U$.

En (3.40) lo que estamos definiendo es un sistema de r ecuaciones con $2\tau-k+1$ incógnitas, donde para cada par (x_i,y_i), la ecuación es de la forma.

$$0=a_0x_i^{[0]}+a_1x_i^{[1]}+\cdots+a_{\tau-1}x_i^{[\tau-1]}+b_0y_i^{[0]}+b_1y_i^{[1]}+\cdots+b_{\tau-k}y_i^{[\tau-k]}.$$

Dicho sistema tendra solución distinta de la solución trivial si y solo si

$$r=l-\rho+t<2\tau-k+1 \tag{3.41}$$

(es compatible indeterminado).

Además, como $f(x)$ es un polinomio linealizado sobre $\mathbb{F}_{q^m}$, también lo será $Q(x,f(x))$ dado por

$$Q(x,f(x))=Q_x(x)+Q_y(f(x))=Q_x(x)+Q_y(x)\otimes f(x). \tag{3.42}$$

Como el grado de $f(x)$ es a lo sumo q^{k-1}, el grado de $Q(x,f(x))$ será $q^{\tau-1}$ como máximo.

Ahora, sea $\{(a_1,b_1),\ldots,(a_{l-\rho},b_{l-\rho})\}$ una base de $U\cap V$, como todos los vectores de U son ceros de $Q(x,y)$, $Q(a_i,b_i)=0$ para todo $i=1,\ldots,l-\rho$, además $b_i=f(a_i)$ por tener que $(a_i,b_i)\in V$. En particular,

$$Q(a_i,b_i)=Q(a_i,f(a_i))=0,\quad i=1,\ldots,l-\rho,$$

por lo que $Q(x,f(x))$ es un polinomio linealizado con $a_1,\ldots,a_{l-\rho}$ como raíces de ese polinomio. Por el Lema 3.16, estas raíces son linealmente independietes. Por lo tanto, $Q(x,f(x))$ es un polinomio linealizado de grado máximo $q^{\tau-1}$ que se anula en un espacio de dimensión $l-\rho$. Si tenemos que

$$l-\rho\geq\tau, \tag{3.43}$$

$Q(x,f(x))$ tendrá más ceros que su grado, por lo que $Q(x,f(x))\equiv 0$. Como en general

$$Q(x,y)=Q_y(y-f(x))+Q(x,f(x)),$$

cuando $Q(x,f(x))\equiv 0$, tenemos que

$$Q(x,y)=Q_y(y-f(x)),$$

con lo que esperamos obtener $y-f(x)$ de $Q(x,y)$. Equivalentemente, esperamos obtener $f(x)$ de la ecuación

$$Q_y(x)\otimes f(x)+Q_x(x)\equiv 0, \tag{3.44}$$

cosa que podemos hacer mediante el Algoritmo **RDiv**.

Es decir, para encontrar un $Q(x,y)$ no nulo se tiene que satisfacer (3.41) y para asegurarnos de que $Q(x,f(x)) \equiv 0$ necesitamos (3.43). Si se satisfacen (3.41) y (3.43) a la vez, decimos que el espacio es decodificable.

Ahora suponemos que U es decodificable. Sustituyendo (3.43) en (3.41), obtenemos la condición $l-\rho+t<2(l-\rho)-k+1$, o lo que es lo mismo

$$\rho + t < l - k + 1, \tag{3.45}$$

luego U decodificable implica (3.45).

Comenzamos al revés, suponiendo que tenemos la condición (3.45). Esto implica que $l-\rho \geq t+k$, o, sumando $l-\rho$ a ambos lados,

$$l - \rho + t + k = r + k \leq 2(l-\rho). \tag{3.46}$$

Como el receptor conoce tanto r como k, si toma

$$\tau = \left\lceil \frac{r+k}{2} \right\rceil, \tag{3.47}$$

se satisface (3.41). Además, sustituyendo (3.47) en (3.46), tenemos que $\tau \leq l-\rho+1/2$, pero como τ, l, ρ son enteros, obtenemos (3.43). Por lo tanto, (3.45) implica que U es decodificable.

Podemos interpolar el polinomio $Q(x,y)$ a través de una base de tamaño r de vectores $(x_1,y_1),\ldots,(x_r,y_r)$ de U, para ello usaremos el algoritmo descrito en [12], que es una adaptación del algoritmo para el problema de interpolación type-Sudan [14].

Para ello, primero, dado un polinomio linealizado en dos variables $f(x,y) = f_x(x) + f_y(y)$ y sea $q^{d_x(f)}$, $q^{d_y(f)}$ su grado para cada variable. Definimos el grado de $f(x,y)$ con peso $(1,k-1)$ como

$$\deg_{1,k-1}(f(x,y)) := \max\{d_x(f), k-1+d_y(f)\}. \tag{3.48}$$

El definir el grado de un polinomio $f(x,y)$ con peso $(1,k-1)$ viene como consecuencia de que $d_x(f)$ puede tomar valor hasta $\tau-1$, mientras que $d_y(f)$ solo tiene grado hasta $\tau-k$. De esta forma tenemos que si $f(x,y)$ tiene grado $\tau-1$ para la variable x y grado $\tau-k$ para la variable y, decimos que tiene el mismo grados para el peso $(1,k-1)$.

Entonces el algoritmo es el siguiente

Algoritmo* 3.2.** ***Algoritmo Interpolación U
Input*: una base* (x_i,y_i), $i=1,\ldots,r$ *de* U.
Output*: un polinomio linealizado en dos variables*
$Q(x,y) = Q_x(x) + Q_y(y)$.
Inicialización $f_0(x,y)=x$, $f_1(x,y)=y$
for $i=1:r$

$\Delta_0 = f_0(x_i, y_i)$; $\Delta_1 = f_1(x_i, y_i)$;
if $\Delta_0 = 0$ ***then***
$f_1(x, y) = f_1^q(x, y) - \Delta_1^{q-1} f_1(x, y)$
else if $\Delta_1 = 0$ ***then***
$f_0(x, y) = f_0^q(x, y) - \Delta_0^{q-1} f_0(x, y)$
else
if $deg_{1,k-1}(f_0) \leq deg_{1,k-1}(f_1)$ ***then***
$f_1(x, y) = \Delta_1 f_0(x, y) - \Delta_0 f_1(x, y)$;
$f_0(x, y) = f_0^q(x, y) - \Delta_0^{q-1} f_0(x, y)$;
else
$f_0(x, y) = \Delta_1 f_0(x, y) - \Delta_0 f_1(x, y)$;
$f_1(x, y) = f_1^q(x, y) - \Delta_1^{q-1} f_1(x, y)$;
endif
endif
end
if $deg_{1,k-1}(f_1) \leq deg_{1,k-1}(f_0)$ ***then***
return $f_1(x, y)$
else
return $f_0(x, y)$
endif

***Ejemplo* 3.5.** *Vamos a dar un ejemplo del funcionamiento del algoritmo. Tomamos el valor* $k = 1$ *y sea* $\{(0,0),(1,1),(2,1)\}$ *una base de* U. *Sea* $(x_1, y_1) = (0,0)$ *el primer elemento de la base, entonces* $\Delta_0 = 0$ *y* $\Delta_1 = 0$. *Recorriendo por primera vez el bucle del algoritmo, obtenemos* $f_0(x, y) = x^q$ *y* $f_1(x, y) = y^q$.

Procedemos ahora a continuar el bucle con el segundo elemento de la base $(x_2, y_2) = (1,1)$, *entonces* $\Delta_0 = 1$ *y* $\Delta_1 = 1$. *Calculamos los grados de los actuales* $f_0(x, y)$ *y* $f_1(x, y)$, *que son* $deg_{1,k-1}(f_0) = 1$ *y* $deg_{1,k-1}(f_1) = 1$. *De esta forma obtenemos* $f_1(x, y) = x^q - y^q$ *y* $f_0(x, y) = x^{q^2} - x^q$.

Por último tomamos el último elemento de la base $(x_3, y_3) = (2,1)$. *Tenemos que* $\Delta_0 = 2^{q^2} - 2^q$ *y* $\Delta_1 = 2^q - 1$. *Los grados de los polinomios* $f_0(x, y)$, $f_1(x, y)$ *actuales son* $deg_{1,k-1}(f_0) = 2$ *y* $deg_{1,k-1}(f_1) = 1$. *De esta manera obtenemos los polinomios* $f_0(x, y) = \left(2^q - 1\right) x^{q^2} + \left(1 - 2^{q^2}\right) x^q + \left(2^{q^2} - 2^q\right) y^q$, $f_1(x, y) = \left(x^q - y^q\right)^q - \left(2^q - 1\right)^{q-1} \left(x^q - y^q\right)$, *concluyendo así con el algoritmo.*

Ambos polinomios cumplen que $f_0(x_1, y_1) = f_0(x_2, y_2) = f_0(x_3, y_3) = 0$, $f_1(x_1, y_1) = f_1(x_2, y_2) = f_1(x_3, y_3) = 0$, *y la última parte del algoritmo, una vez terminado el bucle, nos dice cuál de los dos polinomios tiene menor grado para el peso* $(1, k-1)$, *que será el que devuelva. En el caso de nuestro ejemplo,* $deg_{1,k-1}(f_0) = 2$ *y* $deg_{1,k-1}(f_1) = 2$, *por lo que el algoritmo devuelve* $f_0(x, y)$.

Hemos visto que los polinomios obtenidos para el ejemplo anterior satisfacen la condición de anularse en todos los puntos de la base. Lo último que nos faltaría ahora por ver es que los polinomios que devuelve el algoritmo son los de grado menor posible que cumplen esa condición. Para ello, primero necesitamos poder definir un orden en los polinomios.

Definimos un orden $\prec$ en los polinomios de dos variables de forma que $f(x,y) \prec g(x,y)$ si tenemos que

$$\deg_{1,k-1}(f(x,y)) < \deg_{1,k-1}(g(x,y)). \tag{3.49}$$

En caso de que $\deg_{1,k-1}(f(x,y)) = \deg_{1,k-1}(g(x,y))$, decimos que $f(x,y) \prec g(x,y)$ si

$$d_y(f) + k - 1 < \deg_{1,k-1}(f(x,y)) \tag{3.50a}$$
$$d_y(f) + k - 1 = \deg_{1,k-1}(g(x,y)). \tag{3.50b}$$

En caso de que no tengamos ni (3.49) ni (3.50a)-(3.50b), decimos que $f(x,y)$ y $g(x,y)$ no son comparables. Aunque $\prec$ no es un orden total para los polinomios, nos es suficiente para probar que el Algoritmo 3.2 es correcto. Sin embargo si que nos da un orden total en los monomios, por lo que podemos definir el término líder $lt_{\prec}(f)$ como el monomio más grande bajo el orden $\prec$ del polinomio f (sin tener en cuenta el coeficiente). El siguiente lema nos soluciona el problema de que dos polinomios no sean comparables.

***Lema* 3.6.** *Supongamos que tenemos dos polinomios en dos variables linealizados $f(x,y)$ y $g(x,y)$ que no son comparables para el orden $\prec$. Podemos crear una combinación lineal $h(x,y) = f(x,y) + zg(x,y)$ tal que para una cierta elección de z, $h(x,y) \prec f(x,y)$ y $h(x,y) \prec g(x,y)$.*

Demostración. Si $f(x,y)$ y $g(x,y)$, entonces $lt_{\prec}(f) = lt_{\prec}(g)$. Eligiendo z como menos el cociente de los correspondientes coeficientes de $lt_{\prec}(f)$ y $lt_{\prec}(g)$ tenemos un polinomio $h(x,y)$ tal que $lt_{\prec}(h) \prec lt_{\prec}(f) = lt_{\prec}(g)$. De esta forma eliminamos el término líder común a ambos polinomios y obviamente será menor para el orden establecido. □

Sea A un conjunto de r puntos linealmente independientes $(x_i, y_i) \in W$. Decimos que un polinomio no nulo $f(x,y)$ es x-minimal respecto de A si $f(x,y)$ es el mínimo polinomio para el orden $\prec$ tal que $lt_{\prec}(f) = x^{[d_x(f)]}$ y $f(x,y)$ se anula en los puntos de A. Equivalentemente, decimos que un polinomio no nulo $g(x,y)$ es y-minimal respceto de A si $g(x,y)$ es el mínimo polinomio para el orden $\prec$ tal que $lt_{\prec}(g) = y^{[d_y(g)]}$ y $g(x,y)$ se anula en los puntos de A.

***Teorema* 3.7.** *Los polinomios $f_0(x,y)$ y $f_1(x,y)$ que obtenemos como output del Algoritmo 3.2 son x-minimal e y-minimal, respectivamente, para el conjunto de r puntos independientes $(x_i,y_i) \in W$.*

Demostración. Tenemos que podemos comparar los polinomios x-minial, y-minimal para el orden $\prec$, pues, por definición, tienen distintos monomios líderes. Razonamos por inducción. Comenzamos con los polinomios x, y, que es claro que son minimales respecto del conjunto vacío. Asumimos que, tras j iteraciones del algoritmo de interpolación, $f_0(x,y)$ y $f_1(x,y)$ son x-minimal e y-minimal respecto del conjunto de puntos (x_i,y_i), $i = 1,2,\ldots,j$. Por construcción podemos ver que los polinomios construidos para la siguiente iteración se anulan en el nuevo punto (x_{j+1},y_{j+1}), además de en todos los anteriores. Es decir, se anulan en todos los puntos (x_i,y_i) con $i = 1,\ldots,j+1$.

Separamos ahora por casos. Asumimos que $\Delta_0 \neq 0$ y $\Delta_1 \neq 1$, y que tenemos $f_1(x,y) \prec f_0(x,y)$. Sea

$$f_0'(x,y) = \Delta_1 f_0(x,y) - \Delta_0 f_1(x,y).$$

Entonces $lt_{\prec}(f_0'(x,y)) = lt_{\prec}(f_0(x,y))$, y como $f_0(x,y)$ era minimal, también lo será $f_0'(x,y)$. Sea

$$f_1'(x,y) = f_1^q(x,y) - \Delta_1^{q-1} f_1(x,y).$$

Queremos ahora ver que $f_1'(x,y)$ es y-minimal para los puntos (x_i,y_i), con $i = 1,\ldots,j+1$. Razonamos por reducción al absurdo, suponiendo que $f_1'(x,y)$ no es y-minimal, entonces existiría un polinomio $f_1''(x,y)$ y-minimal para los puntos (x_i,y_i), con $i = 1,\ldots,j+1$ tal que $f_1''(x,y) \neq f_1(x,y)$ y con el mismo término líder de que $f_1(x,y)$. Ambos polinomios son distintos, pues $f_1''(x,y)$ se anula en (x_{j+1},y_{j+1}) mientras que $f_1(x,y)$ no. Pero entonces podríamos encontrar un polinomio $h(x,y)$ como combinación lineal de $f_1(x,y)$ y $f_1''(x,y)$, menor que $f_0(x,y)$ y $f_1(x,y)$ para el orden $\prec$ y que se anula en todos los puntos (x_i,y_i) con $i = 1,\ldots,j$, lo que contradice la minimalidad de $f_0(x,y)$ y $f_1(x,y)$. Podemos hacer uso del mismo razonamiento para el caso en el que $f_0(x,y) \prec f_1(x,y)$.

Consideramos ahora el caso en el que $\Delta_0 = 0$ y $\Delta_1 \neq 0$. Para este caso, al no modificar $f_0(x,y)$, tenemos su x-minimalidad por la iteración anterior, por lo que solo hay que ver que $f_1'(x,y) = f_1^q(x,y) - \Delta_1^{q-1} f_1(x,y)$ es y-minimal para los puntos (x_i,y_i), con $i = 1,\ldots,j+1$. Razonamos de nuevo por reducción al absurdo. Existe un polinomio $f_1''(x,y) \neq f_1(x,y)$ con el mismo término líder que $f_1(x,y)$. Ambos polinomios son distintos, pues $f_1''(x,y)$ se anula en (x_{j+1},y_{j+1}) mientras que $f_1(x,y)$ no. De nuevo, podemos elegir una combinación lineal $h(x,y)$ de $f_1''(x,y)$ y $f_1(x,y)$, menor

que $f_1(x,y)$ para el orden $\prec$. Si el término líder de $h(x,y)$ es $y^{[d_y(h)]}$ o tenemos que $h(x,y) \prec f_0(x,y)$, llegamos a una contradicción negando la y-minimalidad de $f_1(x,y)$ o la x-minimalidad de $f_0(x,y)$. De lo contrario, $h(x,y)$ no se anularía en (x_{j+1}, y_{j+1}) y por tanto no sería un múltiplo, para la operación $\otimes$, de $f_0(x,y)$. Luego, para una cierta elección de t, podemos encontrar un polinomio $h''(x,y)$ como combinación lineal de $h(x,y)$ y $x^{[t]} \otimes f_0(x,y)$, menor que $h(x,y)$ para el orden $\prec$. Si repetimos este procedimiento, obtendremos un polinomio $\hat{h}(x,y)$ que tiene a $y^{[d_y(\hat{h})]}$ como término líder, o que es menor que $f_0(x,y)$ para el orden $\prec$, contradiciendo así la y-minimalidad de $f_1(x,y)$ o la x-minimalidad de $f_0(x,y)$. El caso en el que $\Delta_0 \neq 0$ y $\Delta_1 = 0$ se razona usando de nuevo los mismos argumentos. □

De esta forma, por el Teorema 3.7, el Algoritmo 3.2 resuelve el problema de encontrar un polinomio en dos variables linealizado $Q(x,y)$ de grado mínimo $\tau - 1$ para el peso $(1, k-1)$, que será el polinomio $f_0(x,y)$ ó $f_1(x,y)$ (obtenidos en el Algoritmo 3.2) que tenga menor grado para el peso $(1, k-1)$.

En resumen, suponemos que se envía $V \in C$ a través del canal de transmisión descrito en la primera sección del capítulo, y suponemos que se recibe un espacio U de W de dimensión $(l - \rho + t)$. Para decodificar el espacio transmitido tenemos que

1. Utilizar el Algoritmo 3.2 para encontrar el polinomio linealizado $Q(x,y) = Q_x(x) + Q_y(y)$ de mínimo grado para el peso $(1, k-1)$ que se anula en U.

2. Utilizar el Algoritmo **RDiv** para obtener un polinomio $f(x)$ tal que $-Q_x(x) \equiv Q_y(x) \otimes f(x)$. Si no es posible encontrar tal $f(x)$ no podemos decodificar U.

3. Devolver $f(x)$ como el polinomio que se corresponde con la palabra $\hat{V} \in C$ si $d(U, \hat{V}) < l - k + 1$.

Para concluir con este trabajo, vamos a mostrar un ejemplo en el que un emisor codificará un mensaje haciendo uso del proceso de codificación descrito en la subsección 3.4.1. Simularemos que el subespacio generado V se transmite por un canal de transmisión dos veces. En la primera transmisión, el subespacio recibido U contiene un error y un borrón, y, al no cumplir la condición (3.45), el espacio U no será decodificable. Para la segunda transmisión suponemos que el espacio recibido tiene un borrón únicamente, luego al cumplirse (3.45), podremos decodificar U.

***Ejemplo* 3.6.** *Consideramos el cuerpo $\mathbb{F}_q = \mathbb{F}_2$, y la extensión $\mathbb{F}_{q^m} = \mathbb{F}_{2^5} = \mathbb{F}_{32}$ de $\mathbb{F}_2$. Veremos $\mathbb{F} = \mathbb{F}_{32}$ como un espacio vectorial de dimensión 5 sobre $\mathbb{F}_2$, es decir,*

$$\mathbb{F} : (a,b,c,d,e) \longrightarrow a\alpha^4 + b\alpha^3 + c\alpha^2 + d\alpha + e,$$

con $a,b,c,d,e \in \mathbb{F}_2$, $\alpha^5 + \alpha^2 + 1$ el polinomio mínimo y α un elemento primitivo de $\mathbb{F}$.

Elegimos $A = \{a_1, a_2, a_3, a_4\} \subset \mathbb{F}$ un conjunto de elementos linealmente independientes en este espacio vectorial, donde

$$a_1 = 1, \quad a_2 = \alpha + 1, \quad a_3 = \alpha^3, \quad a_4 = \alpha^4 + \alpha.$$

Estos elemento generan un espacio vectorial $\langle A \rangle$ de dimensión $l = 4$ sobre $\mathbb{F}_2$. Podemos ver fácilmente que este subespacio está formado por 16 elementos:

$$\begin{aligned}\langle A \rangle = \{&0, 1, \alpha, \alpha+1, \alpha^3, \alpha^3+1, \alpha^3+\alpha, \alpha^3+\alpha+1, \alpha^4, \alpha^4+1, \alpha^4+\alpha,\\ &\alpha^4+\alpha+1, \alpha^4+\alpha^3, \alpha^4+\alpha^3+1, \alpha^4+\alpha^3+\alpha, \alpha^4+\alpha^3+\alpha+1\}.\end{aligned}$$

Queremos ahora transmitir un mensaje con $k = 3$ elementos,

$$u = (u_0, u_1, u_2) = (\alpha, \alpha^2 + \alpha, \alpha^3 + \alpha^2 + 1).$$

Para ello construimos el polinomio linealizado $f(x) = \sum_{i=1}^{k-1} u_i x^{[i]}$, es decir,

$$f(x) = \alpha x^{[0]} + (\alpha^2 + \alpha) x^{[1]} + (\alpha^3 + \alpha^2 + 1) x^{[2]}.$$

Ayudándonos ahora de Singular, obtenemos las evaluaciones del polinomio $f(x)$ en los elementos a_i definidos anteriormente, obteniendo así

$$\begin{aligned}\beta_1 &= f(a_1) = \alpha^3 + 1,\\ \beta_2 &= f(a_2) = \alpha^4 + \alpha^3 + \alpha + 1,\\ \beta_3 &= f(a_3) = \alpha^2 + 1,\\ \beta_4 &= f(a_4) = \alpha.\end{aligned}$$

De esta forma generamos el espacio V de dimensión 4, formado por los distintos pares (a_i, β_i), $i = 1, \ldots, 4$,

$$V = \left\langle (1, \alpha^3 + 1), (\alpha + 1, \alpha^4 + \alpha^3 + \alpha + 1), (\alpha^3, \alpha^2 + 1), (\alpha^4 + \alpha, \alpha) \right\rangle.$$

Ahora suponemos que enviamos el espacio V a través de un canal de transmisión, y recibimos el espacio U,

$$U = \left\langle (1+\alpha^3, \alpha^3+\alpha^2), (\alpha+1, 1), (\alpha^4+\alpha^3+\alpha, \alpha^2+\alpha+1), (\alpha^4+\alpha+1, \alpha^3+\alpha+1) \right\rangle,$$

donde vemos que el primer generador de U es la suma del primer y tercer generador de V, el tercero es la suma del tercer y cuarto generador de V, el cuarto es la suma del primer y cuarto generador de V, y el segund generador de U no es combinación lineal de nlos generadores de V.

Este es un espacio de dimensión $l+t-\rho = 4$, en el que se ha producido un borrón ($\rho = 1$) y un error ($t = 1$). Nuestro objetivo es recuperar el espacio V, pero como $\rho + t = 2 \not< D/2 = l-k+1 = 2$, el Teorema 3.1 no garantiza que podamos recuperar el espacio enviado, y vemos que efectivamente es así.

Comenzamos aplicando el algoritmo de interpolación U, que dada una base de U, nos devuelve un polinomio linealizado en dos variables $Q(x,y) = Q_x(x) + Q_y(y)$. Tras ayudarnos de Singular para correr el algoritmo, tenemos que éste devuelve el polinomio linealizado

$$Q(x,y) = (\alpha^3+\alpha^2+\alpha+1)x^{[3]} + (\alpha^4+\alpha^3)x^{[2]} + (\alpha^4+\alpha^3+\alpha)x^{[1]} + (\alpha^4+\alpha)y^{[1]},$$

con

$$\begin{aligned} Q_x(x) &= (\alpha^3+\alpha^2+\alpha+1)x^{[3]} + (\alpha^4+\alpha^3)x^{[2]} + (\alpha^4+\alpha^3+\alpha)x^{[1]} \\ Q_y(y) &= (\alpha^4+\alpha)y^{[1]}, \end{aligned}$$

que vemos que se anula en U.

*Ahora necesitaremos hacer uso del Algoritmo **RDiv** para obtener un polinomio $f'(x)$ tal que $-Q_x(x) \equiv Q_y(x) \otimes f'(x)$. Tomando $a(x) = Q_x(x)$, $b(x) = Q_y(x)$, el algoritmo nos devuelve*

$$\begin{aligned} q_r(x) &= (\alpha^3+\alpha^2+\alpha)x^{[2]} + (\alpha^2+\alpha+1)x^{[1]} + (\alpha^4+\alpha^3+a)x^{[0]}, \\ r_R(x) &= 0, \end{aligned}$$

tal que $a(x) = b(x) \otimes q_R(x) + r_R(x)$, $f'(x) = q_R(x)$.

Los coeficientes de $q_R(x)$ deberían ser los mensajes enviados por el emisor, cosa que no es así, luego no vamos a poder encontrar una palabra del código $\hat{V}$ tal que $d(U,\hat{V}) < l-k+1$ y por tanto no somos capaces de decodificar U.

Para la segunda transmisión, suponemos que se vuelve a enviar V a través del canal, y esta vez el espacio U recibido es

$$U = \left\langle (1, \alpha^3+1), (\alpha^3+1, \alpha^3+\alpha^2), (\alpha^4+\alpha^3+\alpha+1, \alpha^3+\alpha^2+\alpha) \right\rangle,$$

donde los generadores de U son, todos ellos, combinaciones lineales del primer, tercer y cuarto generador de V.

En este caso, $\rho = 1$, $t = 0$, *luego* $t + \rho = 1 < D/2 = l - k + 1 = 2$, *por lo que en esta vez si deberíamos ser capaces de recuperar* V.

De nuevo, hacemos uso del algoritmo de interpolación sobre U, *para obtener de esta forma* $Q(x, y) = Q_x(x) + Q_y(y)$, *apoyándonos en Singular. Obtenemos así*

$$Q(x,y) = (\alpha^2+\alpha)x^{[2]} + (\alpha^4+\alpha)x^{[1]} + (\alpha^3+\alpha^2+\alpha)x^{[0]} + (\alpha^2+\alpha+1)y^{[0]},$$

con

$$\begin{aligned} Q_x(x) &= (\alpha^2+\alpha)x^{[2]} + (\alpha^4+\alpha)x^{[1]} + (\alpha^3+\alpha^2+\alpha)x^{[0]}, \\ Q_y(y) &= (\alpha^2+\alpha+1)y^{[0]}. \end{aligned}$$

Vemos que el polinomio $Q(x, y)$ *se anula en* U.

Finalmente, volviendo a hacer uso del Algoritmo ***RDiv****, obtendremos un polinomio* $f'(x)$ *tal que* $-Q_x(x) \equiv Q_y(x) \otimes f'(x)$.

Tomando $a(x) = Q_x(x)$, $b(x) = Q_y(x)$, *obtenemos*

$$\begin{aligned} q_R(x) &= (\alpha^3+\alpha+1)x^{[2]} + (a^2+a)x^{[1]} + ax^{[0]}, \\ r_R(x) &= 0, \end{aligned}$$

tal que $a(x) = b(x) \otimes f'(x)$, *con* $f'(x) = q_R(x)$.

Esta vez, los coeficientes del polinomio $f'(x)$ *coinciden con los del polinomio* $f(x)$ *que codificamos, luego hemos sido capaces de recuperar el mensaje*

$$u = (u_0, u_1, u_2) = (\alpha^3+\alpha^2+1, \alpha^2+\alpha, \alpha),$$

y de esta forma podemos recuperar la palabra del código $\hat{V}$ *tal que* $d(U, \hat{V}) < l - k + 1 = 3$. *Por lo tanto, para este caso hemos sido capaces de decodificar* U.

Todos las operaciones realizadas a través de Singular para este ejemplo las podemos encontrar en el Capítulo 4.

Capítulo 4

Implementación en Singular

En este capítulo mostramos el código de los distintos procedimientos creados en Singular para la realización de los distintos ejemplos a lo largo del trabajo.

La primera sección del capítulo muestra los dos procedimientos creados para modelar el grafo dirigido, que consisten en la obtención de las distintas matrices A, F y $B^{(r_i)}$, y el cálculo, a través de ellas, del polinomio de transferencia.

La segunda sección nos muestra la implementación del Algoritmo de Jaggi, definido en la parte final del Capítulo 1, junto con una serie de procedimientos auxiliares para simplificar su funcionamiento.

En la tercera sección del capítulo podemos ver distintos procedimientos utilizados para la obtención de las cotas del Capítulo 2.

Por último, las dos últimas secciones muestran el código de los procedimientos creados en Singular que hemos construido para el Ejemplo 3.6, en el que mostramos la codificación para los canales del transmisión y su posterior decodificación. Estos procedimientos son: la codificación del espacio V, la interpolación para obtener el polinomio $Q(x,y)$ a partir del espacio U y el algoritmo RDiv que nos permite realizar la división por la derecha en términos de polinomios linealizados. Para los tres procedimientos definimos previamente en Singular el anillo sobre el que estamos trabajando, junto con su polinomio mínimo.

4.1. Procedimientos de modelización de la red y polinomio de transferencia

En esta sección vemos tres procedimientos relacionados con la modelización de la red y el polinomio de transferencia.

El primer procedimiento, llamado *parringRNC*, nos proporciona el nú-

mero total de variables que vamos a tener en las matrices A, F y $B^{(r_i)}$ asociadas al grafo, definidas en la Sección 1.2 del Capítulo 1, para poder construir el anillo al que pertenece el polinomio de transferencia. Los parámetros de entrada son el entero q sobre el que construimos el cuerpo y el sistema de flujo asociado al grafo en forma de lista. El algoritmo nos devuelve el número de variables x_j asociadas a la matriz F, el número de variables y_k asociadas a la matriz A y el número de variables z_l asociadas a las matrices $B^{(r_i)}$.

```
proc parringRNC(int q, list Fl){
int numvar=0;
int r=size(Fl);
int m=size(Fl[1]);
for(int i=1;i<=r;i++){
for(int j=1;j<=m;j++){
numvar=numvar+size(Fl[i][j])-1;}
}
return(numvar,m^2,r*m^2);
}
```

El siguiente procedimiento que veremos en esta sección es el procedimiento *matRNC*, que dado el conjunto de vértices V en formato vector de enteros, el conjunto de aristas E en formato lista, donde los elementos de la lista son pares de vértices, y el sistema de flujo $\mathcal{F}$ en formato lista, nos devuelve las matrices A, F y la lista de matrices $B^{(r_i)}$ en las variables x_j, y_k, y z_l creadas anteriormente, que hemos estudiado en la Sección 1.2 del Capítulo 1.

```
proc matRNC(intvec V, list E, list Fl){
int numvar=0;
int l=1;
int r=size(Fl);
int m=size(Fl[1]);
for(int i=1;i<=r;i++){
for(int j=1;j<=m;j++){
numvar=numvar+size(Fl[i][j])-1;}
}
matrix F[size(E)][size(E)];
matrix A[m][size(E)];
list B;
for(int i=1;i<=r;i++){
for(int j=1;j<=m;j++){
for(int k=1;k<=(size(Fl[i][j])-1);k++){
F[Fl[i][j][k],Fl[i][j][k+1]]=x(l);
l=l+1;}
```

```
}
}
l=1;
for(int i=1;i<=m;i++){
for(int j=1;j<=m;j++){
A[i,j]=y(l);
l=l+1;}
}
l=1;
for(int i=1;i<=r;i++){
matrix Br[size(E)][size(Fl)];
for(int j=1;j<=m;j++){
for(int k=1;k<=r;k++){
Br[Fl[i][j][size(Fl[i][j])],k]=z(l);
l=l+1;}
}
B[i]=Br;
}
return(A,F,B);
}
```

El tercer procedimiento de esta sección es el procedimiento *poltransfRNC*, que nos permite calcular el polinomio de transferencia asociado a la red. Los parámetros de entrada son las matrices A, F y la lista de las matrices $B^{(r_i)}$, es decir, la salida de *matRNC*. El algoritmo nos devuelve el polinomio de transferencia, estudiado en la Sección 1.3 del Capítulo 1.

```
proc poltransfRNC(matrix A, matrix F, list B){
int r=size(B);
int m=nrows(A);
int e=nrows(F);
list kkk;
matrix Mr[m+e][m+e];
for(int i=1;i<=m;i++){
for(int j=1;j<=m;j++){
Mr[i,j]=A[i,j];}
}
for(int i=1;i<=e;i++){
for(int j=1;j<=e;j++){
Mr[i+m,j]=-F[i,j];}
}
for(int i=1;i<=e;i++){
Mr[i+m,i]=1;}
```

```
poly poltransf=1;
for(int i=1;i<=r;i++){
matrix tmp=Mr;
for(int j=1;j<=e;j++){
for(int k=1;k<=m;k++){
tmp[j+m,k+e]=B[i][j,k];}
}
poltransf=poltransf*det(tmp);}
return(poltransf);
}
```

Por último, para concluir esta sección, vemos un ejemplo en el que empleamos los tres procedimientos anteriores para obtener el polinomio de transferencia de la Red de Mariposa que podemos encontrar en la Figura 1.1 y en el ejemplo que podemos ver en las páginas 11 y 17.

```
> intvec V=0,1,2,3,4,191,192;
> int q=2;
> list E;
> intvec e1=0,1;
> intvec e2=0,3;
> intvec e3=1,2;
> intvec e4=3,2;
> intvec e5=1,191;
> intvec e6=2,4;
> intvec e7=3,192;
> intvec e8=4,191;
> intvec e9=4,192;
> E=e1,e2,e3,e4,e5,e6,e7,e8,e9;
> intvec fl11=1,5;
> intvec fl12=2,4,6,8;
> intvec fl21=1,3,6,9;
> intvec fl22=2,7;
> list Fl1=fl11,fl12;
> list Fl2=fl21,fl22;
> list Fl=Fl1,Fl2;
> parringRNC(q,Fl);
8 4 8
> ring R=(2,a),(x(1..8),y(1..4),z(1..8)),lp;
> matrix Ar;
> matrix Fr;
> list Br;
> (Ar,Fr,Br)=matRNC(V,E,Fl);
```

```
> print(Ar);
y(1),y(2),0,0,0,0,0,0,0,
y(3),y(4),0,0,0,0,0,0,0
> print(Fr);
0,0,x(5),0,   x(1),0,   0,   0,   0,
0,0,0,   x(2),0,   0,   x(8),0,   0,
0,0,0,   0,   0,   x(6),0,   0,   0,
0,0,0,   0,   0,   x(3),0,   0,   0,
0,0,0,   0,   0,   0,   0,   0,   0,
0,0,0,   0,   0,   0,   0,   x(4),x(7),
0,0,0,   0,   0,   0,   0,   0,   0,
0,0,0,   0,   0,   0,   0,   0,   0,
0,0,0,   0,   0,   0,   0,   0,   0
> print(Br[1]);
0,   0,
0,   0,
0,   0,
0,   0,
z(1),z(2),
0,   0,
0,   0,
z(3),z(4),
0,   0
> print(Br[2]);
0,   0,
0,   0,
0,   0,
0,   0,
0,   0,
0,   0,
z(7),z(8),
0,   0,
z(5),z(6)

> matrix A[2][9];
> A[1,1]=1;
> A[2,2]=1;
> matrix B1[9][2];
> B1[5,1]=1;
> B1[5,2]=1;
> B1[8,2]=1;
> matrix B2[9][2];
```

```
> B2[7,2]=1;
> B2[9,1]=1;
> B2[9,2]=2;
> B2[9,2]=1;
> list BB;
> BB[1]=B1;
> BB[2]=B2;
> matrix F[9][9];
> F[1,3]=x(5);
> F[1,5]=x(1);
> F[2,4]=x(2);
> F[2,7]=x(8);
> F[3,6]=x(6);
> F[4,6]=x(3);
> F[6,8]=x(4);
> F[6,9]=x(7);
> poltransfRNC(A,F,BB);
x(1)*x(2)*x(3)*x(4)*x(5)*x(6)*x(7)*x(8)
```

4.2. Procedimiento Jaggi

En esta sección veremos la implementación del Algoritmo de Jaggi (Algoritmo 1.1 en la Sección 1.5) en formato procedimiento, junto con una serie de algoritmos auxiliares. Para este procedimiento debemos definir previamente en Singular el anillo sobre el que queremos trabajar. Comenzaremos describiendo los distintos algoritmos auxiliares contruidos.

El primer procedimiento es el procedimiento *posicionesnulas*, que dada una matriz M y un entero i nos dice el número de posiciones no nulas de la columna i de la matriz M.

```
proc posicionesnulas(matrix M, int i){
int contador=0;
for(int j=1;j<=nrows(M);j++){
if(M[j,i]!=0){
contador=contador+1;}
}
return(contador);
}
```

El siguiente procedimiento es el procedimiento *cualesson*, que dada una matriz M y un entero i nos dice las posiciones no nulas de la columna i de la matriz M, en formato lista.

```
proc cualesson(matrix M,int i){
intvec estasson;
for(int j=1; j<=nrows(M); j++){
if(M[j,i]!=0){
estasson=estasson,j;}
}
return(estasson[2..size(estasson)]);
}
```

El tercer procedimiento auxiliar es el procedimiento *anterior*, que dado un sistema de flujo $\mathcal{F}$ y un elemento i, es decir, una arista, del sistema de flujo, nos dice cual es la arista que la precede. Como este procedimiento lo utilizamos para aristas que sabemos que están una única vez en el sistema de flujo, el procedimiento devuelve la primera que encuentre, no calcula todas las que hay, en caso de haber más de una.

```
proc anterior(list Fl, int i){
int t;
int k;
int l;
```

```
for(int j=1;j<=size(Fl);j++){
for(k=1;k<=size(Fl[j]);k++){
for(l=1;l<=size(Fl[j][k]);l++){
if(Fl[j][k][l]==i){
return(Fl[j][k][l-1]);}
}
}
}
}
```

El cuarto procedimiento auxiliar es el procedimiento *ayudacomprobar*, que lo empleamos en el procedimiento auxiliar *comprobardim* que veremos a continuación. Este procedimiento nos dice, dado un sistema de flujo $\mathcal{F}$ y un entero i que define una arista, qué elementos del sistema de flujo hay que comprobar que generen todo el espacio.

```
proc ayudacomprobar(list Fl,int i){
intvec devolver;
int k;
int l;
for(int j=1;j<=size(Fl);j++){
for(k=1;k<=size(Fl[j]);k++){
for(l=1;l<=size(Fl[j][k]);l++){
if(Fl[j][k][l]==i){
devolver=devolver,j;}
}
}
}
return(devolver[2..size(devolver)]);
}
```

El último procedimiento auxiliar del Algoritmo de Jaggi, que hace uso del procedimiento *ayudacomprobar*, es el procedimiento *comprobardim*, que dado un sistema de flujo $\mathcal{F}$ y un entero i que representa una arista, nos dice cuales son los elementos, junto con el asociado a la arista i, que generan el nuevo espacio B_r. El procedimiento devuelve una lista. En caso de que la arista forme parte de varios flujos del sistema de flujo, el procedimiento nos devuelve la lista con los elementos de B_r (menos el asociado a la arista), para cada receptor r tal que la arista i forme parte del flujo asociado a r.

```
proc comprobardim(list Fl, int i){
list compruebaestos;
intvec ayuda=ayudacomprobar(Fl,i);
int k;
```

```
for(int j=1;j<=size(ayuda);j++){
list tmp;
for(k=1;k<=size(Fl[ayuda[j]]);k++){
int l=1;
while(Fl[ayuda[j]][k][l]<i){
if(l==size(Fl[ayuda[j]][k])){
Fl[ayuda[j]][k][l+1]=i+1;}
l=l+1;}
if(Fl[ayuda[j]][k][l]>i){
tmp=tmp,Fl[ayuda[j]][k][l-1];}
}
compruebaestos[j]=tmp[2..size(tmp)];}
return(compruebaestos);
}
```

Finalmente, con todos estos procedimientos auxiliares, podemos construir el Algoritmo de Jaggi (Algoritmo 1.1). Los parámetros de entrada son el sistema de flujo $\mathcal{F}$ asociado al grafo, la matriz de incidencia I del grafo, que es la matriz F contruida en la sección anterior con todas las variables iguales a 1, y el entero q que nos dice sobre que cuerpo finito $\mathbb{F}_q$ estamos trabajando. El algoritmo devuelve los coeficientes $f_{i,j}$ en forma de matriz.

```
proc Jaggi(list Fl, matrix I, int q){
int r=size(Fl);
int m=size(Fl[1]);
list B;
for(int i=1;i<=m;i++){
matrix tmp[1][m];
tmp[1,i]=1;
B[i]=tmp;}
int e=nrows(I);
matrix coefF[e][e];
for(int i=1;i<=e;i++){
if(posicionesnulas(I,i)==0){
coefF[1..e,i]=I[1..e,i];}
if(posicionesnulas(I,i)==1){
coefF[1..e,i]=a*I[1..e,i];
B[i]=B[anterior(Fl,i)];}
if(posicionesnulas(I,i)>=2){
intvec alea=cualesson(I,i);
poly comprobardependencia=0;
while(comprobardependencia==0){
comprobardependencia=1;
```

```
for(int j=1;j<=size(alea);j++){
coefF[alea[j],i]=random(0,1)*a^random(1,q-1);}
matrix auxx[1][m];
B[i]=auxx;
for(int j=1;j<=size(alea);j++){
B[i]=B[i]+coefF[alea[j],i]*B[j];}
list conquien=comprobardim(Fl,i);
for(int k=1;k<=size(conquien);k++){
matrix span[m][m];
for(int l=1;l<=size(conquien[k]);l++){
span[l,1..m]=B[conquien[k][l]];}
span[m,1..m]=B[i];
comprobardependencia=comprobardependencia*det(span);}
}
}
}
return(coefF);
}
```

Este procedimiento ha sido probado para el ejemplo de la Red de Mariposa (Figura 1.1) y para la red representada en la Figura 1.7, llegando para ambos casos a una solución en $\mathbb{F}_2$, como podemos ver en la Sección 1.5.

```
\\Ejemplo Red Mariposa (Figura 1.1)
> ring R=(2,a),x,lp;
> matrix I[9][9];
> I[1,3]=1;
> I[1,5]=1;
> I[2,4]=1;
> I[2,7]=1;
> I[3,6]=1;
> I[4,6]=1;
> I[6,8]=1;
> I[6,9]=1;
> intvec F11=1,5;
> intvec F12=2,4,6,8;
> intvec F21=1,3,6,9;
> intvec F22=2,7;
> list F1=F11,F12;
> list F2=F21,F22;
> list Fl=F1,F2;
> matrix Jag=Jaggi(Fl,I,2);
> print(Jag);
```

```
0,0,(a),0,  (a),0,  0,  0,  0,
0,0,0,  (a),0,  0,  (a),0,  0,
0,0,0,  0,  0,  (a),0,  0,  0,
0,0,0,  0,  0,  (a),0,  0,  0,
0,0,0,  0,  0,  0,  0,  0,  0,
0,0,0,  0,  0,  0,  0,  (a),(a),
0,0,0,  0,  0,  0,  0,  0,  0,
0,0,0,  0,  0,  0,  0,  0,  0,
0,0,0,  0,  0,  0,  0,  0,  0

\\Ejemplo Figura 2.1
> ring R=(2,a),x,lp;
> matrix I[8][8];
> I[1,3]=1;
> I[1,4]=1;
> I[2,3]=1;
> I[2,4]=1;
> I[3,5]=1;
> I[3,6]=1;
> I[3,7]=1;
> I[3,8]=1;
> I[4,5]=1;
> I[4,6]=1;
> I[4,7]=1;
> I[4,8]=1;
> intvec F11=1,3,5;
> intvec F12=2,4,6;
> intvec F21=1,3,7;
> intvec F22=2,4,8;
> list F1=F11,F12;
> list F2=F21,F22;
> list F=F1,F2;
> matrix Jag2=Jaggi(F,I,2);
> print(Jag2);
0,0,(a),(a),0,  0,  0,  0,
0,0,(a),0,  0,  0,  0,  0,
0,0,0,  0,  0, (a),(a),0,
0,0,0,  0, (a),(a),(a),(a),
0,0,0,  0,  0,  0,  0,  0,
0,0,0,  0,  0,  0,  0,  0,
0,0,0,  0,  0,  0,  0,  0,
0,0,0,  0,  0,  0,  0,  0
```

4.3. Procedimientos relativos al cálculo de cotas

En esta sección veremos cómo calcular las distintas cotas que hemos ido utilizando en los ejemplos realizados a lo largo del Capítulo 2.

Comenzamos describiendo los procedimientos creados para el cálculo de la cota P_{Ho2}, y las cotas P_{FP1}, P_{FP2}, que podemos encontrar, todas ellas, en la Sección 2.1.

Para calcular la cota P_{Ho2}, que podemos ver en la página 33, hemos creado el procedimiento *Ho2RNC*, que tiene como parámetros de entrada q, que lo iremos variando para obtener las distintas cotas para los cuerpos $\mathbb{F}_q$, el entero ν, descrito en la página 31, y el número de receptores $|R|$. Para obtener ν, en caso de necesitar calcularlo, hacemos uso del procedimiento *poltransfRNC* un número total de $|R|$ veces, donde en vez de tomar como input la lista con todas las matrices $B^{(r)}$, tomamos la lista con una única matriz $B^{(r)}$ cada vez. De esta forma, el procedimiento *poltransfRNC* nos devuelve los polinomios de transferencia asociados a cada receptor. Sustituyendo los coeficientes a priori fijados, ν será el máximo del número de incógnitas restantes para todos los polinomios asociados al flujo a cada receptor.

```
proc Ho2RNC(number q, int nu, number r){
number Ho2=((q-r)/q)^nu;
return(Ho2);
}
```

Para calcular las cotas P_{FP1} y P_{FP2}, que podemos ver en la página 30, hemos creado los procedimientos *FP1RNC* y *FP2RNC*, que tienen como input el polinomio de transferencia con los coeficientes elegidos a priori ya sustituidos y el entero q para el cuerpo $\mathbb{F}_q$ elegido.

```
proc FP2RNC(int q, poly poltransf){
intvec prov=leadexp(poltransf);
number FP2=1;
for(int i=1;i<=size(prov);i++){
FP2=FP2*(q-prov[i])/q;}
return(FP2);
}

proc FP1RNC(int q, poly poltransf){
matrix M=coef(poltransf,x(1));
matrix mon[1][ncols(M)];
poly prod;
int j;
for(int i=1;i<=ncols(M);i++){
```

```
prod=1;
for(j=1;j<=nrows(M);j++){
prod=prod*M[j,i];}
mon[1,i]=prod;}
matrix FP2[1][ncols(M)];
number prov;
for(i=1;i<=ncols(M);i++){
prov=1;
for(j=1;j<=size(leadexp(mon[1,i]));j++){
prov=prov*(q-leadexp(mon[1,i])[j])/q;}
FP2[1,i]=prov;}
poly FP;
if(ncols(FP2)>1){
for(i=1;i<=ncols(FP2)-1;i++){
FP=min(FP2[1,i],FP2[1,i+1]);}
return(FP);}
return(FP2[1,1]);
}
```

Hemos hecho uso de estos procedimientos para obtener los valores dados en la Tabla 2.1 (página 37), como puede verse a continuación.

```
> ring R=0,x(1..6),lp;

\\calculo Ho2
> int nu=3;
> number r=2;
> Ho2RNC(4,nu,r);
1/8
> Ho2RNC(8,nu,r);
27/64
> Ho2RNC(16,nu,r);
343/512
> Ho2RNC(132,nu,r);
274625/287496
> Ho2RNC(32,nu,r);
3375/4096
> Ho2RNC(64,nu,r);
29791/32768

\\calculo FP2
> poly poltransf=x(1)*x(2)*x(3)*x(4)*x(5)*x(6);
> FP2RNC(4,poltransf);
```

```
729/4096
> FP2RNC(8,poltransf);
117649/262144
> FP2RNC(16,poltransf);
11390625/16777216
> FP2RNC(32,poltransf);
887503681/1073741824
> FP2RNC(64,poltransf);
62523502209/68719476736

\\calculo FP1
> poly poltransf=x(1)*x(2)*x(3)*x(4)*x(5)*x(6);
> FP1RNC(4,poltransf);
729/4096
> FP1RNC(8,poltransf);
117649/262144
> FP1RNC(16,poltransf);
11390625/16777216
> FP1RNC(32,poltransf);
887503681/1073741824
> FP1RNC(64,poltransf);
62523502209/68719476736
```

A continuación, hemos implementado las cotas P_{FB1} y P_{FB2}, que podemos encontrar en la Sección 2.2, relacionadas con el flujo. Para ello, hemos creado un procedimiento auxiliar *RFRNC*, que, dada una lista de aristas E y una lista con el sistema de flujo $\mathcal{F}$, nos devuelve $|R_{\mathcal{F}}(e)|$ para cada $e \in E$, en forma de matriz $1 \times e$.

```
proc RFRNC(list E,list Fl){
int e=size(E);
matrix RF[1][e];
int tmp;
int i;
int j;
int k;
for(int l=1;l<=e;l++){
tmp=0;
for(i=1;i<=size(Fl);i++){
for(j=1;j<=size(Fl[i]);j++){
for(k=1;k<=size(Fl[i][j]);k++){
if(Fl[i][j][k]==l){
```

```
tmp=tmp+1;}
}
}
}
RF[1,l]=tmp;}
return(RF);
}
```

Haciendo uso de este procedimiento auxiliar, creamos el procedimiento *FBRNC*, que, dado un entero q, pues trabajamos sobre el cuerpo $\mathbb{F}_q$, una lista de aristas E, una lista con el sistema de flujo $\mathcal{F}$ y una matriz in de tamaño $1 \times E$, cuyas entradas $in(e)$ son el número de mensajes generados en e más el número de aristas que llegan a e. Las salidas del procedimiento son las cotas P_{FB1} y P_{FB2}, que podemos encontrar en la página 41.

```
proc FBRNC(int q, list E, list Fl, matrix in){
poly FB1=1;
poly FB2=1;
matrix RF=RFRNC(E,Fl);
for (int i=1;i<=ncols(RF);i++){
FB1=FB1*((q-RF[1,i])/q);
FB2=FB2*((q-RF[1,i])/q+(RF[1,i]-1)/(q^int(leadcoef(in[1,i]))));}
return(FB1,FB2);
}
```

Hemos hecho uso de este procedimiento para calcular las cotas P_{FB1}, P_{FB2} de la Tabla 2.6 (página 53), relativas a la Figura 2.3.

```
> ring R=0,x,lp;
> list E;
> intvec e1=0,1;
> intvec e2=0,2;
> intvec e3=0,3;
> intvec e4=1,191;
> intvec e5=3,192;
> intvec e6=1,4;
> intvec e7=2,4;
> intvec e8=2,5;
> intvec e9=3,5;
> intvec e10=4,6;
> intvec e11=5,6;
> intvec e12=4,191;
> intvec e13=6,191;
> intvec e14=6,192;
```

```
> intvec e15=5,192;
> E=e1,e2,e3,e4,e5,e6,e7,e8,e9,e10,e11,e12,e13,e14,e15;
> intvec F11=1,4;
> intvec F12=2,7,12;
> intvec F13=3,9,11,13;
> intvec F21=1,6,10,14;
> intvec F22=2,8,15;
> intvec F23=3,5;
> list F1=F11,F12,F13;
> list F2=F21,F22,F23;
> list Fl=F1,F2;
> matrix in[1][15]=3,3,3,1,1,1,1,1,1,2,2,2,2,2,2;
> FBRNC(2,E,Fl,in);
0
1/2097152
> FBRNC(3,E,Fl,in);
4096/14348907
4096000/10460353203
> FBRNC(4,E,Fl,in);
531441/134217728
19098395217/4398046511104
> FBRNC(8,E,Fl,in);
373714754427/4398046511104
789875546216766625/9223372036854775808
> FBRNC(9,E,Fl,in);
23570780520448/205891132094649
125928737986859499 52/109418989131512359209
> FBRNC(16,E,Fl,in);
44502993896484375/144115188075855872
597809192207437133789062 5/193428131138340667952988 16
> FBRNC(32,E,Fl,in);
2658361895286355443375/4722366482869645213696
228373848705529056137472581587 21/405648192073033408478945 02
572032
> FBRNC(64,E,Fl,in);
1164586294937127613635765 71/154742504910672534362390528
640245649742200191197165929182925775 37/850705917302346158 65
843651857942052864
```

Finalmente, para acabar esta sección, hemos implementado también el cálculo de P_{Balli}, que podemos encontrar en la página 51 de la Sección 2.3, mediante el procedimiento *BalliRNC*. Los parámetros de entrada son el q, pues queremos la cota sobre $\mathbb{F}_q$, el vector de vértices V, que contiene los

vértices de la red, la lista de aristas E y la lista con el sistema de flujo $\mathcal{F}$.

```
proc BalliRNC(number q, intvec V, list E,list Fl){
number R=size(Fl);
number VI=size(V)-R;
number Balli=(1-R/(q-1))^int(VI);
if(Balli<0){
return(0);}
return(Balli);
}
```

Hemos hecho uso de este procedimiento para calcular la cota P_{Balli} de la Tabla 2.6 (página 53), relativas a la Figura 2.3.

```
> ring R=0,x,lp;
> list E;
> intvec e1=0,1;
> intvec e2=0,2;
> intvec e3=0,3;
> intvec e4=1,191;
> intvec e5=3,192;
> intvec e6=1,4;
> intvec e7=2,4;
> intvec e8=2,5;
> intvec e9=3,5;
> intvec e10=4,6;
> intvec e11=5,6;
> intvec e12=4,191;
> intvec e13=6,191;
> intvec e14=6,192;
> intvec e15=5,192;
> E=e1,e2,e3,e4,e5,e6,e7,e8,e9,e10,e11,e12,e13,e14,e15;
> intvec F11=1,4;
> intvec F12=2,7,12;
> intvec F13=3,9,11,13;
> intvec F21=1,6,10,14;
> intvec F22=2,8,15;
> intvec F23=3,5;
> list F1=F11,F12,F13;
> list F2=F21,F22,F23;
> list Fl=F1,F2;
> intvec V=0,1,2,3,4,5,6,191,192;
> BalliRNC(2,V,E,Fl);
0
```

```
> BalliRNC(3,V,E,Fl);
0
> BalliRNC(4,V,E,Fl);
1/2187
> BalliRNC(8,V,E,Fl);
78125/823543
> BalliRNC(16,V,E,Fl);
62748517/170859375
> BalliRNC(32,V,E,Fl);
17249876309/27512614111
> BalliRNC(64,V,E,Fl);
3142742836021/3938980639167
```

4.4. Procedimiento Codificación

En esta sección vemos el procedimiento que nos permite construir el espacio V a partir del polinomio $f(x) = \sum_{i=0}^{k-1} u_i x^{[i]}$, es decir, el procedimiento de codificación de los códigos estudiados en el Capítulo 3, donde el mensaje que queremos transmitir son los coeficientes del polinomio $f(x)$.

Para ello hemos creado el procedimiento *codiRNC*. Los parámetros de entrada son q, correspondiente al cuerpo $\mathbb{F}_{q^m}$ sobre el que deseamos trabajar, la lista de elementos independientes $A = \{\alpha_1, \ldots, \alpha_l\}$ y la lista de símbolos del mensaje $u = (u_0, \ldots, u_{k-1})$. El procedimiento nos devuelve la lista $B = \{\beta_1, \ldots, \beta_l\}$, tal que $\{(\alpha_1, \beta_1), \ldots, (\alpha_l, \beta_l)\}$ genera el espacio V.

```
proc codiRNC (int q, list A, list U)
{
int Usize=size(U);
int Asize=size(A);
poly f=0;
list B;
for (int i = 1; i <= Usize; i++)
{
f=f+U[i]*x^(q^(i-1));
}
for (int j = 1; j <= Asize; j++)
{
B[j]=subst(f,x,A[j]);
}
return(B);
}
```

A continuación, vemos la utilización de este procedimiento para el cálculo correspondiente del Ejemplo 3.6 (página 85).

```
> ring R=(2,a),x,lp;
> minpoly=a5+a2+1;
> int q=2;
> list A=(1,a+1,a3,a4+a);
> list U=(a,a2+a,a3+a2+1);
> codiRNC(q,A,U);
[1]:
   (a3+1)
[2]:
   (a4+a3+a+1)
```

```
[3]:
   (a2+1)
[4]:
   (a)
```

4.5. Procedimiento Decodificación

En esta sección veremos como decodificar un código de los estudiados en el Capítulo 3. Para ello hemos creado dos procedimientos que nos permiten dar un algoritmo de decodificación para este tipo de códigos.

El primer procedimiento creado para la decodificación es el procedimiento *InterpolRNC*, que nos permite construir el polinomio linealizado $Q(x,y)$ a través de los generadores del espacio U recibido en la transmisión.

Los parámetros de entrada son q, m, correspondientes al cuerpo $\mathbb{F}_{q^m}$ sobre el que deseamos trabajar, k que lo necesitamos para calcular los grados con peso $(1,k-1)$ de los polinomios $f(x,y)$, y las listas alpha y beta, correspondientes a los (α_i,β_i) que generan el espacio U. El procedimiento devolverá el polinomio linealizado en dos variables $Q(x,y)$.

Además, en este procedimiento hacemos uso de la libreria de Singular *crypto.lib*. En concreto haremos uso del procedimiento *babyGiant*, que nos permite calcular el logaritmo discreto. Necesitamos esta función, pues los polinomios con los que estamos trabajando son de la forma x^{q^i}, y para la definición de grado con la que estamos trabajando, su grado es i, luego necesitamos calcular el logaritmo en base q del grado, definido de forma habitual, de x^{q^i}. Además, necesitamos separar casos, pues el procedimiento *babyGiant* nos dice que tanto el polinomio x^{q^0} como una constante $c \in \mathbb{F}_{q^m}$ tienen grado 0.

```
proc interpolRNC (int q, int m, int k, list alpha, list beta)
{
LIB "crypto.lib";
 poly f0=x;
 poly f1=y;
 poly D0;
 poly D1;
 matrix f0m;
 matrix f1m;
 bigint degxf0;
 bigint degyf0;
 bigint degxf1;
 bigint degyf1;
 bigint degf0;
 bigint degf1;
for (int i = 1; i <= size(alpha); i++)
{
D0=subst(f0,x,alpha[i],y,beta[i]);
D1=subst(f1,x,alpha[i],y,beta[i]);
```

```
if (D0==0)
{
  f1=f1^q-f1*D1^(q-1);
}
if (D1==0)
{
  f0=f0^q-f0*D0^(q-1);
}
if ((D0*D1)!=0)
{
  f0m=coef(f0,x);
  f1m=coef(f1,x);
  degxf0=babyGiant(q,deg(f0m[1,1]),30000);
  degyf0=babyGiant(q,deg(f0m[nrows(f0m),ncols(f0m)]),30000)+k-1;
  degxf1=babyGiant(q,deg(f1m[1,1]),30000);
  degyf1=babyGiant(q,deg(f1m[nrows(f1m),ncols(f1m)]),30000)+k-1;
if((deg(f0m[1,1])*deg(f0m[nrows(f0m),ncols(f0m)])*
    deg(f1m[1,1])*deg(f1m[nrows(f1m),ncols(f1m)]))==0){
 if((deg(f0m[1,1])*deg(f0m[nrows(f0m),ncols(f0m)]))==0){
  if(deg(f0m[1,1])==0){degf0=degyf0;}
  else{degf0=degxf0;}
 }
 if((deg(f1m[1,1])*deg(f1m[nrows(f1m),ncols(f1m)]))==0){
  if(deg(f1m[1,1])==0){degf1=degyf1;}
  else{degf1=degxf1;}
 }
}
else{             degf0=max(degxf0,degyf0);
                  degf1=max(degxf1,degyf1);}
  if (degf0<=degf1)
  {
     f1=D1*f0-D0*f1;
     f0=f0^q-f0*D0^(q-1);
  }
  else
  {
     f0=D1*f0-D0*f1;
     f1=f1^q-f1*D1^(q-1);
  }
}
}
  f0m=coef(f0,x);
```

```
  f1m=coef(f1,x);
  degxf0=babyGiant(q,deg(f0m[1,1]),30000);
  degyf0=babyGiant(q,deg(f0m[nrows(f0m),ncols(f0m)]),30000)+k-1;
  degxf1=babyGiant(q,deg(f1m[1,1]),30000);
  degyf1=babyGiant(q,deg(f1m[nrows(f1m),ncols(f1m)]),30000)+k-1;
if((deg(f0m[1,1])*deg(f0m[nrows(f0m),ncols(f0m)])*
    deg(f1m[1,1])*deg(f1m[nrows(f1m),ncols(f1m)]))==0){
 if((deg(f0m[1,1])*deg(f0m[nrows(f0m),ncols(f0m)]))==0){
  if(deg(f0m[1,1])==0){degf0=degyf0;}
  else{degf0=degxf0;}
 }
 if((deg(f1m[1,1])*deg(f1m[nrows(f1m),ncols(f1m)]))==0){
  if(deg(f1m[1,1])==0){degf1=degyf1;}
  else{degf1=degxf1;}
 }
}
else{             degf0=max(degxf0,degyf0);
                  degf1=max(degxf1,degyf1);}
if (deg(f1)<deg(f0))
  {
     return(f1);
  }
else
  {
     return(f0);
  }
}
```

A continuación, vemos también la utilización de este procedimiento para los dos cálculos correspondientes del Ejemplo 3.6 (página 85).

```
> ring R=(2,a),(x,y),lp;
> minpoly=a5+a2+1;
> int q=2;
> int m=5;
> int k=3;
> list alpha=(a3+1,a+1,a4+a3+a,a4+a+1);
> list beta=(a3+a2,1,a2+a+1,a3+a+1);
> interpolRNC(q,m,k,alpha,beta);
(a3+a2+a+1)*x8+(a4+a3)*x4+(a4+a3+a)*x2+(a4+a)*y2
> list alpha2=(1,a3+1,a4+a3+a+1);
> list beta2=(a3+1,a3+a2,a3+a2+a);
> interpolRNC(q,m,k,alpha2,beta2);
(a2+a)*x4+(a4+a)*x2+(a3+a2+a)*x+(a2+a+1)*y
```

Para concluir la decodificación del código, hemos creado el procedimiento *RDivRNC*, que nos permite, una vez obtenido el polinomio linealizado $Q(x,y) = Q_x(x) + Q_y(y)$, calcular el polinomio $f'(x)$ tal que $-Q_x(x) \equiv Q_y(x) \otimes f'(x)$. De esta forma, podemos obtener el mensaje transmitido como los coeficientes del polinomio $f'(x)$ obtenido.

Los parámetros de entrada son q, m, correspondientes al cuerpo $\mathbb{F}_{q^m}$ sobre el que deseamos trabajar, y los polinomios $a(x)$, $b(x)$, que se corresponden con los polinomios $Q_x(x)$, $Q_y(x)$ sobre los que queremos efectuar la división. El programa nos devuelve el cociente $q_R(x)$ y el resto $r_R(x)$ de la división. Si el resto es 0, identificamos el polinomio $q_R(x)$ con el polinomio $f'(x)$, tal que $-Q_x(x) \equiv Q_y(x) \otimes f'(x)$.

Este algoritmo vuelve a hacer uso del procedimiento *babyGiant*, de la libreria *crypto.lib*, que nos permite calcular el logaritmo discreto. Necesitamos este procedimiento, pues los polinomios con los que estamos trabajando son de la forma x^{q^i}, y para la definición de grado con la que estamos trabajando, su grado es i, luego necesitamos calcular el logaritmo en base q del grado, definido de forma habitual, de x^{q^i}.

```
proc RDivRNC (int q, int m, poly aX, poly bX)
{
LIB "crypto.lib";
aX=-aX;
poly qR=0;
poly rR=0;
bigint e=babyGiant(q,deg(bX),30000);
poly be=leadcoef(bX);
bigint d;
poly ad;
poly t;
while (deg(aX)>=deg(bX))
{
d=babyGiant(q,deg(aX),30000);
ad=leadcoef(aX);
t=(ad/be)^(q^(m-int(e)))*x^(q^(int(d)-int(e)));
qR=qR+t;
aX=aX-subst(bX,x,t);
}
rR=aX;
list L=(qR,rR);
return(L);
}
```

Finalmente, vemos la utilización de este procedimiento para los dos cálculos correspondientes del Ejemplo 3.6 (página 85).

```
> ring R=(2,a),x,lp;
> minpoly=a5+a2+1;
> int q=2;
> int m=5;
> poly aX=(a3+a2+a+1)*x8+(a4+a3)*x4+(a4+a3+a)*x2;
> poly bX=(a4+a)*x2;
> RDivRNC(q,m,aX,bX);
[1]:
   (a3+a2+a)*x4+(a2+a+1)*x2+(a2+1)*x
[2]:
   0
> poly aX2=(a2+a)*x^4+(a4+a)*x2+(a3+a2+a)*x;
> poly bX2=(a2+a+1)*x;
> RDivRNC(q,m,aX2,bX2);
[1]:
   (a3+a2+1)*x4+(a2+a)*x2+(a)*x
[2]:
   0
```

Bibliografía

[1] R. Ahlswede, N. Cai, S.-Y.R. Li, R.W.-H. Yeung. *Network Information Flow.* IEEE Transactions on Information Theory (Volume: 46, 2000), pp. 1204 - 11216.

[2] H. Balli, X. Yan, Z. Zhang. *On Randomized Linear Network Codes and Their Error Correction Capabilities.* IEEE Transactions on Information Theory (Volume: 55, 2009), pp. 3148 - 3160.

[3] E.R. Berlekamp. *Algebraic Coding Theory.* New York: McGraw-Hill, 1968.

[4] E.R. Berlekamp. *The technology of error-correcting codes.* Proc. IEEE (Volume: 68, May 1980), pp. 564 - 593.

[5] A.E. Brouwer, A.M. Cohen, A. Neumaier. *Distance-Regular Graphs.* New York: Springer-Verlag, 1989.

[6] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein. *Introduction to Algorithms.* MIT Press and McGraw-Hill, 2nd ed. (2001).

[7] D. Cox, J. Little, D. O'Shea. *Ideals, Varieties, and Algorithms.* 2nd ed. Berlin, Germany: Springer-Verlag, 1997.

[8] O. Geil, T. Høholdt. *Footprints or Generalized Bezout's Theorem.* IEEE Transactions on Information Theory (Volume: 46, March 2000), pp. 635 - 641.

[9] O. Geil, C. Thomsen. *Aspects of Random Network Coding.* In E. M. Moro (Ed.), Algebraic geometry modeling in information theory, pp. 47 - 81. World Scientific. Series on Coding Theory and Cryptology (Volume: 8, 2013).

[10] T. Ho, M. Médard, R. Kötter, D.R. Karger, M. Effros, J. Shi, B. Leong. *A Random Linear Network Coding Approach to Multicast.* IEEE Transactions on Information Theory (Volume: 52, 2006), pp. 4413 - 4430.

[11] S. Jaggi, P. Sanders, P.A. Chou, M. Effros, S. Egner, K. Jain, L. Tolhuizen. *Polynomial Time Algorithms for Multicast Network Code Construction*. IEEE Transactions on Information Theory (Volume: 51, June 2005), pp. 1973 - 1982.

[12] R. Kötter, F.R. Kschischang *Coding for Errors and Erasures in Random Network Coding*. IEEE Transactions on Information Theory (Volume: 54, 2008), pp. 3579 - 3591.

[13] R. Kötter, M. Médard. *An Algebraic Approach to Network Coding*. IEEE/ACM Transactions on Information Theory (Volume: 11, 2003), pp. 782 - 795.

[14] R.J. McEliece. *The Guruswami-Sudan decoding algorithm for Reed-Solomon codes*. Jet Propulsion Lab., Pasadena, CA, Interplanetary Network Progr.Rep. 42 - 153, 2003.

[15] Shuo-Yen Robert Li, Raymond W. Yeung, and Ning Cai. *Linear network coding*. IEEE Trans. Inform. Theory (Volume: 49, 2003), pp. 371 - 381.

[16] R. Lidl, H. Niederreiter. *Finite Fields*. The Encyclopedia of mathematics, G.-C. Rota, Ed. Reading, MA:Addison-Wesley, 1983, vol. 20.

[17] J.H. van Lint, R.M. Wilson. *A Course in Combinatorics*. 2nd edition. Cambridge, U.K.: Cambridge University Press, 2001.

[18] R. Motwani, P. Raghavan. *Randomized Algorithms*. Cambridge, U.K: Cambridge University Press, 1995.

[19] R. Pellikaan, X. Wu, S. Bulygin, R. Jurrius. *Codes, Cryptology and Curves with Computer Algebra*. Cambridge: Cambridge University Press (2017).

[20] R.M. Roth. *Introduction to Coding Theory*. Cambridge, U.K.: Cambridge Univ. Press, 2006.

[21] C. Thomsen. *Random Linear Network Coding, Zeros of Multivariate polynomials and affine variety codes*. Department of Mathematical Sciences, Aalborg University (June 2011).

[22] R.W.-H. Yeung, Z. Zhen. *Distributed Source Coding for Satellite Communications*. IEEE Transactions on Information Theory (Volume: 45, 1999), pp. 1111 - 1120.